

Estimation and control for MAV navigation in GPS-denied cluttered environments

Marzat, Julien; Croon, Guido de; Fraundorfer, Friedrich; Morin, Pascal; Tsourdos, Antonios

DOI

[10.1177/1756829318772901](https://doi.org/10.1177/1756829318772901)

Publication date

2018

Document Version

Final published version

Published in

International Journal of Micro Air Vehicles

Citation (APA)

Marzat, J. (Guest ed.), Croon, G. D. (Guest ed.), Fraundorfer, F. (Guest ed.), Morin, P. (Guest ed.), & Tsourdos, A. (Guest ed.) (2018). Estimation and control for MAV navigation in GPS-denied cluttered environments. *International Journal of Micro Air Vehicles*, 10(2), 125-239.
<https://doi.org/10.1177/1756829318772901>

Important note

To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

Contents

Special Issue: Estimation and control for MAV navigation in GPS-denied cluttered environments

Guest Editors: Julien Marzat, Guido de Croon, Friedrich Fraundorfer, Pascal Morin, Antonios Tsourdos

Editorial

Editorial for special collection on the estimation and control of MAV navigation in GPS-denied cluttered environments 125

J Marzat, G de Croon, F Fraundorfer, P Morin and A Tsourdos

Articles

Micro air vehicle local pose estimation with a two-dimensional laser scanner: A case study for electric tower inspection 127

C Viña and P Morin

Vision-based dynamic target trajectory and ego-motion estimation using incremental light bundle adjustment 157

M Chojnacki and V Indelman

Deep learning for vision-based micro aerial vehicle autonomous landing 171

L Yu, C Luo, X Yu, X Jiang, E Yang, C Luo and P Ren

Persistent self-supervised learning: From stereo to monocular vision for obstacle avoidance 186

K van Hecke, G de Croon, L van der Maaten, D Hennes and D Izzo

Confined spaces industrial inspection with micro aerial vehicles and laser range finder localization 207

P Tripicchio, M Satler, M Unetti and CA Avizzano

Collaborative multiple micro air vehicles' localization and target tracking in GPS-denied environment from range-velocity measurements 225

I Sarras, J Marzat, S Bertrand and H Piet-Lahanier

Editorial for special collection on the estimation and control of MAV navigation in GPS-denied cluttered environments

**Julien Marzat¹, Guido de Croon², Friedrich Fraundorfer³,
Pascal Morin⁴ and Antonios Tsourdos⁵**

New types of missions are being addressed by micro air vehicles (MAVs) in GPS-denied environments, which can be either indoor buildings or plants or outdoor facilities such as electrical substations or forests. These places can be highly uncertain with no previous mapping available and with little prior information, as well as highly cluttered and possibly containing dynamical objects.

Progress in technology and automation has made it possible to embed cameras (monocular, stereo or more) or laser scanners as main sensors on MAVs, which can be associated in a sensor fusion scheme with an inertial measurement unit and – depending on payload mass allowed – small-scale sonar or depth sensors.

However, safe navigation for autonomous surveillance or inspection missions in this type of challenging environment still requires the development of new sensor-based estimation and control algorithms that can be embedded on multi-rotor or flapping-wing MAVs with limited on-board computational capabilities.

This special issue covers several aspects of the research effort on this topic, ranging from localization issue using a limited number of sensors to control or learning-based approaches for achieving specific tasks.

Vina and Morin¹ present a methodology to obtain complete 3D local pose estimates in electric tower inspection tasks (where GPS localization is disturbed) with MAVs, using an on-board sensor setup consisting of a 2D LiDAR, a barometer sensor and an inertial measurement unit (IMU).

Chojnacki and Indelman² present a vision-based method using a light bundle adjustment procedure for simultaneous robot motion estimation and dynamic target tracking, while operating in GPS-denied unknown or uncertain environments.

Yu et al.³ propose an end-to-end landmark detection system based on a deep convolutional neural network

and an associated embedded implementation on a graphics implementation processing unit to perform vision-based autonomous landing.

In van Hecke et al.,⁴ a self-supervised learning strategy is proposed for the safe navigation among obstacles of a flying robot using very light embedded vision sensors. The proposed learning mechanism relies on distance estimates provided by stereo vision and then learns how to perform this estimation using only monocular information.

Trivicchio et al.⁵ address the problem of semi-automatic navigation in confined environments using laser-based localization, with application to the inspection of an industrial combustion chamber with poor lighting conditions, in the presence of magnetic and communication disturbances, iron dust and repetitive patterns on the structure walls.

Sarras et al.⁶ treat the problem of simultaneous collaborative localization and control for a fleet of MAVs tracking a common target using only range and velocity measurements. The proposed solution combines local filters for each agent and cooperative filters to estimate all positions, which are then used in a dynamic consensus control law to track the target without any

¹DTIS, ONERA, Université Paris-Saclay, Palaiseau, France

²Micro Air Vehicle Laboratory, Faculty of Aerospace Engineering, Delft University of Technology, Delft, The Netherlands

³Institute of Computer Graphics and Vision, TU Graz, Austria

⁴Institut des Systèmes Intelligents et de Robotique (ISIR), Sorbonne Universités, Paris, France

⁵School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield, England, United Kingdom

The first author is an Invited Lead Editor; other authors are Guest Editors.

Corresponding author:

Guido de Croon, Delft University of Technology, Delft, the Netherlands.
Email: g.c.h.e.decroon@tudelft.nl



external reference which makes it applicable in GPS-denied environments.

In summary, these papers report a number of contributions on sensor integration, signal processing and control algorithms associated to validations based on simulations and experimental data, which should pave the way to future developments and widespread use of MAV technology in future applicative scenarios involving indoor and cluttered environments.

References

1. Vina C and Morin P. MAV local pose estimation with a 2D laser scanner: a case study for electric tower inspection.
2. Chojnacki M and Indelman V. Vision-based dynamic target trajectory and ego-motion estimation using incremental light bundle adjustment.
3. Yu L, Luo C, Yu X, et al. Deep learning for vision based MAV autonomous landing.
4. van Hecke K, de Croon G, van der Maaten L, et al. Persistent self-supervised learning: from stereo to monocular vision for obstacle avoidance.
5. Tripicchio P, Satler M, Unetti M, et al. Confined spaces industrial inspection with micro aerial vehicles and laser range finder localization.
6. Sarras I, Marzat J, Bertrand S, et al. Collaborative multi-MAV localization and target tracking in GPS-denied environment from range-velocity measurements.

Micro air vehicle local pose estimation with a two-dimensional laser scanner: A case study for electric tower inspection

International Journal of Micro Air Vehicles
2018, Vol. 10(2) 127–156
© The Author(s) 2017
DOI: 10.1177/1756829317745316
journals.sagepub.com/home/mav


Carlos Viña and Pascal Morin

Abstract

Automation of inspection tasks is crucial for the development of the power industry, where micro air vehicles have shown a great potential. Self-localization in this context remains a key issue and is the main subject of this work. This article presents a methodology to obtain complete three-dimensional local pose estimates in electric tower inspection tasks with micro air vehicles, using an on-board sensor set-up consisting of a two-dimensional light detection and ranging, a barometer sensor and an inertial measurement unit. First, we present a method to track the tower's cross-sections in the laser scans and give insights on how this can be used to model electric towers. Then, we show how the popular iterative closest point algorithm, that is typically limited to indoor navigation, can be adapted to this scenario and propose two different implementations to retrieve pose information. This is complemented with attitude estimates from the inertial measurement unit measurements, based on a gain-scheduled non-linear observer formulation. An altitude observer to compensate for barometer drift is also presented. Finally, we address velocity estimation with views to feedback position control. Validations based on simulations and experimental data are presented.

Keywords

Micro air vehicle, airborne laser scanning, two-dimensional light detection and ranging, barometer, inertial measurement unit, iterative closest point, state estimation

Received 18 May 2017; accepted 26 October 2017

Introduction

Power utilities, such as transmission line towers, are subject to deterioration due to the atmospheric conditions to which they are exposed. Ensuring their integrity and avoiding network downtime require extensive monitoring programmes. For this purpose, aerial surveys have been increasingly common as they allow covering vast areas in relatively short periods of time, by relying on remote sensing technologies such as thermal imaging, aerial imaging and optical satellites, among others.^{1,2} In particular, airborne laser scanning (ALS) technologies have recently attracted a large attention due to their capability of achieving high quality 3D models of infrastructure with high spatial resolution.^{2,3} In ALS applications, powerful 3D light detection and ranging (LiDAR) sensors are mounted on manned aircraft, such as helicopters,^{1,2,4} then data acquisition is typically carried out using a GPS sensor and an inertial measurement unit (IMU) to keep track of the aircraft's position and orientation. The geo-referenced range readings are processed afterwards

for a wide variety of classification or reconstruction tasks such as detecting power lines,^{4,5} vegetation management³ and making 3D models of the electric towers.⁶ Nonetheless, the high operational costs of piloted aircraft have constrained the proliferation of these applications. The automation of inspection tasks has thus become a key subject of research in the power industry, in which unmanned air vehicles (UAVs) have surfaced as an attractive solution, as they provide an affordable and flexible means of gathering spatial data.^{7–9} This has been mainly fuelled by developments in lithium polymer batteries that have led to larger flight durations and increased payload capabilities. However, these small platforms currently

Institut des Systèmes Intelligents et de Robotique (ISIR), Sorbonne Universités, Paris, France

Corresponding author:

Carlos Viña, Institut des Systèmes Intelligents et de Robotique (ISIR), Sorbonne Universités, CNRS UMR 7222, Paris, France.
Email: vina@isir.upmc.fr



Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (<http://www.creativecommons.org/licenses/by/4.0/>) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (<https://us.sagepub.com/en-us/nam/open-access-at-sage>).

cannot carry the heavy LiDARs required in most ALS applications, and research on inspection tasks with UAVs has mainly focused on vision-based approaches instead.^{1,8,10,11} Rapid advances in lightweight LiDARs have made them an appealing alternative for UAVs, and while performance and precision remain far from their 3D counterparts, they can be used for basic and affordable ALS applications, which has already been demonstrated in previous works, for example, for power line monitoring.¹²

In the context of power utility inspection, GPS sensors remain the predominant choice for achieving autonomous flight capabilities with UAVs.⁷ Nonetheless, a GPS signal is not always accurate, can be perturbed by the strong electromagnetic fields in the proximity of the power lines¹³ and provides no perception of the surrounding environment. As a result, a safe collision-free flight cannot be achieved relying on GPS measurements uniquely, which is instead limited to waypoint navigation at large distances from the inspected objects.^{1,7,9} On the one hand, vision-based navigation systems have been proposed as a substitute in numerous works, relying mostly on tracking and following the power lines.^{10,11} On the other hand, lightweight LiDARs can also be employed for autonomous navigation purposes and have been successfully used for indoor flights with micro air vehicles (MAVs).^{14–18} These sensors excel when navigating in cluttered environments, as they directly measure the distance to surrounding objects and naturally open the way for sense-and-avoid functionalities required for safe flights. As a consequence, they can allow achieving higher levels of autonomy and close-up inspections in power line corridors, which is hard to accomplish with other sensors. In this work, we focus on the inspection of transmission line towers, and we explore how 2D LiDARs coupled with commonly available sensors can be used for pose estimation purposes in these scenarios.

Problem statement

One of the first tasks that any autonomous platform must achieve is self-localization. Thus, our primary goal is to obtain real-time estimates of a MAV's six degree of freedom (DoF) pose with respect to an electric tower, using uniquely on-board sensors and processing capabilities. Our main interest is steel lattice towers made up of rectangular cross-sections commonly used to support high-voltage transmission lines, such as the one shown in Figure 1. For this first case study, we focus on the tower's body, which makes up the largest portion of the structure. The tower heads have a more complex structure that requires an extensive parameterization^{6,19} and are not considered in this work.

After treating the self-localization problem, the last part of this study focuses on obtaining velocity



Figure 1. A common high voltage transmission line.

estimates and sensor fusion techniques are used for this purpose. Accurate velocity estimates are necessary in the control loop to successfully stabilize a MAV's position. Feedback position control, however, is not addressed in this study. The long-term aim of this work is to achieve autonomous inspection capabilities of electric towers with MAVs.

Related works

While laser range finders have been largely popular among ground robots for autonomous navigation tasks, aerial robots present additional complications that make similar applications not so straightforward. First, flying robots' motion is 3D. Then, payload limitations prevent the use of more powerful sensors such as 3D laser range finders. Finally, flying robots have fast dynamics that make them harder to stabilize and any state estimation has to be made with low delays and an adequate level of accuracy. This means that estimation and control algorithms must be preferably implemented on board, and must run at high speeds, which limits the complexity of the algorithms that can be used, so as to avoid significant processing delays. Nonetheless, fully autonomous capabilities for MAVs equipped with 2D LiDARs have been shown in numerous previous works,^{14–18,20–23} which have mainly focused on indoors scenarios. Most of these studies adopt a similar strategy: the first goal is to obtain fast and accurate 3D pose estimates from the embarked sensors' measurements, preferably using on-board processing capabilities; then, a second goal is to derive estimates of the linear velocities using the pose estimates and sensor fusion techniques. We now present how several previous works have addressed these two tasks.

Laser-based local pose estimation on-board MAVs

Regarding the first goal, this is partly achieved by aligning pairs of laser scans to recover the MAV's

relative displacement, a technique known as *scan matching* or *scan registration*. While these algorithms can pose a heavy computational burden, satisfying real-time results have been obtained from adaptations of well-known techniques, such as the iterative closest point (ICP) algorithm,^{15,17,18,24} and the correlative scan-matching algorithm.^{14,16,25} Following the satisfying results previously obtained on-board MAVs, and due to its simplicity and efficiency, the ICP algorithm was chosen for the scan registrations.

Typical approach with the ICP algorithm on-board MAVs. A classic implementation of the ICP algorithm in navigation tasks consists in aligning the current laser scan to the preceding scan. This is known as *incremental scan matching* and is known to lead to drift over time.^{15,17,18} An alternative is to use a *keyframe* approach,¹⁷ with a reference scan instead fixed at some initial time. As long as the robot remains in the proximity of this keyframe, and as long as there is sufficient overlap, the estimation error remains bounded and the results are drift free. The ICP implementations proposed in this article go along this line of work.

In general, on MAVs equipped with 2D LiDARs, the ICP algorithm is limited to aligning pairs of 2D laser scans to recover 2D pose estimates. The remaining states are estimated from separate sensing (e.g. IMU for attitude estimation¹⁴⁻¹⁸ and laser altimeter for altitude estimation^{14,15}). However, to align pairs of 2D laser scans the measurements must be taken within the same plane. This poses a major drawback for aerial robots and requires coping with the 3D motion. A simple solution is to project the laser points to a common horizontal plane using attitude estimations from IMUs.^{14,15,18,17} Then, the projected scans are aligned with the ICP algorithm. Nonetheless, this has the underlying assumption that surrounding objects are planar and height invariant, which holds for common indoor scenarios, with mainly straight walls. In an inspection scene, this assumption does not hold as the electric towers have a geometry that varies greatly in 3D. Hence, in our scenario aligning pairs of 2D scans in similar way is not possible. In this work we explore alternative ways in which pose information can be recovered from the laser scans, by exploiting basic knowledge of the tower's geometry. We also explore two different ways in which the ICP algorithm can be extended to electric tower case.

Limitations of the ICP algorithm for self-localization. It is important to note that scan-matching techniques, such as ICP, only guarantee local convergence and depend highly on a good initial guess.^{24,26} A bad initialization may lead ICP to converge to a local

minimum far from the optimal solution. Furthermore, these techniques typically cannot recover from large estimation errors. Globally optimal solutions for the ICP algorithm have been studied in the past²⁷ but are typically too slow for state estimation purposes. In literature, to overcome these issues, it is common for simultaneous localization and mapping (SLAM) techniques to be used in parallel.^{14-17,23} These algorithms provide pose estimates with guaranteed global consistency that are less sensitive to initialization errors and that can allow detecting and correcting errors from scan matching. The faster local pose estimates are still required as an odometric input to SLAM, to initialize and speed up the mapping process.^{14,17} However, SLAM remains very computationally expensive and is commonly performed off-board,^{14,16} with only a handful of studies achieving on-board capabilities,^{15,17} at very low rates (2–10 Hz). Thus, the global pose estimates are seldom included directly in the control loop and are mainly limited to providing periodic corrections to the real-time pose estimates from scan matching¹⁴ and to perform higher level tasks such as path planning^{16,17} and obstacle avoidance.¹⁶ For the purposes of this article, we focus only on the local pose estimation problem, keeping in mind that mapping methods can be used in parallel.

Another complex issue is that scan-matching performance has a strong dependence on the shape of the surrounding environment, as the laser scans must capture sufficient geometric detail in order to extract any useful pose information. The algorithm will thus fail under highly unstructured scenarios, often faced outdoors, or featureless scenarios, such as long hallways or circular rooms. This, in reality, corresponds to inherent limitations of laser range sensing.¹⁴ Previous works have addressed this issue incorporating multiple sensing modalities, such as GPS sensors, ultrasonic sensors and cameras.^{21,22} This, however, goes beyond the scope of this work.

Altitude estimation on-board MAVs

On the one hand, on MAVs equipped with 2D LiDARs, altitude is commonly estimated by placing mirrors to reflect multiple laser rays downwards and directly measuring the distance to the ground assuming that the ground elevation is piecewise constant for the most part.^{14,16,17} However, to account for potential discontinuities and changing floor elevations several solutions have been proposed, such as creating multilevel grid maps of the ground¹⁶ or creating histograms of the range measurements to detect edges and floor level changes.¹⁷ While this has proven to be effective when navigating indoors, performance remains highly

dependent on the floor's layout, which can be very irregular in typical outdoor inspection scenarios.

On the other hand, barometric sensors are also popular among commercial MAVs. These sensors estimate the absolute or relative height of an object by measuring the atmospheric pressure. However, fluctuations in pressure due to weather conditions cause these height measurements to drift over time. Sensor fusion techniques are thus used to estimate and compensate this drift by using additional sources such as GPS²⁸ and IMUs.^{29,30} More recently, differential barometry has been gaining popularity.^{31,32} In this configuration, a second barometer is set stationary on the ground and used as a reference measurement to track changes in local pressure, effectively reducing drift and increasing accuracy.

Attitude estimation on-board MAVs

Fast and accurate attitude estimates are an essential part of any MAV platform. Absolute attitude information can be recovered from magnetometers and accelerometers.^{33–35} On the one hand, magnetometers provide measurements of the surrounding magnetic field in the body-attached frame and allow deducing the MAV's heading.^{33,36} However, they are very sensitive to local magnetic fields and measurements can be noisy. On the other hand, accelerometers measure the so-called *specific acceleration*. When the linear acceleration is small, this sensor directly measures the gravity vector, thus acting as an inclinometer and providing direct observations of the roll and pitch angles. This is a common assumption applied in attitude estimation,^{33,35,37} which has shown to work well in practice. On the downside, accelerometers are highly sensitive to vibrations induced by the propellers and require significant filtering to be useful.³⁴ This in exchange can introduce important latencies in the estimations. Thus, complementary attitude information is commonly obtained from gyroscopes, which measure the angular velocity along the three rotational axes in the body-attached frame. These sensors are less sensitive to vibrations and are very reliable. Absolute attitude can be recovered for the three rotational axes by integrating the measured angular rates; however, this causes the estimation error to grow without bound.³⁴

Hence, sensor fusion techniques are used to combine the information from all three sensors to tackle drift and noise issues and to obtain more accurate attitude estimates. In literature, the use of linear stochastic filters, such as Kalman filters³⁴ or extended Kalman filters,^{38,39} as the means to fuse inertial measurements is very common. While these filters have been successful in certain applications, they can have an unpredictable behaviour when applied to non-linear systems.⁴⁰ An alternative is to use non-linear observer design techniques, which

present strong robustness properties and guaranteed exponential convergence.^{33,40} Numerous recent works have shown successful results in obtaining accurate attitude estimates from noisy and biased measurements using low-cost IMUs.^{40,41} In this work we adopt a non-linear observer formulation to obtain attitude estimates.

Velocity estimation on-board MAVs

Literature regarding MAV velocity estimation is very vast and is linked to the type of sensing used on-board. We focus on the approaches applied on MAVs equipped with 2D LiDARs. On one side, directly differentiating the position estimates is avoided as this provides noisy and inaccurate results.^{17,18} Instead, sensor fusion techniques are employed to achieve high-quality results by combining laser estimates and inertial measurements. Stochastic filters, such as EKF, are predominantly used for this purpose,^{14,15,20} while simpler complementary filters have also provided satisfying results.¹⁸ Other works focus on using a cascade of filters for further noise reduction. Dryanovski et al.¹⁷ first used an alpha–beta filter to obtain rough initial velocity estimates from the laser position estimates, which are then used as a correction in a Kalman filter which includes inertial measurements. Shen et al.¹⁵ proposed a cascade of two separate EKFs to achieve accurate results and high rates.

Technical background

Sensor set-up

One of the first design challenges with MAVs is choosing the right on-board sensor set-up, which is tailored to the specific task at hand. In this section we present our choice for the sensor set-up.

2D laser rangefinder. Since odometric sensors to measure raw displacements are not available for MAVs, alternative approaches have to be used. In this work we are interested in using laser range measurements from LiDARs for this purpose. However, due to payload limitations only 2D LiDARs can be used,^{14,16,17} and complete 3D pose estimates cannot be obtained from the laser range measurements alone. Thus, additional sensing has to be used together with sensor fusion techniques to provide reliable 3D pose estimates.

IMU. At the heart of MAV platforms one commonly finds IMUs comprised of a three-axis accelerometer, a three-axis rate gyroscope and a magnetometer.³³ In this work magnetometers are not used as they are highly sensitive to magnetic interference and are very unreliable in the proximity of the power lines. We thus only

rely on an accelerometer and a gyroscope for inertial measurements.

Altitude sensor. With respect to laser altimeters, barometers allow measuring height without any influence of the ground's layout and are thus more appropriate for outdoor navigation. We mainly focus on barometers as a source of altitude information. While recent works have obtained impressive results with differential barometry,^{31,32} the focus of this work is using on-board sensing only, and differential barometry was not considered.



Figure 2. Quadrotor developed at ISIR, equipped with a Hokuyo URG-30LX 2D LiDAR, an MPU6000 3 axis accelerometer/gyrometer unit and an SF10/A laser altimeter from Lightware Optoelectronics.

Experimental set-up

Several experiments were carried out with a quadrotor platform developed at our lab, shown in Figure 2. This MAV was equipped with a Hokuyo URG-30LX 2D laser scanner mounted horizontally on top and providing measurements at 40 Hz. This sensor was connected to an on-board Odroid-XU computer, where all the laser data acquisition was performed. A Quantec Quanton flight controller card based on an STM32 microcontroller was used to estimate the quadrotor's attitude from measurements obtained from an MPU6000 three-axis accelerometer/gyrometer unit. Lastly, at the time of the acquisitions, the MAV was equipped with an SF10/A laser altimeter from Lightware Optoelectronics, which provides readings at 20 Hz of the distance to the ground along the body-fixed vertical axis. This platform was used towards the beginning of this research to conduct several test flights in front of real electric towers (see Figure 3). The acquired data were then analysed and served as a basis to the methodology developed in this work. While our final results are mostly based on simulations, and focus on using barometer sensors for altitude estimation, interesting experimental results from these initial test flights will be presented where altitude information was obtained from the laser altimeter.

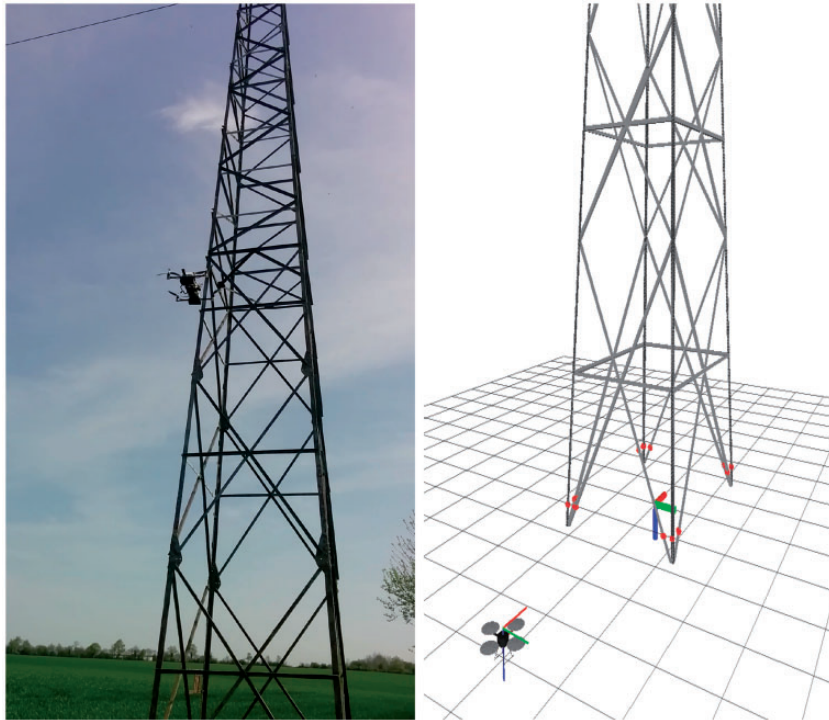


Figure 3. (a) Acquiring laser measurements on an electric tower from a 60 kV distribution line, with the quadrotor from Figure 2 and (b) the equivalent simulation set-up.

Simulation set-up

The approaches proposed in this work were validated in simulations using the Gazebo simulation environment⁴² and ROS as an interfacing middleware,⁴³ on a PC with an Intel 3.4 GHz Quad-Core processor and 8 GB of RAM. The Hector quadrotor stack from ROS⁴⁴ was used to simulate the quadrotor kinematics and dynamics. Regarding the sensors, the simulated IMU published gyrometer and accelerometer readings at 100 Hz, and the barometer sensor provided measurements at 20 Hz. The 2D laser scanner from Gazebo was set to match the characteristics of a Hokuyo URG-30LX sensor: 40 Hz scan frequency, 0.25° angular resolution and 270° field of view (thus 1080 measurements per scan). This sensor was mounted horizontally on top of the simulated quadrotor. A CAD model of an electric tower body was used, whose dimensions are 2.5 m × 3.5 m at the ground level and 1.5 m × 2 m at a height of 10 m. These dimensions roughly correspond to those of the tower from Figure 3(a). The complete simulation set-up is shown in Figure 3(b). All algorithm development was done using C++ and the registration and sample consensus modules from the open source Point Cloud Library.⁴⁵

Notation

Let us denote by \mathcal{I} an inertial North-East-Down frame located at the centre of the tower at the ground level. Let \mathcal{B} denote a body-attached frame in the MAV's centre of mass. For simplicity, we consider that this frame coincides with the sensor frames. Then, let $\xi = (x, y, z)^\top$ denote the position vector of \mathcal{B} with respect to \mathcal{I} (i.e. the position vector of the MAV's centre of mass) expressed in \mathcal{I} . Next, R denotes the rotation matrix from \mathcal{B} to \mathcal{I} . Using the Z–X–Y Euler angle convention with roll ϕ , pitch θ and yaw ψ angles, this rotation matrix is expressed as

$$R(\psi, \phi, \theta) = R_z(\psi)R_x(\phi)R_y(\theta) \\ = \begin{pmatrix} c\psi c\theta - s\psi s\phi s\theta & -c\psi s\theta & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\theta c\psi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{pmatrix} \quad (1)$$

With this notation, $T_{\mathbf{X}}$ denotes a rigid body transformation parameterized by a vector \mathbf{X} , such that

$$T_{\mathbf{X}}(\mathbf{p}) = R(\psi, \phi, \theta)\mathbf{p} + \xi, \quad \mathbf{p} \in \mathbb{R}^3 \quad (2)$$

for $\mathbf{X} = (x, y, z, \phi, \theta, \psi)$. This defines the six DoF rigid body transformation from \mathcal{B} to \mathcal{I} that transforms the coordinates of a point from \mathcal{B} to \mathcal{I} .

We now recall the basic translational dynamics of multirotor aircraft with respect to the inertial frame³³

$$\begin{cases} \dot{\xi} = \mathbf{v} \\ \dot{\mathbf{v}} = g\mathbf{e}_3 + \frac{\mathbf{F}}{m} \end{cases} \quad (3)$$

where $\mathbf{v} = (v_x, v_y, v_z)^\top$ denotes the linear velocity of \mathcal{B} with respect to \mathcal{I} expressed in \mathcal{I} ; g is the gravity constant; $\mathbf{e}_3 = (0, 0, 1)^\top$; m is the MAV's mass and $\mathbf{F} = (F_x, F_y, F_z)^\top$ is the coordinate vector of the aerodynamic forces acting on the MAV, expressed in \mathcal{I} . At zero air velocity, these forces are reduced to the thrust force generated by the propellers. Developing equation (3) one obtains

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{z} = v_z \\ \dot{v}_x = \frac{F_x}{m} \\ \dot{v}_y = \frac{F_y}{m} \\ \dot{v}_z = g + \frac{F_z}{m} \end{cases} \quad (4)$$

These equations will be used in our observer formulation to fuse the information from the multiple embarked sensors and to recover velocity estimates.

2D local pose estimation

In this section we focus on tracking the cross-sections captured by the individual 2D laser scans, which is analogous to determining the 2D pose of the MAV with respect to the electric tower. Specifically, we explore how basic geometric knowledge of the scene can be exploited for this purpose, without the help of additional sensing. As already mentioned, we focus on the body of electric towers made up of rectangular cross-sections. Measurements taken with a 2D LiDAR on the electric tower from Figure 3(a) are shown in Figure 4, where the portion of the tower can be easily identified. The large open spaces on the surface of the tower allow capturing measurements on all of the tower's faces (Figure 4(a)). However, due to occlusions, the entire cross-section is not always visible (Figure 4(b) to (d)) and very different scanned structures can be observed. In the worst-case scenario

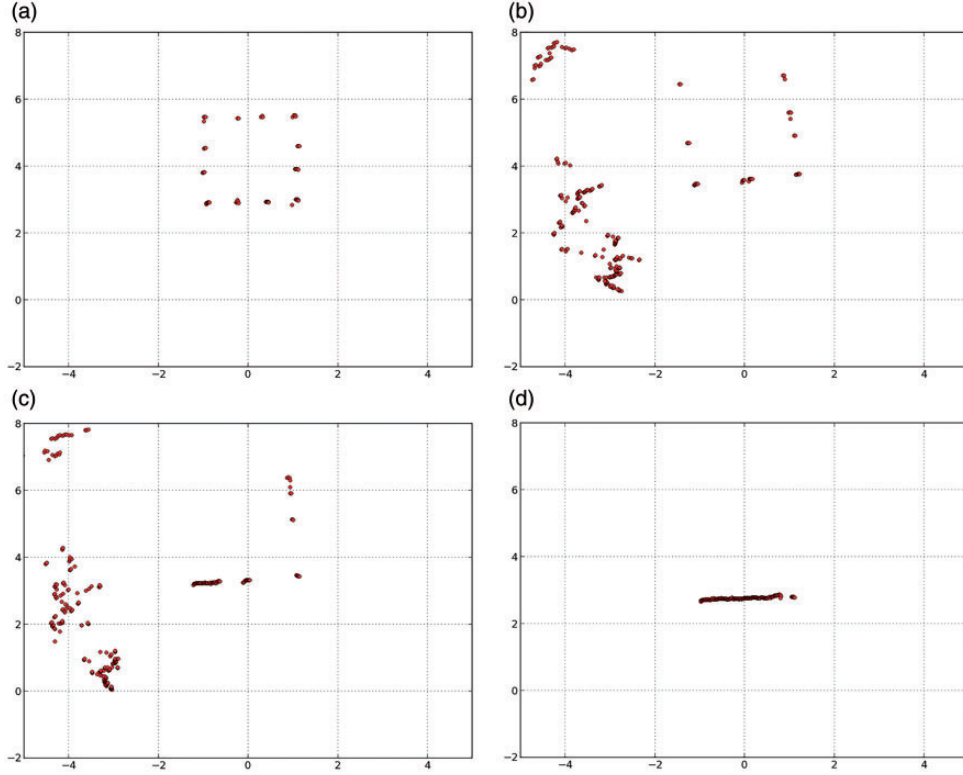


Figure 4. Laser range measurements acquired on the tower from Figure 3(a). In the best case, all sides are visible (a). Occlusions sometimes block the lateral and backsides from view (b)–(d). In the worst case, only the front side is visible (d). This happens when horizontal bars on the tower block the lateral and back sides from view.

(Figure 4(d)), horizontal bars that are part of the tower’s structure block the lateral and back sides from view and only the front side of the tower is captured in the scans.

Tracking the tower thus requires accounting for the different cases that can be faced. The idea is to gradually extract notable features from the laser scans, using basic geometric assumptions, to determine the position and orientation of the tower. The largest concentration of laser beams fall on the side closest to the MAV, and the line segment formed by these points is the most notable feature in the laser scans. This front line, denoted as $\mathcal{L}_{\text{front}}$, allows recovering essential position and orientation information. The coordinate vectors of the left and right corners of this front line segment, expressed in \mathcal{B} , are denoted as \mathbf{p}_{left} and $\mathbf{p}_{\text{right}}$, respectively. Since $\mathcal{L}_{\text{front}}$ remains visible even in the worst-case scenario (Figure 4(d)), tracking this line is at the heart of our proposed approach. Then, as the different sides become visible (Figure 4(a) to (c)), more features are available, such as the sidelines $\mathcal{L}_{\text{left}}$ and $\mathcal{L}_{\text{right}}$, which provide complementary orientation information and allow determining the depth (and hence the centre) of the cross-section. The back side of the tower is not explicitly

modelled, as it is seldom visible and provides unreliable information. Then, the shape of the contour captured by the laser beams allows establishing a connection between the different features. We consider that this contour is rectangular. However, for this assumption to hold, the scan plane must remain horizontal. This will be discussed in more detail at the end of the section.

We now present a parameterization of the laser scans based on our observations of Figure 4. Since the goal is to track the cross-sections directly in the laser scans, let $\{x_c, y_c, \psi_c\}$ denote the 2D pose of the cross-section’s centre with respect to the body frame \mathcal{B} . Then, $\mathcal{C} = \{\mathcal{O}_c, \vec{i}_c, \vec{j}_c\}$ denotes the centre-attached frame, $\xi_c = (x_c, y_c)^\top$ denotes the position vector of \mathcal{C} with respect to \mathcal{B} expressed in \mathcal{B} and ψ_c denotes the orientation of \mathcal{C} with respect to \mathcal{B} . For a completely horizontal scan plane, this frame is aligned with the inertial frame \mathcal{I} . A second frame $\mathcal{F} = \{\mathcal{O}_f, \vec{i}_f, \vec{j}_f\}$ is attached to the front side’s centre, with corresponding position vector ξ_f with respect to \mathcal{B} expressed in \mathcal{B} and similar orientation to \mathcal{C} . Next, the dimensions of the cross-section are the width and the depth, which are denoted as d_{width} and d_{depth} , respectively, which vary considerably with height due to the tower’s structure.

The goal is now to find the geometric model that best fits the extracted points. From the previous step, three different situations can arise. First, if no side was detected, the estimation process stops since no useful information is available. Second, if only the front side $\mathbf{S}_{\text{front}}$ was detected, the coefficients for $\mathcal{L}_{\text{front}}$ are directly provided by the RANSAC algorithm and the orientation can be estimated, but no depth information is available. Lastly, if the front side and at least one of the lateral sides was detected, then the rectangular shape of the cross-section can be taken into account to recalculate $\mathcal{L}_{\text{front}}$ which better fits the data, and to obtain $\mathcal{L}_{\text{left}}$ and $\mathcal{L}_{\text{right}}$. The following formulation applies to the case when both \mathbf{S}_{left} and $\mathbf{S}_{\text{right}}$ are detected, but the same procedure is valid when only one of the lateral sides is found. Since the lateral sides are perpendicular to $\mathcal{L}_{\text{front}}$, then, recalling the

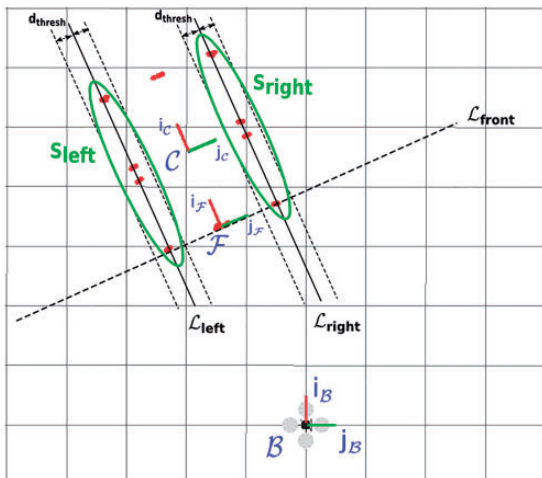


Figure 7. Detecting the left and right sides.

definition from equation (6), their normal vector is $(-n_y, n_x)$ and the cross-section is defined by

$$\begin{cases} \mathcal{L}_{\text{front}} : & c_F + n_x x_F + n_y y_F = 0, \\ \mathcal{L}_{\text{left}} : & c_L - n_y x_L + n_x y_L = 0, \\ \mathcal{L}_{\text{right}} : & c_R - n_y x_R + n_x y_R = 0, \\ & n_x^2 + n_y^2 = 1 \end{cases} \quad (7)$$

Then, evaluating the extracted point sets $\mathbf{S}_{\text{front}}$, \mathbf{S}_{left} and $\mathbf{S}_{\text{right}}$ from equation (5) with their respective line in equation (7), and expressing in matrix form, one obtains

$$\begin{pmatrix} 1 & 0 & 0 & x_{F,1} & y_{F,1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & x_{F,N_F} & y_{F,N_F} \\ 0 & 1 & 0 & y_{L,1} & -x_{L,1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & y_{L,N_L} & -x_{L,N_L} \\ 0 & 0 & 1 & y_{R,1} & -x_{R,1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & y_{R,N_R} & -x_{R,N_R} \end{pmatrix} \begin{pmatrix} c_F \\ c_L \\ c_R \\ n_x \\ n_y \end{pmatrix} = \mathbf{r} \quad (8)$$

where $\mathbf{r} = (r_1, \dots, r_N)^\top$, with $N = N_F + N_L + N_R$, are the residuals, and $|r_i|$ corresponds to the distance from a point to the line. The geometric fitting problem is formulated as finding the coefficients of equation (7) for which the sum of squared distances is minimal. That is

$$\begin{aligned} \min ||r||^2 = \min \sum_{i=1}^N r_i^2, \quad \text{subject to equation (8),} \\ \text{and } n_x^2 + n_y^2 = 1 \end{aligned} \quad (9)$$

which is a constrained least squares problem, with non-linear constraint $n_x^2 + n_y^2 = 1$ to guarantee solution uniqueness. This is solved numerically following the procedure proposed in Gander and Hrebicek.⁴⁷ The end result is an estimate of the parameters of equation (7). At this point, \mathbf{p}_{left} and $\mathbf{p}_{\text{right}}$ are recalculated from the line intersections, as they will be required in the following step.

Calculating the position and orientation

We first determine the position and orientation of the front frame \mathcal{F} . Recovering the orientation of

the tower results straightforward from the coefficients of $\mathcal{L}_{\text{front}}$, as

$$\psi_c = \arctan2(n_y, n_x) \quad (10)$$

Then, $\xi_{\mathcal{F}}$ is calculated as the midpoint between $\mathbf{p}_{\text{right}}$ and \mathbf{p}_{left} as

$$\xi_{\mathcal{F}} = \frac{\mathbf{p}_{\text{right}} + \mathbf{p}_{\text{left}}}{2} \quad (11)$$

Next, the dimensions of the cross-section are determined. The width d_{width} corresponds to the distance between the two front corners, and the depth d_{depth} is chosen as the distance of the point in \mathbf{S}_{left} or $\mathbf{S}_{\text{right}}$ furthest from $\mathcal{L}_{\text{front}}$. Finally, the coordinates of ξ_c are calculated as

$$\xi_c = \xi_{\mathcal{F}} + \frac{d_{\text{depth}}}{2} \begin{pmatrix} \cos\psi_c \\ \sin\psi_c \end{pmatrix} \quad (12)$$

It is important to highlight that the visible cross-section can change drastically from one scan to the other, as is shown in Figure 4. This in return can produce large jumps in the estimates, since they are obtained from each individual laser scan. To reduce this effect and to obtain smoother results, $\xi_{\mathcal{F}}$, ψ_c and d_{depth} are filtered using first-order low-pass filters.

Limitations

Throughout the formulation of the tracking approach it was assumed that the cross-sections captured in the scans were rectangular. For this assumption to hold, the scan plane must remain horizontal. This is reasonable for most inspection tasks, where careful inspections require the MAV to operate at low speeds and inclinations remain small. However, external disturbances, such as strong winds, can produce large inclinations and bring the MAV to a configuration where the geometric model from equation (7) is no longer valid. Under such circumstances, tracking the tower with this approach will result inaccurate.

Another underlying constraint is that the MAV must always fly on the same side of the tower. This occurs because the entire approach is based on tracking $\mathcal{L}_{\text{front}}$. Since this line corresponds to the side of the tower closest to the MAV, if the MAV navigates around the tower eventually a different line will be tracked. This will cause shifts in the position and orientation estimates, since they are defined with respect to $\mathcal{L}_{\text{front}}$ (equations (10) and (11)).

Simulation results

Simulations were carried out using the set-up from Figure 3(b) to evaluate the performance of the proposed tracking algorithm. The initial position of the tower's centre with respect to the MAV was given, and the outlier rejection radius (as discussed in the 'Scan segmentation' section) was set to 4 m. The parameters for the RANSAC scan segmentation were chosen as $d_{\text{thresh}} = 5$ cm and $\psi_{\text{max}} = 10^\circ$. In the first test, the MAV was flown in front of one side of the tower for different heights and distances from the tower as shown in Figure 8. This figure also illustrates an example of a tracked cross-section with its corresponding front and centre frames. The resulting position and orientation estimates are compared to the simulation ground truth in Figure 9 (a), for a portion of the flight. As can be seen, throughout this flight the proposed approach is capable of effectively tracking the tower's centre. This is further verified from the absolute estimation errors, shown in Figure 9 (b), which remains below 5 cm for the translation components, and below 1° for the yaw angle.

In a second test, the MAV was flown around the tower, and the results are shown in Figure 10. In this case, the algorithm clearly fails at $t = 13$ s. This happens when the MAV transitions from one side of the tower to the other and the algorithm then starts tracking a different front line. This causes the 90° error in the orientation as seen in Figure 10. While the algorithm can track the centre of the tower again ($t = 15$ s as the position errors drop), the orientation error is not corrected. This illustrates one of the main limitations of the proposed approach.

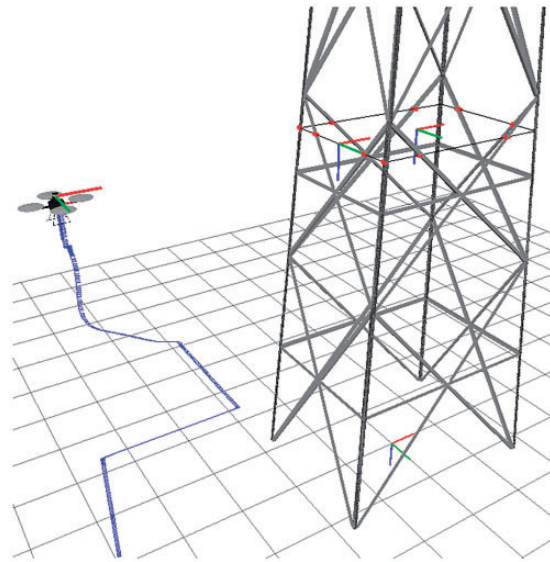


Figure 8. The simulated flight in front of the electric tower. The blue line indicates the trajectory followed by the quadrotor. An example of a tracked cross-section is visible on the right.

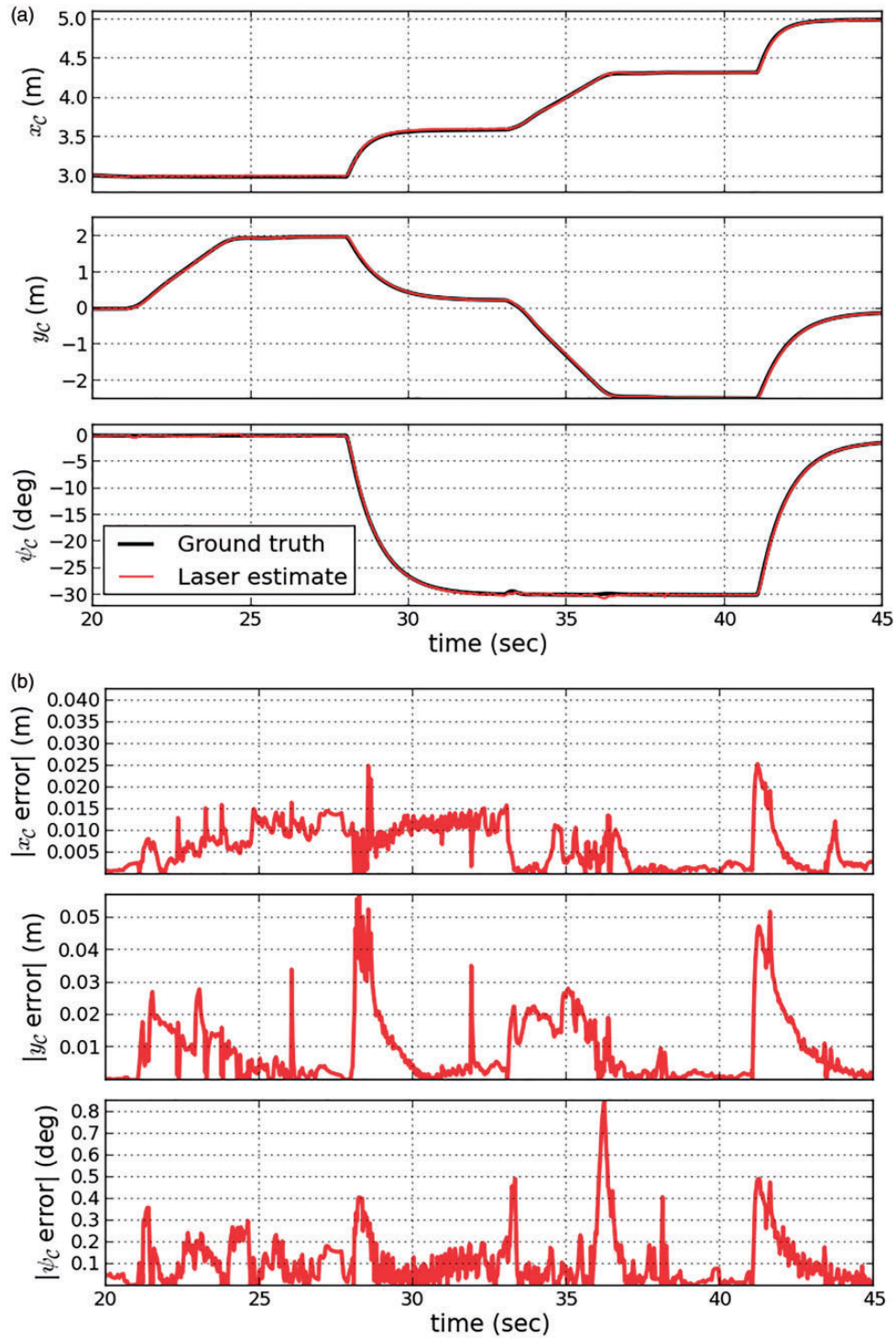


Figure 9. For the simulated flight from Figure 8: (a) The 2D pose of the tracked cross-section compared to the simulation ground truth and (b) absolute estimation errors.

Experimental results

The proposed tracking algorithm was also tested on data previously acquired from several manual test flights, where the MAV from Figure 2 was flown vertically in

front of an electric tower, as shown in Figure 3(a). An initial rough guess of the tower's centre with respect to the MAV was given, and the outlier rejection radius was set to 4 m. As already mentioned, besides the 2D LiDAR, the MAV was additionally equipped with a

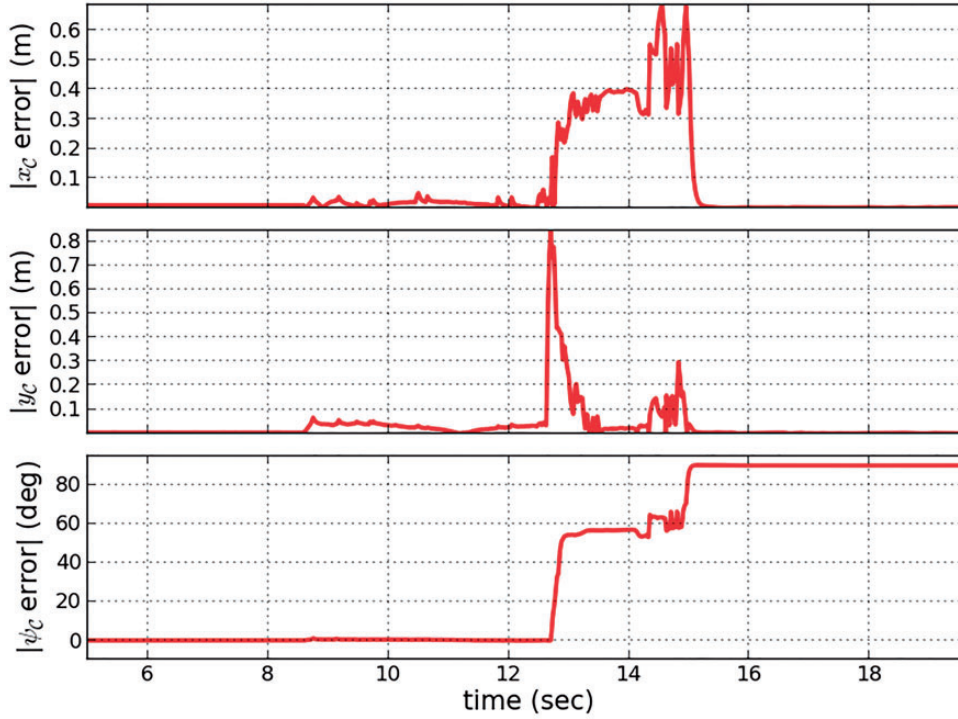


Figure 10. An example of the tracking method failing for a flight around the electric tower. Failure occurs at $t = 13$ s.

laser altimeter and an IMU. Unfortunately, at the time of the acquisitions no GPS sensor was used, and a ground truth is not available to determine the estimation errors. However, recalling that our tracking algorithm estimates the previously unknown depth and width of the tower's cross-sections, an alternative way of validating the approach is to determine if these dimensions are coherent with the 3D geometry of the real tower. Thus, Figure 11 illustrates the estimated dimensions combined with their corresponding estimated height from the laser altimeter readings, for one of the test flights. The efficiency of the 2D tracking algorithm is evident, since electric towers with rectangular cross-sections have a depth and width that vary linearly with height, a behaviour that is clearly reflected in Figure 11.

Modelling the electric tower. A by-product of tracking the cross-section's centre is the possibility of deriving a 3D representation of the electric tower from the observed data, such as a 3D point cloud reconstruction from the laser scans. A simple procedure consists in transforming each 2D scan into the estimated centre frame \mathcal{C} , and projecting into 3D coordinates using the height measurements and the attitude estimates from the IMU measurements. This was tested on the same vertical flight data used to obtain Figure 11, and the final result is shown in Figure 12. Here, the efficiency of the tracking method is also evident, as the point

cloud is capable of capturing a great amount of detail, and presents minimal deformations despite being made from data acquired on-flight.

A second possibility is to instead derive an abstract 3D geometric representation of the tower's body from the estimated dimensions presented in Figure 11. A simple approach is to approximate each face as a planar segment,⁶ and the edges of the tower as the intersection of adjacent planes m_j ($j = 1, \dots, 4$), expressed as

$$m_j: a_j x + b_j y + c_j z + d_j = 0, \quad j = 1, \dots, 4 \quad (13)$$

where each m_j is associated with a face of the tower. Obtaining the planes' coefficients results straightforward from Figure 11, as the slope of the fitted lines is directly related to the slopes of the planes. For example, for this particular case this resulted in

$$\begin{cases} m_1: -x - 0.062z - 1.643 = 0 \\ m_2: y - 0.046z - 1.265 = 0 \\ m_3: x - 0.062z - 1.643 = 0 \\ m_4: -y - 0.046z - 1.265 = 0 \end{cases} \quad (14)$$

which correspond to the front, right, back and left sides, respectively. With respect to an accurate point cloud reconstruction, which would require exploring

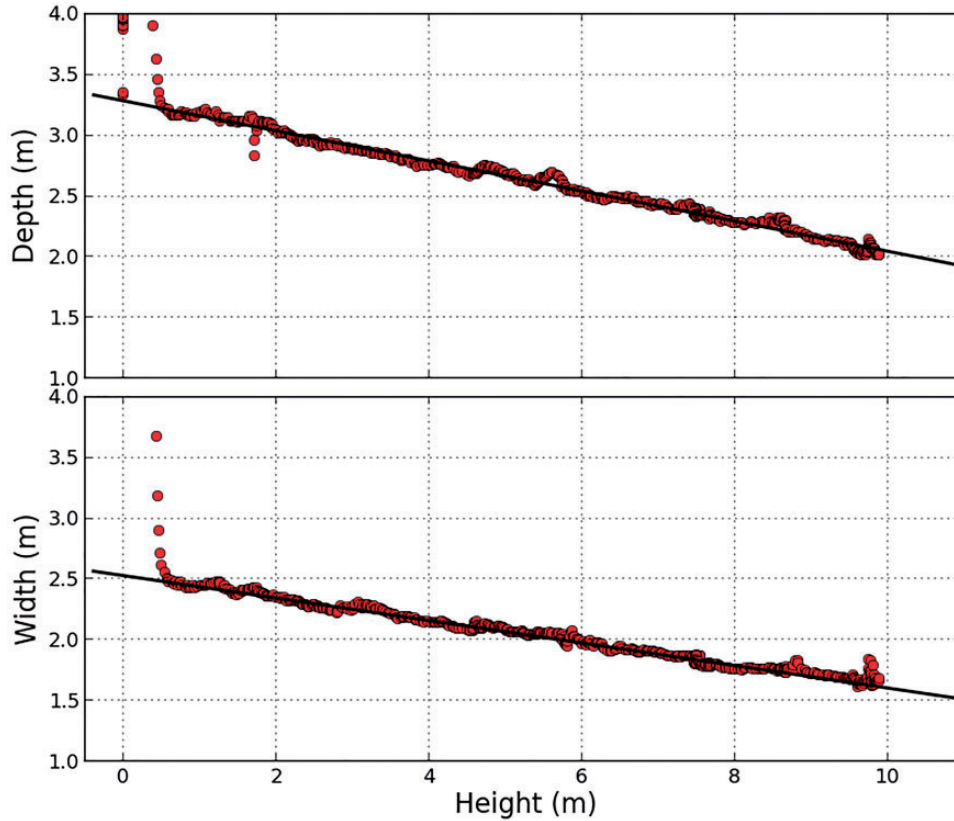


Figure 11. The estimated depth and width as a function of the height for the electric tower from Figure 3(a), fitted with straight lines.

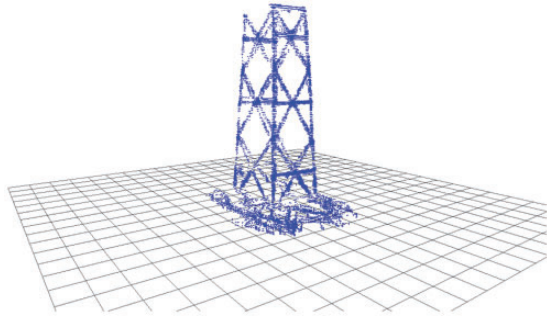


Figure 12. Partial 3D point cloud reconstruction of the electric tower from Figure 3(a), for a vertical flight in front of the tower. The laser scans are aligned using the tracked cross-section's centre, the quadrotor's altitude (from the laser altimeter) and attitude (from the IMU measurements).

extensive sections of the electric tower, this simplified planar representation can be obtained with more ease, as it only requires exploring a portion of the tower. As will be seen in the following sections, the main importance of these results is that both 3D representations of the tower can be exploited for pose estimation purposes.

Discussion

Since the final goal is to achieve autonomous navigation capabilities, all of the MAV's 6 DoF must be determined. For this purpose, this proposed tracking approach could be complemented with additional sensing to recover complete 3D pose estimates, for example, using inertial measurements to estimate the roll and pitch angles, and an altitude sensor, such as a laser altimeter or a barometer. However, the constraints imposed on the MAV's motion by this tracking approach are too restrictive for general inspection tasks that may require navigating continuously on all sides of the tower. An alternative strategy is thus to divide the inspection task into two steps. A first step consists in modelling the electric tower, which would allow to compensate for the limited information captured by the individual laser scans. The idea is to perform an initial vertical flight in front of the tower, in which our tracking algorithm is capable of providing a quantitative model of the tower (Figures 11 and 12). A second step would then focus on 3D pose estimation and navigation, using the estimated model to track the tower in general flight conditions. With such a model-based approach to recover pose estimates, the scan

plane no longer needs to remain horizontal and less restrictions are imposed on the MAV's movement. For the following sections, we consider that the first modelling step has already been performed based on our tracking approach, and instead focus the discussion on how to recover the complete 3D pose estimates.

3D local pose estimation

In this section, we present how to obtain complete 3D pose estimates with our sensor set-up. As is typically done with MAVs, the estimation process is broken down into several components.^{15,17} Recalling that the complete 6 DoF pose from \mathcal{B} to the inertial frame \mathcal{I} is described by $\{x, y, z, \phi, \theta, \psi\}$, the 3D pose is reconstructed as follows: $\{x, y, \psi\}$ are estimated from the laser range measurements; then, as will be discussed, $\{z\}$ is estimated from the laser range measurements fused with the barometer measurements; finally, $\{\phi, \theta\}$ are obtained by fusing accelerometer and gyrometer measurements from the IMU. The following subsections explain each component of the estimation process.

We first explore how the classic ICP algorithm that has been successful indoors can be extended to the case of an electric tower inspection. This technique requires the surrounding environment to have sufficient geometric detail and is not suitable for highly unstructured scenarios often faced outdoors.¹⁷ However, in an outdoor inspection scene, the rigid and well-defined structure of the electric towers has sufficient geometric detail to easily contrast from surrounding unstructured objects. This was exploited in the previous section to retrieve 2D pose estimates and will now be used to adapt the ICP algorithm. While common implementations focus on aligning pairs of scans to retrieve pose information in 2D, we instead treat the problem in 3D by introducing previous knowledge of the tower's geometry in the registration process. We now present two possible implementations of the ICP algorithm.

Adapting the ICP algorithm: First proposed approach

In this first approach we follow a line of work typically adopted with the ICP algorithm in navigation tasks, consisting in aligning point clouds. The idea is to maintain the approach as general as possible, as no specific parameterization of the scene is required and pose information is recovered directly from the point correspondences. Let the *current* scan be represented by a set of 2D points, denoted $\mathbf{S}_p = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{N_p}\}$. For simplicity, consider that \mathbf{S}_p is expressed in the body-attached frame \mathcal{B} . Then, let $\mathbf{S}_q = \{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_{N_q}\}$ denote the 3D *reference* set, expressed in the inertial frame \mathcal{I} , which corresponds to a 3D point cloud reconstruction of the inspection scene, assumed to be

acquired beforehand, e.g. from our tracking approach as discussed in the previous section. The goal is to find the rigid body transformation that best aligns \mathbf{S}_p to \mathbf{S}_q . The baseline ICP²⁴ was used, with several modifications, notably in the minimization step. Each iteration k (starting from $k = 0$) is carried out as follows:

1. **Initialization:** The current estimate $T_{\mathbf{x}_k}$ is used to transform all 2D points $\mathbf{p}_i \in \mathbf{S}_p$ into 3D coordinates in the inertial frame \mathcal{I} , obtaining $\mathbf{S}_{p'}$. For the first iteration, the parameter vector is $\mathbf{X}_0 = (x_{\text{laser}}, y_{\text{laser}}, z_{\text{laser}}, \phi_{\text{imu}}, \theta_{\text{imu}}, \psi_{\text{laser}})$, such that $\{x_{\text{laser}}, y_{\text{laser}}, z_{\text{laser}}, \psi_{\text{laser}}\}$ are obtained from the scan registration for the previous laser scan and $\{\phi_{\text{imu}}, \theta_{\text{imu}}\}$ from the IMU attitude estimation (as will be explained briefly).
2. **Matching:** Corresponding pairs $(\mathbf{p}'_i, \mathbf{q}_i)$ are established by associating each point in $\mathbf{S}_{p'}$ to the closest point in \mathbf{S}_q . This correspondence search is the most time-consuming step of the algorithm.²⁴ To speed up the matching process we make use of K-D trees, as is commonly done with ICP.^{24,26}
3. **Rejection:** Point pairs separated by more than a fixed distance threshold d_{\min} are removed. This is mainly helpful with accuracy and stability in the presence of outliers,²⁶ which in this case are typically due to surrounding vegetation.
4. **Minimization:** The goal is to find the transformation $T_{\mathbf{x}_{\min}}$ that minimizes the sum of squared errors, using the Euclidean distance as the distance metric.²⁴ For the N remaining point pairs $(\mathbf{p}'_i, \mathbf{q}_i)$, this leads to the following optimization problem

$$\mathbf{X}_{\min} = \arg \min_{\mathbf{x}} \sum_{i=1}^N \|T_{\mathbf{x}}(\mathbf{p}'_i) - \mathbf{q}_i\|^2, \quad (15)$$

such that $(\phi, \theta) = (0, 0)$

which is solved with the Levenberg–Marquardt algorithm, since it allows to obtain accurate results and deal with initialization errors without significant speed losses.⁴⁸ The components ϕ and θ of \mathbf{x} are neglected during the minimization, since ϕ_{imu} and θ_{imu} used at the initialization are precise and reliable. This reduces the optimization problem from a 6D space to a 4D space, which further limits the risk of divergence due to local minima, and provides a more reliable solution. This is the main modification of the algorithm.

5. Finally, the current estimate is updated as

$$T_{\mathbf{x}_{k+1}} = T_{\mathbf{x}_{\min}} \cdot T_{\mathbf{x}_k} \quad (16)$$

Due to the previous step, $T_{\mathbf{x}_{\min}}$ only updates the $\{x, y, z, \psi\}$ components of $T_{\mathbf{x}_k}$ in each iteration.

The end result of the scan registration process is an estimation of the 3D translation vector $(x_{\text{laser}}, y_{\text{laser}}, z_{\text{laser}})$ and the yaw angle ψ_{laser} . The main novelty is thus that altitude estimates can now be recovered, which is a direct consequence of introducing a 3D point cloud reconstruction of the tower in the registration process.

Limitations. Besides the drawbacks inherent to the ICP algorithm discussed at the beginning of this article, other limitations can be pointed out. Evidently, this approach is restricted to sections of the tower captured in the 3D point cloud reconstruction. Pose estimates cannot be recovered in previously unexplored or occluded sections. For this approach to be effective, the 3D point cloud must accurately capture the complete electric tower, which is a complex task. With our tracking algorithm from the previous section this requires exploring extensive portions of the tower. Other existing solutions rely on offline processing of data from powerful and expensive 3D LiDARs capable of capturing dense measurements from long distances.^{6,19} This, however, goes beyond the scope of this work.

Further complications arise regarding the altitude estimates. For a 2D LiDAR, measurements from the individual scans fall within the same plane and do not directly capture the MAV's altitude, which is instead determined from the point correspondences with the 3D point cloud uniquely. The altitude estimates are thus more unreliable and prone to errors, as will be seen in the simulation results. Furthermore, altitude estimation is highly dependent on the inclination of the faces of the tower. In the worst-case scenario, no altitude information can be recovered for completely vertical faces, which is a situation rarely faced with high voltage electric towers considered in this work. These drawbacks justify the use of an additional barometer sensor. However, as will be seen, this proposed ICP implementation will overall perform well if the electric tower remains within the sensor's field of view, and particularly stable results can be achieved for near-hovering conditions. This quality holds for altitude estimates and will be exploited to track the barometer drift.

Adapting the ICP algorithm: Second proposed approach

The difficulties in obtaining an accurate 3D point cloud reconstruction of the inspection scene can render the previous approach impractical. Nonetheless, the ICP algorithm can be applied to a wide variety of representations of geometric data such as line sets, triangle sets,

parametric surfaces, among others.²⁴ Therefore, an alternative is to align the laser scans onto the simplified planar representation of the tower body from equation (13), which is simpler to obtain than a complete point cloud reconstruction, as previously discussed. To achieve this, we adopt a projection-based matching strategy,^{49,50} where, after initialization, the corresponding points \mathbf{q}_i are calculated as the orthogonal projection of every point $\mathbf{p}'_i \in \mathbf{S}_{p'}$ onto the closest planar segment from m_j . This substitutes the time-consuming correspondence search previously used, and, as will be seen, allows obtaining significant speed gains.²⁶

Thus, in this approach, the matching step (step 2) for each point \mathbf{p}'_i is now carried out as follows:

- For the tower face m_j (starting with $j = 1$), calculate the two edge lines \mathcal{L}_A and \mathcal{L}_B as the intersection with the two adjacent planes.
- Project \mathbf{p}'_i orthogonally to the plane equation of m_j (equation (13)), obtaining \mathbf{q} . We have to determine if \mathbf{q} falls within the planar segment delimited by \mathcal{L}_A and \mathcal{L}_B . This is done as follows:
 - Project \mathbf{p}'_i to the edge lines \mathcal{L}_A and \mathcal{L}_B , obtaining \mathbf{q}_A and \mathbf{q}_B , respectively.
 - Let $AB = \mathbf{q}_B - \mathbf{q}_A$.
 - Calculate the normalized projection $\rho = \frac{(\mathbf{q} - \mathbf{q}_A) \cdot (AB)}{\|AB\|^2}$.
 - If $0 < \rho < 1$, then \mathbf{q} falls within the planar segment, and the projection is \mathbf{q} .
 - If $\rho \leq 0$, then \mathbf{q} falls outside of the planar segment and the projection is \mathbf{q}_A .
 - If $\rho \geq 1$, then \mathbf{q} falls outside of the planar segment and the projection is \mathbf{q}_B .

These steps are repeated for the four faces of the tower, and the projected point which yields the minimum distance to \mathbf{p}'_i is chosen as the corresponding point \mathbf{q}_i . Then, the remaining steps from the previous implementation are left unchanged. As before, the output is $(x_{\text{laser}}, y_{\text{laser}}, z_{\text{laser}}, \psi_{\text{laser}})$.

Limitations. One of the main drawbacks of this formulation is that it applies specifically to the case of rectangular cross-sections. The projection strategy would have to be changed for a different tower geometry. In contrast, the point cloud approach is more general in this matter and would not require any modifications. As before, no altitude information can be recovered if the faces of the tower are completely vertical.

Altitude estimation

The altitude estimates obtained previously from the laser range measurements have a strong dependence on the shape of the tower and can result unreliable. In contrast, barometer measurements are independent

from the shape of surrounding structures, but suffer from drift over time due to varying atmospheric conditions. Here, we seek to combine both sources of altitude information in order to tackle their respective drawbacks. We first recall the MAV's vertical dynamics with respect to an inertial frame \mathcal{I} from equation (4)

$$\begin{cases} \dot{z} = v_z \\ \dot{v}_z = g + \frac{F_z}{m} \end{cases} \quad (17)$$

Accurate vertical velocity estimates can be obtained by fusing the barometer and IMU measurements,^{31,32} and are thus obtained separately, as will be addressed in a later discussion. Therefore, in this section we consider that v_z is a known input, and instead use the following system

$$\begin{cases} \dot{z} = v_z \\ \dot{b}_z = 0 \end{cases} \quad (18)$$

where b_z is the unknown barometer drift, which is modelled as a constant as it varies slowly in time, and is defined by the relationship $z_{\text{baro}} = z + b_z$, with z_{baro} denoting the barometer measurement. This leads to a simple second-order feedback observer formulation

$$\begin{cases} \dot{\hat{z}} = v_z - k_z(\hat{z} - \bar{z}_1) \\ \dot{\hat{b}}_z = -k_{b_z}(\hat{z} - \bar{z}_2) \end{cases} \quad (19)$$

where (k_z, k_{b_z}) are the estimation gains, and \bar{z}_n is an auxiliary variable defined as

$$\bar{z}_n = \lambda_n(z_{\text{baro}} - \hat{b}_z) + (1 - \lambda_n)z_{\text{laser}} \quad \text{with} \quad 0 \leq \lambda_n \leq 1, \quad n = 1, 2 \quad (20)$$

which is the weighted sum of the laser altitude estimates z_{laser} and barometer readings z_{baro} compensated for bias. The weights λ_n allow one to determine how each sensor contributes to the estimation of each state. In particular, as λ_n increases, higher priority is given to the barometer readings. The reasoning behind this parameterization is to use the laser estimates mainly to keep track of slowly varying barometer bias \hat{b}_z and to maintain the more reliable barometer measurements to estimate \hat{z} . Choosing the weights $\lambda_1 = 1$ and $\lambda_2 = 0$ achieves this purpose. The stability analysis for this observer and details on how to tune the gains (k_z, k_{b_z}) are given in Appendix 1.

Attitude estimation

We now present our proposed non-linear observer formulation using the accelerometer and gyroscope measurements. As yaw estimates are already obtained from the laser scan registration, the main goal is to recover estimates of the roll ϕ and pitch θ angles. First, let $\gamma = (\gamma_1, \gamma_2, \gamma_3)^\top$ denote the vertical axis of \mathcal{I} expressed in \mathcal{B} as

$$\gamma = R^\top e_3 \quad (21)$$

with $e_3 = (0, 0, 1)^\top$. From the rotation matrix definition of equation (1), it follows that γ contains implicitly the MAV's roll and pitch angles, since

$$\begin{aligned} \phi &= \arcsin(\gamma_2) \\ \theta &= \text{atan2}(-\gamma_1, \gamma_3) \end{aligned} \quad (22)$$

Recalling that a MAV's rotational kinematics is given by³³

$$\dot{R} = RS(\omega) \quad (23)$$

with $S(\cdot)$ the skew-symmetric matrix associated with the cross-product (i.e. $S(\mathbf{x})\mathbf{y} = \mathbf{x} \times \mathbf{y}$, $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^3$), and ω the angular velocity vector from \mathcal{B} to \mathcal{I} , expressed in \mathcal{B} . Then, the kinematics of γ can be deduced from equations (21) and (23), and results in

$$\dot{\gamma} = \gamma \times \omega \quad (24)$$

This is the basis of our observer formulation. As previously mentioned, the goal is to recover roll and pitch estimates from the gyrometer and accelerometer readings. Let \mathbf{a}_m denote the accelerometer measurements expressed in \mathcal{B} , which measure the specific acceleration acting on the MAV's airframe³³

$$\mathbf{a}_m = R^\top (\dot{v} - g e_3) = R^\top \dot{v} - g \gamma \quad (25)$$

Then, under the assumption of negligible linear acceleration, one has³⁵

$$\mathbf{a}_m \approx -g \gamma \quad (26)$$

which shows that accelerometers provide direct observations of the roll and pitch angles (and of γ). Thus, the following non-linear observer for γ is proposed

$$\dot{\hat{\gamma}} = \hat{\gamma} \times (\omega_m - k_\gamma(\mathbf{a}_m \times \hat{\gamma})), \quad k_\gamma > 0 \quad (27)$$

with ω_m the angular velocities measured by the gyrometer in \mathcal{B} .

To analyse the stability of this estimator, consider the candidate Lyapunov function $L = 1 - \gamma^T \hat{\gamma}$. From equations (26) and (27) one has

$$\dot{\hat{\gamma}} \approx \hat{\gamma} \times (\omega_m - k_\gamma g(\hat{\gamma} \times \gamma)) \quad (28)$$

Then, assuming that this approximation of $\hat{\gamma}$ is perfect, and that $\omega_m = \omega$, it can be proven that $\dot{L} = -k_\gamma g \|\hat{\gamma} \times \gamma\|^2$, which is decreasing along the solutions of the system if, initially, $\hat{\gamma}$ and γ are not opposite to each other, and $k_\gamma > 0$. This implies in particular the convergence of $\hat{\gamma}$ to γ .

Gain scheduling. The approximation from equation (26) is commonly used in attitude estimation when dealing with accelerometers,³⁵ but only holds when flying at constant velocity or near stationary flight conditions. An added benefit of non-linear observer formulations is that the estimation gains can be tuned in real time during flight.³³ This can be exploited to adapt the observer to changing dynamic conditions, in particular, to high acceleration states where the assumption from equation (26) is no longer valid and estimation performance is deteriorated. In such situations, which typically last for short periods of time, it is better to lower the estimation gains and to rely on the gyrometer measurements since they are scarcely affected by the linear accelerations,⁵¹ and can provide short-term rotations accurately.³⁸

A basic strategy is thus to detect highly accelerated states by comparing the magnitude of the accelerometer readings to the gravity acceleration.^{36,37,51} Let \tilde{a}_m

denote the absolute accelerometer measurement error with respect to gravity as

$$\tilde{a}_m = ||a_m|| - g, \quad g = 9.81 \frac{m}{s^2} \quad (29)$$

This magnitude provides a simple criteria to determine the dynamic state of the MAV, as $\tilde{a}_m \approx 0$ for near-hovering conditions, and large values of \tilde{a}_m correspond to highly dynamic motion. The estimation gains can then be adapted accordingly. Yoo et al.³⁶ adopt a simple switching strategy to choose the gain between a set of nominal values corresponding to no-acceleration, low-acceleration or high-acceleration states. Instead, Valenti et al.⁵¹ set a nominal gain for hovering state, which is then decreased linearly during transitions to high acceleration states. We adopt a strategy similar to Valenti et al.⁵¹ Let k_L and k_H denote the nominal gains during low- and high-acceleration states, respectively; the idea is to transition smoothly between these gains. The following gain scheduling approach is proposed

$$k_\gamma(\tilde{a}_m) = k_L e^{-\alpha \tilde{a}_m} + k_H (1 - e^{-\alpha \tilde{a}_m}), \quad \alpha > 0 \quad (30)$$

where α is an arbitrary positive constant that determines the steepness of the transitions between k_L and k_H . It is simple to verify that as $\tilde{a}_m \approx 0$, then k_γ remains near k_L , and as \tilde{a}_m increases, then k_γ decreases towards k_H , which is the desired behaviour. This is further illustrated in Figure 13 for $k_L = 0.1$, $k_H = 0.01$ and different values of α . It can be noted that $\alpha = 0$ corresponds to the constant gain case, and as α increases, the gains decrease faster towards k_H .

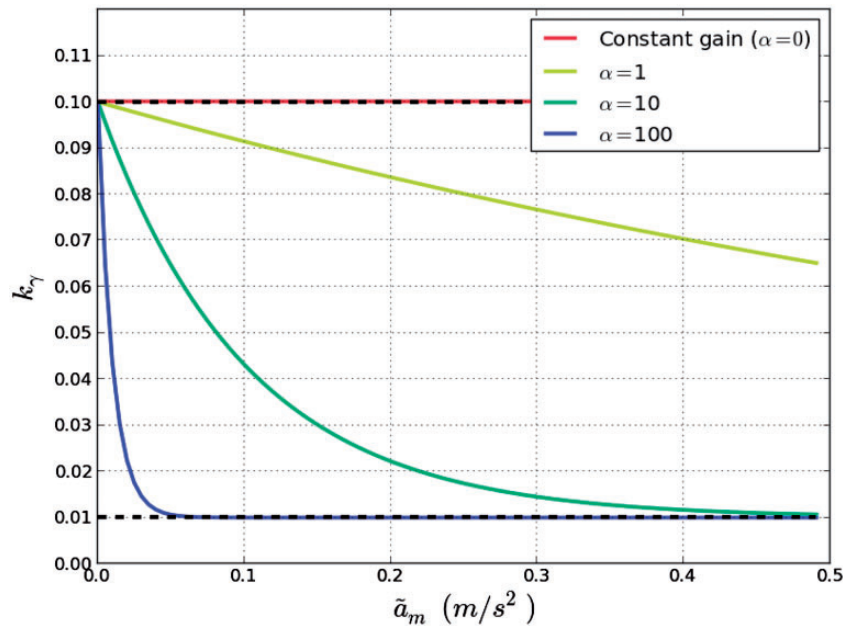


Figure 13. An example of the attitude observer gains according to equation (30), for different values of α .

Complete rotation matrix reconstruction. The estimated roll ϕ_{imu} and pitch θ_{imu} angles are recovered from $\hat{\gamma}$ and equation (22) as

$$\begin{aligned}\phi_{\text{imu}} &= \arcsin(\hat{\gamma}_2) \\ \theta_{\text{imu}} &= \text{atan2}(-\hat{\gamma}_1, \hat{\gamma}_3)\end{aligned}\quad (31)$$

Finally, the complete estimated rotation matrix \hat{R} is recovered by combining the estimated angles as

$$\hat{R} = R_z(\psi_{\text{laser}})R_x(\phi_{\text{imu}})R_y(\theta_{\text{imu}}) \quad (32)$$

This matrix is subsequently used at each initialization of the laser scan registration, and for the velocity estimation described in the following section.

Velocity estimation

In the previous section, the complete six DoF pose of the MAV was determined from the sensor measurements. The goal is now to derive velocity estimates by combining the pose estimates with the inertial measurements from the IMU. For this purpose, we make use of the translational dynamics of the MAV with respect to the inertial frame \mathcal{I} from equation (3) to formulate velocity observers. In the following analysis, the external aerodynamic forces $\mathbf{F} = (F_x, F_y, F_z)^\top$ from equation (4) are determined from the accelerometer readings a_m and the estimated attitude \hat{R} as

$$\mathbf{F} = m\hat{R}\mathbf{a}_m \quad (33)$$

Since different sensors are used for the different states, the horizontal and vertical velocity components are analysed separately.

Horizontal velocity estimation

From equation (4) it follows that the dynamics for $\{x, y\}$ in \mathcal{I} are two independent second-order systems. Estimating the horizontal velocities results straightforward and is achieved with simple feedback state observers defined as

$$\begin{cases} \dot{\hat{x}} = \hat{v}_x - k_x(\hat{x} - x_{\text{laser}}) \\ \dot{\hat{v}}_x = \frac{F_x}{m} - k_{v_x}(\hat{x} - x_{\text{laser}}), & k_x, k_{v_x} > 0 \\ \dot{\hat{y}} = \hat{v}_y - k_y(\hat{y} - y_{\text{laser}}) \\ \dot{\hat{v}}_y = \frac{F_y}{m} - k_{v_y}(\hat{y} - y_{\text{laser}}), & k_y, k_{v_y} > 0 \end{cases} \quad (34)$$

where (k_x, k_{v_x}) and (k_y, k_{v_y}) are the scalar observer gains, which guarantee exponential convergence if they are positive, and $(x_{\text{laser}}, y_{\text{laser}})$ are the estimates obtained from the laser scan registration described in the previous section.

Vertical velocity estimation

As previously mentioned, satisfying estimates of the vertical velocity can be recovered from barometer and accelerometer measurements.^{31,32} As will be seen, these estimates remain accurate even in the presence of barometer drift. Recalling the vertical dynamics from equation (17), we now formulate the following feedback state observer

$$\begin{cases} \dot{\hat{z}} = \hat{v}_z - k_z(\hat{z} - z_{\text{baro}}) \\ \dot{\hat{v}}_z = g + \frac{F_z}{m} - k_{v_z}(\hat{z} - z_{\text{baro}}), & k_z, k_{v_z} > 0 \end{cases} \quad (35)$$

where (k_z, k_{v_z}) are the observer gains, and z_{baro} are the barometer altitude measurements. The altitude estimates from the laser scan registration are not included as they only degrade the performance. The vertical velocity estimates \hat{v}_z are subsequently used as an input to the altitude observer from equation (19).

Simulation results: 3D local pose estimation

The purpose of this section is to assess the performance of the different components of the pose estimation process. The results presented here were obtained from simulated flights carried out using the previously discussed simulation set-up, illustrated in Figure 3(b). For the flights, a set of waypoints was given for the quadrotor to follow, accounting for a complete displacement around the tower. Meanwhile, the MAV's yaw angle was oriented towards the centre of the tower, so that the latter remains in the LiDAR's field of view. Since the focus of this section is to assess the quality of the pose estimates, the simulation ground truth is directly used to stabilize the MAV's position and attitude. The complete flight is shown in Figure 14.

Attitude estimation results

The attitude observer from equation (27) was used to fuse the accelerometer and gyrometer measurements and recover estimates of the roll and pitch angles $\{\phi, \theta\}$. We now analyse the performance of this observer throughout the flight. Figure 15(a) illustrates the deviations of the accelerometer readings from the acceleration of gravity (\tilde{a}_m from equation (29)) for a portion

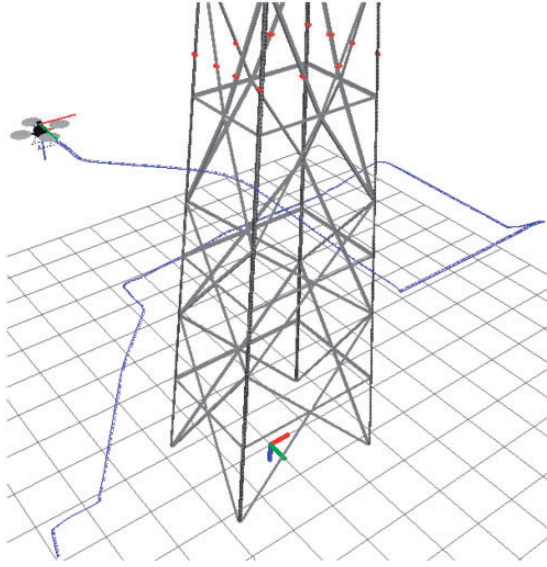


Figure 14. The simulated flight around the tower. The blue line indicates the trajectory. Throughout the flight the quadrotor was oriented towards the centre of the tower.

of the flight. As can be seen, the MAV spends larger amounts of time in low acceleration states ($\tilde{a}_m \approx 0$). Then, the peaks correspond to instants when the MAV accelerates towards a different waypoint. The idea is to adapt the observer to these peaks by lowering the estimation gain k_γ . This was carried out with the gain scheduling approach from equation (30). Based on results observed in the simulations, the nominal gains were set to $k_L = 0.1$ and $k_H = 0.01$. Moreover, the estimation process was repeated for different values of the parameter α , from $\alpha = 0$, which corresponds to the constant gain case since $k_\gamma = k_L$ (from equation (30)), to $\alpha = 100$. As explained, this parameter determines the steepness of the transitions between the two nominal gains. The resulting scheduled gains for a portion of the flight are shown in Figure 15(b). When comparing the two figures, it can be noted the gain k_γ rapidly drops in the presence of acceleration peaks (e.g. $t = 26$ s and $t = 30$ s), which is the desired behaviour. However, as α increases, the gains can result overly sensitive to small changes in \tilde{a}_m ($t = 38$ s for $\alpha = 100$).

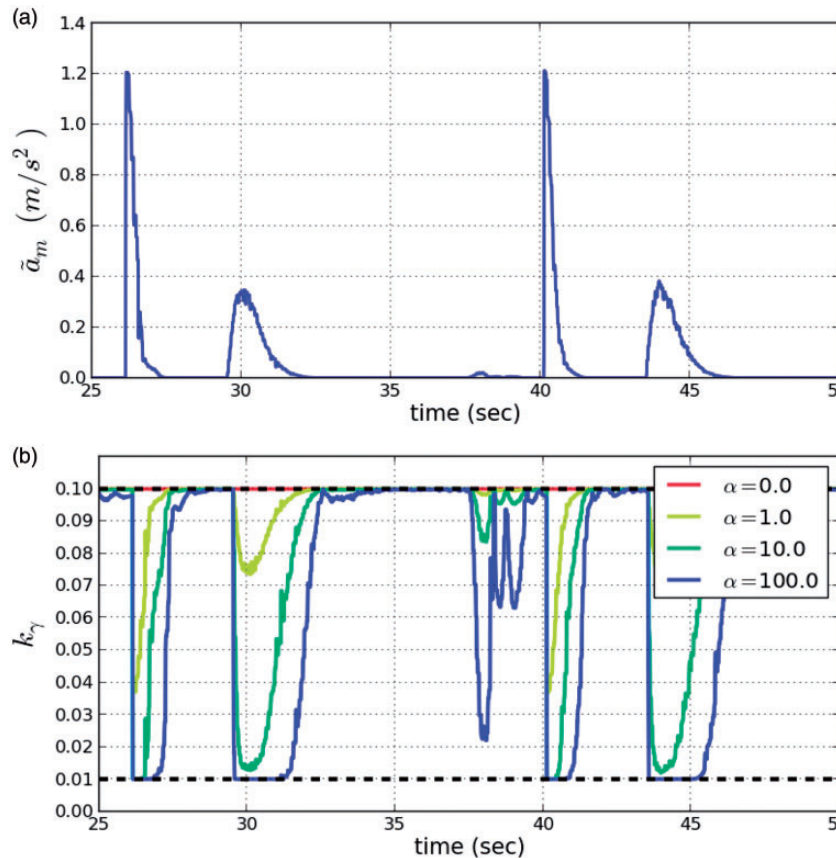


Figure 15. For a portion of the flight from Figure 14: (a) The deviations of the accelerometer readings from gravity according to equation (29). (b) The resulting scheduled gains for different values of α . The gains become more reactive for larger values of α .

Next, the absolute estimation errors with respect to the simulation ground truth are shown in Figure 16(a) and (b) for the roll and pitch angles, respectively. On the one hand, when comparing Figure 16(a) and (b) to (c), it can be noted that the observer can accurately trace the roll and pitch angles in low acceleration states (when $\tilde{a}_m \approx 0$) for all cases, and the errors for the most part remain below 1° . On the other hand, the largest estimation errors correspond to peaks in \tilde{a}_m (e.g. $t = 26$ s and $t = 40$ s),

reaching a maximum for the constant gain observer of 2.45° for the roll angle, and 2.62° for the pitch angle. In contrast, as the parameter α is increased, error peaks related to \tilde{a}_m are now largely suppressed and the overall performance is improved with the simple gain scheduling strategy. Based on these results, a gain scheduled attitude estimation with $\alpha = 10$ was used for the following sections, as it offers a good trade-off between sensibility to changes in \tilde{a}_m and estimation error reduction.

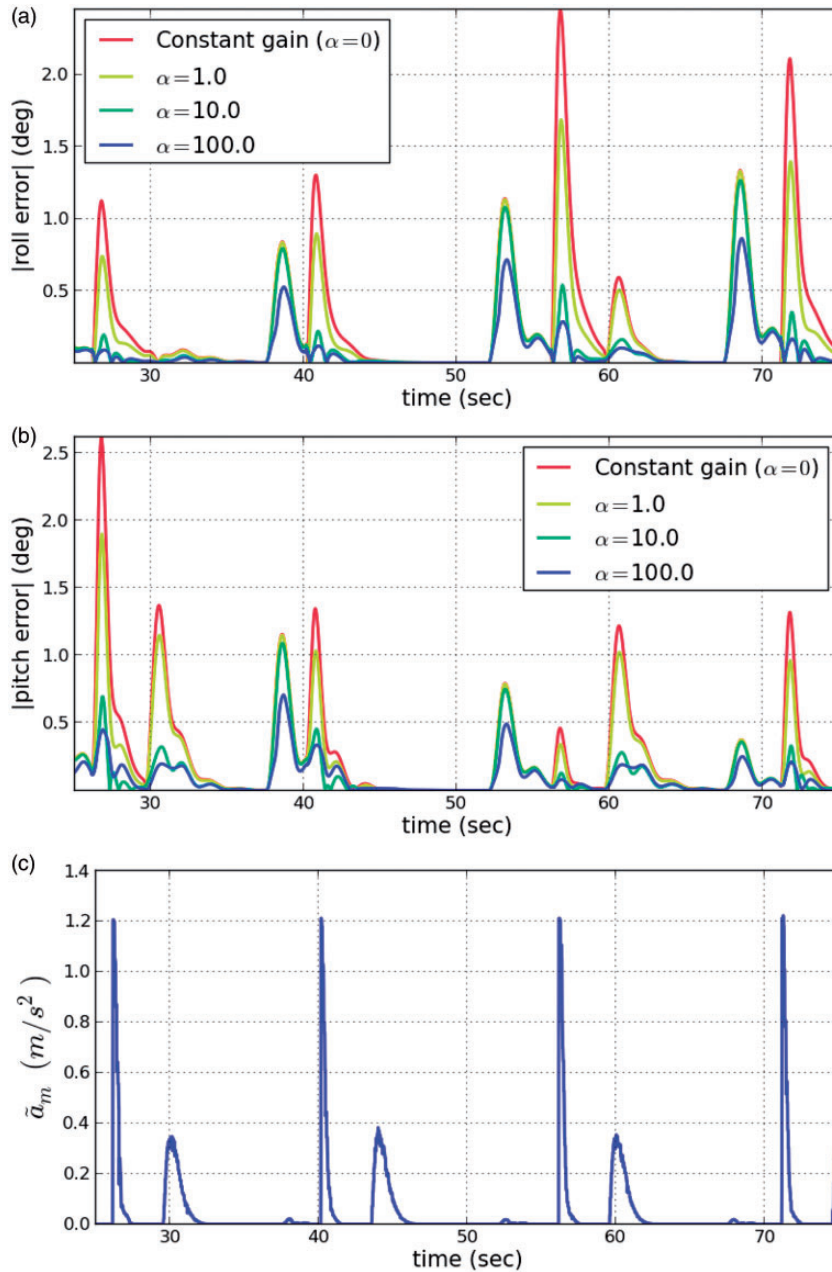


Figure 16. For the attitude observer from equation (27) and different values of α : (a) Absolute roll estimation error, (b) absolute pitch estimation error and (c) the corresponding \tilde{a}_m . As α increases, the errors caused by peaks in \tilde{a}_m are reduced.

Laser-based local pose estimation results

The two proposed implementations of the ICP algorithm were tested in the simulations to recover estimates of the remaining states $\{x, y, z, \psi\}$. In the first case, the laser scans were aligned to the 3D point cloud reconstruction shown in Figure 17(a), which was obtained beforehand from the simulated tower shown in Figure 3(b), following the same procedure for Figure 12 based on our tracking approach. In the second case, the laser scans were instead aligned to the planar representation of the tower illustrated in Figure 17(b), which was also obtained beforehand following the procedure used for Figure 11 and equation (14), relying on our tracking approach. Here, the estimated coefficients from equation (7) resulted in

$$\begin{cases} m_1 : -x - 0.076z - 1.749 = 0 \\ m_2 : y - 0.046z - 1.219 = 0 \\ m_3 : x - 0.076z - 1.749 = 0 \\ m_4 : -y - 0.046z - 1.219 = 0 \end{cases} \quad (36)$$

which correspond to the front, right, back and left sides, respectively. In both ICP implementations, an initial rough knowledge of the MAV's position with respect to the tower was given for the first scan registration. For each subsequent laser scan, the estimation

process was initialized with the results from the previous scan registration and attitude estimates from the IMU measurements.

We now analyse the performance of the two approaches. First, the computation time required for the scan registration in both cases is shown in Figure 18. As expected, using the planar model results in significantly faster estimates with an average of 1.4ms, compared to the point cloud case average of 16ms. This shows the effectiveness of the projection-based matching strategy used to establish point correspondences, which avoids the computationally extensive correspondence search required for the point cloud registration.

Next, Figure 19 compares the MAV's ground truth position with the estimates from both approaches. As can be seen, the results obtained with the planar model approach effectively follow the ground truth for the duration of the flight. However, the point cloud approach ultimately fails before completing the flight. This can be further observed from the absolute errors shown in Figure 20. For the planar model case, the $\{x, y\}$ errors remain below 5cm (Figure 20(a) and (b)). Furthermore, the yaw estimates are also very precise, with a maximum error of 0.8° (Figure 20(d)). In these figures it can be noted that the point cloud approach achieves similar performance before failing (near $t = 40$ s). Then, particular attention must be given to

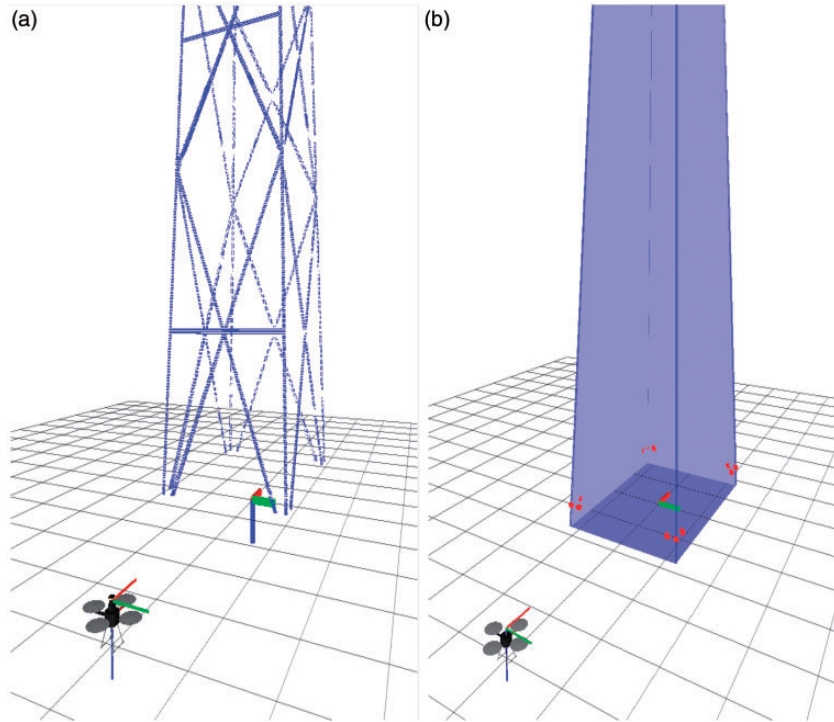


Figure 17. The two models used as reference in the ICP algorithm to align the laser scans: (a) Point cloud reconstruction and (b) planar model.

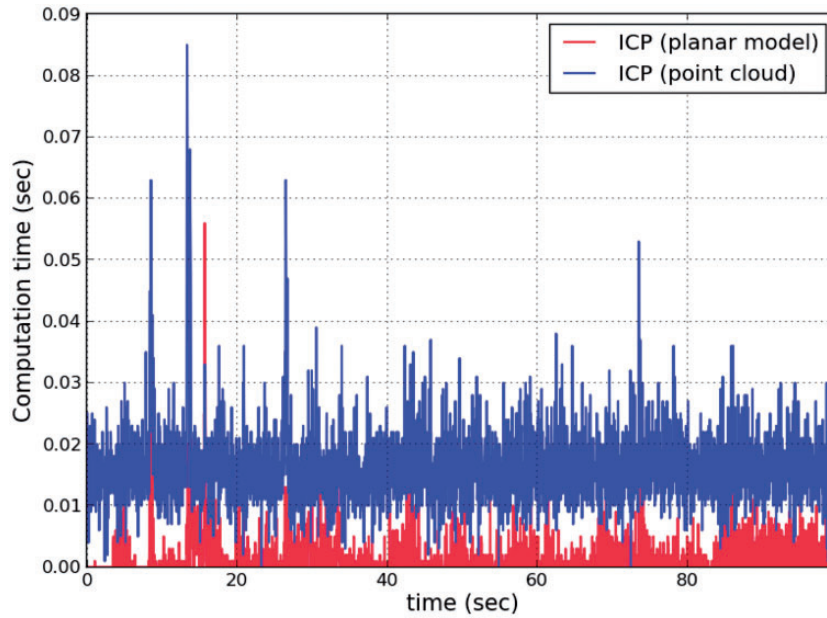


Figure 18. Comparing the computation time for the laser scan registrations. The planar model approach is approximately 10 times faster. ICP: iterative closest point.

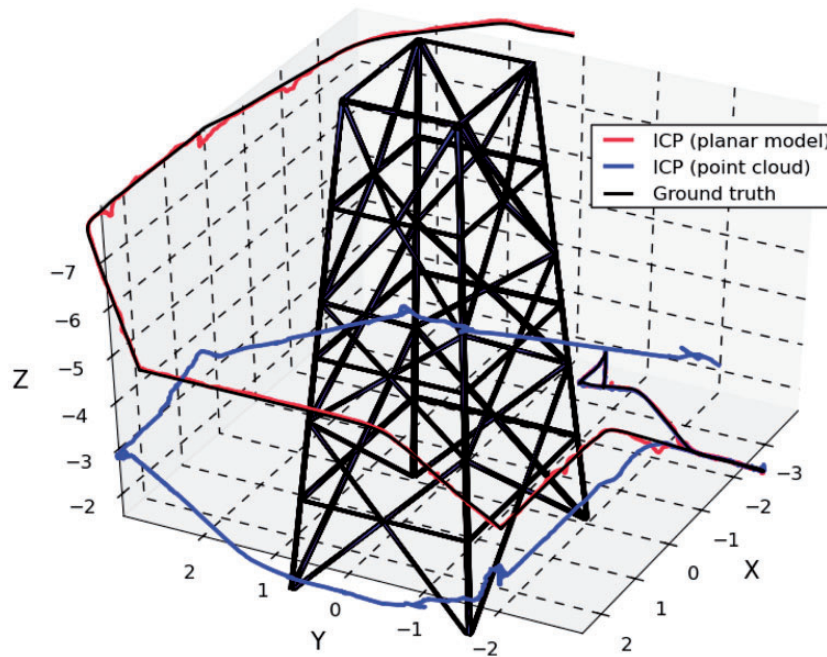


Figure 19. Comparing the ICP position estimates with the ground truth, for the simulated flight from Figure 14. The point cloud approach fails before finishing the flight. ICP: iterative closest point.

the altitude estimation errors from Figure 20(c). As previously pointed out, the horizontally placed 2D laser scanner captures very limited altitude information. As a result, it was observed throughout the simulations that the altitude estimates were easily deteriorated in complicated situations, for example,

when the horizontal bars on the tower block most of the sides from the sensor's view (as in Figure 4(d)). For the planar model case, this typically caused spurious estimates, with the absolute error jumping above 20 cm (e.g. $t = 20$ s and $t = 30$ s in Figure 20(c)). For the point cloud case, this eventually caused the

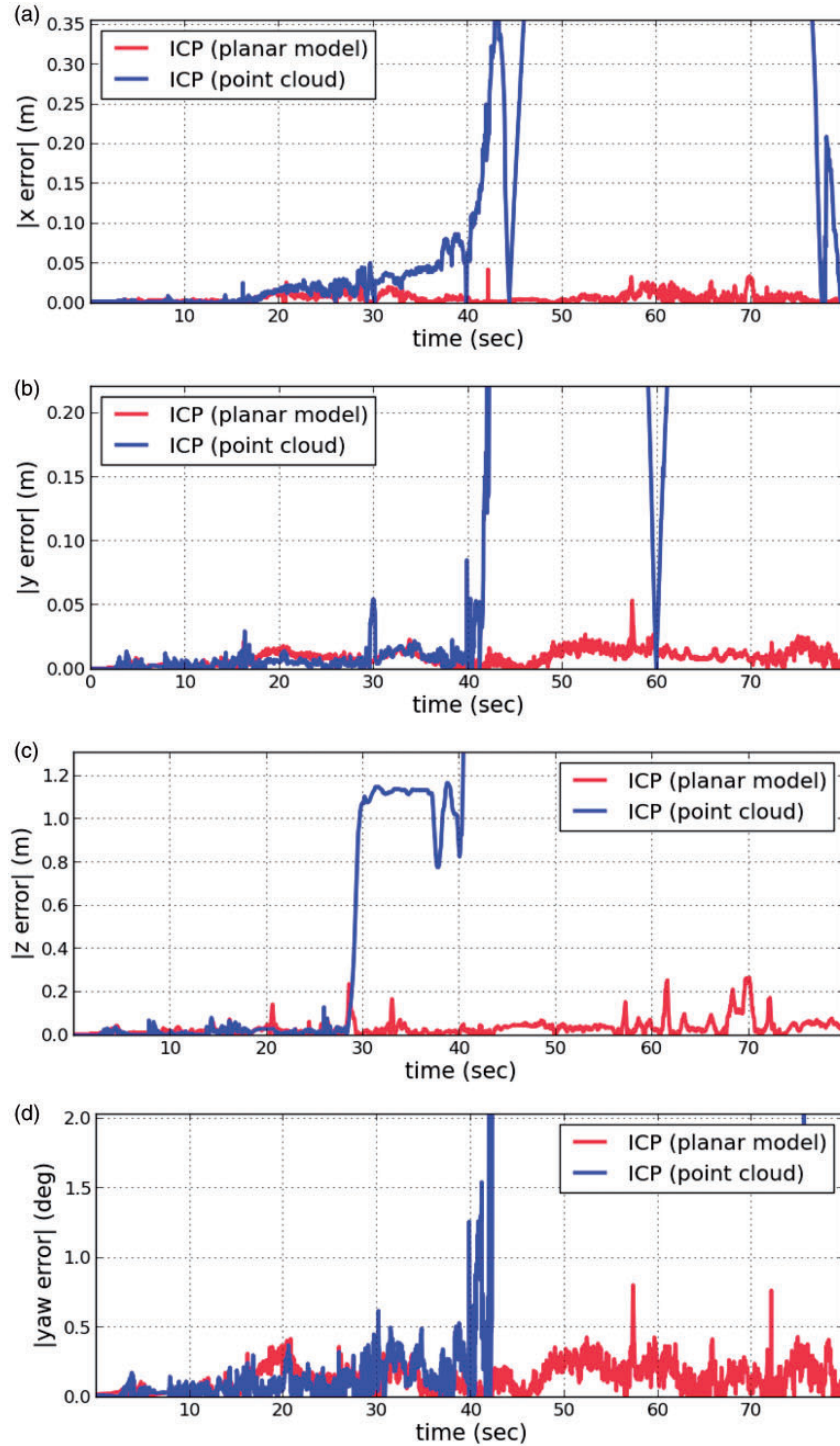


Figure 20. Absolute estimation errors with respect to the simulation ground truth for the results from Figure 19, for both implementations of the ICP algorithm. ICP: iterative closest point.

approach to completely fail (around $t = 30$ s in Figure 20(c)). Despite these complications, in Figure 20(c) it can be observed that the altitude errors remain at acceptable levels below 10 cm throughout most of the

flight for the planar case, and similarly for the point cloud approach before failure. Properly exploiting the limited altitude information requires special attention and is addressed in the following section.

Altitude estimation results

We now present the results for the altitude observer from equation (19), which fuses the laser altitude estimates from both implementations of the ICP algorithm with the barometer measurements. Barometer readings are sensitive to changes in atmospheric conditions (strong winds, temperature changes), which generally translates into a slowly varying drift. In order to study the observer's behaviour under large barometer drift, this was simulated as a sinusoid with a maximum speed of 1 m/min. As previously mentioned, the weights from equation (20) were chosen as $\lambda_1 = 1$, to rely mainly on the barometer measurements to estimate the altitude, and $\lambda_2 = 0$, to rely on the laser estimates to estimate the barometer drift. Then, the estimation gains were set to $(k_z, k_{b_z}) = (6.6, -1.36)$, which achieved a good performance in the simulations. An explanation on how to determine these gains is given in Appendix 1. In these simulations, the observer's output was used at each scan registration initialization, instead of the laser altitude estimates. The impact of this can be observed in Figure 21, where the complete position estimates of the ICP implementations are once again compared to the simulation ground truth. In contrast to the results from Figure 19, it can be noted that the introduction of the altitude observer allows correcting the large altitude errors

that previously caused the point cloud case to fail, which now offers a similar performance to the planar model case. Furthermore, Figure 22 presents a comparison of the absolute errors of the barometer measurements, the altitude estimates of the ICP algorithm (without the aid of the observer) and the altitude observer's output. In both cases it can be noted that, while the barometer readings accumulate a large error over time, the presence of this drift does not significantly degrade the quality observer's altitude estimates, which instead provides notable improvements. For the planar model case in Figure 22(a), the spurious error peaks are largely filtered, and the maximum error is lowered to 10 cm. More importantly, for the point cloud case in Figure 22(b), the observer manages to avoid failing at $t = 30$ s and provides continuous estimates throughout the entire flight. The effectiveness of this formulation is further verified in Figure 23, as the observer manages to estimate the previously unknown barometer drift, with less than 10 cm of error.

Simulation results: Velocity estimation

Horizontal velocity estimation results

The $\{x, y\}$ estimates from the laser scan registration were used as an input to the velocity observers from equation (34), where they were fused with

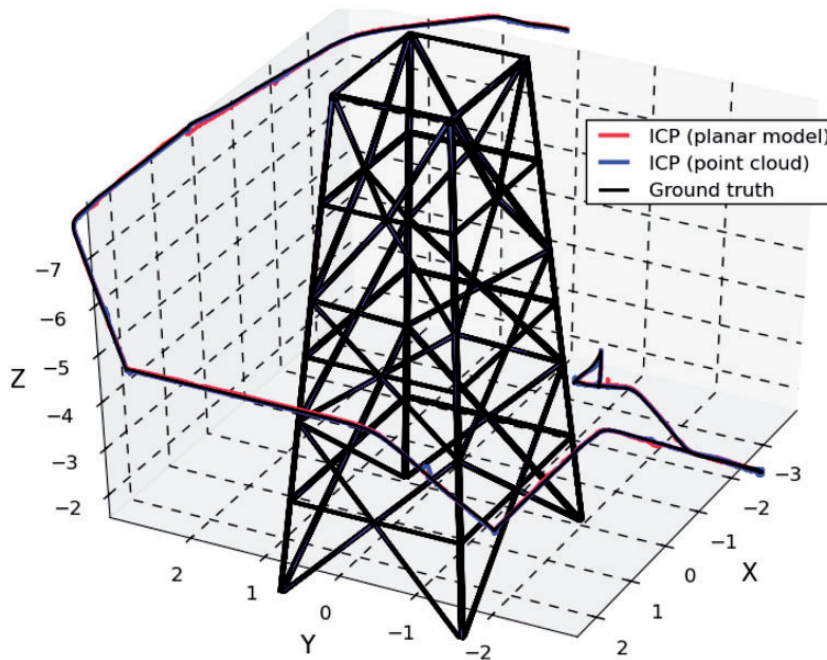


Figure 21. The ICP position estimates after introducing the altitude observer. The large altitude errors are corrected and the point cloud approach no longer fails. ICP: iterative closest point.

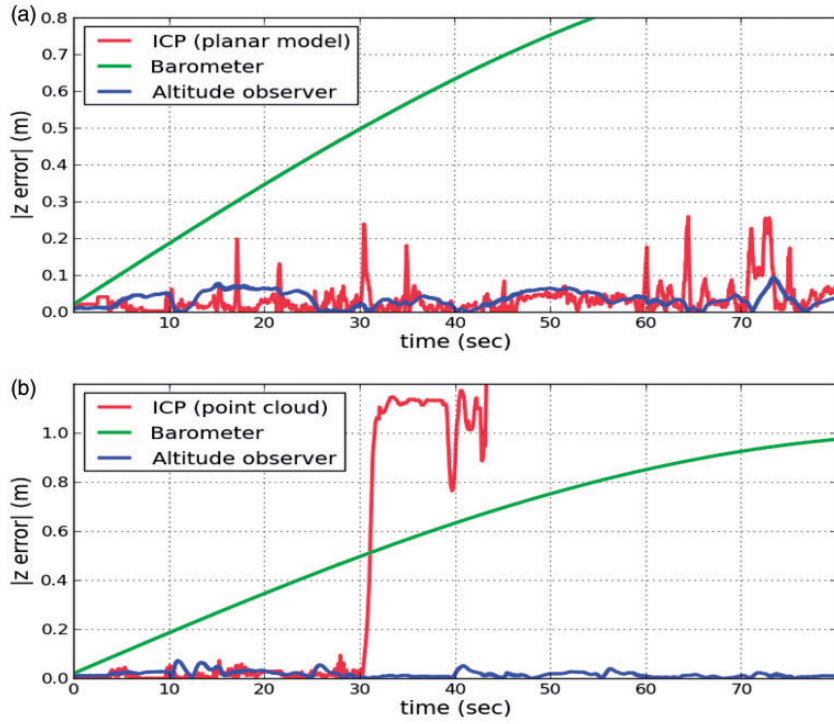


Figure 22. The absolute altitude errors for ICP without the aid of the altitude observer, the barometer measurements with drift and the altitude observer for (a) ICP with planar model and (b) ICP with point cloud reconstruction. ICP: iterative closest point.

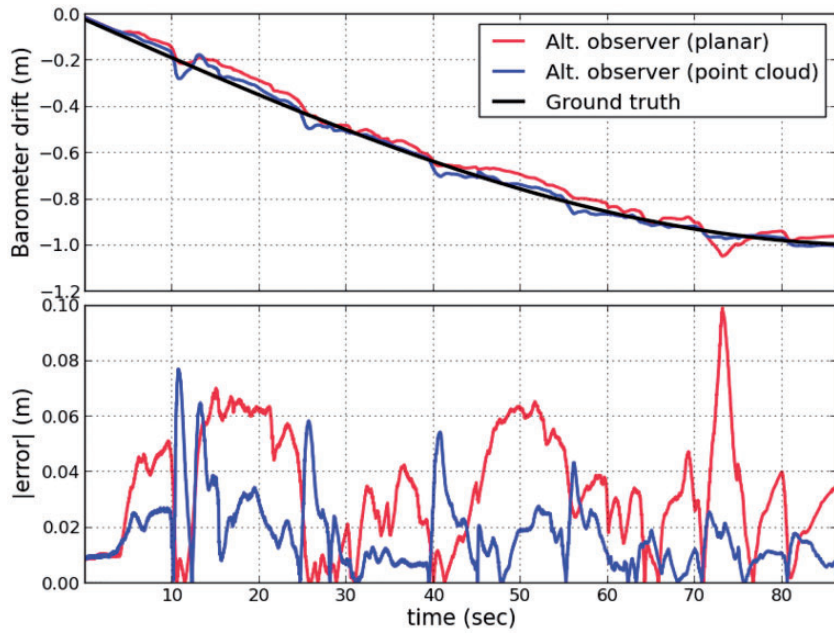


Figure 23. For the altitude observer and both ICP implementations: (Top) Comparing the barometer drift estimates with the ground truth. (Bottom) Absolute estimation errors. The observer succeeds in both cases.

the accelerometer readings to recover the horizontal velocity estimates. The estimation gains were chosen identical for both axis as $(k_x, k_{v_x}) = (6.4, 16)$ and $(k_y, k_{v_y}) = (6.4, 16)$. The estimated horizontal velocities

for a portion of the flight and both implementations of the ICP algorithm are shown in Figure 24. For both axes, the high estimation gains allow the speed estimates to converge fast towards the ground truth.

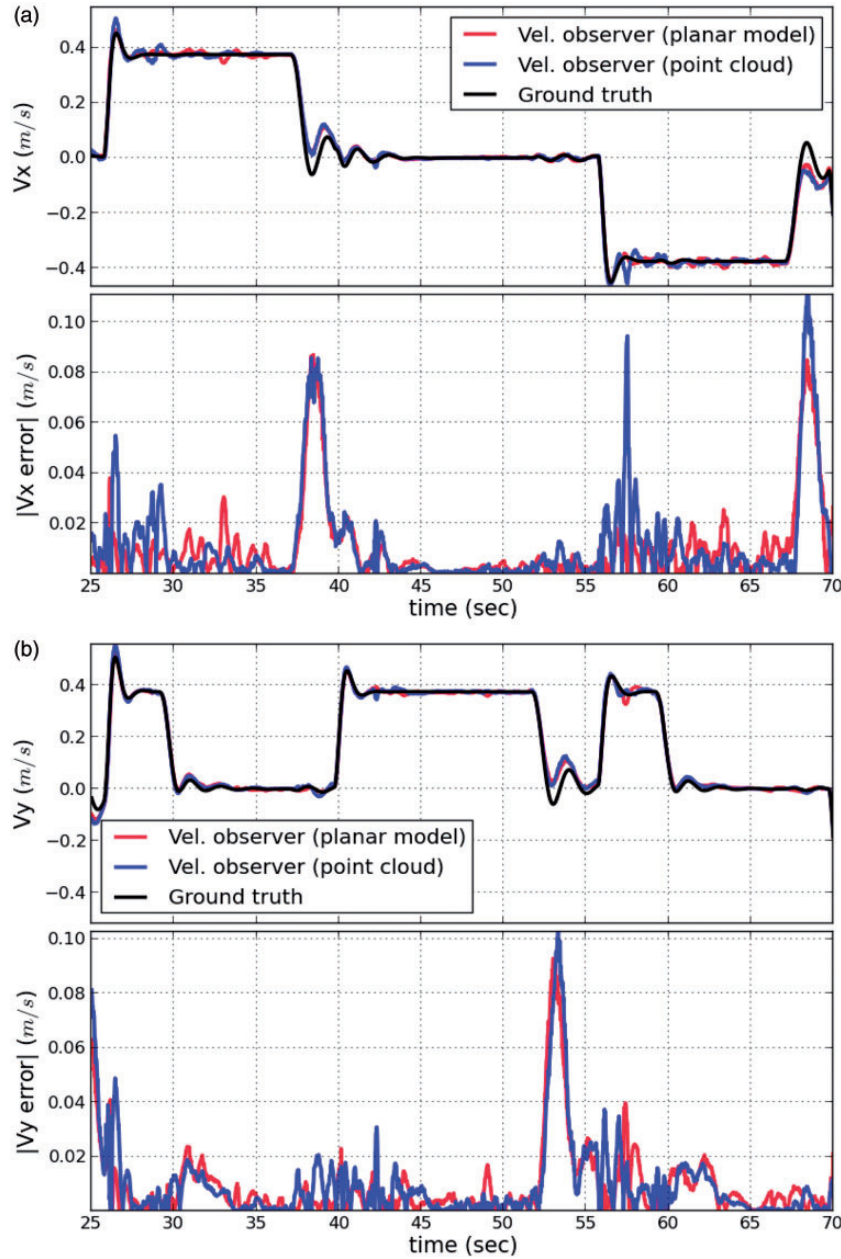


Figure 24. Horizontal velocity estimation results for the observer from equation (34). (a) x-component (V_x) and (b) y-component (V_y).

Furthermore, the good quality of the position estimates from the scan registration allows the velocity errors to remain below $10 \frac{\text{cm}}{\text{s}}$.

Vertical velocity estimation results

Finally, the observer from equation (35) was used to recover vertical velocity estimates from the barometer and accelerometer readings. As before, the observer

gains were chosen as $(k_z, k_{v_z}) = (6.4, 16)$. Figure 25(a) shows the estimation results without barometer drift. As can be seen, the vertical velocity estimates are sufficiently accurate without the need of the laser estimates, as they remain below $1.5 \frac{\text{cm}}{\text{s}}$. Then, the velocity estimates in the presence of barometer drift are shown in Figure 25(b). With respect to the previous case, the estimation error slightly increases but remains within acceptable levels.

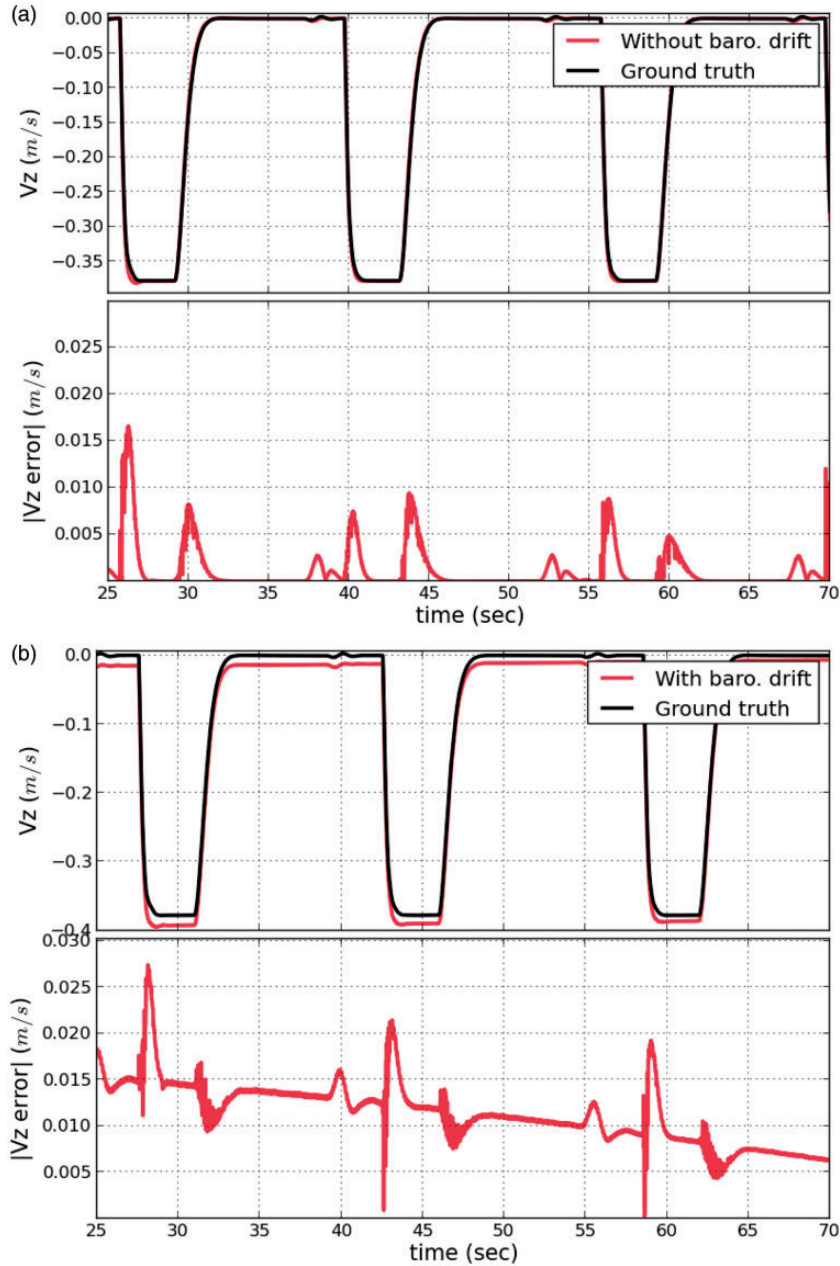


Figure 25. Vertical velocity estimates (V_z) obtained by fusing barometer and IMU measurements. (a) Without barometer drift and (b) with barometer drift.

Conclusions and future work

In this article we have presented a methodology to recover complete 3D pose estimates in electric tower inspection tasks with MAVs, using a sensor set-up consisting of a 2D LiDAR, a barometer sensor and an IMU. First, we addressed 2D local pose estimation using uniquely the laser range measurements. Basic geometric knowledge of the tower was used to extract the

notable features captured in the individual laser scans, which were then used to track the cross-sections and to estimate their previously unknown dimensions. Simulations yielded satisfying results under simple flight conditions, but the assumptions used by this approach proved too restrictive for general inspection tasks. It was shown that this tracking method could instead be used with additional sensing to model the tower. This was tested on data acquired from real flights, and results were presented for a partial point cloud

reconstruction of the tower's body and a simplified planar representation derived from the dimensions estimated on-flight. The inspection task was thus divided into two steps, tower modelling and pose estimation.

Then, we focused on 3D local pose estimation using the complete sensor set-up, which was divided into three components. At the lowest level, a non-linear observer formulation to estimate the roll and pitch angles from the accelerometer and gyrometer measurements was presented. A gain scheduling approach to adapt the observer to changing flight dynamics was also introduced. Then, the four remaining states were determined from the laser scans with two proposed implementations of the ICP algorithm. The first approach consisted in aligning the 2D laser scans to a 3D point cloud reconstruction of the tower, and the second approach relied instead on a simplified planar representation and a projection-based matching strategy. In both cases, the registration process was carried out in 3D and aided by the attitude estimates from IMU measurements, which allowed recovering altitude information. Lastly, a third component fused the barometer measurements and the altitude estimates from the scan registration. This simple formulation allowed estimating the unknown barometer drift in the process. Each of these components was validated in simulations. When combined, they showed satisfying results in terms of accuracy and computation time.

Finally, velocity estimation was achieved with simple feedback observers to exploit the MAV's dynamics. On the one hand, the pose estimates were fused with inertial measurements to recover horizontal velocity estimates. On the other hand, the barometer measurements were fused with accelerometer measurements to recover the vertical velocity component. Simulations were also used to validate the efficiency of these estimations.

An immediate continuation of this work includes introducing the pose and velocity estimates in the feedback control loop to stabilize the MAV's position. We are also interested in conducting further experimental validations of the methods proposed in this work. Since this study was limited to electric tower bodies with rectangular cross-sections, it would also result interesting to extend the methodology to the complete structure, including the head of the tower, and to more complex tower geometries.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

1. Matikainen L, Lehtomäki M, Ahokas E, et al. Remote sensing methods for power line corridor surveys. *ISPRS J Photogramm Remote Sens* 2016; 119: 10–31.
2. Li Z, Bruggemann T, Ford J, et al. Toward automated power line corridor monitoring using advanced aircraft control and multisource feature fusion. *J Field Robot* 2012; 29: 4–24.
3. Ussyshkin R, Theriault L, Sitar M, et al. Advantages of airborne lidar technology in power line asset management. In: *IEEE International workshop on multi-platform/multi-sensor remote sensing and mapping (M2RSM)*, pp.1–5, Xiamen, China, 10–12 January 2011. IEEE.
4. McLaughlin R. Extracting transmission lines from airborne LIDAR data. *IEEE Geosci Remote Sens Lett* 2006; 3: 222–226.
5. Sohn G, Jwa Y and Kim HB. Automatic powerline scene classification and reconstruction using airborne lidar data. *ISPRS Ann Photogramm Remote Sens Spat Inf Sci* 2012; 13: 28.
6. Li Q, Chen Z and Hu Q. A model-driven approach for 3D modeling of pylon from airborne LiDAR data. *Remote Sens* 2015; 7: 11501–11524.
7. Montambault S, Beaudry J, Toussaint K, et al. On the application of VTOL UAVs to the inspection of power utility assets. In: *IEEE First international conference on applied robotics for the power industry (CARPI)*, pp.1–7, Montreal, Canada, 5–7 October 2010. IEEE.
8. Ktrašnik J, Pernuš F and Likar B. A survey of mobile robots for distribution power line inspection. *IEEE Trans Power Deliv* 2010; 25: 485–493.
9. Luque-Vega L, Castillo-Toledo B, Loukianov A, et al. Power line inspection via an unmanned aerial system based on the quadrotor helicopter. In: *IEEE Mediterranean electro-technical conference (MELECON)*, pp.393–397, Beirut, Lebanon, 13–16 April 2014. IEEE.
10. Jones D. Power line inspection – a UAV concept. In: *The IEE forum on autonomous systems*. pp. 8–pp, London, UK, 28–29 November 2005. IET.
11. Golightly I and Jones D. Visual control of an unmanned aerial vehicle for power line inspection. In: *Proceedings of IEEE international conference on advanced robotics (ICAR)*, pp.288–295, Seattle, USA, 18–20 July 2005. IEEE.
12. Kuhnert K and Kuhnert L. Light-weight sensor package for precision 3D measurement with micro UAVs eg power-line monitoring. *ISPRS Int Arch Photogramm Remote Sens Spatial Inform Sci* 2013; 1: 235–240.
13. Richard PL, Pouliot N and Montambault S. Introduction of a lidar-based obstacle detection system on the linescout power line robot. In: *IEEE/ASME international conference on advanced intelligent mechatronics (AIM)*, pp.1734–1740, Besancon, France, 8–11 July 2014. IEEE.
14. Bachrach A, Prentice S, He R, et al. RANGE robust autonomous navigation in GPS-denied environments. *J Field Robot* 2011; 28: 644–666.

15. Shen S, Michael N and Kumar V. Autonomous multi-floor indoor navigation with a computationally constrained micro aerial vehicle. In: *IEEE international conference on robotics and automation (ICRA)*, pp.20–25, Shanghai, China, 9–13 May 2011. IEEE.
16. Grzonka S, Grisetti G and Burgard W. A fully autonomous indoor quadrotor. *IEEE Trans Robot* 2012; 28: 90–100.
17. Dryanovski I, Valenti R and Xiao J. An open-source navigation system for micro aerial vehicles. *Auton Robots* 2013; 34: 177–188.
18. Sa I and Corke P. System identification, estimation and control for a cost effective open-source quadcopter. In: *IEEE international conference on robotics and automation (ICRA)*, pp.2202–2209, Saint Paul, USA, 14–18 May 2012. IEEE.
19. Guo B, Huang X, Li Q, et al. A stochastic geometry method for pylon reconstruction from airborne lidar data. *Remote Sens* 2016; 8: 243.
20. Kohlbrecher S, Von Stryk O, Meyer J, et al. A flexible and scalable SLAM system with full 3D motion estimation. In *IEEE international symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp.155–160, Kyoto, Japan, 1–5 November 2011. IEEE.
21. Scherer S, Rehder J, Achar S, et al. River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Auton Robots* 2012; 33: 189–214.
22. Droschel D, Stückler J and Behnke S. Local multi-resolution surfel grids for mav motion estimation and 3D mapping. In: *Intelligent autonomous systems 13*, 2016, pp.429–442, Padova, Italy, 15–18 July 2014. Springer.
23. Zhang J and Singh S. Loam: Lidar odometry and mapping in real-time. In: *Proceedings of Robotics: science and systems*. Berkeley, USA, 12–16 July 2014.
24. Besl PJ and McKay ND. Method for registration of 3-D shapes. In: *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 2, pp. 239–256, February 1992. IEEE.
25. Olson EB. Real-time correlative scan matching. In: *IEEE international conference on robotics and automation (ICRA)*, pp.4387–4393, Kobe, Japan, 12–17 May 2009. IEEE.
26. Rusinkiewicz S and Levoy M. Efficient variants of the ICP algorithm. In: *IEEE Proceedings of third international conference on 3-D digital imaging and modeling*, pp.145–152, Quebec City, Canada, 28 May–1 June 2001. IEEE.
27. Yang J, Li H, Campbell D, et al. Go-ICP: a globally optimal solution to 3D ICP point-set registration. *IEEE Trans Patt Anal Mach Intell* 2016; 38: 2241–2254.
28. Zaliva V and Franchetti F. Barometric and GPS altitude sensor fusion. In: *IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp.7525–7529, Florence, Italy, 4–9 May 2014. IEEE.
29. Sabatini A and Genovese V. A sensor fusion method for tracking vertical velocity and height based on inertial and barometric altimeter measurements. *MDPI Sensors* 2014; 14: 13324–13347.
30. Son Y and Oh S. A barometer-IMU fusion method for vertical velocity and height estimation. In: *IEEE sensors*, pp.1–4, Busan, South Korea, 1–4 November 2015. IEEE.
31. Tanigawa M, Luinge H, Schipper L, et al. Drift-free dynamic height sensor using MEMS IMU aided by MEMS pressure sensor. In: *Workshop on positioning navigation and communication*, pp.191–196, Hannover, Germany, 27 March 2008. IEEE.
32. Gsior P, Bondyra A, Gardecki S, et al. Robust estimation algorithm of altitude and vertical velocity for multirotor UAVs. In: *IEEE international conference on methods and models in automation and robotics (MMAR)*, pp.714–719, Miedzydroje, Poland, 29 August–1 September 2016. IEEE.
33. Mahony R, Kumar V and Corke P. Multirotor aerial vehicles: modeling, estimation, and control of quadrotor. *IEEE Robot Autom Mag* 2012; 19: 20–32.
34. Hoffmann G, Huang H, Waslander S, et al. Quadrotor helicopter flight dynamics and control: theory and experiment. In: *Proc. of the AIAA guidance, navigation, and control conference*, volume 2, Hilton Head, USA, 20–23 August 2007. AIAA.
35. Martin P and Salan E. The true role of accelerometer feedback in quadrotor control. In: *IEEE international conference on robotics and automation (ICRA)*, pp.1623–1629, Anchorage, USA, 3–7 May 2010. IEEE.
36. Yoo T, Hong S, Yoon H, et al. Gain-scheduled complementary filter design for a MEMS based attitude and heading reference system. *Sensors* 2011; 11: 3816–3830.
37. Rehlinger H and Hu X. Drift-free attitude estimation for accelerated rigid bodies. *Automatica* 2004; 40: 653–659.
38. Zhao H and Wang Z. Motion measurement using inertial sensors, ultrasonic sensors, and magnetometers with extended Kalman filter for data fusion. *IEEE Sens J* 2012; 12: 943–953.
39. Lynen S, Achtelik MW, Weiss S, et al. A robust and modular multi-sensor fusion approach applied to MAV navigation. In: *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp.3923–3929, Tokyo, Japan, 3–7 November 2013. IEEE.
40. Mahony R, Hamel T and Pflimlin JM. Nonlinear complementary filters on the special orthogonal group. *IEEE Trans Autom Control* 2008; 53: 1203–1218.
41. Tayebi A, McGilvray S, Roberts A, et al. Attitude estimation and stabilization of a rigid body using low-cost sensors. In: *IEEE conference on decision and control*, pp.6424–6429, New Orleans, USA, 12–14 December 2007. IEEE.
42. Koenig N and Howard A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In: *IEEE international conference on intelligent robots and systems (IROS)*, volume 3, pp.2149–2154, Sendai, Japan, 28 September–2 October 2004. IEEE.
43. Quigley M, Conley K, Gerkey B, et al. ROS: an open-source robot operating system. In: *ICRA workshop on open source software*, volume 3, p.5, Kobe, Japan, 17 May 2009. IEEE.
44. Meyer J, Sendobry A, Kohlbrecher S, et al. Comprehensive simulation of quadrotor UAVs using ROS and Gazebo. In: *International conference on simulation, modeling, and programming for autonomous robots (SIMPAR)*, pp.400–411, Tsukuba, Japan, 5–8 November 2012. Springer.

45. Bogdan Rusu R and Cousins S. 3D is here: point cloud library (PCL). In: *IEEE international conference on robotics and automation (ICRA)*, pp.1-4, Shanghai, China, 9-13 May 2011. IEEE.
46. Fischler MA and Bolles RC. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun ACM* 1981; 24: 381–395.
47. Gander W and Hrebicek J (eds) *Solving problems in scientific computing using Maple and Matlab®*. 4th ed. Berlin Heidelberg: Springer-Verlag, 2004.
48. Fitzgibbon A. Robust registration of 2D and 3D point sets. *Image Vis Comput* 2003; 21: 1145–1153.
49. Blais G and Levine MD. Registering multiview range data to create 3D computer objects. *IEEE Trans Patt Anal Mach Intell* 1995; 17: 820–824.
50. Park SY and Subbarao M. A fast point-to-tangent plane technique for multi-view registration. In: *IEEE international conference on 3-D digital imaging and modeling (3DIM)*, pp.276–283, Banff, Canada, 6-10 October 2003. IEEE.
51. Valenti R, Dryanovski I and Xiao J. Keeping a good attitude: a quaternion-based orientation filter for IMUs and MARGs. *Sensors* 2015; 15: 19302–19330.

Appendix I: Stability analysis and gain tuning of the altitude observer

To analyse the stability of our proposed altitude observer formulation from equation (19), we first deduce error dynamics of the system. Modelling the barometer measurements as $z_{\text{baro}} = z + b_z$ and the laser estimates as $z_{\text{laser}} = z$, and substituting in equation (20), one obtains

$$\ddot{z}_n = z - \lambda_n \tilde{b}_z, \quad 0 \leq \lambda_n \leq 1, \quad n = 1, 2 \quad (37)$$

where $\tilde{b}_z = \hat{b}_z - b_z$ is the bias estimation error. Substituting this in equation (19), and subtracting the vertical dynamics from equation (17), one obtains the error dynamics of the system as

$$\begin{cases} \ddot{\tilde{z}} = -k_z(\tilde{z} + \lambda_1 \tilde{b}_z) \\ \ddot{\tilde{b}_z} = -k_{b_z}(\tilde{z} + \lambda_2 \tilde{b}_z) \end{cases} \quad (38)$$

where $\tilde{z} = \hat{z} - z$. In matrix form, this is expressed as

$$\begin{bmatrix} \ddot{\tilde{z}} \\ \ddot{\tilde{b}_z} \end{bmatrix} = \begin{bmatrix} -k_z & -\lambda_1 k_z \\ -k_{b_z} & -\lambda_2 k_{b_z} \end{bmatrix} \begin{bmatrix} \tilde{z} \\ \tilde{b}_z \end{bmatrix} = \tilde{\mathbf{A}} \begin{bmatrix} \tilde{z} \\ \tilde{b}_z \end{bmatrix} \quad (39)$$

Stability analysis follows, by analysing the roots of the characteristic polynomial of equation (39), obtained from solving $\det(s\mathbf{I} - \tilde{\mathbf{A}}) = 0$. This results in

$$s^2 + (\lambda_2 k_{b_z} + k_z)s + k_{b_z} k_z (\lambda_2 - \lambda_1) = 0, \quad \lambda_1 \neq \lambda_2 \quad (40)$$

where the $\lambda_1 \neq \lambda_2$ condition avoids a null constant term in the polynomial. Then, exponential convergence is guaranteed if the two roots of the characteristic polynomial have negative real parts. This can be achieved with a simple pole placement approach. Recalling the characteristic polynomial for a second-order system

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0 \quad \zeta, \omega_n > 0 \quad (41)$$

where the damping ratio ζ and the natural frequency ω_n define the closed-loop poles, which have negative real part if $\zeta, \omega_n > 0$. The observer gains are then determined by comparing the coefficients of both polynomials. This results in two cases depending on the value of λ_2 .

On the one hand, if $\lambda_2 = 0$, then solving by substitution one obtains

$$\begin{cases} k_z = 2\zeta\omega_n \\ k_{b_z} = -\frac{\omega_n^2}{\lambda_1 k_z}, \quad \lambda_2 = 0, 0 < \lambda_1 \leq 1 \end{cases} \quad (42)$$

In this simple case, to determine (k_z, k_{b_z}) , one must first choose the closed-loop poles for the desired system response, which defines the value of ζ and ω_n , and then set λ_1 to the desired value. This was the case considered in the simulations, where $\zeta = 1.1$ (overdamped response), $\omega_n = 3.0$ and $\lambda_1 = 1$ lead to $(k_z, k_{b_z}) = (6.6, -1.36)$.

On the other hand, if $\lambda_2 > 0$, this leads to a quadratic expression for k_z and k_{b_z} , obtaining

$$\begin{cases} k_z = \frac{2\zeta\omega_n \mp \sqrt{(2\zeta\omega_n)^2 - \frac{4\lambda_2\omega_n^2}{\lambda_2 - \lambda_1}}}{2} \\ k_{b_z} = \frac{2\zeta\omega_n \pm \sqrt{(2\zeta\omega_n)^2 - \frac{4\lambda_2\omega_n^2}{\lambda_2 - \lambda_1}}}{2\lambda_2}, \quad \lambda_2 > 0, \lambda_1 \neq \lambda_2 \end{cases} \quad (43)$$

Then, to avoid complex gains, the discriminant Δ must be nonnegative. That is

$$\Delta = (2\zeta\omega_n)^2 - \frac{4\lambda_2\omega_n^2}{\lambda_2 - \lambda_1} \geq 0 \quad (44)$$

leading to the following inequality

$$\zeta^2 \geq \frac{\lambda_2}{\lambda_2 - \lambda_1} \quad (45)$$

which conditions the values of ζ and λ_n . In this case, a simple way of tuning the gains is to first choose the closed-loop poles, obtaining ζ and ω_n , then set λ_2 to the desired value and finally set λ_1 ensuring that equation (45) holds.

Vision-based dynamic target trajectory and ego-motion estimation using incremental light bundle adjustment

International Journal of Micro Air Vehicles
2018, Vol. 10(2) 157–170
© The Author(s) 2018
DOI: 10.1177/1756829318756354
journals.sagepub.com/home/mav


Michael Chojnacki¹ and Vadim Indelman²

Abstract

This paper presents a vision-based, computationally efficient method for simultaneous robot motion estimation and dynamic target tracking while operating in GPS-denied unknown or uncertain environments. While numerous vision-based approaches are able to achieve simultaneous ego-motion estimation along with detection and tracking of moving objects, many of them require performing a bundle adjustment optimization, which involves the estimation of the 3D points observed in the process. One of the main concerns in robotics applications is the computational effort required to sustain extended operation. Considering applications for which the primary interest is highly accurate online navigation rather than mapping, the number of involved variables can be considerably reduced by avoiding the explicit 3D structure reconstruction and consequently save processing time. We take advantage of the light bundle adjustment method, which allows for ego-motion calculation without the need for 3D points online reconstruction, and thus, to significantly reduce computational time compared to bundle adjustment. The proposed method integrates the target tracking problem into the light bundle adjustment framework, yielding a simultaneous ego-motion estimation and tracking process, in which the target is the only explicitly online reconstructed 3D point. Our approach is compared to bundle adjustment with target tracking in terms of accuracy and computational complexity, using simulated aerial scenarios and real-imagery experiments.

Keywords

Simultaneous localization and mapping, bundle adjustment, navigation, computer vision, target tracking

Received 18 May 2017; accepted 4 December 2017

Introduction

Ego-motion estimation and target tracking are core capabilities required in a wide range of applications. While motion estimation is essential to numerous robotics tasks such as autonomous navigation^{1,2,3,4} and augmented reality,^{5,6} target tracking has been essential, amongst others, for video surveillance⁷ and for military purposes.⁸ Although researched for decades, target tracking methods have mostly assumed a known or highly predictable sensor location. Recent robotics applications such as autonomous aerial urban surveillance⁹ or indoor navigation require the ability to track dynamic objects from platforms while moving in unknown or uncertain environments. The ability to simultaneously solve the ego-motion and target tracking problems becomes therefore an important task. Furthermore, attention has grown for cases

in which external localization systems (e.g. GPS) are unavailable and the estimation process must be performed using on-board sensors only. In particular, the capability to perform those tasks based on vision sensors has become of great interest in the past two decades, mostly thanks to the ever-growing advantages these sensors present.¹⁰

Vision-based ego-motion estimation is typically performed as part of a process known as bundle adjustment (BA) in computer vision, or simultaneous

¹Technion Autonomous Systems Program (TASP), Technion, Haifa, Israel

²Faculty of Aerospace Engineering, Technion, Haifa, Israel

Corresponding author:

Michael Chojnacki, Technion Autonomous Systems Program (TASP), Technion, 32000 Haifa, Israel.
Email: michaelchoch@gmail.com



localization and mapping (SLAM) in robotics, where the differences between the actual and the predicted image observations are minimized. Therefore, the combined process of SLAM and tracking of a moving object usually involves an optimization over the camera's motion states, the target's navigation states, and the observed structure (3D landmarks). This optimization is performed incrementally as new information and variables are added to the process, constantly increasing the computational complexity of the problem. One of the main challenges in extended operation is thus keeping computational efforts to a minimum despite the growing number of variables. However, many robotics applications do not require actual online mapping of the environment. Avoiding this expensive task would therefore be beneficial in terms of processing time.

This work presents a computationally efficient approach for simultaneous camera ego-motion estimation and target tracking, while operating in unknown or uncertain GPS-deprived environments. Our focus lies on robotic applications for which online 3D structure reconstruction is of no interest, although recovering the latter offline from optimized camera poses is always possible.¹¹ We propose to take advantage of the recently developed incremental light bundle adjustment (iLBA)^{11–13} framework, which uses multi-view constraints to algebraically eliminate the (static) 3D points from the optimization, therefore allowing the dynamic target to become the only explicitly reconstructed 3D point in the process. The reduced number of variables involved in the optimization allows therefore for substantial savings in computational efforts. Incremental smoothing and mapping (iSAM)¹⁴ technique is applied to re-use calculations, allowing to further reduce running time, in a similar fashion to the static-scene-oriented iLBA approach. We demonstrate, using simulations on synthetic datasets and real-imagery experiments, that while our methods provide similar levels of accuracy to full BA with target tracking, they compare favorably in terms of computational complexity.

The simultaneous ego-motion and dynamic object tracking relate to numerous works on SLAM and target tracking, both individually and combined. Early approaches used the extended Kalman filter (EKF) to solve the SLAM problem^{15,16} but were eventually overtaken by other techniques due to their quadratic computational complexity, which limits them to relatively small environments or to relatively small state vectors. Numerous SLAM methods have been proposed to overcome computational complexity, for example, by exploiting the sparsity of the involved matrices,^{17,18} or by approximating the full problem with a reduced non-linear system.¹⁹ A more recent

technique, used in the frame of this work, performs incremental smoothing¹⁴ to recover the solution while recalculating only part of the variables at each optimization step and allows for a significant reduction of the computational cost. Still, full BA methods involve the reconstruction of the 3D observed structure, increasing unnecessarily the number of estimated variables in cases online mapping is of no interest. Several “structure-less” BA approaches have been developed, where the optimization satisfies constraints which do not involve 3D structure reconstruction. Rodriguez et al.²⁰ use epipolar constraints between pairs of views, while Steffen et al.²¹ utilize trifocal tensor constraints. The recently developed LBA method,¹² used in this work, applies two kinds of multi-view constraints: the two-view and three-view constraints. Pose-SLAM techniques^{22,23} avoid explicit mapping by maintaining the camera trajectory as a sparse graph of relative pose constraints, which are calculated using the landmarks in a separate process. In contrast to standard Pose-SLAM, LBA formulates multi-view geometry constraints for each feature match, thereby avoiding to rely on the uncertainty of the abovementioned separate process.

The target tracking problem, referred more generally as *detection and tracking of moving objects* (DTMO)²⁴ in the robotics literature, has been extensively studied for several decades.^{25,26} The combined SLAM and DTMO problem, which is assessed in our work, has attracted considerable attention in the recent years, mostly in order to improve SLAM accuracy, which can be greatly degraded by the presence of dynamic objects in the environment, if the latter is considered as static.²⁷ The first mathematical framework to the combined process of simultaneous localization, mapping, and moving object tracking (SLAMMOT) was presented by Wang,²⁸ where the problem is decomposed into two separate estimators, one for the SLAM problem given the static landmarks and another for the tracking problem. Occupancy grid-based approaches were proposed later by Vu et al.²⁹ and Vu,¹ where SLAM was solved by calculating the maximum likelihood of occupancy grid maps. Ortega³⁰ introduced a geometric and probabilistic approach to the vision-based SLAMMOT problem, providing a comparison between the different kinds of optimization methods, while Hahnel et al.³¹ used sampled-based joint probabilistic data association filter to track people and occupancy grids for static landmarks. An extensive overview of the literature concerning SLAM and DTMO is presented in Pancham et al.³²

The rest of this paper is structured as follows: First, we formulate the simultaneous ego-motion estimation and moving object tracking problem. Next, we review the LBA method, which is extended to address the

mentioned problem. Then, we present experimental results, comparing our method with full BA in terms of computation time and accuracy. Finally, we conclude and share thoughts about further possible developments.

Problem formulation and notations

We consider a scenario where a monocular camera mounted on a mobile robot is tracking a dynamic target while operating in a GPS-deprived unknown environment.

The BA problem

The process of determining the camera poses and the stationary 3D structure given measurements is called BA or SLAM. Let x_k represent the camera pose (i.e. 6DOF position and orientation) at time step t_k , and denote all such states up to that time by $X_k \doteq \{x_0 \dots x_k\}$. We also use $L_k \doteq \{l_1 \dots l_n\}$ and $Z_k \doteq \{z_0 \dots z_k\}$ to represent, respectively, all the n landmarks observed by time t_k , and the corresponding sensor observations. Here, for each time index $i \in [0, k]$, z_i corresponds to all image observations obtained at time t_i . In particular, we use the notation z_i^j to denote an observation of the j th landmark at time t_i .

Using probabilistic representation, the BA problem can be expressed by the joint pdf

$$P(X_k, L_k | Z_k) \quad (1)$$

Using Bayes' rule, the general recursive Bayesian formula for BA can be derived as³³

$$P(X_k, L_k | Z_k) \propto \text{priors} \cdot \prod_{i=1}^k \prod_{j \in \mathcal{M}_i} p(z_i^j | x_i, l_j) \quad (2)$$

where \mathcal{M}_i is the set of landmarks observed at time index i and *priors* represent prior information on the estimated variables.

Considering a standard pinhole camera, the corresponding observation model can be defined as³⁴

$$z_i^j = \text{proj}(x_i, l_j) + v_{ij} \quad (3)$$

where $\text{proj}(\cdot)$ is the projection operator³⁴ and $v_{ij} \sim \mathcal{N}(0, \Sigma_v)$ is a zero-mean white noise with measurement covariance Σ_v . Under Gaussian distribution assumption, the likelihood of the perception measurement can be expressed as

$$p(z | x, l) \doteq \frac{1}{\sqrt{|2\pi\Sigma_v|}} \exp\left(-\frac{1}{2} \|z - \text{proj}(x, l)\|_{\Sigma_v}^2\right) \quad (4)$$

where $\|a\|_{\Sigma}^2 \doteq a^T \Sigma^{-1} a$ is the squared Mahalanobis distance with the measurement covariance matrix Σ . We assume camera calibration is known; otherwise, the uncertain calibration parameters could be incorporated into the optimization framework as well.

Solving the BA problem would therefore consist in calculating the maximum a posteriori estimate over the joint pdf, defined as

$$X_k^*, L_k^* = \arg \max_{X_k, L_k} P(X_k, L_k | Z_k) \quad (5)$$

Due to the monotonic characteristics of the logarithmic function, calculating the MAP estimate X_k^*, L_k^* becomes equivalent to minimizing the negative log-likelihood of the BA pdf 1

$$X_k^*, L_k^* = \arg \min_{X_k, L_k} -\log P(X_k, L_k | Z_k) \quad (6)$$

This leads to a non-linear least-squares optimization, where the cost function

$$J_{BA}(X_k, L_k) = \sum_i \sum_{j \in \mathcal{M}_i} \|z_i^j - \text{proj}(x_i, l_j)\|_{\Sigma}^2 \quad (7)$$

is to be minimized. Note that, to avoid clutter, the prior terms are not explicitly shown in equation (7).

BA and target tracking

We investigate scenarios in which a dynamic target is tracked by the camera. Based on the camera's observations of the target, we seek to estimate its trajectory and velocity over time. We assume the target moves randomly; however, its motion is assumed to follow a known stochastic kinematic model (e.g. constant velocity or constant acceleration).

Let y_k represent the target state at time step t_k , defined generally as

$$y_k \doteq [y_{T_k} \, d_{T_k}]^T = [x_{T_k}, y_{T_k}, z_{T_k}, \dot{x}_{T_k}, \dot{y}_{T_k}, \dot{z}_{T_k}, \dots]^T \quad (8)$$

where y_{T_k} denotes the target's tri-dimensional position and d_{T_k} its higher order time derivatives required to accommodate the assumed motion model. In the frame of this work, we focus on the target's position

and velocity. y_k is therefore a six element vector defined as

$$y_k = \begin{bmatrix} y_{T_k} \\ \dot{y}_{T_k} \end{bmatrix} \in \mathbb{R}^{6 \times 1} \quad (9)$$

We denote $Y_k \doteq \{y_0 \dots y_k\}$ the set of all target's states up to time step t_k .

Assuming a known Markovian motion model for the target, which likelihood is represented by $p(y_i|y_{i-1})$, we define a joint pdf for the random variables involved in the considered problem, given all information thus far, as

$$P(X_k, Y_k, L_k|Z_k) \propto \text{priors} \cdot \prod_{i=1}^k \left(p(y_i|y_{i-1}) p(z_i^{y_i}|x_i, y_i) \prod_{j \in M_i} p(z_i^j|x_i, l_j) \right) \quad (10)$$

where $z_i^{y_i}$ denotes the observation of the target by the i th camera and $p(z_i^{y_i}|x_i, y_i)$ refers to the observation model described in equation (3). M_i is the set of landmarks observed at time index i and we consider $\text{priors} = p(x_0)p(y_0)$ as given information.

In this work, as in many robotics applications, we consider a constant velocity model,³⁵ characterized by the equation

$$\ddot{y}(t) = \tilde{w}(t) \quad (11)$$

where $\tilde{w}(t)$ is a continuous time zero-mean white noise representing the slight velocity changes from its actual value.

The target state linear continuous propagation is generally noted as $\dot{y}(t) = Ay(t) + Dw(t)$, where $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ and $D = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, or under its discrete form

$$y_{k+1} = \Phi_k y_k + G_k w_k \quad (12)$$

where G_k is the process noise Jacobian defined as $G_k = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in \mathbb{R}^{6 \times 3}$ and Φ_k is the state transition matrix and is defined as $\Phi_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{6 \times 6}$ with $\Delta t \doteq t_{k+1} - t_k$. The discrete-time process noise $w_k \sim \mathcal{N}$

$(0, \Sigma_w)$ relates to the continuous-time one as $w_k = \int_0^{\Delta t} e^{A(\Delta t-\tau)} D \tilde{w}(k\Delta t + \tau) d\tau$. Under Gaussian distribution assumption, the motion model likelihood is therefore expressed

$$p(y_{k+1}|y_k) \doteq \frac{1}{\sqrt{|2\pi\Sigma_{mm}|}} \exp\left(-\frac{1}{2} \|y_{k+1} - \Phi_k y_k\|_{\Sigma_{mm}}^2\right) \quad (13)$$

where $\Sigma_{mm} \doteq G\Sigma_w G^T$.

Finally, solving the combined BA and target state estimation process consists in calculating the MAP estimate over the joint pdf from equation (10)

$$X_k^*, Y_k^*, L_k^* = \arg \max_{X_k, Y_k, L_k} P(X_k, Y_k, L_k|Z_k) \quad (14)$$

Factor graph representation

As mentioned earlier, the factorization of the joint pdf described in equation (10) can be represented using a factor graph,³⁶ which will be used later to efficiently solve the optimization problem using incremental inference. Using the same observation (equation (3)) and motion (equation (12)) models, this pdf is expressed in factor graph notation as

$$P(X_k, Y_k, L_k|Z_k) \propto \text{priors} \times \prod_{i=1}^k \left(f_{mm}(y_i, y_{i-1}) f_{proj}(x_i, y_i) \prod_{j \in M_i} f_{proj}(x_i, l_j) \right) \quad (15)$$

An illustration expressing the above factorization for a small example is shown in Figure 1. The corresponding factors in equation (15) are straightforwardly defined as follows: The factor $f_{mm}(y_i, y_{i-1})$ corresponds to the target motion model and, referring to equations (12) and (13), is defined as

$$f_{mm}(y_i, y_{i-1}) \doteq \exp\left(-\frac{1}{2} \|y_i - \Phi_{i-1} y_{i-1}\|_{\Sigma_{mm}}^2\right) \quad (16)$$

The projection factors $f_{proj}(x_i, l_j)$ and $f_{proj}(x_i, y_i)$ correspond to the landmarks and target observation models; these factors are defined, respectively, as

$$f_{proj}(x_i, l_j) \doteq \exp\left(-\frac{1}{2} \|z_i^j - \text{proj}(x_i, l_j)\|_{\Sigma_v}^2\right) \quad (17)$$

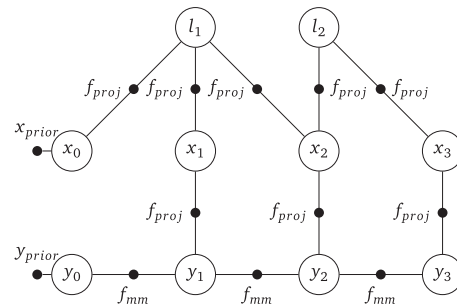


Figure 1. Factor graph representing a factorization of the joint pdf for bundle adjustment with single target tracking.

$$f_{proj}(x_i, y_i) \doteq \exp\left(-\frac{1}{2}\|z_i^{y_i} - proj(x_i, y_i)\|_{\Sigma_v}^2\right) \quad (18)$$

Similarly to the previous section, the MAP estimate is defined as

$$X_k^*, Y_k^*, L_k^* = \arg \max_{X_k, Y_k, L_k} P(X_k, Y_k, L_k | Z_k) \quad (19)$$

and can be efficiently calculated by exploiting the inherent sparse structure of the problem while re-using calculations.

This corresponds to the state of the art where inference is performed over camera poses, landmarks, and target states. Yet, when the primary focus is navigation rather than mapping, explicit estimation of the observed landmarks in an online process is not actually required. Conceptually, estimating only the camera poses and the dynamic target (but not the landmarks) involves less variables to optimize and could be attractive from a computational point of view. In this work, we develop an approach based on this idea.

LBA and dynamic target tracking

BA is a non-linear iterative optimization framework typically applied for estimating camera poses and observed landmarks. In this section, we integrate target tracking to a structure-less BA technique called light bundle adjustment (LBA).¹³ First, we formulate the LBA equations while considering a static scene. These equations are then extended to incorporate the dynamic target tracking problem.

Using factor graph notations, the joint pdf $P(X_k, L_k | Z_k)$, which corresponds to the static problem, can be factorized similarly to equation (15) as

$$P(X_k, L_k | Z_k) \propto \text{priors} \cdot \prod_{i=1}^k \left(\prod_{j \in M_i} f_{proj}(x_i, l_j) \right) \quad (20)$$

where $\text{priors} = p(x_0)p(y_0)$ represents the prior information on the camera and target states.

As mentioned, this works considers robotics applications in which the online reconstruction of the 3D structure is of no interest. One way to avoid explicit estimation of the landmarks in the solution is by marginalizing out the latter from the joint pdf as in

$$P(X_k | Z_k) = \int P(X_k, L_k | Z_k) dL_k \quad (21)$$

However, this involves a series of calculations which, in the case of online operation, could be

penalizing: First, performing the exact marginalization would initially require to solve the full BA problem, including landmarks, before applying a Gaussian approximation to compute the marginal. Secondly, marginalization in the information form involves the expensive calculation of the Schur complement over the variables we wish to keep.²² Moreover, marginalization introduces fill-in, destroying the sparsity of the information matrix.

In contrast, structure-less BA methods approximate the BA cost function, allowing for estimation of the camera poses without involving the reconstruction of the 3D structure.^{20,21} In this work, we use the recently developed LBA approach,^{11,12} which algebraically eliminates the landmarks from the optimization, using multi-view constraints and in particular, three-view constraints.

LBA

LBA allows for reduction of the number of variables involved in the optimization compared to standard BA. By algebraically eliminating the landmarks from the problem, the optimization can be performed over the camera poses only. The key idea is to use geometrical constraints relating three views from which the same landmark is observed.

Considering a set of three overlapping poses k, l and m from which a common landmark is observed, it is possible to derive constraints that relate the three poses while eliminating the landmark.³⁷ These constraints can be formulated as two two-view constraints g_{2v} between two pairs of poses (e.g. (k, l) and (l, m)) and one three-view constraint g_{3v} between the three involved poses.^{37,38} Conceptually, the two-view constraint is equivalent to the epipolar constraint,³⁴ while the three-view constraint relates between the scales of the two translations $t_{k \rightarrow l}$ and $t_{l \rightarrow m}$. Writing down the appropriate projection equations, we get

$$g_{2v}(x_k, x_l, z_k, z_l) = q_k \cdot (t_{k \rightarrow l} \times q_l) \quad (22)$$

$$g_{2v}(x_l, x_m, z_l, z_m) = q_l \cdot (t_{l \rightarrow m} \times q_m) \quad (23)$$

$$\begin{aligned} g_{3v}(x_k, x_l, x_m, z_k, z_l, z_m) \\ = (q_l \times q_k) \cdot (q_m \times t_{l \rightarrow m}) - (q_k \times t_{k \rightarrow l}) \cdot (q_m \times q_l) \end{aligned} \quad (24)$$

$q_i \doteq R_i^T K_i^{-1} z$ for the i th view and image observation z , where K_i is the calibration matrix, R_i represents the rotation matrix from some reference frame to the i th view, and $t_{i \rightarrow j}$ denotes the translation vector from view i to view j , expressed in the global frame.

The resulting probability distribution $P_{LBA}(X|Z)$ can thus be factorized as

$$P_{LBA}(X|Z) \propto \prod_{i=1}^{N_h} f_{2v/3v}(X_i) \quad (25)$$

where $f_{2v/3v}$ represents the involved two- and three-view factors and X_i is the relevant subset of camera poses. Referring to equations (22) to (24), under Gaussian distribution assumption, f_{2v} and f_{3v} are defined as

$$f_{2v}(x_k, x_l) \doteq \exp\left(-\frac{1}{2} \|g_{2v}(x_k, x_l, z_k, z_l)\|_{\Sigma_{2v}}^2\right) \quad (26)$$

and

$$f_{3v}(x_k, x_l, x_m) \doteq \exp\left(-\frac{1}{2} \|g_{3v}(x_k, x_l, x_m, z_k, z_l, z_m)\|_{\Sigma_{3v}}^2\right) \quad (27)$$

which correspond to the likelihoods of the two- and three-views constraints involving x_k and x_l in equation (26) and involving x_k , x_l and x_m in equation (27). The covariances Σ_{2v} and Σ_{3v} are defined as

$$\begin{aligned} \Sigma_{2v} &\doteq (\nabla_{z_k, z_l} g_{2v}) \Sigma (\nabla_{z_k, z_l} g_{2v})^T, \\ \Sigma_{3v} &\doteq (\nabla_{z_k, z_l, z_m} g_{3v}) \Sigma (\nabla_{z_k, z_l, z_m} g_{3v})^T \end{aligned} \quad (28)$$

Figure 2 shows a comparison between the factor graph representation of LBA and standard BA for a small example.

Therefore, rather than optimizing the cost function 7, that involves the camera and landmark states, the optimization is performed on the cost function¹¹

$$J_{LBA}(X) \doteq \sum_{i=1}^{N_h} \|h_i(X_i, Z_i)\|_{\Sigma_i}^2 \quad (29)$$

where $h_i \in \{g_{2v}, g_{3v}\}$ represents a single two- or three-view constraint involving the set of poses X_i and the set of image observations Z_i , N_h being the number of resulting constraints.

Practically, when a landmark is observed by a new view x_k and some earlier views x_l and x_m , a single two-view (between x_k and one of the two other views) and a single three-view constraint are added (between the three views). The reason for not adding the second two-view constraint (between views x_l and x_m) is that this constraint was already added when processing

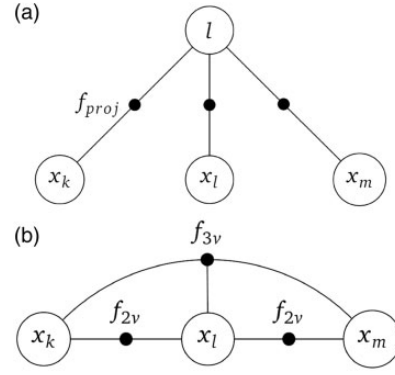


Figure 2. Factor graph representation for a small example including three views x_k , x_l , x_m . (a) The BA problem, where the three views are related to the landmark l with projection factors. (b) The LBA problem, where the landmark l has been eliminated, and the three views are related by two- and three-view constraints.

these past views. In case a landmark is observed by only two views, we add a single two-view constraint.

LBA and dynamic target tracking

In this section, we integrate dynamic target tracking into the LBA framework. As will be shown, the resulting approach provides comparable accuracy for both target tracking and camera pose estimation while significantly reducing running time, compared to an equivalent BA approach.

The idea behind the proposed method is to incorporate the target tracking problem into the LBA framework in order to yield a proxy for the joint pdf $P(X_k, Y_k|Z_k)$ which involves significantly less variables than the joint pdf $P(X_k, Y_k, L_k|Z_k)$, while somewhat avoiding the expensive calculations involved in the marginalization process.¹¹ Indeed, if $X_k \in \mathbb{R}^{M_k \times 1}$, $Y_k \in \mathbb{R}^{N_k \times 1}$ and $L_k \in \mathbb{R}^{O_k \times 1}$, then the amount of variables involved in the optimization is decreased from $M_k + N_k + O_k$ to $M_k + N_k$ only, which would reduce computational complexity (we note that $O_k \gg M_k$ and $O_k \gg N_k$).

We integrate the factors $f_{2v/3v}$ corresponding to the camera poses described in equations (26) and (27) with the target tracking-related factors f_{mm} and f_{proj} defined in equations (16) and (18) to yield the joint pdf $P(X_k, Y_k|Z_k)$ over the relevant states only. The target becomes, therefore, the only 3D point to be estimated in the process

$$P(X_k, Y_k|Z_k) \propto \text{priors} \times \prod_{i=1}^{k-1} \left(f_{mm}(y_i, y_{i-1}) f_{proj}(x_i, y_i) \prod_{j=1}^N f_{2v/3v}(X_j) \right) \quad (30)$$

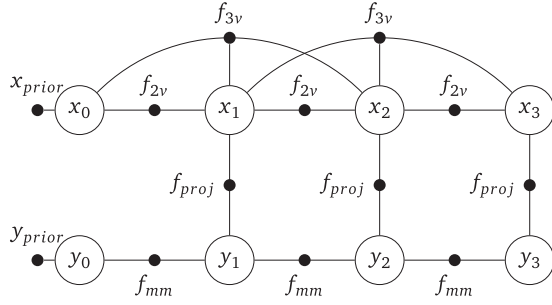


Figure 3. Factor graph representing a factorization of the joint pdf for LBA and target tracking.

where, similar to equation (15), $priors = p(x_0)p(y_0)$ represents the prior information and X_j is the relevant subset of views for the i th frame.

Solving the localization and target tracking problem then corresponds to estimating the MAP

$$X_k^*, Y_k^* = \arg \max_{X_k, Y_k} P(X_k, Y_k | Z_k) \quad (31)$$

which is equivalent to minimizing the cost function

$$\begin{aligned} J(X_k, Y_k) = & \|x_0 - \hat{x}_0\|_{\Sigma_x}^2 + \|y_0 - \hat{y}_0\|_{\Sigma_y}^2 \\ & + \sum_{i=1}^k (\|y_i - \Phi_i y_{i-1}\|_{\Sigma_{mm}}^2 + \|z_i^{y_i} \\ & - proj(x_i, y_i)\|_{\Sigma_y}^2 \\ & + \sum_j^{N_h} \|h_j(X_j, Z_j)\|_{\Sigma_j}^2) \end{aligned} \quad (32)$$

An illustration expressing the above factorization for the same example as in Figure 1 is shown in Figure 3.

LBA and multi-target tracking

The considered problem can be straightforwardly extended to multi-target tracking by integrating the additional targets into the formulation from equation (30). Considering n targets, the corresponding joint pdf can be written

$$\begin{aligned} P(X_k, \bar{Y}_k | Z_k) \propto & priors \\ & \times \prod_{i=1}^{k-1} \left(\prod_{l=1}^n f_{mm}(y_i^l, y_{i-1}^l) \prod_{p \in T_i} f_{proj}(x_i, y_i^p) \prod_{j=1}^N f_{2v/3v}(X_j) \right) \end{aligned} \quad (33)$$

where $\bar{Y}_k = \{Y_k^1, Y_k^2, \dots, Y_k^n\}$ is the set of all targets' states up to time-step t_k and Y_k^n refers to the states of

the n th target up to time-step t_k . We denote T_i the set of targets observed at time-step t_i . Here, we assume the identification of the targets that leave and re-enter the camera's field of view as given. Solving this data-association problem is a challenging task by itself and is outside the scope of this work.

Incremental smoothing

Solving the abovementioned non-linear least square problems is achievable using several optimization methods. Online operation requires this task to be performed efficiently, and therefore, cost-efficient techniques were implemented in this work.

Batch optimization performs factorization of the Jacobian matrix A from scratch each time new variables are added to the problem. In contrast, incremental smoothing updates the problem as new measurements and variables arrive, by directly updating the square root information matrix R and recalculating only the matrix entries that actually change.³⁹ Furthermore, instead of performing batch re-ordering, eliminating the corresponding factor graph into a Bayes tree¹⁴ allows for incremental variable ordering, which keeps the R matrix sparsity at a relatively constant level. Additionally, rather than fully re-linearizing the whole set of variables at a determined point in time, iSAM2 performs fluid re-linearization, which triggers re-linearization of a variable only when the deviation between its current estimate and the linearization point is larger than a defined threshold, set heuristically or as part of a "tuning" process.

Results

We demonstrate the benefits of the proposed method with simulations performed on synthetic datasets and with real-imagery experiments. Experiments were performed considering a downward-facing camera mounted on a flying vehicle, which tracks a single target, for the sake of simplicity. For each scenario, target tracking and ego-pose estimation using LBA and full BA are compared in terms of accuracy and processing time. All experiments were run on an Intel i7-4720HQ quadcore processor with 2.6 GHz clock rate and 8GB of RAM. The methods used for comparison were implemented using the GTSAM library (<https://research.cc.gatech.edu/borg/download>).

Experimental evaluation with synthetic datasets

A series of simulations were performed on synthetic datasets in order to compare our method with full BA technique and to demonstrate its capability in

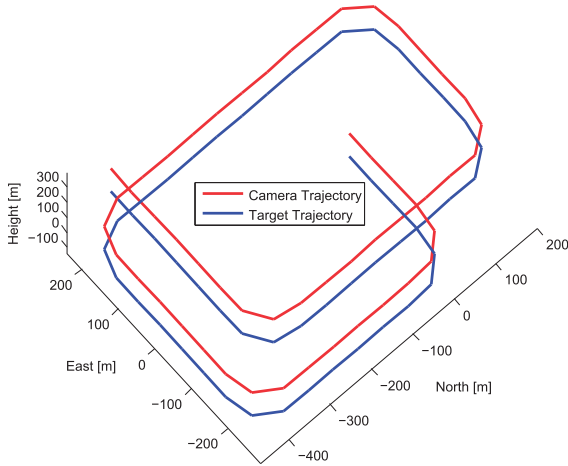


Figure 4. Scenario used for statistical study. Camera and target trajectories are shown in red and blue, respectively. At this scale, ground truth and estimated trajectories are indistinguishable (see Figure 5).

terms of computational performance and estimation accuracy for both camera and target states. We present two types of studies: A statistical performance study on an approximately 3-km long aerial scenario (Figure 4), and a case study in a larger aerial scenario (Figure 6(a)). In both cases, the downward-facing camera operates in GPS-denied environments and occasionally re-visits previously explored locations, providing occasional loop-closure measurements. The priors $p(x_0)$ and $p(y_0)$ are Gaussians with means equal to their initial values, and with $\sigma = 2$ [m] standard deviation. The measurement model assumes an image noise $\sigma = 0.5$ [pix]. The continuous-time system is discretized with time-step $\Delta t = 3$ [s]. Regarding target motion, we use the constant velocity model and assume a zero-mean, white Gaussian noise $\sigma = [30, 30, 0.001]^T$ [m/s]. Here, we constrained the noise on the z axis to prevent divergence, both with LBA and BA, which use data only from a single monocular camera. Addressing this issue would probably require additional information or constraints on the target motion (multi-robot setup, additional sensors, geometric constraints, etc.).

Statistical simulation results

A performance comparison between the proposed method and BA with target tracking is presented in a 45-run Monte-Carlo study. The scenario used in this simulation, shown in Figure 4, contains 52 frames, gathered over ~ 160 s. Loop-closures can be noticed around views 20 and 38. The simulated target takes a similar course on the ground and for the sake of simplicity, stays in the camera's field of view throughout the process. The comparisons presented in Figure 5(a)

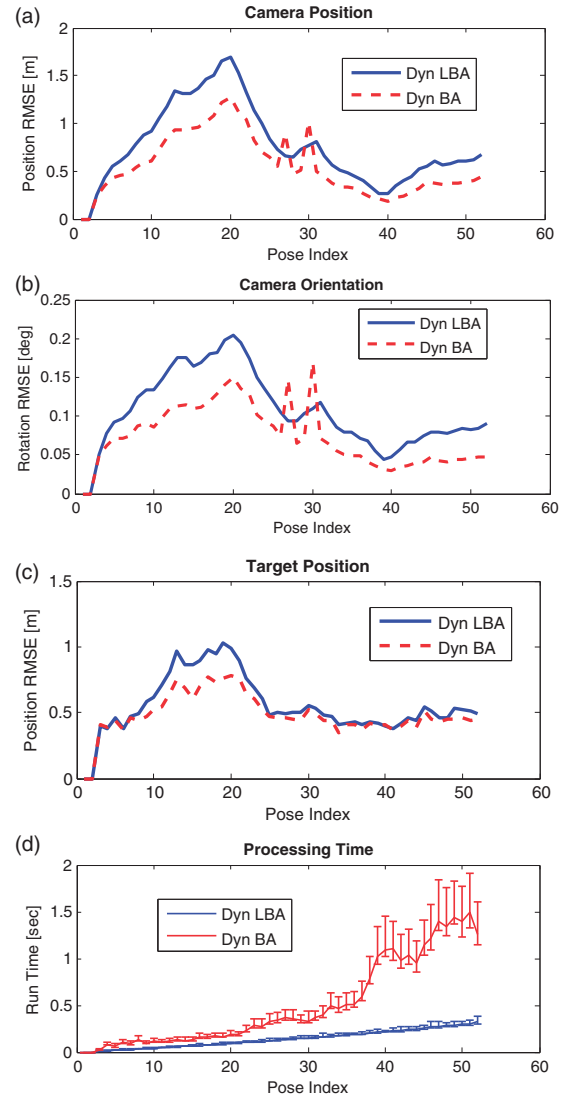


Figure 5. Monte-Carlo study results comparing between the proposed method and full BA with target tracking (a) camera position RMSE; (b) camera orientation RMSE (including close-up); (c) target position RMSE; (d) running time average with lower and upper boundaries. LBA: light bundle adjustment; BA: bundle adjustment; RMSE: root-mean-square error.

to (c) are given in terms of root-mean-square error (RMSE), calculated over the norms of the error vectors. All results refer to incremental estimations, i.e. at each time t_k performance is evaluated given Z_k , which is in particular important for online navigation.

Figure 5(a) and (b) describes the camera incremental position and orientation errors and Figure 5(c) shows the target position error. We observe similar levels of accuracy with the two techniques. The camera pose and target trajectory errors are bounded, with clear negative trend in both the camera and target position errors around view 20, upon loop closure. We note that, in

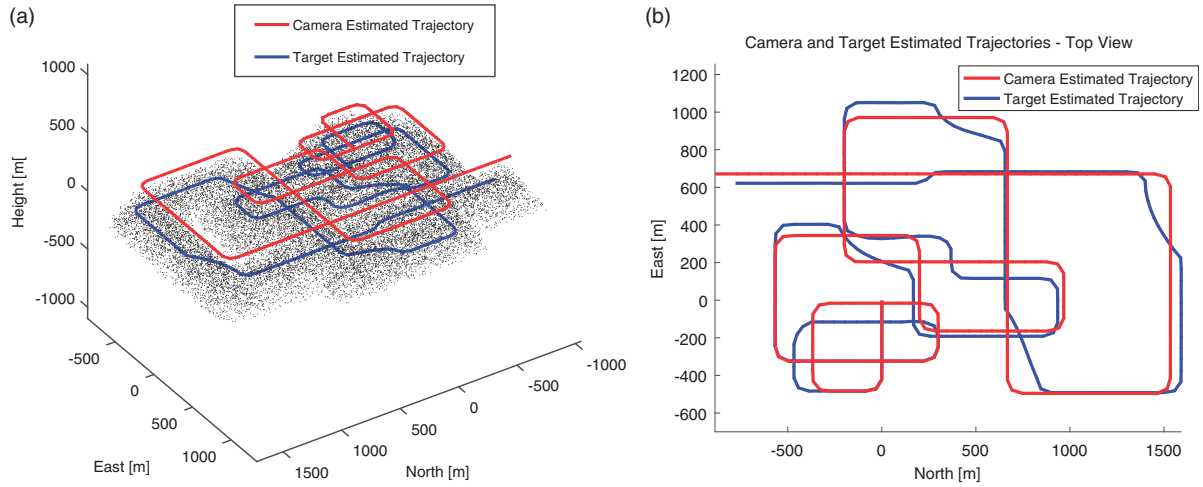


Figure 6. (a) Estimated camera (red) and target (blue) trajectories for the large synthetic scenario with 24,500 observed landmarks (shown in black). (b) Top view of the target and camera estimated trajectories for the large-scale synthetic scenario. At this scale, ground truth and estimated trajectories are indistinguishable (see Figure 7).

this case, the navigation is performed relatively to the camera's and target's initial positions. Those were initialized from their ground truth values, causing initial errors to be zero for all the estimated states.

Figure 5(d) shows statistics over running time between the proposed method and full BA with target tracking. For BA, a distinct increase in computational time can be observed at view 38, where a loop closure occurs. While one can already observe a significant difference in running time between the two methods in favor of LBA, we confirm this observation further in a larger scenario and with real imagery experiments in the next sections.

Large scenario

The large scenario, shown in Figure 6(a), simulates an approximately 14.5-km-long aerial path and involves a series of loop closures, resulting in variables recalculation during optimization. The target takes a different course on the ground (as shown in Figure 6(b)), which causes losses of target sight for approximately a seventh of the frames. In these cases, only the motion model factor is taken into consideration.

The obtained average camera position incremental errors for LBA and BA are 1.27 and 0.51 m, respectively, with a maximum error of 5.11 and 2.33 m. While the accuracy levels are similar, one can easily notice the difference in running time. Loop closures have a high impact on BA running time due to the landmark re-elimination and re-linearization they trigger; this process is avoided with LBA. It results in an average processing time of 3.3 s for LBA with target tracking, versus 22.2 s for BA method. The obtained overall

processing time for the same scenario is 809 s for the proposed method, versus 5329 s with BA.

Since we are interested to assess the similarity in terms of accuracies between the two techniques, we show in Figure 7(a) to 7(c) the *relative* errors between LBA and BA methods, meaning the difference between the estimation errors using both methods. Then, a comparison of the processing time is shown in Figure 7(d).

Experimental evaluation with real-imagery datasets

Further evaluations were performed through real-world experiments conducted at the Autonomous Navigation and Perception Lab (ANPL). Similarly to the synthetic dataset evaluation, these experiments involve a downward-facing camera which performed an aerial pattern while tracking a dynamic target moving on the ground. Ground truth data were gathered for the camera and the dynamic target using an independent real-time 6DOF optical tracking system. A scheme of the lab setup is presented in Figure 8 and two samples of typical captured images are presented in Figure 9. The recorded datasets are available online and can be accessed at <http://vindelman.net.technion.ac.il>.

Two different datasets were studied. In the first dataset, *ANPL1*, the camera and the target perform circular patterns, while in the second, *ANPL2*, they move in a more complex and unsynchronized manner, with occasional loss of target sight. Both cover an area of approximately $10[m] \times 6[m]$. In *ANPL1*, the camera and target travel 26.9 and 34.6 m, respectively, while in *ANPL2*, the distance traveled is 19 and 21.1 m, respectively. Image sensing was

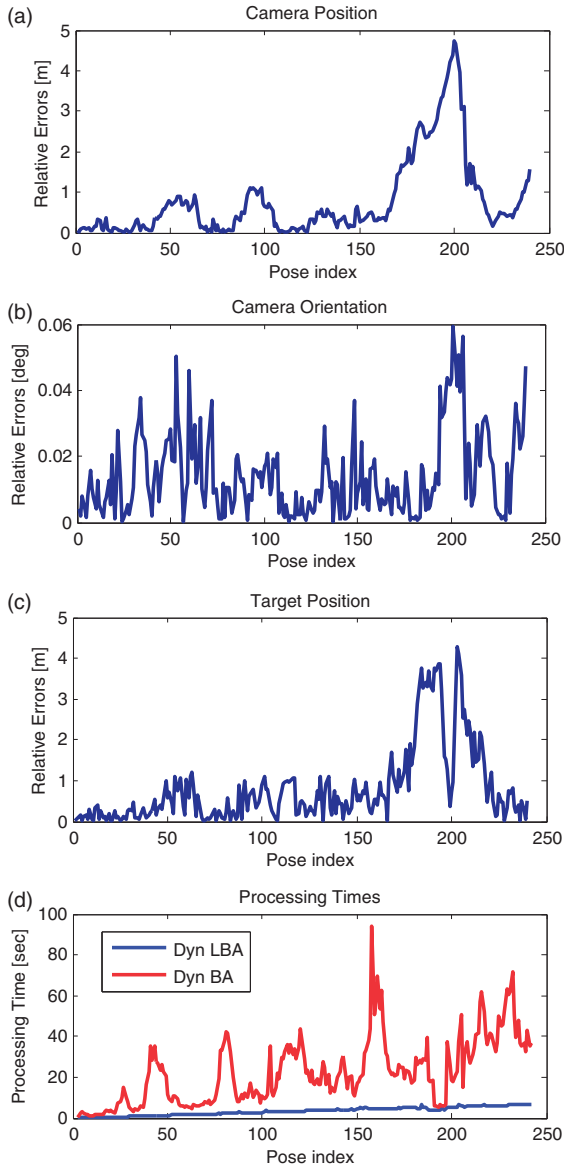


Figure 7. Incremental relative errors of LBA method with respect to BA method for the (a) camera position, (b) camera orientation, (c) target position, in the large-scale synthetic scenario. (d) a comparison of the processing times per frame. LBA: light bundle adjustment; BA: bundle adjustment.

performed using a high definition, wide angle camera and image distortion was corrected using calibration data. Table 1 provides further details regarding the number of views and observations, camera settings, and dataset durations.

Data association is performed using an implementation of the RANSAC algorithm⁴⁰ on the SIFT features that were extracted from the images and stored for potential loop closures. Since the experiments were conducted in a relatively constrained area with a wide field-of-view camera, numerous loop closures occur, as locations are often re-visited. For LBA, a single

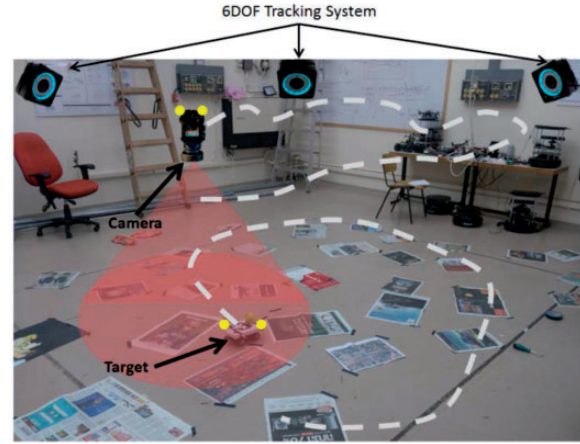


Figure 8. Conceptual scheme of the lab setup for the real-imagery experiments. The camera was manually held facing downwards and moved around the lab, in pre-defined patterns. Trackers, represented by yellow dots, were installed on the camera and on the target, allowing for detection by the ground truth system and measurement of their 6DOF poses. Images were scattered on the floor to densify the observed environment. Best seen in colour.

three-view constraint is added for each landmark observed more than twice in the past. This three-view constraint involves the current observation, the earliest observation and the one in the middle. A similar concept is used for 3D points triangulation, meaning the current observation and the earliest observation are taken into account. The target is detected by identification of the most highly recurrent SIFT feature, meaning we assumed that the SIFT feature which was detected in the highest number of frames belongs to the target. Although more advanced techniques exist, they are outside the scope of this work.

We compare the pose estimation errors of the camera and the position errors of the dynamic target with respect to ground truth for both LBA with target tracking and full BA cases. Incremental smoothing was applied for both methods in *ANPL1* dataset and standard batch optimization in *ANPL2*. QR factorization was used in all cases. We assume priors $p(x_0)$ and $p(y_0)$ on the initial camera and target states with means equal to their respective ground truth values and a $\sigma = 0.3$ [m] standard deviation. For the rest of the estimation process, new camera states are initialized by composition of last estimated pose with the relative motion from ground truth, corrupted with a white Gaussian noise $\sigma = 0.1$ [m] for position (i.e. the typical distance traveled between two frames) and $\sigma \sim 5$ [deg] (0.09 [rad]) on each axis for orientation. A different option, tested with the LBA method, consisted in composing the previous estimate and the relative motion extracted from the essential matrix calculated during



Figure 9. Typical images from the *ANPL1* real-imagery dataset.

Table 1. Dataset details.

	Camera resolution (pix)	Frames	Duration (s)	Landmarks	Observations
<i>ANPL1</i>	1280×960	80	40	2439	31,333
<i>ANPL2</i>	1920×1080	40	117	3366	25,631

ANPL: Autonomous Navigation and Perception Lab.

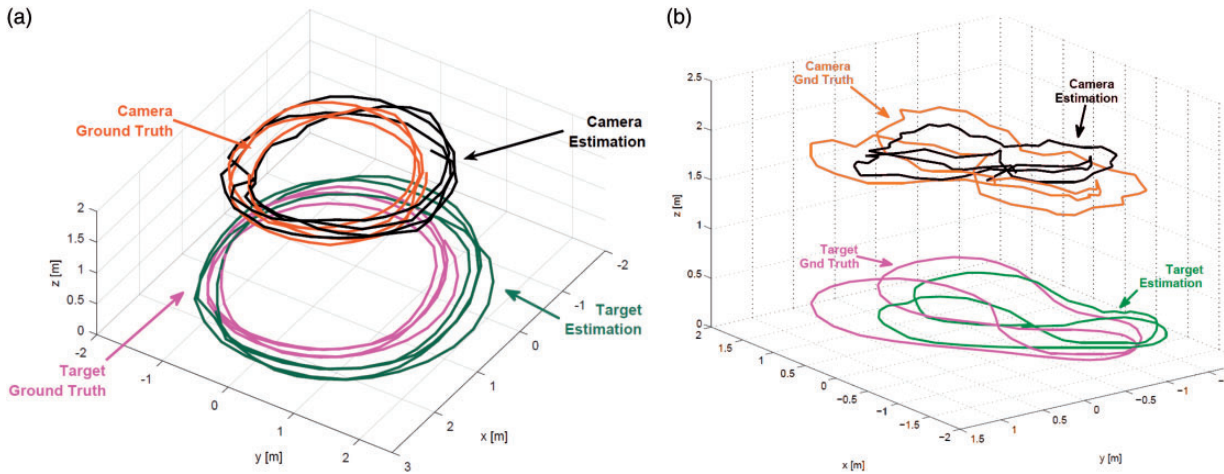


Figure 10. Estimated vs. ground truth 3D trajectories with real-imagery datasets for LBA approach in (a) *ANPL1* dataset (b) *ANPL2* dataset. BA approach produces similar results in terms of estimation errors, as shown in Table 2.

the data association process.³⁴ Results using the latter initialization method indicate similar performance with respect to the former initialization method. Here again, we use the constant velocity model for the dynamic target. This motion model becomes the only available information for trajectory estimation when the target moves out of the camera's field of view, as it is the case for $\sim 15\%$ of the frames in *ANPL2*. Similarly to the synthetic simulations, we assume the target moves on the ground, and thus constrain the first vertical velocity to zero. The measurement model assumes an image noise $\sigma = 0.5$ [pix].

Figure 10 shows the estimated trajectories and ground truth for the camera and the dynamic target in both datasets, using LBA method. We calculate an average error in position estimation of 22 and 38 cm for the camera and the target, respectively, in *ANPL1* dataset, and of 49 and 47 cm in the *ANPL2* dataset. The same level of position accuracy is calculated for the BA method. These errors are due (at least partially) to a specific practical data synchronization issue (ground truth data vs. image sequence) during the experiment. Similarly to the large-scale simulation case, we show in Figures 11(a) to (c), the relative errors between LBA

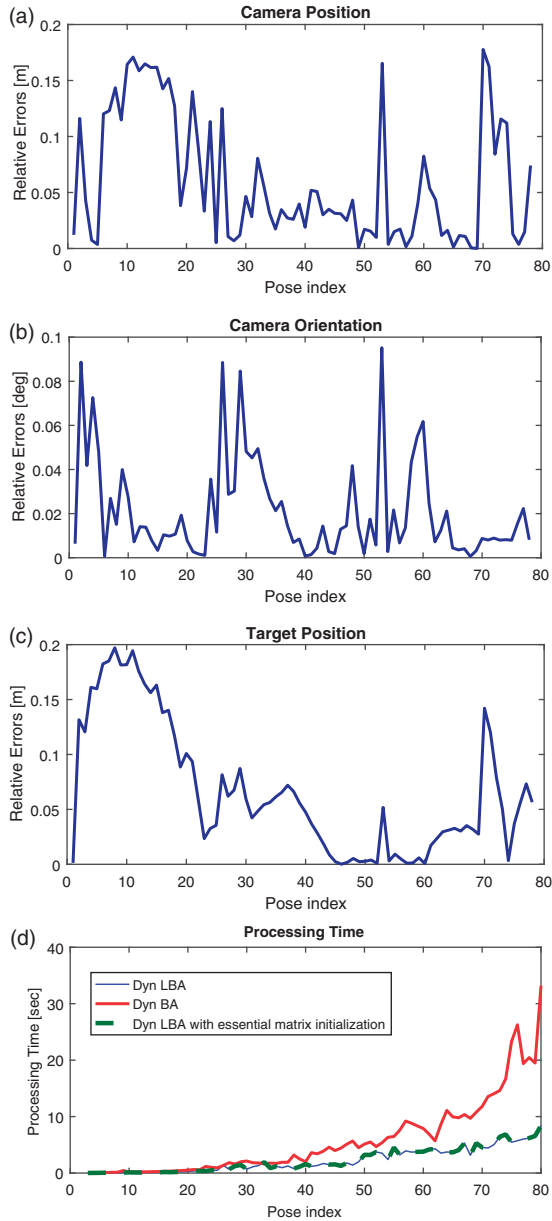


Figure 11. Incremental relative errors for the (a) camera position, (b) camera orientation, (c) target position, in *ANPL1* dataset, (d) a comparison of the processing times per frame. LBA: light bundle adjustment; BA: bundle adjustment.

and BA methods and a comparison of the processing time in Figure 11(d).

Tables 2 and 3 summarize the absolute values of the relative errors and the processing times for the two datasets. In both cases, the two methods show similar levels of accuracy: The average values for target and camera positions are 7 and 6 cm, respectively, for *ANPL1* dataset, and 14 and 8 cm for *ANPL2* dataset. In contrast, LBA with dynamic target tracking shows consequently better computational performances.

Table 2. Relative estimation errors summary of LBA method with respect to BA method for the camera and target positions in *ANPL1* and *ANPL2* datasets.

Dataset	Target position error (m)		Camera position error (m)	
	Mean	Max	Mean	Max
<i>ANPL1</i>	0.07	0.19	0.06	0.18
<i>ANPL2</i>	0.14	0.42	0.08	0.34

Note: The table entries are absolute values.

ANPL: Autonomous Navigation and Perception Lab.

Table 3. Summary of the processing times with LBA and BA methods for the *ANPL1* dataset.

Dataset	Method	Processing time (s)	
		Mean	Total
<i>ANPL1</i>	BA	5.6	447.8
	LBA	2.2	177.1
<i>ANPL2</i>	BA	3.1	222.9
	LBA	1.9	139.4

LBA: light bundle adjustment; BA: bundle adjustment; ANPL: Autonomous Navigation and Perception Lab.

The mean processing time per step is reduced by 61% for *ANPL1* and by 39% for *ANPL2*.

Conclusions and future work

We presented an efficient method for simultaneous ego-motion estimation and target tracking using the LBA framework. By algebraically eliminating the observed landmarks from the optimization, we allow the target to become the only reconstructed 3D point in the process. This reduces significantly the number of variables compared to full BA methods, and thus, allows for processing time improvements. We presented the mathematical process involved in the integration of the target tracking problem into the LBA framework, leading to a cost function that is formulated in terms of multi-view constraints, target motion model, and observations of the target. Computational efforts are further reduced by applying incremental inference over factor graphs representing the optimization problem, thus performing partial calculations at each optimization step.

We investigate the performance of the proposed approach and compare it to the corresponding BA formulation using synthetic and real-imagery datasets. While the two approaches exhibit similar accuracy levels, a significantly reduced running time was obtained for the proposed approach with both experimental methods. In particular, the presented method

was up to seven times faster than full BA in the simulations and up to two and a half times faster in the real-imagery experiments. This difference, however, is expected to vary with the number of landmarks observed per frame. The created real-imagery datasets have been made available to the research community through the ANPL website. These datasets include recorded images with synchronized ground truth for both the camera and the target, and is seen as a contribution by itself.

As for future work, aerial experiments including scale estimation for both BA and LBA methods (potentially using fusion with additional sensors such as IMU) could further improve the realism of the scenario. Also, an experimental implementation of our method to the multi-target tracking problem seems a natural continuation. In this case, the method used for targets detection and data-association represents a real challenge.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

1. Vu T-D. *Vehicle perception: localization, mapping with detection, classification and tracking of moving objects*. PhD Thesis, Institut National Polytechnique de Grenoble-INPG, France, 2009.
2. Wang C, Thorpe C and Thrun S. Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas. In: *IEEE international conference on robotics and automation (ICRA)*, Taipei, Taiwan, 2003, pp.842–849.
3. Eustice R, Singh H, Leonard J, et al. Visually navigating the RMS titanic with SLAM information filters. In: *Robotics: science and systems (RSS)*, Massachusetts, USA, 8–11 June, 2005.
4. Hover FS, Eustice RM, Kim A, et al. Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *Int J Rob Res* 2012; 31: 1445–1464.
5. Chekhlov D, Gee AP, Calway A, et al. Ninja on a plane: automatic discovery of physical planes for augmented reality using visual slam. In: *Proceedings of the 2007 6th IEEE and ACM international symposium on mixed and augmented reality*. Washington, DC, USA: IEEE Computer Society, 2007, pp.1–4.
6. Zhou F, Been-Lirn Duh H, and Billinghurst M. Trends in augmented reality tracking, interaction and display: a review of ten years of ISMAR. In: *Proceedings of the 7th IEEE/ACM international symposium on mixed and augmented reality*. IEEE Computer Society, Washington DC, USA, 2008, pp.193–202.
7. Lipton AJ, Fujiyoshi H and Patil RS. Moving target classification and tracking from real-time video. In: *Proceedings fourth IEEE workshop on applications of computer vision. WACV'98*. IEEE, USA, 1998.
8. Clendenin RM and Freeman RS. *Optical target tracking and designating system*. US Patent 4,386,848, USA, 1983.
9. Wright SE. UAVs in community police work. *AIAA Infotech@Aerospace 2005-6955*, Arlington, Virginia, September 26 - 29, 2005.
10. Neira J, Davison AJ and Leonard JJ. Guest editorial special issue on visual slam. *IEEE Trans Rob* 2008; 24: 929–931.
11. Indelman V, Roberts R and Dellaert F. Incremental light bundle adjustment for structure from motion and robotics. *Rob Auton Syst* 2015; 70: 63–82.
12. Indelman V. Bundle adjustment without iterative structure estimation and its application to navigation. In: *IEEE/ION position location and navigation system (PLANS) conference*, Myrtle Beach, SC, USA, 23–26 April, 2012.
13. Indelman V, Roberts R, Beall C, et al. Incremental light bundle adjustment. In: *British machine vision conference (BMVC)*, Guildford, UK, 3–7 September 2012.
14. Kaess M, Johannsson H, Roberts R, et al. iSAM2: incremental smoothing and mapping using the Bayes tree. *Int J Rob Res* 2012; 31: 217–236.
15. Dissanayake MWMG, Newman PM, Durrant-Whyte HF, et al. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans Rob Automat* 2001; 17: 229–241.
16. Smith R and Cheeseman P. On the representation and estimation of spatial uncertainty. *Int J Rob Res* 1987; 5: 56–68.
17. Konolige K. Sparse sparse bundle adjustment. In: *British Machine Vision Conference (BMVC)*, Aberystwyth, Wales, UK, August 30 - September 2, 2010.
18. Lourakis MIA and Argyros AA. SBA: a software package for generic sparse bundle adjustment. *ACM Trans Math Softw* 2009; 36: 1–30.
19. Konolige K and Agrawal M. FrameSLAM: from bundle adjustment to realtime visual mapping. *IEEE Trans Rob* 2008; 24: 1066–1077.
20. Rodríguez AL, López de Teruel PE and Ruiz A. Reduced epipolar cost for accelerated incremental SFM. In: *IEEE conference on computer vision and pattern recognition (CVPR)*, 2011, Providence, RI, pp.3097–3104.
21. Steffen R, Frahm J-M and Förstner W. Relative bundle adjustment based on trifocal constraints. In: *ECCV workshop on reconstruction and modeling of large-scale 3D virtual environments*, Greece, September 11, 2010.
22. Eustice RM, Singh H and Leonard JJ. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Trans Rob* 2006; 22: 1100–1114.
23. Ila V, Porta JM and Andrade-Cetto J. Information-based compact pose SLAM. *IEEE Trans Rob* 2010; 26: 78–93.

24. Wang C-C and Thorpe C. Simultaneous localization and mapping with detection and tracking of moving objects. In: *Proceedings 2002 IEEE international conference on robotics and automation* (vol. 3). IEEE, Washington, DC, USA, 11-15 May, 2002, pp.2918–2924.
25. Bar-Shalom Y and Fortmann TE. *Tracking and data association*. New York: Academic Press, 1988.
26. Huang G, Zhou K, Trawny N, et al. A bank of maximum a posteriori (map) estimators for target tracking. *IEEE Trans Rob* 2015; 31: 85–103.
27. Montesano L, Minguez J and Montano L. Modeling the static and the dynamic parts of the environment to improve sensor-based navigation. In: *Proceedings of the 2005 IEEE international conference on robotics and automation*. IEEE, Barcelona, Spain, 18-22 April, 2005, pp.4556–4562.
28. Wang C-C. *Simultaneous localization, mapping and moving object tracking*. PhD Thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2004.
29. Vu T-D, Burlet J and Aycard O. Grid-based localization and local mapping with moving object detection and tracking. *Inform Fusion* 2011; 12: 58–69.
30. Ortega JS. *Towards visual localization, mapping and moving objects tracking by a mobile robot: a geometric and probabilistic approach*. PhD Thesis, Institut National Polytechnique de Toulouse-INPT, France, 2007.
31. Hähnel D, Schulz D and Burgard W. Mobile robot mapping in populated environments. *Adv Rob* 2003; 17: 579–597.
32. Pancham A, Tlale N and Bright G. *Literature review of SLAM and DATMO*. 4th Robotics and Mechatronics Conference of South Africa (RobMech 2011), CSIR International Conference Centre, Pretoria, 23–25 November 2011.
33. Thrun S, Burgard W and Fox D. *Probabilistic robotics*. MIT Press, 2005.
34. Hartley RI and Zisserman A. *Multiple view geometry in computer vision*. 2nd ed. Cambridge: Cambridge University Press, 2004.
35. Bar-Shalom Y, Rong Li X and Kirubarajan T. *Estimation with applications to tracking and navigation: theory algorithms and software*. New Jersey, USA: John Wiley & Sons, 2004.
36. Kschischang FR, Frey BJ and Loeliger H-A. Factor graphs and the sum-product algorithm. *IEEE Trans Inform Theory* 2001; 47: 498–519.
37. Indelman V, Gurfil P, Rivlin E, et al. Real-time vision-aided localization and navigation based on three-view geometry. *IEEE Trans Aerosp Electron Syst* 2012; 48: 2239–2259.
38. Indelman V. *Navigation performance enhancement using online mosaicking*. PhD Thesis, Technion, Israel Institute of Technology, Israel, 2011.
39. Kaess M, Ranganathan A and Dellaert F. iSAM: incremental smoothing and mapping. *IEEE Trans Rob* 2008; 24: 1365–1378.
40. Fischler M and Bolles R. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun ACM* 1981; 24: 381–395.

Deep learning for vision-based micro aerial vehicle autonomous landing

International Journal of Micro Air Vehicles
2018, Vol. 10(2) 171–185
© The Author(s) 2018
DOI: 10.1177/1756829318757470
journals.sagepub.com/home/mav


Leijian Yu¹, Cai Luo², Xingrui Yu¹, Xiangyuan Jiang¹, Erfu Yang³,
Chunbo Luo⁴ and Peng Ren¹

Abstract

Vision-based techniques are widely used in micro aerial vehicle autonomous landing systems. Existing vision-based autonomous landing schemes tend to detect specific landing landmarks by identifying their straightforward visual features such as shapes and colors. Though efficient to compute, these schemes only apply to landmarks with limited variability and require strict environmental conditions such as consistent lighting. To overcome these limitations, we propose an end-to-end landmark detection system based on a deep convolutional neural network, which not only easily scales up to a larger number of various landmarks but also exhibit robustness to different lighting conditions. Furthermore, we propose a separative implementation strategy which conducts convolutional neural network training and detection on different hardware platforms separately, i.e. a graphics processing unit work station and a micro aerial vehicle on-board system, subject to their specific implementation requirements. To evaluate the performance of our framework, we test it on synthesized scenarios and real-world videos captured by a quadrotor on-board camera. Experimental results validate that the proposed vision-based autonomous landing system is robust to landmark variability in different backgrounds and lighting situations.

Keywords

Micro aerial vehicle, vision-based autonomous landing, convolutional neural networks

Received 19 May 2017; accepted 11 January 2018

Introduction

In recent years, Unmanned Aerial Vehicles (UAVs) have been widely utilized in both military and civilian fields, such as military real-time monitoring, resource exploration, civil surveillance, cargo transportation and agricultural planning.¹ One key issue for safely applying UAVs to these tasks is to maneuver UAV flights in an accurate manner. Traditional UAV flights tend to be controlled through human manipulation with certain navigational aids. State-of-the-art UAV flights operate in an autonomous manner, which not only unleashes human labor but also enables safer and more accurate maneuvers. Specifically, three basic phases for UAV autonomous flights include takeoff, hovering and landing.² Among them, autonomous landing is the most crucial phase because 80% of the UAV accidents occur during landing.³ Therefore, how to build robust autonomous landing systems has become one of the most important and challenging topics for the UAV research.⁴

Existing autonomous landing systems of UAVs can be roughly classified into two groups, i.e. electromagnetically guided landing systems and vision-based landing systems. The electromagnetically guided landing systems include those based on inertial navigation

¹College of Information and Control Engineering, China University of Petroleum (East China), Qingdao, China

²College of Mechanical and Electronic Engineering, China University of Petroleum (East China), Qingdao, China

³Space Mechatronic Systems Technology Laboratory, Strathclyde Space Institute, Department of Design, Manufacture and Engineering Management, Glasgow, UK

⁴Department of Computer Science, College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, UK

Corresponding author:

Peng Ren, College of Information and Control Engineering, China University of Petroleum (East China), 66 Changjiang West Road, Qingdao 266580, China.

Email: pengren@upc.edu.cn



systems (INS) and global positioning systems (GPS). INS provides instant positioning information but cannot guarantee long-term positioning accuracy. GPS provides a global availability in open areas but may incur positioning errors up to 10 m and may be blocked by buildings.⁵ The electromagnetic landing systems are suitable for large scale landing problems (e.g. a large sized UAV landing on a large open area) with considerable tolerance for position errors. However, they cannot be straightforwardly applied to micro aerial vehicle (MAV) landing problems, which require accurate positioning within a small sized space. The electro-optical navigation can be considered as a transitioning landing technique between the electromagnetically guided landing and the vision-based landing, and generally serve as an auxiliary to electromagnetically guided landing systems. Vision-based landing systems use cameras to capture environmental visual features for the purpose of guided landing. One way to achieve this goal is to arrange cameras surrounding a landing area for capturing UAV/MAV status and environmental situations. One representative vision-based landing system in this regard is the VICON motion capture system. It is expensive and its application is limited to small indoor environments. In contrast to arranging off-board MAV cameras like VICON, one more general configuration for a vision based landing system is to attach a camera on a MAV. By using optimal images of landing targets as source information for navigation, on-board vision systems can achieve positioning accuracy in terms of centimeters. This is especially valuable for MAVs that require more effective precise landing than larger sized UAVs. One on-board landing system identifies visual features of specific landing landmarks observed by the camera and accordingly guides MAV autonomous landing actions. In this scenario, specific landmarks are required to be designed as prerequisites for performing autonomous landing. In the literature, different specific landmarks are developed for different landing systems. Tsai et al.⁶ designed the T-shaped landmark (Figure 1(a)) for their MAV autonomous landing systems. Saripalli et al.⁷ designed the H-shaped landmark (Figure 1(b)) for their helicopter autonomous landing. Lin et al.⁸ designed a landmark composed of eight equal-sized squares that are enclosed by a big white border (Figure 1(c)). Verbandt et al.⁹ designed a landmark consisting of a series of concentric circles with exponentially distributed radii (Figure 1(d)). Jung et al.¹⁰ designed an H-shaped landmark with concentric circles (Figure 1(e)). These landmarks are designed to contain sharp or contrastive features that are easy to identify and segment from the background.

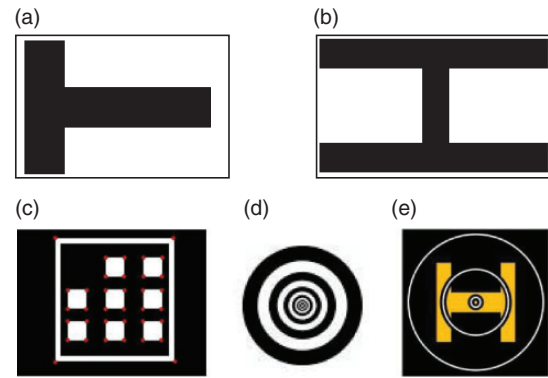


Figure 1. Samples of five widely used landmarks: (a) the T-shaped landmark; (b) the H-shaped landmark; (c) the landmark consisting of eight equal-sized squares and a big white border; (d) the landmark composed of a series of concentric circles; (e) the landmark composed of an H-shaped and concentric circles.

The key factor for vision-based landing systems is accurate landmark detection. Each existing vision based landing system is able to provide acceptable detection accuracy on its own landmark, but can hardly accurately detect landmarks for another system. This is because existing vision-based landing systems tend to detect specific landmarks through matching low level visual features such as shapes and colors between captured images and designed landmarks.¹¹ This case-by-case detection strategy is restricted to predefined landmarks and can hardly be generalized to a broad variety of landmarks. Furthermore, the visual quality of low level features extracted from captured images may also be easily devastated by inconsistent lighting conditions.

One intrinsic reason for these limitations is the machine learning techniques employed in the existing vision-based landing systems lack capability of learning high level visual features. In order to overcome these limitations, we exploit deep learning models for detecting the various landmarks under inconsistent lighting conditions. Deep learning is referred to a series of multilayer representational learning models that have attracted extensive research interests and achieved state-of-the-art performance on a number of artificial intelligence tasks.¹² In this paper, we describe how to exploit a deep convolutional neural network (CNN) model for detecting landmarks.

Traditional detection methods tend to first extract low level handcrafted features from captured images and then search the whole image for features that can match the predefined targets.^{13,14} Recently, low level features are characterized in terms of region proposals, which are generated by feature learning techniques possibly being deep models. R-FCN¹⁵ and MASK R-CNN¹⁶ use region proposal networks to generate

detection proposals. Detection is then conducted via proposal classification. These methods consider feature extraction and feature matching as two separate steps and the speeds of these methods are slow because the networks in two stages are trained separately. In contrast to the two-step detection schemes, the end-to-end methodologies such as the Yolo methods use one network to predict the objects.^{17,18}

Meanwhile, convolutional neural networks have been extensively studied in the deep learning literature. A number of attempts have been made for developing deeper and more complicated networks to achieve high accuracy.^{19,20,21} However, these networks require extensive computational resources. On the other hand, resource limited platforms, such as the MAV on-board processors, are not qualified to implement these complicated networks.

Our framework is motivated by the recent proposed detection model Yolo¹⁷ and the neural network architecture SqueezeNet²² to achieve real-time landmark detection. The Yolo model frames object detection in terms of deep learning based regression for the purpose of determining spatially separated bounding boxes and associated class probabilities. The SqueezeNet aims at modeling a CNN with few parameters. In order to develop an effective end-to-end landmark detection system with implementation efficiency, we establish our CNN framework sharing advantages of the Yolo regression and the SqueezeNet efficient architecture. Specifically, our CNN-based landmark detection method regresses landmark positions directly from captured raw images through a multilayer architecture such that the feature extraction and matching are indistinguishably integrated into an overall framework. Furthermore, the strong representational power of the CNN not only increases the adaptability of an autonomous landing system from one specific landmark to multiple landmarks but also improves the detection robustness with respect to light variation.

Training our CNN based detection model is always time consumptive with heavy computational overheads. On the other hand, conducting detection based the trained CNN requires instant operations. To address these contradicted problems, we propose to train our CNN based detection model on a GPU workstation and operate the trained CNN model for detecting landmarks in the MAV on-board system. The separative implementations take advantages of both the GPU computational power and the on-board instant feedback, resulting in a novel strategy which leverages between comprehensively training and instantly operating deep models for MAV applications.

We experimentally test our CNN based landmark detection framework on synthesized scenarios and real-world videos captured by the on-board camera of a

quadrotor. Experimental results validate that the proposed vision-based autonomous landing system is robust across various landmarks and different lighting situations.

Training a convolutional neural network for landmark detection

Inspired by the Yolo model¹⁷ and SqueezeNet²² modeling methodologies, we develop a convolutional neural network that performs end-to-end landmark detection. In this section, we first introduce the architecture for our convolutional neural network, and then describe how to train the CNN model for landmark detection on a GPU platform.

Convolutional neural network architecture for landmark detection

The convolutional neural network architecture of our proposed detection model is shown in Figure 2. Each input into the model is an RGB three channel image captured by an MAV on-board camera, and the corresponding outputs of the model are the predicted location of a detected landmark in the image and the predicted category label of the landmark. Specifically, one input image is first processed four convolutional and pooling layers, i.e. C_1, C_2, C_3, C_4 , followed by one fully connected layer and one detection layer for regression. The blue cubes in Figure 2 indicate feature maps in each layer. Specifically, the C_{n-1} layer consists of K feature maps, i.e. $X_{n-1}^{(1)}, \dots, X_{n-1}^{(K)}$, and these feature maps are the sources for computing the feature maps in the layer C_n .

To generate the l th feature map $X_n^{(l)}$ in the n th layer C_n , the feature maps in the $(n-1)$ th layer C_{n-1} are processed by convolutional-activation-pooling (CAP) operations, which are basic operations in convolutional neural networks. Each feature map $X_{n-1}^{(k)}$ in the $(n-1)$ th layer is convolved with learnable weights $W_n^{(k,l)}$. The sum of the K convolved results are added with learnable biases $b_n^{(l)}$, and further processed by a leaky rectified linear activation function $f(\cdot)$ which is depicted as follows:

$$f(m) = \begin{cases} m, & m > 0 \\ 0.1m, & m \leq 0 \end{cases} \quad (1)$$

The convolution-activation (CA) operations on the $(n-1)$ th layer is formulated as follows:

$$\mathfrak{X}_n^{(l)} = f\left(\sum_{k=1}^K (W_n^{(k,l)} * X_{n-1}^{(k,l)}) + b_n^{(l)}\right) \quad (2)$$

where $*$ denotes the convolution operation.

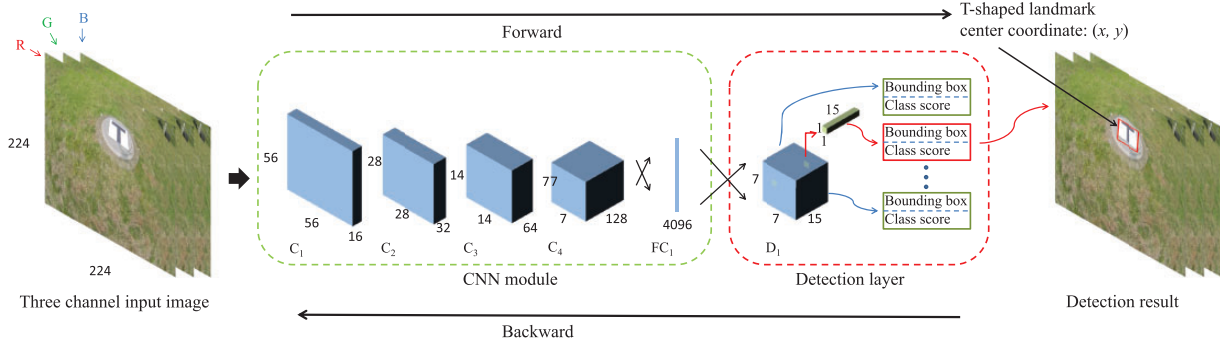


Figure 2. The full architecture of the proposed regression-based detection model. The architecture is composed of a CNN module and a detection layer. The input image is fed into the CNN module, followed by a detection layer. The detection layer provides the capability of regressing the coordinates and the class probabilities of the landmarks.

CNN: convolutional neural network.

$\mathfrak{X}_n^{(l)}$ is an intermediate feature map in C_n . It is further processed in terms of a pooling (P) operation for generating the feature map $X_n^{(l)}$ in C_n . Specifically, the pooling operation reduces the size of the $\mathfrak{X}_n^{(l)}$ by shrinking its 2×2 or 4×4 patches into single elements. This is done by replacing the patch by the largest valued element among the patch. The computation of the (i, j) th entry of the feature map $X_n^{(k,l)}$ is formulated as follows:

$$X_{n;i,j}^{(k,l)} = \max\{\mathfrak{X}_{n;i,j}^{(k,l)}, \mathfrak{X}_{n;i+1,j}^{(k,l)}, \mathfrak{X}_{n;i,j+1}^{(k,l)}, \mathfrak{X}_{n;i+1,j+1}^{(k,l)}\} \quad (3)$$

We describe the detailed configuration of our CNN landmark detection framework shown in Figure 2. In the layer C_1 , the input three channel image is processed by a CAP operation and generate 16 feature maps of the size of 56×56 . Similarly, C_2 has 32 feature maps of the size of 28×28 , C_3 has 64 feature maps of the size of 14×14 , and C_4 has 128 feature maps of the size 7×7 . The detailed configuration is described in Table 1.

The parameter values 3×3 for the Conv Filter in C_1 refer to the size of each filter $W_1^{(k,l)}$ and the following parameter value 16 refers to the number K of filters applied in the layer. The parameter value 2 for Stride indicates the sliding step size for. The parameter values 2×2 for the Maxpooling indicate that pooling operations take place within a region of size 2×2 . For C_2 , C_3 and C_4 , the parameter values have similar implications.

There are differences between the layers C_1 and C_4 and the layers C_2 and C_3 . We design the layers C_1 and C_4 following Yolo.¹⁷ On the other hand, different from Yolo, we design C_2 and C_3 by applying Conv Filters of the ‘squeezed’ size 1. This methodology is motivated by SqueezeNet²² which replaces one big Conv Filter by parallel ‘squeezed’ Conv Layers and yields a simplified structure with reduced number of parameters. We

Table 1. The CNN layer configuration.

C_n	Layer configuration
C_1	Conv Filter $3 \times 3 \times 16$, Stride 2 Maxpooling 2×2 , Stride 2
C_2	Conv Filter $1 \times 1 \times 8$, Stride 1 Conv Filter $1 \times 1 \times 32$, Stride 1 Conv Filter $3 \times 3 \times 32$, Stride 1 Maxpooling 2×2 , Stride 2
C_3	Conv Filter $1 \times 1 \times 16$, Stride 1 Conv Filter $1 \times 1 \times 64$, Stride 1 Conv Filter $3 \times 3 \times 64$, Stride 1 Maxpooling 2×2 , Stride 2
C_4	Conv Filter $3 \times 3 \times 128$, Stride 1 Maxpooling 2×2 , Stride 2

CNN: convolutional neural network.

exploit this advantage of SqueezeNet for conducting simplified CNN computation in an on-board system with limited computational resources. However, for on-board small CNNs, the SqueezeNet parallelism sacrifices certain accuracy for simplifying the CNN model. To remedy this ineffectiveness, we modify the parallelism into a serial implementation which is deeper and more effective to learn more complex feature representations.

The layer C_4 is followed by one fully-connected layer (FC_1) and then fully connected to a vector with 4096 dimensions. The full connection is depicted by cross arrows in Figure 2. Finally, the 4096 dimensional vector is processed by the detection layer (D_1) to generate a prediction tensor of the size $7 \times 7 \times 15$.

One prediction outputted by the CNN is represented as a 15 dimensional vector in the prediction tensor and the CNN generates 49 such prediction vectors for one input image. For each prediction vector, the first five entries represent the first prediction $(x_0, y_0, w_0, h_0, c_0)$, and the subsequent five entries represent the second

prediction $(x_1, y_1, w_1, h_1, c_1)$. Here (x_i, y_i) represent one predicted landmark centric coordinate and c_i reflects the confidence that a landmark is located within a grid cell surrounding (x_i, y_i) for $i \in \{1, 2\}$. Specifically, the confidence score is zero when no object falls into the grid cell. The confidence score is computed in terms of the intersection over union (IoU) between the predicted landmark and the ground truth as follows:

$$c_i \equiv \text{IoU} = \frac{\mathcal{A}_O}{\mathcal{A}_U} \quad (4)$$

where \mathcal{A}_O and \mathcal{A}_U denote the area of overlap and the area of union of the predicted landmark and the true landmark, respectively. We assume that there are totally five different categories of landmarks (as illustrated in Figure 1) used for landing, and the final five entries represent the probabilities $\text{Pr}(L_i), i = 1, \dots, 5$ of the detected landmark belonging to one of the five candidate categories.

The class score s is computed by multiplying the conditional class probabilities and the individual confidence score for each landmark:

$$s = \text{Pr}(L_i) * \text{IoU} \quad (5)$$

where L_i denotes one of the landmark categories illustrated in Figure 1.

The best prediction is selected from the 49 predictions according to the highest class score s^* . For each prediction, the class score s is computed by equation (5). The best prediction consists of two components – the bounding box (x^*, y^*, w^*, h^*) and the class score s^* .

In the next subsection, we will comprehensively describe how to optimize the learnable parameters $W_n^{(k,l)}$ and b_n based on the prediction tensor.

Training the convolutional neural network on a GPU workstation

For training the CNN detection framework, we first resize the input image into 224×224 and then divide it into a 7×7 equally-sized grid cells. The cells are responsible for detecting the landmark if the center of the landmark falls into one of them. As described in the previous subsection, the CNN framework generates a $7 \times 7 \times 15$ prediction tensor for one input image, with each 15 dimensional vector in the tensor corresponding to one cell of the input image. The training procedure is to optimize the learnable parameters by minimizing the loss function measuring the differences between the prediction tensors and target tensors for input images.

The loss function consists of three parts, i.e. the area loss \mathcal{L}_{area} , categorical loss \mathcal{L}_{cls} , and the IoU loss \mathcal{L}_{IoU} , which are separately formulated as follows:

$$\begin{aligned} \mathcal{L}_{area} = & \lambda_c \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{i,j}^L [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ & + \lambda_c \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{i,j}^L [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{i,j}^L (\mathcal{C}_i - \hat{\mathcal{C}}_i)^2 \\ & + \lambda_{no} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{I}_{i,j}^{no} (\mathcal{C}_i - \hat{\mathcal{C}}_i)^2 \end{aligned} \quad (6)$$

where \mathbb{I}_i^L denotes the confidence of landmark appearing in cell i , $\mathbb{I}_{i,j}^L$ denotes the j th (first or second) predictor in cell i is responsible for that prediction, and \mathcal{C}_i indicates the class label of landmark in cell i . λ_c and λ_{no} are two balance parameters for making the training of the detection model more stable. In our design, we empirically set $\lambda_c = 5$ and $\lambda_{no} = 0.5$.

$$\mathcal{L}_{cls} = \sum_{i=0}^{S^2} \mathbb{I}_{i,j}^L (p_i(\mathcal{C}') - \hat{p}_i(\mathcal{C}'))^2 \quad (7)$$

where $p_i(\mathcal{C}')$ is the conditional probability for landmark with label \mathcal{C}' .

$$\mathcal{L}_{IoU} = \sum_{i=0}^{S^2} \mathbb{I}_{i,j}^L (1 - c_i)^2 \quad (8)$$

where c_i is the confidence score, i.e. IoU, can be computed by equation (4).

The overall loss function is:

$$\mathcal{L} = \mathcal{L}_{area} + \mathcal{L}_{cls} + \mathcal{L}_{IoU} \quad (9)$$

Given a batch of training data (m image samples), we train the detection model via the stochastic gradient descent (SGD) algorithm. We first initialize the learnable weights W and biases b to a small random value near to zero subject to a normal distribution – $\text{Normal}(0, \epsilon^2)$ with $\epsilon = 0.01$. During training, the input images are pushed forward (marked with the right-facing arrow in Figure 2) through the whole network to generate predictions. Then the errors in terms of the cost function equation (9) are measured. The error

gradients for the weights and biases are computed by equation (10) and backwardly propagated (marked with the left-facing arrow in Figure 2) for updating the parameter values.

$$\begin{aligned}\nabla_{W^{(l)}} \mathcal{L} &= \frac{\partial \mathcal{L}}{\partial W^{(l)}}. \\ \nabla_{b^{(l)}} \mathcal{L} &= \frac{\partial \mathcal{L}}{\partial b^{(l)}}.\end{aligned}\quad (10)$$

Algorithm 1: One iteration of stochastic gradient descent.

Set $\Delta W^{(l)} := 0, \Delta b^{(l)} := 0$
for $i = 1$ **to** m **do**

1. Compute the gradients $\nabla_{W^{(l)}} \mathcal{L}$ and $\nabla_{b^{(l)}} \mathcal{L}$ as equation (10).
2. Set $\Delta W^{(l)} = \Delta W^{(l)} + \nabla_{W^{(l)}} \mathcal{L}$.
3. Set $\Delta b^{(l)} = \Delta b^{(l)} + \nabla_{b^{(l)}} \mathcal{L}$.

Update the parameters:

$$\begin{aligned}W^{(l)} &= W^{(l)} - \alpha \left[\left(\frac{1}{m} \Delta W^{(l)} \right) + \lambda W^{(l)} \right] \\ b^{(l)} &= b^{(l)} - \alpha \left[\frac{1}{m} \Delta b^{(l)} \right]\end{aligned}\quad (11)$$

where α is the learning rate, λ denotes the weight decay parameter for adjusting the influence of model complexity, m is the number of images.

end

To train the CNN based detection model, we repeatedly take steps of the stochastic gradient descent as described in Algorithm 1 to minimize the loss function \mathcal{L} in equation (9).

There are two things that need to be noted in our training procedure. First, the data jittering approach is employed to augment the landmark dataset. Specifically, the augmentation strategies operated on the landmark dataset include adjusting the image exposure, saturation and hue. The data augmentation increases the intraclass variability of training data and thus further improves the robustness of the detection model. Second, the training is carried out on a Graphics Processing Unit (GPU) work station, which is widely used for training deep learning models. However, it is not suitable for an MAV on-board system to perform CNN training by involving GPUs, which are normally physically big in size and comparatively power consumptive. On the other hand, it is not necessary to train a CNN in an on-board system because the MAV instant manipulation just requires implementing a trained CNN model rather than training it. In the light of this observation,

we use an off-board GPU workstation for efficiently training our CNN framework. After the training procedure, we obtain about 4000 optimal parameters, i.e. optimal weights W^* and optimal bias b^* . We then transfer these trained parameter values to an on-board system for manipulating MAV landing, which is described in the next section.

Landmark detection based on the trained convolutional neural network

Landmark detection via CNN inference

The training procedure first forwardly computes the prediction tensor based the initial parameter values, and then adjusts the parameter values backwardly via back propagation optimization. In contrast to the forward-backward training procedure, the inference procedure processes each frame of a captured video through the CNN network only forwardly, based on the optimal parameter values (weights W^* and biases b^*). The forward computation generates the predicted coordinate (x^*, y^*) and class scores s^* for the detected landmark. Following the forward procedure (marked with the right-facing arrow in Figure 2), the inference procedure of an input three channels image I is summarized in Algorithm 2:

Algorithm 2: Landmark detection procedure with the optimal parameters W^* and b^* .

Set $L = 5, X_0^0 = I, Ks = \{3, 16, 32, 64, 128\}$

1. **for** $l = 1$ **to** L **do**

(a) Set $K = Ks(l)$.

(b) **for** $n = 1$ **to** 3 **do**

Generate feature maps according to equation (2) with the optimal parameters

$$X_n^{(l)} = f \left(\sum_{k=1}^K (W_n^{*(k,l)} * X_{n-1}^{(k,l)}) + b_n^{*(l)} \right) \quad (12)$$

end

end

2. Process feature maps in the layer FC_1 based on W^* and b^* .

3. Process feature maps in the layer D_1 based on W^* and b^* to generate 49 predictions (15-dimension vector).

Each prediction can be presented as:

$$\begin{aligned}&[(x_0, y_0, w_0, h_0, c_0), (x_1, y_1, w_1, h_1, c_1), \\ &(\Pr(L_1), \Pr(L_2), \Pr(L_3), \Pr(L_4), \Pr(L_5))]\end{aligned}\quad (13)$$

4. Compute the class score s for each prediction according to equation (5).
5. Select the best coordinate prediction (x^*, y^*) according to the highest class score s^* .

Implementing detection on an MAV on-board system

We implement the detection model on the MAV on-board system Manifold, which consists of a quad-core ARM Cortex-A15 processor and 2 GB memory. The trained network is used for detecting landmarks from unlabeled video frames captured by the on-board camera. To run the inference procedure on Manifold, we build detection model with the 4000 optimal parameters (W^* and b^*) loaded on-board. Once the Manifold starts processing video captured by on-board camera, the inference procedure summarized in Algorithm 2 begins. The predictions in the form of $(x^*, y^*, w^*, h^*, s^*)$ is generated for guiding the landing.

It should be noted that we train and implement the CNN detection framework based on separate hardware platforms, i.e. the GPU workstation and the Manifold MAV on-board system. Though the CNN framework is developed with simplified architecture, it still requires considerable computational overheads especially in the training phase. Specifically, inferencing with the trained CNN is much less computational consumptive than training it, because the inference implementation does not involve the costly backward gradient computation. Therefore, in contrast to exploiting the computational power of the GPU workstation to handle the complex computation in the training phase, we implement the less complex detection procedure in the Manifold

MAV on-board system, which is not only smaller in size, lighter in weight and less costive in power than the GPU workstation but also qualified to conduct an on-line real-time landmark detection.

Landing system

Figure 4 illustrates the operating procedures of the autonomous landing system. The image acquisition procedure is completed by capturing video frames using an on-board camera with universal serial bus (USB). The coordinates of the landmarks are predicted by forwarding the trained CNN detection model. The predicted coordinates of the landmarks are then converted into x-axis and y-axis angular offsets (in radians) of the landmark center. These angular offsets are sent to the autopilot via the Micro Air Vehicles Communication Protocol (MAVLINK). This information is used to generate control signals to supervise the landing process via the autopilot. The detailed procedures are described in the following subsections.

Software architecture

The predicted landmark coordinates from videos captured by an MAV on-board camera are transformed to the MAV's own coordinates. We assume that the roll, pitch and yaw angles of the MAV can be neglected while computing the x-axis and y-axis angular offsets of the landmark coordinates in images. The x-axis and y-axis angular offsets are obtained as follows:

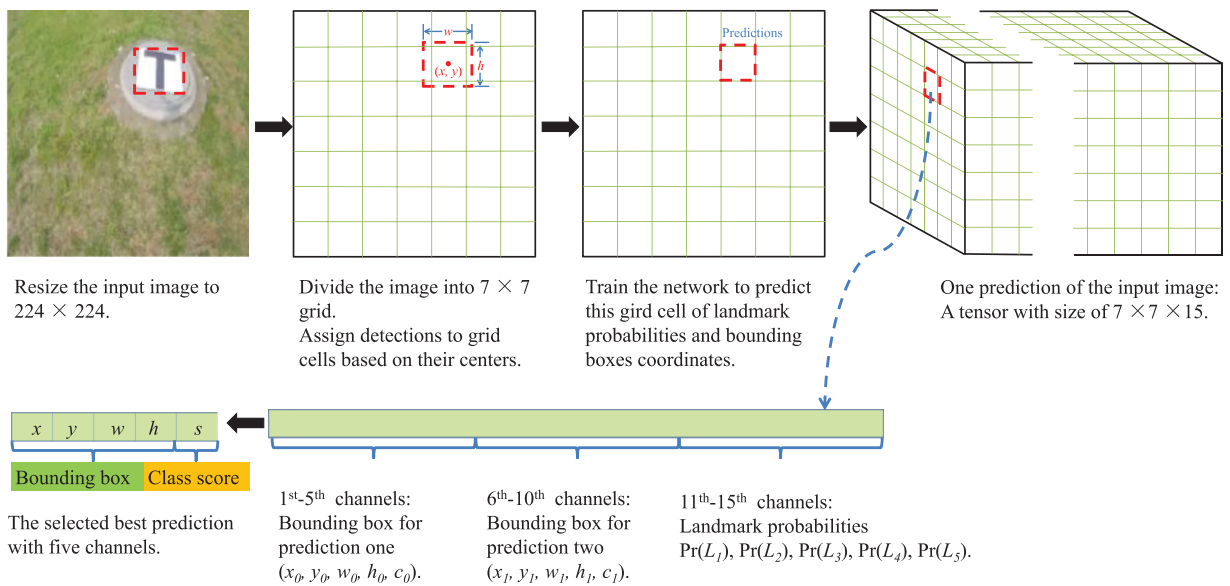


Figure 3. Detection procedure of the CNN-based detection model.
CNN: convolutional neural network.

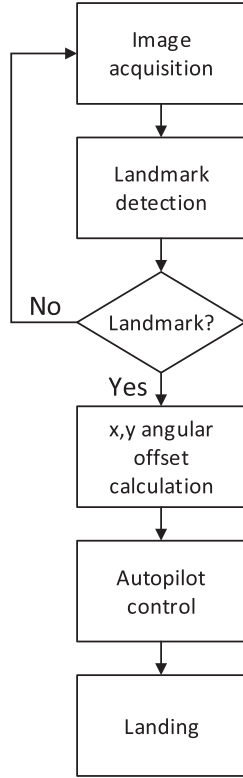


Figure 4. The flow chart of the vision-based MAV landing system.
MAV: micro aerial vehicle.

$$R_x = \frac{(x^* - hr/2) \times hf}{hr} \quad (14)$$

$$R_y = \frac{(y^* - vr/2) \times vf}{vr} \quad (15)$$

Here (x^*, y^*) denotes the coordinates detected by the trained detection model. hr , hf , vr and vf are on-board camera parameters. hr indicates the horizontal resolution, hf indicates the horizontal field of view (FoV), vr denotes the vertical resolution, and vf indicates the vertical FoV. Both hf and vf should be used in radians.

The R_x and R_y are sent to the autopilot as MAVLINK message. This information is processed with sonar data to compute landmark position relative to the MAV.

$$p_x = h_s \times \tan(R_x) \quad (16)$$

$$p_y = -h_s \times \tan(R_x) \quad (17)$$

where h_s is the height above the ground measured by sonar, and p_x and p_y are the x and y axis positions of the landmark relative to the MAV. The landmark

velocities v_x and v_y in the x and y axis relative to the MAV can be calculated by doing differential operations with p_x and p_y .

A Kalman filter is then exploited for supervising the landmark, with the state vector defined as $\mathcal{X} = [p_x, p_y, v_x, v_y]^T$. The landmark equations are modeled as a linear system as follows:

$$\mathcal{X}_{k+1} = A\mathcal{X}_k + w_k \quad (18)$$

$$Z_k = H\mathcal{X}_k + u_k \quad (19)$$

where \mathcal{X}_k is the true state vector describing the target position and velocity relative to the MAV at time k , A is the state transition parameter, w_k is the random process noise, Z_k is the measurement vector, H is the observation model, and u_k is the measurement noise. Let $p = [p_x, p_y]^T$ and $v = [v_x, v_y]^T$, and the motion can be characterized as follows:

$$p_{k+1} = p_k + v_k T_s + a_k T_s^2 / 2 \quad (20)$$

$$v_{k+1} = v_k + a_k T_s \quad (21)$$

where a_k is a random acceleration and T_s is the time step size. Furthermore, the Kalman state transition is:

$$\begin{bmatrix} p_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + w_k \quad (22)$$

Assume that the modeling noise w_k is white zero-mean Gaussian noise with a covariance matrix Q , and the measurement noise u_k is white zero-mean Gaussian noise with a covariance matrix R :

$$p(w) \sim N(0, Q) \quad (23)$$

$$p(v) \sim N(0, R) \quad (24)$$

The state propagation and update equations for the Kalman filter are summarized as follows.

- The predicted state estimation is:

$$\hat{\mathcal{X}}_{k|k-1} = A\hat{\mathcal{X}}_{k-1} \quad (25)$$

- The predicted estimation covariance is:

$$P_{k|k-1} = AP_{k-1}A^T + Q \quad (26)$$

- The innovation covariance is:

$$S_k = HP_{k|k-1}H^T + R \quad (27)$$

- The optimal Kalman gain is:

$$\mathcal{K}_k = P_{k|k-1} H^T S_k^{-1} \quad (28)$$

- The updated state estimation is:

$$\hat{\mathcal{X}}_k = \hat{\mathcal{X}}_{k|k-1} + \mathcal{K}_k (Z_k - H \hat{\mathcal{X}}_{k|k-1}) \quad (29)$$

- The posterior covariance is:

$$P_k = (I - \mathcal{K}_k H) P_{k|k-1} \quad (30)$$

In this group of equations, the superscript T indicates matrix transposition, $\hat{\mathcal{X}}_{k|k-1}$ means the predicted state estimate value, $\hat{\mathcal{X}}_{k-1}$ denotes the optimal state estimate value in the last step, $P_{k|k-1}$ indicates the covariance of the prediction error, \mathcal{K}_k is the Kalman gain matrix, P_k denotes the covariance of the posterior estimate error, and $\hat{\mathcal{X}}_k$ indicates the state estimate value.

The position and velocity estimate $\hat{\mathcal{X}}_k$ of the landmark relative to the MAV are fed into the position controller of the autopilot, and the command is generated to manipulate the MAV to land on the landmark safely.

Hardware architecture

The hardware architecture is shown in Figure 5. In this system, we develop a customized do-it-yourself (DIY) quad-rotor with an embedded development board (Manifold). As the main processing unit, the Manifold board carries out the following tasks: (1) processing all images captured by the on-board camera; (2) calculating the angular offset; (3) communicating with the autopilot. The communication between the Manifold and the MAV main body is enabled by the Dronekit.

In our system, the Manifold connects to the UAV's autopilot through USB TTL Serial cables. The baud

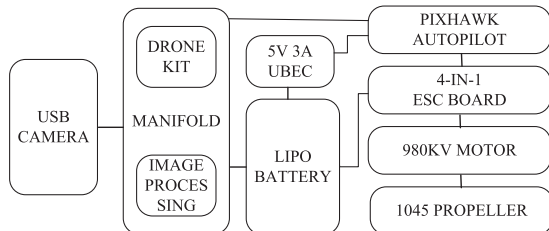


Figure 5. Hardware architecture of the vision-based autonomous landing system.

USB: universal serial bus.

rate of the serial connection is 1,500,000, and the MAVLINK is used for communication. The Dronekit API is utilized as the development tool for our system.

The experimental quadrotor is shown in Figure 6, with a DJI NAZA F450 X-shaped frame and the Pixhawk autopilot with ArduPilot Firmware. Pixhawk integrates a 14 bit accelerometer, a 16 bit gyroscope, a magnetometer and an MS5611 barometer. A sonar is used to measure the height of the MAV above the ground. A USB camera is used to detect the landmark. The camera and sonar are configured as shown in Figure 7. The key component of the UAV's computing system is the Manifold, which consists of a quad-core ARM Cortex-A15 processor and 2 GB memory.



Figure 6. The developed DIY quadrotor (from the top view of the MAV).

DIY: do-it-yourself; MAV: micro aerial vehicle.

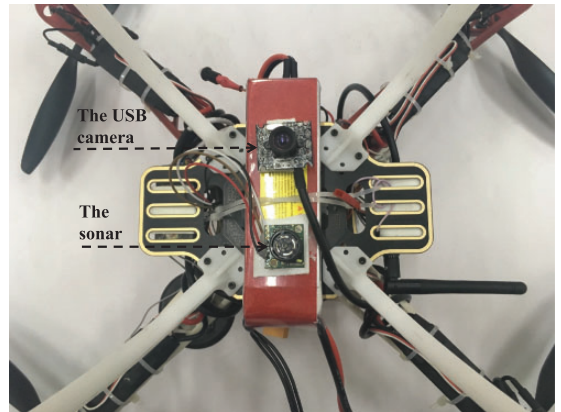


Figure 7. The configuration of camera and sonar (from the bottom view of the MAV).

MAV: micro aerial vehicle; USB: universal serial bus.

Experimental results

In the experiments, we use five categories of landmarks (as shown in Figure 1) for testing the performance of our framework. The landmarks size is $50\text{ cm} \times 50\text{ cm}$. The MAV with a downward looking camera is used to capture outdoor videos, and the flying height varies from 1 m to 5 m. We also use a forward looking camera for the MAV to capture indoor videos. The captured videos are separated into image frames. In order to train our model to be robust with respect to inconsistent lighting, we intentionally change image brightness and thus obtain various lighting training data. The brightness value ranges from 0 representing complete darkness to 100 representing complete whiteness. In our training process, the brightness values of 50 images from each landmark category are set to be 10, and those of another 50 images from each landmark category are set to be 90. We also use the data augmentation strategy presented in the previous section to enlarge the intraclass variability of the training data for the purpose of training a comprehensive model.

The images for each landmark category are annotated by using an open source tool – labelImg¹ <https://github.com/tzutalin/labelImg>. An obtained annotation file includes the category names and landmark coordinates. We build a training dataset containing images and their annotations. In our experiment, 200 images for each of the five landmark types are used to train our landmark detection model and the training data contains totally 1000 annotated images. The training process is realized on a workstation with a NVIDIA GTX860M GPU.

The values of IoU of our CNN model are shown in Figure 8. The IoU value reaches the peak after 35,000 iterations. We choose the CNN model with 35,000 iterations during the training procedure as our landmark detection model.

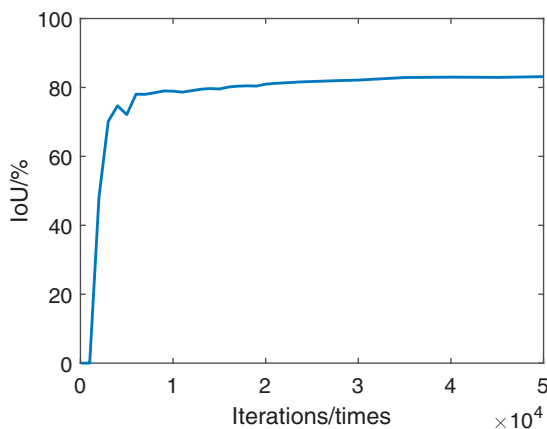


Figure 8. The values of IoU with different iterations. IoU: intersection over union.

In order to empirically evaluate the proposed on-board CNN-based landmark detection model, we use the test data sets, which are different from the training data sets, for testing our trained model. Specifically, we perform experiments on the Manifold board, which processes the outdoor and indoor real-time video frames as test data sets. Four tests are carried out:

- In the first set of experiments, we evaluate the robustness of the detection model for various landmark shapes.
- In the second set of experiments, we evaluate the robustness of the detection model for different illumination intensities.
- In the third set of experiments, we evaluate the robustness of the detection model for different background environments.
- In the fourth set of experiments, we evaluate the efficiency of the detection model.

Evaluation of the robustness of the detection model for various landmark shapes

To evaluate the effectiveness of our model for detecting various landmark shapes, we test our CNN model in terms of detecting each type of landmarks in 1000 images captured by the MAV camera. Results for the five landmarks are shown in Figure 9 and Table 2. Our CNN model successfully recognizes the landmarks in 4973 frames. The T-shaped landmark is mistaken for the H-shaped landmark in several rotation frames. This is because both the T-shaped landmark and the H-shaped landmark are simply featured by less discriminative black lines, and we only use 200 images of each type of landmarks to train our model. The landmark composed of an H-shaped and concentric circles is mistaken for the H-shaped landmark in several frames. We observe that these errors occur when the distance between the camera and the landmark is large. One reason for the misidentification of the H-shaped and circle landmarks is that they are symmetric shapes and are less distinguishable from distant views. To validate this observation, we use a fake target landmark, which appears similar to the circle landmark but is asymmetric, to replace the circle landmark for testing our model (Figure 9(f)). The experimental result reveals that our model does not misidentify the fake landmark as the true landmark. Our detection model can distinguish the landmark composed of a series of concentric circles and the fake targets from distant views correctly.

Qualitative evaluation results of the CNN model on the video frames captured from various flying heights and from different rotation angles are shown in Figure 10. The performance of our model is fairly stable when the MAV searching for the landmark at

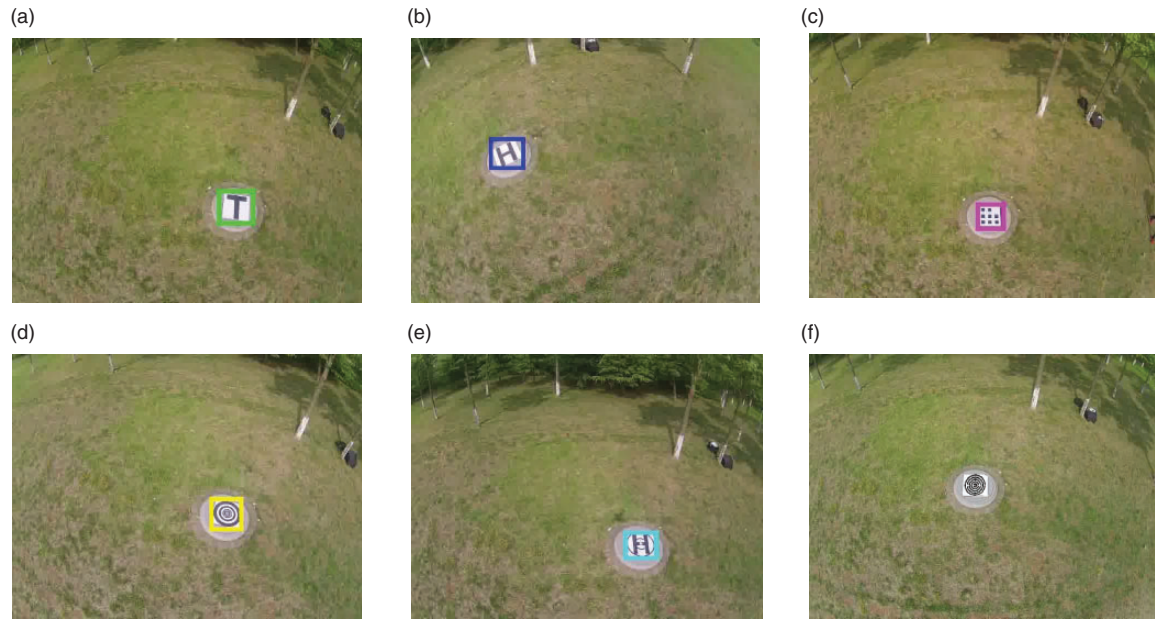


Figure 9. Landmark detection results: (a) the T-shaped landmark; (b) the H-shaped landmark; (c) the landmark consisting of eight equal-sized squares and a big white border; (d) the landmark composed of a series of concentric circles; (e) the landmark composed of an H-shaped and concentric circles; (f) the fake landmark.

Table 2. Results for evaluating the detection model for various landmark shapes.

	T-shaped	H-shaped	Square landmark	Circle landmark	Combined landmark
Number of test frames	1000	1000	1000	1000	1000
Correctly recognized frames	989	992	998	999	994
Detection accuracy	98.9%	99.2%	99.8%	99.9%	99.4%
Average time cost	47.53 ms	48.26 ms	47.35 ms	47.73 ms	48.62 ms

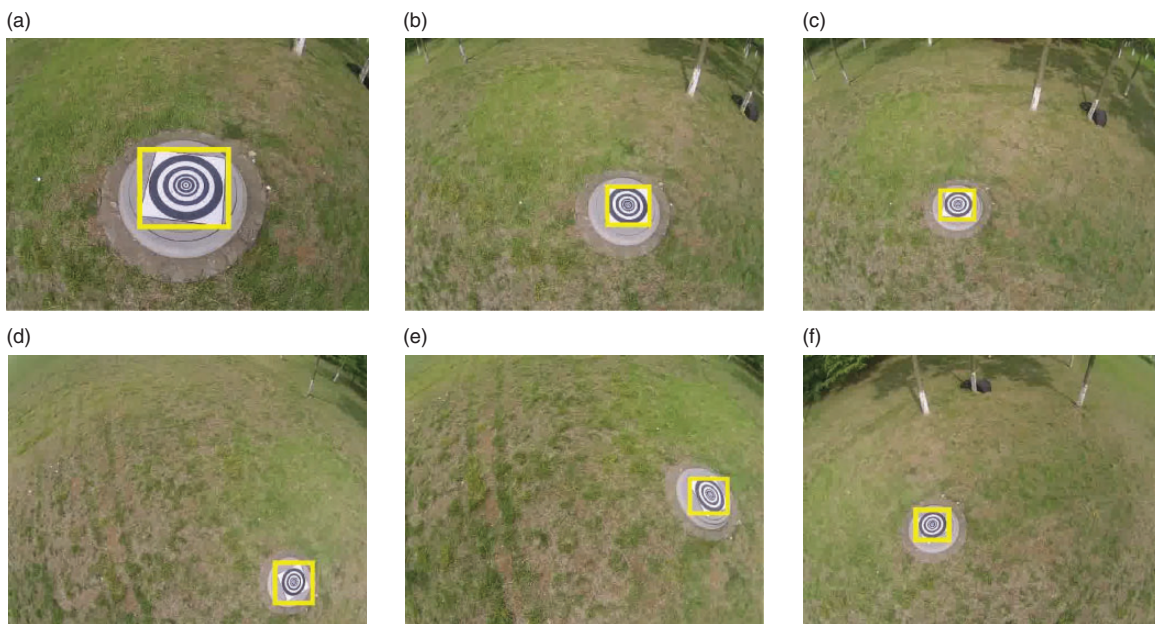


Figure 10. Detection results of the detection model on video frames captured from various flight heights ((a), (b) and (c)) and from different rotation angles ((d), (e) and (f)).

different heights and rotations. The processing rate is 21 frames per second.

To make the evaluation of the model robustness one step further, we use the MAV shown in Figure 6 to test the landing system with respect to different landmarks. The MAV takes off near the landmark to 4 m. When the landmark is detected, the vision-based landing system manipulates the MAV to autonomously land to the landmark region. All the actions in our test are performed by the MAV automatically. We test four flights for each landmark, and measure the horizontal distance between the MAV center to the landmark center in x and y axis. The position errors (e_x and e_y) are recorded in Table 3.

We can see from Table 3 that the position errors are acceptable for practical landing, because they are relatively small compared with the landmark size 50 cm \times 50 cm. One reason for the position errors is that we assume the roll and pitch angles of the MAV to be zero during the MAV landing for the purpose of making condition controlled evaluations. This assumptive

constraint causes some position measuring errors that do not arise from the detection model. Although suffering from these artificial errors, the MAV can still automatically land within the landmark region safely.

Evaluation of the robustness of the landmark detection model for different illumination intensities

In this set of experiments, 100 variously illuminated images for each landmark category are used for validation. Specifically, we generate different lighting situations by changing the brightness value of test images. The brightness values of the 100 images are set to be from 10 to 90 to test our landmark detection model. Visual results are shown in Figure 11. Experimental observations reveal that the detection results are stable for the MAV landmark navigation under various light conditions.

Evaluation of the robustness of the landmark detection model for different background environments.

To make the empirical evaluation of our model one step further, we test the detection performance of our method for detecting landmarks in different backgrounds. Specifically, we perform experiments on the videos captured in four different background

Table 3. Position errors.

	Mean	Variance	Max	Min
e_x	8.20 cm	14.88	14.35 cm	2.85 cm
e_y	9.11 cm	9.13	12.36 cm	4.05 cm

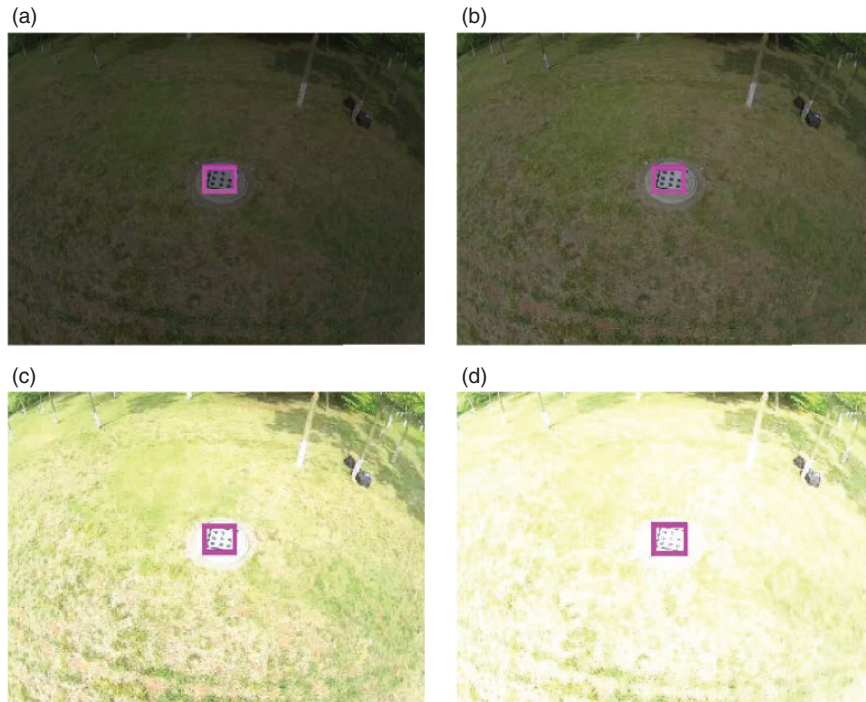


Figure 11. The detection results for the images with different brightness values: (a) the brightness value is 10; (b) the brightness value is 30; (c) the brightness value is 70; (d) the brightness value is 90.

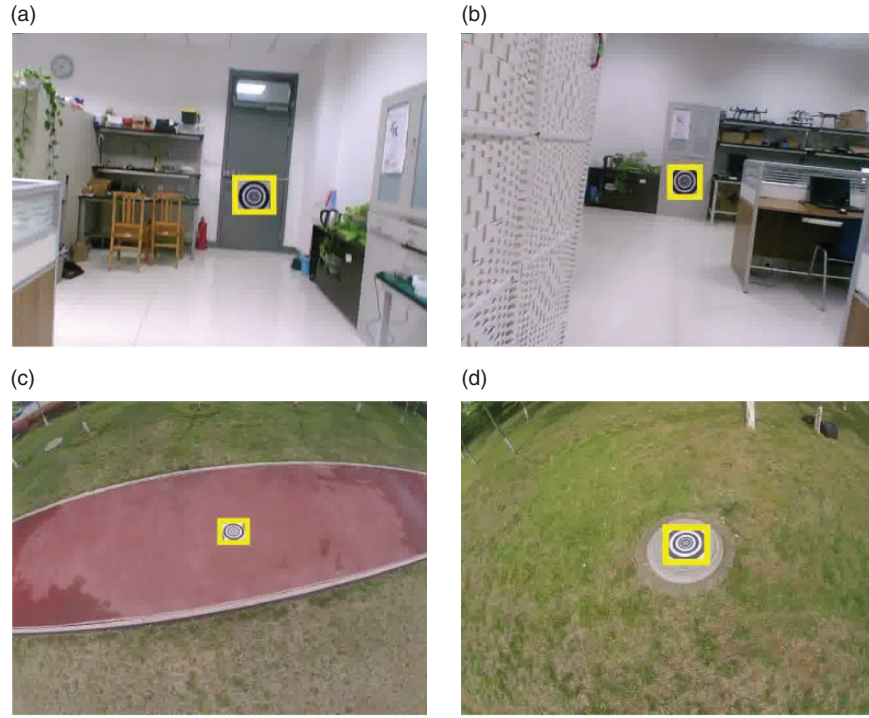


Figure 12. The landmark in various environments: (a) and (b) are indoor environments; (c) and (d) are outdoor environments.

Table 4. Results for evaluating the detection model for different background environments.

	T-shaped	H-shaped	Square landmark	Circle landmark	Combined landmark
Number of test frames	800	800	800	800	800
Correctly recognized frames	786	789	796	798	790
Detection accuracy	98.25%	98.625%	99.5%	99.75%	98.75%
Average time cost	47.62 ms	47.36 ms	47.25 ms	47.93 ms	48.32 ms

environments with the on-board camera. Two videos are captured outdoors, and the other two are captured indoors. We use 200 images captured in each environment to test our model. The results are shown in Figure 12 and Table 4. The experimental results validate that our landmark detection model is able to detect landmarks under various environments.

Evaluation of the efficiency of the landmark detection model

To make a quantitative comparison between our method and alternative state-of-the-art methods, we train the tiny models of Yolo¹⁷ and Yolo v2¹⁸ based on the same training dataset as that of our model. We then test the three trained models in terms of average IoU and processing rate (i.e. processed frames per second). The experiments are performed based on the same image frames as those in the first set of experiments. The comparison results are shown in Table 5.

Table 5. Comparison in terms of accuracy and efficiency.

	Yolo	Yolo v2	Our model
IoU	84.24%	89.29%	83.70%
Frames per second	7.5	5.3	21

IoU: intersection over union.

We observe that though the Yolo methods are slightly better in terms of accuracy, our model is much more efficient. The MAVs usually have limited computing resources on board (e.g. Manifold), which make the Yolo methods hardly achieve real-time implementation. On the other hand, our model achieves efficient implementation, which enables practical MAV on-board computation.

The usage of the CPU and memory the detection procedure is reported in Table 6. The landmark detection image processing algorithms use 3/4 of computing resources. Therefore, it allows more accurate control

Table 6. CPU and memory usage. CPU: central processing unit.

CPU1	CPU2	CPU3	CPU4	Memory usage
79%	74%	68%	68%	1407 MB/1892 MB

algorithms to execute during the landing procedure, which provides a possible route for improving the landing accuracy.

These experiments validate that our CNN model can not only detect the various landmarks, but also produce correct detections under different conditions. We put the detection results in video forms on the URL <https://youtu.be/fifCK6BeDH8> for public observation. It is clear that our method is robust and efficient to process landmark information for guiding the MAV autonomous landing.

Conclusions

We have introduced a novel vision guided MAV autonomous landing system based on deep learning. Specifically, we have made three novel contributions. First, we have incorporated a modified SqueezeNet architecture into the Yolo scheme to develop a simplified CNN for detecting landmarks. Second, we have designed a separative implementation strategy which leverages the complex CNN training and the instant CNN detection. We have tested our novel framework in both synthesized and real-world environments and validated its effectiveness for MAV autonomous landing.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship and/or publication of this article: This work was supported by National Natural Science Foundation of China (Grant No.: 61671481 and 61701541), Shandong Provincial Natural Science Foundation (Grant No.: ZR2017QF003), Qingdao Applied Fundamental Research Project (Grant No.: 16-5-1-11-jch), the Royal Society of Edinburgh and National Natural Science Foundation of China joint project 2017-2019 (Grant No.: 6161101383) and the Fundamental Research Funds for Central Universities (Grant No.: 15CX05042A and 16CX05004B).

References

1. Geng L, Zhang Y, Wang J, et al. Mission planning of autonomous UAVs for urban surveillance with evolutionary algorithms. In: *IEEE international conference on control and automation*, 2013, pp.828–833.
2. Yang S, Scherer SA and Zell A. An onboard monocular vision system for autonomous takeoff, hovering and landing of a micro aerial vehicle. *J Intell Robot Syst* 2013; 69: 499–515.
3. Tang D, Li F, Shen N, et al. UAV attitude and position estimation for vision-based landing. In: *International conference on electronic and mechanical engineering and information technology*, 2011, pp.4446–4450.
4. De Croon G, Ho H, De Wagter C, et al. Optic-flow based slope estimation for autonomous landing. *Int J Micro Air Vehicles* 2013; 5: 287–297.
5. De Croon G, De Clercq K, Ruijsink R, et al. Design, aerodynamics, and vision-based control of the delfly. *Int J Micro Air Vehicles* 2009; 1: 71–97.
6. Tsai AC, Gibbens PW and Stone RH. Terminal phase vision-based target recognition and 3D pose estimation for a tail-sitter, vertical takeoff and landing unmanned air vehicle. In: *Advances in image and video technology*. 2006, pp.672–681.
7. Saripalli S, Montgomery JF and Sukhatme G. Visually guided landing of an unmanned aerial vehicle. *IEEE Trans Robot and Autom* 2003; 19: 371–380.
8. Lin F, Chen BM and Tong HL. Vision aided motion estimation for unmanned helicopters in GPS denied environments. In: *Cybernetics and intelligent systems*. 2010, pp.64–69.
9. Verbandt M, Theys B and De Schutter J. Robust marker-tracking system for vision-based autonomous landing of vtol UAVs. In: *International micro air vehicle conference and competition*, 2014, pp.84–91.
10. Jung Y, Lee D and Bang H. Close-range vision navigation and guidance for rotary UAV autonomous landing. In: *IEEE international conference on automation science and engineering*, 2015, pp.342–347.
11. Fan Y, Haiqing S and Hong W. A vision-based algorithm for landing unmanned aerial vehicles. In: *International conference on computer science and software engineering*, 2008, pp.993–996.
12. Carrio A, Sampedro C, Rodriguez-Ramos A, et al. A review of deep learning methods and applications for unmanned aerial vehicles. *J Sensor* 2017;2: 1–13.
13. Felzenszwalb PF, Girshick RB, McAllester D, et al. Object detection with discriminatively trained part-based models. *IEEE Trans Pattern Anal Mach Intell* 2010; 32: 1627–1645.
14. Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *IEEE conference on computer vision and pattern recognition*, 2014, pp.580–587. *IEEE conference on computer vision and pattern recognition*, 2014, pp.2014.
15. Dai J, Li Y, He K, et al. R-FCN: object detection via region-based fully convolutional networks. In: *Advances in neural information processing systems*, 2016, pp.379–387.

16. He K, Gkioxari G, Dollár P, et al. Mask R-CNN. 2017, *arXiv preprint arXiv:1703.06870*.
17. Redmon J, Divvala S, Girshick R, et al. You only look once: unified, real-time object detection. In: *IEEE conference on computer vision and pattern recognition*, 2016, pp.779–788.
18. Redmon J and Farhadi A. Yolo9000: better, faster, stronger. 2016, *arXiv preprint arXiv:1612.08242*.
19. Simonyan K and Zisserman A. Very deep convolutional networks for large-scale image recognition. 2014, *arXiv preprint arXiv:1409.1556*.
20. He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp.770–778.
21. Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, inception-resnet and the impact of residual connections on learning. In: *AAAI*. 2017, pp.4278–4284.
22. Iandola FN, Han S, Moskewicz MW, et al. *Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 mb model size*, 2016, *arXiv preprint arXiv:1602.07360*.

Persistent self-supervised learning: From stereo to monocular vision for obstacle avoidance

International Journal of Micro Air
Vehicles
2018, Vol. 10(2) 186–206
© The Author(s) 2018
DOI: 10.1177/1756829318756355
journals.sagepub.com/home/mav
 SAGE

Kevin van Hecke¹, Guido de Croon¹, Laurens van der Maaten²,
Daniel Hennes³ and Dario Izzo³

Abstract

Self-supervised learning is a reliable learning mechanism in which a robot uses an original, trusted sensor cue for training to recognize an additional, complementary sensor cue. We study for the first time in self-supervised learning how a robot's learning behavior should be organized, so that the robot can keep performing its task in the case that the original cue becomes unavailable. We study this persistent form of self-supervised learning in the context of a flying robot that has to avoid obstacles based on distance estimates from the visual cue of stereo vision. Over time it will learn to also estimate distances based on monocular appearance cues. A strategy is introduced that has the robot switch from flight based on stereo to flight based on monocular vision, with stereo vision purely used as “training wheels” to avoid imminent collisions. This strategy is shown to be an effective approach to the “feedback-induced data bias” problem as also experienced in learning from demonstration. Both simulations and real-world experiments with a stereo vision equipped ARDrone2 show the feasibility of this approach, with the robot successfully using monocular vision to avoid obstacles in a 5×5 m room. The experiments show the potential of persistent self-supervised learning as a robust learning approach to enhance the capabilities of robots. Moreover, the abundant training data coming from the own sensors allow to gather large data sets necessary for deep learning approaches.

Keywords

Persistent self-supervised learning, stereo vision, monocular depth estimation, robotics

Received 19 May 2017; accepted 30 November 2017

Introduction

It is generally acknowledged that robots operating in the real world benefit greatly from learning mechanisms. Learning allows robots to adapt to environments or circumstances not specifically foreseen at design time. However, the outcome of learning and its influence on the learning robot's behavior can by definition not be predicted completely. This is a major reason for the delay in introducing successful learning methods such as Reinforcement Learning (RL) in the real world. For instance, with RL it is a major challenge to ensure an exploratory behavior that is safe for both the robot and its environment.¹

Learning from demonstration (LfD) can in this respect be regarded as more reliable. However, in the case of a mobile robot, LfD faces a “feedback-induced data bias” problem.^{2,3} If the robot executes its trained

policy on real sensory inputs, its actions will be slightly different from the expert's. As a result, the trajectory of the robot will be different to when the human expert was in control, leading to a test data distribution that is different from the training distribution. This difference worsens the performance of the learned policy, further

¹Micro Air Vehicle Lab, Control and Simulation, Faculty of Aerospace Engineering, TU Delft, Delft, Netherlands

²Faculty of Computer Science, TU Delft, Delft, Netherlands

³Advanced Concepts Team of the European Space Agency, ESTEC, Noordwijk, Netherlands

Corresponding author:

Kevin van Hecke and Guido de Croon, Micro Air Vehicle Lab, Control and Simulation, Faculty of Aerospace Engineering, TU Delft, Zoutmanstraat 62, Den Haag 2518 GS, Netherlands.
Emails: k.g.vanhecke@tudelft.nl; g.c.h.e.decroon@tudelft.nl



increasing the discrepancy between the data distributions. The solution proposed in Ross et al.^{2,3} is to have the human expert provide novel training data for the sensory inputs experienced by the robot when being in control itself. This leads to an iterative process that requires quite a time investment of the human expert.

There is a relatively new learning mechanism for robots that combines reliability with the advantage of not needing any human supervision. *Self-supervised learning (SSL)* does not learn a control policy as LfD and RL, but rather focuses on improving the sensory inputs used in control. Specifically, in SSL, the robot uses the outputs of an original, trusted sensor cue to learn recognizing an additional, complementary sensor cue. The reliability comes from the fact that the robot has access to the trusted cue during the entire learning process, ensuring a baseline performance of the system.

Until now, the purpose of SSL has mostly been the exploitation of the complementarity between the sensor cues. To illustrate, perhaps the most well-known example is the use of SSL on Stanley, the car that won the grand DARPA challenge.⁴ Stanley used a laser scanner to detect the road ahead. The range of the laser scanner was rather limited, which placed a considerable restriction on the robot's driving speed. SSL was used in order to extend the road detection beyond the range of the laser scanner. In particular, the laser scanner-based road classifications were used to train a color model of drivable terrain in the images from a camera. This color model was then applied to image regions not covered by the laser scanner. These image regions higher up in the image allowed to detect the road further away. The use of SSL permitted Stanley to speed up considerably and was an important factor in winning the competition.

A characterizing feature of many of the earlier SSL studies,⁴⁻¹⁰ is that the complementary cue is always used in combination with the original sensor cue. More recent studies aim to replace the function of the original cue with that of the complementary cue.¹¹⁻¹⁴ For instance, in Baleia et al.,¹¹ the sense of touch is used to teach a vision process how to recognize traversable paths through vegetation with the goal of gradually reducing time-intensive haptic interaction. Hence, the learning of recognizing the complementary cue will have to persist in time. However, the consequences of this persistent form of SSL on the robot's behavior when acting on the complementary cue have not been addressed in the above-mentioned studies.

The main contribution of this article is that we perform an in-depth study of the *behavioral* aspects of persistent SSL. We do so in the context of a scenario in which the robot should keep performing its task even when the supervisory cue becomes completely

unavailable. Importantly, when the robot relies only on the complementary cue, it will encounter the feedback-induced data bias problem known from LfD. We suggest a novel behavior strategy during learning to handle this problem in persistent SSL. Specifically, we study a flying robot with a stereo vision system that has to avoid obstacles in a global positioning system (GPS)-denied environment. The robot uses SSL to learn a mapping from monocular appearance cues to stereo-based distance estimates. We have chosen this context because it is relevant for any stereo-based robot that needs to be robust to a permanent failure of one of its cameras. In computer vision, monocular distance estimation is also studied. There, the main challenges are the gathering of sufficient data (e.g., for deep neural networks) and the generalization of the learned distance estimation to an unforeseen operation environment. Both of these challenges are addressed to some extent by SSL, as learning data are abundant and the robot learns in the environment in which it operates. We regard SSL as an important supplement to machine learning for robots. Therefore, we end the study with a discussion on the position of (persistent) SSL in the broader context of robot and machine learning, comparing it among others with RL, LfD, and supervised learning.

The remainder of the article is set up as follows. First, we discuss related work. Then, we more formally introduce persistent SSL and explain our implementation for the stereo-to-mono learning. We analyze the similarity of the specific SSL case studied in this article with LfD approaches. Subsequently, we perform offline vision experiments in order to assess the performance of various parameter settings. Thereafter, we compare various learning strategies in simulation. The best learning strategy is implemented for experiments with a Parrot ARDrone2, and the results of these robotic experiments are analyzed. Finally, the broader implications of the findings on persistent SSL are discussed, and conclusions are drawn.

Related work

We study persistent SSL in the context of a stereo-vision equipped robot that has to learn to navigate with a single camera. In this section, we discuss the state-of-the-art in the most relevant areas to the study: monocular navigation and SSL.

Monocular navigation

The large majority of monocular navigation approaches focuses on motion cues. The optical flow of world points allows for the detection of obstacles¹⁵ or even the extraction of structure from motion, as in

monocular simultaneous localization and mapping (SLAM).¹⁶ The main issue of using monocular motion cues for navigation is that optical flow only conveys information on the ratio between distance and the camera's velocity. Additional information is necessary for retrieving distance. This information is typically provided by additional sensors,¹⁶ but can also be retrieved by performing specific optical flow maneuvers.^{17,18}

In contrast, it is well known that the appearance of objects in a single, still image does contain distance information. Successfully estimating distances in a single image allows robots to evaluate distances without having to move. In addition, many appearance extraction and evaluation methods are computationally efficient. Both these factors can aid the robot in the making of quick navigation decisions. Below, we give an overview of work in the area of monocular appearance-based distance estimation and navigation.

Appearance-based navigation without distance estimation.

There are some appearance-based navigation methods that do not involve an explicit distance estimate. For instance, in de Croon et al.,¹⁹ an appearance variation cue is used to detect the proximity to obstacles, which is shown to be successful at complementing optical flow-based time-to-contact estimates. A threshold is set that makes the flying robot turn if the variation drops too much, which will lead to turns at different distances.

An alternative approach is to directly learn the mapping from visual inputs to control actions. In order to fly a quad rotor through a forest, Ross et al.³ use a variant of LfD²⁰ to acquire training data on avoiding trees in a forest. First a human pilot controls the drone, creating a training data set of sensory inputs and desired actions. Subsequently, a control policy is trained to mimic the pilot's commands as good as possible. This control policy is then implemented on the drone.

A major problem of this approach is the *feedback-induced data bias*: A robot has a feedback loop of actions and sensory inputs, so its control policy determines the distribution of world states that it encounters (with corresponding sensory inputs and optimal actions). Small deviations between the trained controller and the human may bring the robot in unknown states for which it has received no training. Its control policy may generalize badly to such situations. The solution proposed in Ross et al.³ is a transitional model called DAgger,² in which actions from the expert are mixed with actions from the trained controller. In the real-world experiments in Ross et al.,³ several iterations have been performed in which the robot flies with the trained controller, and the captured

videos are labeled offline by a human. This approach requires skilled pilots and significant human effort.

Many current studies focus on using deep RL²¹ to learn a mapping from images to actions. One of the most successful current attempts is the work in Sadeghi and Levine,²² which learns a deep neural network completely in simulation and then ports the network to a real robot in a yet unseen environment. The trained network performs quite admirably in the real environment. Key to the success of learning is to ensure a large variety of textures, lighting, and obstacle arrangements in simulation. Of course, if the real environment is still very different from the training environments, performance can degrade considerably.

Offline monocular distance learning. Humans are able to see distances in a single still image, and there is a growing body of work in computer vision utilizing machine learning to do the same. Interest in single image depth estimation was sparked by work from Hoiem et al.²³ and Saxena et al.^{24,25} Hoiem's automatic photo pop-up tries to group parts of the image into segments that can be popped up at different depths. Saxena's Make-3D uses a Markov random field (MRF) approach to classify a depth per image patch on different scales. These studies focus on creating a dense depth map with a machine learning computer vision approach. Both methods use supervised learning on a large training data set. Some work was done on adopting variants of Saxena's MRF work for driving rovers and even for MAVs. Lenz et al.²⁶ proposed a solution based on a MRF to detect obstacles on board an MAV, but it does not infer how far the objects are. Instead, it is trained offline to recognize three different obstacle class types. Any different objects could hence lead to navigation problems.

Recently, again focusing on creating a dense depth map from a single image, Eigen et al.²⁷ propose a multi-scale deep neural network approach trained on the KITTI data set, making it more resilient for practical robot data. Training deep neural networks require a large data set, which is often obtained by deforming training data or by artificially generating training data. Michels et al.²⁸ use artificial data to learn recognizing obstacles on a rover, but in order to generalize well it requires the use of a very realistic simulator. In addition, the same work reports significant improvement if the artificial data are augmented with labeled real-world data.

Other groups acquire training data for supervised learning by having another separate robot or system acquire data. This data are then processed and learned offline, after which the learned algorithm is deployed on the target robot. Dey et al.²⁹ use an RC car with a stereo vision setup to acquire data from an

environment, apply machine learning on this data offline, and navigate a similar but unseen environment with an MAV based on the trained algorithm. Creating and operating a secondary system designed to acquire training data, however, is no free lunch. Moreover, it introduces inconvenient biases in the training data, because an RC car will not behave in a similar way both in terms of dynamics, camera viewpoint, and the path chosen through the environment.

None of the above methods have the robot gather the data and learn while in operation.

Self-supervised learning

The idea of SSL has been around since the late 1990s,³⁰ but the successful application of it to terrain classification on the autonomously driving car Stanley³¹ demonstrated its first major practical use. A similar approach was taken by Hadsell et al.,⁹ but now using a stereo vision system instead of a LIDAR system, and complex convolutional filters instead of simple and fixed color-based features. These approaches largely forgo the need for manually labeled data as they are designed to work in unseen environments.

In most studies on SSL for terrain classification, the ground truth is always used during operation. In contrast, two very recent studies have as goal that the robot takes some decisions based on the complementary sensor cue alone. Since the complementary cue then has to persist in the absence of the original cue, this form of SSL can be termed “persistent SSL.” Baleia et al.¹¹ study a rover with a haptic antenna sensor. In their application of terrain mapping, they try to map monocular cues to obstacles based on earlier events of encountering similar situations that resulted in either a hard obstacle, a traversable obstacle, or a clear path. The monocular information is used in a path-planning task, requiring a cost function for either exploring unknown potential obstacles or driving through a terrain on the current available information. Since checking whether a potential obstacle is traversable is costly (the rover needs to travel there in order for the antenna to provide ground truth on that), the robot learns to classify the terrain ahead with vision. On each sample an analysis is performed to determine whether the vision-based classifier is sufficiently confident: it either decides the terrain is traversable, not traversable, or unsure. In the unsure case, the sample is sensed using the antenna. Gradually this will become necessary less often, thus learning to navigate using its Kinect sensor alone. In Ho et al.,¹² a flying robot first uses optical flow to select a landing site that is flat and free of obstacles. In order for this to work, the robot has to move sufficiently with respect to the objects on the landing site. While flying, the robot uses SSL to learn

a regression function that maps an (appearance-based) texton distribution to the values coming from the optical flow process. The learned function extends the capabilities of the robot, as after learning it is also able to select landing sites without moving (from hover). The article shows that if the robot is uncertain on its appearance-based estimates, it can switch back to the original optical flow-based cue.

The main contribution of this article is that we focus on the behavioral aspects of persistent SSL. We study how to best set up the behavior during the learning process, so that the robot will be able to keep performing its task when the original sensor cue becomes completely unavailable. Furthermore, we use stereo vision as the trusted, original cue, something which has not been done before in SSL. Concerning a comparison with the largest body of work on SSL that deals with terrain traversability classification, learning depth estimates directly is likely more complex. To illustrate, the robot would not only have to recognize sand, but it also has to make the difference between sand at 1-m distance and at 3-m distance. As mentioned above though, successful algorithms exist even to estimate complete dense depth maps from single images. In this article, since the emphasis is on behavior, we will deal with simplified environments and the estimation is restricted to the average depth in the field of view of the camera.

Methodology overview

In this section, we describe the persistent SSL learning mechanism and describe our implementation of this mechanism for our specific proof of concept case, monocular depth estimation in flying robots. Note that since our interest is in the behavioral aspects of persistent SSL, it is most important that the robot flies and learns in real time. Moreover, for the applicability to small flying robots, it is important that the processing for such SSL can be performed onboard, even considering the significant restrictions in onboard processing. The focus here is not yet on dealing with complex environments or on the accomplishment of missions like exploration or navigation. In this study, we investigate a very simple environment and obstacle avoidance behavior. We end this section with a description of the three learning behaviors that we compare with each other.

Persistent SSL principle

The persistent SSL principle is schematically depicted in Figure 1. In persistent SSL, an original, pre-wired sensory cue provides supervised outputs to a learning process that takes a different, complementary sensory cue as input. The goal is to be able to replace the

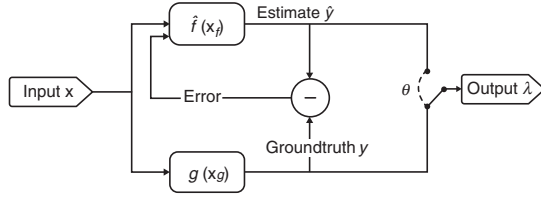


Figure 1. The persistent self-supervised learning principle.

pre-wired cue if necessary. When considering the system as a whole, learning with persistent SSL can be considered as unsupervised; it requires no manual labeling or pre-training before deployment in the field. Internally it uses a supervised learning method that in fact needs ground truth labels to learn. This ground truth is, however, assumed to be provided online and autonomously without human or outside interference.

In the schematic, the input variable \mathbf{x} represents the sensory inputs available on board. The variables x_g and x_f are possibly overlapping subsets of these sensory inputs. In particular, function $g(x_g)$ extracts a trusted ground truth sensory cue from the sensory inputs x_g . In classical systems, $g(x_g)$ provides the required functionality on its own:

$$g : x_g \rightarrow y, \quad x_g \subseteq \mathbf{x} \quad (1)$$

The function $f(x_f)$ is learned with a supervised learning algorithm in order to approximate $g(x_g)$ based on x_f :

$$f : x_f \rightarrow \hat{y}, \quad f \in F, \quad x_f \subseteq \mathbf{x} \quad (2)$$

$$\hat{f} = \underset{f \in F}{\operatorname{argmin}} \mathbb{E}[l(f(x_f), g(x_g))] \quad (3)$$

where $l(f(x_f), g(x_g))$ is a suitable loss function that is to be minimized. The system can either choose or be forced to switch θ , so that λ is either set to $g(x_g)$ or $\hat{f}(x_f)$ for use in control. Future work may also include fusing the two, but in this article, we focus on using the complementary cue in a stand-alone fashion. It must be noted that while both $x_g \subseteq \mathbf{x}$ and $x_f \subseteq \mathbf{x}$, in general it may be that x_f does not contain all necessary information to predict y . In addition, even if $x_g = x_f$, it is possible that F does not contain a function f that perfectly models g . The information in x_f and the function space F may not allow for a perfect estimate of $g(x_g)$. On the other hand, there may be an $f(x_f)$ that handles certain situations better than $g(x_g)$ (think of landing site selection from hover, as in Ho et al.¹²). In any case, fundamental differences between $g(x_g)$ and $\hat{f}(x_f)$ are to be expected, which may significantly influence the

behavior when switching θ . Handling these differences is of central interest in this article.

Stereo-to-mono proof of concept

Figure 2 presents a block diagram of the proposed proof of concept system in order to visualize how the persistent SSL method is employed in our application: estimating monocular depth with a flying robot. Input is provided by a stereo vision camera, with either the left or right camera image routed to the monocular estimator. We use a Visual Bag of Words (VBoW) method for this estimator. The ground truth for persistent SSL in this context is provided by the output of a stereo vision algorithm. In this case, the average value of the disparity map is used, both for training the monocular estimator and as an input to the switch θ . Based on the switch, the system either delivers the monocular or the stereo vision average disparity to the behavior controller.

Stereo vision processing

The stereo camera delivers a synchronized gray-scale stereo-pair image per time sample. A stereo vision algorithm first computes a disparity map, but often this is a far from perfect process. Especially in the context of an MAV's size, weight and computational constraints, errors caused by imperfect stereo calibration, resolution limits, etc. can cause large pixel errors in the results. Moreover, learning to estimate a dense disparity map, even when this is based on a high quality and consistent data set, is already very challenging. Since we use the stereo result as ground truth for learning, we minimize the error by averaging the disparity map to a single scalar. A single scalar is much easier to learn than a full depth map and has been demonstrated to provide elementary obstacle avoidance capability.^{28,32,33}

The disparity λ relates to the depth d of the input image:

$$d \propto \frac{1}{\lambda} \quad (4)$$

Using averaged disparity instead of averaged depth fits the obstacle avoidance application better, because small but close by objects are emphasized due to the nonlinear relation of equation (4). However, linear learning methods may have difficulty mapping this relation. In our final design, we thus choose to learn the disparity with a nonparametric approach, which is resilient to nonlinearities.

Monocular disparity estimation

The monocular disparity estimator forms a function from the image's pixel values to the average disparity in the image. Since the main goal of the article is to study SSL on board a drone in real time, efficiency of both the learning and execution of this function is very important. Hence, we converged to a computationally extremely efficient VBoW approach for the robotic experiments. We have also explored a deep neural network approach, but the hardware and time available for learning did not allow for having the deep neural learning on board the drone at this stage.

The VBoW method uses small image patches of $w \times h$ pixels, as successfully introduced in Varma and Zisserman³⁴ for a complex texture classification problem. First, a dictionary is created by clustering the image patches with Kohonen clustering (as in De Croon et al.³³). The n cluster centroids are called “textons.” In this work, two types of textons are used: normal intensity textons and gradient textons obtained similarly but based upon the gradient of the images. Gradient textures have been shown in Wu et al.³⁵ to be an important depth cue. An example dictionary of each is depicted in Figure 3. Gradient

textons are shown with a color range (from blue = low, to red = high). The intensity textons in Figure 3 are based on grayscale intensity pixel values.

When an image is received, m patches are extracted from the $W \times H$ pixel image. Each patch is compared to the dictionary by means of a distance function, in order to form a texton occurrence histogram for the image; the texton bin with the smallest Euclidean distance to a given patch is increased by 1. The histogram is normalized to sum to 1. Then, each normalized histogram is supplemented with its Shannon entropy, resulting in a feature vector of size $n + 1$. The idea behind adding the entropy is that the variation of textures in an image decreases when the camera gets closer to obstacles.³³ To illustrate the change in texton histograms when approaching an obstacle, a time series of texton histograms can be seen in Figure 4. Note how the entropy of the distribution indeed decreases over time, and that especially the fourth bin is much higher when close to the poster on the wall. A machine learning algorithm will have to learn to notice such relationships itself for the robot's environment, by learning a mapping from the feature vector to a disparity scalar. We have investigated different function representations and learning methods to this end.

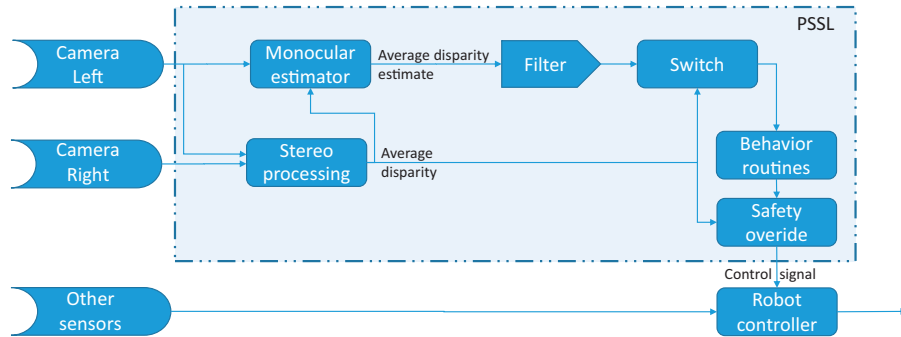


Figure 2. System overview. See the text for details.

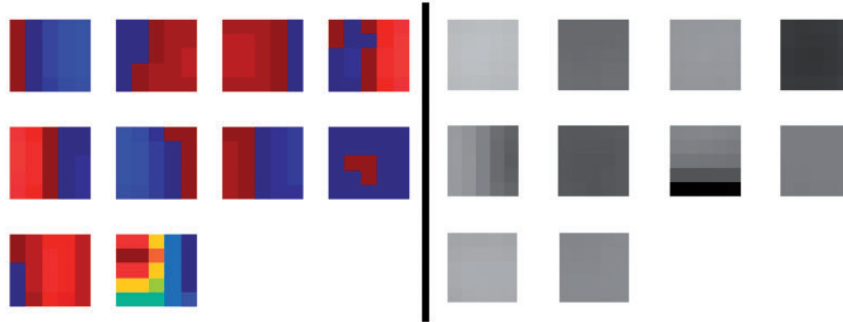


Figure 3. The texton library used in the experiments. The right set of textons is based on pixel intensities, the left set contains (artificially colored) gradient textons (i.e., textons based on gradient images).

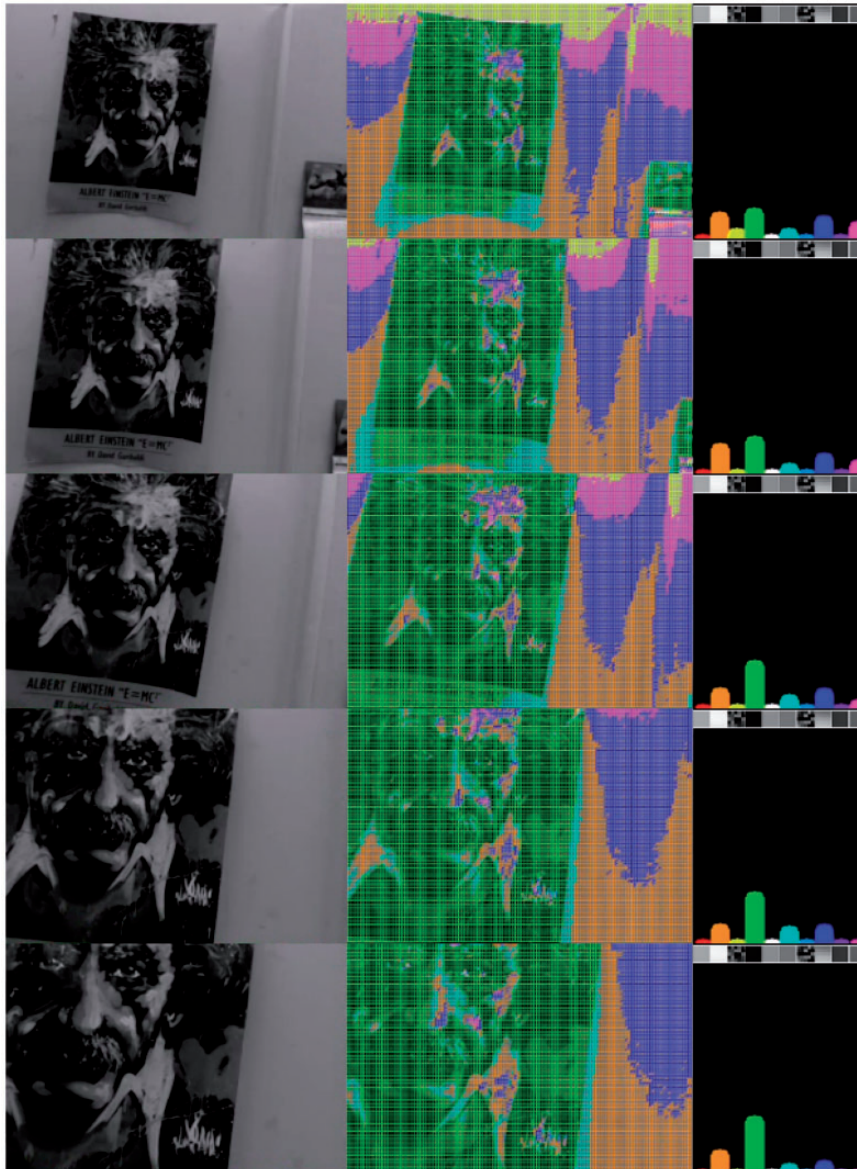


Figure 4. Approaching a poster on the wall. Left: monocular input. Middle: overlaid textons annotated with the color used in the histogram. Right: texton distribution histogram with the corresponding texton shown beneath it.

Control behavior

The proposed system uses a straightforward behavior heuristic to explore, navigate, and persistently learn a room. The heuristic is depicted as a finite state machine (FSM) in Figure 5. The FSM detects obstacles by means of a threshold t applied to the average disparity λ . In state 0, the robot flies in the direction of the camera's principal axis. When an obstacle is detected ($\lambda > t$), the robot stops and goes to state 1 in which it randomly chooses a new direction for the principal axis. It immediately passes to state 2 in which the robot rotates toward the new direction, reducing the error e between the principal axis' current and desired

direction. If in the new direction obstacles are far enough away ($\lambda \leq t$), the robot starts flying forward again (state 0). Else, the robot continues to turn in the same direction as before (clockwise or counter clockwise) until $\lambda \leq t$. When this is the case, it starts flying straight again (state 0). Choosing this rather straightforward behavior heuristic enables autonomous exploration based on only one scalar obtained from a distance sensor.

Performance

The average disparity λ , coming either from stereo vision or from the monocular distance estimation

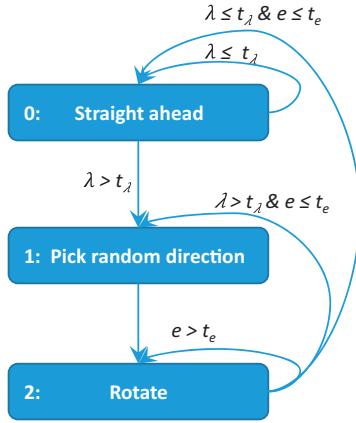


Figure 5. The behavior heuristic FSM. λ is average disparity, e is the attitude error (meaning the difference between the newly picked direction and the current attitude), t_n the respective thresholds.

function $f(x_f)$, is thresholded for determining whether to turn. This leads to a binary classification problem, where all samples for which $\lambda > t$ are considered as “positive” ($c=1$) and all samples for which $\lambda \leq t$ are considered as “negative” ($c=0$). Hence, the quality of $\hat{f}(x_f)$ can be characterized by a receiver operating characteristic (ROC) curve. The ground truth for the ROC curve is determined by the stereo vision. This means that a true positive ratio (TPR) of 1 and false positive ratio (FPR) of 0 lead to the same obstacle detection performance as with the stereo vision system. Generally, of course, this performance will not be reached, and the robot has to determine what threshold to set for a sufficiently high TPR and low FPR.

This leads to the question how to determine what a “sufficient” TPR/FPR is. We evaluate this matter in the context of the robot’s obstacle avoidance task. In particular, we first look at the probability of a collision with a given TPR and then at the probability of a spurious turn with a given FPR.

In order to model the probability of a collision, consider a constant velocity approach with input samples (images) $\langle x_1, x_2, \dots, x_n \rangle$ of n samples long, ending at an obstacle. A minimum of u samples before actual impact, an obstacle must be detected by at least one TP or a FP in order to prevent a collision. Since the range of samples $\langle x_{(n-u+1)}, x_{(n-u+2)}, \dots, x_n \rangle$ does not matter for the outcome, we redefine the approach range to be $\langle x_1, x_2, \dots, x_{(n-u)} \rangle$. Consider that for each sample x_i holds:

$$1 = p(TP|x_i) + p(FP|x_i) + p(TN|x_i) + p(FN|x_i), \quad (5)$$

since $p(FN|x_i) = p(TP|x_i) = 0$ if x_i is a negative and $p(TN|x_i) = p(FP|x_i) = 0$ if x_i is a positive. Let us first

assume independent, identically distributed (i.i.d.) data. Then, the probability of a collision p_c can be written as:

$$\begin{aligned} p_c &= \prod_{i=1}^{n-u} (p(FN|x_i) + p(TN|x_i)) \\ &= \prod_{i=1}^q p(TN|x_i) \prod_{i=q+1}^{n-u} p(FN|x_i), \end{aligned} \quad (6)$$

where q is a time step separating two phases in the approach. In the first phase all x_i are negative, so that any false positive will lead to a turn, preventing the collision. Only if all negative samples are correctly classified as negatives (true negatives), will the robot enter the second phase in which all x_i are positive. Then only a complete sequence of false negatives will lead to a collision, since any true positive will lead to a timely turn.

We can use equation (6) to choose an acceptable $TPR = 1 - FNR$. Assuming a constant velocity and frame rate, it gives us the probability of a collision. For instance, let us assume that the robot flies forward at 0.50 m/s with a frame rate of 30 Hz. To avoid collisions, it has a minimal required detection distance of 1.0 m, while “positives” are defined to be closer than 1.5 m. This leads to 0.5 m of flight during which the robot can detect an oncoming collision, corresponding to $s=30$ samples that all have to be classified as (false) negatives for a collision to occur. In the case of i.i.d. data, if we think a probability $p_c \leq 10^{-6}$ is acceptable, then the desired $TPR \geq 1 - 2(\log_2(10^{-6})/30) \sim 0.369$.

The analysis of the effect of false positives is straightforward, as it can be expressed in the number of spurious turns per second or, equivalently if assuming a constant velocity, per meter traveled. With the same scenario as above, an $FPR=0.05$ will on average lead to three spurious turns per traveled meter, which is unacceptably high. An $FPR=0.0017$ will approximately lead to 1 spurious turn per 10 m.

The above analysis seems to indicate that quite many false negatives are acceptable, while there can only be very few false positives. However, there are two complicating factors. The first factor is that equation (6) only holds when X can be assumed i.i.d., which is unlikely due to the nature of the consecutive samples of an approach toward an obstacle. Some reasoning about the nature of the dependence is, however, possible. Assuming a strong correlation between consecutive samples results in a higher probability of x_i being classified the same as $x_{(i+1)}$. In other words, if a sample in the range $\langle x_1 \dots x_m \rangle$ is an FN, the chance increases that more samples are FNs. Hence, the expected dependencies significantly impact performance of the system making equation (6a) best case scenario for FNs and a worst case scenario for FPs.

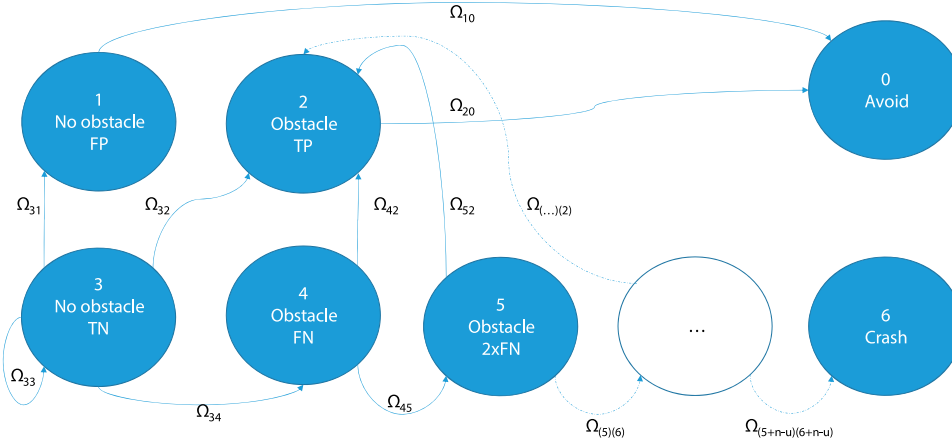


Figure 6. A Markov model of the probability of a collision. Due to the nature of the consecutive samples, state transition probabilities $\Omega_{(5+n)(6+n+u)}$ are not equal to $\Omega_{(3)(4)}$ but are likely to be relatively high. (Once one frame was wrongly classified as no obstacle, it is likely the upcoming frames will also be wrongly classified as they are similar.)

The system can be more realistically modeled as a Markov process as depicted in Figure 6. From this it can be seen that the system can be split in a reducible Markov process with an absorbing *avoid* state, and a chain of states that leads to the absorbing *collision* state. The values of the transition matrix Ω can be determined from the data gathered during operation. This would allow the robot to better predict the consequences of a chosen *TPR* and *FPR*.

As an illustration of the effects of sample dependence, let us suppose a model in which each classification has a probability of being identical to the previous classification, $p(I(c_{i-1}, c_i))$. If not identical, the sample is classified independently. This dependency model allows us to calculate the transition $\Omega_{4,5}$ in Figure 6. Given a previous negative classification, the transition probability to another negative classification is: $\Omega_{4,5} = p(I(c_{i-1}, c_i)) + (1 - p(I(c_{i-1}, c_i)))(1 - TPR)$. If $p(I(c_{i-1}, c_i)) = 0.8$ and $TPR = 0.95$, $\Omega_{4,5} = 0.81$. The probability of a collision in such a model is $p_c = \Omega_{4,5}^{(s-1)} = 1.8 \cdot 10^{-3}$, no longer an inconceivably small number.

This leads us to the second complicating factor, which is specific to our SSL setup. Since the robot operates on the basis of the ground truth, it should theoretically hardly ever encounter positive samples. Namely, the robot should turn when it detects a positive sample. This implies that the uncertainty on the estimated *TPR* is rather high, while the *FPR* can be estimated better. A potential solution to this problem is to purposefully have the mono-estimation robot turn earlier than the stereo vision-based one.

Similarity with LfD

The core of SSL is a supervised algorithm that learns the function $\hat{f}(x_f)$ on the basis of supervised

outputs $g(x_g)$. Normally, supervised learning assumes that the training data are drawn from the same data probability distribution \mathcal{D} as the test data. However, in persistent SSL, this assumption generally does not hold. The problem is that by using control based on \hat{f} , the robot follows a control policy $\pi_{\hat{f}} \neq \pi_g$ and hence will induce a different state distribution, $\mathcal{D}_{\pi_{\hat{f}}} \neq \mathcal{D}_{\pi_g}$. On these different states, no supervised outputs have been observed yet, which typically implies an increasing difference between \hat{f} and g .

A similar problem of inducing a different state distribution is well known in the area of LfD.^{2,3} Actually, we will show that under some mild assumptions, the persistent SSL problem studied in this paper is equivalent to an LfD problem. Hence, we can draw on solutions in LfD such as DAGGER.²

The goal of LfD is to find a policy $\hat{\pi}$ that minimizes a loss function l under its induced distribution of states, from Ross et al.²:

$$\hat{\pi} = \underset{\pi \in \Pi}{\operatorname{argmin}} \mathbb{E}_{s \sim \mathcal{D}_{\pi}} [l(s, \pi)] \quad (7)$$

where an optimal, teacher policy π^* is available to provide training data for specific states s .

At first sight, SSL is quite different, as it focuses only on the state information that serves as input to the policy. Instead of optimizing a policy, the supervised learning in persistent SSL can be defined as finding the function f' that best matches the trusted function g' under the distribution of states induced by the use of the thresholded version f for control:

$$\underset{f' \in F}{\operatorname{argmin}} \mathbb{E}_{x \sim \mathcal{D}_{\pi_f}} [l(f'(x), g(x))] \quad (8)$$

meaning that we perform regression of f on states that are induced by the control policy π_f , which uses the thresholded version of f .

To see the similarity to equation (7), first realize that the stereo-based policy is in this case the teacher policy: $\pi_g = \pi^*$. For this analysis, we simplify the strategy to flying straight when far enough away from an obstacle, and turning otherwise:

$$\begin{aligned} \pi_g(s) : \quad & p(\text{straight}|s=0) = 1 \\ & p(\text{turn}|s=1) = 1 \end{aligned} \quad (9)$$

where s is the state, with $s=1$ when $g(x) > t_g$ and $s=0$ otherwise. Note that π_g is a deterministic policy, which is assumed to be optimal.

When we learn a function \hat{f} , it generally will not give exactly the same outputs as g . Using $\hat{s} := \hat{f} > t_f$ will result in the following stochastic policy:

$$\begin{aligned} \pi_{\hat{f}}(s) : \quad & p(\text{straight}|s=0) = TNR \\ & p(\text{turn}|s=0) = FPR \\ & p(\text{turn}|s=1) = TPR \\ & p(\text{straight}|s=1) = FNR \end{aligned} \quad (10)$$

a stochastic policy which by definition is optimal, $\pi_{\hat{f}} = \pi_g$, if $FPR = FNR = 0$. In addition, then $\mathcal{D}_{\pi_{\hat{f}}} = \mathcal{D}_{\pi_g}$. Thus, if we make the assumption that minimizing $l(f(x), g(x))$ also minimizes FPR and FNR , capturing any preference for one or the other in the cost function for the behavior $l(s, \pi)$, then minimizing f in equation (8) is equivalent to minimizing the loss in equation (7).

Learning schemes

The interest of the above-mentioned similarity lies in the use of proven techniques from the field of LfD for training the persistent SSL system. In this article, we study a well-known method from this field, named DAgger,² and compare it with two additional methods. All three learning schemes start with an initial learning period in which the drone is controlled purely by means of stereo vision. The three methods are different though as follows.

1. In the first learning scheme, the drone will continue to fly based on stereo vision for the remainder of the learning time. After learning, the drone immediately switches to monocular vision. For this reason, the first scheme is referred to as “cold turkey.”
2. In the second learning scheme, the drone will perform a stochastic policy, selecting the stereo

vision-based actions with a probability β_i and monocular-based actions with a probability $(1 - \beta_i)$, as was proposed in the original DAgger article.² In the experiments, $\beta_i = 0.25$.

3. In the third learning scheme, the drone will perform monocular-based actions, with stereo vision only used to override these actions when the drone gets too close to an obstacle. Therefore, we refer to this scheme as “training wheels.”

Offline vision experiments

In this section, we perform offline vision experiments. The goal of these experiments is to determine how good the proposed VBoW method is at estimating monocular depth, and to determine the best parameter settings.

To measure the performance, we use two main metrics: the mean square error (MSE) and the area under the curve (AUC) of an ROC curve. MSE is an easy metric that can be directly used as a loss function, but in practice many situations exist in which a low MSE can be achieved while inadequate performance is reached for the basis of reliable MAV behavioral control. The AUC captures the trade-off between TPR and FPR and hence is a good indication of how good the performance is in terms of obstacle detection.

We use two data sets in the experiments. The first data set is a video made on a drone during an autonomous flight using an onboard 128×96 pixels stereo camera. The second data set is a video made by manually walking with a higher quality 640×480 pixel stereo camera through an office cubicle in a similar fashion as the robot should move in the later online experiments. The data sets #1 and #2 used in this section are made available for download publicly.³⁶ An example image from each data set is shown in Figure 7.

Our implementation of the VBoW method has six main parameters, ranging from the number of intensity and gradient textons to the number of samples used to smooth the estimated disparity over time. An exhaustive search of parameters being out of reach, we have performed various investigations of parameter changes along a single dimension. Table 1 presents a list of the final tuned parameter values. Note that these parameter values have not only been optimized for performance. Whenever performance differences were marginal, we have chosen the parameter values that saved on computational effort. This choice was guided by our goal to perform the learning on board of a computationally limited drone. Below we will show a few of the results when varying a single parameter, deviating from the settings in Table 1 in the corresponding dimension.

Figure 8 shows the results for different numbers of textons, $\in \{4, 8, 12, 16, 20\}$, always consisting half out of pixel intensity and half of gradient textons. From the



Figure 7. Example from data set #1 (left, 128×96 pixels) and data set #2 (right, 640×480).

Table 1. Parameter settings.

Parameter	Value
Number of intensity textons	10
Number of gradient textons	10
Patch size	5×5
Subsampling samples	500
kNN	$K=5$
Smooth size	4

results, we can see that the performance saturates around 20 textons. Hence we selected this combination of 10 intensity and 10 gradient textons for the experiments.

The VBoW method involves choosing a regression algorithm. In order to determine the best learning algorithm, we have tested four regression algorithms, limiting the choice mainly based on feasibility for implementing the regression algorithm on board a constrained embedded system. We have tested two non-parametric (kNN and Gaussian process regression) and two parametric (linear and shallow neural network regression) algorithms. Figure 9 presents the learning curves for a comparison of these regressors. Clearly, in most cases the kNN regression comes out best. A naive implementation of kNN suffers from having a larger training set in terms of CPU usage during test time, but after implementation on the drone, this did not become a bottleneck.

The final offline results on the two data sets are quite satisfactory. They can be viewed online (note 1). After a training set of roughly 6000 samples, the kNN approximates the stereo vision-based disparities in the test set rather well. Given a desired TPR of 0.82, the learner has an FPR of 0.26. Considering the high inter-dependability of concurrent frames, this should be sufficient for usage of the estimated disparities in control.

Simulation experiments

We argued that a persistent form of SSL is similar to LfD. The relevance of this similarity lies in the

behavioral schemes used for learning. In this section, we compare the three learning schemes, as introduced in the section Learning Schemes, in simulation.

Setup

We simulate a “flying” drone with stereo vision camera in SmartUAV,³⁷ an in-house developed simulator that allows for 3D rendering and simulation of the sensors and algorithms used on board the real drone. Figure 10 shows the simulated “office room.” The room has a size of 10×10 m, and the drone has an average forward speed of 0.5 m/s. All the vision and learning algorithms are exactly the same as the ones that run on board of the drone in the real experiments.

We compare the three learning schemes, cold turkey, Dagger, and training wheels, in simulation. As mentioned, these schemes all have the same initial training period with stereo vision being in control, but they differ in the remaining learning period. After all learning, the drone will use its monocular disparity estimates for control. The stereo vision remains active only for overriding the control if the drone gets too close to a wall. During this testing period, we register the number of turns and the number of overrides. The number of overrides is a measure of the number of potential collisions. The performed number of turns during testing is compared to the number of turns performed when solely using stereo vision, to evaluate the number of spurious turns. The initial learning period is 1 min, the remaining learning period is 4 min, and the test time is 5 min. These times have been selected to allow a full experiment on a single battery of the real drone.

Results

Table 2 contains the results of 30 experiments with the three learning schemes and a purely stereo-vision-controlled drone. The first observation is that “cold turkey” gives the worst results. This result was to be expected on the basis of the similarity between persistent SSL and LfD: the learned monocular distance estimates do not generalize well to the test distribution

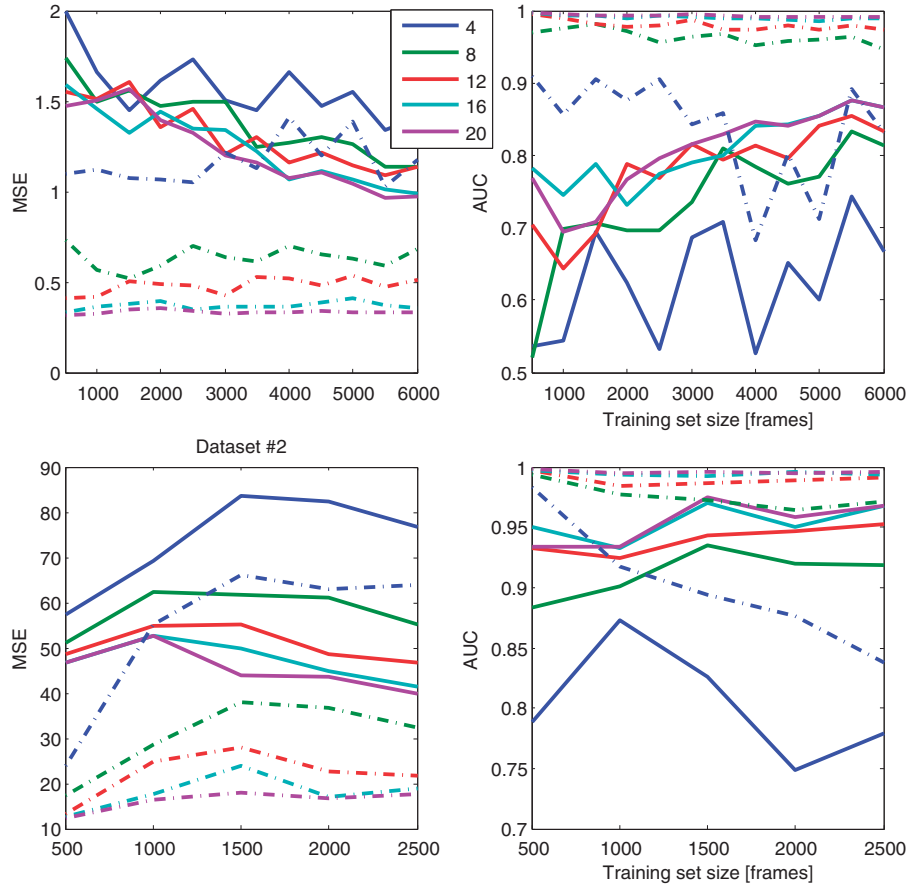


Figure 8. MSE and AUC number of textons. Dashed/solid lines refer to results on train/test set.

when the monocular vision is in control. The originally proposed DAGger scheme performs better, while the third learning scheme termed “training wheels” seems most effective. The third scheme has the lowest number of overrides of all learning schemes, with a similar total number of turns as a pure stereo vision run. The intuition behind this method being best is that it allows the drone to best learn from samples when the drone is beyond the normal stereo vision turning threshold. The original DAGger scheme has a larger probability to turn earlier, exploring these samples to a lesser extent. Double-sided statistical bootstrap tests³⁸ indicate that all differences between the learning methods are significant with $p < 0.05$.

The differences between the learning schemes are well illustrated by the positions the drone visits in the room during the test phase. Figure 11 contains “heat maps” that show the drone positions during turning (top row) and during straight flight (bottom row). The position distribution has been obtained by binning the positions during the test phase of all 30 runs. The results for each scheme are shown per column in Figure 11. Right is the pure stereo vision scheme, which shows a clear border around the straight flight

trajectories. It can be observed that this border is best approximated by the “training wheels” scheme (second from the right).

Robotic experiments

The simulation experiments showed that the “training wheels” setup resulted in the fewest stereo vision overrides when switching to monocular disparity estimation control. In this section, we test this online learning setup with a flying robot.

The experiment is set up in the same manner as the simulation. The robot, a Parrot ARDrone2, first explores the room with the help of stereo vision. After 1 min of learning, the drone switches to using the monocular disparity estimates with stereo vision running in the background for performing potential safety overrides. In this phase, the drone still continues to learn. After learning 4 to 5 min, the drone stops learning and enters the test phase. Again, also for the real robot the main performance measure consists of the number of safety overrides performed by the stereo vision during the testing phase.

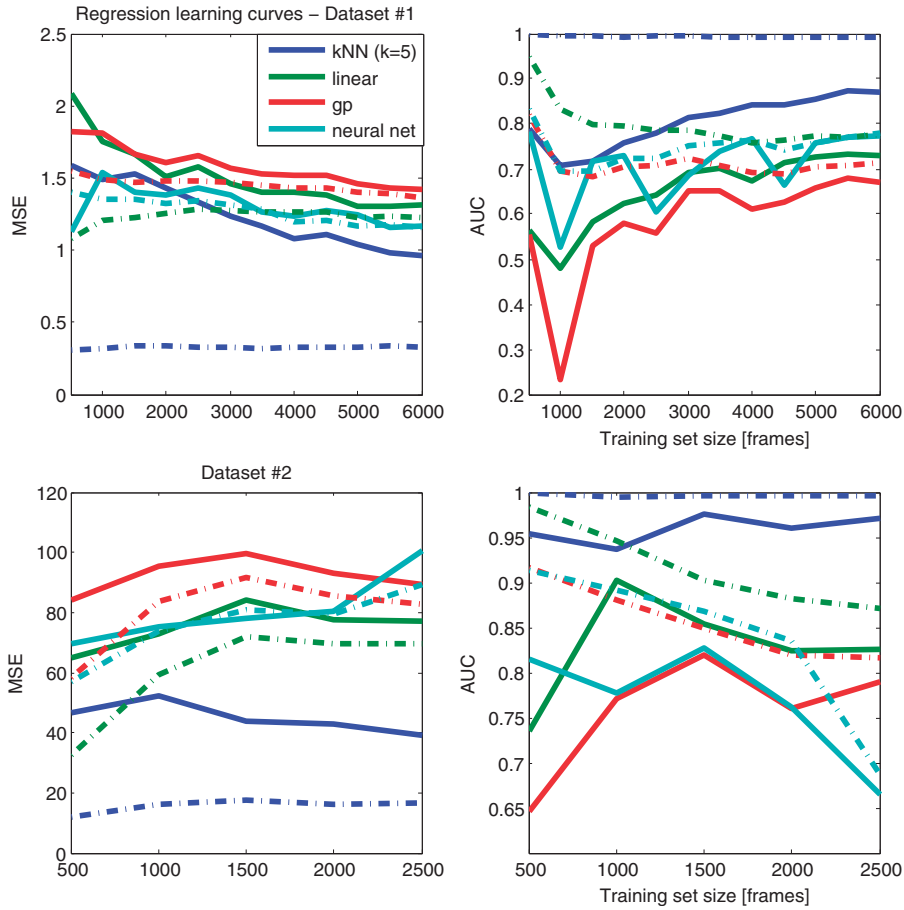


Figure 9. VBoW regression algorithms learning curves. Dashed/solid lines refer to results on train/test set.

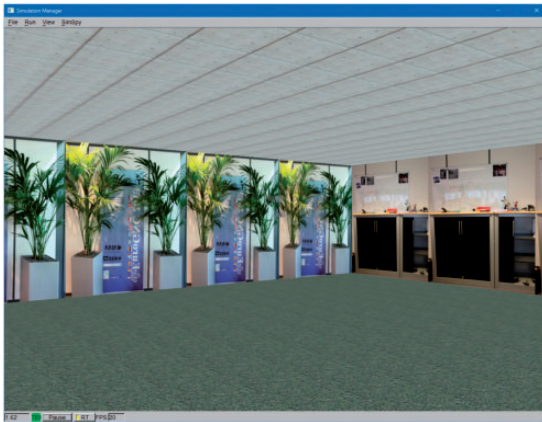


Figure 10. SmartUAV simulation environment.

The ARDrone2 is standard not equipped with a stereo vision system. Therefore, an in-house-developed 4 g stereo vision system is used,³⁹ which sends the raw images over USB to the ARDrone2 (see figure 12). The grayscale stereo camera has a resolution of 128×96 px and is limited to 10 fps. The ARDrone2 comes with a 1 GHz ARM cortex A8 processor and 128 MB RAM, and

Table 2. Test results for the three learning schemes.

Method	Overrides	Turns
Pure stereo	N/A	45.6 ($\sigma = 3.0$)
1. Cold turkey	25.1 ($\sigma = 8.2$)	42.8 ($\sigma = 3.7$)
2. DAgger	10.7 ($\sigma = 5.3$)	41.4 ($\sigma = 3.2$)
3. Training wheels	4.3 ($\sigma = 2.6$)	40.4 ($\sigma = 2.6$)

The average and standard deviation are given for the number of overrides and turns during the testing period. A lower number of overrides is better. In the table, the best results are shown in boldface.

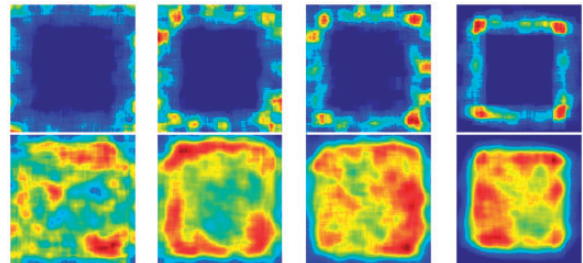


Figure 11. Simulation heatmaps, from left to right: cold turkey, DAgger, training wheels, stereo only. Top images are turn locations, lower images are the approaches.

normally runs the Parrot firmware as an autopilot. For the experiments, we replace this firmware with the open source Paparazzi autopilot software.^{39,40} This allowed us to implement all vision and learning algorithms on board the drone. The length of each test is dependent on the battery, which due to wear has considerable variation, in the range of 8–15 min.

The tests are performed in an artificial room that has been constructed within a motion-tracking arena. This allows us to track the trajectory of the drone and facilitates post-experiment analysis. The room is approximately 5×5 m, as delimited by plywood walls. In order to ensure that the stereo vision algorithm gave reliable results, we added texture in the form of duct-tape to the walls. In five tests, we had a textured carpet hanging over one of the walls (Figure 13 left, referred to as



Figure 12. The used multicopter.



Figure 13. Two test flight rooms.

“room 1”), in the other five tests it was on the floor (Figure 13 right, referred to as “room 2”).

Results

Table 3 shows the summarized results obtained from the monocular test flights. Two main observations can be made from this table. First, the average number of stereo overrides during the test phase is 3, which is very close to the number of overrides in simulation. The monocular behavior also has a similar heat map to simulation. Figure 14 shows a heat map of the drone’s position during the approaches and the avoidance maneuvers (the turns). Again, the stereo-based flight performs better in the sense that the drone explores the room much more thoroughly and the turns happen consistently just before an obstacle is detected. On the other hand, especially in room 2, the monocular performance is quite good in the sense that the system is able to explore most of the room.

Second, the selected TPR and FPR are on average 0.47 and 0.11. The TPR is rather low compared to the offline tests. However, this number is heavily influenced by the monocular estimator-based behavior. Due to the goal of the robot, avoiding obstacles slightly before the stereo ground truth recognizes them as positives, positives should hardly occur at all. Only in cases of FNs where the estimator is slower or wrong, positives will be registered

Table 3. Test flight summary.

Description	Room 1					Room 2					Avg.
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	
Stereo flight time m:ss	6:48	7:53	2:13	3:30	4:45	4:39	4:56	5:12	4:58	5:01	4:59
Mono flight time m:ss	3:44	8:17	6:45	7:25	4:54	10:07	4:46	9:51	5:23	5:12	6:39
Mean square error	0.7	1.96	1.12	0.95	0.83	0.95	0.87	1.32	1.16	1.06	1.09
False positive rate	0.16	0.18	0.13	0.11	0.11	0.08	0.13	0.08	0.1	0.08	0.11
True positive rate	0.9	0.44	0.57	0.38	0.38	0.4	0.35	0.35	0.6	0.39	0.47
Stereo approaches	29	31	8	14	19	22	22	19	20	21	20.5
Mono approaches	10	21	20	25	14	33	15	28	18	15	19.9
Auto-overrides	0	6	2	2	1	5	2	7	3	2	3
Overrides ratio	0	0.72	0.3	0.27	0.2	0.49	0.42	0.71	0.56	0.38	0.41

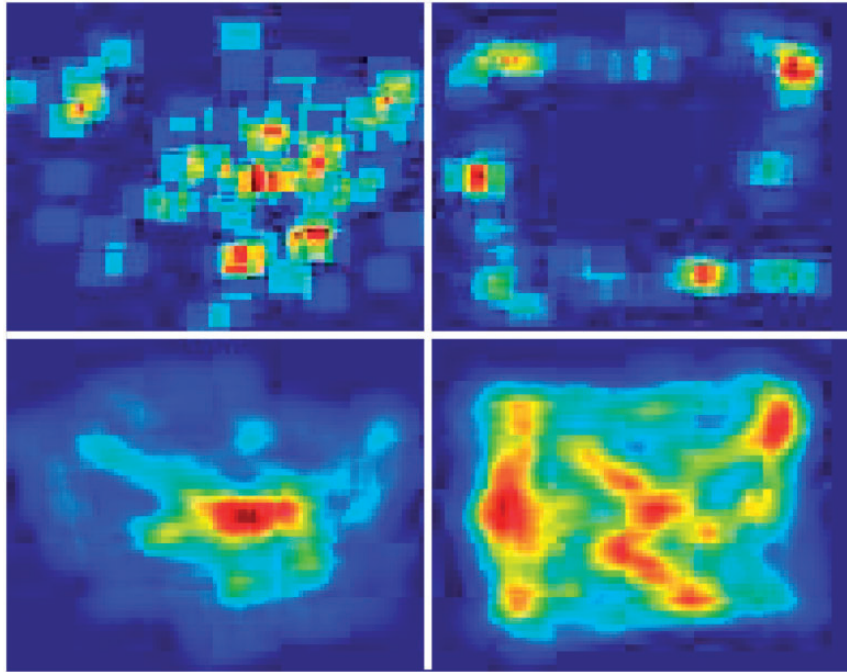


Figure 14. Room 1 (plain texture) position heat map. Top row is the binned position during the avoidance turns, bottom row during the obstacle approaches, right column during stereo ground truth based operation, left column during learned monocular operation.

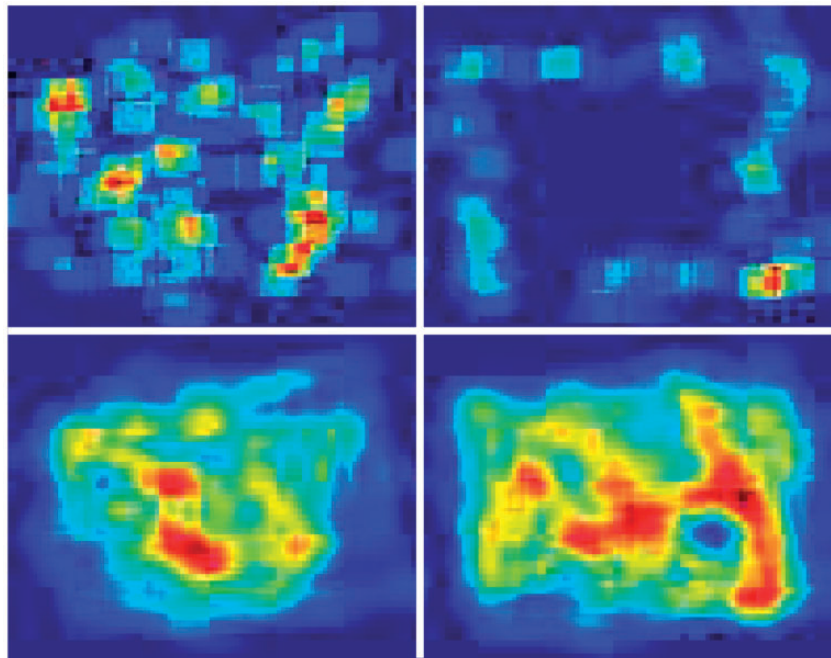


Figure 15. Room 2 (carpet natural texture) position heat map. Top row is the binned position during the avoidance turns, bottom row during the obstacle approaches, right column during stereo ground truth based operation, left column during learned monocular operation.

by the ground truth. Similarly, the FPR is also lower in the context of the monocular-based behavior.

ROC curves of the 10 flights are shown in Figure 15. A comparison based on the numbers between the first

five flights (room 1) and the last five flights (room 2) does not show any significant differences, leading to the suggestion that the system is able to learn both rooms equally well. However, when comparing the heat maps of the

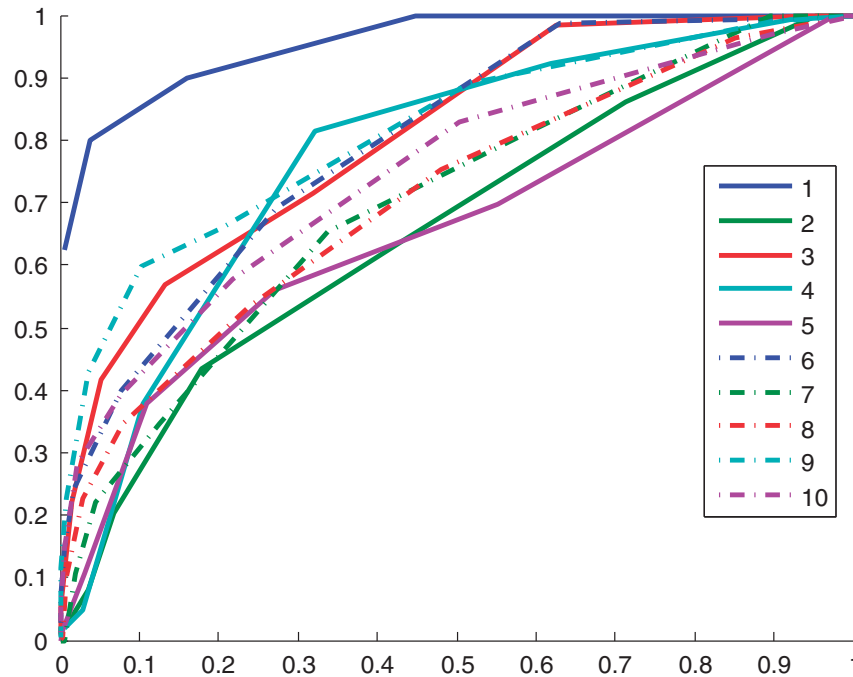


Figure 16. ROC curves of the 10 test flights. Dashed/solid lines refer to results on room #1/#2.

two situations in the monocular system in Figures 14 and 16, it seems that the system shows slightly different behavior. The monocular system appears to explore room 2 better, getting closer to copying the behavior of the stereo-based system. This is also pointed out by the peak in the heat map in the bottom row, left column of Figure 14; the binned position occurrence of the monocular behavior during the straights. It shows the behavior was affected by false positives, it sometimes turned too soon and too far from the walls, resulting in a peak in the middle of the room. Similarly, this is also visible when comparing the monocular turn locations (Figure 14, top row) to the stereo turn locations (Figure 14 top right). The stereo algorithm turns consistently close to the walls, while the monocular behavior shows a lot of spread and turns often quite far away from the walls. Interestingly, this problem partly disappears when more varied and natural texture is applied to the same room as shown by the results in Figure 16.

The experimental setup with the room in the motion tracking arena allows for a more in-depth analysis of the performance of both stereo and monocular vision. Figure 17 shows the spatial view of the flight trajectory of test #10 (note 2). The flight is segmented into approaches and turns which are numbered accordingly in these figures. The color scale in Figure 17(a) is created by calculating the theoretically visible closest wall based on the tracking the systems measured heading and position of the drone, the known position of the walls, and the FOV angle of the camera. It is clearly

visible that the stereo ground truth in Figure 17(b) does not capture this theoretical disparity perfectly. Especially in the middle of the room, the disparity remains high compared to the theoretical ground truth due to noise in the stereo disparity map. The results of the monocular estimator in Figure 17(c) show another decrease in quality compared to the stereo ground truth.

Discussion

We start the discussion with an interpretation of the results from the simulation and real-world experiments, after which we proceed by discussing persistent SSL in general and provide a comparison to other machine learning techniques.

Interpretation of the results

Using persistent SSL, we were able to autonomously navigate our multicopter on the basis of a stereo vision camera, while training a monocular estimator on board and online. Although the monocular estimator allows the drone to continue flying and avoiding obstacles, the performance during the approximately 10-min flights is not perfect. During monocular flight, a fairly limited amount of (autonomous) stereo overrides was needed while at the same time the robot was not fully exploring the room like when using stereo.

Several improvements can be suggested. First, we can simply have the drone learn for a longer time,

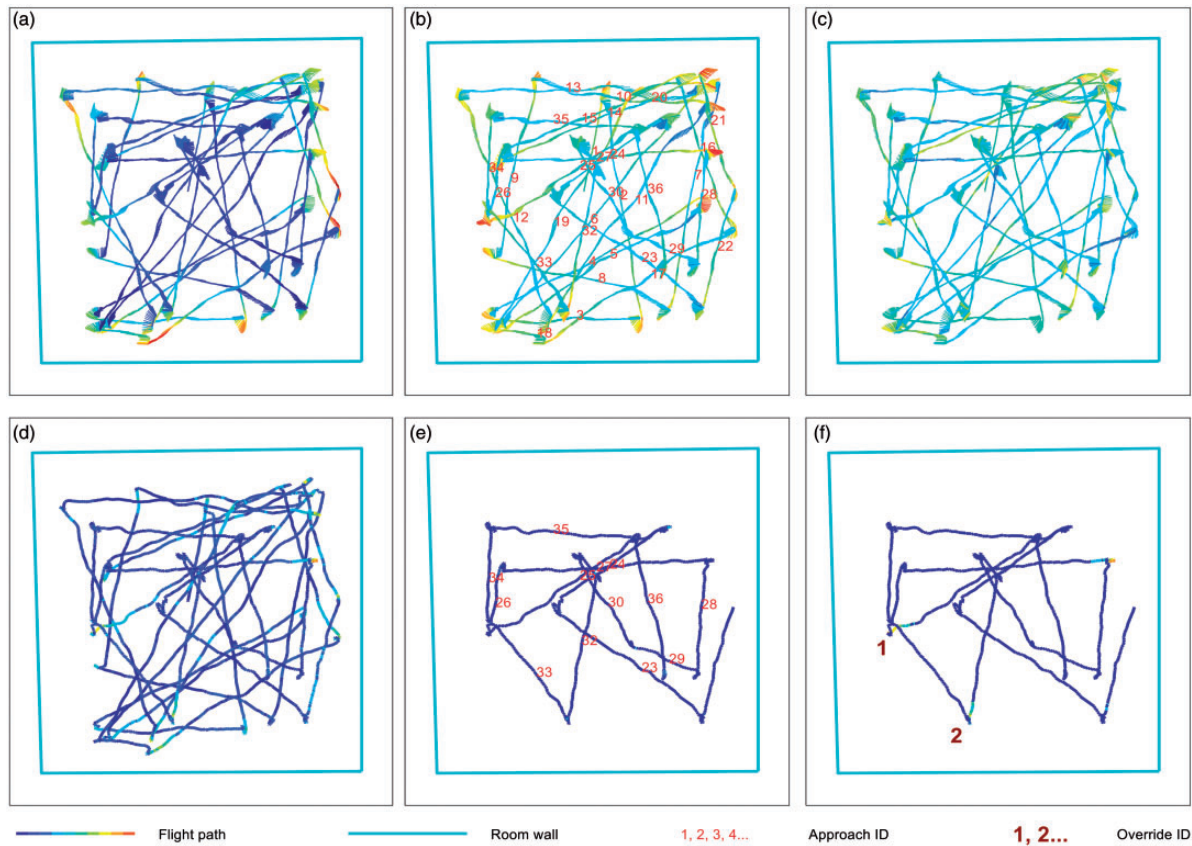


Figure 17. Flight path of test 10 in room 2. Monocular flight starts form approach 23. The meaning of the color of the flightpath differs per image; (a): the approximated disparity based on the external tracking system. (b): the measured stereo average disparity, (c): the monocular estimated disparity, (d): the error between the stereo disparity and monocular estimated disparity with dark blue meaning zero error, (e): error during FP, (f): error during FN. (e and f) only show the monocular part of the flight.

accumulating training data over multiple flights. In an extended offline test, our VBoW method shows saturation at around 6000 samples. Using additional features and more advanced learning methods may result in improved performance if training set sizes increase.

During our tests in different environments, it proved unnecessary to tune the VBoW learning algorithm parameters to a new environment as similar performance was obtained. The learned results on the robot itself may or may not generalize to different environments; however, this is of less concern as the robot can detect a new environment and then decide to continue the learning process if the original cue is still available. In order to detect an inadequacy of the learned regression function, the robot can occasionally check the estimation error against the stereo ground truth. In fact our system already does so autonomously using its safety override. Methods on checking the performance without using the ground truth, e.g. by employing a learner that gives an estimate of uncertainty, are left for future work.

Deep learning

At the time of our robotic experiments, implementing state-of-the-art deep learning methods on-board a flying drone was deemed infeasible due to hardware restrictions. One of the major advantages of persistent SSL is the unprecedented amount of available training data. This amount of data will be more useful to more complex learning methods such as deep learning methods than to less complex, but computationally efficient methods such as the VBoW method used in our experiments. Today, with the availability of strongly improved hardware such as the NVidia Jetson TX1, close-to state-of-the-art models can be trained and run on-board a drone, which may significantly improve the learning results.

Persistent SSL in relation to other machine learning techniques

In order to place persistent SSL in the general framework of machine learning, we compare it with several

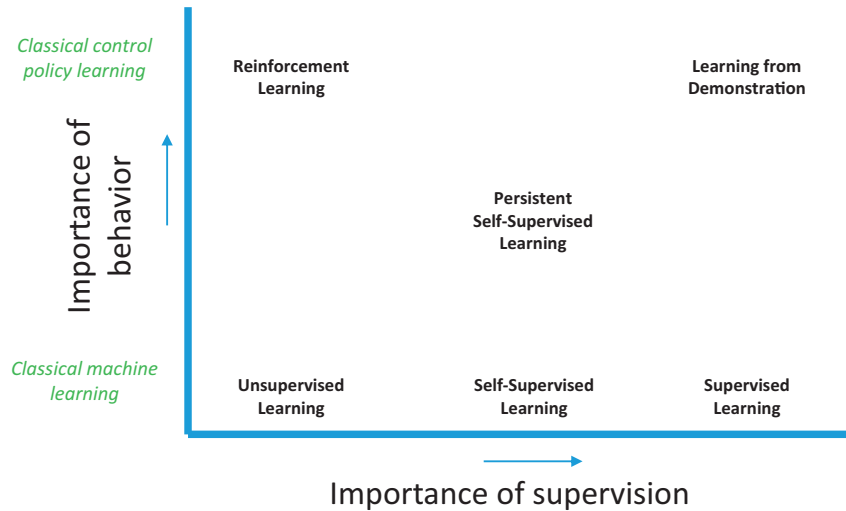


Figure 18. Lay of the machine learning land.

techniques. An overview of this comparison is presented in Figure 18.

Un-/semi-/supervised learning. Unsupervised learning does not require labeled data, semi-supervised learning requires only an initial set of labeled data,⁴¹ and supervised learning requires all the data to be labeled. Internally, persistent SSL uses a standard supervised learning scheme, which greatly facilitates and speeds up learning. The typical downside of supervised learning—acquiring the labels—does not apply to SSL, since the robot continuously provides these labels itself.

A major difference between the typical use of supervised learning and its use in persistent SSL, is that the data for learning are generally assumed to be i.i.d. However, persistent SSL controls a behavioral component which, in turn, affects both the data set obtained during training as well as during testing time. Operation based on ground truth induces a certain behavior that differs significantly from behavior induced from a trained estimator, even more so for an undertrained estimator.

SSL. The persistent form of SSL is set apart in the figure from “normal” SSL, because the persistence property introduces a much more significant behavioral component to the learning. While normal SSL expects the trusted cue to remain available, persistent SSL assumes that the robot may sometimes act in the absence of the trusted cue. This introduces the feedback-induced data bias problem, which, as we have seen, requires specific behavior strategies for best learning the robot’s task.

Learning from demonstration. Imitation learning, or LfD, is a close relative to persistent SSL. Consider for instance teleoperation, an LfD scheme in which a (human or robot) teacher remotely operates a robot

in order for it to learn demonstrated actions in its environment.²⁰ This can be compared to persistent SSL if we consider the teacher to be the ground truth function $g(x_g)$ in the persistent SSL scheme. In most cases described in literature, the teacher shows actions from a control policy taken on the basis of a state instead of just the results from a sensory cue (i.e., the state). However, LfD does contain exceptions in which the learner only records the states during demonstration, e.g. when drawing a map through a 2D representation of the world in case of a path planning mission in an outdoor robot.⁴² Like persistent SSL, test time decisions taken in LfD schemes influence future observations which may or may not be contained in known demonstrated territory. However, one key difference between LfD and persistent SSL arguably sets them apart. All LfD theory known to the authors implicitly assumes the teacher is never the same entity as the learner. It may be that all relevant sensors are on the learner, and even that the learners body is used to execute teacher commands (like in teleoperation), but the teachers intelligence is always an external entity.

Reinforcement learning. Lastly, we compare persistent SSL with RL, which is a distinctively different technique.⁴³ In RL, a policy is learned using a reward function. Due to the evaluative feedback provided in RL, defining a good reward function is one of fundamental difficulties of RL known as reward shaping.^{43,44} Since persistent SSL uses supervised feedback, reward shaping is less of an issue in persistent SSL, only requiring a choice of a loss function between $g(x_g)$ and $f(x_f)$. Secondly, the initial exploration phase of RL often infers a lot of trial-and-error, making it a dangerous time in which a physical system may crash and be damaged. Although this particular problem is often solved

by better initialization, e.g. by using for instance LfD or using policy search instead of value function-based approaches, persistent SSL does not require an untrained initialization phase at all as a reliable ground truth function guarantees a certain minimal correct behavior.

Persistent SSL differs from other learning techniques in the sense that no complete training data set is needed to train the algorithm beforehand. Instead it requires a ground truth $g(x_g)$, which must be available online in real time while training $\hat{f}(x_f)$, but can be switched off when $\hat{f}(x_f)$ is learned to satisfaction. This implies that learning needs to be persistent and that the switch θ must be included in the model. Note that in cases where the environment of the robot may change, measures can be put in place to detect the output uncertainty of $\hat{f}(x_f)$. If the uncertainty goes up, the robot can switch back to using the ground truth function and learning can then be activated again. Developing such measures is, however, left for future work.

Feedback-induced data bias

The robot induces how its environment is perceived, meaning it influences the acquired training samples based on its behavior. The problems arising from this feedback-induced data bias are known from other machine-learning disciplines, such as RL and LfD.⁴³ In particular, Ross et al. have proposed DAgger² to solve a similar problem in the LfD domain, which iteratively aggregates the data set with induced training samples and the experts reaction to it. However, in the case of LfD, obtaining the induced training samples requires a careful engineered and often additional setup, while in persistent SSL, this functionality is inherently available. Secondly, the performance of the LfD expert (i.e., in many cases, a human) is not easy to control, often reacting too late or too early. The control policy of the persistent SSL ground truth override system can, on the other hand, be very deterministic. In the case of a DAgger application with drones flying through a forest,³ it proved infeasible to reliably sample the expert in an online fashion. Acquired videos had to be processed offline by the expert, hence the need for (offline \leftrightarrow online) iterations. Moreover an additional online human safety override interface was still necessary to prevent damage to the drone while learning. Thirdly, due to the cost of (and need for) iterative demonstration sessions, the emphasis of DAgger is on converging fast with needing as little expert sessions as possible. In persistent SSL, there are no costs for using the teacher signals coming from the original sensor cue. With persistent SSL, we can directly focus on effectively using the

available amount of training samples instead of minimizing the number of iterations like in DAgger.

Another reason why persistent SSL handles the induced training sample issue better than other state of the art robot learning methods, is that in persistent SSL part of the learning problem itself can be easily separated and tested from the behavior; i.e. in a traditional supervised learning setting. In our proof of concept, this has allowed us to test the learning algorithms and thoroughly investigate its limits before deployment.

Conclusion

We have investigated the behavioral aspects of an SSL scheme, in which the supervisory signal is switched off after an initial learning period. In particular, we have studied an instance of such “persistent SSL” for the task of obstacle avoidance, in which the robot uses trusted stereo vision distance estimates in order to learn appearance-based monocular distance estimation. We have shown that this particular setup is very similar to LfD. This similarity has been corroborated by experiments in simulation, which showed that the worst learning strategy is to make a hard switch from stereo vision flight to mono vision flight. It is best to have the robot fly based on mono vision and using stereo vision only as “training wheels,” to take over when the robot would otherwise collide with an obstacle. The real-world robot experiments show the feasibility of the approach, giving acceptable results already with just 4–5 min of learning.

The findings also indicate interesting future venues of investigation. First, and perhaps most importantly, in the 4–5 min of the real-world experiments, the robot already experiences roughly 7000–9000 supervised learning samples. It is clear that longer learning times can lead to very large supervised data sets, which are suitable for deep learning approaches. Such approaches likely allow the learning to extend to much larger and more varied environments, such as outdoor forests or multiple rooms inside larger buildings. In addition, they could allow the learning to improve the resolution of disparity estimates from a single value to a full image size disparity map. Second, in the current experiments, the robot stayed in a single environment. We mentioned that a different environment can make the learned mapping invalid, and that this can be detected by means of the ground truth. Another venue, as studied in Ho et al.,¹² is to use a machine learning method with an associated uncertainty value. For instance, one could obtain uncertainty estimates by using a learning method such as a Gaussian Process or by using dropout with deep neural networks. This can help with a further integration of the behavior with learning, for

instance by tuning the forward velocity based on the certainty. These venues together could allow for persistent SSL to reach its full potential, significantly enhancing the robustness of robots operating in real-world environments.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

Notes

- a. A video of VBoW visualizations on data set #1 and #2 can be viewed online: https://www.youtube.com/playlist?list=PL_KSX9GOn2P9v0rtjSGonDC0V0T3DXYf6
- b. Onboard, external a visualization video of flight #10 can be viewed at: https://www.youtube.com/playlist?list=PL_KSX9GOn2P9v0rtjSGonDC0V0T3DXYf6

References

1. García J and Fernández F. A comprehensive survey on safe reinforcement learning. *J Mach Learn Res* 2015; 16: 1437–1480.
2. Ross S, Gordon GJ and Bagnell JA. A reduction of imitation learning and structured prediction. In: *14th International conference on artificial intelligence and statistics*, Ft. Lauderdale, FL, USA, p. 15, 2011.
3. Ross S, Melik-Barkhudarov N, Shankar KS, et al. Learning monocular reactive UAV control in cluttered natural environments. In: *2013 IEEE international conference on robotics and automation*, Karlsruhe, Germany, pp. 1765–1772, 2013.
4. Thrun S, Montemerlo M, Dahlkamp H, et al. Stanley: the robot that won the darpa grand challenge. In: *The 2005 DARPA grand challenge*, pp. 1–43. Springer, 2007.
5. Lieb D, Lookingbill A and Thrun S. Adaptive road following using self-supervised learning and reverse optical flow. In: Sebastian T, Gaurav S, Sukhatme and Stefan S (eds.) *Robotics: science and systems*, 2005, MIT Press, Cambridge, Massachusetts, pp. 273–280.
6. Lookingbill A, Rogers J, Lieb D, et al. Reverse optical flow for self-supervised adaptive autonomous robot navigation. *Int J Comp Vision* 2007; 74: 287–302.
7. Procopio MJ, Mulligan J and Grudic G. Learning terrain segmentation with classifier ensembles for autonomous robot navigation in unstructured environments. *J Field Robot* 2009; 26: 145–175.
8. Bajracharya M, Howard A, Matthies LH, et al. Autonomous off-road navigation with end-to-end learning for the lagr program. *J Field Robot* 2009; 26: 3–25.
9. Hadsell R, Sermanet P, Ben J, et al. Learning long-range vision for autonomous off-road driving. *J Field Robot* 2009; 26: 120–144.
10. Muller UA, Jackel LD, LeCun Y, et al. Real-time adaptive off-road vehicle navigation and terrain classification. In: *SPIE defense, security, and sensing*, pp. 87410A–87410A. San Diego, California United States: International Society for Optics and Photonics, 2013.
11. Baleia J, Santana P and Barata J. On exploiting haptic cues for self-supervised learning of depth-based robot navigation affordances. *J Intellig Robot Syst* 2015; 80 (3-4): 455–474.
12. Ho HW, De Wagter C, Remes BDW, et al. Optical flow for self-supervised learning of obstacle appearance. In: *Intelligent robots and systems (IROS), 2015 IEEE/RSJ international conference*, pp. 3098–3104. IEEE, September 28 - October 2, 2015, Congress Centre Hamburg, Hamburg, Germany.
13. Lamers K, Tijmons S, De Wagter C, et al. Self-supervised monocular distance learning on a lightweight micro air vehicle. In: *Intelligent robots and systems (IROS), 2016 IEEE/RSJ international conference*, Daejeon, Korea, October 9-14, 2016, pp. 1779–1784. IEEE, 2016.
14. Gandhi D, Pinto L and Gupta A. Learning to fly by crashing. *arXiv preprint arXiv:1704.05588*, 2017.
15. Mori T and Scherer S. First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles. In: *Proceedings—IEEE international conference on robotics and automation*, Karlsruhe, Germany, pp. 1750–1757, 2013.
16. Engel J, Sturm J and Cremers D. Scale-aware navigation of a low-cost quadcopter with a monocular camera. *Robot Auton Syst* 2014; 62: 1646–1656.
17. van Breugel F, Morgansen K and Dickinson MH. Monocular distance estimation from optic flow during active landing maneuvers. *Bioinspirat Biomimet* 2014; 9: 025002.
18. de Croon GCHE. Monocular distance estimation with optical flow maneuvers and efference copies: a stability-based strategy. *Bioinspirat Biomimet* 2016; 11: 016004.
19. de Croon GCHE, Groen MH, De Wagter C, et al. Design, aerodynamics and autonomy of the DelFly. *Bioinspirat Biomimet* 2012; 7: 025003.
20. Argall BD, Chernova S, Veloso M, et al. A survey of robot learning from demonstration. *Robot Auton Syst* 2009; 57: 469–483.
21. Guo X, Lee H, Wang X, et al. Deep learning for real-time Atari game play using offline Monte Carlo tree search planning. In: *Proceedings of conference on neural information processing systems (NIPS)*, vol. 2600, Montréal, Canada, pp. 1–9, 2014.
22. Sadeghi F and Levine S. (cad)2rl: real single-image flight without a single real image. *arXiv preprint arXiv:1611.04201*, 2016.
23. Hoiem D, Efros AA and Hebert M. Automatic photo pop-up. *ACM Trans Graph* 2005; 24: 577.
24. Saxena A, Chung SH and Ng AY. 3-D depth reconstruction from a single still image. *Int J Comput Vis* 2007; 76: 53–69.

25. Saxena A, Sun M and Ng AY. Make3D: learning 3D scene structure from a single still image. *IEEE Trans Pattern Anal Mach Intellig* 2009; 31: 824–840.
26. Lenz I, Gemici M and Saxena A. Low-power parallel algorithms for single image based obstacle avoidance in aerial robots. In: *IEEE international conference on intelligent robots and systems*, Vilamoura, Algarve, Portugal, pp. 772–779, 2012.
27. Eigen D, Puhrsch C and Fergus R. Depth map prediction from a single image using a multi-scale deep network. In: *Advances in neural information processing systems*, (NIPS 2014), Montréal, Canada, pp. 1–9, 2014.
28. Michels J, Saxena A and Ng AY. High speed obstacle avoidance using monocular vision and reinforcement learning. In: *Proceedings of the 22nd international conference on machine learning*, vol. 3, Bonn, Germany, August 07 - 11, 2005, pp. 593–600. ACM Press, 2005.
29. Dey D, Shankar KS, Zeng S, et al. Vision and learning for deliberative monocular cluttered flight, pp.391-409. Springer, 2016.
30. Yamauchi K, Oota M and Ishii N. A self-supervised learning system for pattern recognition by sensory integration. *Neural Netw* 1999; 12: 1347–1358.
31. Thrun S, Montemerlo M, Dahlkamp H, et al. Stanley: the robot that won the DARPA grand challenge. *Springer Tracts Adv Robot* 2007; 36: 1–43.
32. De Wagter C, Tijmons S, Remes BDW, et al. Autonomous flight of a 20-gram flapping wing MAV with a 4-gram onboard stereo vision system. In: *IEEE international conference on robotics & automation (ICRA)*, number Section IV, 31 May - 07 Jun 2014, Hong Kong, China, pp. 4982–4987, 2014.
33. De Croon GCHE, De Weerd E, De Wagter C, et al. The appearance variation cue for obstacle avoidance. *IEEE Trans Robot* 2012; 28: 529–534.
34. Varma M and Zisserman A. Texture classification: are filter banks necessary? 1 Introduction 2 A review of the VZ classifier. In: *Computer vision and pattern recognition 2003 proceedings 2003 IEEE computer society conference*, 2003, Madison, Wisconsin June 16-22, 2003.
35. Wu B, Ooi TL and He ZJ. Perceiving distance accurately by a directional process of integrating ground information. *Nature* 2004; 428: 73–77.
36. van Hecke K. Monocular obstacle avoidance with persistent self-supervised learning dataset. data.4TU.nl, 2017, Available at <https://doi.org/10.4121/uuid:a3599d11-d56a-4402-93f8-2e7c22cf5dab>
37. De Wagter C and Amelink MHJ. Development of inertial navigation, onboard vision and adaptive control for autonomous long-distance mav operations. In: *Conference: European micro air vehicle conference and competition*, Toulouse, France, 2007.
38. Cohen PR. Empirical methods for artificial intelligence. *IEEE Intellig Syst* 1996; (6): 88.
39. B.D.W. Remes, Dino Hensen, Freek Van Tienen, Christophe De Wagter, Erik Van der Horst, and G.C. H.E. De Croon. Paparazzi: how to make a swarm of parrot AR drones fly autonomously based on GPS. In: *IMAV 2013: proceedings of the international micro air vehicle conference and flight competition*, Toulouse, France, 17–20 September 2013.
40. Brisset P, Drouin A, Gorraz M, et al. The Paparazzi solution, MAV 2006, 2nd US-European Competition and Workshop on Micro Air Vehicles, Oct 2006, Sandestin, United States.
41. Zhu X. Semi-supervised learning literature survey. *Sciences-New York*, 2007, pp. 1–59 (1530). Computer Sciences, University of Wisconsin-Madison.
42. Ratliff N, Bradley D, Bagnell JA, et al. Boosting structured prediction for imitation learning for imitation learning. In: *Advances in neural information processing systems (NIPS)*, 2006, Vancouver, B.C., Canada, p. 54.
43. Kober J, Bagnell Ja and Peters J. Reinforcement learning in robotics: a survey. *Int J Robot Res* 2013; 32: 1238–1274.
44. Littman ML. Reinforcement learning improves behaviour from evaluative feedback. *Nature* 2015; 521: 445–451.

Confined spaces industrial inspection with micro aerial vehicles and laser range finder localization

International Journal of Micro Air Vehicles
2018, Vol. 10(2) 207–224
© The Author(s) 2018
DOI: 10.1177/1756829318757471
journals.sagepub.com/home/mav


Paolo Tripicchio , **Massimo Satler** , **Matteo Unetti** and **Carlo A Avizzano** 

Abstract

This work addresses the problem of semi-automatic inspection and navigation in confined environments. A system that overcomes many challenges at the state of the art is presented. It comprises a multicopter able to inspect an industrial combustion chamber thus working in a GPS-denied environment with poor lighting conditions, in the presence of magnetic and communication disturbances, iron dust and repetitive patterns on the structure walls. The presented system is able to pass through narrow entrances but still capable of acquiring high resolution images and to allow operators to perform inspection of the structures. Starting from the captured data, the system is able to provide a 3D reconstruction of the inspected environment for offline analysis.

Keywords

Industrial inspection, simultaneous localization and mapping, collision avoidance, navigation, GPS-denied, inspection, confined space

Received 19 May 2017; accepted 10 January 2018

Introduction

Maintenance is an important aspect that requires periodic control of equipment, systems, machineries and infrastructures. It is defined by the European standard (prEN 13306, 1998) as “Combination of all technical, administrative and managerial actions during the life cycle of an item intended to retain it in, or restore it to, a state in which it can perform a required function”. Depending on the objective of the maintenance activities, such a process can be distinguished into five main categories: preventive maintenance; predictive maintenance; corrective maintenance; zero-hours maintenance and periodic maintenance. Common goal of all these procedures is to avoid unexpected failure/inefficiency. Indeed, an effective maintenance program is fundamental to improve equipment life and possibly avoid unplanned maintenance activities. Unfortunately, any maintenance program is in general time consuming and costly. In the industrial field for example, maintenance process could require plant downtime and service interruptions.

Periodic visual inspection is in general the first step performed in any industrial maintenance program to

detect typical defective in the materials status such as, among others, corrosion in iron and steel components, cracks in building walls and chimneys, etc. These inspections are typically done by experienced surveyors employing eyesight or special contactless measurement devices like for instance thermal/multispectral camera if required.

Although simple in principle, such operations are complex and time consuming since surveyors need some facilities (e.g. scaffolding) that allow him/her to be at a close distance from the inspected structure. Furthermore, such inspections are carried out numerous times in hazardous environments or in confined spaces, where the access is usually difficult and the

Gustavo Stefanini Advanced Robotics Research Center, TeCIP Institute, Scuola Superiore Sant’Anna, Pisa

Corresponding author:

Paolo Tripicchio, Gustavo Stefanini Advanced Robotics Research Center, TeCIP Institute, Scuola Superiore Sant’Anna, Pisa.
Email: p.tripicchio@santannapisa.it



Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (<http://www.creativecommons.org/licenses/by/4.0/>) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (<https://us.sagepub.com/en-us/nam/open-access-at-sage>).

working condition turns out to be extreme for a human being.

Taking into account the former considerations, it is evident that the introduction of any automation into the inspection process can improve a lot the safety of the operator and hopefully it can speed up the failure reaction chain. According to the International Federation of Robotics (IFR) "A service robot is a robot which operates semi-or fully autonomously to perform services useful to the well-being of humans and equipment, excluding manufacturing operations". Service robots in general, assist human being performing repetitive jobs in dirty, distant or dangerous environment.

Typical examples here are climbing robots used to inspect walls¹ or metallic structure.² However, in the last decade, the interest of the European Community on the Aerial service robots research field has grown a lot.³⁻⁵ The aim here is to use Aerial Vehicles in real applicative scenario to support or replace human operators in all those activities that are either repetitive or dangerous for the human beings. More recently, the research community moves towards applications where the aerial vehicles are used as systems to interact with the environment and collaborate with other units to accomplish robotic tasks such as assembling or manipulating an object.^{6,7}

Satler et al.⁸ presented a system for remote visual inspection of indoor environment employing a micro aerial vehicle (MAV) endowed with embedded computing power. Initial tests were performed on an indoor office environment. The device has been proposed as operator replacement for a first and safe inspection process in order to detect surface damages and/or decide if additional intervention is required.

In a later work,⁹ we extended the indoor inspection task with the ability to explore multiple floors buildings introducing a floor recognition and level merging algorithm.

In this work, we extend the previous systems introducing solutions to allow a complete industrial inspection task in a confined yet dangerous space. The scenario taken as reference is a non-structured industrial boiler, which is basically composed by a big combustion chamber covered by pipes that has to be verified by the maintenance process. The system here presented is able to locate itself and navigate efficiently in the environment in order to complete an industrial maintenance task in such kind of scenarios. A collision avoidance procedure allows the system to correct desired trajectory while avoiding collision with the external structures or objects moving in the environment.

The rest of the paper is organized as follows. The next section presents related works at the state of the art in the fields of industrial inspection and robot

localization. Then the following section introduces a reference scenario that has been used to define project requirements and to demonstrate the capabilities of the developed system. In the subsequent section, the hardware and software components of the overall system are presented. This is followed by a section which discusses about developed control system algorithms for the localization and mapping of the robot in the environment. Then, the user teleoperation interface and the automatic MAV navigation module are discussed. Then, we present some post processing technique to obtain inspection information that can be analyzed off-line. The preliminary evaluation flight test performed inside an industrial combustion chamber is presented. In the last section, the main conclusions are drawn.

Related work

MAVs have become increasingly popular for autonomous navigation in unstructured environment thanks to their agility and fast dynamics compared with ground robots. There are several works focused on inspection tasks developed in the recent years; early works appeared in the literature were based on visual inspection without any contact with the environment, while recently works founded by EU projects proposed contact-based approaches.^{6,10}

In Luque-Vega et al.,¹¹ a quadrotor is used for high voltage power line inspection. The system payload is composed of a thermal infrared and a color camera for the inspection purpose and a GPS, IMU and altimeter for navigation capability. The vision algorithms, however, run on a remote ground control station in order to detect real-time anomaly/defect alarms.

In Bonnin-Pascual et al.,¹² an MAV is used to visually inspect vessels in order to detect cracks and corrosion in the metallic structures. The solution is based on supervisory autonomy, i.e. the surveyor controls the inspection process teleoperating the vehicle which in turn is provided with on-board algorithms to ease the navigation and control from unexperienced people.

In Gohl et al.,¹³ a first attempt driving an MAV in underground mine field is presented. The system is based on a hexacopter endowed with Skybotix visual-inertial sensor and 2D laser scanner used to collect data during a manual flight. A 3D environment reconstruction is then performed off-line evaluating the quality of the acquired data as well as of the localization process.

In Nikolic et al.,¹⁴ an MAV endowed with an integrated visual-inertial SLAM sensor is employed for industrial boiler inspection. This work uses a front looking stereo camera and an IMU to estimate the vehicle pose and navigate inside the industrial boiler following a reference trajectory.

In addition to the aforementioned applications, such devices have proven their flexibility in many other fields including but not limited to agriculture,¹⁵ search and rescue,¹⁶ exploration and mapping,⁹ dam inspection, early fire detection and forest protection, traffic monitoring, aerial photography, surveillance and reconnaissance, chemical spraying and entertainment industry and filming. A detailed analysis for civil application has been provided in Sarris and Atlas.¹⁷

Each system mainly differs for the platform autonomy level, i.e. how much computation is empowered to the ground control station supporting the vehicle, the sensor payload and the assumptions about the environment knowledge, i.e. structured/unstructured or known/unknown. Typical navigation sensor suites are based on GPS, laser scanner, infrared or ultrasound sensors and more recently on camera. This latter solution represents the richest data supplier combined with low weight and low prices (compared to lasers) at the cost of increased computational cost required to run vision algorithms.

Simultaneous localization and mapping (SLAM)¹⁸ algorithm is a fundamental component constantly present on each platform used in real application domain. Graph-based methods¹⁹ or probabilistic methods²⁰ have been proposed by the research community in the last three decades. Recent works on SLAM on the other hand, referred as visual-SLAM or vision-based navigation in the literature, are based alternatively on features tracking using either a mono or stereo frontal camera,^{21,22} or a ground-looking camera²³ and on direct methods.²⁴

Although promising and accurate, the outcome and the robustness of all vision-based methods strongly depend on two assumptions: (i) enough lighting conditions and (ii) environment texture richness. While the former assumption can be met using custom designed lighting systems, the latter is likely to fail in real scenarios that are generally characterized by repetitive elements or that are poor in texture features.

Power plant boiler inspection

The proposed solution has been designed considering as reference task the inspection of the interior part of a thermal power plant boiler.

The boiler is classified as confined space and it basically consists in a big combustion chamber completely covered by pipes transporting water for the steam production. Later on, the “wet steam” passes through the superheater, the reheater and the economizer to improve the energy exchange efficiency.

Maintenance task is typically divided into daily, weekly, monthly, semi-annual and annual tasks. Portholes along the whole structure are used during

frequent and elementary inspections, e.g. daily, weekly, to access corresponding points of interest. Semi-annual and annual maintenance routine on the other hand, is performed entering into the boiler from the bottom part.

In the latter case, it is required to stop the steam production process in advance, wait the required time to decrease the temperature and then wash the combustion chamber. When the environmental condition is feasible for a human being, the bottom part of the boiler (64×64 cm entrance) is used to access the interior and hence start setting up scaffolding structure to get close to the structure to inspect. Obviously, this is a long process and demanding for the worker first and the surveyor later.

The employment of an MAV allows to localize losses which in turn will speed up the intervention time. Such devices provide easy and fast ways to detect surface damages allowing to study the most convenient intervention strategy, to prepare all the materials and to schedule the maintenance intervention in advance. Finally, yet importantly, the employment of such a technology will improve the surveyor working conditions.

The target of the inspection is the combustion chamber, which is composed by a parallelepiped of $7.6 \times 11 \times 21$ m, see Figure 1. The other parts of the boiler have too many obstacle and few free-spaces for an MAV safe flight.

Requirements

The system requirements have been drawn taking into account both the operators experience and preliminary real field tests.

Considering the goal of the proposed system, i.e. provide a mean for tele-visual inspection, the preferred vehicle flight characteristics are: the vertical take-off and landing (VTOL) capability; the possibility to perform stationary flights and low-speed movements without compromising the flight stability; allow indoor flight as well as provide robustness to accidental contact with the structure being considered within the inspection process. These requirements rapidly discard fixed-wing platforms since they do not allow VTOL capability and require a minimum velocity to guarantee vehicle lift. The indoor flight requirements discard vehicles powered by combustion engines which, moreover, are not suitable to flight in most of the confined spaces characterized by fire and explosion risks. All these aspects focus the search on electrically powered MAVs in the form factor of quadrotor or hexarotor that can be designed with ducted fan in order to guarantee robustness for unwanted environmental collision. Moreover, the ducted fan vehicles reduce hovering power consumption.²⁵

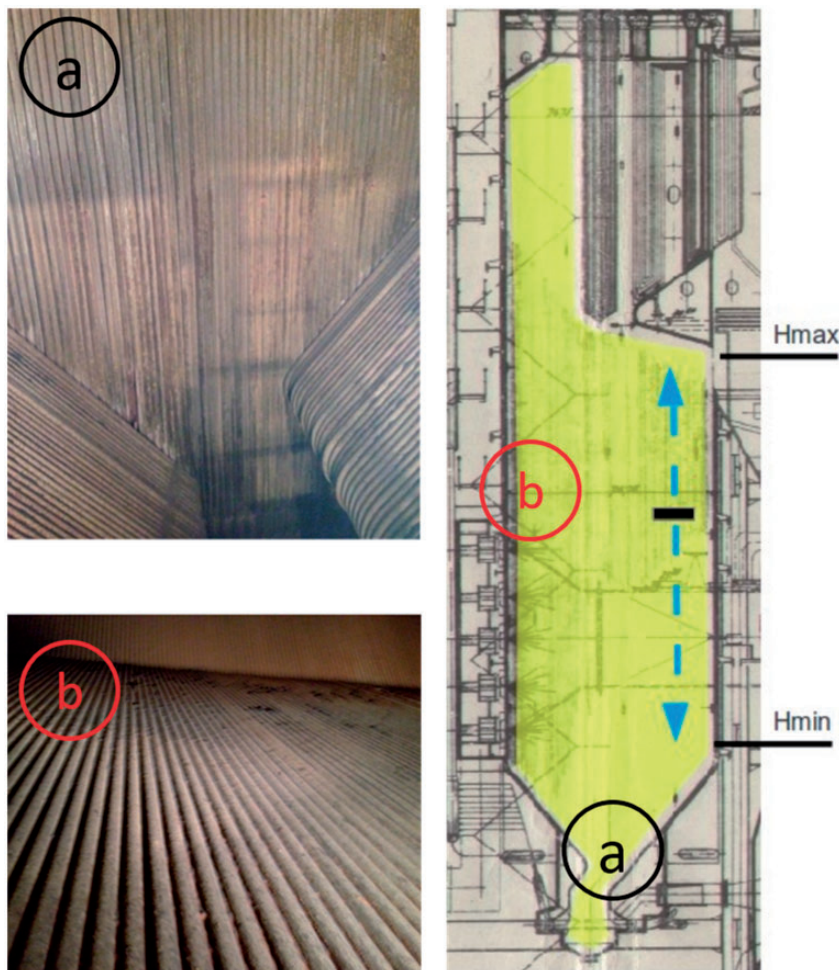


Figure 1. Section of the boiler chamber and details of the pipe structures. (a) Entrance of the chamber (funnel-shaped portion). (b) Pipes wall.

Regarding the sensor payload, it is worth noting that most inspection tasks are usually performed in GPS-denied environment which on the other hand cannot be structured since this activity is in general practically not feasible or it will vanish the benefit introduced by the aerial service robots usage. For this reason, the aerial platform has to estimate its own state relying only on embedded sensors and on-board computation resources.

From the end-user point of view, the proposed system has to be as user friendly as possible and in theory it should allow unexperienced pilots to fly the system in complex and cluttered environment. To this end, the system has to provide the pilot with assistive control features able to implement shared control capability, e.g. follow a line or move towards a point of interest. However, one of the main requests from the operators is to let the human in control of the inspection avoiding to completely automate the task. The main rationale under this request is the possibility to exploit the cognitive capability of the human being

and in particular, the experience gained over the years by the surveyor that will guide the inspection process focusing on critical points.

Challenges

Given the partial or total enclosure of confined spaces, GPS technology cannot be reliably used, and this requires solving the localization problem employing other sensing technologies. GPS-denied environment has been addressed in the past relying on vision-based localization.²⁶

Many times, industrial settings are surrounded by metallic structure or elements and these could interfere with electromagnetic signals and sensors. In particular, within the boiler scenario, the presence of pipes, tubes and similar elements greatly reduce the quality and the bandwidth of wireless communication and the reliability of compass in IMU sensor used to stabilize the vehicle. The former constraint does not allow to close the control loop externally exploiting high-power

computing unit. Hence, on-board computation requires automatic behaviors to overcome rapidly collisions and flight stability issues. Thus, the need of efficient algorithms runs with good performances on embedded hardware without consuming much battery power.

In addition, the boiler scenario presents (i) variable air pressures that limits the use of barometric sensors to estimate for instance the altitude of the robot, (ii) the presence of dust (iron dust in the case of combustion chambers) could generate visual occlusions, visual feature outliers and also interfere with the electronic equipment, (iii) lack of light which reduces vision algorithms performance requiring a custom designed illumination system.

Finally, yet importantly, typical industrial settings present repetitive textures and elements that affect correct data association of visual features in algorithms at the state of the art.

This work presents a system that overcomes all the challenges and limitations discussed above employing specific solutions to each problem. We have chosen a foldable MAV as hardware platform but with good payload capabilities, that make use of laser sensors for localization and navigation that are not affected by visual artifacts as could be the case for classical camera systems. The MAV is equipped with an illumination source, high-resolution cameras for inspection analysis and all the control algorithms runs on an embedded ARM-based control board. To obtain real-time performances on an embedded computing platform, a custom performant SLAM algorithm has been developed.²⁷ To estimate altitude inside confined chambers, two sets of sensors have been used, ultrasonic for proximity sensing and camera based for long range sensing. The communication between the MAV and the ground station is reduced to small footprint high-level control packets for inspection guidance. Moreover, the system here presented was designed to work in unstructured environments. This means that the proposed system does not require a specific environment nor external sensing unit (like markers for instance) to accomplish inspection tasks.

System description

The proposed system is composed of two elements. The first element is constituted by a portable computer used as operator control unit (OCU) in order to give high level teleoperation commands to guide the inspection task (see ‘Teleoperation interface’ section). The second element is an MAV equipped with an embedded processing board and a suite of navigation and inspection sensors. According to the previously discussed requirements, the selected vehicle is a ducted fan quadrotor designed by Cyber Technology (CyberQuad MAXI). It is a quadrotor equipped with

four brushless motors suitable to be used in critical environments (presence of flammable gases) since they do not produce sparks. The device has highly optimized ducted fans, allowing the platforms to be less than half the size of a helicopter rotor, with the same lifting efficiency. The protection allows easily flight through doorways, down hallways and through tight spaces without risking a rotor strike. The MAV dimension is $69 \times 56 \times 20$ cm.

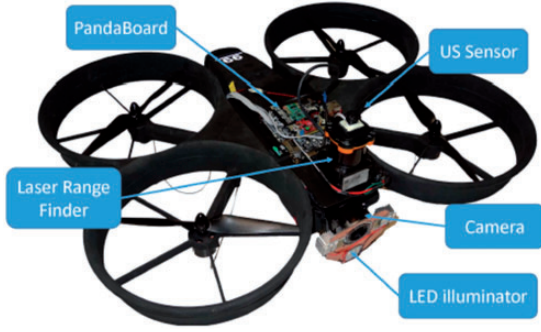
The CyberQuad MAXI is provided with the Navy control and Flight control from MikroKopter which embeds a fast motor controller and an IMU composed by three single axis gyro on all three axes (ADXRS610), three-axis accelerometer (LIS344ALH), barometric pressure sensor (MPX4115A) and a compass (HMC5843). The installed custom payload is composed by the following elements. A front mounted Sony camera (HDV-CX350VE) allows visibility over large elevation ranges and it is used by the remote operator for inspection purpose. In fact, the camera output is streamed throughout the Wifi link by means of a USB grabber (EasyCap DC60). The camera is mounted on a pan-tilt base which automatically stabilizes the pan with respect to the horizontal direction and allows the user to adjust the tilt towards the desired direction. Moreover, the optical camera zoom ($12\times$) is remotely controlled by the operator as well. Near the camera, a custom illumination system has been designed by means of several LEDs which assure the adequate amount of light for the inspection task. Two sonars (XLMaxSonarEZ4), one on the top and one on the bottom of the device, are used to detect upper and lower obstacles as well as to measure distances from the ceiling and the ground during the automatic ascending inspection flight and the takeoff and landing procedure, respectively.

In the case of high elevation industrial environments, the sonars are replaced by long-range distance measuring camera sensors (Leddar M16) without the need of modifying the algorithmic structure of the system. For the localization purpose, the environmental mapping and the fusion algorithms, able to improve the attitude estimation, a Hokuyo laser (UTM-30LX) has been mounted on the top of the device. All these components have been powered and interfaced with a Pandaboard by means of a custom electronic board which provides the required voltage regulation as well as the levels transition. The Pandaboard is the computing unit of the system and it is also responsible for the external communication with the OCU ground station. Table 1 summarizes major design information.

The PandaBoard is a low-power, low-cost single-board development platform based on the Texas Instruments OMAP4430 which features a dual-core 1 GHz ARM CortexA9 MPCore CPU, a 304 MHz PowerVR SGX540 GPU, IVA3 multimedia hardware

Table 1. Characteristics of the proposed MAV system.

Dimensions	690 × 560 × 200 mm
Payload	0.8 kg
Maximal speed	10 m.s ⁻¹
Operating life	15 min
Embedded sensors	2 Ultrasound sensors or 2 Leddar Cameras 1 2D laser scanner Maximum range: 30 m Wide angle: 270° Angular resolution: 0.25° 1 IMU: 3 axis accelerometer 3 axis gyro barometric sensor compass sensor 1920 × 1080 camera with optical zoom (12×)



accelerator with a programmable DSP, and 1 GB of DDR2 SDRAM. The connectivity is provided by wired 10/100 Ethernet as well as wireless Ethernet and Bluetooth. It also has two USB host ports and one USB OnThe-Go port, supporting USB 2.0. In our system, the device runs Ubuntu Linux distribution.

Framework architecture

The system can be operated with high level commands by an operator for inspection purposes or alternatively, by an automatic navigation component that uses a harmonic potential field (HPF). To maintain a stable flight during the inspection task, the system acquires information from the environment by means of a laser range finder (LRF) and two distance measuring sensors. This information is processed by an SLAM component that feeds the current position and the asset of the MAV to the navigation and the low-level control components. In parallel, a collision avoidance element, independent from the SLAM algorithm, is responsible for avoiding collisions with external objects. The underlying components of the software system are depicted in Figure 2.

The sensor acquisition and the algorithm computation are performed by the embedded computing system, i.e. the PandaBoard. The next sections will introduce and discuss each module component in detail.

Control algorithms

This section introduces the MAV dynamic model employed to design the low-level control algorithms, the global and relative reference frames for the equations of motion, the algorithm responsible for the localization, mapping and feature extraction procedures and

finally the velocity estimation and collision avoidance algorithms.

Dynamic model and low-level control

The dynamic model used for the analysis and development of the control algorithms is a simplified model based on the work by Martinez.²⁸ Considering the world and the robot reference frames as in Figure 3, if we define with F_f , F_b , F_l , F_r the thrust forces exerted by the front, back, left and right rotors blades, we can write the simplified dynamical model as

$$F_t = F_f + F_b + F_l + F_r$$

$$F_\theta = F_f - F_b$$

$$F_\phi = F_l + F_r$$

$$F_\psi = -F_f - F_b + F_l + F_r$$

$$\ddot{x} = \frac{F_t(\sin\psi\sin\phi + \cos\psi\sin\theta\cos\phi) - k_1\dot{x}}{m}$$

$$\ddot{y} = \frac{F_t(\sin\psi\sin\theta\cos\psi + \cos\psi\sin\phi) - k_2\dot{y}}{m}$$

$$\ddot{z} = \frac{F_t\cos\theta\cos\phi - k_3\dot{z}}{m} - g$$

$$\ddot{\theta} = \frac{(F_\theta - k_5\dot{\theta})l}{I_y}$$

$$\ddot{\phi} = \frac{(F_\phi - k_4\dot{\phi})l}{I_x}$$

$$\ddot{\psi} = \frac{(F_\psi - k_6\dot{\psi})l}{I_z}$$

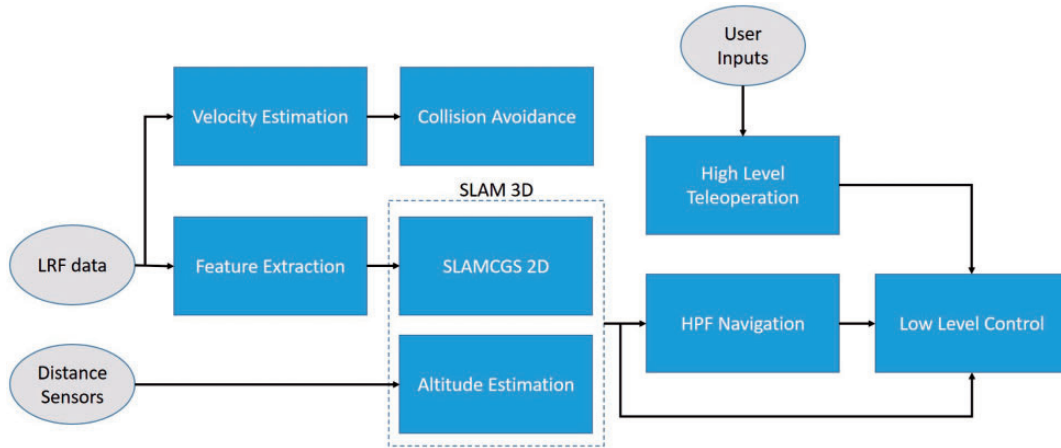


Figure 2. Software components relationship diagram.

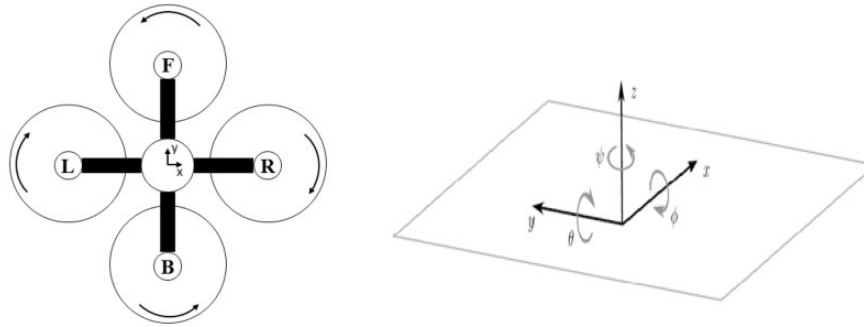


Figure 3. MAV reference frame (body frame) on the left and World reference frame (absolute frame) on the right. The reference frames follow the right hand convention.

MAV: micro aerial vehicle.

where I_x , I_y and I_z are the inertia moments with respect to the x , y and z axes, θ , ϕ and ψ are the pitch, roll and yaw angles, $k_{i=1,2,3,\dots,6}$ are aerodynamic drag coefficients and g is the gravity vector.

Transfer functions for the roll, pitch and yaw responses, taking into account also the embedded flight control board response, have been modeled with a Box–Jenkins model²⁹ resulting in first-order systems.

The low-level control has been implemented decoupling the x and y axes motion. For the pitch and roll angles, the control systems are composed by two feedback loops. The outer loop is responsible for the regulation of the maximum velocity of the vehicle as a function of the distance from the reference input, while the inner loop regulates the asset angle (pitch or roll) taking as reference the velocity computed from the outer loop. For the yaw rate, the control system is composed by a single loop that regulates the angular velocity of the vehicle to follow a reference angle. Each control loop has been implemented with PID regulators. In particular, the low-level control of the pitch angle takes into account also the presence of a bias

that is present in the real system and changes at each switching on/off of the vehicle. Without a manual compensation of the bias, the low-level control provides a compensation strategy in the first few seconds after takeoff.

SLAM

The proposed system localizes itself in the environment by means of two independent SLAM algorithms which will be discussed within the following sections. The first SLAM algorithm is used to estimate the pose of the MAV on a plane, while the second SLAM algorithm is used to estimate the altitude of the MAV. Fusing both the information, the complete 3D pose of the multi-rotor can be obtained.

Rao–Backwellized particle filter. The Rao–Backwellized particle filter (RBPF)³⁰ is a Bayesian filter method, which approximates the posterior probability by a set of sample particles drawn from the posterior. In such a framework, each particle represents a robot path and a

map. The key idea of RBPF is to decompose the joint posterior probability into a posterior probability of the map M and a posterior probability of the trajectory X . In particular, the solution implements the Montemerlo's factorization³¹ resulting in

$$p(X_t, M | Z_t, U_t, D_t) = p(X_t | Z_t, U_t, D_t) \prod_{n=1}^N p(m_n | X_t, Z_t, U_t, D_t)$$

where t represents the time instant, M is composed of N features $\{m_1, m_2, \dots, m_N\}$. Z_t is the measurements set at time t , U_t is the control sequence of the robot and D_t is the data association. Thanks to this factorization, it is possible to estimate the N features independently by means of low-dimensional extended Kalman filters (EKF).

Hence, the posterior probability of the trajectory is computed by a particle filter and then the map is updated according to the current measurements and the trajectory posterior contribution.

Map representation. As mentioned before, the map M is defined as a collection of features or landmark points. Considering that the majority of indoor environments are typically enclosed and divided by walls or elements that could be assimilated to walls, the SLAM algorithm presented in the next section uses as map features a special set of parameters that are used to define walls. Nevertheless, in the case of confined spaces where the environment is enclosed by curved surfaces, it is possible to substitute the feature representation and make use of the optimized embedded SLAM algorithm as well.

Using the Hessian representation (Figure 4), the wall coordinates are given by the triplet (r, α, v) :

- $r = \|\mathbf{OP}^*\|$ is the distance from O to the closest point P^* in the wall.
- α represents the counter-clockwise angle between the versor \mathbf{i} of the x axis and the outward normal \mathbf{n} of the wall. α belongs to the interval $[0, 2\pi[$ and is computed using the function acos2 defined below where the function \det computes the determinant of two vectors

$$\alpha = \text{acos2}(\mathbf{i}, \mathbf{n}) = \begin{cases} 2\pi - \text{acos}(\mathbf{i}, \mathbf{n}), & \text{if } \det(\mathbf{i}, \mathbf{n}) < 0 \\ \text{acos}(\mathbf{i}, \mathbf{n}), & \text{if } \det(\mathbf{i}, \mathbf{n}) \geq 0 \end{cases}$$

- $v = -\mathbf{O} \cdot \frac{\mathbf{P}^* \mathbf{n}}{\|\mathbf{OP}^*\|}$ states if the frame O is front of the wall ($v = 1$) or behind it ($v = -1$)

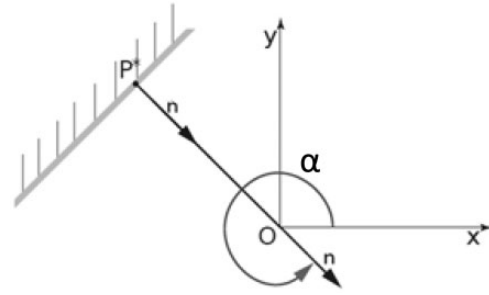


Figure 4. Hessian representation of wall coordinates with respect to the frame O

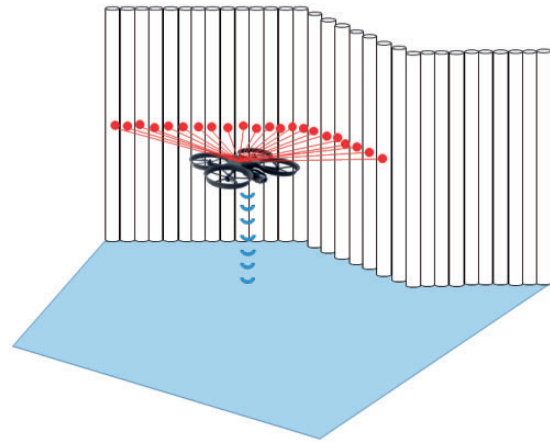


Figure 5. Schematic view of MAV inspection operation with laser and altitude sensors for SLAM purposes. Here the wall made of tubes resembles the interior of an industrial combustion chamber.

MAV: micro aerial vehicle; SLAM: simultaneous localization and mapping.

Embedded SLAM algorithm

The SLAM algorithm here presented is used to determine the robot pose on a plane $(x, y, \psi)^T$, and estimate features locations on such plane (Figure 5). The algorithm makes the assumption that the MAV flies maintaining an asset which could be approximated parallel to the ground reference frame. This assumption is easily fulfilled when the platform moves at low speed and thus avoiding aggressive maneuvers. Thanks to this assumption, it is possible to recover the robot pose in a plane employing an LRF sensor. The device altitude on the other hand, has to be obtained separately.

Fusing the particle filter robot pose estimation on a 2D map with an estimation filter of the altitude ('Altitude estimation' section), the complete pose estimation can be obtained.

The developed algorithm²⁷ consists of eight main steps listed in Algorithm 1, each k particle in the Particle set is described by its pose x_i^k and its own

map with N^k features represented by the mean and covariance pair: $(f_{n,t}^k, F_{n,t}^k)$. Each j th feature of the k th particle has a corresponding visibility counter: $i_{j,t}^k$ used to discard unreliable features. In essence, the k th particle is described as follow

$$x_{t-1}^k; \left\{ (f_{1,t-1}^k, F_{1,t-1}^k, i_{1,t-1}^k); \dots; (f_{N_{t-1}^k,t-1}^k, F_{N_{t-1}^k,t-1}^k, i_{N_{t-1}^k,t-1}^k) \right\}$$

Each algorithm cycle starts from the state obtained from the previous step and incorporates the input u_t and the measurement vector Z containing the features extracted at time step t .

In the following sections, the algorithm key points are discussed.

Pose prediction. To predict the MAV pose, data available from the embedded IMU are integrated in an odometry model. Without considering thermal drift

Algorithm 1. Overview of the Embedded SLAM algorithm

FastSLAM_CGS (Particles)

1. **for** each particle in *Particles* **do**
 2. Predict pose
 3. Assign covariance pose
 4. Find data association
 5. Update pose
 6. Sample the particle pose
 7. Update particle features
 8. Remove dubious features
 9. **end for**
 10. Resample(*Particles*)
 11. **Return** *Particles*
-

terms, we can write the relationship between raw IMU readings and true signals as

$$a_{Measured} = a_{IMU} + R_{IMU}^w \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + b_a + \mathcal{N}(0, \sigma_a)$$

$$\omega_{Measured} = \omega_{IMU} + b_g + \mathcal{N}(0, \sigma_g)$$

where $\omega_{Measured} \in \mathbb{R}^3$ and $a_{Measured} \in \mathbb{R}^3$ indicate the measured angular rate and acceleration, $\omega_{IMU} \in \mathbb{R}^3$ and $a_{IMU} \in \mathbb{R}^3$ are the true signals; $b_a \in \mathbb{R}^3$ and $b_g \in \mathbb{R}^3$ are slowly varying bias terms for the accelerometer and gyroscope, g is the gravity acceleration constant. Zero means Gaussians model the measurement noises. The rotation matrix $R_{IMU}^w \in SO(3)$ is used to transform from world coordinates to IMU frame coordinates.

It is possible to write the accelerations in the world frame following the matrix transformation³²

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = R_{IMU}^w a_{IMU} + \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}$$

$$R_{IMU}^w = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}$$

In the previous equation, c and s state for the cosine and the sine function, respectively, and ϕ , θ , ψ are the roll, pitch and yaw angle obtained from the gyro sensor.

We can thus write the prediction for the k th particle as

$$\hat{\mu}_t^k = \begin{bmatrix} x_t^k \\ y_t^k \\ \psi_t^k \end{bmatrix} = \hat{g}(x_{t-1}^k, u_t) = \begin{bmatrix} x_{t-1}^k + v_{x,t-1}^k dt + \frac{a_x dt^2}{2} \\ y_{t-1}^k + v_{y,t-1}^k dt + \frac{a_y dt^2}{2} \\ \dot{\psi} dt \end{bmatrix}$$

where $\dot{\psi}$ is the yaw rate given by the gyro sensor (ω_{IMU}); v_x and v_y are components of the velocity vector estimated in ‘Velocity estimation’ section.

The covariance of the robot pose is set equal to the covariance of the Gaussian white noise that models the measurement error of the IMU sensor.

Feature extraction. Having at disposal an embedded computing unit, and considering the need to process a large amount of laser range data, an abstraction layer has been developed. This layer represents a virtual sensor that produces high level data starting from raw laser range readings. This high-level data are usually called features or landmarks in mapping literature. As discussed above, the Hessian representation of walls has been selected as appropriate feature for confined space inspection purpose. To be able to detect features, our implementation starts from the Split and Merge algorithm³³ that is a fast and performant solution to the problem. With respect to the original implementation, we introduce filtering kernels to obtain a more reliable and precise measurement. In particular, five filters have been designed and are applied at the output of the Split and Merge algorithm:

- **Cut segment's edges.** This filter is introduced to remove artifacts in the neighborhood of edges. It removes a certain number of points from the beginning and from the end of every set of points obtained from the split and merge algorithm.
- **Remove too scattered segments.** If the return laser signal is not correctly received, some measurements are scattered. In the case that there are too many holes in the obtained segment, the segment is filtered out because not reliable.
- **Point of view filter.** This filter removes segments observed from an adverse point of view, which is when the points are on a surface far away from the source or with a big incident angle.
- **Merge non-consecutive segments.** This is used to filter redundant measures. If an obstacle interrupts a wall, the algorithm obtains two features with the same Hessian coordinates: one for the points preceding the obstacle and one for the followers. With this filter, the two features are fused together.
- **Remove too short segments.** Features composed by a small number of points are considered unreliable and filtered out.

The correspondences between the detected features and the already known landmarks are found using maximum likelihood (line 4 of SLAM algorithm overview). If the obtained likelihood is under a certain threshold p_0 , the current detected feature is considered as a new landmark.

Figure 6 shows the outcome of the proposed algorithm compared with the standard Split and Merge algorithm. The result of the standard

approach is depicted in blue, whereas the refined outcome after passing the filtering stage is depicted in red. The algorithm reconstructs four segments whose details are shown in the right part of the figure.

It is worth to point out that the detected features are compute with respect to the MAV reference frame. If global mapping is required, a transformation into the global map reference frame is required.

Finally, note that the threshold p_0 as well as the ones used in the filtering process is chosen by experience in a trial and error phase.

Measurement model. Prediction target response for planes³⁴ is used to predict the LRF measurement. We consider the feature f_n modelled by the Hessian triplet (r_n, α_n, v_n) and the MAV pose $x = (x_t, y_t, \psi_t)^T$.

The measurement prediction \hat{z} is computed as follow

$$\hat{z} = \begin{bmatrix} \hat{r} \\ \hat{\alpha} \end{bmatrix} = h(f_n, x_t) = \begin{bmatrix} v_n(r_n - x_t \cos(\alpha_n) - y_t \sin(\alpha_n)) \\ \alpha_n - \psi_t \end{bmatrix}$$

the Jacobian of the measurement prediction with respect to the landmarks is computed differentiating h with respect to the map features as follow

$$H_F = \nabla_F h(f_n, x_t) = \begin{bmatrix} v_n & v_n(x_t \sin(\alpha_n) - y_t \cos(\alpha_n)) \\ 0 & 1 \end{bmatrix}$$

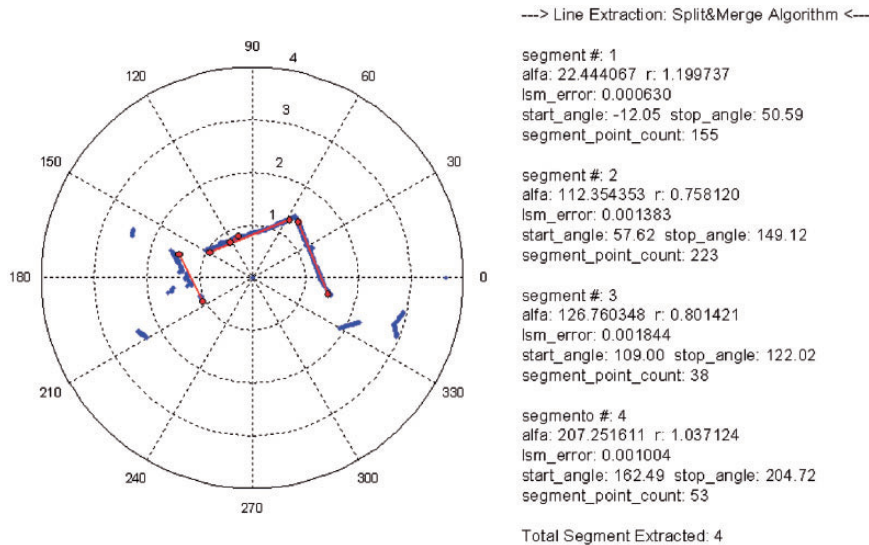


Figure 6. Output of the split and merge algorithm with custom filtering.

On the other hand, the Jacobian H_x of the measurement prediction with respect to the pose is computed differentiating h with respect to the state vector as follows

$$H_x = \nabla_x h(f_n, x_t) = \begin{bmatrix} -v_n \cos(\alpha_n) & -v_n \sin(\alpha_n) & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Resampling phase. The *importance resampling* step is performed with a classic low variance sampler.²⁰ Before such a step, the unreliable particles features are removed by means of a visibility counter and a threshold. If a measurement is not associated with any feature in the particle map, it is considered as a new landmark and its visibility counter is initialized such as $i_j^k = 2$.

On the other hand, when a feature is associated to a detected landmark, its counter is incremented keeping it alive ($i_j^k = i_j^{k-1} + 2$).

In order to remove unreliable features, at each cycle, every visible particle has its counter decremented. When the particle visibility counter goes under a certain threshold, the feature is considered unreliable and removed.

Velocity estimation

To reduce the computational time of the implemented SLAM algorithm, the vehicle velocity estimation is performed by an EKF. In this way, the estimation is performed once using the output of the SLAM algorithm and not inside each particle, thus reducing the particles dimensions and the computational time. The dynamic equations involved in the prediction step are the following

$$\tilde{g}_t = \begin{cases} x_t = x_{t-1} + [\dot{x}_{t-1}^{(b)} \cdot \cos(\theta_{t-1}) - \dot{y}_{t-1}^{(b)} \cdot \sin(\theta_{t-1})] \cdot dt + \mathcal{N}(0, \sigma_x) \\ y_t = y_{t-1} + [\dot{x}_{t-1}^{(b)} \cdot \sin(\theta_{t-1}) + \dot{y}_{t-1}^{(b)} \cdot \cos(\theta_{t-1})] \cdot dt + \mathcal{N}(0, \sigma_y) \\ \theta_t = \theta_{t-1} + \mathcal{N}(0, \sigma_\theta) \\ \dot{x}_t = \dot{x}_{t-1} + \mathcal{N}(0, \sigma_{\dot{x}}) \\ \dot{y}_t = \dot{y}_{t-1} + \mathcal{N}(0, \sigma_{\dot{y}}) \end{cases}$$

While the position is expressed with respect to the global reference frame, the velocity is computed with respect to the vehicle body frame $\{b\}$.

The correction step uses as sensor the output of the SLAMCGS algorithm and adds a Gaussian noise $\mathcal{N}(0, Q_t)$ to model the measurement error.

Collision avoidance

For safety purposes, the system has been equipped with an algorithm estimating the distance of the MAV from the obstacles. This module is independent from the localization module and it computes both the objects distance and approaching speed exploiting the laser sensor. In particular, the sensor span (270 degrees) has been divided in six regions (45° each) in which the minimum distance measure ρ_t is selected.

The approaching speed (v_ρ) is estimated by means of an EKF filter. From this information, the algorithm computes the time to collision (TTC). The TTC is an estimate on the time in seconds before a probable impact considering a constant velocity profile of the MAV. Based on the TTC, the system is able to alert the operator or respond in order to prevent collisions. The prediction step of the EKF is given by

$$\tilde{g}_t = \begin{cases} \rho_t = \rho_{t-1} + v_{\rho t-1} \cdot dt + \mathcal{N}(0, \sigma_\rho) \\ v_{\rho t} = v_{\rho t-1} + \mathcal{N}(0, \sigma_{v_\rho}) \end{cases}$$

The sensor used in the correction step is the obstacle sensor that provides the minimum distance from each obstacle (ρ) in each sector. A Gaussian noise $\mathcal{N}(0, Q_t)$ is added to model the measurement errors.

Considering the state vector of the EKF filter, for each sector i , the estimated TTC is given by

$$TTC_i = -\frac{\rho_i}{v_{\rho_i}}$$

Based on this value, it is possible to decide if the MAV is in safety or if there is a possible danger and the system should intervene to prevent collisions with the environment.

Considering Figure 7, if TTC_i is greater than a certain threshold TTC_{Max} , the MAV is considered safe in that sector, if $TTC_i < TTC_{Max}$, a thrust factor is computed in the range $\alpha_i \in (0, Thrust_{OD})$. For each sector the direction of the thrust vectors is computed as shown in Figure 7. Once obtained all the thrust factors α_i , the total thrust vector is given as $\vec{v}_{TOT} = \vec{v}_1 \cdot \alpha_1 + \vec{v}_2 \cdot \alpha_2 + \vec{v}_3 \cdot \alpha_3 + \vec{v}_4 \cdot \alpha_4 + \vec{v}_5 \cdot \alpha_5 + \vec{v}_6 \cdot \alpha_6$.

Depending on the chosen setup, if the system is in control, a thrust force is applied in the direction of the thrust vector in order to prevent collisions with the environment, alternatively if the operator is in control, the system fires an alarm on the operator GUI.

Altitude estimation

As pointed out in the challenges description, the barometric sensor readings are affected by the temperature

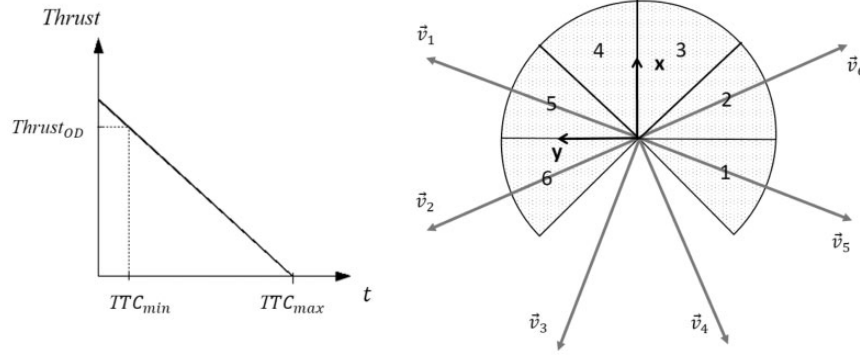


Figure 7. Time to collision graph and thrust vectors.

gradient of the environment. This, in turn, results in a low-precision accuracy estimation of the altitude. Moreover, estimating the altitude by means of a Kalman Filter and a downward looking sensor is not robust since the drone can fly over obstacles or even a non-flat ground.

For this reason, in the proposed system, the drone height is computed from upwards and downwards looking sensors data using an SLAM algorithm consisting of Kalman filters. The altitude estimation method firstly designed for an indoor multi-floor building exploration,⁹ adds upwards looking sensor data to the formulation by Gronska et al.³⁵ As anticipated in the hardware description ('System description' section), the system can be equipped alternatively with sonars sensors or with long range camera sensors. The camera sensors option allows greater range of distance measurement and is composed by an embedded module containing both a camera and an LED emitter. The Leddar M16 sensor does not provide punctual information but measures distances of objects within 16 sectors of 2.8° aperture from the camera viewpoint. We can consider both sensors equivalent from an algorithmic point of view.

The first step of the algorithm is to compute the drone height and vertical velocity according to the model prediction, as detailed below

$$\hat{x}_t = \begin{bmatrix} \hat{z}_t \\ \hat{v}_{z,t} \end{bmatrix} = A \begin{bmatrix} z_{t-1} \\ v_{z,t-1} \end{bmatrix} + B a_z$$

$$\text{with } A = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix}; \quad B = \begin{bmatrix} 0.5dt^2 \\ dt \end{bmatrix}$$

$$\hat{\Sigma}_t = A \Sigma_{t-1} A^T + R \quad \text{with } R = \begin{bmatrix} \sigma_z^2 & \sigma_z v_z \\ \sigma_z v_z & \sigma_{v_z}^2 \end{bmatrix}$$

Then, the ground and ceiling elevation beneath and above the drone are predicted according to the measurement prediction

$$\hat{h}_t = \begin{bmatrix} \hat{h}_{ground} \\ \hat{h}_{ceiling} \end{bmatrix} = \begin{bmatrix} \hat{z}_t - z_{downwards} \\ \hat{z}_t + z_{upwards} \end{bmatrix}$$

The next step is to find the matches with the already known levels close to the robot current pose (the set of corresponding levels C). Once obtained the matches, those are merged into a single level elevation. Each level is represented in the levels map L by its own pose $(x_l, y_l, \mu_l)^T$ and uncertainty σ_l .

To merge N levels from C , one can prove that it can be achieved using the next equation

$$(\mu_{1:N}, \sigma_{1:N}^2)^T = \text{mergeLevels}(C)$$

$$= \left(\frac{\sum_{k=1}^N \mu_k \prod_{j=1, j \neq k}^N \sigma_j^2}{\sum_{k=1}^N \prod_{j=1, j \neq k}^N \sigma_j^2}, \frac{\prod_{k=1}^N \sigma_k^2}{\sum_{k=1}^N \prod_{j=1, j \neq k}^M \sigma_j^2} \right)^T$$

After merging the levels elevations, the drone altitude and the current levels elevations beneath and above the drone are updated as follows.

$$Q_t = \sigma_{laser} + \sigma_{ground}$$

$$K = \widehat{\Sigma}_t D^T (D \widehat{\Sigma}_t D^T + Q)^{-1}$$

with $D = [1 \ 0]$

$$x_t = \begin{bmatrix} z_t \\ v_{z,t} \end{bmatrix} = \hat{x}_t + K(\mu_{ground} + z_{downwards} - D \hat{x}_t)$$

$$\Sigma_t = (I_2 - KD) \widehat{\Sigma}_t$$

$$\hat{h}_t = \begin{bmatrix} \hat{h}_{ground} \\ \hat{h}_{ceiling} \end{bmatrix} = \begin{bmatrix} z_t - z_{downwards} \\ z_t + z_{upwards} \end{bmatrix}$$

Updating the drone altitude and levels from the ceiling measurement, implies to modify the above equation such as

$$x_t = \hat{x}_t + K(\mu_{ceiling} - z_{upwards} - D\hat{x}_t)$$

Once updating the drone altitude, the elevation levels are updated using for each one a Kalman filter. Then, the new levels are inserted into the levels map L .

Eventually, the final step of the algorithm aims to merge closest levels. It only considers the levels close to the current drone pose. Afterward, within this subset, the algorithm compares the difference between the levels elevation, if it is under a certain threshold δ_2 , merges them according to $mergeLevels(C)$.

Navigation

Once the localization problem has been solved, the environment navigation and thus the inspection procedure can be addressed. Two navigation modalities have been proposed: (i) *teleoperation modality* – the operator drives the system with high-level commands using the custom designed OCU; (ii) *autonomous modality* – the MAV navigates autonomously the surroundings using a HPF in order to have a full coverage of the environment to be inspected.

Both the approaches are introduced in the following paragraphs.

Teleoperation interface

To provide high-level teleoperation functionalities for personnel without expertise in drone flight, the system is equipped with an OCU that allows the user to guide the inspection task via a simple interface. In particular, the operator can use a joystick to command the direction of motion of the MAV and a display shows both the captured video stream, the reconstruction of the local map and, if selected, internal algorithm parameters (see Figure 8).

The OCU has been designed in order to allow the operator to perform the inspection from a remote and safe location. The OCU is composed by a Notebook which runs the developed graphical user interface (GUI) and a joystick which is used by the operator to move the camera point of view as well as to set the desired device pose and to control the remote functionalities (like for instance starting/stopping video recording, modify the camera zoom, or select inspection points of interest). By means of the joystick, the operator can also move a virtual point which is then followed by the vehicle control system satisfying additional safety constraints (obstacles avoidance and device autonomy). The GUI has been realized by a C++ based Qt (Digia) application and it is composed by two main windows that can be arranged on the screen in a custom way. On the right side of the windows, all the information related to the vehicle, like for instance the estimated position and velocity, are shown. The current map and the measured features with respect to the device position are also presented in a polar view. The two windows have been specifically designed for the two operators that in general perform the inspection. One is in charge for the visual

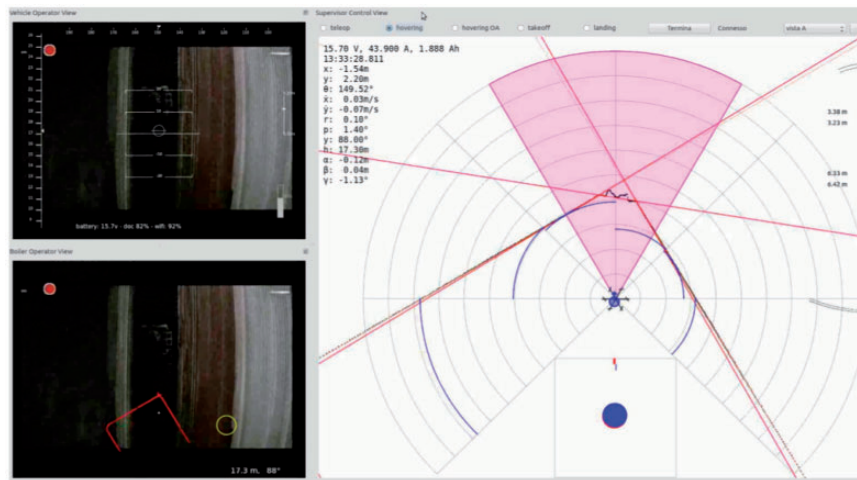


Figure 8. Operator control unit displaying frontal camera views with current laser scan and MAV asset parameters of the internal SLAM control as well as estimated features and collision distances.

MAV: micro aerial vehicle; SLAM: simultaneous localization and mapping.

inspection, whereas the other controls or supervises the MAV navigation aspects.

Virtual potential function for autonomous exploration

To allow some degrees of autonomy during the environment exploration and to leverage the operator effort during the inspection task, the system implements a potential field based path planning algorithm, firstly introduced in Khatib.³⁶

Among the possible approaches in literature, in frontier-based exploration³⁷ methods, the frontiers represent the borders between the known free space and the unknown environment. These are computed using algorithms that are similar to edge detectors. Eventually, the robot moves towards the nearest frontier following the shortest path that avoids obstacles computed with a best-first search algorithm.³⁸ In Kim and Eustice³⁹ the robots perform a default policy exploration based on the Boustrophedon motion^{40,41} and revisits the interesting areas based on their visual saliency following a path defined again by a best-first search algorithm. In Shen et al.,⁴² virtual gas molecules are used to detect unknown areas, these particles move toward free and unknown space with a Brownian motion, colliding with already known obstacles. Once detected, the robot moves towards the navigation goals. Finally, in Silva et al.,⁴³ a potential field is incrementally computed using harmonic functions where unknown areas have an attractive potential, whereas obstacles have a repulsive potential. Therefore, the robot is attracted towards the nearest unexplored space.

The approach presented in Silva et al.⁴³ has been selected as exploration policy for its efficiency, easy implementation and above all because it avoids the use of best-first algorithm which can be time-consuming if the goal is far from the robot. The algorithm employs harmonic functions⁴⁴ to solve the local minima issue.⁴⁵

Our approach (shown in Algorithm 2) differs from the work by Silva et al. substituting the histogrammic in-motion mapping (HIMM)⁴⁶ with an occupancy grid environment map computed as in Pepe et al.⁹ The basic idea of the occupancy grid algorithm is to compute the posterior probability of the map only knowing the observation data and the previously estimated robot path. The space is partitioned into small grid cells where each cell o_k has an occupancy probability $p(o_k)$ which takes a value between 0 (free) and 1 (occupied). Due to performance constraints, the map posterior is approximated by computing the posterior for each discrete cell within the partitioned space O , and has been proved in Thrun et al.²⁰ that its occupancy probability can be computed recursively using Bayes rules..

Algorithm 2. Harmonic Potential Field Based Exploration

HPF_Exploration(X_t, O, Φ)

// X_t the robot pose, O the occupancy grid, Φ the potential field grid

1. **1/1. Update the sliding window potential field until reaching the desired accuracy**
2. **while** $\epsilon_{max} > \epsilon_0$ **do**
3. **for each cell** i in Φ **do**
4. $\phi(x_t, y_t) = 1$ **//Set the current robot cell potential value**
5. **if** $\sqrt{(x_t - x_i)^2 + (y_t - y_i)^2} < R$ **do** **//check if the cell belongs to the sliding window**
6. $\phi_{old} = \phi(x_i, y_i)$
7. **if** $p(o_{kZ_{1:k}}, x_{1:k}) > p_{obstacles}$ **do**
8. $\phi(x_i, y_i) = 1$ **//Set the obstacles potential value**
9. **else**
10. $\phi(x_i, y_i) = Gauss_Seidel(x_i, y_i, \Phi)$
11. **end if**
12. $\epsilon = \left| \frac{\phi^k(x, y) - \phi_{old}}{\phi_{old}} \right|$
13. **if** $\epsilon > \epsilon_{max}$ **do**
14. $\epsilon_{max} = \epsilon$
15. **end if**
16. **end if**
17. **end for**
18. **end while**
19. **2/2. Compute the numerical gradient on the current robot cell** (x_t, y_t)
20. $G_x = \frac{\partial \phi}{\partial x}(x_t, y_t) \approx \frac{1}{2}(\phi(x_t + 1, y_t) - \phi(x_t - 1, y_t))$
21. $G_y = \frac{\partial \phi}{\partial y}(x_t, y_t) \approx \frac{1}{2}(\phi(x_t, y_t + 1) - \phi(x_t, y_t - 1))$
22. **3/3. Compute the new reference position**
23. $X_R = (x_R, y_R, \Phi_R)$
24. $x_R = x_t + \frac{-G_x}{\sqrt{G_x^2 + G_y^2}}$
25. $y_R = y_t + \frac{-G_y}{\sqrt{G_x^2 + G_y^2}}$
26. $\psi_R = atan2(\frac{-G_y}{\sqrt{G_x^2 + G_y^2}}, \frac{-G_x}{\sqrt{G_x^2 + G_y^2}})$
26. **return** X_R, Φ

Starting from the estimated environment map, the potential field is updated within a sliding window, centered on the MAV. This means that cells which belong to the circle of a certain radius R centered on the robot pose have their potential updated. The ones with a high occupancy probability (above a certain threshold) are considered as obstacles and therefore their potential is set equal to 1. The free cells are updated according to the Gauss-Seidel method while taking into account cells at the borders. This procedure is repeated until a certain accuracy is obtained.

At the beginning of the exploration, all cells have their potential set equal to 0, and therefore unknown areas are cells which have never been updated inside

the sliding window, they represent navigation goals and the robot is attracted to them. As suggested in Shade and Newman,⁴⁷ the potential value of the cell containing the robot is set equal to 1 (line 4).

Once reaching the desired accuracy, the negative gradient is numerically computed. Eventually, in order to follow an optimal path towards the goal position (unknown space), a gradient descent method as described in Gupta et al.⁴⁸ has been adopted.

Post processing and structure reconstruction

The elements and features to inspect can be various depending on the environment itself. For this reason, we did not focus on the inspection features that can be assessed by qualified personnel through visual inspection. Having at disposal all the information of the mapping and localization algorithm is however possible to generate offline a 3D model of the explored environment for detailed analysis. It is in fact possible to associate frames captured by the frontal camera with the reconstructed positions of the MAV, and thus generate a 3D Mosaic of images, or better to employ an offline optimization technique known in literature as structure from motion (SfM)⁴⁹ and create a complete 3D reconstruction of the environment. An example of reconstruction of a sector of pipe wall captured inside an industrial boiler is shown in Figure 9.

Test flight in an industrial combustion chamber

Several test flights have been carried out inside industrial combustion chambers to verify the robustness of

the system. In this section, a preliminary test flight is presented and discussed. The goal of the preliminary demonstration flight was to prove the effectiveness of the semi-autonomous aerial vehicle in the execution of an inspection task concerning a combustion chamber's conditions. The main sensor used to conduct the inspection was a camera in the visible light spectrum.

In details, the task consisted in a visual analysis of a boiler's internal wall and of the burners' rows that are located on the internal edges of the chamber.

The chamber was already prepared by means of an internal scaffolding to provide a takeoff platform for the vehicle. This platform should have been placed immediately above the funnel-shaped portion of the chamber; instead, the scaffolding was installed around 5 m below the hopper.

Before the actual flight, proper communication between the vehicle and the OCU has been verified. A brief test flight has been conducted to verify overall system correct functioning. Unfortunately, random interruption in the WiFi communication channel, likely due to the metallic structure of the combustion chamber has been noticed.

For the demonstration, a task sequence following the takeoff of the vehicle has been planned:

(1) An ascent phase to reach the hopper limit followed by an approach phase to the left edge of the chamber. (2) A following turn of the vehicle's camera (front of the vehicle) toward the burners and subsequent ascent of the vehicle up to 30 m (altitude reported by the vehicle's altimeter). (3) A second turn of the vehicle in order to have the vehicle's camera orthogonal with respect to the chamber's wall, to get best viewing condition and a descent to the hopper limit. (4) Horizontal movement on the right for some

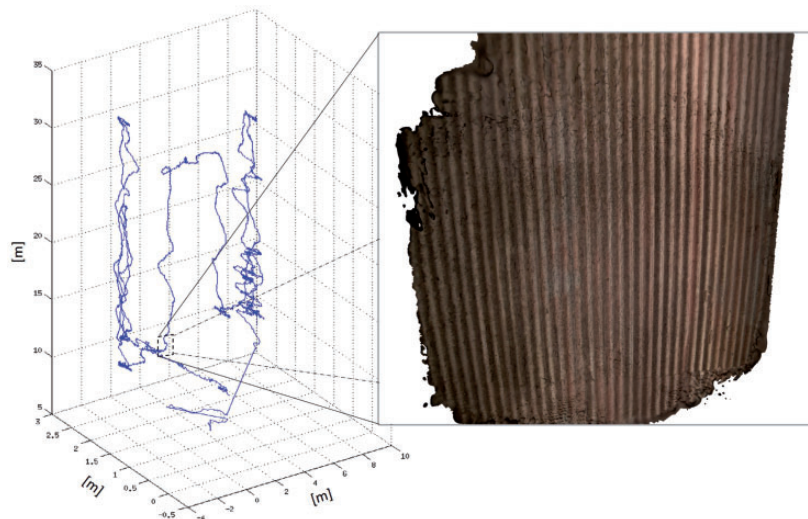


Figure 9. Estimated trajectory during an inspection and SfM reconstruction example of a sector of the environment from 50 frames. SfM: structure from motion.

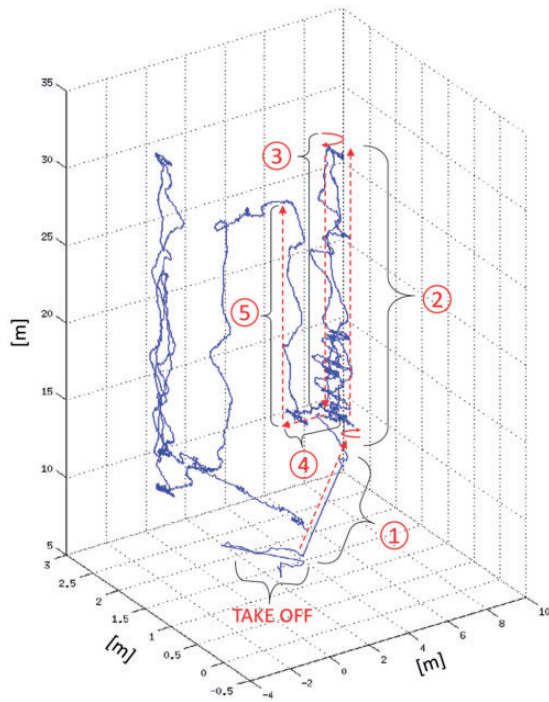


Figure 10. Test flight in an industrial combustion chamber. The intended sequence phases are shown. After take-off, phase (1) is an approach to the left edge of the chamber. (2) MAV turn and ascension up to 30 m. (3) MAV turn and descent. (4) Lateral movement. (5) The sequence of action repeats until the entire wall is inspected.

MAV: micro aerial vehicle

meters to scan another section of the chamber's wall. (5) These steps are repeated in order to cover all the wall surface. (6) If possible (enough remaining battery charge), scan of another burners row. (7) Coming back of the vehicle to the takeoff position and landing.

The takeoff maneuver has been carried out very carefully because of the unexpected dangerous presence of a banister. Anyway, the takeoff proceeded smoothly, thanks to the robustness of the localization algorithm.

After some seconds from the takeoff, necessary for the vehicle self-stabilization and dynamic parameters identification, the task advanced according to the task plan, so the vehicle has been controlled by the vehicle operator, using the OCU's keyboard, and brought near to the selected chamber's burners row. Once reached the planned position, the vehicle operator started to make the vehicle move as shown in Figure 10 following the scheduled sequence. It should be noticed that, during the flight, many random interruptions of the WiFi link have been encountered. However, these temporary lacks of communication did not compromise the overall good result of the task, but they diminished the total amount of time available for the actual inspection.

Conclusion

The manuscript introduces the problem of semi-automatic inspection and navigation in confined environments with a special focus on demanding environments like industrial combustion chambers. A system that overcomes many challenges imposed by such a problematic environment is presented. The presented system is composed of a multirotor flying robot and a control ground station and allows a human expert to easily inspect an industrial facility. The main problems for such inspection task have been introduced in detail at the beginning of the manuscript. The proposed solution system is described both for what concerns the hardware selection and the algorithmic and control strategies. Some of the presented software components have been previously introduced by the authors and tested in indoor office environments. The same algorithms displayed equivalent efficiency in the industrial environment presented here. Thanks to the choice of a laser sensor for the navigation and localization part and of a small computational footprint SLAM algorithm, the system resulted robust to communication losses, air flow disturbances and presence of undesired obstacles inside the combustion chambers. The presented system allows non-expert MAV pilots to navigate an aerial vehicle inside an unstructured confined space and visually inspect the structural condition of the environment. While this is not the intended usage, starting from the captured data, the system is also able to provide a 3D reconstruction of the inspected environment for offline analysis. To increase the flight time and lower the overall inspection time, strategies should be investigated in the future in order to reduce energy consumption by means of reducing the MAV payload or introducing techniques to recover energy from the environment.

Declaration of conflicting interests


The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

ORCID iD

Paolo Tripicchio  <http://orcid.org/0000-0003-3225-2782>

Massimo Satler  <http://orcid.org/0000-0001-6731-3114>

Carlo Alberto Avizzano  <http://orcid.org/0000-0001-5802-541X>

References

1. Longo D and Muscato G. The Alicia 3 climbing robot: a three-module robot for automatic wall inspection. *IEEE Robot Automat Mag* 2006; 13: 42–50.
2. Eich M and Vögele T. Design and control of a light-weight magnetic climbing robot for vessel inspection. In: *19th Mediterranean conference on IEEE control automation (MED)*, 20–23 June 2011, Corfù, Greece 2011.
3. Marconi L, Basile F, Caprari G, et al. Aerial service robotics: the AIrobots perspective. In: *2nd international conference on applied robotics for the power industry (CARPI)*, 11–13 Sept. 2012, Zurich, Switzerland 2012.
4. ARCAS - Aerial Robotics Cooperative Assembly System, FP7-ICT project 287617, web-site: <http://www.arcas-project.eu/>, 2011.
5. De Cubber G, Doroftei D, Serrano D, et al. Ourevitch. The EU-ICARUS project: developing assistive robotic tools for search and rescue operations. In: *IEEE international symposium on safety, security, and rescue robotics (SSRR)*, 21–26 Oct. 2013, Linköping, Sweden, 2013.
6. Albers A, Trautmann S, Howard T, et al. Semi-autonomous flying robot for physical interaction with environment. In: *IEEE conference on robotics automation and mechatronics (RAM)*, 28–30 June 2010, Singapore, 2010.
7. Michael N, Fink J and Kumar V. Cooperative manipulation and transportation with aerial robots. *Autonomous Robots* 2011; 30: 73–86.
8. Satler M, Unetti M, Giordani N, et al. Towards an autonomous flying robot for inspections in open and constrained spaces. In: *11th international multi-conference on systems, signals & devices (SSD)*, 11–14 Feb. 2014, Barcelona, Spain, 2014.
9. Pepe G, Satler M and Tripicchio P. Autonomous exploration of indoor environments with a micro-aerial vehicle. In: *Workshop on research, education and development of unmanned aerial systems (RED-UAS)*, 23–25 Nov. 2015, Cancun, Mexico, 2015.
10. Bartelds T, Capra A, Hamaza S, et al. Compliant aerial manipulators: toward a new generation of aerial robotic workers. *IEEE Robot Automat Lett* 2016; 1: 477–483.
11. Luque-Vega LF, Castillo-Toledo B, Loukianov A, et al. Power line inspection via an unmanned aerial system based on the quadrotor helicopter. In: *17th IEEE conference on Mediterranean electrotechnical (MELECON)*, 13–16 April 2014, Beirut, Lebanon, 2014.
12. Bonnin-Pascual F, Ortiz A, Garcia-Fidalgo E, et al. A micro-aerial platform for vessel visual inspection based on supervised autonomy. In: *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 28 Sept.–2 Oct. 2015, Hamburg, Germany, 2015.
13. Gohl P, Burri M, Omari S, et al. Towards autonomous mine inspection. In: *3rd international conference on applied robotics for the power industry (CARPI)*, 14–16 Oct. 2014, Foz do Iguassu, Brazil, 2014.
14. Nikolic J, Burri M, Rehder J, et al. A UAV system for inspection of industrial facilities. In: *IEEE aerospace conference*, 2–9 March 2013, Big Sky, MT, USA, 2013.
15. Tripicchio P, Satler M, Dabisias G, et al. Towards smart farming and sustainable agriculture with drones. In: *International conference on intelligent environments (IE)*, 15–17 July 2015, Prague, Czech Republic, 2015.
16. Sun J, Li B, Jiang Y, et al. A camera-based target detection and positioning UAV system for search and rescue (SAR) purposes. *Sensors* 2016; 16: 1778.
17. Sarris Z and Atlas S. Survey of UAV applications in civil markets. In: *Proceedings of the 9th Mediterranean conference on control and automation*, June 27–29, Dubrovnik, Croatia, 2001.
18. Durrant-Whyte H and Bailey T. Simultaneous localization and mapping: part I. *IEEE Robot Automat Mag* 2006; 13: 99–110.
19. Kümmerle R, Grisetti G, Strasdat H, et al. G²o: a general framework for graph optimization. In: *IEEE international conference on robotics and automation (ICRA)*, 9–13 May 2011, Shanghai, China, 2011.
20. Thrun S, Burgard W and Fox D. *Probabilistic robotics*. Cambridge MA: The MIT press, 2005.
21. Lim H, Lim J and Kim HJ. Real-time 6-DOF monocular visual SLAM in a large-scale environment. In: *IEEE international conference on robotics and automation (ICRA)*, 31 May–7 June 2014, Hong Kong, China 2014.
22. Tomono M. Robust 3D SLAM with a stereo camera based on an edge-point ICP algorithm. In: *IEEE international conference on robotics and automation*, 12–17 May 2009, Kobe, Japan 2009.
23. Blösch M, Weiss S, Scaramuzza D, et al. Vision based MAV navigation in unknown and unstructured environments. In: *IEEE international conference on robotics and automation (ICRA)*, 3–7 May 2010, Anchorage, AK, USA, 2010.
24. Engel J, Koltun V and Cremers D. Direct sparse odometry. *IEEE Transac Pattern Anal Mach Intell* 2017; pp.1–14, Issue: 99.
25. Pereira JL. *Hover and wind-tunnel testing of shrouded rotors for improved micro air vehicle design*. PhD dissertation, University of Maryland, College Park, MD, USA, 2008.
26. Schmid K, Tomic T, Ruess F, et al. Stereo vision based indoor/outdoor navigation for flying robots. In: *IEEE/RSJ international conference on intelligent robots and systems*, Tokyo, 3–7 Nov. 2013.
27. Tripicchio P, Unetti M, Giordani N, et al. A Lightweight slam algorithm for indoor autonomous navigation. In: *Proceedings of the Australasian conference on robotics and automation (ACRA 2014)*, 2–4 December, 2014, Melbourne, Australia.
28. Martinez V. Modelling of the flight dynamics of a quadrotor helicopter. MSc Thesis, Cranfield University, Cranfield, UK, Vol. 71, pp.149–438, 2007.
29. Box GE, Jenkins GM, Reinsel GC, et al. *Time series analysis: forecasting and control*. Chichester, UK: John Wiley & Sons, 2015.
30. Grisetti G, Tipaldi GD, Stachniss C, et al. Fast and accurate SLAM with Rao-Blackwellized particle filters. *Robot Autonom Syst* 2007; 55: 30–38.

31. Montemerlo M, Thrun S, Koller D, et. al. FastSLAM: a factored solution to the simultaneous localization and mapping problem. In: *Proceeding of the national conference on artificial intelligent (AAAI)*, Edmonton, Canada, July 28 – August 01 2002.
32. Benini A, Mancini A and Longhi S. An IMU/UWB/ Vision-based extended Kalman filter for mini-UAV localization in indoor environment using 802.15.4a wireless sensor network. *J Intell Robot Syst* 2013; Volume 70, issue 1-4 : 1–16.
33. Borges GA and Aldon MJ. A split-and-merge segmentation algorithm for line extraction in 2D range images. In: *Proceedings of 15th international conference on pattern recognition*, 3-7 Sept. 2000, Barcelona, Spain 2000.
34. Leonard J and Durrant-Whyte HF. Directed sonar sensing for mobile robot navigation. Vol. 175. Switzerland: Springer, Science & Business Media, vol. 175, 2012.
35. Grzonka S, Grisetti G and Burgard W. A fully autonomous indoor quadrotor. *IEEE Transac Robot* 2012; 28: 90–100.
36. Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. *Int J Robot Res* 1986; 5: 90–98.
37. Yamauchi B. A frontier-based approach for autoumous exploration. In: *IEEE international symposium on computational intelligence in robotics and automation*, Monterey, CA, 10-11 July 1997.
38. Hart PE, Nilsson NJ and Raphael B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transac Syst Sci Cybernet* 1968; 4: 100–107.
39. Kim A and Eustice R. Perception-driven navigation: active visual SLAM for robotic area coverage. In: *IEEE international conference on robotics and automation (ICRA)*, Karlsruhe, 6-10 May 2013.
40. Choset H and Pignon P. Coverage path planning: the Boustrophedon decomposition. In: *Proceedings of international conference on field and service robotics*, Canberra, Australia, 8-10 December, 1997.
41. Ntawumenyikizaba A, Viet HH and Chung T. An online complete coverage algorithm for cleaning robots based on boustrophedon motions and A* search. In: *8th international conference on information science and digital content technology (ICIDT)*, Jeju Island, South Korea, 26-28 June 2012.
42. Shen S, Michael N and Kumar V. Autonomous indoor 3D exploration with a micro-aerial vehicle. In: *IEEE international conference on robotics and automation (ICRA)*, Saint Paul, MN, 14-18 May 2012.
43. Silva EP, Engel PM, Trevisan M, et al. Exploration method using harmonic functions. *Robot Autonom Syst* 2002;40: 25–42.
44. Kim JO and Khosla PK. Real-time obstacle avoidance using harmonic potential functions. *IEEE Transac Robot Automat* 1992; Volume: 8, Issue: 3: 338–349.
45. Connolly CI and Grupen RA. The applications of harmonic functions to robotics. *J Robot Syst* 1993; 10: 931–946.
46. Borenstein J and Koren Y. Histogramic in-motion mapping for mobile robot avoidance. *IEEE J Robot Automat* 1991; 7: 535–539.
47. Shade R and Newman P. Choosing where to go: complete 3D exploration with stereo. In: *IEEE international conference on robotics and automation*, Shanghai, 9-13 May 2011.
48. Gupta RA ,Massoud AA and Chow MY. A delay-tolerant potential-field-based network implementation of an integrated navigation system. *IEEE Transac Ind Electron* 2010; 57: 769–783.
49. Bolles RC, Baker HH and Marimont DH. Epipolar-plane image analysis: an approach to determining structure from motion. *Int J Comput Vision* 1987; 1: 7–55.

Collaborative multiple micro air vehicles' localization and target tracking in GPS-denied environment from range–velocity measurements

Ioannis Sarras, Julien Marzat, Sylvain Bertrand and
Hélène Piet-Lahanier

International Journal of Micro Air
Vehicles
2018, Vol. 10(2) 225–239
© The Author(s) 2018
DOI: 10.1177/1756829317745317
journals.sagepub.com/home/mav


Abstract

We treat the problem of simultaneous collaborative multiple micro air vehicles' localization and target tracking using time-varying range and (relative and absolute) velocity measurements. The proposed solution combines robustly local nonlinear observers that estimate the relative positions between agents and their neighbors, and cooperative filters that fuse each agent's local estimates to globally localize them with respect to the target (and therefore to each other). These estimates are then introduced in a dynamic consensus-type control law that ensures the global collective target tracking while simultaneously estimating the target's velocity, without needing any external reference which makes it applicable in GPS-denied environments. Finally, a simulation scenario is studied in order to show the efficiency of the proposed solution.

Keywords

Collaborative tracking, noncooperative target, localization, multiple micro air vehicles, Global Positioning System-denied environment

Received 31 May 2017; accepted 12 September 2017

Introduction

Over the last decades we have witnessed the explosion of applications incorporating networks of robotic vehicles. Inspired by the behavior of animals in nature and motivated by the fact that a variety of objectives can be more efficiently, rapidly, and robustly accomplished collaboratively rather than independently, multiagent systems have been in the core of attention from both theoreticians and practitioners. Of particular interest have been applications involving multiple (aerial, ground, marine) vehicles that need to collaborate to achieve a common goal such as to ensure the exploration of unknown environments, to follow targets, to seek dangerous emitting sources, or to ensure high-precision photography.¹ Note that in order to attain the corresponding desired objective the location of the vehicles is an information of paramount importance. It is exploited in the guidance, control, and estimation algorithms that ensure the

successful undertaking of the mission scenario. However, such global information, as obtained for example by GPS receivers, is not available in indoor environments² and in general, due to hardware malfunction or unavailability of the minimum number of GPS satellites. Instead local, low-cost sensors (cameras, infrared sensors, sonars) are usually incorporated to provide a sufficient localization.

This work focuses on the design of a distributed control law which in an ideal (perfect measurements, relative positions available) scenario ensures that a number of multiple micro air vehicles (MAVs) track the unknown motion of a target. Our precise objective

ONERA—The French Aerospace Lab, Palaiseau, France

Corresponding author:

Ioannis Sarras, ONERA—The French Aerospace Lab, DTIS-SAGP,
Palaiseau F-91123, France.
Email: ioannis.sarras@onera.fr



Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (<http://www.creativecommons.org/licenses/by/4.0/>) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (<https://us.sagepub.com/en-us/nam/open-access-at-sage>).

is to propose a robust law based on (local) relative distance measurements and noisy velocity measurements to combine localization, target velocity estimation, and target tracking algorithms to obtain a globally exponential solution for the simultaneous collective agent localization and target tracking problem. The target can either be static, in which case this boils down to a consensus problem, or dynamic, yielding a setting similar to the classical leader–follower problem. In both cases, only a set of agents (at least one) usually has access to a relative information about the motion of the target while each agent has only access to local information. However, depending on the measurements available and the interagent communication characteristics, ensuring exact target tracking can be impossible and the mission objective is relaxed into following the target at a certain distance. Such a scenario arises exactly when relative distance measurements are available, instead of relative positions. These are possibly complemented by noisy relative or absolute velocity measurements of the agents. The reason why exact tracking cannot be achieved lies on the fact that distributed controllers frequently require relative position information. For that to be obtained the observability analysis reveals that a persistent relative motion between any pair of agents has to be present.^{3,4} The distributed control law has to account for this additional motion.

We propose a solution to the problem of collective target tracking for a target with unknown constant velocity (or unknown varying velocity but known acceleration) based on the agents' relative distance, relative velocity, and absolute velocity measurements. The communication topology between agents is considered as undirected and connected, while the communication topology between the target and the connected followers (at least one) is directed, which yields a strongly connected digraph. Our contribution consists of: (a) designing a localization algorithm that provides for every agent an estimation of the relative positions with respect to its neighbors and the target, (b) designing an estimator for each agent that provides an estimate of the target's velocity, (c) designing an estimator that filters the noisy relative velocity measurements and explicitly take it into account in the global stability analysis, and (d) designing a distributed control law that allows for the followers to collaboratively track the target and ensures a persistent relative motion.

Concerning the localization, in the literature we distinguish two large types of scenarios^{5,6}: (a) Mutual localization, referring to the scenario where each agent needs to find its own (static) position in a reference frame common to the entire network and (b) collaborative localization referring to the localization of a (dynamic) target using an already mutually localized

network. Our problem is of the second type. Depending on the community (control, robotics, sensors) and the mission objective, we can have 2D or 3D models, centralized or distributed algorithms, a variety of available measurements, for example absolute position (GPS), relative positions, distances, bearings or IMU measurements, and additional known points (anchors, markers). Additionally, the localization solutions can be signal/information based or model based which are essentially divided into optimization based and observer based. Observer-based, distributed estimation algorithms have recently been shown to present some rather interesting robust characteristics. In particular, it was established that observers which are distributed can enhance the quality of estimation by eliminating noise, see Tabarea et al.⁷ and Li and Sanfelice,⁸ which is of great interest in all applications.

Hence motivated by these recent developments and unlike the probabilistic and Kalman filter-based approaches,^{9–11} which cannot in general guarantee analytical global convergence, following Sarras et al.¹² we adopt an observer-based approach to treat the problem of multivehicle collaborative localization using time-varying range and relative velocity measurements without requiring any global positioning information. The range measurements can be obtained using a variety of sensors such as stereo-vision systems that typically equip robotic vehicles¹³ or by combining monocular cameras from different vehicles.¹⁴ This measurement scenario renders our obtained algorithm applicable to GPS-denied environments. We show that if each agent can obtain a good estimate of the target's velocity then it successfully localizes itself with respect to the target by the combination of local estimates of its neighbors' relative positions and the fusion with the neighbors' own estimates.

As opposed to other works, for example Bahr et al.,¹⁵ Dandach et al.,¹⁶ Deghat et al.¹⁷ our algorithm does not require global information (absolute position) but rather local measurements. Compared to the relevant work in Chai et al.⁶ that treats the collaborative localization problem with respect to a static target, instead of single integrators we consider double integrator dynamics to model the agents' translational dynamics and require no knowledge on the rate of change of the distances. Furthermore, we extend these results to the scenario of a dynamic target and show that by adopting an approach inspired by the recent developments on dynamically scaled Lyapunov functions^{18,19} we are able to show relative localization with a uniform global exponential convergence using a strict Lyapunov function.

Concerning the distributed control design, we follow the control paradigm laid in Hong et al.^{20,21} and followed by many others, for example Ren and Beard,¹

Cai and Huang,²² Liu and Huang,²³ and references therein. We show that in the case of known relative positions the interconnection of a consensus-type tracking law and a consensus-type target velocity estimator provides a globally exponential tracking solution. Compared to the landmark work,²¹ that more generally treats directed and switching graph topologies, we provide an alternative Lyapunov function that does not require knowledge of the global graph topology properties and thus, contributes in rendering it a truly distributed solution. While Ren and Beard,¹ Cai and Huang,²² Liu and Huang²³ consider also cases of nonlinear models for the agents' dynamics and more general graph topologies, they also assume that the relative positions are readily available and that all measurements are not corrupted by measurements. Works using mainly range measurements are focused on formation control, see for example Tron et al.,⁵ Montijano et al.,¹³ Oh and Ahn,²⁴ and Oh et al.²⁵

Although our results focus on target tracking, by straightforwardly modifying the control terms to incorporate desired distances between neighbor agents we can obtain a solution to target tracking with a prescribed formation of the followers and interagent collision avoidance, for the latter see the recent work of Franchi et al.²⁶ and references therein. These features are illustrated in the simulation scenario.

Model and problem formulation

Network topology

We consider that the interconnection graph describing the communication between the $N+1$ agents forming the multiagent system, target included, is formed by an undirected graph describing the network of the N followers and a directed graph connecting the target to some followers. The complete (directed, strongly connected) network topology can be modeled using the Laplacian matrix $\mathcal{L} := [l_{ij}] \in \mathbb{R}^{(N+1) \times (N+1)}$, $i, j \in \{0, \dots, N\}$, whose elements are defined as

$$l_{ij} = \begin{cases} \sum_{j \in \mathcal{N}_i} w_{ij} & i = j \\ -w_{ij} & i \neq j \end{cases} \quad (1)$$

where $w_{ij} = 0$ if $i = j$, $w_{ij} > 0$ if $j \in \mathcal{N}_i$ and $w_{ij} = 0$ otherwise. In this case, \mathcal{N}_i stands for the set of agents transmitting information to the i th agent. Note that, by construction, \mathcal{L} has zero row sum, i.e. $\mathcal{L}\mathbf{1}_{N+1} = \mathbf{0}$, where $\mathbf{1}_{N+1}$ is a column vector of size $N+1$ filled with ones or, equivalently, $l_{ii} - \sum_{j \in \mathcal{N}_i} l_{ij} = 0$. Following this definition, we note that the undirected

graph topology between the N followers with the $N \times N$ (symmetric) is described by the Laplacian matrix $\mathcal{L}_u := [l_{ij}]$, for $i, j \in \{1, \dots, N\}$. Further, define as \mathcal{B} an $N \times N$ diagonal matrix whose i th element is either $b_i > 0$ or 0 based on whether the i th agent receives information from the target, i.e. belongs to the set \mathcal{N}_0 . For more details on network topologies and their properties refer, for example to Ren and Beard.¹

Dynamic model

We consider that the dynamics of each of the $N+1$ identical agents composing the multivehicle system of interest can be described by the double integrator model

$$\dot{x}_i = v_i \quad (2)$$

$$\dot{v}_i = u_i, \quad i = \{0, \dots, N\} \quad (3)$$

with $x_i, v_i \in \mathbb{R}^3$ denoting the position and velocity vectors of the i th vehicle in the inertial frame, while $u_i \in \mathbb{R}^3$ is the applied acceleration.

By the index $i=0$ we denote the (static or dynamic) target with respect to which the localization will be referred. As is evident, the static scenario corresponds to a target's dynamics

$$\dot{x}_0 = 0 \quad (4)$$

$$\dot{v}_0 = 0 \quad (5)$$

In the dynamic case, we assume that the target's acceleration u_0 is known or zero, with the latter corresponding to a scenario of a noncooperative target moving in straight line at maximal velocity.

Now, we naturally define the relative position, velocity, and acceleration between two agents as

$$x_{ij} = x_i - x_j \quad (6)$$

$$v_{ij} = v_i - v_j \quad (7)$$

$$u_{ij} = u_i - u_j \quad (8)$$

that yield the required relative dynamics

$$\dot{x}_{ij} = v_{ij} \quad (9)$$

$$\dot{v}_{ij} = u_{ij} \quad (10)$$

For our localization problem, we consider that the available measurements consist of the relative velocities and distances^a

$$y_i = \text{col}(v_{ij}^T, d_{ij}^T) \quad (11)$$

with the distance d_{ij} between agent i and its neighbor j defined as

$$d_{ij} := |x_i - x_j| = |x_{ij}| \quad (12)$$

A simple derivation provides

$$\dot{d}_{ij} = \frac{x_{ij}^T v_{ij}}{d_{ij}} = \frac{v_{ij}^T x_{ij}}{d_{ij}} \quad (13)$$

In conclusion, the complete model on which our design will be based is summarized as

$$\dot{x}_{ij} = v_{ij} \quad (14)$$

$$\dot{v}_{ij} = u_{ij} \quad (15)$$

$$\dot{d}_{ij} = \frac{v_{ij}^T x_{ij}}{d_{ij}} \quad (16)$$

Cooperative localization

Before presenting our main results, we define some additional notation and then remind the definition of a persistently exciting (PE) function. The notation for a matrix A being positive (semi-)definite is expressed by $A \succ 0$ (≥ 0), while for the case of a positive scalar a we write instead $a > 0$. We note $\lambda_m(A)$ the minimal eigenvalue of a square matrix A . The notation $|\cdot|$ will refer, depending on its argument, either to the absolute value of a scalar function, to the Euclidean norm of a vector, or to the induced two-norm of a matrix.

Definition 1. Let the function $v_{ij} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^3$ be continuous. It is PE if there exist some $T > 0$ and $\mu > 0$ such that

$$\int_t^{t+T} v_{ij}(\tau) v_{ij}^T(\tau) d\tau \geq \mu I \succ 0, \quad \forall t \quad (17)$$

For the distance-based localization scenario we have at hand, we require that certain relative velocities are PE which means that in order for an agent to be able to reconstruct a relative position with respect to a neighboring agent, it is necessary to move out of the line of

sight (straight line connecting two agents) for some time which in fact is required for the relative position to be observable.^{3,4} In practice, this condition imposes a requirement on the applied accelerations (control inputs) which can always be ensured for each agent by including an excitation term but however, might complicate the stability analysis.

Single vehicle localization from direct local measurements: Static target

In this subsection we will consider the problem of localization of each agent with respect to its neighbors by incorporating local, noiseless measurements, and considering a static target. This will be achieved by means of a designed nonlinear observer based on the invariant-manifold observer methodology, see Astolfi et al.²⁷ and Karagiannis and Astolfi¹⁸ for the general setting and Martin and Sarras,²⁸ Martin and Sarras,²⁹ and Sarras et al.¹² for recent applications on MAVs.

Proposition 1. Consider the dynamical system defined in equations (14) to (16) and assume that v_{ij} is PE. Then, the dynamical system

$$\dot{\xi}_i := -\frac{K_{ij} d_{ij}^2}{2} u_{ij} - K_{ij} v_{ij} v_{ij}^T \hat{x}_{ij} + v_{ij} \quad (18)$$

$$\hat{x}_{ij} := \xi_i + \frac{d_{ij}^2}{2} K_{ij} v_{ij} \quad (19)$$

is a globally exponential observer with gain $K_{ij} > 0$.

Proof. First, let us define the relative position estimation error

$$z_{ij} := \xi_i + \beta_i(y_i, \hat{y}_i) - x_{ij} =: \hat{x}_{ij} - x_{ij} \quad (20)$$

for a certain mapping β_i , that generally can also depend on a filtered y_i denoted \hat{y}_i , that will be properly selected. At this point we examine the case where β_i is a function only of y_i . Then, the general form of the z_i dynamics gives

$$\begin{aligned} \dot{z}_{ij} &:= \dot{\xi}_i + \partial_{y_i} \beta_i \dot{y}_i - \dot{x}_{ij} = \dot{\xi}_i + \partial_{d_{ij}} \beta_i \dot{d}_{ij} + \partial_{v_{ij}} \beta_i \dot{v}_{ij} - \dot{x}_{ij} \\ &= \dot{\xi}_i + \partial_{d_{ij}} \beta_i \frac{v_{ij}^T x_{ij}}{d_{ij}} + \partial_{v_{ij}} \beta_i u_{ij} - v_{ij} \end{aligned}$$

With the choice

$$\dot{\xi}_i := -\partial_{d_{ij}} \beta_i \frac{v_{ij}^T \hat{x}_{ij}}{d_{ij}} - \partial_{v_{ij}} \beta_i u_{ij} + v_{ij}$$

and the β_i mapping as

$$\beta_i(y_i) := \frac{d_{ij}^2}{2} K_{ij} v_{ij} \quad (21)$$

the z_i dynamics obtains the more explicit form

$$\dot{z}_{ij} = -K_{ij} v_{ij} v_{ij}^T z_{ij} \quad (22)$$

From Lemma 5 of Loria and Panteley³⁰ we know that the nominal system (22) has a uniformly global exponentially stable (UGES) equilibrium at the origin for a PE and uniformly bounded v_{ij} .

Remark 1. From the converse Lyapunov lemma (Lemma 1 of Loria and Panteley³⁰) we know that there exists a quadratic Lyapunov function

$$V_{z_i} := \frac{1}{2} z_{ij}^T P(t) z_{ij} \quad (23)$$

with $P(t)$ such that $0 \prec c_1 I \preceq P(t) = P^T(t) \preceq c_2 I$, the unique solution of the equation

$$\dot{P} - PK_{ij} v_{ij} v_{ij}^T - v_{ij} v_{ij}^T K_{ij} P = -Q \quad (24)$$

with $Q(t) = Q^T(t)$ such that $0 \prec c_3 I \preceq Q(t) \preceq c_4 I$.

This lemma will be exploited in the construction of a strict, dynamically scaled Lyapunov function of the more general solution that follows in the next subsection.

Remark 2. Notice that in our algorithm, we further require that the relative acceleration between neighboring agents be either available or can be reconstructed. As is common in the literature for example, the agents might transmit their respective control actions (accelerations or resulting predicted positions) to their neighbors.³¹ If these signals are imperfectly known, due for example to transmission perturbations, we can explicitly provide a robustness analysis by treating the imperfections as additive disturbances and using our Lyapunov function in an input-to-state stable (ISS) analysis. Alternatively, and under the assumption that relative motion varies slowly, we can consider that the relative acceleration is reconstructed by numerical differentiation of the available relative velocities.

Remark 3. Let us mention that in the case where relative orientations (rotation matrices) are available, by means of bearing measurements, and assuming that each agent is equipped with a gyro, we can adapt the obtained algorithms to such scenario. In such case the agents do not need to be already mutually localized and the transmitted signals are communicated in the proper local frame of each agent.

Single vehicle localization from filtered local measurements: Dynamic target

In continuation of the previous scenario, we proceed to extend the localization algorithm to the case of a dynamic target in the presence of noisy velocity measurements, without assuming any particular noise characteristics.

Proposition 2. Consider the dynamical system defined in equations (14) to (16) and assume that v_{ij} is PE. Then, the dynamical system

$$\begin{aligned} \dot{\xi}_i := & -\frac{K_{ij} d_{ij}^2}{2} (u_{ij} - K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r)(\hat{v}_{ij} - v_{ij})) \\ & - K_{ij} \hat{v}_{ij} \hat{v}_{ij}^T \hat{x}_{ij} + \hat{v}_{ij} \end{aligned} \quad (25)$$

$$\hat{x}_{ij} := \xi_i + \frac{d_{ij}^2}{2} K_{ij} \hat{v}_{ij} \quad (26)$$

$$\dot{r} := -c_7(r-1) + \frac{c_2^2 K_{ij}^2}{c_1 c_5} |v_{ij}|^2 |\hat{v}_{ij} - v_{ij}|^2 \quad (27)$$

$$\hat{v}_{ij} := u_{ij} - K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r)(\hat{v}_{ij} - v_{ij}) \quad (28)$$

is a globally exponential observer, for some $c_i > 0$, with $r(0) \geq 1$ and gains

$$K_{ij} := c_8 + \frac{c_5 + c_6 + c_7 c_2}{c_3}$$

$$\begin{aligned} K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r) := & c_9 I + (r-1) \frac{c_2^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 I \\ & + \frac{c_2^2}{c_6 r} (K_{ij}^2 |\hat{v}_{ij}|^2 |\hat{x}_{ij}|^2 + 1) I \end{aligned}$$

Proof. First, let us define the relative position estimation error as in equation (20). Then, the general form of the z_i dynamics gives

$$\begin{aligned} \dot{z}_{ij} := & \dot{\xi}_i + \partial_{y_i} \beta_i \dot{\hat{y}}_i + \partial_{y_i} \beta_i \dot{y}_i - \dot{x}_{ij} \\ = & \dot{\xi}_i + \partial_{d_{ij}} \beta_i \dot{d}_{ij} + \partial_{v_{ij}} \beta_i \dot{\hat{v}}_{ij} + \partial_{d_{ij}} \beta_i \dot{d}_{ij} + \partial_{v_{ij}} \beta_i \dot{v}_{ij} - \dot{x}_{ij} \\ = & \dot{\xi}_i + \partial_{d_{ij}} \beta_i \dot{d}_{ij} + \partial_{v_{ij}} \beta_i \dot{\hat{v}}_{ij} + \partial_{d_{ij}} \beta_i \frac{v_{ij}^T x_{ij}}{d_{ij}} + \partial_{v_{ij}} \beta_i u_{ij} - v_{ij} \end{aligned}$$

which with the choice

$$\dot{\xi}_i := -\partial_{d_{ij}} \beta_i \dot{d}_{ij} - \partial_{v_{ij}} \beta_i \dot{\hat{v}}_{ij} - \partial_{d_{ij}} \beta_i \frac{\hat{v}_{ij}^T \hat{x}_{ij}}{d_{ij}} - \partial_{v_{ij}} \beta_i u_{ij} + \hat{v}_{ij}$$

reduces, after defining $e_{v_{ij}} := \hat{v}_{ij} - v_{ij}$, to

$$\begin{aligned}\dot{z}_{ij} &= -\partial_{d_{ij}}\beta_i \left(\frac{\hat{v}_{ij}^T \hat{x}_{ij}}{d_{ij}} - \frac{v_{ij}^T x_{ij}}{d_{ij}} \right) + \hat{v}_{ij} - v_{ij} \\ &= -\partial_{d_{ij}}\beta_i \frac{v_{ij}^T}{d_{ij}} z_{ij} - \partial_{d_{ij}}\beta_i \frac{\hat{x}_{ij}^T}{d_{ij}} e_{v_{ij}} + e_{v_{ij}}\end{aligned}$$

Selecting further the β_i mapping as

$$\begin{aligned}\beta_i(v_i, \hat{v}_i) &:= \frac{d_{ij}^2}{2} K_{ij} \hat{v}_{ij} = \frac{d_{ij}^2}{2} K_{ij} (v_{ij} + e_{v_{ij}}) \\ \partial_{d_{ij}}\beta_i &= d_{ij} K_{ij} (v_{ij} + e_{v_{ij}})\end{aligned}$$

the z_i dynamics obtains the more explicit form

$$\begin{aligned}\dot{z}_{ij} &= -K_{ij} v_{ij} v_{ij}^T z_{ij} - K_{ij} e_{v_{ij}} v_{ij}^T z_{ij} - K_{ij} (v_{ij} + e_{v_{ij}}) \hat{x}_{ij}^T e_{v_{ij}} + e_{v_{ij}} \\ &= -K_{ij} v_{ij} v_{ij}^T z_{ij} - K_{ij} e_{v_{ij}} v_{ij}^T z_{ij} - (K_{ij} \hat{v}_{ij} \hat{x}_{ij}^T - I) e_{v_{ij}}\end{aligned}$$

Taking the function V_{z_i} defined in equation (23) and computing its time derivative along trajectories of the z_i dynamics yields

$$\begin{aligned}\dot{V}_{z_i} &:= \frac{1}{2} z_{ij}^T \left(\dot{P}(t) - P(t) K_{ij} v_{ij} v_{ij}^T - v_{ij} v_{ij}^T K_{ij} P(t) \right) z_{ij} \\ &\quad - z_{ij}^T P(t) K_{ij} e_{v_{ij}} v_{ij}^T z_{ij} - z_{ij}^T P(t) (K_{ij} \hat{v}_{ij} \hat{x}_{ij}^T - I) e_{v_{ij}} \\ &\leq -\frac{c_3}{2} |z_{ij}|^2 + c_2 |z_{ij}|^2 K_{ij} |e_{v_{ij}}| |v_{ij}| \\ &\quad + c_2 |z_{ij}| (K_{ij} |\hat{v}_{ij}| |\hat{x}_{ij}| + 1) |e_{v_{ij}}| \\ &\leq -\left(\frac{c_3}{2} - \frac{c_5 + c_6}{2} \right) |z_{ij}|^2 + \frac{c_2^2}{2c_5} K_{ij}^2 |v_{ij}|^2 |e_{v_{ij}}|^2 |z_{ij}|^2 \\ &\quad + \frac{c_2^2}{c_6} (K_{ij}^2 |\hat{v}_{ij}|^2 |\hat{x}_{ij}|^2 + 1) |e_{v_{ij}}|^2\end{aligned}$$

where we applied Young's inequality to the two cross-terms of the first inequality. In order to handle the last two cross-terms in the above right-hand side we employ a dynamic scaling of the form

$$W_{z_i} := \frac{V_{z_i}}{r} \quad (29)$$

with r dynamics given, with $r(0) \geq 1$, as

$$\dot{r} := -c_7(r-1) + \frac{c_2^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 |e_{v_{ij}}|^2$$

Then, the time derivative of W_{z_i} can be shown to be

$$\begin{aligned}\dot{W}_{z_i} &= \frac{\dot{V}_{z_i}}{r} - W_{z_i} \frac{\dot{r}}{r} \\ &\leq \frac{\dot{V}_{z_i}}{r} + c_2 |z_{ij}|^2 c_7 \frac{(r-1)}{r} \\ &\quad - c_1 \frac{|z_{ij}|^2}{r} \frac{c_2^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 |e_{v_{ij}}|^2 \\ &\leq -\left(\frac{c_3}{2} - \frac{c_5 + c_6 + c_7 c_2}{2} \right) \frac{|z_{ij}|^2}{r} \\ &\quad + \frac{c_2^2}{c_6} (K_{ij}^2 |\hat{v}_{ij}|^2 |\hat{x}_{ij}|^2 + 1) \frac{|e_{v_{ij}}|^2}{r}\end{aligned}$$

with the last right-hand side term depending on the error between the filtered \hat{v}_{ij} and the true measurements v_{ij} .

Choosing

$$\hat{v}_{ij} := u_{ij} - K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r) e_{v_{ij}} \quad (30)$$

with K_{v_i} a (free) positive gain function of $\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r$, yields the dynamics of the filtering error $e_{v_{ij}} := \hat{v}_{ij} - v_{ij}$

$$\dot{e}_{v_{ij}} := -K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r) e_{v_{ij}} \quad (31)$$

By simple derivations one can show that the following function

$$V_{e_v} := \frac{1}{2} |e_{v_{ij}}|^2 \quad (32)$$

is a Lyapunov function for the $e_{v_{ij}}$ dynamics since it satisfies

$$\dot{V}_{e_v} = -e_{v_{ij}}^T K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r) e_{v_{ij}}$$

and hence, ensuring global exponential convergence of the estimate \hat{v}_{ij} to v_{ij} . Similarly, for the r dynamics we take the function

$$V_r := \frac{1}{2} (r-1)^2 \quad (33)$$

that gives

$$\dot{V}_r = -c_7(r-1)^2 + (r-1) \frac{c_2^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 |e_{v_{ij}}|^2$$

Selecting then the functions

$$K_{ij} := c_8 I, \quad c_8 > c_3 - c_5 + c_6 + c_7 c_2$$

and

$$K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r) := c_9 I + (r-1) \frac{c_2^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 I \\ + \frac{c_2^2}{c_6 r} (K_{ij}^2 |\hat{v}_{ij}|^2 |\hat{x}_{ij}|^2 + 1) I$$

with $c_9 > 0$, we can finally establish that the composite function $W_{z_i} + V_{e_v} + V_r$ serves as a Lyapunov function for the complete dynamics with

$$\overbrace{W_{z_i} + V_{e_v} + V_r} \leq -c_8 \frac{|z_{ij}|^2}{r} - c_7 (r-1)^2 - c_9 |e_{v_{ij}}|^2$$

which establishes UGES of the origin.

Remark 4. Notice that in the case where the mapping β_i is simply defined as

$$\beta_i(y_i) := \frac{d_{ij}^2}{2} K_{ij} v_{ij}$$

then the resulting error dynamics is described as

$$\dot{z}_{ij} = -K_{ij} v_{ij} v_{ij}^T z_{ij} - e_{v_{ij}}$$

Then, using the PE condition, UGES of the nominal z_i system with respect to the origin, and UGES of the origin for the $e_{v_{ij}}$ system we can immediately conclude, for example from cascaded systems³² or ISS arguments,⁶ UGES of the interconnected system.

Collaborative localization from fusion of local estimates and measurements

In this subsection, we take advantage of the collaborative setting between the agents, which share information with their local neighbors, in order to enhance the localization capabilities of the agents, in particular, that do not have direct relative measurements with respect to the target. Of course for a static target the measurement v_i of each agent suffices.

To this end, define the fused estimate of the relative coordinates between agent j and the target as

$$\hat{x}_{i0}^j := \rho_j - \hat{x}_{ij} \quad (34)$$

$$\rho_0 := 0 \quad (35)$$

Then, the proposed consensus-based estimation mechanism for agent i , that exploits the fusion of its

own estimate with the ones of its neighbors to produce a more accurate fused estimate, is given by

$$\dot{\rho}_i := v_i - \hat{v}_0^i + \lambda_0 \sum_{j \in \mathcal{N}_i} (\hat{x}_{i0}^j - \rho_i), \quad \lambda_0 > 0 \quad (36)$$

with \hat{v}_0^i an estimation of the target's velocity v_0 by the i th agent to be defined in the following section.

We now state the following result.

Proposition 3. Consider the dynamical system defined in equations (14) to (16) and assume that v_{ij} is PE. Then, the dynamical system

$$\dot{\xi}_i := -\frac{K_{ij} d_{ij}^2}{2} (u_{ij} - K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r)(\hat{v}_{ij} - v_{ij})) \\ - K_{ij} \hat{v}_{ij} \hat{v}_{ij}^T \hat{x}_{ij} + \hat{v}_{ij} \quad (37)$$

$$\hat{x}_{ij} := \xi_i + \frac{d_{ij}^2}{2} K_{ij} \hat{v}_{ij} \quad (38)$$

$$\dot{r} := -c_7 (r-1) + \frac{c_2^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 |\hat{v}_{ij} - v_{ij}|^2 \quad (39)$$

$$\dot{\hat{v}}_{ij} := u_{ij} - K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r)(\hat{v}_{ij} - v_{ij}) \quad (40)$$

$$\dot{\rho}_i := v_i - \hat{v}_0^i + \lambda_0 \sum_{j \in \mathcal{N}_i} (\hat{x}_{i0}^j - \rho_i) \quad (41)$$

with $r(0) \geq 1$, ensures that when \hat{v}_0^i is such that $v_0 - \hat{v}_0^i \rightarrow 0$ every agent is globally exponentially localized with respect to the target, for some $c_i > 0$ and with gains

$$K_{ij} := c_8 + \frac{c_5 + c_6 + c_7 c_2}{c_3}$$

$$K_{v_i}(\hat{x}_{ij}, \hat{v}_{ij}, v_{ij}, r) := c_9 I + (r-1) \frac{c_2^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 I \\ + \frac{c_2^2}{c_6 r} (K_{ij}^2 |\hat{v}_{ij}|^2 |\hat{x}_{ij}|^2 + 1) I$$

Furthermore, when $v_0 - \hat{v}_0^i$ is bounded the localization error is bounded.

Proof. For $i = 1, \dots, N$, we define

$$\sigma_i := \rho_i - x_{i0} \quad (42)$$

$$\sigma_0 := 0, \quad \dot{\sigma}_0 = 0 \quad (43)$$

Then we obtain the consensus system

$$\begin{aligned}\dot{\sigma}_i &= -\lambda_0 \sum_{j \in \mathcal{N}_i} (\sigma_i - \sigma_j) + \lambda_0 \sum_{j \in \mathcal{N}_i} (\hat{x}_{ij} - x_{ij}) + v_0 - \hat{v}_0^i \\ &= -\lambda_0 \sum_{j \in \mathcal{N}_i} (\sigma_i - \sigma_j) + \lambda_0 \sum_{j \in \mathcal{N}_i} z_{ij} + v_0 - \hat{v}_0^i\end{aligned}$$

with σ_i seen as the individual states of the N agents and σ_0 the state of a leader, while the last two terms are seen as external signals. Defining the stacked variables

$$\sigma := \text{col}(\sigma_0, \dots, \sigma_N)$$

$$\tau_i := \sum_{j \in \mathcal{N}_i} z_{ij}$$

$$\tau := \text{col}(\tau_0, \dots, \tau_N)$$

$$\psi := \text{col}(v_0, \dots, v_N) - \text{col}(\hat{v}_0^0, \dots, \hat{v}_0^N), \quad \hat{v}_0^0 := 0$$

we obtain the dynamics

$$\dot{\sigma} := -\lambda_0(\mathcal{L} \otimes I_3)\sigma + \lambda_0\tau + \psi \quad (44)$$

As is well known, from the properties of the assumed underlying graph topology, we have that the nominal system $\dot{\sigma} = -\lambda_0(\mathcal{L} \otimes I_3)\sigma$ has a UGES equilibrium at the origin. Furthermore, we know there exists a quadratic Lyapunov function, defined here as

$$V_\sigma := \sigma^T(\Xi \otimes I_3)\sigma, \quad \Xi = \Xi^T \succ 0 \quad (45)$$

that establishes the claim of the nominal system. The first part of the proof is concluded by standard arguments on cascaded systems (see e.g. Lemma 2.1 or Proposition 2.3 of Loria and Panteley³²) since the complete error system consists of two nominal UGES subsystems interconnected through the terms τ , ψ that satisfy a linear growth condition. The second part of the proof follows immediately from the observation that the σ system is ISS with $v_0 - \hat{v}_0^i$ as an input.

Remark 5. Although not presented here, notice that our results are also applicable for switched communication graphs (due e.g. to loss of communication link or measurements) under the additional assumption of uniform connectivity as is done, for example for the single-landmark multiagent localization in the recent work of Chai et al.⁶ We stress again that in our setting, however, the derivative of the relative distances is not required and furthermore, measurement noise is explicitly treated by means of additional filters.

Remark 6. Notice that in an all-to-all communication scenario it is not mandatory to have different scaling dynamics \dot{r} for every pair of neighboring agents. A single one is sufficient by modifying the right-hand side of equation (39) to include the sum of all terms $\frac{c_{ij}^2}{c_1 c_5} K_{ij}^2 |v_{ij}|^2 |\hat{v}_{ij} - v_{ij}|^2$ for all neighbors i and j .

Collaborative tracking control with unknown target velocity

We now examine a distributed control law that ensures the tracking of a target with unknown velocity. Furthermore, the target's velocity needs to be estimated by means of a (globally) converging observer. Finally, we will modify the proposed control law by adding an additional term in order to impose a motion to each agent such that the persistence-of-excitation condition (17), required by the localization algorithm, is readily satisfied.

For ease of reference we remind the main working assumptions:

1. The acceleration u_0 of the target is known or zero.
2. The interagent communication is defined by a static undirected, connected graph topology modeled by its Laplacian $\mathcal{L}_u := [l_{ij}]$ as in (1) but for $i, j \in \{1, \dots, N\}$.
3. The topology between target and agents (at least one) is described by a directed path (at least one) and as such the complete topology is strongly connected.

Case: Known relative positions

In this subsection we consider first the ideal scenario where the relative positions x_{ij} are measured. This is summarized in the following assumption.

Assumption 1. The relative positions x_{ij} between agents are available and at least one agent has access to its relative position with respect to the target.

This assumption will naturally be removed when we consider the interconnection between the localization algorithm and the control law.

Let us first define by \hat{v}_0^i the estimate of the target's velocity v_0 by agent i . Now, similarly to Hong et al.,²¹ we define the control law as

$$u_i = u_0 - k_v(v_i - \hat{v}_0^i) - k_x \left(\sum_{j \in \mathcal{N}_i} l_{ij} x_{ij} + \sum_{k \in \mathcal{N}_0} b_k x_{k0} \right) \quad (46)$$

with positive constants k_x, k_v , while the observer that provides the target's velocity v_0 for agent i is selected as

$$\dot{\hat{v}}_0^i := u_0 - \frac{k_v}{k_x} \left(\sum_{j \in \mathcal{N}_i} l_{ij} x_{ij} + \sum_{k \in \mathcal{N}_0} b_k x_{k0} \right) \quad (47)$$

Furthermore, define the error variables^b

$$\mathcal{X} := \text{col}(x_1, \dots, x_N) - \mathbf{1}_N \otimes x_0 \quad (48)$$

$$\mathcal{V} := \text{col}(v_1, \dots, v_N) - \mathbf{1}_N \otimes v_0 \quad (49)$$

$$\mathcal{S} := -\text{col}(\hat{v}_0^1, \dots, \hat{v}_0^N) + \mathbf{1}_N \otimes v_0 \quad (50)$$

The error dynamics can then be shown to take the following form

$$\dot{\mathcal{S}} = \frac{k_x}{k_v} ((\mathcal{L}_u + \mathcal{B}) \otimes I_3) \mathcal{X} \quad (51)$$

$$\dot{\mathcal{X}} = \mathcal{V} \quad (52)$$

$$\dot{\mathcal{V}} = -k_x ((\mathcal{L}_u + \mathcal{B}) \otimes I_3) \mathcal{X} - k_v \mathcal{V} - k_v \mathcal{S} \quad (53)$$

Proposition 4. *Consider the error dynamics given in equations (51) to (53). Then, the origin is uniformly globally exponentially stable for $k_x, k_v > 0$.*

Proof. In order to establish the convergence properties we consider the following Lyapunov function, that is composed of four parts

$$V_1 := \frac{1}{2} (\mathcal{S}^T P_1 \mathcal{S} + \mathcal{X}^T P_2 \mathcal{X} + \mathcal{V}^T P_3 \mathcal{V}) \quad (54)$$

$$V_2 := \epsilon_2 \mathcal{X}^T \mathcal{V} \quad (55)$$

$$V_3 := \epsilon_3 \mathcal{S}^T \mathcal{V} \quad (56)$$

$$V_4 := \epsilon_4 \mathcal{X}^T \mathcal{S} \quad (57)$$

$$V := V_1 + V_2 + V_3 + V_4 \quad (58)$$

Its time derivative along trajectories of the error dynamics gives

$$\begin{aligned} \dot{V} := & -k_v \epsilon_3 |\mathcal{S}|^2 - \frac{k_x}{k_v} (\epsilon_2 k_v - \epsilon_4) \mathcal{X}^T ((\mathcal{L}_u + \mathcal{B}) \otimes I_3) \mathcal{X} \\ & - \mathcal{V}^T (P_3 - \epsilon_2 I) \mathcal{V} \end{aligned}$$

$$\begin{aligned} & + \mathcal{S}^T \left(k_x \left(\frac{1}{k_v} P_1 - \epsilon_3 I \right) ((\mathcal{L}_u + \mathcal{B}) \otimes I_3) - \epsilon_2 k_v I \right) \mathcal{X} \\ & - \mathcal{V}^T (P_3 + \epsilon_3 k_v I - \epsilon_4 I) \mathcal{S} \\ & + \mathcal{V}^T (P_2 - \epsilon_2 k_v I - k_x \left(P_3 - \frac{\epsilon_3}{k_v} I \right) ((\mathcal{L}_u + \mathcal{B}) \otimes I_3)) \mathcal{X} \end{aligned}$$

Then with the selection

$$P_1 := k_v \epsilon_3 I + \frac{k_v \epsilon_2}{k_x} ((\mathcal{L}_u + \mathcal{B})^{-1} \otimes I_3) \quad (59)$$

$$P_2 := k_v \epsilon_2 I + k_x \left(\epsilon_1 - \frac{\epsilon_3}{k_v} I \right) (\epsilon_0 \otimes I_3) \quad (60)$$

$$P_3 := \epsilon_1 I \quad (61)$$

$$\epsilon_0 := -(\mathcal{L}_u + \mathcal{B}) \otimes I_3 \quad (62)$$

all cross-terms disappear apart from the one in \mathcal{S}, \mathcal{V} . In order to establish our claim we need to ensure that the Lyapunov function V in equation (58) is positive definite with respect to the state $(\mathcal{S}, \mathcal{X}, \mathcal{V})$ and that the negative terms in \dot{V} dominate the remaining cross-term. The former is ensured if the matrix

$$\begin{bmatrix} P_1 & \epsilon_4 I & \epsilon_3 I \\ \epsilon_4 I & P_2 & \epsilon_2 I \\ \epsilon_3 I & \epsilon_2 I & P_3 \end{bmatrix} \succ 0 \quad (63)$$

while the latter if the matrix

$$\begin{bmatrix} \epsilon_1 - \epsilon_2 & \frac{1}{2} (k_v (\epsilon_1 + \epsilon_3) - \epsilon_4) \\ \frac{1}{2} (k_v (\epsilon_1 + \epsilon_3) - \epsilon_4) & \epsilon_3 k_v \end{bmatrix} \succeq \epsilon_5 I \quad (64)$$

for some (gain adjustable) $\epsilon_5 > 0$. By applying Schur's complement to the above matrices we obtain the sufficient conditions

$$\epsilon_1 > \epsilon_2 \quad (65)$$

$$\epsilon_2 \ll 1, \quad \epsilon_3 \ll 1, \quad \epsilon_4 \ll 1 \quad (66)$$

$$k_v > \frac{\min(\epsilon_3, \epsilon_4)}{\epsilon_1} \quad (67)$$

We finally obtain

$$\begin{aligned} \dot{V} &\leq -\epsilon_5(|\mathcal{S}|^2 + |\mathcal{V}|^2) - \frac{k_x}{k_v}(\epsilon_2 k_v - \epsilon_4) \\ &\times \lambda_m(\mathcal{L}_u + \mathcal{B})|\mathcal{X}|^2 < 0, \quad \forall (\mathcal{S}, \mathcal{X}, \mathcal{V}) \neq (0, 0, 0) \end{aligned}$$

which concludes the proof of global exponential stability of the origin. \square

Remark 7. Let us stress the fact that the proposed strict Lyapunov function is derived with gain conditions independent of the network topology characteristics, apart of course from the fact that the multiagent system is connected. Notice instead that in Chai et al.⁶ a strict Lyapunov was obtained under conditions on the gains k_x, k_v (denoted k, l in that reference) that depend on the minimum and maximum eigenvalues of the matrix $\mathcal{L}_u + \mathcal{B}$, which signifies that knowledge of the entire network topology is a priori required.

Case: Estimated relative positions

We now couple the proposed control law with the observer for relative positions. The control law (46) and the target velocity estimator become

$$u_i = u_0 - k_v(v_i - \hat{v}_0^i) - k_x \left(\sum_{j \in \mathcal{N}_i} l_{ij} \hat{x}_{ij} + \sum_{k \in \mathcal{N}_0} b_k \rho_k \right) \quad (68)$$

$$\dot{\hat{v}}_0^i := u_0 - \frac{k_v}{k_x} \left(\sum_{j \in \mathcal{N}_i} l_{ij} \hat{x}_{ij} + \sum_{k \in \mathcal{N}_0} b_k \rho_k \right) \quad (69)$$

To this end, and in order to simplify notation, we define the column stack of the relative position estimates as

$$\mathcal{Z} := \text{col}[z_{ij}] \quad (70)$$

Then, we can write compactly the complete closed-loop system as

$$\begin{aligned} \dot{\sigma} &:= -\lambda_0(\mathcal{L} \otimes I_3)\sigma + \lambda_0\tau + \mathcal{S} \\ \dot{\mathcal{S}} &= \frac{k_x}{k_v}((\mathcal{L}_u + \mathcal{B}) \otimes I_3)\mathcal{X} + \frac{k_x}{k_v}B\sigma + \frac{k_x}{k_v}A\mathcal{Z} \\ \dot{\mathcal{X}} &= \mathcal{V} \\ \dot{\mathcal{V}} &= -k_x((\mathcal{L}_u + \mathcal{B}) \otimes I_3)\mathcal{X} - k_v\mathcal{V} - k_v\mathcal{S} - k_xB\sigma \\ &\quad - k_xA\mathcal{Z} \end{aligned}$$

with A, B constant matrices of appropriate dimensions while the \mathcal{Z} terms are seen as exponentially decaying

perturbations. The global exponential stability of the composite system is concluded either by a spectral analysis or by a straightforward direct Lyapunov analysis based on the sum of the Lyapunov functions for each subsystem (local estimator for \hat{x}_{ij} , fusion for \hat{x}_{i0} , target velocity v_0 estimator and controlled system).

Imposing the PE condition through the control

Now we consider an additive term to our control law that should be defined in a way to enforce the PE condition, i.e. produce a sufficiently rich motion for every pair of neighboring agents, but conserve the network's stability properties.

The modified control law for each agent now takes the form

$$u_i = u_{PE}^i + u_0 - k_v(v_i - \hat{v}_0^i) - k_x \left(\sum_{j \in \mathcal{N}_i} l_{ij} \hat{x}_{ij} + \sum_{k \in \mathcal{N}_0} b_k \rho_k \right) \quad (71)$$

$$\dot{\hat{v}}_0^i := u_0 - \frac{k_v}{k_x} \left(\sum_{j \in \mathcal{N}_i} l_{ij} \hat{x}_{ij} + \sum_{k \in \mathcal{N}_0} b_k \rho_k \right) \quad (72)$$

For the stability analysis we define the stack of all

$$u_{PE} := \text{col}([u_{PE}^j]) := \text{col}([u_{PE}^i - u_{PE}^j]), \quad \forall j \in \mathcal{N}_i$$

Then we write the complete closed-loop system as

$$\begin{aligned} \dot{\sigma} &:= -\lambda_0(\mathcal{L} \otimes I_3)\sigma + \lambda_0\tau + \mathcal{S} \\ \dot{\mathcal{S}} &= \frac{k_x}{k_v}((\mathcal{L}_u + \mathcal{B}) \otimes I_3)\mathcal{X} + \frac{k_x}{k_v}B\sigma + \frac{k_x}{k_v}A\mathcal{Z} \\ \dot{\mathcal{X}} &= \mathcal{V} \\ \dot{\mathcal{V}} &= -k_x((\mathcal{L}_u + \mathcal{B}) \otimes I_3)\mathcal{X} - k_v\mathcal{V} - k_v\mathcal{S} - k_xB\sigma \\ &\quad - k_xA\mathcal{Z} + u_{PE} \end{aligned}$$

Based on the analysis of the previous subsection and by treating the input u_{PE} as a disturbance, we can show that our closed-loop system is ISS with respect to u_{PE} from which we can conclude practically global exponential stability since convergence is ensured to a neighborhood of the desired equilibrium trajectory that can be made (by assignment of the free function u_{PE}) very small but not identically zero.

Remark 8. We remind that the main results in Chai et al.⁶ and Hong et al.²¹ on which we are based hold also for switched graphs, under of course a condition of uniform connectivity, and as such our algorithm is also applicable to the case of switching communication graphs.

Simulation results

In this section we study the efficiency of the obtained algorithms by means of numerical simulations that serve as proof of concept. We will consider a two-dimensional scenario with three agents pursuing a target with constant linear motion. This motion is selected so that the target does not contribute to the satisfaction of the PE condition but rather is a task to be ensured by the agents. Additionally, we consider that the target is (and stays) in the field of view of only the first agent which is thus the only agent having available information about the target. We consider that all (relative, absolute) velocity measurements and accelerations are corrupted by Gaussian white noise.

The initial positions (in m) and velocities (in m/s) of the agents are, respectively, given as $x_0(0) = [20, 0]^T$, $x_1(0) = [2, 0]^T$, $x_2(0) = [10, -5]^T$, $x_3(0) = [3\sin(\pi/8), 5\cos(\pi/8)]^T$, $v_0(0) = [3]^T$, $v_1(0) = [0, 2]^T$, $v_2(0) = [1, 1]^T$, $v_3(0) = [2\cos(\pi/8), -2\sin(\pi/8)]^T$. The parameters related to the observer are chosen as $c_1 = c_3 = 0.9$, $c_2 = c_4 = c_9 = 1$, $c_5 = c_6 = c_8 = 0.01$, $c_7 = 0.005$ and the observer gains as $K_{10} = c_8 + (c_5 + c_6 + c_7 c_2)/c_3$, $K_{12} = K_{13} = K_{21} = K_{23} = K_{31} = K_{32} = 0.03$. The gains were given small values to reduce the effect of noise and avoid unwanted phenomena such as overshooting but high enough to ensure an acceptably fast convergence. This was ensured by properly selecting the eigenvalues of the linearized error system to have negative real part.

We assume that we do not have any prior knowledge on the relative positions and thus, choose the estimates as $\hat{x}_{ij}(0) = 0$ which translates to initial observer states given by $\xi_i(0) = -\frac{d_{ij}^e(0)}{2} K_{ij} v_{ij}(0)$. In addition, the initial conditions for the fused estimates are again taken as $\rho_1(0) = [0, 0]^T$, $\rho_2(0) = [0, 0]^T$ while the initial condition for the dynamic scaling $r(t)$ is selected as $r(0) = 1$. We also select the fusion gain $\lambda_0 = 2$ and the initial estimates of the target's velocity as $\hat{v}_0^i(0) = 0$.

Furthermore, we consider the standard scenario where velocity measurements are corrupted by band-limited Gaussian white noise n_{ij} (although any type of noise can be considered) with noise power intensity $\sigma_m = 10^{-4}/5 \text{ (m/s)}^2/\text{Hz}$ and a sampling period of $T_s = 10^{-3} \text{ (s)}$.

On the other hand, the control gains for the distributed law are chosen, respectively, as $k_x = 1$ and $k_y = 0.5$. Finally, the persistent terms in the controllers are chosen as $u_{PE}^1 = [-2\cos(t), -2\sin(t)]^T$, $u_{PE}^2 = [-\frac{1}{5}\sin(\frac{t}{5}) + \sin(t)\sin(\frac{t}{5}) - \frac{1}{5}\cos(t)\cos(\frac{t}{5}), \frac{1}{5}\cos(\frac{t}{5}) - \sin(t)\cos(\frac{t}{5}) - \frac{1}{5}\cos(t)\sin(\frac{t}{5})]^T$, $u_{PE}^3 = [4\sin(t + \frac{\pi}{2}), -4\sin(2t)]^T$. These were selected with different frequencies and amplitudes in order to illustrate the effect in both estimation and tracking as will be shown in the figures below. Of course the richer (larger, faster) the motion of each agent the faster the convergence of

the local estimates and consequently, of the localization error and the tracking error.

The resulting positions of the target and the agents are depicted in Figure 1. In the ideal scenario where the relative positions would be available, and hence not requiring a persistent motion of the agents, and with no measurement noise, the positions of all agents would exactly converge to the target's position. However, since a persisting motion is required to successfully estimate the relative positions, the positions of all agents converge in neighborhoods around the target's position with their size depending on the amplitude of the corresponding persistent input. Of course, the smaller the amplitude of the persistent input the slower the convergence of the estimated relative positions.

From Figure 2 we see that the velocities are PE (and linearly independent) and thus, we can obtain converging local estimates of the relative positions (see Figures 3 to 5). Figure 6 depicts the noisy, estimated and true values of the relative velocity v_{10} in order to illustrate

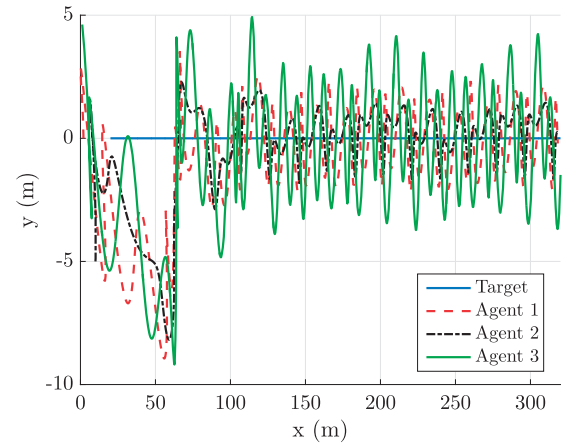


Figure 1. Positions of the target and the agents.

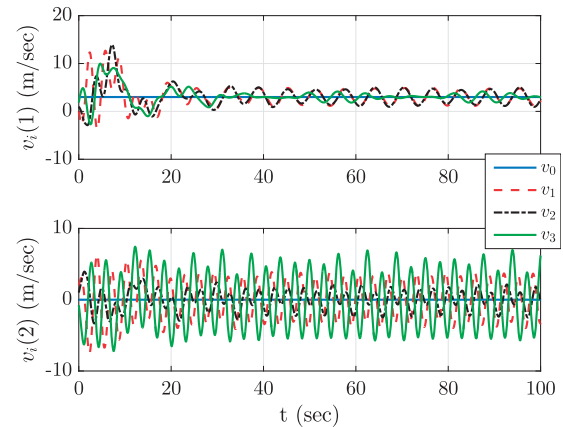


Figure 2. Agents' true velocities.

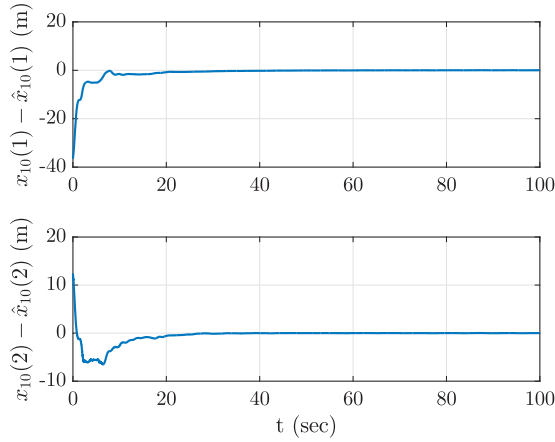


Figure 3. Estimation error for x_{10} .

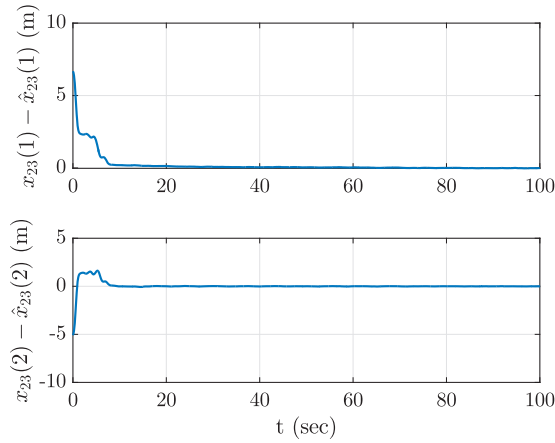


Figure 4. Estimation error for x_{23} .

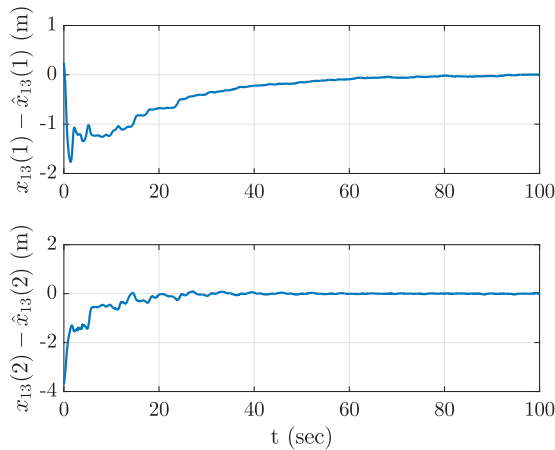


Figure 5. Estimation error for x_{13} .

the persistent excitation and the effect of the applied filtering. In particular, by zooming on a specific time interval we observe the effect of the noise as well as the result of the filtering. Of course, the former can be

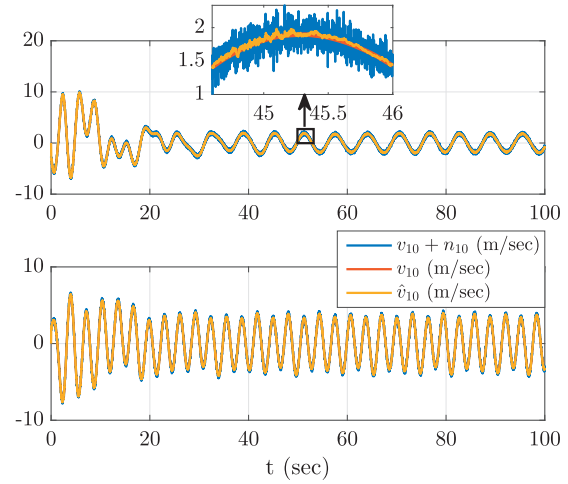


Figure 6. Relative (noisy, estimated, true) velocity v_{10} (upper: first component, lower: second component).

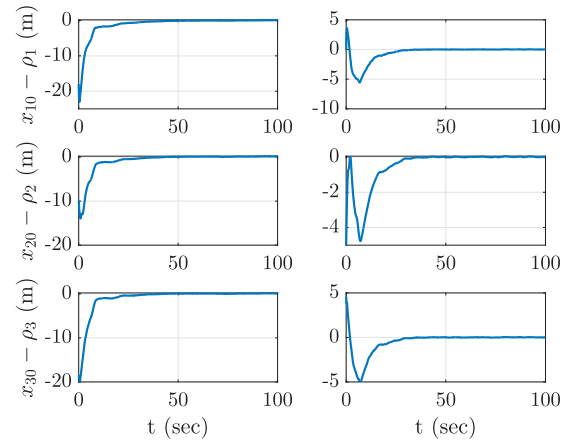


Figure 7. Error between fused estimates ρ_i and true relative positions x_{10} (left column: first component, right column: second component).

further adjusted by proper selection of the filter gains. Similar effects are observed for the other relative and absolute velocities. In addition, notice in both components the effect of the persistent motion required in order to be able to estimate the relative position. As such, this persistent motion (and its magnitude) is imposed by the persistent part of the i th agent's (agent 1 for this figure) control law (u_{PE}^1) that leads to the persistent relative velocity of the figure. Finally, we can visualize the fused estimates for the three agents in Figure 7. We observe that all agents are successfully localized with respect to the target and furthermore, that the effect of the noisy measurements has been significantly removed (although some slight oscillations do appear). Hence, the transient behavior is quite smooth and the convergence is exponential as was proposed by the theoretical analysis.

Now, in order to show the robustness of the proposed approach and how it can be adapted to other scenarios of interest, we consider the same scenario, with same initial conditions, noises, and gains, but now taking into account a desired formation geometry and the interagent collision avoidance. For the former requirement we need only to modify the control law and the target velocity estimator in order to include the desired distances (relative positions), which we simply select as $x^d = [3, 4]^T$, $x_{12}^d = x^d$, $x_{21}^d = -x_{12}^d$, $x_{13}^d = x^d$, $x_{31}^d = -x_{13}^d$, $x_{23}^d = x^d$, $x_{32}^d = -x_{23}^d$. For the latter requirement to be satisfied, we add in the control law an additional term inspired by the avoidance strategy in Kahn et al.,³³ call it u_c^i for the i th agent, which for our scenario gives

$$\begin{aligned} u_c^1 &= \frac{k_c}{q} (\exp(-\hat{x}_{10}^T \hat{x}_{10}/q) \hat{x}_{10} \\ &+ \exp(-\hat{x}_{12}^T \hat{x}_{12}/q) \hat{x}_{12} + \exp(-\hat{x}_{13}^T \hat{x}_{13}/q) \hat{x}_{13}) \\ u_c^2 &= \frac{k_c}{q} (\exp(-\hat{x}_{21}^T \hat{x}_{21}/q) \hat{x}_{21} \\ &+ \exp(-\hat{x}_{23}^T \hat{x}_{23}/q) \hat{x}_{23}) \\ u_c^3 &= \frac{k_c}{q} (\exp(-\hat{x}_{31}^T \hat{x}_{31}/q) \hat{x}_{31} \\ &+ \exp(-\hat{x}_{32}^T \hat{x}_{32}/q) \hat{x}_{32}) \end{aligned}$$

with gain $k_c = 10$ and the parameter $q = 5$ that defines the repulsion distance. Finally we slightly decrease the amplitude of the persistent terms u_{PE} as, $u_{PE}^1 = [-\cos(t), -\sin(t)]^T$, $u_{PE}^2 = [-\frac{1}{5}\sin(\frac{t}{5}) + \sin(t)\sin(\frac{t}{5}) - \frac{1}{5}\cos(t)\cos(\frac{t}{5}), \frac{1}{5}\cos(\frac{t}{5}) - \sin(t)\cos(\frac{t}{5}) - \frac{1}{5}\cos(t)\sin(\frac{t}{5})]^T$, $u_{PE}^3 = [\frac{2}{3}\sin(2t + \frac{\pi}{2}), -\frac{2}{3}\sin(2t)]^T$, to show its impact on the convergence of the estimated relative positions, that will be larger.

The evolution of the agents' positions, the geometric formation of the followers in different time instances, and the relative distances are depicted in Figures 8 to 10, respectively. We observe that the additional requirements (formation and interagent collision avoidance) are readily satisfied and that the relative distances among followers converge around the desired nominal value. The discrepancies observed are due to the contribution of the persistent terms, that are required to ensure the observability, and the desired formation geometry.

For completeness, we illustrate also the time evolution of the true velocities of all agents in Figure 11 as well as the fused estimates for the relative positions in Figure 12. As expected, the estimates have a slower convergence with respect to the previous scenario and present some slight oscillations due to noise around the true values.

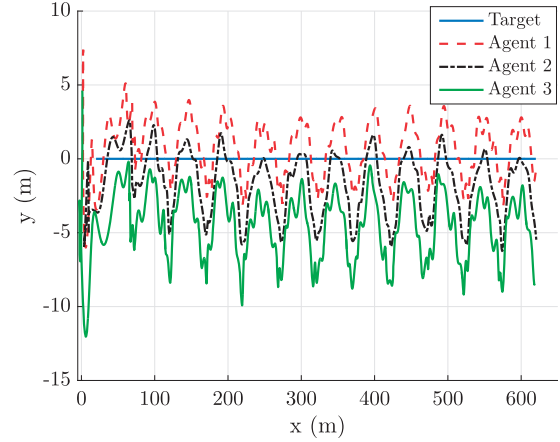


Figure 8. Formation scenario: positions of the target and the agents.

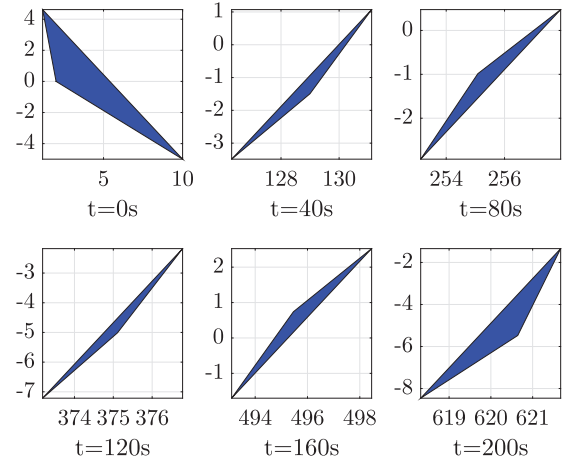


Figure 9. Formation scenario: evolution of the formation of follower agents (($x-y$) in (m)).

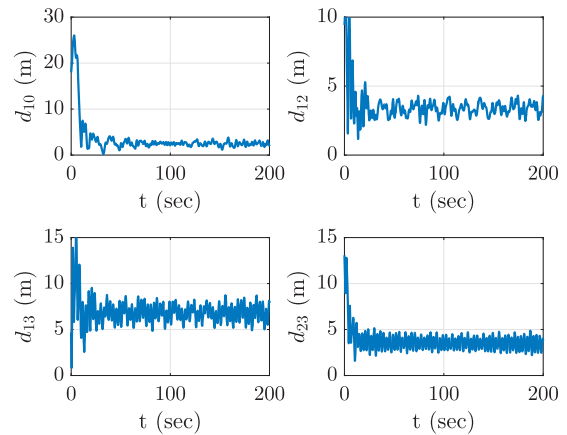


Figure 10. Formation scenario: relative distances.

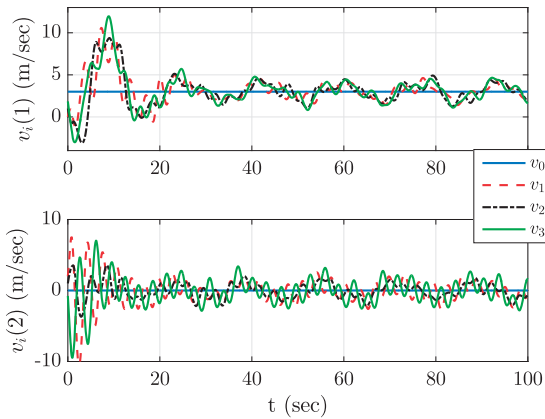


Figure 11. Formation scenario: agents' true velocities.

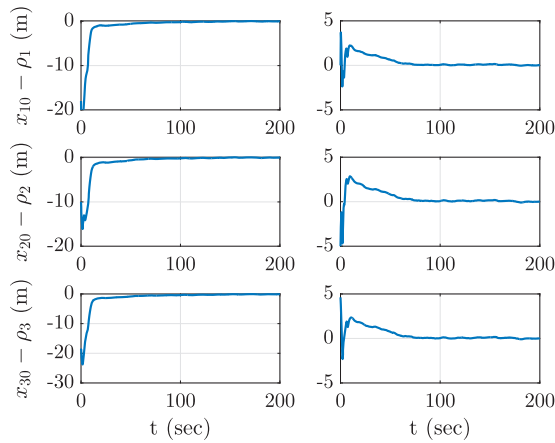


Figure 12. Formation scenario: error between fused estimates ρ_i and true relative positions x_{i0} (left column: first component, right column: second component).

Conclusions

We have proposed a robust algorithm for simultaneous collaborative localization and target tracking problem. The localization mechanism exploits (local) relative distances and noisy velocity measurements so that each agent first obtains an estimation of the relative positions with respect to its neighbors and then fuses this estimate with the ones communicated by the neighbors. These estimates are then fed to a consensus-type distributed control law that includes an estimation of the target's velocity, to achieve exact target tracking. Our algorithm is designed to ensure the observability of the system, represented by a persistence-of-excitation condition on the relative motion of the agents, and the attenuation of noise. The stability properties induced by our algorithm are established through a thorough Lyapunov analysis. Finally, the performance of our scheme is analyzed by means of two simulation scenarios that show among others the robustness to

unaccounted acceleration measurement noise and the possibility to consider a formation geometry and inter-agent collision avoidance.

In the near future, these theoretical results are expected to be tested experimentally on our fleet of quadrotors and under realistic environmental and communication scenarios.

Acknowledgements

The first author would like to acknowledge the fruitful discussions with prof. Emmanuel Nuño of the University of Guadalajara, Mexico.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the PR-GUIMAUVE of ONERA.

Notes

- With some slight abuse of notation we denote the relative measurements for each agent as y_i instead of the more correct y_{ij} . Similarly, in what follows we define the state of the observer as ξ_i instead of the more appropriate ξ_{ij} that would be coherent also with the notation of the corresponding vector x_{ij} . The same notation will be adopted for the mapping β_i .
- Observe that compared to Hong et al.²¹ we define slightly differently the error variable S .

References

- Ren W and Beard RW. *Distributed consensus in multi-vehicle cooperative control*. London: Springer-Verlag, 2008.
- Bachrach A, He R and Roy N. Autonomous flight in unknown indoor environments. *Int J Micro Air Veh* 2009; 1: 217–228.
- Bishop AN, Fidan B, Anderson BDO, et al. Optimality analysis of sensor-target localization geometries. *Automatica* 2010; 46: 479–492.
- Jauffret C and Pillon D. Observability in passive target motion analysis. *IEEE Trans Aerosp Electron Syst* 1996; 32: 1290–1300.
- Tron R, Thomas J, Loianno G, et al. A distributed optimization framework for localization and formation control: applications to vision-based measurements. *IEEE Control Syst Mag* 2016; 36: 22–44.
- Chai G, Lin C, Lin Z, et al. Single landmark based collaborative multi-agent localization with time-varying

- range measurements and information sharing. *Syst Control Lett* 2016; 87: 56–63.
7. Tabarea N, Slotine JJ and Pham QC. How synchronization protects from noise. *PLoS Comput Biol* 2010; 6: 1–9.
 8. Li Y and Sanfelice RG. Interconnected observers for robust decentralized estimation with performance guarantees and optimized connectivity graph. *IEEE Trans Control Netw Syst* 2016; 3: 1–11.
 9. Fox D, Burgard W, Kruppa H, et al. A probabilistic approach to collaborative multi-robot localization. *Auton Robots* 2000; 81: 325–344.
 10. Roumeliotis SI and Bekey GA. Distributed multirobot localization. *IEEE Trans Robot* 2002; 18: 781–795.
 11. Kia SS, Rounds S and Martinez S. Cooperative localization for mobile agents: a recursive decentralized algorithm based on Kalman-filter decoupling. *IEEE Control Syst Mag* 2016; 36: 86–101.
 12. Sarras I, Marzat J, Bertrand S, et al. Collaborative multi-vehicle localization with respect to static/dynamic target from range and velocity measurements. In: *International conference on unmanned aircraft systems (ICUAS)*, Miami, FL, USA, 13–16 June 2017, pp.850–859.
 13. Montijano E, Cristofalo E, Zhou D, et al. Vision-based distributed formation control without an external positioning system. *IEEE Trans Robot* 2016; 32: 339–351.
 14. Piasco N, Marzat J and Sanfourche M. Collaborative localization and formation flying using distributed stereo-vision. In: *IEEE international conference on robotics and automation*, Stockholm, Sweden, 16–21 May 2016, pp.1202–1207.
 15. Bahr A, Leonard JJ and Fallon MF. Cooperative localization for autonomous underwater vehicles. *Int J Robot Res* 2009; 28: 714–728.
 16. Dandach SH, Fidan B, Dasgupta S, et al. A continuous time linear adaptive source localization algorithm, robust to persistent drift. *Syst Control Lett* 2009; 58: 7–16.
 17. Deghat SH, Shames I, Anderson BD, et al. Localization and circumnavigation of a slowly moving target using bearing measurements. *IEEE Trans Autom Control* 2014; 59: 2182–2188.
 18. Karagiannis D and Astolfi A. Dynamic scaling and observer design with application to adaptive control. *Automatica* 2009; 45: 2883–2889.
 19. Praly L, Carnevale D and Astolfi A. Dynamic versus static weighting of Lyapunov functions. *IEEE Trans Autom Control* 2013; 58: 1557–1561.
 20. Hong Y, Hu J and Gao L. Tracking control for multi-agent consensus with an active leader and variable topology. *Automatica* 2006; 42: 1177–1182.
 21. Hong Y, Chen G and Bushnell L. Distributed observers design for leader-following control of multi-agent networks. *Automatica* 2008; 44: 846–850.
 22. Cai H and Huang J. The leader-following consensus for multiple uncertain Euler-Lagrange systems with an adaptive observer. *IEEE Trans Autom Control* 2016; 61: 3152–3157.
 23. Liu W and Huang J. Adaptive leader-following consensus for a class of higher-order nonlinear multi-agent systems with directed switching networks. *Automatica* 2017; 79: 84–92.
 24. Oh KK and Ahn HS. Formation control of mobile agents based on inter-agent distance dynamics. *Automatica* 2011; 47: 2306–2312.
 25. Oh KK, Park M and Ahn HS. A survey of multi-agent formation control. *Automatica* 2015; 53: 424–440.
 26. Franchi A, Stegagno P and Oriolo G. Decentralized multi-robot encirclement of a 3D target with guaranteed collision avoidance. *Auton Robot* 2016; 40: 245–265.
 27. Astolfi A, Karagiannis D and Ortega R. *Nonlinear and adaptive control*. London: Springer-Verlag, 2008.
 28. Martin P and Sarras I. A simple model-based estimator for the quadrotor using only inertial measurements. In: *IEEE 55th Conference on Decision and Control (CDC)*, Las Vegas, NV, USA, 12–14 December 2016, pp.7123–7128.
 29. Martin P and Sarras I. A global observer for attitude and gyro biases from vector measurements. In: *Twentieth IFAC world congress*, Toulouse, France, 9–14 July, 2017, pp.15979–15986.
 30. Loria A and Panteley E. Uniform exponential stability of linear time-varying systems: revisited. *Syst Control Lett* 2002; 47: 13–24.
 31. Rochefort Y, Piet-Lahanier H, Bertrand S, et al. Model predictive control of cooperative vehicles using systematic search approach. *Control Eng Pract* 2014; 32: 204–217.
 32. Loria A and Panteley E. Cascaded nonlinear time-varying systems: analysis and design. *Lect Notes Control Inform Sci* 2005; 311: 23–64.
 33. Kahn A, Marzat J, Piet-Lahanier H, et al. Cooperative estimation and fleet reconfiguration for multi-agent systems. In: *Proceedings of the IFAC workshop on multivehicle systems*, Genova, Italy, 18 May 2015, pp.11–16.