Robotica, Volume 34, Issue 3, March 2016, pp. 703-722
DOI: 10.1017/S0263574714001829

Robust L_∞ convex pose-graph optimisation for monocular localisation solution for UAVs

Mohammed Boulekchour, Nabil Aouf and Mark Richardson

Centre for Electronic Warfare, Cranfield University, Defence Academy of the United Kingdom
Shrivenham, SN6 8LA, United Kingdom
{m.boulekchour, n.aouf, m.a.richardson}@cranfield.ac.uk

Abstract

In the present work, a robust L∞ convex pose-graph optimisation solution for UAVs monocular motion estimation with loop closing is presented. Most solutions proposed in literature formulate the pose-graph optimisation as a least-squares problem by minimising a cost function using iterative methods such as Gauss-Newton or Levenberg Marquardt algorithms. However, with these algorithms, there is no guarantee to converge to a global minimum as they, with high probabilities, converge to a local minimum or even to an infeasible solution. The solution we propose in this work uses a new robust convex optimisation pose-graph technique, which efficiently corrects the UAV's pose after loop-closures detections. Uncertainty estimation using derivative method and its propagation through multiview geometry algorithms are included in the developed solution. The detection of the visual loop closures, in appearance-based navigation, is achieved with our innovative, fast and efficient method based on Bayes Decision Theory with Gaussian Mixture Model (GMM) in combination with the KD-Tree data structure. Our navigation solution has been validated using real-world data in both indoor and outdoor environments acquired by a UAV equipped with monocular system.

Keywords

Unmanned Aerial Vehicles (UAV), robust convex optimisation, pose-graph optimisation, loop closures, Features uncertainties.

1. Introduction

Unmanned and micro aerial vehicles will shortly be as crucial asset to use in important operations, such as inspection, reconnaissance, surveillance and search and rescue. Moreover, they are expected to achieve similar performances as manned aircraft. GPS and other satellite navigation systems may offer valuable assist for these aerial vehicles to accomplish their tasks. However, relying solely on satellite-based navigation systems is not fully reliable especially in crucial and rapid operations. Urban and indoor environment operations present real handicaps to these navigation systems where its availability is extremely limited or even does not exist at all. Inertial Navigation System (INS) or GPS-aided INS system might be used in these situations. However, eventual growth in INS errors is prohibitive to these applications. Therefore, investigating other alternative solutions, such as visual systems, has become a priority for many research programs. Indeed, vision systems are relatively more convenient for Unmanned Aerial Vehicles (UAVs) due to their light-weight, low-cost and the great quality of information they provide.

On the same wavelength, in this work we address the problem of UAVs localisation in unknown environments using monocular visual systems as the only information source. However, the inherent difficulties in UAVs localisation in unknown environments relying on visual sensing impose great research challenges especially when big area coverage from higher altitudes capability is required.

In literature, many studies have been focussing on using vision as a perception means for UAVs navigation[1,2] or even as guidance tool for safely landing of aerial vehicle[3]. Visual navigation approach has been widely studied in the last years as an alternative navigation solution for autonomous aerial systems. It estimates the pose of moving UAVs using visual inputs only with single camera, stereo cameras or multi camera systems [4,5,6]. An aided INS with stereo vision system may be used in a cooperative Visual Simultaneous Localisation and Mapping (VSLAM) design for multiple UAVs in which INS localization errors are corrected in a combination with vision algorithms[7]. In addition, 3D texture mapping models for UAV applications are used as well[8].

Stereo systems exploit the known distance between the two cameras, usually called the baseline, to remove any ambiguity in motion estimation. These systems, however, present an important drawback when this baseline is relatively too small regarding the distance to the scene in consideration. In reality, for aerial systems, the scene in consideration has to be observed from a sufficiently large baseline which is unpractical for UAVs exploring large areas from higher altitudes. Therefore, observing scenes from much greater altitude in comparison to the cameras' baseline reduces a stereo setup to almost a bearing-only sensor suffering similar depth estimation problems as the monocular case (using one single camera). This evidence spurs more research to focus on monocular systems.

Indeed, monocular systems become an unavoidable solution for autonomous aerial navigation systems since it is practical and offers cheap and compact installations. In these systems, the multiple view geometry of a moving UAV is created by distributing frame sequences over time. Correspondences between two or more frames along with the camera calibration parameters are then used to estimate the motion parameters. However, one of the challenging issues of the monocular visual odometry based solution is the scale ambiguity due to the projective effects. One way to tackle this problem is to use pre-knowledge information about the real world as a scaling factor. Few algorithms have successfully been implemented to deal with this issue, where initially known landmarks are used to build a well scaled map [9].

Nutzi et al. presented an approach to estimate the unknown absolute scale in a monocular SLAM frames by fusing through an Extended Kalman Filter (EKF) data from an Inertial Measurement Unit (IMU) with vision [10]. However, this method is tested only on simulated two-meter-cube data which limits its applicability. I. Estiban et al[11], presented a monocular visual odometry algorithm relying on a linear computation of the scale ratio between frame pairs. A more robust solution using based on H∞ filter framework and only scene visual data is presented as well [12].

In practice, and similarly to other navigation systems, errors in UAVs position estimates for aerial visual navigation are continuously growing due to the integration of noisy measurements over time and imperfect computational techniques. This unavoidable drift in motion estimation due to inherent inaccuracy of the devices as well needs to be corrected. Thus, providing additional correction tools would have a crucial impact on the final estimates of the navigation solution. During the last years, visual loop closures have been given more attention and thought of as a powerful and practical tool for motion drift correction. Indeed, after a long navigation into unknown environment, detecting that the autonomous aerial platform has returned to a previously visited place offers the opportunity to correct and to increase the accuracy and the consistency of the vehicle motion estimate. To achieve this, the scientific community started thinking of providing efficient techniques for visual loop closures detection.

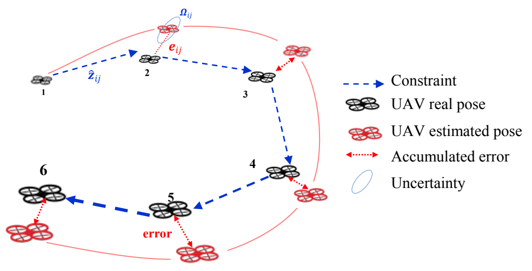


Figure 1: A typical pose-graph representation. The UAV navigates around unknown environment. Poses are connected through constraints (blue dashed arrows). Constraints depict a rigid body transformation between poses.

Solving the problem of detecting loop closure will positively improve the localisation algorithm's performances. On the other hand it also induces additional computational charges. Hence, an efficient real time visual loop closure detection solution is crucial. Most solutions in literatures use classical appearance information only relying on visual Bag-of-Words (BoW) based on SIFT or SURF features [13]. However, these solutions present a number of drawbacks, which can be summarized in the compulsory offline learning and the perceptual aliasing problem caused by clustering [14]. Alternatively, in this work, we propose to use a fast and efficient method for loop closures detection relying on a combination of Gaussian Mixture Model (GMM) with the KD-Tree data structure. In addition and while all other methods use one single feature space to describe images, SIFT features usually, our method adds local colour histograms as a second feature space in order to avoid problems that shape-based description features have.

Once a loop closure is detected, which means the UAV has returned to a previously visited place; the challenge is to ensure a consistent map representation despite of the drift. In the recent years, pose-graph optimisation has become a great technique for loop closure correction where relative constraints, between poses, are used especially those imposed in the loop closure. In visual odometry, each node of the graph represents a vehicle pose. Constraints between these poses are represented by edges between the nodes (Figure 1). These constraints are defined from observations which depict a rigid body transformation between poses. Obviously these constraints are affected by noise and drift (Figure 1). The main objective of the optimisation algorithm then is to recover the optimal configuration of the nodes that best satisfies the constraints (maximises the observation likelihood determined in the constraints) [15]. Therefore, this involves solving a large error minimisation problem. In other words, the principal aim is to jointly optimise the vehicle poses in order to minimise the errors dictated by the constraints (Figure 1).

Pose-graph representation was first proposed by Lu and Milios in 1997 [16]. However, this approach took many years to become a solution for solving error minimisation problems [17]. This work was followed by Gutmann and Konolige who proposed a graph construction technique by including loop closures constraints [18]. In literature, optimisation techniques that recover the optimal poses given the constraints are usually called back-ends. In contrast, front-ends techniques recover the input data to obtain the constraints that are the basis for the optimisation.

Optimisation plays an important role into delivering the accuracy required in motion estimation based visual navigation. Over the last decade, numerous optimisation algorithms have been recommended. These algorithms can be classified into two main categories:

The first category employs linear approaches, which are based on least squares minimisation. In this approach, Singular Value Decomposition (SVD) is usually adopted where an algebraic cost function is minimised. These methods have the advantage of offering a closed-form solution with a simple implementation. However, the quantity being minimized is not geometrically or statistically meaningful.

The second category relies on iterative estimation techniques where a cost function is minimised using iterative algorithms such as Levenberg-Marquardt, Gauss-Newton, gradient descent or conjugate gradient. The cost function here is geometrically interpretable and can statistically be optimal under an assumption of Gaussian noise. Commonly, this category is the adopted technique as a solution for pose-graph optimisation problems.

In order to have a deep sight on this technique, let us consider this example (Figure 1). Let $X = (x_1, \cdots, x_i)$ be a vector where its entries represent the poses of a moving UAV. Let z_{ij} and Ω_{ij} represent the measurements and their covariance between the nodes i and j. Let $\hat{z}_{ij}(x_i, x_j)$ encodes the measurement prediction given the poses $x_i$and x_j which is the relative transformation between the two poses. Let e_{ij} be the error between the predicted observation $\hat{z}_{ij}$ and the real observation z_{ij}. The main aim of the optimisation problem is then to find the configuration of the nodes X^* that minimises the negative log likelihood $F(X)$ of all observations, where $F(X)$ is defined as:

$$F(X) = \sum \boldsymbol{e}_{ij}{}^{\top} \boldsymbol{\Omega}_{ij} \boldsymbol{e}_{ij} \qquad (1)$$

This leads to solve the following optimisation problem:

$$X^* = \operatorname{argmin} F(X) \qquad (2)$$

Here, the optimisation problem is formulated as a non-linear least squares problem, where the error is the squared L_2 norm. The error function is approximated by its first order Taylor expansion around the current initial values $\breve{X}$, which are selected usually by guess. This leads to solve a linear system of the following form:

$$A\Delta x = -b \qquad (3)$$

Here Δx is the increment of the system. Then, the solution X^* is obtained by adding the recovered increments to the initial values:

$$X^* = \breve{X} + \Delta x \qquad (4)$$

Algorithms of this category iterate the solution in (3) and update the equation (4). In each iteration, the previous solution is used as the initial guess. This procedure is repeated until a satisfactory convergence standard is achieved. Usually, until a predefined termination criterion is met. However, and notwithstanding of their dependency on good initialisation guess, these algorithms present high probabilities of convergence to a local minimum or even an infeasible solution. As a valid alternative and to get around the drawbacks of linear and iterative techniques, we present a third manner to solve the pose graph optimisation problem for visual navigation. This alternative is the convex optimisation through more robust norm such as the $L\infty$ norm.

The remainder of the paper is structured as follows: Convex optimisation section provides an overview of the convex optimisation. Overview of the proposed solution section details the general navigation solution pipeline. Monocular motion estimation module section presents the solution adopted for motion estimation. Loop closure module section explains our loop closure detection technique. The Pose optimisation module section describes the implementation design for pose-graph convex optimisation. Experimental validations are given followed by giving the main conclusions of this work in the last two sections.

2. Convex Optimisation

In contrast to linear and iterative methods, convex optimisation ensures getting a single global minimum, and the cost function is geometrically meaningful. One more extremely important feature of convex optimisation in a pose-graph representation is the possibility of adding constraints to the optimisation problem allowing to the designer to incorporate any prior knowledge about the solution into the optimisation problem. This certainly would increase the convergence speed [19].

2.1 Convex optimisation formulation

A convex optimisation problem has the form:

$$\begin{aligned} &\mathbf{min}_x \quad f_0(x) \\ &\textbf{Subject to} \quad f_i(x) \leq 0\,, i = 1, \cdots, m \end{aligned} \tag{5}$$

It is basically the problem of finding an x that minimises $f_0(x)$ among all x that satisfy the constraints $f_i(x) \leq 0$. The vector $x \in \mathbb{R}^{\mathrm{n}}$ is the optimisation variable. The function $f_0 : \mathbb{R}^{\mathrm{n}} \mapsto \mathbb{R}$ is the objective function and the inequalities $f_i(x) \leq 0$ are called the inequality constraints. All functions here are convex.

The particular convex optimisation problems where the objective function f_0 is linear and the constraints are of the form: $\|A_i x + b_i\|_2 \leq {c_i}^T x + d_i$ are called Second-Order Cone Programming (SOCP). Therefore, a SOCP is an optimisation problem of the form:

$$\begin{aligned} &\mathbf{min}_x \quad f^{\top} x \\ &\textbf{Subject to} \ \|A_i x + b_i\|_2 \leq {c_i}^{\top} x + d_i \quad \text{for } i = 1, \dots, m \\ &\qquad {g_i}^{\top} x = h_i \qquad\qquad\quad \text{for } i = 1, \dots, p \end{aligned} \tag{6}$$

Where vectors $x, f, c_i, g_i \in \mathbb{R}^n$, scalars $d_i, h_i \in \mathbb{R}$, matrix $\mathrm{A}_i \in \mathbb{R}^{(n_i-1)\times n}$ and $b_i \in \mathbb{R}^{n_i - 1}$.

In our solution for pose-graph convex optimisation, we adopted the L∞ optimisation that is formulated as a min-max form:

$$\begin{aligned} &\textbf{find} \quad \mathbf{min}_x \quad \max_i \frac{\|A_i x + b_i\|_2}{{c_i}^{\top} x + d_i} \\ &\textbf{Subject to} \quad {c_i}^{\top} x + d_i > 0 \end{aligned} \tag{7}$$

This problem may be transformed into an equivalent problem by incorporating a new variable δ: [19,20]:

$$\textbf{find} \quad \min_{x,\delta} \delta$$

$$\text{Subject to} \quad \|A_i x\|_2 \leq \delta\, c_i^{\top} x \tag{8}$$

For a given value of $\delta \in \mathbb{R}$, our optimisation problem will become a sequence of SOCP feasibility problems [19,20]. This leads toward using a bisection search to find a minimum value δ^* for which the optimisation problem still feasible [20,21]. If the SOCP problem is feasible then there must exist a more optimal solution $\delta^* \leq \delta$. However, if the SOCP is infeasible, then the optimal solution must be greater than δ ($\delta^* > \delta$).

Thus, any problem that could be *formulated* as the problem in (8) can be solved as a SOCP sequence.

2.2 Robust convex optimisation formulation

Extracting feature points is the first step for our solution for pose-graph optimisation. Since the detected feature points, regardless the nature of the detector, have some uncertainty [22,23], we propose in this work to include these uncertainties in our pose-graph optimisation problem. We then solve it using robust L∞ convex optimisation via the Second-Order Cone Programming (SOCP). In addition to uncertainties in feature positions, our solution takes as well into account the rotational and the translation motion uncertainties, which are estimated through the propagation of feature position uncertainties via multiple view geometry algorithms [24]. Most researchers avoid uncertainty due to the added complexity in constructing the robust optimisation model and to the lack of knowledge of the nature of these uncertainties especially their propagation. On the contrary, our work proposes new robust solutions via convex optimisation along with estimating the uncertainties in every step of the algorithm.

Robust optimisation, in general form, deals with two sets of entities: decision variables and uncertain variables. Here, the first aim of worst-case robust optimisation is to recover the optimal solution on the decision variables such that the worst-case objective function is minimised and the

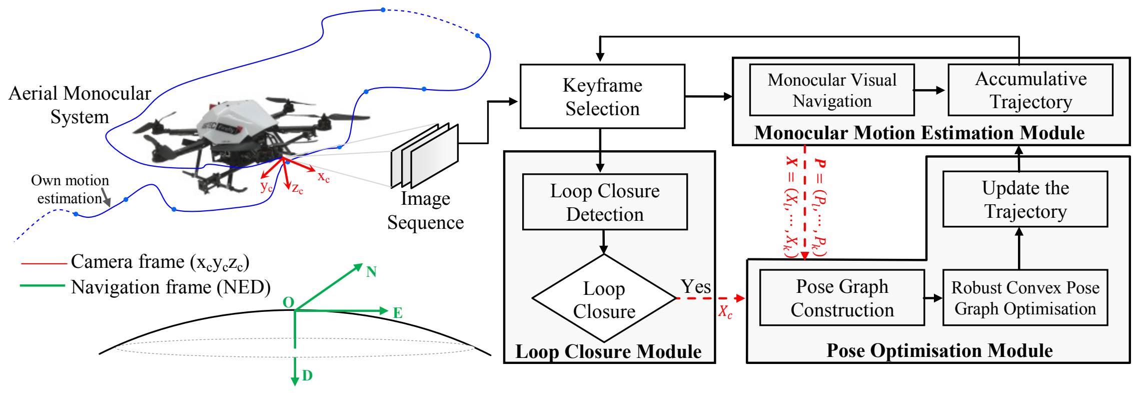


Figure 2: A block diagram showing the main architecture of the proposed solution. The setup consists on an UAV equipped with a monocular system capturing sequences of images and estimating its own motion as it moves (Monocular Motion Estimation Module). When the UAV navigates around a cycle and returns close to a visited position (Loop closure Module); the convex Pose-graph Optimisation Module will be triggered to optimise the UAV poses around the loop.

constraints are robustly feasible. This is done while the uncertainty is allowed to take arbitrary values in a defined uncertainty set [25]. The general form of this robust optimisation is given by:

$$\begin{aligned} &\min_{x} \ \max_{\omega} \quad f(x,\omega) \\ &\textbf{subject to} \quad g(x,\omega) \le 0 \ \ (\forall \omega \in \mathcal{W}) \end{aligned} \tag{9}$$

Here ω is the uncertain variable, $\mathcal{W}$ is the uncertainty set and x is the decision variable. Similarly to (5), this problem can efficiently be recast and solved using Second-Order Cone Programming (SOCP)[26].

3. Overview of the Proposed Solution

The overall of the proposed solution is described in Figure 2. The setup consists on an Unmanned Aerial Vehicle (UAV) equipped with a fully calibrated monocular system with known intrinsic parameters K capturing sequences of images as it moves. In a loop closure scenario, the UAV navigates around a cycle and returns close to an already visited position. Unfortunately and because of the drift, there is an error between the final UAV pose and its estimate. The whole solution can be divided into three sub-tasks (Figure 2): First, we estimate the motion using a monocular visual navigation algorithm (The Monocular Motion Estimation Module). Second, we detect that a place is revisited (Loop Closure Module). The output of this module is a geometric constraint between two camera keyframes, which typically belong to the same place. Third, we correct the drift in the final pose estimate by distributing the error around the loop.

Therefore, the main steps of the solution can be defined as follows:

- Acquisition of image sequences from the onboard camera.
- Keyframe selection based on the amount of the translation regarding the previous selected keyframe.
- Calling the Monocular Motion Estimation Module (MMEM) using the Monocular visual navigation algorithm [12].
- In parallel, a Loop Closure Module (LCM) is launched. This module processes the data asynchronously, as they arrive, in order to detect previously visited places. This module uses Bayes Decision Theory with Gaussian Mixture Model (GMM) in combination with the KD-Tree data structure.
- If the output of the LCM is positive, which means the vehicle has returned to an already visited location, a convex Pose-graph Optimisation Module (POM) will in turn be triggered to optimise the vehicle poses around the loop. And continuously, the whole trajectory is updated accordingly.

4. Monocular Motion Estimation Module

The Monocular Motion Estimation Module (MMEM) estimates the UAV's rotations and translations as it moves from visual input alone. For this purpose we propose to adopt and extend our previously presented algorithm[12] where an UAV equipped with a fully calibrated monocular system with known intrinsic parameters K is used. The final goal of this module is to estimate the UAV pose at each time step relying only on the captured images and the incorporation of the uncertainties. The main steps of this algorithm are:

- Extraction of image feature points using SIFT detector and estimating their uncertainties.
- Estimating the initial relative rotations R_i and the translation t_i via the essential matrix.
- Estimation of the propagated uncertainties to R_i and t_i through the normalized 8-point algorithm and SVD.
- Estimation of the 3D scene points using convex optimisation along with their uncertainties.
- Optimising the motion using robust L∞ convex optimisation taking in consideration all sources of uncertainties with a sequence of camera resectioning /triangulation.
- Computing the unknown absolute scale ratio using robust least squares approach.

The outputs of this module at time step k are the UAV's absolute positions $\boldsymbol{P} = (P_1, \cdots, P_k)$ where $P_i = (x_i, y_i, z_i)^\top$ and the relative-pose estimates $\boldsymbol{X} = (X_{12}, X_{23}, \cdots, X_{k-1,k})$ between the selected keyframes. These relative transformations are in reality the relative rotation and translation $(R_{k-1,k}, t_{k-1,k})$ between the consecutive selected keyframes.

5. Loop Closure Module

After long navigation, the drift would affect the estimated positions from the monocular motion estimation module. Even though drift during exploration is unavoidable, it is important to keep it as small as possible. This drift is due to the integration of noisy measurements over time and to imperfections and inherent inaccuracy of the devices. Consequently, complementary information to overcome these cumulative drift errors would become crucial. As a great correction tool, our solution relies on visual loop closures detection. This is modelled through position constraints given by the Loop Closure Module when the vehicle returns to a previously visited place. Recognizing previously optimised locations would restore correct estimates and certainly allows generating consistent maps and reduces their uncertainty.

5.1 Image description

For loop closures detection, most methods presented in the literature rely on just one single feature space to describe images. The vast majority of them use SIFT features[27]. However, SIFT features present some drawbacks related to their modest robustness to illumination and affine changes. On the other hand, colours in images provide great and valuable information about the scene. Therefore, and in order to compensate the issues of SIFT descriptor, our solution uses local colour histograms descriptors as a second space along with SIFT features. For this second feature space, images are decomposed into regular-sized subareas. Then, the normalized hue histograms in the Hue Saturation Value (HSV), which are divided into 37 bins, are used as features[28]. Our navigation solution uses this additional feature space descriptor in order to ensure the required visual feature processing robustness especially in less structured environments.

5.2 KD-trees Structure

After feature extraction, all visual loop-closures detection approaches rely on their matching as a way to compare between candidate images for loop closure. The most used technique for that is the nearest neighbour search [29,30]. Adopting a more robust and efficient matching technique is essential especially when dealing with high-dimensional descriptors (which is the case in our solution: 128 and 37-dimensional descriptors for SIFT and colour features respectively). Tree structures have the ability to decrease the search complexity from linear to logarithmic by comparing one dimension of the features each time so they avoid the distance computations[31]. In addition, tree structures are able to provide already sorted neighbours by traversing the tree just once.

In order to exploit their properties, our solution uses two separate set of KD-trees containing descriptors of each feature space of all previously selected images (i.e. places already visited). In parallel, an inverted index, which refers each feature to the image where it was extracted, is also created. In case of a query keyframe, KD-tree provides the top nearest neighbours' features among all images already inserted in the tree.

5.3 Gaussian mixture modelling (GMM)

Our method also adopts Gaussian Mixture Modelling (GMM) with Bayesian theory for loop closure detection. The aim is to classify a query image into previously seen images whose GMM parameters are accumulated and stored into some GMM dictionaries built up for each feature space. When a keyframe arrives to the Loop Closure Module, after image description, a GMM will be fitted to their descriptors. The GMM modelling outputs, for each keyframe, are the mean μ_k, the covariance Σ_k and the mixture weight π_k , $k = 1,2,\cdots,K$ where K is the number of mixtures. These parameters will be inserted into the GMM dictionaries μ, Σ, π, which will be used as input to the Bayesian loop closure detection algorithm.

5.4 Loop closure detection pipeline using combination of GMM with KD-Trees

This combination takes advantages of the robustness of the KD-Trees in features matching and the efficiency of the GMM representation. All previously proposed KD/Tree-based solutions compare the descriptors of a query keyframe to *all* descriptors in *all* already selected keyframes. In these solutions, the tree is continuously incremented with all descriptors of any new keyframe. Then, the keyframe that contains the most matched features with the query keyframe will be declared as a loop closure after obviously a second step of further verification. The big problem with these solutions is the significant increase of the size of the tree especially for vehicles navigating for long distances. This makes the nearest neighbours search a lot costly in terms of computational time, even with KD-Tree structures. This issue limits their adoption in real life applications.

Our solution, however, does not insert all the descriptors of the new keyframes into the tree structure. Instead of that, a Gaussian Mixture is modelled for each new keyframes, and only the means μ of these mixtures are inserted into the KD-Tree. Therefore, the descriptors of a query keyframe are compared to just the means of the descriptors of all keyframes already selected (i.e.

inserted into the tree). Hence, a significant reduction of the size of the tree would be achieved. Consequently, the search time for the nearest neighbours would be considerably reduced as well. Our solution may be divided into three main stages:

- Stage 1: KD-Tree with Gaussian mixture Model:

  - First, for any new keyframe and after image description (features extraction), a Gaussian mixture model will be fitted to their descriptors, where a KD-Tree is updated over the means μ_k.
  - An index vector is created which maps the means μ_k to their corresponding images from which they are modelled.
  - Given a query keyframe, a nearest neighbours search to their descriptors in the already built up KD-Tree is performed. The top P nearest frames from the sorted images are returned.

- Stage 2: Full representation technique:

  - A new KD-Tree of ***all*** the descriptors, this time, of the P frames is constructed.
  - Then, a search through this new KD-Tree for nearest neighbours with query keyframe is done.
  - Every nearest neighbour votes for the image it comes from. The frames with higher scores will be selected for the third stage.

- Stage 3: geometric consistency check:

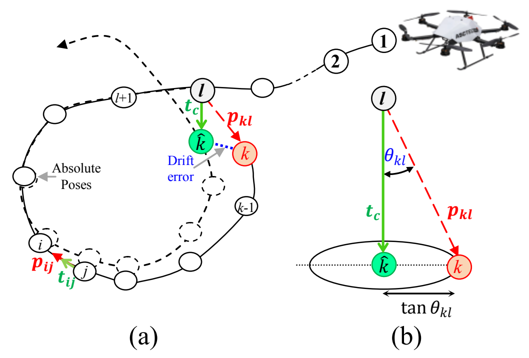


Figure 3: Convex pose-graph optimisation problem. (a) Solid line shows the recovered trajectory of the UAV before optimisation. Nodes represent UAV's poses and edges represent constraints. After loop closure detection -between nodes k (red) and l (gray)-, the loop-closure relative pose constraint t_c (green) is estimated indicating that node k is supposed to be in $\hat{k}$. Clearly, the angular error between t_c and p_{kl} is considerable while it is not between t_{ij} and p_{ij}. Convex pose-graph optimisation corrects the trajectory (dashed line) by distributing the drift error (or the angular error) around the loop. (b) Depicts the geometry of the convex optimisation for angular error minimisation for one constraint.

– The selected frames for this stage will be confirmed or rejected by a further condition derived from multiple-view geometry. The two images are declared as a loop closure if they satisfy the epipolar geometry constraint.

Mainly the principal output of this module is the loop closure constraint of the current pose of the vehicle. This pose constraint $X_c = (R_c\,, t_c)$ indicates the relative rotation R_c and the relative translation t_c between current keyframe and the keyframe that closes the loop with it. This pose constraint is estimated using multiple view geometry algorithms. This constraint, in fact, evaluates the drift in motion estimation. In other words, this pose constraint tells us where the vehicle should be located (Figure 3).

6. Pose Optimisation Module

Once a loop-closure is detected, convex pose-graph optimisation module performs the correction of any drift occurred during the monocular motion estimation. In this section, we present a new convex pose-graph optimisation approach, which robustly corrects the rotation and the translation drift at loop closures.

Let $\boldsymbol{X} = (X_{12}, X_{23}, \cdots, X_{ij}, \cdots, X_{k-1,k})$ be the relative pose estimates and $\boldsymbol{P} = (P_1, \cdots, P_k)$ be the UAV's absolute positions estimated from the Monocular Motion Estimation Module, where k is the index of the current pose (Figure 3.a). When a loop closure is detected, assuming between nodes k and l, then the loop-closure constraint is estimated. This constraint defines the relative pose between the two keyframes of the loop closure:

$$X_c = (R_c\,, t_c) \tag{10}$$

In our case $X_c = X_{kl} = (R_{kl}\,, t_{kl})$. The aim of the pose-graph optimisation is to find the optimal configuration of the UAV's positions $\widehat{\boldsymbol{P}} = (\hat{P}_l, \cdots, \hat{P}_k) \in \mathbb{R}^{3n}$ that satisfies all the constraints including the loop closure constraints. In reality, this configuration would be obtained by an optimal distribution of the drift error over all relative constraints.

Most solutions proposed in literature formulate this pose-graph optimisation as a least-squares problem by minimising a cost function similar to one given in equation (2) above. Most of these solutions use iterative estimation methods for minimisation such as Gauss-Newton or Levenberg Marquardt algorithms [16,17,18,32]. However, with these methods there is no guarantee of convergence to the global minima. Furthermore, they could lead to an infeasible solution. As such, these methods are also very dependent on good initialisation.

In contrast, our solution recovers the optimal positions configuration by using convex optimisation through the adoption of a more robust norm such as the L∞ norm. Contrarily to linear and iterative optimisation methods, convex optimisation guarantees convergence to a single and global minimum.

6.1 Convex pose-graph optimisation formulation

Throughout this section, we consider the scenario in which a loop closure between the keyframes with indices k and l is confirmed (Figure 3.a). In this case, the Pose Optimisation Module extracts, from the Monocular Motion Estimation Module, all the UAV's absolute positions $\boldsymbol{P} = (P_l, \cdots, P_k)$ involved in the loop along with their relative poses $\boldsymbol{X} = \left(X_{l,l+1}, \cdots, X_{k-1,k}\right)$, from index k back to index l (Figure 3.a). In addition to that, the loop closure constraint $X_c = X_{kl}$ (which is the relative pose between the loop-closure keyframes k and l) is extracted as well from the Loop Closure Module (red dashed arrows in Figure 2). These poses will serve as inputs to the Pose Optimisation Module.

Among all the extracted absolute positions $\boldsymbol{P} = (P_l, \cdots, P_k)$, let us consider first the two consecutive positions P_i and P_j and their relative pose $X_{ij} = \left(R_{ij}, t_{ij}\right)$, where $l \leq i < j \leq k$. Let us now consider the two 3-vectors p_{ij} and t_{ij}, where $p_{ij} = \left(P_j - P_i\right)$ represents a vector linking the two absolute positions P_i and P_j and the 3-vector t_{ij} is the relative translation vector in the relative pose X_{ij} which was estimated by the Monocular Motion Estimation Module. The relative constraint between these two nodes may be defined as:

$$t_{ij} \equiv p_{ij} \qquad (11)$$

This relative constraint in (11) means that vectors t_{ij} and p_{ij} are identical and have exactly the same orientation or superposed (Figure 3.a). In our scenario of a loop closure, this remains valid for all frames i and j where $i, j = l, \cdots, k$. However, due to error drift, this is not the case for the loop closure constraint between k and l:

$$t_{kl} \not\equiv p_{kl} \tag{12}$$

This inequality (or drift error) can be expressed as angular error θ_{kl} between the two vectors. This is illustrated in Figure 3.b. It is clear that, due to the drift error, the node k should be moved into the node $\hat{k}$ in order to satisfy the loop closure constraint. To do that, we have to minimise the angle θ_{kl}. However, minimising θ_{kl} will imply changing all the graph's nodes around the loop. Thus, the job of convex pose-graph optimiser is to find the optimal nodes' positions.

Note that each relative translation t_{ij}, including the loop closure relative translation t_{kl}, forms a cone in $\mathbb{R}^3$ with vertex the node j, axis t_{ij} and angle determined by θ_{ij} (Figure 3.b). The aim of the Pose-graph Optimisation Module is then to distribute this angular error around the loop. Hence, closing the loop as illustrated in Figure 3.a (dashed line). In other words, the objective of the optimisation is to modify all the absolute positions P_i, in a way such that all angular errors (cones' angles) are as close to zero as possible. To do that, we consider all the relative constraints in (11) and the loop closure constraint in (12) as measurements (constants) and the new vehicle's positions $\widehat{\boldsymbol{P}} = (\hat{P}_l, \cdots, \hat{P}_k) \in \mathbb{R}^{3n}$ is our optimisation variable where n is the number of nodes involved in the loop.

6.2 *The optimisation problem*

To minimise the angular error θ_{kl}, we may instead minimise the tangent of this angle. Then the error residual associated with all t_{ij} is $\epsilon = tan\,\theta_{ij}$ where $0 < \theta_{ij} < \frac{\pi}{2}$. These error residuals give the error vector:

$$\epsilon = \left(\varepsilon_{l,l+1}, \cdots, \varepsilon_{k-1,k}, \varepsilon_{kl}\right)^{\top} \tag{13}$$

In minimising the angular errors, the most important part of t_{ij} is its orientation, which indicates the direction between nodes. Therefore, using unit vectors divided by its norm is more appropriate. The estimated $\widehat{\boldsymbol{P}}$ is then the one that minimises the norm of this error vector. In our solution, we adopted L∞ norm. Thus, the cost function is defined as:

$$F(x) = \|\epsilon\|_\infty = \max_{i,j} |\varepsilon_{ij}| = \max_{i,j} \tan \theta_{ij} \quad (14)$$

This may be formulated as the min-max optimisation problem:

$$\textbf{find} \quad \mathbf{min}_{\hat{P}} \quad \mathbf{max}_{i,j} \quad \tan \theta_{ij} \qquad (15)$$

This means that we perform a min-max optimisation over all the cones of the graph. As detailed in section 2.12.A above, this may be reformulated as the problem [20,21]:

$$\textbf{find} \quad \mathbf{min}_{\hat{P},\delta}\, \delta \ \textbf{subject to} \tan \theta_{ij} \leq \delta \ \forall\, i,j \, (16)$$

In order to solve this problem as a SOCP sequence, it has to be formulated as the problem in (8). To do that, let us reformulate $\tan \theta_{ij}$. Note that the dot product of the two vectors that form the angle θ_{ij}, t_{ij} and p_{ij}, is $t_{ij}{}^{\top}.p_{ij} = \|t_{ij}\|\|t_{ij}\| \cos \theta_{ij}$, and their cross product's length is $\|t_{ij} \times p_{ij}\| = \|t_{ij}\|\|t_{ij}\| \sin \theta_{ij}$. Therefore, dividing the cross product's length by the dot product yields:

$$\tan \theta_{ij} = \frac{\|t_{ij} \times p_{ij}\|_2}{t_{ij}{}^{\top}.p_{ij}} \qquad (17)$$

$$\tan \theta_{ij} = \frac{\|t_{ij} \times (\hat{P}_i - \hat{P}_j)\|_2}{t_{ij}{}^{\top}(\hat{P}_i - \hat{P}_j)} \qquad (18)$$

$$\tan \theta_{ij} = \frac{\left\|[t_{ij}]_\times (\hat{P}_i - \hat{P}_j)\right\|_2}{t_{ij}{}^{\top}(\hat{P}_i - \hat{P}_j)} \qquad (19)$$

Thus, the optimisation problem in (16) may be rewritten as:

$$\mathbf{min}_{\hat{P},\delta} \quad \delta$$

$$\textbf{Subject to} \left\|[t_{ij}]_\times (\hat{P}_i - \hat{P}_j)\right\|_2 \leq \delta \left(t_{ij}{}^{\top}(\hat{P}_i - \hat{P}_j)\right) \qquad (20)$$

$$t_{ij}.\left(\hat{P}_i - \hat{P}_j\right) > 0 \ \text{ for each } t_{ij}$$

This falls exactly under the desired SOCP form (8) with $A_i = [t_{ij}]_\times$.

This problem with the form in (20) is only solvable up to a translation and a scale. We use the constraint $\hat{P}_1 = P_1$ to remove the translation ambiguity. More importantly, to deal with scale ambiguity we exploit the known initial absolute positions before optimisation P_i so: $(\hat{P}_i - \hat{P}_j) \leq (P_i - P_j)$. Hence, our optimisation problem (20) may be rewritten as:

$$\mathbf{min}_{\hat{P},\delta} \quad \delta$$

$$\textbf{Subject to} \left\|[t_{ij}]_\times (\hat{P}_i - \hat{P}_j)\right\|_2 \leq \delta \left(t_{ij}{}^{\top}(\hat{P}_i - \hat{P}_j)\right) \qquad (21)$$

$t_{ij}.\left(\hat{P}_i - \hat{P}_j\right) > 0 \quad \text{for each } t_{ij}$

$\left(\hat{P}_i - \hat{P}_j\right) \leq \left(P_i - P_j\right)$

The minimisation in (21) is as the desired form (8), which may be solved via a sequence of SOCP feasibility problems as described in Section 2 [20].

6.3 Robust Convex pose-graph optimisation formulation

In the previous section, the convex optimisation problem is formulated with the assumption that image keypoints have been extracted perfectly with no uncertainties. However, detected feature points, regardless of the feature detector, have some uncertainty in their positions. The common way of expressing this uncertainty information is in terms of covariance matrices.

In our scenario of loop closure between nodes k and l, the input to pose-graph optimisation are the relative poses $X = \left(X_{l,l+1}, \cdots, X_{k-1,k}\right)$ where $X_{ij} = \left(R_{ij}, t_{ij}\right)$ and the loop-closure relative pose $X_c = X_{kl} = (R_{kl}, t_{kl})$. Therefore, since image keypoints correspondences are used to estimate t_{ij}; let $\Delta_{t_{ij}}$ be its uncertainties which are estimated through the propagation of feature position uncertainties via multiple view geometry algorithms [24].

Thus, the optimisation problem in (20) becomes:

$\mathbf{min}_{\hat{P},\delta} \quad \delta$

Subject to:

$$\left\|\left([t_{ij}]_\times + \Delta_{t_{ij}}\right)\left(\hat{P}_i - \hat{P}_j\right)\right\|_2 \leq \delta\left(\left(t_{ij} + \Delta_{t_{ij}}\right)^\top \left(\hat{P}_i - \hat{P}_j\right)\right). \tag{22}$$

$\left(t_{ij} + \Delta_{t_{ij}}\right).\left(\hat{P}_i - \hat{P}_j\right) > 0$ for each t_{ij}.

This again is of the desired form in (8) with $A_i = \left[t_{ij}\right]_\times + \Delta_{t_{ij}}$ which can be solved using a sequence of robust SOCP.

6.4 Computational complexity

This section first illustrates the computational complexity advantage of deploying a combination of KD-tree structure with GMM where a comparison with the complexity methods that use KD-tree structure alone is given. Then, we compare the L∞ convex pose-graph optimisation complexity with the classical Levenberg-Marquardt algorithm.

6.4.1 *Computational complexity of the combination of KD-tree structure with GMM solution*

It is well known that the search complexity of tree structure is decreased from linear to logarithmic. Suppose we are looking for a loop-closure in m previously visited frames in which each frame is described with an average of n features (n in general is between 1000 and 4000), then building a KD-tree has $O(nm \log nm)$ time complexity and $O(K \log nm)$ space complexity (K is the tree dimension). A search for M nearest neighbours costs closely to $O(M \log n\, m)$. However, by introducing Gaussian Mixture Modelling (GMM), instead of inserting n features for each frame, only k means μ_k mixtures are inserted into the KD-Tree where k is the number of mixtures (GMM dimension) (in our solution we use 5 mixtures). Consequently, in comparison to methods that use KD-tree only, the computational complexity would be significantly reduced by a factor of:

$$f = \frac{n}{k}\frac{\log(nm)}{\log(km)} \qquad (23)$$

Note that k is significantly smaller than n. Then the computational cost of our solution becomes: $O(km \log km)$ plus the cost of the GMM modelling. This later modelling is done using the Expectation Maximization algorithm where for each frame with n features, the computational complexity is given by: $O(nkd + nk)$ for the E-step and $O(2nkd)$ for the M-step (d is the feature dimension).

6.4.2 *Computational complexity of the convex pose-graph solution*

For the optimisation problem, Levenberg-Marquardt algorithm has cubic complexity in the number of parameters: $O(N^3)$ per iteration (N is the number of the poses in the graph)[5]. Convex pose-graph optimisation, however, is solved by a bisection algorithm[21],[20]. The convex optimisation problems given in (21) and (22) are solved at each step by a feasibility check. This leads to solve N second order cone feasibility problem. Therefore, this problem has a computational complexity of just $O(\sqrt{N})$ and a memory requirement of $O(N)$[33].

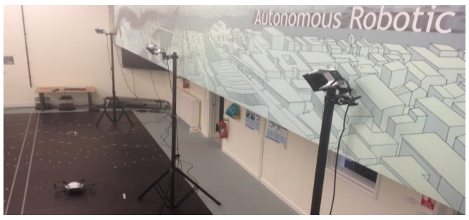


Figure 4: Setup used in our indoor experiments.

The discussion above concerns the computational complexity per iteration. In fact, the number of iterations required for convergence is extremely important. In our solution, the upper and the lower parameters of the bisection algorithm are chosen according to the previous optimisation parameters where a global solution was found. This technique reduces the search area and consequently less iterations would be required for convergence.

7. Experimental Validation

This section discusses the experimental evaluations of the proposed solution. Comparison with iterative methods based on Levenberg-Marquardt algorithm is given as well. We compare our convex optimisation with the state-of-the-art implementations using the open-source implementation[32] in which pose-graph optimisation problem is formulated as a least-squares minimisation problem and solved iteratively using Levenberg-Marquardt.

Experiments are performed using an AscTec Firefly MAV platform with a fully-calibrated forward looking camera. Implementation of these techniques is conducted using real-world data in both indoor and outdoor environments. Indoor experiments are held in our laboratory as shown in Figure 4. Ground-truth in indoor experiment is collected from an OptiTrack motion-capture system that provides absolute position information with millimetre accuracy at 100 Hz. Implementations to generate the results shown in this section is based on a sequence of robust SOCP feasibility problem for the convexity task using SeDuMi toolbox[34] and Yalmip[25] toolbox for uncertainties modeling. The estimated UAV motions are aligned with the ground truth and the Euclidian distance errors on UAV position are computed.

First, we want to illustrate the performances of our solution in term of computational complexity for loop-closure detection. Here, we compare our results to equivalent methods which use tree structure but with full representation, like the one presented in [31]. In this work, the average time for tree building is 16.2ms. This operation is equivalent, in our solution, to GMM modelling, building KD-tree with GMM parameters and building the full KD-Tree of the P frames. All these operations are done together in just 7.51 ms; which is less than half time required for the technique that use all frames. Note that our solution uses 5 mixtures and an average value of P of 15. Search and indexing take less than 5 ms in the proposed solution while it exceeds 10 ms in techniques that search in all frames.

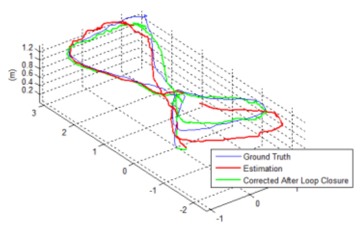
As our global solution is modular, any alternative algorithm could be used in any module among the three modules of the solution (Figure 2). Therefore and in order to independently compare the performance of each module, two comparison scenarios were retained as shown in Table 1. These scenarios serve to assess the performances of our pose-graph optimisation method (CVX) without the influence of the loop closure method. In the first scenario, we compare our convex pose-graph optimisation (CVX) with iterative LM pose-graph optimisation (LM) and for loop-closure detection; the same classical bag-of-words (BoW) method is used. The second scenario is similar to the first one but we use our GMM/KD-Tree method for loop closure detection instead.

Table 1. The two comparison scenarios

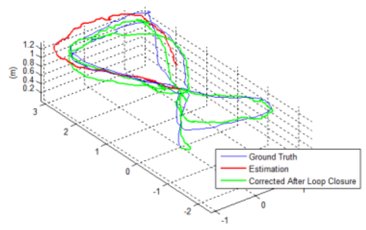
	Pose-graph method		Loop-closure method
Scenario 1	CVX	+	BoW
	LM	+	
Scenario 2	CVX	+	GMM/KD-Tree
	LM	+	

Investigation on the effect of using a particular loop-closure detection method on the global performances of the solution is conducted as well in this section. Robust convex pose-graph optimisation performances, in which uncertainties in relative translations are incorporated, are also compared to normal convex pose-graph optimisation.

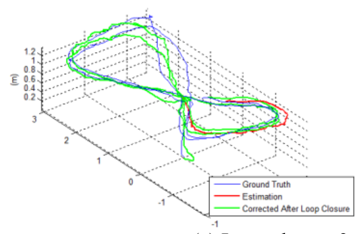
In order to illustrate the final output of our solution, let us consider Figure 5 and Figure 6. These plots show the outcome of the convex pose-graph optimisation process after loop-closures detection. Trajectories before loop closure are shown in red lines. For every loop-closure detection, and before resuming monocular motion estimation, convex pose-graph optimisation is



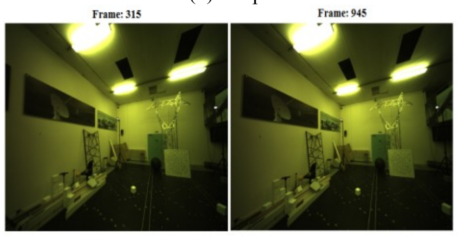
(a) Loop closure 1



(b) Loop closure 2

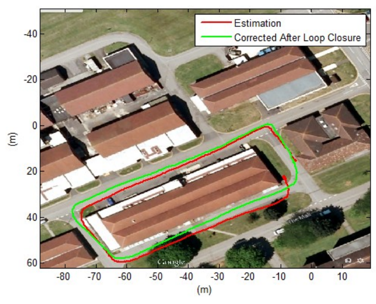


(c) Loop closure 3

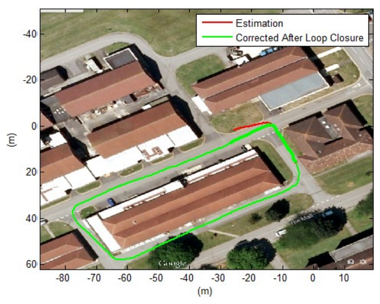


(d) Sample of loop closure frames

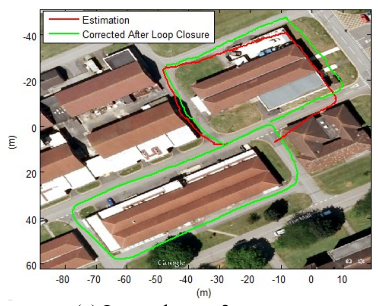
Figure 5: Convex pose-graph optimisation results on indoor experiment. Blues lines show the ground truth. Red lines show the UAV motion estimation before convex pose-graph optimisation and green lines illustrate the corrected estimates after loop closure detection. (a) shows results when the vehicle closes the first loop. (b) and (c) depict results after detecting the second and the third loop closures. (d) shows a sample of loop closure frames.



(a) Loop closure 1



(b) Loop closure 2



(c) Loop closure 3

Figure 6: Convex pose-graph optimisation results on outdoor experiment. Red lines show the estimated trajectories before convex pose-graph optimisation. Green lines illustrate their corrections after loop closures detection.

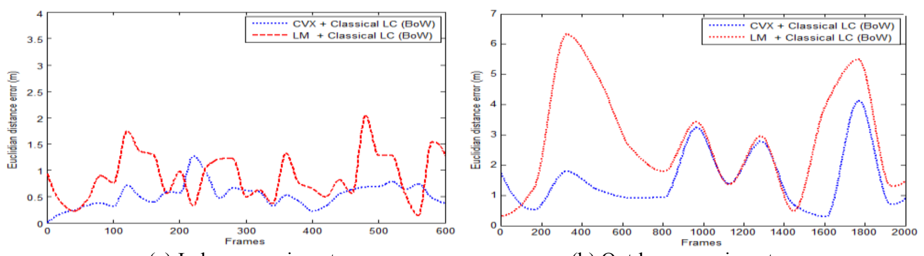
performed on all frames included in this particular loop as shown in Figure 5.a to Figure 6.c. Clearly, robust convex pose-graph optimisation is accurately and effectively able to correct any drift during navigation in both indoor and outdoor environments. These figures confirm that our solution is suitable to multiple loop closures as well.

7.1 Assessment of pose-graph optimisation with BoW method (Scenario 1)

In this experiment we investigate the performance of our convex pose-graph optimisation (CVX) against classical method using Levenberg-Marquardt (LM) where for both methods classical Bag-of-Word (BoW) is employed. This experiment is conducted on indoor and outdoor data as shown in Figure 7. Results obtained using convex optimisation look significantly better and for both environments. This accuracy of the proposed convex optimisation is in accordance with the theory as the estimates should be globally optimal. Due to its landscape nature, higher errors are still noticed after pose-graph optimisation in outdoor experiment especially for the LM method.

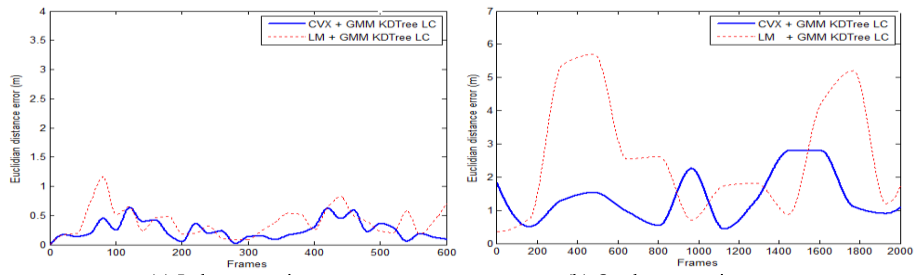
7.2 Assessment of pose-graph optimisation with GMM/KD-Tree method (Scenario 2)

Similarly to the previous setup, in this experiment we use our loop-closure detection method based on Gaussian Mixture Modelling (GMM) with Bayesian theory and KD-Tree structure to test our



(a) Indoor experiment (b) Outdoor experiment

Figure 7: Comparison of convex pose-graph optimisation to classical LM method using the same classical loop-closure detection technique (BoW).



(a) Indoor experiment (b) Outdoor experiment

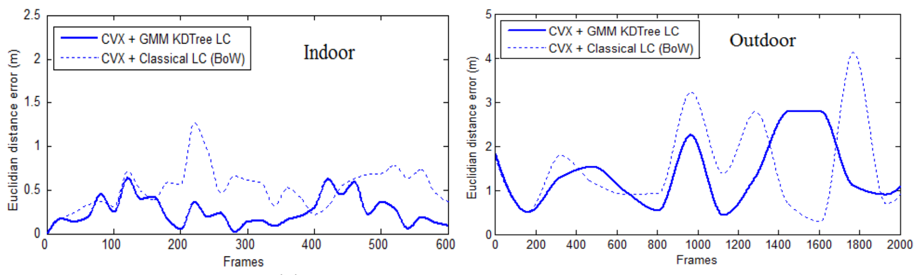
Figure 8: Comparison of convex pose-graph optimisation to classical LM method using the same GMM/KD-Tree method for loop closure detection.

convex pose-graph optimisation. Results are shown in Figure 8. In this scenario, we want to assess the performance of our convex pose-graph optimisation with our method of loop-closure based on GMM/KD-Tree. Similarly to the previous experiment, convex pose-graph optimisation outperforms traditional LM optimisation technique. The average error in indoor environment with convex method does not exceed 0.28 metres while it is about 0.59 metres for LM method. More significant improvement can be seen in outdoor experiment where average of Euclidean distance errors have dropped from 3.35 metres when using for LM method to just 1.15 metres with convex optimisation.

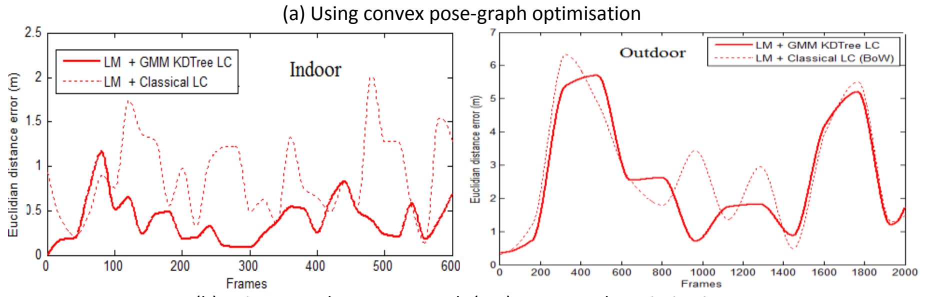
It can be seen, from the Euclidian distance errors, and regardless the employed loop-closure technique, convex optimisation approach is more accurate in all environments than classical techniques using Levenberg-Marquardt approach. Indeed, convex optimisation with L∞ norm has shown its ability to ensure the global minima in recovering the motion parameters in comparison to iterative least square based methods where a predefined termination criterion is set which favours convergence to local minima.

7.3 Effect of loop-closure detection method

We have assessed so far the performance of convex pose-graph optimisation regardless the employed loop-closure technique. We want to investigate, in this section, to what extent using a particular loop-closure detection method can affect the global performances of the solution. To do that,



(a) Using convex pose-graph optimisation



(b) Using Levenberg-Marquardt (LM) pose-graph optimisation

Figure 9: Effects of loop-closure detection method on the global trajectories estimates. (a) and (b) compare GMM/KD-Tree (solid line) to classical BoW technique (dotted line). (a) Convex optimisation is used (blue lines). (b) Levenberg-Marquardt (LM) technique is used.

let us consider Figure 9 where both pose-graph optimisation techniques (convex optimisation and Levenberg-Marquardt optimisation) are tested under the two loop-closure detection techniques (GMM/KD-Tree and classical BoW). The top row in this figure shows Euclidian distance errors using convex pose-graph optimisation with GMM/KD-Tree (solid blue lines) and classical BoW (dotted blue lines) in both indoors and outdoors environments. These plots show that camera positions can be estimated well with GMM/KD-Tree for loop-closure detection even this improvement is not very considerable in outdoor environment. Bottom row in the same figure illustrates the effects of GMM/KD-Tree for loop-closure with Levenberg-Marquardt (LM) pose-graph optimisation. Similar pattern is noticed here as well.

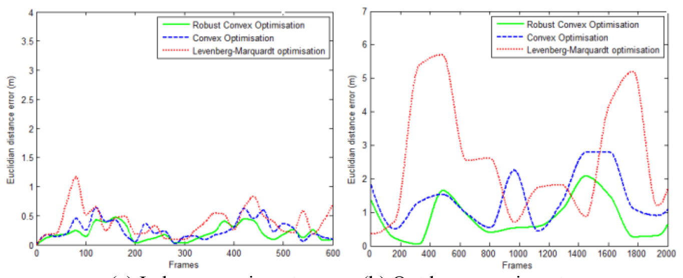
From these results, we learn that in addition to the convex optimisation properties, in which solutions are guaranteed to be globally optimal, using robust and accurate methods for loop-closure detection improves remarkably the global performances of the solution.

7.4 Robust Convex pose-graph optimisation

In this section, we investigate the performance of our solution when uncertainties are incorporated as given in (22). These uncertainties are originally from image feature's imperfect positions, due to deterministic perturbations, and then propagated through multiple views geometry algorithms to the relative translations.

Figure 10 illustrates the outcome of the proposed solution when uncertainties are included. Even though the improvement in indoor experiment is only reasonable (the average error of 0.20 metres when including uncertainties against 0.28 metres using normal convex optimisation), it demonstrates the robustness of the algorithm (Figure 10.a solid green line). This improvement can be remarkably seen in outdoor experiment in which the average error has dropped even more when uncertainties are taken into consideration reaching a value of 0.72 metres (was 1.15 metres when using normal convex optimisation) (Figure 10.b solid green line).

From these results, we learn that taking the uncertainties into consideration in the proposed method, would lead to robust and more accurate estimations than the normal convex optimisation as expected since it encodes large intervals in its



(a) Indoor experiment (b) Outdoor experiment

Figure 10: Robust convex pose-graph optimisation. Solid green lines plot Euclidian distance errors after performing robust convex pose-graph optimisation. Dashed blue line shows errors from normal convex optimisation and dotted red lines shows results from classical LM method.

optimisation and models well the uncertainties.

8. Conclusions

In this work, we presented a new robust convex pose-graph optimisation solution for UAVs monocular motion estimation systems. Through a variety of experimental validations which are conducted on real-world data from indoor and outdoor environments and comparison to state-of-the-art methods, using convex optimisation in pose-graph problems has proven its efficiency in motion estimation correction after loop-closures detections.

Even more, including uncertainty estimations, based on SIFT derivative approach and their propagation through multiple views geometry algorithms have contributed in the improvement of the global motion estimation. The unavoidable drift in motion estimation due to imperfect computational tools and to inherent inaccuracy of the devices would be robustly and accurately corrected using robust convex optimisation. In addition, the proposed solution is suitable to multiple loop-closures circumstances.

Furthermore, a solution based on the combination of Gaussian Mixture Model (GMM) with the KD-Tree data structure has confirmed its capability of efficiently detecting loop-closures leading to better motion estimates.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

1. Omead A, Kanade T, Fujita K. Autonomous Systems A visual odometer for autonomous helicopter flight. Rob Auton Syst. 1999;28(02):185-193.
2. Corke P, Sikka P, Roberts J. Height Estimation for an Autonomous Helicopter. Lect Notes Contr Inform ScisNotes Contr Inform Scis. 2001;271:101-110.
3. Saripalli S, Montgomery JF, Sukhatme GS. Visually-Guided Landing of an Unmanned Aerial Vehicle. IEEE Trans Robot Autom. 2003;19(3):371-380.
4. Levin A, Szeliski R. Visual odometry and map correlation. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on.Vol 1.; 2004:I-611-I-618 Vol.1. doi:10.1109/CVPR.2004.1315088.

5. Hartley RI, Zisserman A. Multiple View Geometry in Computer Vision. Second. Cambridge University Press, ISBN: 0521540518; 2004.

6. Uyttendaele M, Criminisi A, Kang SB, Winder S, Hartley R, Szeliski R. High-quality Image-based Interactive Exploration. IEEE Comput Graph Appl. 2004;24(3):52–63.

7. Nemra A, Aouf N. Robust cooperative UAV Visual SLAM. 2010 IEEE 9th Int Conf Cyberntic Intell Syst. 2010:1-6. doi:10.1109/UKRICIS.2010.5898125.

8. Li X, Aouf N, Nemra A. 3D mapping based VSLAM for UAVs. In: 2012 20th Mediterranean Conference on Control & Automation (MED). Ieee; 2012:348-352. doi:10.1109/MED.2012.6265662.

9. Davison AJ. Real-time simultaneous localisation and mapping with a single camera. In: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on.; 2003:1403-1410 vol.2. doi:10.1109/ICCV.2003.1238654.

10. Nützi G, Weiss S, Scaramuzza D, Siegwart R. Fusion of IMU and Vision for Absolute Scale Estimation in Monocular SLAM. J Intell Robot Syst. 2011;61(1-4):287-299. doi:10.1007/s10846-010-9490-z.

11. Esteban I, Dorst L, Dijk J. Closed form solution for the scale ambiguity problem in monocular visual odometry. In: Proceedings of the Third International Conference on Intelligent Robotics and Applications - Volume Part I. ICIRA'10. Berlin, Heidelberg: Springer-Verlag; 2010:665-679.

12. Boulekchour M, Aouf N. L∞ Norm Based Solution for Visual Odometry. In: CAIP13.; 2013:II:185-192.

13. Sivic J, Zisserman A. Video Google: a text retrieval approach to object matching in videos. In: Proceedings Ninth IEEE International Conference on Computer Vision. Ieee; 2003:1470-1477 vol.2. doi:10.1109/ICCV.2003.1238663.

14. Zhang H. BoRF: Loop-closure detection with scale invariant visual features. In: 2011 IEEE International Conference on Robotics and Automation. Ieee; 2011:3125-3130. doi:10.1109/ICRA.2011.5980273.

15. Olson E, Leonard J, Teller S. Fast iterative alignment of pose graphs with poor initial estimates. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006. Ieee; 2006:2262-2269. doi:10.1109/ROBOT.2006.1642040.

16. F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. Auton Robot. 1997;4:333–349.

17. Grisetti, G., Kummerle, R., Stachniss, C., & Burgard W. A tutorial on graph-based SLAM. Intell Transp Syst Mag IEEE. 2010:31-43.

18. Gutmann J, Konolige K, International SRI, Park M. Incremental Mapping of Large Cyclic Environments. In: IEEE Int. Symp. Computational Intelligence in (CIRA).; 1999.

19. Boyd S, Vandenberghe L. Convex Optimization. New York, NY, USA: Cambridge University Press; 2004.

20. Kahl F. Multiple view geometry and the L infin;-norm. In: Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on.Vol 2.; 2005:1002-1009 Vol. 2. doi:10.1109/ICCV.2005.163.

21. Ke Q, Kanade T. Quasiconvex Optimization for Robust Geometric Reconstruction. Pattern Anal Mach Intell IEEE Trans. 2007;29(10):1834-1847. doi:10.1109/TPAMI.2007.1083.

22. Kanazawa Y, Kanatani K. Do we really have to consider covariance matrices for image features? Proc Eighth IEEE Int Conf Comput Vision ICCV 2001. 2001;2:301-306.

23. Zeisl B, Georgel PF, Schweiger F, Steinbach E, Navab N. Estimation of Location Uncertainty for Scale Invariant Feature Points. Procedings Br Mach Vis Conf 2009. 2009:57.1-57.12.

24. Sur F, Noury N, Berger M. Computing the uncertainty of the 8 point algorithm for fundamental matrix estimation. 19th Br Mach Vis Conf …. 2008.

25. Löfberg J. Automatic robust convex programming. Optim methods Softw. 2012;27(00):115-129.

26. Boni O, Ben-Tal A, Nemirovski A. Robust solutions to conic quadratic problems and their applications. Optim Eng. 2007;9(1):1-18.

27. Lowe DG. Distinctive Image Features from Scale-Invariant Keypoints. Int J Comput Vis. 2004;60(2):91-110.

28. Weijer J Van De, Schmid C. Coloring Local Feature Extraction. In: ECCV.; 2006:334-348, Graz.

29. Silpa-anan C, Hartley R. Optimised KD -trees for fast image descriptor matching. In: IEEE Conference on Computer Vision and Pattern Recognition.; 2008.

30. Muja M, Lowe DG. Fast Matching of Binary Features. In: 2012 Ninth Conference on Computer and Robot Vision. Ieee; 2012:404-410. doi:10.1109/CRV.2012.60.

31. Liu Yang and Zhang Hong. Indexing visual features: Real-time loop closure detection using a tree structure. In: 2012 IEEE International Conference on Robotics and Automation. Ieee; 2012:3613-3618. doi:10.1109/ICRA.2012.6224741.

32. Strasdat H, Davison AJ. Scale Drift-Aware Large Scale Monocular SLAM. Robot Sci Syst. 2010;2(3).

33. M.S.Lobo, L. Vandenberghe SB and HL. Applications of second-order cone programming. Linear Algebra Appl. 1998;284:193-228.

34. Sturm JF. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. 1998.