Noname manuscript No.
(will be inserted by the editor)

An Efficient RANSAC Hypothesis Evaluation using Sufficient Statistics for RGB-D Pose Estimation

Ilankaikone Senthooran · Manzur Murshed · Jan Carlo Barca · Joarder Kamruzzaman · Hoam Chung

Received: date / Accepted: date

Abstract Achieving autonomous flight in GPS-denied environments begins with pose estimation in three-dimensional space, and this is much more challenging in an MAV in a swarm robotic system due to limited computational resources. In vision-based pose estimation, outlier detection is the most time-consuming step. This usually involves a RANSAC procedure using the reprojection-error method for hypothesis evaluation. Realignment-based hypothesis evaluation method is observed to be more accurate, but the considerably slower speed makes it unsuitable for robots with limited resources. We use sufficient statistics of least-squares minimisation to speed up this process. The additive nature of these sufficient statistics makes it possible to compute pose estimates in each evaluation by reusing previously computed statistics. Thus estimates need not be calculated from scratch each time. The proposed method is tested on standard RANSAC, Preemptive RANSAC and R-RANSAC using benchmark datasets. The results show that the use of sufficient statistics speeds up the outlier detection process with realignment hypothesis evaluation for all RANSAC variants, achieving an execution speed of up to 6.72 times.

Ilankaikone Senthooran
Faculty of Information Technology, Monash University, Caulfield East VIC 3145, Australia

E-mail: ilankaikone.senthooran@monash.edu

Manzur Murshed
Faculty of Science and Technology, Federation University Australia, Churchill VIC 3842, Australia

Jan Carlo Barca
Faculty of Information Technology, Monash University, Clayton VIC 3800, Australia

Joarder Kamruzzaman
Faculty of Science and Technology, Federation University Australia, Churchill VIC 3842, Australia

Hoam Chung
Department of Mechanical and Aerospace Engineering, Monash University, Clayton VIC 3800, Australia

1 Introduction

Swarm robotics has gained popularity due to its numerous potential applications. Particularly, autonomous swarms of Micro Aerial Vehicles (MAVs) have numerous indoor applications such as surveillance, monitoring, collapsed building exploration and aiding in disaster relief operations. A swarm robotic system comprises individuals that are less capable and dispensable, enabling robustness and graceful degradation of performance in the case of individual failure. Thus in a swarm of MAVs, individuals are very limited in resources compared to a single specialised MAV with many sensors and high computational resources.

To automate the flight of an MAV in GPS-denied environments, one needs to know its location with respect to the environment. For an aerial robot this involves estimating position and orientation in three dimensional (3-D) space, known as pose estimation. The computational speed of pose estimation is more crucial in aerial robots than in ground robots since the performance in pose estimation is directly linked to the stability of a vehicle. It is obvious that the problem becomes much more challenging in limited resourced MAVs because of their low processing power and memory.

Various exteroceptive sensors such as laser range finders (Bachrach et al. 2011; Chowdhary et al. 2012; Bry et al. 2012), monocular cameras (Weiss et al. 2012; Wang et al. 2012) and stereo cameras (García Carrillo et al. 2012; Voigt et al. 2011) have been used effectively for indoor navigation in MAVs. In recent years, RGB-D visual pose estimation methods have become popular among the MAV research community for indoor applications (Stowers et al. 2011; Bachrach et al. 2012; Scherer and Zell 2013; Valenti et al. 2014a). The main reason behind this trend is that RGB-D cameras are independent modules providing rich 3-D information about its environment (Khoshelham and Elberink 2012), which eliminates the need for any further processing to calculate depth information, such as triangulation in stereo cameras. In addition, their small size, light weight and economical price make them ideal for use in low-cost MAVs.

Some of the first works that used RGB-D cameras for real-time robot control did not completely rely on the RGB-D information for computing full pose estimates. For example, RGB-D information was used for estimating and controlling the flying height of an MAV (Stowers et al. 2011) and for indoor exploration where pose estimation was done using a laser range finder (Shen et al. 2011). Huang et al. (Bachrach et al. 2012) was the earliest notable case of using an RGB-D camera for onboard visual odometry. The more recent works (Scherer and Zell 2013; Valenti et al. 2014a) achieved fully onboard localisation and mapping using RGB-D camera. However, all the afore-mentioned visual odometry methods (Bachrach et al. 2012; Scherer and Zell 2013; Valenti et al. 2014a) used single board computers that are relatively high in processing power and cost.

A recent comparison study (Fang and Scherer 2014) provided a detailed analysis and experimental comparison of several state-of-the-art real-time odometry estimation methods that use RGB-D cameras, focusing on algorithms suitable for limited-resourced MAVs. They categorised the existing visual odometry methods into three groups according to sensor data types as (1) image-based, (2) depth-based, and (3) both image- and depth-based. Image-based methods are further subdivided as (a) sparse visual feature based methods, (b) sparse visual feature based methods combining depth data, and (c) dense feature based methods. For experimental comparison they used several existing real-time RGB-D visual odometry methods that fall into different categories mentioned above. The methods included, among others, Libviso2 (Geiger et al. 2011), Fovis (Bachrach et al. 2012), DVO (Kerl et al. 2013) and FastICP (Pomerleau et al. 2013). It should be noted that among the algorithms considered, only Fovis (Bachrach et al. 2012), which is a sparse visual feature based method combining depth data, was originally implemented on an MAV. The results from the experiments in (Sturm et al. 2012) using benchmark RGB-D dataset as well as self-recorded datasets on a laptop computer show that Fovis (Bachrach et al. 2012) is the fastest (least CPU-intensive) method.

However, in our preliminary experiments using a Beaglebone Black, we discovered that even a sparse visual feature based method, very similar to Fovis (Bachrach et al. 2012), produces pose estimation at a rate less than 1Hz on our system (see section 5.2 for system description). This was not fast enough for real-time navigation of our MAV swarm. Our analysis showed that the outlier detection using RAndom SAmple Consensus (RANSAC) was the most time consuming step in this visual pose estimation process.

This paper presents a method for speeding up RANSAC in the problem of relative-pose estimation using 3-D point correspondences. The key idea of the approach lies in the observation that the computation of relative pose depends on a number of ‘sufficient statistics,’ that is, a number of functions on the input data. These sufficient statistics have been previously proposed in bioinformatics (Konagurthu et al. 2014) for efficiently aligning protein structures. Even though pose estimation involves a similar alignment process for computing the transformation between camera poses, the 3-D feature point sets contain noisy data that are normally pruned out by the RANSAC procedure.

RANSAC involves hypothesis evaluation where the transformation that aligns 3D point pairs is computed numerous times to eliminate the outliers. This evaluation is typically done by computing the re-projection error of all matches. An alternative method uses realignment based hypothesis evaluation where transformations are calculated for each of the matches (Li et al. 2013). By nature realignment based method is more accurate in rejecting outliers than the re-projection error based hypothesis evaluation as extend the measurement errors are reduced during realignment. However, the realignment based method is slower due to re-fitting points at each iteration.

The use of the sufficient statistics of least-squares minimisation makes it possible to speed-up this more accurate hypothesis evaluation by computing these statistics incrementally in each RANSAC iteration, not from scratch. The proposed method is capable of improving the computational performance of different RANSAC variants namely, standard RANSAC, Preemptive RANSAC and R-RANSAC.

This paper is a comprehensive and extended version of our previous conference paper presented at IROS 2015 (Senthooran et al. 2015). In our previous work, we only applied our method to standard RANSAC and presented the results from only one benchmark dataset sequence. In addition to presenting the concept introduced in the previous work with more detail, this paper makes the following contributions.

- Time complexity analysis of proposed algorithm.
- Modified algorithms for other RANSAC variants, namely Preemptive RANSAC and Randomised RANSAC, by applying the proposed method.
- Results from an extended set of experiments conducted on five additional sequences from different benchmark datasets.
- Experimental comparison of the performance of the proposed hypothesis evaluation method when used in different RANSAC variants.

The rest of this paper is organised as follows. Section 2 introduces preliminaries including how pose estimation from RGB-D sensor information involves 3-D feature-based alignment which is essentially a least-squares minimisation problem and the use of RANSAC due to uncertainty of feature matches. Section 3 gives a detailed description of the proposed method. In this section we explain how we use sufficient statistics in the RANSAC-based outlier detection for the case of standard RANSAC (3.1) as well as other variants (3.2). In Section 4 we analyse the time complexity of our proposed method. We present the experimental results in Section 5. This includes evaluations of accuracy and computation time of our pose estimation method using benchmark datasets. Finally, concluding remarks and future work are mentioned in Section 6.

2 Preliminaries

In this section some preliminaries regarding the problem in hand is elaborated.

2.1 Pose estimation from RGB-D information

Pose estimation using RGB-D information is a case of 3-D feature-based alignment, which is the problem of estimating the motion between two or more sets of matched 3-D points. This process starts by extracting features from RGB images and then tracking matching features across different image frames. Using the corresponding depth information and camera parameters, these two-dimensional feature point pairs are converted into 3-D feature point pairs. Assuming that the environment is static, the geometric transformation in 3-D space (rotation R and translation T) between two poses directly corresponds to the camera movement. The camera pose can thus be determined by computing the inverse of the geometric transformation that maps the 3-D features in one image to the other. That is,

$$\Delta R = R^\top,\ \Delta T = -R^\top T, \tag{1}$$

where ΔR and ΔT are the relative rotation and translation of the current frame to the camera's previous body frame, respectively.

2.2 3-D feature-based alignment

Let us denote the two 3-D feature sets obtained from two different RGB-D camera frames as $U = \{u_1, u_2, \ldots, u_n\}$ and $V = \{v_1, v_2, \ldots, v_n\}$, where n is the number of features in each set and $u_i \in \mathbb{R}^3$ corresponds to $v_i \in \mathbb{R}^3$, $i = 1, 2, \ldots, n$. The objective is to find the rotation $R \in \mathrm{SO}(3)$ and translation $t \in \mathbb{R}^3$ that aligns the point sets U and V while minimising the alignment error defined as

$$\min_{R,t} \sum_{i=1}^{n} \|Rv_i + T - u_i\|^2. \tag{2}$$

By evaluating the derivative of (2), it can be shown that the geometric centres of the two feature sets $c(U)$ and $c(V)$ can be used to estimate the value of t at the optimum as $T = c(U) - Rc(V)$ (Szeliski 2011). Note that $c(X) \triangleq \sum x_i/n$ is the geometric centre of arbitary set of points $X = \{x_1, x_2, \ldots, x_n\}$. We are now left with the problem of estimating the rotation between two sets of points $u'_i = u_i - c(U)$ and $v'_i = v_i - c(V)$ that are both centered at the origin. This yields an objective function that is independent of T

$$\min_{R} \sum_{i=1}^{n} \|Rv'_i - u'_i\|^2. \tag{3}$$

One commonly used technique to solve this optimisation problem involves computing the singular value decomposition (SVD) of the correlation matrix in $\mathbb{R}^{3\times 3}$ (Szeliski 2011)

$$C = \sum_i u'_i v_i'^{\mathsf{T}}. \tag{4}$$

Another technique is the *absolute orientation algorithm* (Horn 1987) for estimating the unit quaternion corresponding to the rotation matrix R, which involves forming a matrix in $\mathbb{R}^{4\times 4}$ from the entries of C and then finding the eigenvector associated with its maximum eigenvalue. A similar approach for comparing molecular structures is presented in (Kearsley 1989). Here the 3D alignment problem is constrained since the transformation only involves rotation and translation, which is the same in our case.

Even though the regular least squares method can account for Gaussian noise, more robust methods are required when there are many outliers among the correspondences (Szeliski 2011). This is particularly true in our case since the 3-D point sets may contain mismatched feature point pairs in addition to inaccuracies of sensor data. Due to such uncertainties in the feature correspondences, it is necessary to use iterative methods such as Iterative Closest Points (ICP) (Besl and McKay 1992) and RAndom SAmple Consensus (RANSAC) (Fischler and Bolles 1981) for computing optimal estimates of relative rotation and translation.

RANSAC is a non-deterministic algorithm for fitting a model to a dataset contaminated with outliers. It is composed of the two steps, *hypothesise* and *evaluate* that are repeated in an iterative manner. This process includes generating several hypotheses using randomly selected minimal sample sets and then evaluating each of these hypotheses on the entire dataset to build up consensus sets of inliers. To ensure that the random sampling has a good chance of finding a true set of inliers, a sufficient number of trials must be tried, after which the hypothesis with the highest number of inliers can be taken as the final solution. Since RANSAC is capable of handling a significant percentage of gross errors, it is ideally suited for our application due to error-prone range data and feature detectors/trackers.

When determining the optimal alignment of our two feature point sets, the RANSAC process starts by selecting, at random, a subset of 3-D point pairs, which is then used to compute an initial estimate for the transformation ($\hat{R}$ and $\hat{T}$). Note here that the minimal sample set size is three since only three point correspondences are necessary to obtain an alignment. The estimated transformation is then tested against all the point pairs excluding the subset taken as the sample (termed 'test set') and those within a given error tolerance are classified as inliers and added to the consensus set.

The widely used method for hypothesis evaluation in 3-D feature-based alignment is to calculate the 're-projection error' or 'residual' for each feature point pair in the test set as

$$\epsilon_i = |\hat{R}v'_i - u'_i|, \; i = 1, 2, \ldots, n \tag{5}$$

where $\hat{R}v'_i$'s are the estimated (mapped) locations and u'_i's are the sensed (detected) feature point locations. The point pairs whose re-projection errors are within an acceptable limit are identified as inliers. The steps involved are outlined in Algorithm 1 and will be referred to as HT1 (Hypothesis Testing 1) throughout this paper.

An alternative way to evaluate the initial estimate for the transformation would be to add each point pair in the test set to the sample set individually and recompute the transformation as outlined in Algorithm 2. The point pairs for which the difference between the initial and the new transformation estimates are within a certain acceptable limit are identified as inliers. This evaluation method, which is referred to as HT2 (Hypothesis Testing 2) in this paper, is indeed slower due to all the re-computations of the transformation. However, it is not difficult to imagine that this method would be more accurate since the realignment step may reduce the effect of measurement errors in the sample set on the identification of inliers. Therefore, if we can computationally accelerate HT2, then we would achieve better accuracy without compromising the efficiency.

The method we propose is to reduce the time taken for the re-computations of the transformation by using a set of sufficient statistics for the 3D point alignment problem derived in (Konagurthu et al. 2014). In this previous work in bio-informatics, this set of sufficient statistics has been proposed for efficiently aligning protein structures. In essence the problem of protein structure alignment is similar to 3-D feature alignment, as they both involve computing the transformation that would superimpose two sets of points in 3-D space onto each other so that the corresponding points are as close as possible. Usually, in the case of protein structure alignment, these points will be the locations of the amino acids, which are approximated by the locations of α Carbon ($C\alpha$) atoms along each structure's backbone. Aligning two such structures involve finding the rotation and translation which minimizes the distance (RMSD) between the equivalent atoms, which can be solved using the least squares method. The main difference of our alignment problem is the presence of many outliers in the 3-D point correspondences, which need to be pruned out to in order to arrive at an accurate solution. The next section explains how the sufficient statistics for 3D point alignment is incorporated into our RANSAC-based outlier removal routine.

3 Applying Sufficient Statistics to RANSAC Hypothesis Evaluation

Sufficient statistics (Hogg and Craig 1994) are essentially a set of statistics that summarises all of the information in a sample about a certain parameter. A set of sufficient statistics with respect to the least squares alignment was derived in (Konagurthu et al. 2014), and furthermore, these statistics were demonstrated to be additive. Therefore, by using these statistics in the RANSAC process, computing transformations of the

Algorithm 1 HT1: Re-projection error based hypothesis evaluation

```
Input: Initial transformation estimation $(R, T)$ that best aligns point pairs in sample set $U_s$, $V_s$ and test sets $U = \{u_i\}$ ,$V = \{v_i\}$, $i = 1, 2, \ldots, n$
Output: Inlier point pairs
Begin:
  for each corresponding points $(u_i,\ v_i)$ in $U$, $V$ do
      Transform point $v_i$ into point $\hat{v}_i$ by applying transformation $(R, T)$
      Calculate euclidean distance $\epsilon_i$ between $\hat{v}_i$ and $u_i$
      If $\epsilon_i$ is within the acceptable limit consider point pair $(u_i$,$v_i)$ as an inlier.
  end for
```

Algorithm 2 HT2: Realignment based hypothesis evaluation

```
Input: Initial transformation estimation $(R, T)$ that best aligns point pairs in sample set $U_s = \{u_i\}$ ,$V_s = \{v_i\}$, $i = 1, 2, \ldots, m$, corresponding alignment residual error $\epsilon_s = \sum_{i=1}^{m} \|Rv_i + T - u_i\|^2$ and test sets $U = \{u_i\}$ ,$V = \{v_i\}$, $i = 1, 2, \ldots, n$
Output: Inlier point pairs
Begin:
  for each corresponding points $(u_i,\ v_i)$ in $U$, $V$ do
      Calculate transformation $(R_i, T_i)$ that best aligns points in $V_s \cup v_i$ onto points in $U_s \cup u_i$
      Calculate the corresponding alignment residual error $\epsilon_i$
      If $\epsilon_i - \epsilon_s$ is within the acceptable limit consider point pair $(u_i$,$v_i)$ as an inlier.
  end for
```

sample set plus each additional point pair can be performed in constant time by reusing the previous solutions of the sample set. This process is much faster than recomputing transformations from scratch. Below we re-derive the set of sufficient statistics of (Konagurthu et al. 2014) with respect to our problem.

Rigid-body alignment is a general regression problem and we assume that it produces error terms, $\epsilon_i = \hat{R}v'_i - u'_i$ that are normally distributed as $\mathcal{N}(0, \sigma)$, which is minimised by the solution of (3). Thus, the 'likelihood' of the normally distributed error terms after alignment is given as

$$f(\epsilon_1, \epsilon_2, \ldots, \epsilon_n|\sigma) = (2\pi\sigma^2)^{-\dfrac{n}{2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\|\epsilon_i\|^2\right). \tag{6}$$

By examining the decomposition of $\|\epsilon_i\|^2$, the summation term in (6) can be reduced to a form containing a set of statistics which does not take into account its data explicitly. As these statistics are also sufficient to estimate σ, they form a set of sufficient statistics of the least-squares minimisation problem. This set consists of 24 distinct statistics as described below. First, let us define

$$s_m^{ij} \triangleq u'_{ij} - v'_{ij},\ s_p^{ij} = u'_{ij} + v'_{ij},\ i = 1, 2, \ldots, n,\ j = x, y, z,$$

where u'_{ix} indicates the x-axis component of vector u'_i and so on. From these, we can define the set of sufficient statistics as

$$\Omega = \left\{\text{for } j, k \in \{x, y, z\}, \sum_{i=1}^{n} s_m^{ij}, \sum_{i=1}^{n} s_p^{ij}, \sum_{i=1}^{n} s_m^{ij} s_m^{ik}, \sum_{i=1, j\neq k}^{n} s_m^{ij} s_p^{ik}, \sum_{i=1}^{n} s_p^{ij} s_p^{ik}\right\}. \tag{7}$$

Let us consider two pairs of corresponding feature sets, (U'_s, V'_s) and (U'_t, V'_t). Let Ω_1 and Ω_2 be the sufficient statistics of the alignment of the first and the second pair respectively. As shown through Lemmas 1–3 and Corollaries 1–4 in (Konagurthu et al. 2014), sufficient statistics of least-squares minimisation are additive. Therefore, we can use Ω_1 and Ω_2 to derive a set of sufficient statistics Ω' of the alignment of U' with V', where $U' = U'_s \cup U'_t$ and $V' = V'_s \cup V'_t$.

The updated sufficient statistics Ω' is now used with a method proposed in (Kearsley 1989) to recompute rotation R. This method transforms the least-squares minimisation problem to an eigenvalue problem of the form $Q(\Omega)q = \lambda q$, where $Q(\Omega) \in \mathbb{R}^{4\times 4}$ is a symmetric matrix that is defined as

$$Q = \begin{pmatrix} \sum(x_{im}^2 + y_{im}^2 + z_{im}^2) & \sum(y_{ip}z_{im} - y_{im}z_{ip}) & \sum(x_{im}z_{ip} - x_{ip}z_{im}) & \sum(x_{ip}y_{im} - x_{im}y_{ip}) \\ \sum(y_{ip}z_{im} - y_{im}z_{ip}) & \sum(x_{im}^2 + y_{ip}^2 + z_{ip}^2) & \sum(x_{im}y_{im} - x_{ip}y_{ip}) & \sum(x_{im}z_{im} - x_{ip}y_{ip}) \\ \sum(x_{im}z_{ip} - x_{ip}z_{im}) & \sum(x_{im}y_{im} - x_{ip}y_{ip}) & \sum(x_{ip}^2 + y_{im}^2 + z_{ip}^2) & \sum(y_{im}z_{im} - y_{ip}z_{ip}) \\ \sum(x_{ip}y_{im} - x_{im}y_{ip}) & \sum(x_{im}z_{im} - x_{ip}z_{ip}) & \sum(y_{im}z_{im} - y_{ip}z_{ip}) & \sum(x_{ip}^2 + y_{ip}^2 + z_{im}^2) \end{pmatrix}, \tag{8}$$

where $x_{im} = u'_{ix} - v'_{ix}$, $y_{im} = u'_{iy} - v'_{iy}$, $z_{im} = u'_{iz} - v'_{iz}$, $x_{ip} = u'_{ix} + v'_{ix}$, $y_{im} = u'_{iy} + v'_{iy}$, $z_{im} = u'_{iz} + v'_{iz}$,

$$q = (q_1, q_2, q_3, q_4)^T = (\cos\frac{\theta}{2}, m_x \sin\frac{\theta}{2}, m_y \sin\frac{\theta}{2}, m_z \sin\frac{\theta}{2})^T$$

is the rotation R in quaternion form, and $\sum$ denotes the summation over terms $i = 1, 2, \ldots, n$. The eigenvector of the minimum eigenvalue gives the optimal rotation which minimises (3). Each element of the matrix $Q(\Omega)$ can be computed by combining a subset of the sufficient statistics Ω given in (7). Since these statistics are additive, matrix Q of the sample set with an additional pair of points can be constructed by finding the elements of Q for the additional pair and adding it to the previously computed Q of the sample set. This eliminates the need of recomputing Q from the entire set. Once Q is updated using sufficient statistics, it is diagonalised to find the corresponding rotation.

3.1 Applying to standard RANSAC

As explained above, adopting sufficient statistics in RANSAC-based pose estimation can drastically increase the processing speed by performing least-square minimisation using the previous partial solutions. Our pose estimation algorithm that uses the standard RANSAC procedure with HT2 and SS (Sufficient Statistics) is shown in Algorithm 3. It begins by randomly selecting a pair of sample sets (U_s, V_s) from (U, V), in which each element in U_s has a corresponding element in V_s (line 5, subroutine randomMinimalSampleSet). The remaining element pairs then constitute the test pair of sets (U_t, V_t) where $U_t = U \setminus U_s$ and $V_t = V \setminus V_s$. The sufficient statistics in (7) are computed for the sample set. These statistics are denoted by Ω_s (line 6, subroutine suffStats). Then Ω_s is used to compute the relative pose estimation p_s for the sample set by constructing the corresponding Q matrix (line 7, subroutine relativePose). Next, a point pair (u_i, v_i) is picked and the sufficient statistics of that pair Ω_i is computed (line 10). We then compute the relative pose corresponding to the sample set with the additional point pair p_i from its sufficient statistics Ω' which is computed using Ω_s and Ω_i (line 11, subroutine updateSuffStats). If the root mean square error (rmse) difference of p_i and p_s is within an acceptable bound, the additional point pair is considered as an inlier and added to the consensus set (U_c, V_c) (line 12–15). Sufficient statistics for the pair Ω_i is added to the sufficient statistics of the consensus set Ω_c (line 16). At the end the iteration, precomputed values p_s, Ω_s and Ω_c are used to compute the corresponding pose estimation p_{sc} (subroutine updateRelativePose) for $(U_s, V_s) \cup (U_c, V_c)$ (line 20, 21). Finally, the hypothesis with less error E and larger consensus set size is selected (line 22–27).

Algorithm 3 Standard RANSAC with HT2 and SS

```
Input: Two sets of corresponding 3D points $U = \{u_i\}$, $V = \{v_i\}$, $i = 1, 2, \ldots, n$
Output: Rotation $R$ and translation $T$ (pose $p*$).
Begin:
 1: $l \leftarrow 0$
 2: $C_{max} \leftarrow 0$
 3: $E_{min} \leftarrow$allowable_error
 4: while $l <$ max_iterations do
 5:     $U_s, V_s \leftarrow$ randomMinimalSampleSet($U, V$)
 6:     $\Omega_s \leftarrow$ suffStats($U_s, V_s$)
 7:     $p_s \leftarrow$ relativePose($U_s, V_s$)
 8:     $e_s \leftarrow$ rmse($(U_s, V_s), p_s$)
 9:     for each $u_i \in U_t = U \setminus U_s$ and $v_i \in V_t = V \setminus V_s$ do
10:         $\Omega_i \leftarrow$ suffStats($u_i, v_i$)
11:         $\Omega' \leftarrow$ updateSuffStats($\Omega_s, \Omega_i$)
12:         $e_i \leftarrow$ rmse($\Omega', p_s$) $- e_s$
13:         if $|e_i| < threshold$ then
14:             add $u_i$, $v_i$ to consensus set $U_c$, $V_c$
15:             $\Omega_c \leftarrow$ updateSuffStats($\Omega_c, \Omega_i$)
16:         end if
17:     end for
18:     $\Omega_{sc} \leftarrow$ updateSuffStats($\Omega_s, \Omega_c$)
19:     $p_{sc} \leftarrow$ updateRelativePose($\Omega_{sc}, p_s$)
20:     $E \leftarrow$ rmse($(U_s, V_s) \cup (U_c, V_c), p_{s+c}$)
21:     if $E < E_{min} \wedge |(U_s, V_s) \cup (U_c, V_c)| > C_{max}$ then
22:         $E_{min} \leftarrow E$
23:         $C_{max} \leftarrow |(U_s, V_s) \cup (U_c, V_c)|$
24:         $p* \leftarrow p_{sc}$
25:     end if
26:     $l \leftarrow l + 1$
27: end while
28: return estimated pose $p*$
```

3.2 Applying to other RANSAC variants

Out of the numerous descendants from the standard form of RANSAC, there are a few variants that can significantly speed up its computation. One of those variants is called Preemptive RANSAC (Nister 2003). In this method, using a breadth-first approach, all the hypotheses are evaluated on a subset of the dataset and scored based on their alignment error, according to which hypotheses are filtered out in stages. At each subsequent stage, the remaining hypotheses are further evaluated on another subset of the dataset. The last remaining hypothesis or the hypothesis with the least alignment error is chosen as the final solution.

In another variant of RANSAC, called R-RANSAC (Randomised RANSAC with $T_{d,d}$ Test) (Matas and Chum 2004), only a fraction of data points are evaluated using a preliminary test to identify hypotheses contaminated with outliers. Full evaluation is only performed for the hypotheses passing this initial test. The hypothesis with the most number of inliers is taken as the final solution.

We used Preemptive RANSAC and R-RANSAC to confirm that the hypothesis evaluation using re-alignment is more accurate and that the use of sufficient statistics speeds up computation for different RANSAC variants. The modified algorithm for Preemptive RANSAC is shown in Algorithm 4. Note that in its hypothesis evaluation stage, sufficient statistics need to be calculated only once for each point pair in the test set and those can be reused to score all the hypotheses. The modified R-RANSAC algorithm is shown in Algorithm 5. Here, the use of sufficient statistics is identical to the standard RANSAC. The only difference between the two algorithms lies in the hypotheses evaluation step where R-RANSAC employs a preliminary test in filtering hypotheses.

Algorithm 4 Preemptive RANSAC with HT2 and SS

```
Input: Two sets of corresponding 3D points $U = \{u_i\}$, $V = \{v_i\}$, $i = 1, 2, \dots, n$
Output: Rotation $R$ and translation $T$ (pose $p$).
Begin:
 1: $h \leftarrow 0$
 2: while $h <$ max_iterations do
 3:     $U_s^h, V_s^h \leftarrow$ randomMinimalSampleSet$(U, V)$
 4:     $p_s^h \leftarrow$ relativePose$(U_s^h, V_s^h)$
 5:     $\Omega_s^h \leftarrow$ suffStats$(U_s^h, V_s^h)$
 6:     $e_s^h \leftarrow$ rmse$(U_s^h, V_s^h)$
 7:     $h \leftarrow h + 1$
 8: end while
 9: for each $u_i \in U$ and $v_i \in V$ do
10:     $\Omega_i \leftarrow$ suffStats$(u_i, v_i)$
11:     for each $h \in 1..F(i)$ do
12:         $\Omega^h \leftarrow$ updateSuffStats$(\Omega_s^h, \Omega_i)$
13:         $e_i \leftarrow$ rmse$(\Omega^h, p_s^h)$
14:         $E^h \leftarrow E^h + |e_i - e_s^h|$
15:     end for
16:     reorder the hypotheses (h) according to $E^h$ in ascending order so that it contains best $F(i+1)$ remaining hypotheses
17:     if $F(i+1) = 1$ then
18:         break
19:     end if
20: end for
21: select hypotheses that has least cumulative error as the best hypotheses $p_s^b$
22: for each $u_i \in U$ and $v_i \in V$ do
23:     $e_i \leftarrow |$rmse$(\Omega_i, p_s^b) - e_s^b|$
24:     if $|e_i| < threshold$ then
25:         add $u_i$, $v_i$ to consensus set $U_c$, $V_c$
26:         $\Omega_c \leftarrow$ updateSuffStats$(\Omega_c, \Omega_i)$
27:     end if
28: end for
29: $p* \leftarrow$ updateRelativePose$(\Omega_c, p_s^b)$
30: return estimated pose $p*$
```

4 Time complexity analysis

The time complexity of three hypothesis evaluation methods HT1, HT2, and HT2+SS are discussed below. Here we analyse the computations involved in standard RANSAC iterations. First, we define the following:

- n: number of point pairs involved in alignment
- m: minimum number of point pairs required to generate a hypothesis
- N: number of hypothesis evaluations or RANSAC iterations
- d: number of dimensions, in our case it is 3

Algorithm 5 R-RANSAC with $T_{d,d}$ test, HT2, and SS

```
Input: Two sets of corresponding 3D points $U = \{u_i\}$, $V = \{v_i\}$, $i = 1, 2, \dots, n$
Output: Rotation $R$ and translation $T$ (pose $p*$).
Begin:
 1: $l \leftarrow 0$
 2: $C_{max} \leftarrow 0$
 3: $E_{min} \leftarrow$allowable error
 4: while $l <$ max_iterations do
 5:     $U_s, V_s \leftarrow$ randomMinimalSampleSet$(U, V)$
 6:     $p_s \leftarrow$ relativePose$(U_s, V_s)$
 7:     $\Omega_s \leftarrow$ suffStats$(U_s, V_s)$
 8:     $e_s \leftarrow$ rmse$(U_s, V_s)$
 9:     randomly select $u_d \in U_t = U \setminus U_s$ and $v_d \in V_t = V \setminus V_s$
10:     $e_d \leftarrow$ rmse$((u_d, v_d), p_s)$
11:     if $|e_d| < threshold$ then
12:         for each $u_i \in U_t = U \setminus U_s$ and $v_i \in V_t = V \setminus V_s$ do
13:             $\Omega_i \leftarrow$ suffStats$(u_i, v_i)$
14:             $\Omega' \leftarrow$ updateSuffStats$(\Omega_s, \Omega_i)$
15:             $e_i \leftarrow$ rmse$(\Omega', p_s) - e_s$
16:             if $|e_i| < threshold$ then
17:                 add $u_i$, $v_i$ to consensus set $U_c$, $V_c$
18:                 $\Omega_c \leftarrow$ updateSuffStats$(\Omega_c, \Omega_i)$
19:             end if
20:         end for
21:         $\Omega_{sc} \leftarrow$ updateSuffStats$(\Omega_s, \Omega_c)$
22:         $e_i \leftarrow$ rmse$(\Omega_{sc}, p_s)$
23:         $E \leftarrow E + |e_i - e_s|$
24:         $p_{sc} \leftarrow$ updateRelativePose$(\Omega_{sc}, p_s)$
25:         if $E < E_{min} \wedge |(U_s, V_s) \cup (U_c, V_c)| > C_{max}$ then
26:             $E_{min} \leftarrow E$
27:             $C_{max} \leftarrow |(U_s, V_s) \cup (U_c, V_c)|$
28:             $p* \leftarrow p_{sc}$
29:         end if
30:     end if
31:     $l \leftarrow l + 1$
32: end while
33: return estimated pose $p*$
```

4.1 Hypothesis generation

Calculation of the rotation R and translation T that aligns randomly chosen minimum sample sets U_s and V_s of corresponding 3D points involves following steps:

- Translation is the difference between the geometric centroids of two corresponding points sets. The centroid is calculated through additions of m points and followed by a division for each dimension and each set ($6(m+1)+3$ operations).
- Computation of rotation involves translating the points such that the corresponding geometric centroid is at the origin ($6m$), calculation of terms $x_{im}, x_{ip}, \ldots$ needed for matrix Q ($6m$), formation of matrix Q (diagonal elements - $4 \times 6m$, other elements - $12 \times 4m$), and diagonalisation of Q to obtain the solution (approximately 8 operations).

Therefore, the total number of operations for hypothesis generation amounts to $90m + 17$ (say G).

4.2 HT1: Re-projection error based hypothesis evaluation

Computation of re-projection error of point pair u_i, v_i in the test sets given transformation (R,T) involves translating (6 operations) and rotating the 3-D point v_i (18 operations: 3×3 rotation matrix multiplication with 3×1 vector representing a point) and calculating the Euclidean distance between the transformed point and u_i (10 operations). So in total 34 operations for testing a single point pair. Therefore, the total number of operations needed for all point pairs in the test is $34(n-m)$ (say H_1).

4.3 HT2: Realignment based hypothesis evaluation

The operations involved in realignment based evaluation is similar to hypothesis generation process. The only difference is that the number of point pairs involved is $m+1$. So, we get the number of operations needed for single evaluation by substituting $m+1$ in G ($90(m+1)+17 = 90m+107$ operations). Therefore, the total number of operations needed for all point pairs in the test set is $(n-m)(90m+107)$ (say H_2). This is the operation cost of HT2 without the use of sufficient statistics (SS). When SS is applied, the evaluation step only updates each element the matrix Q with the point pair from the test set, and then the solution is obtained through diagonalisation of Q. So, the computational cost is easily computed by applying the value 1 for m in G, which yields 107 operations. Therefore, for the computational cost of HT2 with SS is $107(n-m)$ operations (say H_3).

Computational cost for standard RANSAC with HT1 evaluation method and N iterations amounts to $N(34n - 56m + 17)$ $(= N(G + H_1))$.

Similarly, cost of standard RANSAC with HT2 amounts to $N(90mn + 107n - 90m^2 - 17m + 17)$ $(= N(G + H_2))$.

Similarly, cost of standard RANSAC with HT2+SS amounts to $N(107n - 17m + 17)$ $(= N(G + H_3))$.

In our case, m is always three, and in our experiments, we expect n to be in the order of tens or hundreds. So we assume that $n >> m$. Thus, from above we can write the order of RANSAC with HT1, HT2 and HT2+SS as $\mathcal{O}(34nN)$, $\mathcal{O}((90m + 107)nN)$ and $\mathcal{O}(107nN)$, respectively. Note that we have kept the coefficients as the order of complexity in all three evaluations are the same. Based on the above analysis HT2+SS should be approximately 3.52 times faster than HT2 ($\frac{(90m+107)nN}{107nN}$, where $m = 3$). According to the results given in Table 3, the average ratio for standard RANSAC on all datasets is 2.76, which is quite similar to the value from the analysis. This small discrepancy may be mainly due to not accounting for differences in floating-point operations (e.g. division vs. addition) as well as integer and logical operations depending on the underlying hardware. Execution time comparison given in Table 3 is based on an ARM processor board. When the algorithms were allowed to run on a standard PC, we observed the said ratio to be on average 3.49.

5 Experiment Design and Results

In order to test the proposed hypothesis evaluation method, we implemented a feature-based pose estimation method which combines depth and visual information, since this category of methods has been identified as the fastest in (Fang and Scherer 2014). A simple block diagram of this method is shown in Fig. 1.

The pose estimation process starts by prepossessing and extracting features from an RGB image using the 'Good Features to Track' (Shi and Tomasi 1994) method. Detected features are then tracked across successive RGB image frames using the pyramidal implementation of the Lucas-Kanade feature tracker (Bouguet 2000). We adopt a keyframe technique for reducing short-scale drift, where features are only detected for keyframes and then tracked across several successive frames until a new keyframe is assigned. The optical flow step yields two sets of feature points which are then projected onto 3-D coordinates by using their two-dimensional feature positions and the corresponding depth data. Once outliers in the 3-D point correspondences are removed using a RANSAC procedure, the transformation between these two sets of filtered 3D points is computed and in turn used to calculate the relative pose of the camera between those frames.

This pose estimation method was implemented in C++ using OpenCV[1] for feature detection and tracking as well as the open-source implementation[2] of (Konagurthu et al. 2014) for alignment of 3D points and computing sufficient statistics. For feature detection and tracking the OpenCV implementations include several parameters which can be adjusted to fine tune its performance. The values of parameters used in the experiments are listed in Table 2.

5.1 Scalability with increasing number of features

Figure 2 shows a comparison of run-times for one standard RANSAC iteration using HT2 with and without

[1] `http://opencv.org/`
[2] `http://www.csse.monash.edu.au/~karun/superpose3D/`

Table 1: List of RGB-D benchmark dataset sequences used

Sequence	**Duration**	**Length**	**Avg. Trans.**	**Avg. Rot.**	**Bounding**
	[s]	**[m]**	**Vel. [m/s]**	**Vel. [deg/s]**	**Box**
TUM dataset (Sturm et al. 2012), Device: Kinect v1					
fr3_long_office	87.10	21.455	0.249	10.188	5.12 x 4.89 x 0.54
fr2_desk	69.15	18.88	0.193	6.338	3.90 x 4.13 x 0.57
fr1_desk	23.35	9.263	0.413	23.327	2.42 x 1.34 x 0.66
Microsoft 7-Scenes dataset (Glocker et al. 2013), Device: Kinect v1					
Chess - seq1	na	9.48	na	na	1.78 x 0.55 x 2.41
CoRBS dataset (Wasenmüller et al. 2016), Device: Kinect v2					
D5 sequence	39.2	16.4	0.419	33.73	1.18 x 2.32 x 0.76
E2 sequence	66.7	23.0	0.344	26.33	3.40 x 3.70 x 1.34

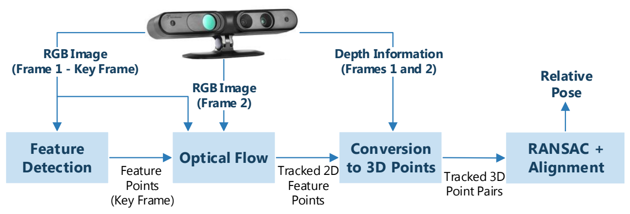


Fig. 1: Information flow in Pose Estimation: for feature detection and optical flow commonly used OpenCV C++ implementations of goodFeaturesToTrack and calcOpticalFlowPyrLK are used, respectively

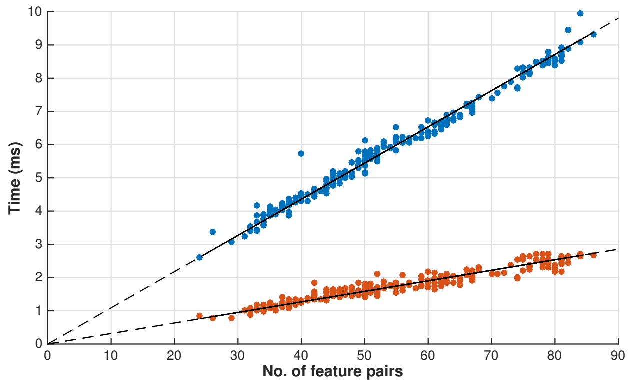


Fig. 2: Run-time of one standard RANSAC iteration using HT2 with and without SS are shown in blue and red, respectively.

SS. Without sufficient statistics, the run-time grows linearly with a steep gradient of 0.109 as the number of input feature pairs is increased. When sufficient statistics are used, the run-time growth rate is much lower (slope=0.032). These empirical run-time results showing a speed-up of 0.109/0.032 = 3.4 is consistent with

Table 2: Parameters used for feature detection and tracking

	Parameter	Value
KLT	winSize	31 × 31
	maxLevel	3
	criteria	maxCount(10) or epsilon(0.03)
	flags	OPTFLOW_LK_GET_MIN_EIGENVALS
	minEigThreshold	0.001
GFTT	maxCorners	35
	qualityLevel	0.01
	minDistance	25
	blockSize	3
	useHarrisDetector	false

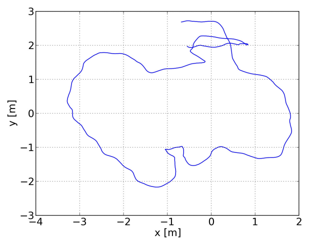
the theoretical speed-up of 3.52 obtained in the time complexity analysis.

5.2 System Overview

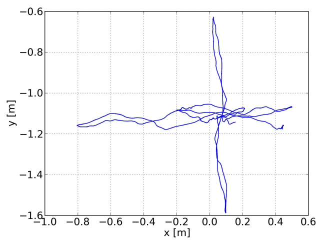
We have applied our algorithm to several RGB-D benchmark datasets in order to verify accuracy and computational performance on the processing system of our low cost MAV. The system used in this work includes an Odroid-XU4 running Ubuntu 16.04 LTS and containing Samsung Exynos5422 Cortex^{TM}-A15 2Ghz, Cortex^{TM}-A7 Octa core CPUs, and 2 GB LPDDR3 RAM.

5.3 Test with benchmark datasets

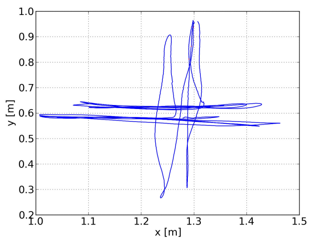
Since the introduction of RGB-D cameras numerous datasets have been released. Based on an assessment of these datasets provided in (Firman 2016), we selected the dataset sequences listed in Table 1 to evaluate the accuracy of our method. Graphical presentations of the ground truth trajectories of these datasets are shown in Figure 3, allowing visual comparison of the camera movement patterns and trajectory lengths.



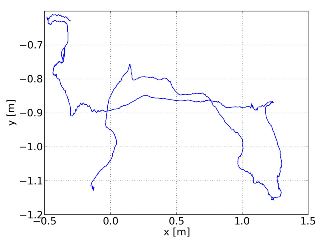
(a) TUM fr3_long_office



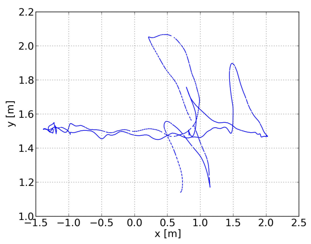
(b) TUM fr2_desk



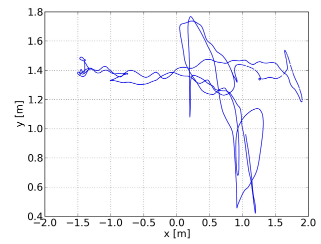
(c) TUM fr1_desk



(d) Microsoft chess-seq1



(e) CoRBS D5



(f) CoRBS E2

Fig. 3: Ground truth trajectories of the different datasets used for evaluation

In order to validate that HT2 with SS (Algorithm 3) can improve two RANSAC variants as well as the standard RANSAC, we implemented these RANSAC variants in three forms; 1) with HT1, 2) with HT2, and 3) with HT2 and SS. Parameters like *threshold* were fine tuned to minimise the estimation error using benchmark sequences fr3_long_office and fr1_desk, and these set values were applied to all other sequences.

Table 3 shows the root-mean-square error (RMSE) for the trajectory as reported by the tool provided with the benchmark datasets, averaged over five independent test runs varying the seed of the random number generator. All of the pose estimation routines were tested with the same set of random seeds. Table 3 also shows the execution time for a single RANSAC iteration (i.e. the time taken to test one hypothesis), averaged over the five independent test runs of each routine. It can be seen that pose estimation with HT2 produces less error compared to HT1 in all the cases, standard RANSAC, preemptive RANSAC and R-RANSAC. Furthermore, the use of sufficient statistics speeds up the routine with HT2 requiring only 36.45%, 14.87% and 36.03% of execution time, respectively in comparison with routine with HT2 only. In Preemptive RANSAC, which is the fastest method among the three, HT2 needs an execution time of around 9.73ms while employing SS significantly boosts up the processing speed requiring only around 1.45ms. In other words, a speed up of about 6.72 times is achieved.

Based on our results in Table 3, compared to HT1, HT2 has improved pose estimation accuracy (RMSE) by as much as 35%, 21%, and 18% and on average 18%, 6%, and 9% for standard RANSAC, preemptive RANSAC, and R-RANSAC, respectively. Average accuracy gain for preemptive RANSAC is relatively small compared to other two variants of RANSAC; but 6% is still a significant improvement.

As alluded in Section 2.2, HT2 is expected to be slower due to re-computation of all transformations. Compared to HT1, HT2 slows down pose estimation significantly by 14.6 msec on average. However, with the help of SS, HT2 + SS slows down pose estimation marginally by only 2.2 msec on average, which is insignificant compared to significant gain in pose estimation accuracy.

Even though our earlier assumption on the error terms being normally distributed with $\mathcal{N}(0, \sigma)$ may have sounded counter-intuitive, it is clear that our test results underline this assumption to be valid in practice. As shown in the columns HT2 and HT2+SS in Table 3, RMS errors have remained the same after applying HT2 with SS. It may be argued that having sufficiently low numbers of outliers could have hidden this issue. However, this is not the case in our experiments as all the benchmark datasets we used contain high percentages of outliers; for example, TUM fr1_desk has about 40% of outliers on average. This high percentage of outliers justifies our use of RANSAC. Due to its ability to make robust estimates in the presence of a significant percentage of gross errors, RANSAC is widely used in RGB-D-base pose estimation methods, for example in (Heredia et al. 2015; Li et al. 2015; Valenti et al. 2014b; Endres et al. 2014).

Figure 4 shows a comparison between the execution times of HT2 and HT2 with SS on RANSAC variants over the entire E2 sequence. Figure 5 shows detailed comparisons of position estimation accuracy of HT1 and HT2 with respect to the ground truth on the benchmark dataset E2 sequence from one of the experiments run. Displacements from the ground truth

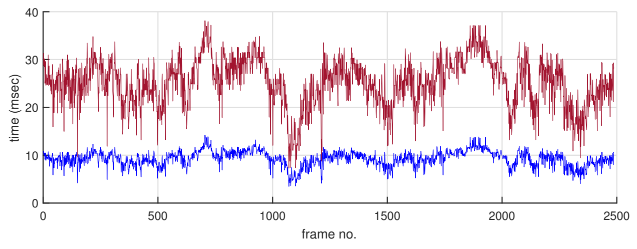
Table 3: RMSE and Execution time for RANSAC variants

RANSAC Variant	**RMSE (m)**			**Execution Time (msec)**		
	HT1	HT2	HT2+SS	HT1	HT2	HT2+SS
fr3_long_office						
Standard RANSAC	0.461	0.369	0.369	5.80	25.34	9.03
Preemptive RANSAC	0.635	0.620	0.620	0.94	9.73	1.46
R-RANSAC with T_{dd} test	0.446	0.415	0.415	4.81	19.68	7.15
fr2_desk						
Standard RANSAC	0.066	0.062	0.062	6.12	24.95	8.86
Preemptive RANSAC	0.053	0.052	0.052	0.92	9.84	1.44
R-RANSAC with T_{dd} test	0.074	0.069	0.069	5.80	22.89	8.43
fr1_desk						
Standard RANSAC	0.153	0.125	0.125	6.17	24.91	9.06
Preemptive RANSAC	0.165	0.130	0.130	0.93	9.70	1.46
R-RANSAC with T_{dd} test	0.137	0.113	0.113	5.76	23.44	8.58
Chess - seq1						
Standard RANSAC	0.428	0.398	0.398	6.45	27.49	10.11
Preemptive RANSAC	0.412	0.407	0.407	0.92	9.73	1.46
R-RANSAC with T_{dd} test	0.406	0.402	0.402	5.84	24.06	8.53
D5 sequence						
Standard RANSAC	0.539	0.351	0.351	6.33	27.02	10.01
Preemptive RANSAC	0.357	0.317	0.317	0.94	9.75	1.45
R-RANSAC with T_{dd} test	0.550	0.456	0.546	5.72	23.98	8.51
E2 sequence						
Standard RANSAC	0.730	0.571	0.571	6.27	26.81	9.93
Preemptive RANSAC	0.606	0.601	0.601	0.89	9.68	1.42
R-RANSAC with T_{dd} test	0.581	0.546	0.546	5.68	23.78	8.43

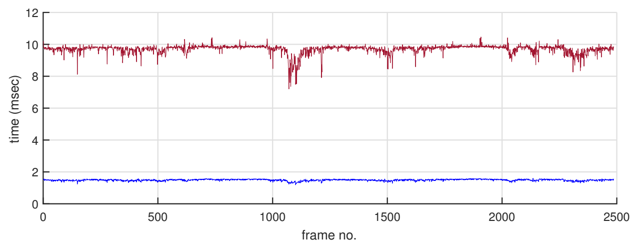
positions (signed position estimation errors) along each of the three axes can be easily observed. The caption also reports the final drift (magnitude of 3-D displacement) as 0.8349m and 0.4461m for HT1 and HT2, respectively, with standard RANSAC. HT2 reduces drift in position estimation from the ground truth by 46.6%. Relative position estimation errors between each consecutive frames for the same run of the experiment as in Figure 5 are shown in Figure 6. It is evident that our method consistently maintains better evaluation speed across all frames while maintaining the accuracy of HT2.

6 Conclusion

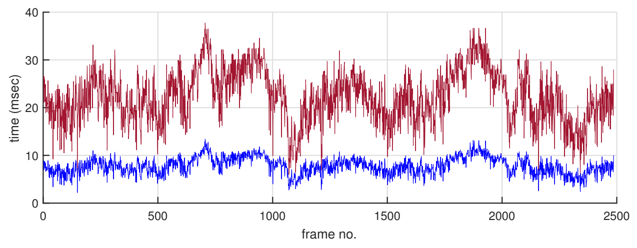
This paper presents a method to accelerate RANSAC-based outlier detection in the problem of relative pose estimation using 3-D point correspondences.RANSAC involves hypothesis evaluation where the transformation that aligns 3D point pairs is computed numerous times to eliminate the outliers. This evaluation is typically done by computing the re-projection error of all matches. An alternative method uses realignment based hypothesis evaluation where transformations are calculated for each of the matches. By nature, the re-alignment based method is more accurate in rejecting outliers than the re-projection error based hypothesis



(a) Standard

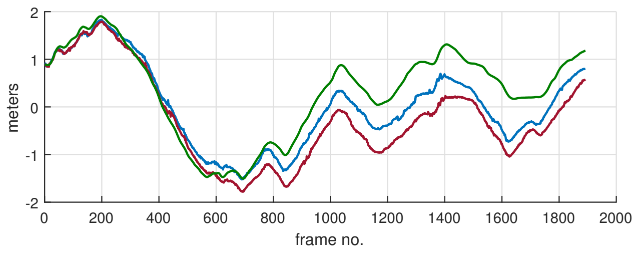


(b) Preemptive

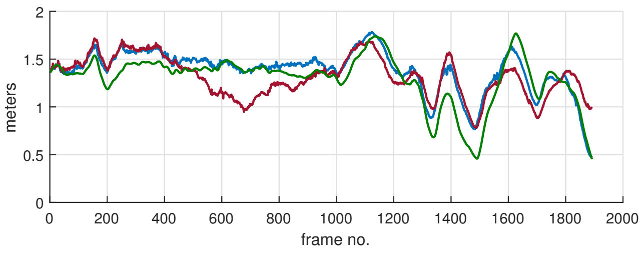


(c) R-RANSAC

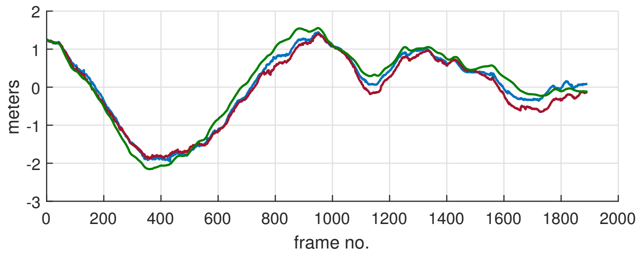
Fig. 4: Detail comparison of frame-by-frame execution time between HT2 (red) and HT2 with SS (blue) on variants of RANSAC over the E2 sequence.



(a) X Direction



(b) Y Direction



(c) Z Direction

Fig. 5: Comparison of position estimation with HT1 (red), HT2 (blue) and the ground truth (green) on E2 sequence. Drift of the final position of the camera when applied standard RANSAC with HT1 and HT2 are 0.8349m and 0.4461m, respectively.

evaluation as effects of the measurement errors are reduced during realignment. However, the realignment based method is slower due to re-fitting points at each iteration.

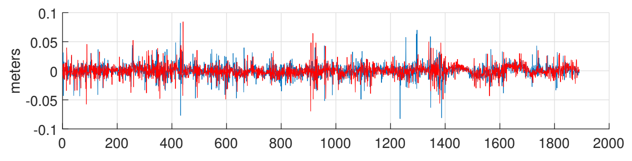
The key idea of our approach lies in the observation that the computation of relative pose depends on a number of 'sufficient statistics,' that is, a number of functions on the input data. The use of the sufficient statistics of least-squares minimisation makes it possible to speed-up the more accurate hypothesis evaluation by computing these statistics incrementally in each RANSAC iteration, not from scratch. Due to the additive nature of these statistics, the transformations in each evaluation can be quickly calculated by reusing previously computed statistics. The accuracy and performance of the proposed method were tested on three RANSAC versions, namely standard RANSAC, Preemptive RANSAC and R-RANSAC, using several public real-world benchmark datasets by running them on an ARM-based embedded computer. The experiments showed that the use of sufficient statistics accelerates the outlier detection process on all cases, requiring only 15% of execution time in case of Preemptive RANSAC without sufficient statistics. Our algorithm is currently being implemented on multiple quadrotor MAVs and will be tested in an indoor environment for solving collaborative localisation problem.

References

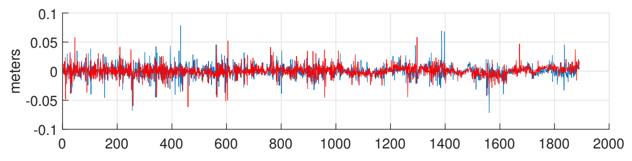
Bachrach A., Prentice S., He R., Roy N. (2011) Range–robust autonomous navigation in gps-denied environments. Journal of Field Robotics 28(5):644–666

Bachrach A., Prentice S., He R., Henry P., Huang A. S., Krainin M., Maturana D., Fox D., Roy N. (2012) Estimation, planning, and mapping for autonomous flight using an rgb-d camera in gps-denied environments. The International Journal of Robotics Research 31(11):1320–1343

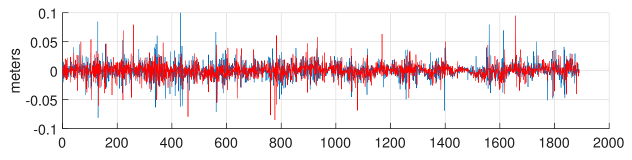
Besl P., McKay N. D. (1992) A method for registration of 3-d shapes. Pattern Analysis and Machine Intelligence, IEEE Transactions on 14(2):239–256, DOI 10.1109/34.121791



(a) X Direction



(b) Y Direction



(c) Z Direction

Fig. 6: Comparison of relative position estimation error with HT1 (red) and HT2 (blue) corresponding to the experiment in Figure 5.

Bouguet J.-Y. (2000) Pyramidal implementation of the lucas kanade feature tracker. Tech. rep., Intel Corporation, Microprocessor Research Labs

Bry A., Bachrach A., Roy N. (2012) State estimation for aggressive flight in gps-denied environments using onboard sensing. In: Robotics and Automation (ICRA), 2012 IEEE International Conference on, pp. 1–8

Chowdhary G., Sobers D. M., Pravitra C., Christmann C., Wu A., Hashimoto H., Ong C., Kalghatgi R., Johnson E. N. (2012) Self-contained autonomous indoor flight with ranging sensor navigation. Journal of Guidance, Control, and Dynamics 35(6):1843–1854

Endres F., Hess J., Sturm J., Cremers D., Burgard W. (2014) 3-d mapping with an RGB-D camera. IEEE Trans Robotics 30(1):177–187

Fang Z., Scherer S. (2014) Experimental study of odometry estimation methods using rgb-d cameras. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Firman M. (2016) RGBD Datasets: Past, Present and Future. In: CVPR Workshop on Large Scale 3D Data: Acquisition, Modelling and Analysis

Fischler M. A., Bolles R. C. (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395

García Carrillo L., Dzul López A., Lozano R., Pégard C. (2012) Combining stereo vision and inertial navigation system for a quad-rotor uav. Journal of Intelligent & Robotic Systems 65(1-4):373–387

Geiger A., Ziegler J., Stiller C. (2011) Stereoscan: Dense 3d reconstruction in real-time. In: Intelligent Vehicles Symposium (IV), 2011 IEEE, pp. 963–968

Glocker B., Izadi S., Shotton J., Criminisi A. (2013) Real-time rgb-d camera relocalization. In: Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on, pp. 173–179

Heredia M., Endres F., Burgard W., Sanz R. (2015) Fast and robust feature matching for RGB-D based localization. CoRR abs/1502.00500

Hogg R. V., Craig A. (1994) Introduction to Mathematical Statistics, 5th edn. Prentice Hall

Horn B. K. P. (1987) Closed-form solution of absolute orientation using unit quaternions. Journal of the Optical Society of America A 4(4):629–642

Kearsley S. K. (1989) On the orthogonal transformation used for structural comparisons. Acta Crystallographica Section A 45(2):208–210

Kerl C., Sturm J., Cremers D. (2013) Robust odometry estimation for rgb-d cameras. In: Robotics and Automation (ICRA), 2013 IEEE International Conference on, pp. 3748–3754

Khoshelham K., Elberink S. O. (2012) Accuracy and resolution of kinect depth data for indoor mapping applications. Sensors 12(2):1437–1454

Konagurthu A., Kasarapu P., Allison L., Collier J., Lesk A. (2014) On sufficient statistics of least-squares superposition of vector sets. In: Sharan R. (ed) Research in Computational Molecular Biology, Lecture Notes in Computer Science, vol 8394, Springer International Publishing, pp. 144–159

Li D., Li Q., Cheng N., Wu Q., Song J., Tang L. (2013) Combined rgbd-inertial based state estimation for mav in gps-denied indoor environments. In: Asian Control Conference (ASCC), 2013 9th, pp. 1–8

Li D., Li Q., Tang L., Yang S., Cheng N., Song J. (2015) Invariant observer-based state estimation for micro-aerial vehicles in gps-denied indoor environments using an RGB-D camera and MEMS inertial sensors. Micromachines 6(4):487–522

Matas J., Chum O. (2004) Randomized {RANSAC} with td,d test. Image and Vision Computing 22(10):837 – 842, british Machine Vision Computing 2002

Nister D. (2003) Preemptive ransac for live structure and motion estimation. In: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on, pp. 199–206 vol.1

Pomerleau F., Colas F., Siegwart R., Magnenat S. (2013) Comparing icp variants on real-world data sets. Autonomous Robots 34(3):133–148

Scherer S., Zell A. (2013) Efficient onboard rgbd-slam for autonomous mavs. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, pp. 1062–1068

Senthooran I., Barca J. C., Kamruzzaman J., Murshed M. M., Chung H. (2015) An efficient pose estimation for limited-resourced mavs using sufficient statistics. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015, Hamburg, Germany, September 28 - October 2, 2015, pp. 3735–3740

Shen S., Michael N., Kumar V. (2011) Autonomous multi-floor indoor navigation with a computationally constrained mav. In: Robotics and Automation (ICRA), 2011 IEEE International Conference on, pp. 20–25

Shi J., Tomasi C. (1994) Good features to track. In: Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on, pp. 593–600

Stowers J., Hayes M., Bainbridge-Smith A. (2011) Altitude control of a quadrotor helicopter using depth map from microsoft kinect sensor. In: Mechatronics (ICM), 2011 IEEE International Conference on, pp. 358–362

Sturm J., Engelhard N., Endres F., Burgard W., Cremers D. (2012) A benchmark for the evaluation of rgb-d slam systems. In: Proc. of the International Conference on Intelligent Robot Systems (IROS)

Szeliski R. (2011) Computer Vision - Algorithms and Applications. Texts in Computer Science, Springer-Verlag London Limited

Valenti R. G., Dryanovski I., Jaramillo C., Perea Strom D., Xiao J. (2014a) Autonomous quadrotor flight using onboard rgb-d visual odometry. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp. 5233–5238

Valenti R. G., Dryanovski I., Jaramillo C., Strom D. P., Xiao J. (2014b) Autonomous quadrotor flight using onboard RGB-D visual odometry. In: 2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014, pp. 5233–5238

Voigt R., Nikolic J., Hurzeler C., Weiss S., Kneip L., Siegwart R. (2011) Robust embedded egomotion estimation. In: Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, pp. 2694–2699

Wang C., Wang T., Liang J., Chen Y., Wu Y. (2012) Monocular vision and imu based navigation for a small unmanned helicopter. In: Industrial Electronics and Applications (ICIEA), 2012 7th IEEE Conference on, pp. 1694–1699

Wasenmüller O., Meyer M., Stricker D. (2016) CoRBS: Comprehensive rgb-d benchmark for slam using kinect v2. In: IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, vol .

Weiss S., Achtelik M., Lynen S., Chli M., Siegwart R. (2012) Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments. In: Robotics and Automation (ICRA), 2012 IEEE International Conference on, pp. 957–964