J Real-Time Image Proc manuscript No.
(will be inserted by the editor)

Hamid Bazargani · Olexa Bilaniuk · Robert Laganière

A Fast and Robust Homography Scheme for Real-time Planar Target Detection

Received: date / Revised: date

Abstract The present paper is concerned with the problem of robust pose estimation for planar targets in the context of real-time mobile vision. For robust recognition of targets at very low computational costs, we employ feature based methods which are based on local binary descriptors allowing fast feature matching at run-time. The matching set is then fed to a robust parameter estimation algorithm in order to obtain a reliable estimate of homography. The robust estimation of model parameters, which in our case is a 2D homographic transformation, constitutes an essential part of the whole recognition process. We present a highly optimized and device-friendly implementation of homography estimation through a unified hypothesize-and-verify framework. This framework is specifically designed to meet the growing demand for fast and robust estimation on power-constrained platforms. The focus of the approach described in this paper is not only on developing fast algorithms for the recognition framework but also on the optimized implementation of such algorithms by accounting for the computing capacity of modern CPUs. The experimentations shows that the resulting homography estimation implementation proposed in this paper brings a speed up of 25x over the regular OpenCV RANSAC homography estimation function.

Keywords target matching, robust estimation, Homography, RANSAC, hypothesize-and-verify

H. Bazargani, Olexa Bilaniuk, Robert Laganière (✉)
School of Electrical Engineering and Computer Science,
University of Ottawa, Ottawa ON, Canada K1N 6N5
`http://www.site.uottawa.ca/research/viva/`

R. Laganière
E-mail: laganier@uottawa.ca

H. Bazargani
E-mail: hbaza043@uottawa.ca

O. Bilaniuk
E-mail: obila060@uottawa.ca

1 Introduction

The detection of objects in video is an important and challenging problem in computer vision. When this detection must be performed in live video captured from a mobile device, object detection becomes even more challenging. Mobile devices have limited computational resources and memory and these constraints must be taken into account. For this reason, feature-based approaches are often preferred. Assuming the object of interest contains sufficient textural details, these approaches achieve excellent results while limiting the analysis of each frame to a small subset of image areas.

The basic ingredients of a feature-based object detection framework are feature detection and feature description. For feature detection, the FAST algorithm [27] constitutes the prime choice since it allies good repeatability at low computational cost. For feature description however, there exists a large number of alternatives. In the embedded vision world, binary descriptors are often preferred because they use a fast-to-compute binary string to describe a keypoint patch, thus allowing efficient matching at run-time using only simple bit-wise operations.

Most binary descriptors use the influential pair-wise intensity comparison exploited in [24, 15]. To supply insensitivity to noise and increase keypoint quality, Gaussian smoothing kernels are applied before conducting the pair-wise tests. Following a similar concept, BRIEF [7] selects pairs' location using an experimentally chosen random distribution over the keypoint patch. The Oriented FAST and Rotated BRIEF (ORB) [28], as it is aptly named, is a rotation invariant version of BRIEF that incidentally provides more robustness to scale by extracting FAST corners across the image pyramids. In BRISK [17], concentric receptive fields are uniformly distributed around the center of a keypoint. Similarly, the sampling grid in FREAK method [2] consists of a circular overlapping pattern in which the size of Gaussian kernels exponentially increases. According to the binary property of these local binary descriptors, features' similarity can be quickly evaluated by computing their Hamming distance.

Matching the detected features can be simply accomplished through a classical brute force nearest neighbor search. Additional constraint can also be imposed in order to produce a match set of better quality. For instance, a distance ratio test limits the number of tentative matches by rejecting candidates whose first to the second nearest neighbor distance is greater than a predefined threshold (typically set to 0.6-0.8). Imposing a symmetrical constraint can further improve the match set quality by only maintaining matches in which each point is the others nearest neighbor [33]. Obviously, the large number of representative features makes the matching process extremely slow. Therefore, for quick access to the nearest neighbors, a special data structure is used in [29]. They proposed approximate Best Bin First (BBF) strategy based on k-d tree [5] searching algorithm. In case of binary descriptors, an indexing approach can be used to avoid exhaustive search over the entire set of features. For instance, Locality Sensitive Hashing (LSH) strategy has been introduced in [13] to assign an index number to each feature by computing a hash function on the region around that feature point. So, at run-time, features are only compared against those with the same index number.

Nevertheless, the match set obtained will always be contaminated by a noticeable proportion of outliers. This is where a robust estimation phase is required. To validate the matches, a parametric model is defined and the match set is used to estimate the parameters of this model. The most popular framework to perform this estimation is RANdom SAmpling Consensus (RANSAC) firstly introduced by Fischler and Bolles [11]. This hypothesize-and-verify approach randomly selects a minimal set of samples and estimates the model parameters until achieving consensus with the strongest support for the parameters.

Despite the simplicity and efficiency of the RANSAC algorithm even when samples are significantly contaminated, it still suffers from serious drawbacks which results in overall performance degradation. In the past few years, different RANSAC flavors have been developed to account for these boundaries. PROSAC [9] is a variant of RANSAC that aims at an early identification of model parameters under the assumption that samples with higher quality are more likely to generate a hypothesis with a strong support. PROSAC follows a non-uniform sampling approach in which samples are selected from a smaller subset of correspondences. The size of this subset progressively increases until it eventually coincides with the full match set. FT-RANSAC extends the PROSAC scheme to the case of multi-modal datasets [3]. Randomized RANSAC (R-RANSAC) [20] is introduced to address a practical limitation of the RANSAC algorithm in the context of real-time applications. R-RANSAC specifically targets the evaluation step in order to adapt the process for real-time applications by benefiting from a randomized pre-evaluation process that attempts to reduce the number of verifications per model. This pre-evaluation step that plays a key role in the performance of the process is first started with the $T_{d,d}$ test [8], then leads to an optimal sequential test called SPRT in [19]. To accelerate the rejection of degenerate configurations, a geometrical degeneracy test is introduced in [22] that assures selection of consistent points by accounting for geometrical constraints imposed by homography transformation.

Recently, Raguram et al. have proposed USAC [25], a universal RANSAC framework that incorporates most of the recent development in RANSAC-based schemes. Inspired by this work and with the objective of performing real-time object detection in mobile device, this paper revisits the hypothesize-and-verify loop in a fast matching process. Our contributions resides in the run-time process of the target recognition task. We comprehensively analyzed the problem of parameter estimation for fast and robust target matching through an optimal hypothesize-and-verify scheme. We have investigated possible limitations of RANSAC-style approaches to optimize our framework at both algorithmic level and implementation level. The optimized framework is specifically designed to be efficiently run on smart-phones and tablets. We also have developed a fast C++ software package by leveraging the state-of-the-art algorithms that have been studied over the years. The resulting homography estimation function brings a speed up of 25x over the regular OpenCV RANSAC homography estimation function. A robust method for homography estimation based on a computationally efficient implementation of the well known Gaussian Elimination (GE) algorithm has been presented in [6]. This paper describes a fast model estimation framework in a robust estimation scheme that uses this GE homography estimation component inside the hypothesis generation step. We present the implementation of a PROSAC-based approach, perform comparative studies and a thorough analysis of the model estimation loop demonstrating both the efficiency and the reliability of our proposed framework.

From the following section onwards, we explore crucial steps toward a real-time object recognition framework. In Sect. 2, we outline feature-based target detection system overview. For this end, popular methods are briefly reviewed in the context of real-time target recognition. Sect. 3 presents a unified and efficient framework for robust homography estimation. In Sect. 4, experimental results are presented. We benchmark our framework against different implementations and we show that our framework outperforms the state-of-the-art algorithms. We conclude the paper in Sect. 5 and state the main findings of our research.

2 Target Detection System Overview

Feature recognition and matching are two important parts of many target detection algorithms. Matching algorithms are generally grouped into two categories namely *local descriptor-based* and *global classification-based* algorithms. In the descriptor-based methods, image information is abstracted by using distinctive local descriptors, which are invariant to fairly large level of deformations, illumination changes and noise. Despite the robustness and distinctiveness of descriptor-based algorithms such as SIFT [18], PCA-SIFT [14] and SURF [4], the matching process requires high computational power to extract and compare local descriptors. The computational complexity and dimensionality of these methods, especially when considering low-powered platforms, stimulated the development of new approaches [15, 16, 24, 34, 30]. These approaches take advantage of a training stage to which a great extent of computational burden at run-time is transferred. In this particular direction, Lepetit et al. [16], put forward *matching as classification*, as a strategy for fast matching approaches.

In the training phase of their algorithm, a number of prominent feature points is extracted from the object to be trained. Under a local planarity assumption, patches around the detected features and their distorted version under a large level of affine transformation, participate in the learning of a classifier. Thereafter, at run-time, the classifier determines to which class, if any, a specific patch belongs. Principle Component Analysis (PCA) is then performed to reduce the dimensionality of patches.

Lepetit et al. further proposed to use a randomized decision tree [15] instead of PCA-based classification to select features, which tend to provide higher recognition rate at run-time. In the randomized tree approach, a reference model is built by considering random pairs of pixel locations inside a defined neighborhood of the keypoint. It is shown that, simple intensity comparisons would yield distinctive description and efficient matching at run-time. Random Fern [24] is another classification method proposed by Özuysal et al. that is perceived as an improvement over random trees bringing higher recognition rate at lower computational cost. In Fern, a simple binary test is performed to compare the intensity values of those pixels leading to binary features that are grouped together to create small binary space partitions called Ferns. Bearing in mind that a full joint probability model of each Fern is needed, there could be an issue with the memory requirements of this method that exponentially grows with the size and number of Ferns.

For efficient matching at run-time, Taylor and Drummond introduced a fast and efficient local binary descriptor well suited for real-time target detection. In their *Histogrammed Intensity Patches* (HIP) method [30, 31], they introduced the idea of grouping artificially generated random views into viewpoint bins. The model is built by first computing coarse histograms of intensities of neighboring pixels around a keypoint for each viewpoint bin. Once computed, each histogram is averaged and binarized. Given a pair of HIP models, a dissimilarity score is computed by identifying bins that are rarely hit with the idea that corresponding patches in the live view should have a small number of pixel values falling into these rare bits.

Although, the aforementioned methods exhibit fast and reliable matching at the online stage, they suffer from a high training time and/or memory consumption. Being inspired from the training method of [30], Akhoury et al. [1] proposed a similar framework which reduces the training time by more than 80% as compared to [30, 31]. The time reduction gain in [1], is mainly rooted in applying more representative perspective transformations to synthesize the viewset images instead of affine transformations. Unlike the HIP method, in which model descriptors are aggregated by identifying rarely hit bits, feature aggregation for most binary descriptors like BRIEF can be done by applying a majority vote on each bit of the descriptor.

Similar to the Fern method, BRIEF [7] also uses the concept of pair-wise intensity comparison to generate a binary string describing a keypoint patch. Additionally, it is proposed to apply a fixed size Gaussian kernel to each pair's location for better robustness against noise. The pattern used in BRIEF method has been experimentally chosen through a set of pre-defined distributions over a patch of size $S \times S$. These spatial distributions were evaluated based on their capability to recognize a planar target established beforehand. Among five combinations in [7], experiments show that spatial Gaussian distribution with $(x, y) \sim$ i.i.d. $\mathcal{G}(0, \frac{S^2}{25})$ outperforms the other patterns and exhibits a higher recognition rate.

For real-time detection and tracking processes, BRIEF can be allied with FAST which is a highly expedited feature detection technique originally proposed by Rosten and Drummond [26, 27]. At each pixel location, FAST requires to carry out a simple test on a ring of 16 pixels encircling the pixel of interest. Pixel p is considered to be an interest point, if there are n pixels in adjacent locations on the ring that are all either darker or all brighter than the center by a threshold δ_t. The threshold δ_t is used to control the number of detected keypoints. In the very early work of Rosten and Drummond, they chose $n = 12$ leading to an algorithm that discards large portion of non-corner points as quickly as examining only 3 out of 4 pixels in specific locations on the 16-pixel ring. In [27], a machine learning approach is introduced to adjust segment's locations and identify optimal ordering of comparisons eventually yielding the efficient FAST-9 algorithm.

3 Robust Target Matching Framework

The first step toward a framework for a real-time target matching is to extract stable features from a video frame. These features are then matched against reference features that are extracted during an offline model generation process from the view synthesis framework as described in Sect. 2. Considering the limited time budget for feature detection, we propose the FAST-9 feature detection algorithm [27]. Among different feature descriptors, we experimentally found the BRIEF method [7] to be efficient enough to fit the requirements of real-time mobile applications.

Once an initial match set is available, the next step consists in the robust estimation of a global transformation that best explains these matches. This transformation is a 2D homographic relation which is estimated through a hypothesize-and-verify scheme. In the hypothesis generation step, homography parameters are evaluated using a computationally efficient implementation of a Gaussian elimination algorithm [6] that has been designed for real-time planar target matching on handheld devices. Using Gaussian Elimination in homography estimation scheme is not new; it has been used, for example, in pose estimation problem from UAV imagery [21]. However, taking into account modern CPU architecture, our implementation exploits the structure of the problem and the sparsity of the matrices involved to the fullest. It is observed that the DLT transform has, after some row shuffling operations, 16 zeros and a symmetrical structure. A generic, numerically-stable, pivoting GE implementation for an arbitrary matrix size costs $O(n^3)$ FLOPs, on top of any control logic and pivot selection logic overhead. For an 8-variable system, a total of 428 FLOPs is required, not counting floating-point comparisons required for pivot selection. Our specialized GE implementation, by contrast, is a non-pivoting, non-branching, straight-line implementation with almost no overhead requiring a total of 208 FLOPS. Thus, our specialized GE requires half as many FLOPs as the generic GE, and has almost nil overhead.

3.1 Homography Estimation by Gaussian Elimination

In the case of planar targets, a parametric model that describes a camera pose can be formulated by a homography relation mapping two views of a planar object. This transformation can be obtained by directly solving a system of linear equations. A homography or projection transformation is a plane-to-plane ($\mathbb{P}^2 \to \mathbb{P}^2$) relation in a projective space. It is algebraically defined by a non-singular 3×3 matrix H that maps two views of a planar object. Let $\mathbf{X} = [X, Y, Z]^T$ and $\mathbf{x} = [x, y, 1]^T$ be respectively the homogeneous coordinates of an identical point on a reference plane and its projection that is seen from a different view; the 2D homography transformation is described as:

$$\mathbf{X} = H\mathbf{x} \tag{1}$$

A typical and perhaps the simplest approach for retrieving the homography relation with 8 degrees of freedom is the *Direct Linear Transformation* (DLT). DLT treats the pose constraints as a system of linear equations that can be solved by minimizing an algebraic distance. The homogeneous system of equations corresponding to the homography relation can be solved by posing:

$$X_i \times Hx_i = 0 \tag{2}$$

in which H is computed using the homogeneous source points $(x_i, y_i, 1)$ and target points (X_i, Y_i). This equation can be expressed in a vector-form as:

$$\begin{pmatrix} \mathbf{0^T} & -\mathbf{x_i}^\mathbf{T} & Y_i\mathbf{x_i}^\mathbf{T} \\ \mathbf{x_i}^\mathbf{T} & \mathbf{0^T} & -X_i\mathbf{x_i}^\mathbf{T} \\ -Y_i\mathbf{x_i}^\mathbf{T} & X_i\mathbf{x_i}^\mathbf{T} & \mathbf{0^T} \end{pmatrix} \mathbf{h} = \mathbf{0} \tag{3}$$

which results in equations of the form $\mathbf{A_i h} = \mathbf{0}$, with $\mathbf{A_i}$ being the lines of the left matrix and $\mathbf{h}$ being a 9×1 vector made of the entries of the homography matrix. *Singular Value Decomposition* (SVD) is a common technique to find non trivial solution of this system of equations. Alternative solutions have also been proposed to solve this problem, such as the *hyper-accurate* estimator presented in [23]. In this case, the focus was on obtaining very accurate estimates while in our work, we aim at very fast, yet reliable, computation of homographies in real-time scenarios. In matching applications, robust estimation of a homography from a set of putative matches is achieved based on the RANSAC algorithm. RANSAC randomly selects four point correspondences from the match set and estimates a homography relation. By repetitively estimating a homography from different random pairs, the best homography is identified as the one that is supported by the largest number of point correspondences in the set.

Even if the SVD estimation from four point correspondences can be performed with relative efficiency, its repetitive computation can still impose a significant computational load in the context of real-time estimation using low-power devices. This observation leads us to consider simpler approaches to resolve the 4-point homography estimation problem. In particular, we selected the well-known Gaussian elimination scheme [12] that can be used to solve a 4-point non-homogeneous set of equations. Even if this approach is known to be less numerically stable, we show here that in the context of target recognition, stable and accurate solutions can still be obtained. As a starting point, we consider the vector-form of homogeneous system of Eq. (3). Our implementation of the reduction to reduced-row-echelon form of the matrix is summarized here. It assumes that the minimum configuration used to estimate the homography is 4 matches. If we take the matrix in Eq. (3) to be decomposed as (after appropriate row shuffling):

$$\begin{pmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0X_0 & -y_0X_0 & X_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1X_1 & X_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -y_2X_2 & X_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3X_3 & -y_3X_3 & X_3 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0Y_0 & -y_0Y_0 & Y_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1Y_1 & -y_1Y_1 & Y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2Y_2 & -y_2Y_2 & Y_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3Y_3 & -y_3Y_3 & Y_3 \end{pmatrix} \tag{4}$$

We notice here that the matrix is somewhat sparse, and what's more, the top left 4×3 matrix minor is identical to the bottom middle 4×3 minor. This is of great help, since it means that initially, the same operations will be applied to the top 4 rows and bottom 4 rows of the matrix. Even better, when 4-lane or 8-lane vector processing engines (such as SSE, AVX, Altivec or NEON) are available, the loads of x_i, X_i, y_i and Y_i, the multiplies xX, xY, yX and yY and the row operations can be done in parallel.

After the reduction procedure given in Appendix A, the right-most column of the resulting matrix would contain elements of homography matrix. With this non-homogeneous solution, poor estimation would be obtained if the element h_{22} has a value close to zero. Gaussian elimination is however numerically stable for diagonally dominant or positive-definite matrices. For general matrices, Gaussian elimination is usually considered to be stable, when using partial pivoting [12]. In practice, we observed reliable stability when the Z-component of the translation is significant with respect to the $X - Y$ ones; this is the common situation when a hand-held device is used for target recognition.

3.2 Robust Parameter Estimation

Once a hypothesis is generated through the proposed Gaussian elimination algorithm, it is verified by evaluating its support using the complete correspondence set. The support for each homography model is determined by counting the number of correspondences whose reprojective error lies within a specific threshold. Among the different variants of RANSAC, we chose PROSAC [9] in our implementation which uses a similar random sampling strategy in which a smaller ordered subset of hypotheses are verified. Hamming distance from the matching procedure is used to sort data points based on their match quality. Thus, compared to the typical RANSAC method, PROSAC terminates in much fewer iterations by examining higher quality samples earlier.

Algorithm 1 summarizes our PROSAC implementation for a fast homography estimation. The k_{MAX} parameter defines the computational budget available for performing recognition on a frame. It starts from a maximum acceptable value and can be decreased based on the current estimate of inliers' rate.

Algorithm 1 Robust framework for H estimation

```
Require: Set of all detected matches $\mathcal{U}_N$ and $\eta_0$.
  Sort the set of correspondences with respect to the similarity score.
  Pre-compute $\chi^2$ approximate of $I_{n^*}^{min}$ satisfying Eq. (8).
  Initialize $I_{best} = 0$, $m = 4$ and $k_{MAX}$
  for $k = 0$ to $k_{MAX}$ do
    Select $m$ non-degenerate pairs using the PROSAC non-uniform approach (see Sect. 3.6 for the degeneracy test).

    $H \leftarrow$ Generate a hypothesis by Gaussian elimination approach.
    $I_k \leftarrow$ Evaluate the current hypothesis support.
    if $I_k < I_{best}$ then
      $I_{best} \leftarrow I_k$
      Update $H$ with the hypothesis with the strongest support.
      if $I_k \geq I_{n^*}^{min}$ then
        Break out of the for loop.
      end if
      $k_{MAX} \leftarrow$ Apply the maximality constraint to update $k_{MAX}$ (see Eq. (5)).
    end if
  end for
```

3.3 Speed vs Quality Trade-off in RANSAC Iterations

An often-overlooked element of RANSAC is the trade-off between *speed of iteration* and *quality of iteration.* This is primarily determined by the choice of RANSAC *kernel* – the core of the hypothesize-and-verify loop.

Currently, most RANSAC-based homography estimation methods lean heavily towards an accurate, but slower, kernel. We have here taken a different approach: we attempt to speed up as much as possible the critical path from sample selection to acceptance or rejection of the resultant homography. To do this we i) pre-validate the sample using the *strong geometric constraint* test first presented by [22]; ii) compute the homography from the sample by way of a specialized Gaussian Elimination that exploits the structure and sparsity of the matrix to be decomposed; ii) evaluate the homography using the early-exiting SPRT scheme.

Such an approach abandons work on any putative candidate as soon as it becomes clear that it is invalid or is worse than the current front-runner. It also dedicates only a minimum of effort to do so. Key to this approach is the computation of the homography, because it is the computational bottleneck, because it is incapable of rejecting a sample by itself, and because its output is the dependency of the evaluation stage, which makes the final accept/reject decision. We selected Gaussian Elimination for the homography calculation for two reasons: First, GE is extremely fast and second, it can be specialized for our particular case, for even greater gains. This processing, together with the pre-validation and SPRT, greatly shortens the critical path from sample selection to acceptance or rejection. We use GE as the kernel of RANSAC, and thus any homography corrupted by GE is rejected by the same mechanisms that reject homographies computed from samples that were contaminated to begin with. In this way, GE and RANSAC synergize: GE provides the inner speed that RANSAC needs, and RANSAC naturally makes up for GE's numerical errors by supplying GE with alternate samples. This is the trade-off that our GE strategy exposes: for a slightly higher risk of missing a good homography due to a lower quality of iteration, we gain greatly in speed of iteration.

3.4 Termination criterion

Since the hypothesize-and-verify scheme is an iterative process, it has to be terminated once a specific termination criterion is met. There are different options that can be considered for the termination criterion.

3.4.1 The Maximality constraint [11, 9]

The RANSAC loop is designed to keep iterating until assuring, with some level of confidence η_0, that a set of uncontaminated samples has been already selected. Taking this desired level of confidence η_0 into account, one can determine a maximum number of required iterations k_{max} to guarantee this level of confidence. The probability of selecting n pairs, all as inliers is w^n where w is the inlier rate of a data set. Consequently the probability of selecting n pairs with at least one outlier is $1 - w^n$. Thus, $(1-w^n)^k$ becomes the probability of never selecting outlier-free samples in k iterations and this needs to be less than $1 - \eta_0$. So at each iteration, k is updated as follows:

$$k \leq \frac{\log(1-\eta_0)}{\log(1-w^n)} \tag{5}$$

In practice, to obtain an *a priori* knowledge of w, one can estimate it by using the support of the best hypothesis found so far. In RANSAC [11], this termination constraint is known as the maximality constraint.

3.4.2 The Non-randomness Constraint [9]

Although the non-uniform sampling property of PROSAC speeds up the hypothesis convergence, applying standard maximality constraint (5) under the assumption of uniform sampling results in a larger number of samples drawn than what is actually required. Non-randomness constraint introduced in [9], is a statistical significance test that guaranties the goodness of a solution by the probability of evaluating true outliers, which are consistent with a good model, falling below a specific significance level (typically set to $5 - 10$ percent). The probability distribution of evaluating i outliers out of n points all consistent with the sought model abides by the binomial distribution. The binomial distribution is denoted as follows:

$$P_n(i) = \beta^{i-m}(1-\beta)^{n-i+m}\binom{n-m}{i-m} \tag{6}$$

in which β is the of probability of a random point evaluated as inlier given an incorrect model. For each subset of the size n, we compute the minimum number of inliers satisfying non-randomness constraint whose accumulative p-value is less than a significance level ψ.

$$I_n^{min} = min\{j : \sum_{i=j}^{n} P_n(i) < \psi\} \tag{7}$$

Thus, the PROSAC loop is terminated once the sampling subset size for a candidate solution satisfies the maximality condition (5) and

$$I_n \geq I_{n^*}^{min}. \tag{8}$$

3.4.3 The Chi-squared Approximation

Although non-randomness criterion accelerates the PROSAC algorithm by reducing the required number of iterations, it still suffers from a heavy computational burden caused by the recursive computation of binomial probabilities. Considering the randomness of a solution as a null hypothesis H_0, p-value determines the significance of rejecting H_0, given a specific level ψ. Therefore, a Chi-squared χ^2 test can be used as a common statistical interpretation of p-value. So the larger observed χ^2 corresponds to the lower p-value and thus stronger evidence against the null hypothesis. Figure 1 shows a graphical illustration of normal and χ^2 approximation to binomial probability density function.

From the central limit theorem, for sufficiently large n, Eq. (6) can be approximated by a standard normal distribution with $\mu = n\beta$ and $\sigma^2 = n\beta(1-\beta)$ and that has thus a 1 degree of freedom Chi-squared distribution under the null hypothesis. Let χ^2_ψ be the corresponding Chi-squared value for the significance level of ψ, which is itself approximated by a normal PDF, Eq. (7) thus reduces to the form

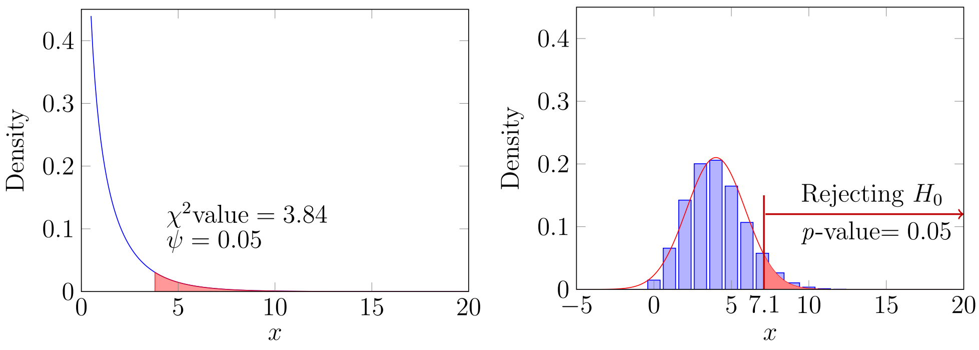
$$I_{min} = \text{ceil}\big(m + n\beta + \chi_\psi\sqrt{n\beta(1-\beta)}\big) \tag{9}$$

It will be shown in Sect. 4, that the Chi-squared approximation significantly reduces the computational load by computing $I_{n^*}^{min}$ only once, prior to the PROSAC loop, while exhibiting nearly similar accuracy to the standard non-randomness approach.

3.5 Model verification

In order to evaluate the quality of a hypothesized model, the standard RANSAC model verification step can be further optimized by using quick hypothesis filtering strategies such as the $T_{d,d}$ test or the *Sequential Probability Ratio Test* (SPRT) [10].

Fig. 1: Left shows χ^2 distribution with 1 degree of freedom. Right shows Binomial PDF and normal approximation for $n = 30$ and $\beta = 0.1$. The red area under the curves indicate p-value $= 0.05$ under H_0.



3.5.1 The $T_{d,d}$ Test [8]

In the RANSAC process, we wish to minimize the number of samples that have to be drawn as well as the average time $\bar{t}$ required to validate each hypothesis. While the early termination criterion is responsible for optimizing the number of hypotheses, verification tests such as $T_{d,d}$ [8] are designed to minimize $\bar{t}$ (which is proportional to the number of verified points). The $T_{d,d}$ verification algorithm is divided into two simple steps. In the first step, a small portion of N data points are verified from a randomly selected subset $\mathcal{U}_d \subset \mathcal{U}_N$. The second step involves the verification of all remaining data points, only if the first pre-test passes that is, all d selected points are consistent with the hypothesis.

It is proven in [8] that the optimal solution minimizing the average number of verified points $\bar{t}$, leads us to the $T_{1,1}$ ($d = 1$) test. Considering the probabilities of drawing 'good' samples P_g and its complement $1 - P_g$ (drawing 'bad' samples), one can derive $\bar{t}$ as a function of d:

$$\bar{t}(d) = P_g(\alpha N + (1-\alpha)\bar{t}_\alpha) + (1-P_g)(\beta N + (1-\beta)\bar{t}_\beta) \quad (10)$$

Equation (10) states that when a 'good' sample is provided, it yields the verification of N points with the probability of α (that an uncontaminated point passes the pre-test). Otherwise it requires $\bar{t}_\alpha$ points in average. Similarly when a 'bad' sample is provided, N points have to be verified with the probability of β that a contaminated sample successfully passes the pre-test. Else, averagely $\bar{t}_\beta$ points are verified.

Since the significance of the $T_{d,d}$ is mainly rooted in a quick rejection of contaminated samples, it is prone to reject uncontaminated samples accordingly. Although this behavior results in drawing more samples than the standard RANSAC, the performance gain is noticeable when employing fast hypothesis generation approaches such as the *Gaussian elimination* method.

3.5.2 The SPRT Test [19]

The idea behind the *Sequential Probability Ratio Test* inspired from Wald's theory [35] is to conduct a statistical test on a smaller number of data points to take an earlier decision on accepting or rejecting the generated model. The strategy used by SPRT is to optimize the verification time, yet maintaining reliability, given probability bounds on pair of hypotheses namely H_g and H_b that state whether a model is 'good' or 'bad' respectively. The Wald's likelihood ratio [35] to make such a decision is

$$\lambda_j = \prod_{r=1}^{j} \frac{p(x_r|H_b)}{p(x_r|H_g)} \quad (11)$$

In this conditional probabilities' ratio, x_r is 1 if r_{th} data point is consistent with the generated model, and 0 otherwise. In practice, $p(1|H_g)$ (the probability of a random point to be consistent with the model) is *priori* unknown and we approximate it with its lower bound ε which is equal to the best inliers ratio found so far. In addition, the probability of a random point consistent with a degenerate model $p(1|H_b) = \delta$ is a Bernoulli distributed probability function that can be estimated using average inlier's fraction of the discarded model.

The time optimization of this statistical decision making algorithm highly depends on an accepting threshold A. That is that the model is known as a degenerate model if for a specific j, λ_j in Eq. (11) becomes greater than A. The authors of [10] found the optimal solution for A by minimizing the time $t = k(t_M + \overline{m}_S t_v)$. Let k be the average number of points to be verified and $\overline{m}_S$ implies the average number of solutions found given a minimal sample set. t_m indicates the time needed to generate a model while t_v is the time to verify each sample.

3.6 The Degeneracy Test

According to the randomness of the RANSAC algorithm, a large fraction of hypotheses may be generated from degenerate configurations of the points. By considering the time needed for evaluating each degenerate hypothesis against all correspondences, one can improve efficiency by applying a cheap pre-filtering test prior to the model generation stage.

In principle, to accelerate the pose estimation algorithm, it is preferable to reject samples quickly rather than speed up the *hypothesize-and-verify* processes. In

[31], a pre-filtering test is proposed that prunes out degenerate samples by relying on the rotational and positional consistency of correspondences. This pre-test that requires local orientation of points, ensures the consistency of orientation differences between the selected points in the query frame and the corresponding points in the reference frame. Since in many practical cases, the orientation of participating points are not provided, a *Geometric Constraint* is introduced in [22] to discard samples with degenerate configurations. In the case of homography estimation, the selected samples inducing a unique plane satisfy the geometric constraint only if the points in the query frame do not violate a relative order of their correspondences in the reference frame. Let $S = \{(\mathbf{a_0}, \mathbf{a_1}), (\mathbf{b_0}, \mathbf{b_1}), (\mathbf{c_0}, \mathbf{c_1}), (\mathbf{d_0}, \mathbf{d_1})\}$ be a set of four selected points, it is proven in [22] that the relative ordering of three out of four points is held if and only if

$$\operatorname{sign}\left((\mathbf{a_0} \times \mathbf{b_0})^T \cdot \mathbf{c_0}\right) = \operatorname{sign}\left((\mathbf{a_1} \times \mathbf{b_1})^T \cdot \mathbf{c_1}\right) \tag{12}$$

Where $\times$ and $\cdot$ are cross and dot product operators respectively. A *Weak Constraint* is defined when the first three combinations of points hold Eq. (12) while a *Strong Constraint* ensures the relative ordering of all possible subsets from S.

4 Comparative Results

This section presents experimental results showing the performance of our homography estimation method. We compare it with the OpenCV SVD solution and with a non-optimized Gaussian elimination implementation. We embedded the homography estimation step into a planer target recognition application using a PROSAC hypothesize-and-verify scheme.

4.1 Homography Estimation Performance

In order to validate our homography estimation algorithm, we used the test set proposed in [25] to benchmark the USAC framework. USAC includes a number of key elements for building a computationally efficient solution and thus offers an ideal tool for benchmarking new approaches. We produced the same performance tables as in [25] in which a homography is estimated using different image pairs. The matches are provided at different level of contamination allowing better evaluation of robustness to outliers.

Referring to Table 1, the first column shows the performance we obtained using the standard USAC 1.0 framework. In the second column, we simply replaced the USAC SVD estimation by our Gaussian Elimination (GE) implementation. Very similar performances are obtained which demonstrate that GE estimation is also able to provide accurate estimates. The computational timings are also similar and this is explained by the fact that under the full USAC framework, the homography estimation stage does not represent a significant portion of the total computation. We therefore ran a new set of experiments in which we removed the more costly local optimization and symmetrical re-projection error steps as these are not necessarily required in a target recognition context. In such a case, the benefit of using GE in the estimation of the homograhy becomes apparent. Compared to SVD results (the third column), GE results (the fourth column) are 2 to 5 times faster. Finally, the last column shows the performance of our PROSAC implementation based on GE estimation of the homography.

For further evaluation, we compared in Table 2, overall performance of our framework against several variants of RANSAC aiming at real-time performance. For instance, in the first and second column, R-RANSAC [8, 19] with $T_{1,1}$ and SPRT verification are respectively evaluated. Although, these two algorithms reduced the number of verifications, they consume more time due to a larger number of samples drawn and models generated. In the next two columns, performance results for PROSAC scheme with MLE (MLESAC [32]) and standard cost function are illustrated. MLESAC has a higher computational cost due to the more expensive estimation of a mixture parameter, while PROSAC with standard cost function produces slightly different errors. The last two columns show best performing USAC selected from the previous experiment and our proposed framework. These two approaches significantly improved the execution time by decreasing the number of samples drawn and models generated. However, our proposed approach is 3x-10x faster than the USAC framework.

4.1.1 Accuracy and Recognition Rate

In order to assess the performance of our optimized framework in the context of planar target recognition, we captured four sets of image sequences for four different types of targets, with each sequence comprising of around 250 to 300 frames[1]. The image sequences were captured using an LG Optimus 2X smartphone camera with a resolution of 480×480. The camera was rotated by approximately $45°$ in all directions (i.e. $45°$ in- and out-of-plane rotation). The scale of the target varies from full resolution (where the target fully occupies the frame) to about one third the image size. A majority of the images suffer from perspective distortions and severe motion blur in some cases. The ground truth target locations were manually obtained by identifying the four corners of the target in each image of each sequence.

The matching scheme based on BRIEF described in Sect. 2 was used to match the target features with the ones detected in each frame of the test sequences. Each

[1] Available online at `www.eecs.uottawa.ca/~laganier/projects/mobilevision`

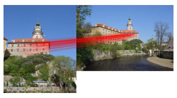
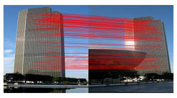
		USAC 1.0	USAC GE	USAC SVD (No LO)	USAC GE (No LO)	**our PROSAC GE**
A: $\epsilon = 0.46$, $N = 2540$	I	1147.6 ± 0.1	1147.7 ± 0.1	1074.4 ± 9.1	1017.2 ± 10.1	969.6 ± 10.2
	K	4.8 ± 0	5.9 ± 0.1	7.8 ± 0.1	9.1 ± 0.2	8.4 ± 0.2
	K_rej	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	models	4.8 ± 0	5.9 ± 0.1	7.8 ± 0.1	9.1 ± 0.2	8.4 ± 0.1
	VPM	755.6 ± 15.6	667 ± 16.4	1021.6 ± 16.6	869.3 ± 16.1	1193.5 ± 14.8
	error	1.27	1.27	1.18	2.22	2.27
	time(ms)	24.78	24.4	**0.4494**	**0.3477**	**0.0810**
B: $\epsilon = 0.15$, $N = 514$	I	68.1 ± 0.0	68.0 ± 0.0	67.7 ± 0.5	61.5 ± 0.9	64.3 ± 0.4
	K	925 ± 316	14557 ± 3676	57.0 ± 11.9	165.7 ± 23.0	13.6 ± 0.4
	K_rej	711.2 ± 263.8	12446.8 ± 3226	35.2 ± 10.2	128.0 ± 19.7	3.0 ± 0.1
	models	214.1 ± 53.7	2104.8 ± 451.3	21.8 ± 1.8	36.4 ± 3.3	10.6 ± 0.3
	VPM	49 ± 1.4	42.4 ± 2.3	29.6 ± 2.1	100.4 ± 3.6	294.3 ± 3.6
	error	0.87	0.87	2.08	2.35	2.38
	time(ms)	4.93	3.78	**0.2873**	**0.07323**	**0.02511**
C: $\epsilon = 0.23$, $N = 1317$	I	301.0 ± 0.0	300.56 ± 0.3	211.4 ± 1.2	210.9 ± 1.3	202.9 ± 1.4
	K	4.8 ± 0.1	7.5 ± 0.3	4.7 ± 0.1	6.3 ± 0.2	5.0 ± 0.1
	K_rej	0.3 ± 0.0	0.5 ± 0.0	0.3 ± 0.0	0.4 ± 0.1	2.3 ± 0.0
	models	4.5 ± 0.1	4.9 ± 0.3	4.4 ± 0.1	3.9 ± 0.2	2.7 ± 0.1
	VPM	372.6 ± 4.5	593.5 ± 17.5	435.1 ± 6.6	694.9 ± 16.8	1215.7 ± 7.8
	error	0.80	0.8	0.98	1.42	1.35
	time(ms)	6.33	6.3	**0.1127**	**0.07363**	**0.03406**
D: $\epsilon = 0.34$, $N = 495$	I	146.2 ± 0.1	146.3 ± 0.1	137.0 ± 1.0	139.6 ± 1.1	136.7 ± 1.2
	K	14.0 ± 0.4	16 ± 0.5	5.1 ± 0.1	5.8 ± 0.1	5.5 ± 0.1
	K_rej	3.7 ± 0.1	4.2 ± 0.2	1.9 ± 0.0	2.0 ± 0.0	2.7 ± 0.0
	models	10.3 ± 0.3	10.8 ± 0.4	3.2 ± 0.1	3.0 ± 0.1	2.8 ± 0.1
	VPM	103.4 ± 2.3	111.4 ± 3.4	307.3 ± 5.4	342.1 ± 5.9	482.4 ± 1.5
	error	1.16	1.16	5.72	5.70	5.87
	time(ms)	2.73	2.68	**0.07764**	**0.04241**	**0.016903**

Table 1: Performance result of Homography estimation as in [25]. (I) is the number of inliers found. (K) and (K_rej) are the number of samples drawn and the number of samples rejected by the degeneracy test. (models) is the number of total hypotheses, (VPM) the number of verifications per model. The symmetrical reprojection (error) is measured w.r.t. the ground truth. (time) indicates the execution time per frame in ms. Note that all reported results are averaged over a total of 500 runs.

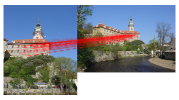
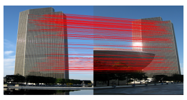
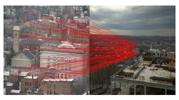
		R-RANSAC ($T_{1,1}$)	R-RANSAC (SPRT)	MLE-SAC	Classic PROSAC	USAC (No LO)	**our PROSAC GE**
A: $\epsilon = 0.46$, $N = 2540$	I	1329.7 ± 2.6	1356.4 ± 1.8	1027.9 ± 9.5	1021.5 ± 9.6	1074.4 ± 9.1	969.6 ± 10.2
	K	62.8 ± 0.7	55.9 ± 0.4	7.5 ± 0.1	7.4 ± 0.1	7.8 ± 0.1	8.4 ± 0.2
	K_rej	0.3 ± 0.0	0.3 ± 0.0	0 ± 0	0 ± 0	0.0 ± 0.0	0.0 ± 0.0
	models	62.5 ± 0.7	55.6 ± 0.3	7.5 ± 0.1	7.4 ± 0.1	7.8 ± 0.1	8.4 ± 0.1
	VPM	493.4 ± 5.9	889.0 ± 5.7	2540 ± 0	2540 ± 0	1021.6 ± 16.6	1193.5 ± 14.8
	error	2.04	2.01	2.22	2.24	1.18	2.27
	time(ms)	**8.4516**	**6.9208**	**155.965**	**1.2111**	**0.4494**	**0.0810**
B: $\epsilon = 0.15$, $N = 514$	I	55.9 ± 1.0	75.3 ± 0.1	70.5 ± 0.2	70.7 ± 0.2	67.7 ± 0.5	64.3 ± 0.4
	K	2000 ± 0	2000 ± 0.0	16.1 ± 1.5	19.3 ± 4.0	57.0 ± 11.9	13.6 ± 0.4
	K_rej	5.2 ± 0.1	5.3 ± 0.1	2.7 ± 0.0	2.6 ± 0.1	35.2 ± 10.2	3.0 ± 0.1
	models	1994.8 ± 0.1	1994.8 ± 0.1	13.4 ± 1.5	16.7 ± 4.0	21.8 ± 1.8	10.6 ± 0.3
	VPM	6.0 ± 0.0	20.4 ± 0.0	514 ± 0	514 ± 0	29.6 ± 2.1	294.3 ± 3.6
	error	420.0	1.68	2.07	2.06	2.08	2.38
	time(ms)	**251.354**	**191.439**	**58.844**	**1.8423**	**0.2873**	**0.02511**
C: $\epsilon = 0.23$, $N = 1317$	I	283.0 ± 1.0	299.7 ± 0.2	202.4 ± 1.1	202.0 ± 1.1	211.4 ± 1.2	202.9 ± 1.4
	K	1861.7 ± 6.5	1713.7 ± 4.1	3.8 ± 0.0	3.8 ± 0.1	4.7 ± 0.1	5.0 ± 0.1
	K_rej	10.7 ± 0.1	10.8 ± 0.1	2 ± 0.0	2.0 ± 0.0	0.3 ± 0.0	2.3 ± 0.0
	models	1851.0 ± 6.5	1702.9 ± 4.1	1.8 ± 0.0	1.8 ± 0.0	4.4 ± 0.1	2.7 ± 0.1
	VPM	17.0 ± 0.1	72.7 ± 0.3	1317 ± 0	1317 ± 0	435.1 ± 6.6	1215.7 ± 7.8
	error	1.30	1.2	0.99	0.98	0.98	1.35
	time(ms)	**229.184**	**186.348**	**21.6036**	**0.31071**	**0.1127**	**0.03406**
D: $\epsilon = 0.34$, $N = 495$	I	179.3 ± 0.0	179.8 ± 0.1	136.7 ± 1.1	137.4 ± 1.1	137.0 ± 1.0	136.7 ± 1.2
	K	266.3 ± 0.2	263.5 ± 0.1	5.0 ± 0.1	5.0 ± 0.1	5.1 ± 0.1	5.5 ± 0.1
	K_rej	4.1 ± 0.1	3.9 ± 0.1	2.0 ± 0.0	2.1 ± 0.0	1.9 ± 0.0	2.7 ± 0.0
	models	262.2 ± 0.2	259.6 ± 0.1	3.0 ± 0.1	2.9 ± 0.1	3.2 ± 0.1	2.8 ± 0.1
	VPM	47.0 ± 0.4	142.2 ± 0.4	495 ± 0	495 ± 0	307.3 ± 5.4	482.4 ± 1.5
	error	3.31	3.36	5.76	5.80	5.72	5.87
	time(ms)	**33.265**	**29.2401**	**11.7479**	**0.34246**	**0.07764**	**0.016903**

Table 2: Performance result of Homography estimation for different RANSAC variants.

Target	Total matches	Total Inliers		Iterations		Recognition rate(%)	
		GE	SVD	GE	SVD	GE	SVD
Book	169.0 ± 33.1	67.1 ± 41.3	61.0 ± 34.7	756.0 ± 710.6	772.0 ± 722.5	48.82	45.40
Map	75.3 ± 18.7	38.1 ± 17.2	38.6 ± 17.3	317.4 ± 546.8	299.4 ± 526.3	72.24	74.75
Football	232.7 ± 41.9	82.2 ± 44.8	79.8 ± 40.8	747.9 ± 723.7	742.2 ± 717.4	84.58	79.06
Adv	200.8 ± 52.8	74.8 ± 41.0	83.0 ± 39.6	693.1 ± 726.8	604.1 ± 661.0	88.09	90.49
Average	175.6 ± 65.6	67.5 ± 41.6	67.3 ± 38.9	658.8 ± 709.7	635.4 ± 694.0	**73.44**	**72.56**

Table 3: Average number of total matches, inliers, required iterations and recognition rate are shown for the four targets with both GE and SVD method.

matching set obtained is then fed to our PROSAC estimator in order to obtain a putative homography. The same experiments were repeated for the different homography estimation methods, all of them using the same initial match sets.

Table 3 shows the number of matches in the initial set and the number of matches in the final set with best support as found by PROSAC. We report these results for the SVD solution (as implemented in OpenCV) and for our Gaussian elimination implementation.

The recognition rate is determined by analyzing the maximum error between the estimated target corner locations to the corresponding ground truth corner location. This error, given in pixels, is obtained as follows:

$$\mathcal{E}_i(\tilde{C}) = \max_j \|H_i \hat{p}_j - \tilde{p}_{ij}\|, 1 \leq j \leq 4 \;, \tag{13}$$

where H_i is the estimated homography at frame i, $\hat{p}_j$ is the coordinate of target corner j in the reference frame and $\tilde{p}_{ij}$ is the manually obtained location of corner j in frame i.

If we consider, from empirical observations, that a target is successfully detected if $\mathcal{E}(\tilde{C}) \leq 10$ pixels, we then obtain a recognition rate of 72.56% for SVD and 73.44% for Gaussian elimination averaged over four test sequences (last column of Table 3). Here we empirically chose 10 pixels to provide a representative measure of recognition rate to compare the two algorithms (the exact value is not crucial). Individual results corresponding to each row of Table 3 are plotted in Fig. 2.

To illustrate the behavior of the two tested homography estimation methods, we show in Fig. 3 the evaluation of the maximal positional error (reported every 5 frames) for one of the test sequences. As it can be seen, except for one large error made by Gaussian elimination, both estimation schemes exhibit very similar behavior. Figure 4 correspondingly illustrates results of homography estimates with some failure cases.

Finally, we also evaluated the accuracy of our GE homography estimation module outside the PROSAC loop. To this end, we performed the experiments reported in [23] that were used to validate the accuracy of their hyper-accurate homography estimation solution. Five pairs of matching were identified in two image sets, shown in Fig. 5, from which homographies were estimated. We used OpenCV `cv::findHomography` function based on SVD and our GE solution to estimate these two homographies. From Table 4, as expected, our raw solution initially gave a larger error but after Levenberg-Marquardt refinement, the two estimation schemes produced comparable errors.

4.2 Target Recognition Performance

4.2.1 Computational Efficiency

We report in this section the global computational efficiency of different homography estimation methods in the context of robust target recognition. Speed is here measured in count of instruction fetches. For completeness, we evaluate the performance of different methods under different contexts; the results are shown in Fig. 6. First, we measured the speed of the OpenCV `cv::findHomography` function (version 2.4) under the RANSAC mode. We also built our own implementation of the RANSAC scheme inside which we used the OpenCV 2.4 SVD function. We then integrated the same OpenCV 2.4 function under our PROSAC implementation. We also tested the DEGSVD function from the LAPACK package. We also tested a publicly available but non-optimized Gaussian elimination implementation [2]. Finally, the last bar shown in Fig. 6 is the one obtained by our proposed optimized PROSAC Gaussian elimination scheme. For a device equipped with a 2.26 GHz CPU, a $30fps$ detection rate corresponds to a maximum number of about 75 millions of cycles. Note that all these algorithms have a complexity of $O(KN)$, with K being the number of iterations and N being the number of points. Our linear gain in computational efficiency can be attributed to our speed of iterations that is the main bottleneck in most methods.

We have also compared the speed of our homography estimation with available results as reported in [22]. On the basis of our results, the run-time for homography estimation is on the order of 5 milliseconds for the approach used in [22], while our optimized framework performs nearly 25x faster on average. Figure 7 illustrates the corresponding per-frame numbers for putative matches,

[2] Available online at `https://github.com/camilosw/ofxVideoMapping`

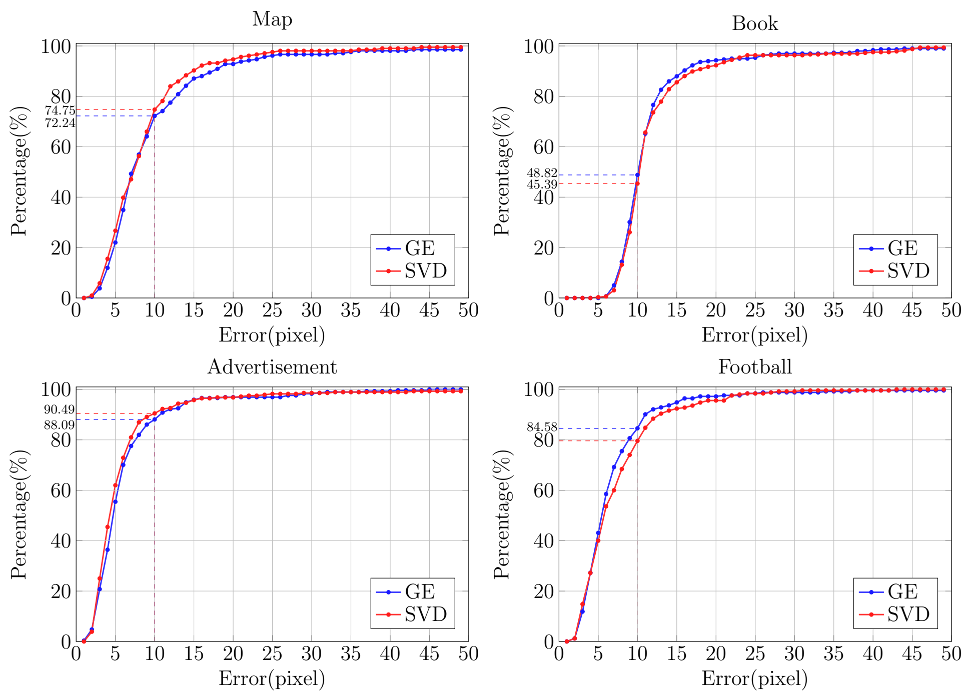


Fig. 2: Recognition rate for each test sequence, reported as the percentages of frames with maximal positional error less than 10 pixels.

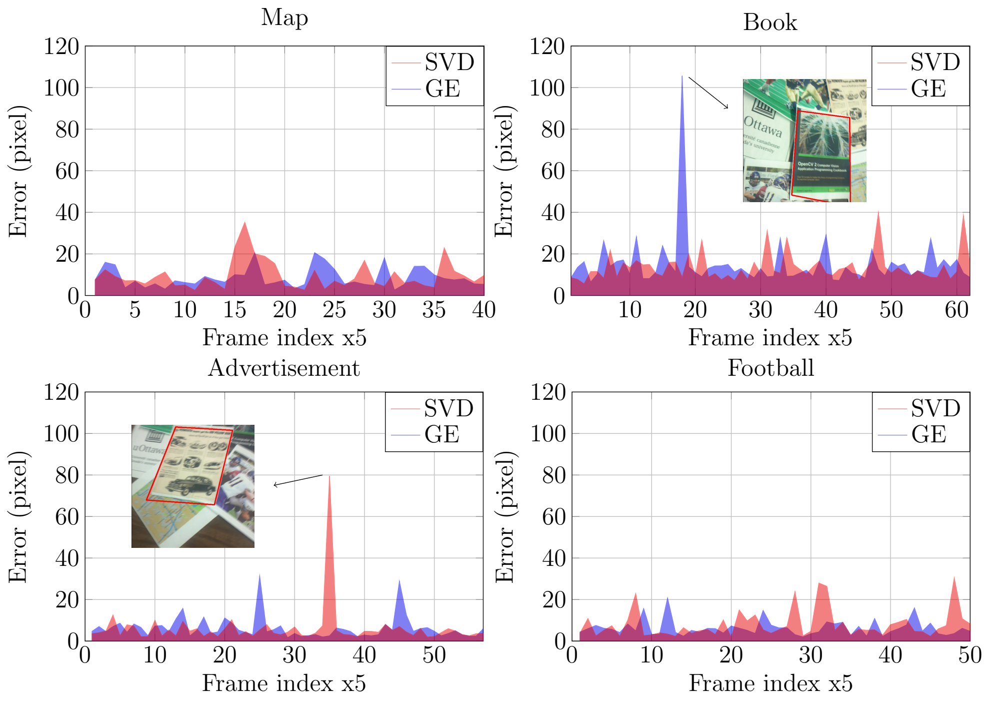


Fig. 3: Maximum positional error of the four sequences reported every 5 frames.

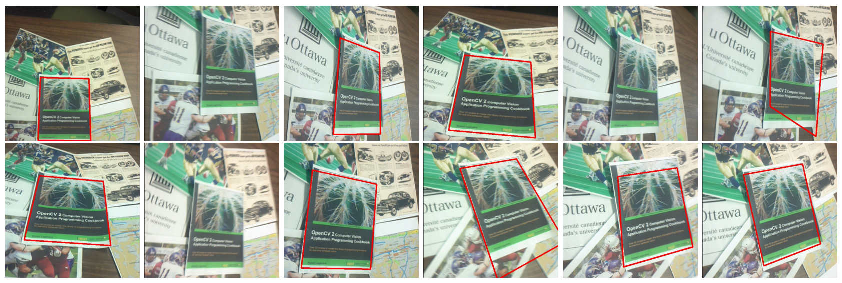


Fig. 4: Homography estimates for the sequence of images corresponding to Fig. 3. Images with no bounding box represent some failure cases.

Target	OpenCV method		GE method	
	Non-Refined	Refined	Non-refined	Refined
Food	0.00729	0.00284	0.51133	0.00289
Map	0.00110	0.00248	0.19794	0.00251

Table 4: Frobenius norm of the difference between estimated homography and hyper-accurate homography as in [23].

(a) Food

(b) Map

Fig. 5: Two image sets used for accuracy evaluation as in [23].

inliers ratio and processing time. In Fig. 8, several estimates of homography are depicted for the sequence of images used in this experiment.

4.2.2 Performance Results of the Model Verification

To compare the performance of the verification stage, we repeated the previous tests for different verification methods (see Table 5). The GE approach that obtained the best performance in the previous test was also used in this new test. The results of the first and the third columns show that the number of verifications per model (VPM) is considerably decreased from the standard approach to the verification approach based on $T_{1,1}$. But because the number of generated models is slightly larger than in the case of the standard verification approach, the USAC framework with $T_{1,1}$ does not achieve significant speed gain. The same reasoning applies to PROSAC with the standard and $T_{1,1}$ verifications. The only difference is that our highly optimized PROSAC framework performs 2-4 times faster than the USAC framework. The last two columns of the table demonstrate that the verification based on SPRT speeds up the process by a factor of 5 to 10 compared with the standard verification approach. Additionally, SPRT returns more inliers in a smaller number of hypotheses compared with the $T_{1,1}$ test.

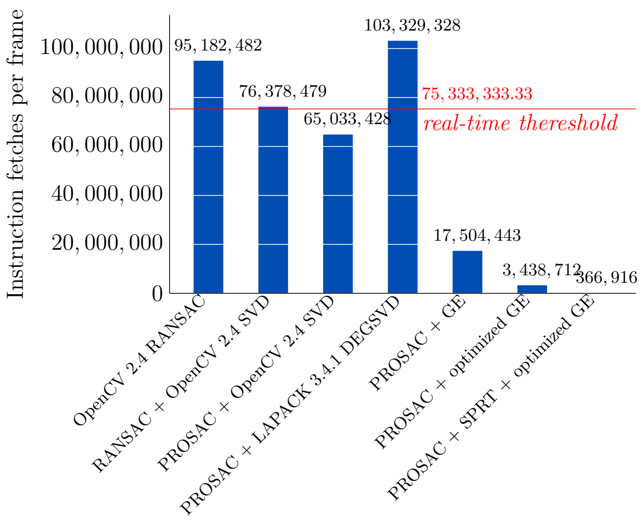


Fig. 6: Per-frame average instruction fetch count for each H-estimator.

4.2.3 Influence of Termination Criteria

As we pointed out earlier, the stopping condition for the iterative hypothesize-and-verify scheme has a significant effect on the computational speed of the process. The fitness of the sought model also depends on the stopping condition. According to the semi-random sampling strategy of the PROSAC scheme, we altered the maximality termination criterion by imposing a non-randomness constraint that yields a faster convergence. To carry out a thorough evaluation, we compared our proposed method with an intuitive termination condition proposed in [31]. This condition determines if the ratio between good inliers (within $2px$) to close inliers (within $15px$) is above a predefined threshold (set to 0.65-0.75). The corresponding results are tabulated in Table 6. It can be seen that the number of samples drawn in column 1 with the maximality constraint is significantly larger than of the ratio and non-randomness constraints (column 2-5).

As a consequence, execution times for the ratio and non-randomness constraints have been speeded up by a factor ranging between 3x-30x. Comparing the results of column 2 and 3 with column 1, indicates that a termination criteria based on the ratio test require much fewer samples to be drawn. However, the number of generated models are still noticeably larger than the non-randomness approaches. This is due to the fact that, there is no statistical analysis behind the ratio test regarding the quality of models. Indeed, the aim of adding ratio tests to Table 6 is to provide another early termination criterion as a reference to better evaluate the non-randomness constraint. It can also be concluded that the criterion based on χ^2 approximation performs quantitatively similarly to the non-randomness one.

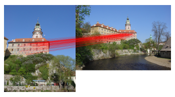
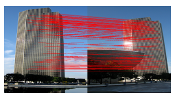
		USAC STD_Verif	PROSAC STD_Verif	USAC $T_{1,1}$	PROSAC $T_{1,1}$	USAC SPRT	**PROSAC SPRT**
A: $\epsilon = 0.46$,	I	1015.7 ± 10.0	957.5 ± 10.4	986.7 ± 11.2	973.9 ± 12.1	1017.2 ± 10.1	969.6 ± 10.2
$N = 2540$	K	8.8 ± 0.2	8.5 ± 0.2	14.2 ± 0.4	15.3 ± 0.4	9.1 ± 0.2	8.4 ± 0.2
	K_rej	0.0 ± 0.0	0.0 ± 0.0	0.1 ± 0.0	0.1 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	models	$\mathbf{8.8 \pm 0.2}$	$\mathbf{8.5 \pm 0.2}$	$\mathbf{14.1 \pm 0.4}$	$\mathbf{15.2 \pm 0.4}$	$\mathbf{9.1 \pm 0.2}$	$\mathbf{8.4 \pm 0.2}$
	VPM	2540 ± 0	2540 ± 0	488.3 ± 15.5	371.9 ± 9.0	869.3 ± 16.1	1193.5 ± 14.8
	error	2.23	2.63	2.31	1.13 ± 0.0	2.22	2.27
	time(ms)	**0.70**	**0.117412**	**0.232**	**0.070258**	**0.3477**	**0.08103**
B: $\epsilon = 0.15$, $N = 514$	I	58.1 ± 0.5	72.8 ± 0.1	47.1 ± 1.0	39.0 ± 1.5	61.5 ± 0.9	64.3 ± 0.4
	K	11.1 ± 0.1	69.74 ± 13.1	348.0 ± 31.4	411.2 ± 42.4	165.7 ± 23.0	13.6 ± 0.4
	K_rej	1.5 ± 0.0	44.824 ± 11.2	276.0 ± 26.9	361.1 ± 36.5	128.0 ± 19.7	3.0 ± 0.1
	models	$\mathbf{8.3 \pm 0.2}$	$\mathbf{24.916 \pm 1.9}$	$\mathbf{70.5 \pm 4.5}$	$\mathbf{97.1 \pm 5.9}$	$\mathbf{36.4 \pm 3.3}$	$\mathbf{10.6 \pm 0.3}$
	VPM	514 ± 0	514 ± 0	44.2 ± 2.3	51.1 ± 1.4	100.4 ± 3.6	294.3 ± 3.6
	error	2.45	2.50	2.80	3.87	2.35	2.4
	time(ms)	**0.12**	**0.05374**	**0.073427**	**0.080152**	**0.07323**	**0.025110**
C: $\epsilon = 0.23$,	I	205.9 ± 1.3	204.5 ± 1.2	192.1 ± 2.0	206.9 ± 1.5	210.9 ± 1.3	202.9 ± 1.4
$N = 1317$	K	4.9 ± 0.1	4.6 ± 0.1	20.4 ± 1.3	30.7 ± 4.5	6.3 ± 0.2	5.0 ± 0.1
	K_rej	0.3 ± 0.0	2.3 ± 0.1	2.2 ± 0.5	9.6 ± 2.4	0.4 ± 0.1	2.3 ± 0.1
	models	$\mathbf{2.7 \pm 0.1}$	$\mathbf{2.4 \pm 0.1}$	$\mathbf{15.4 \pm 0.8}$	$\mathbf{21.1 \pm 2.1}$	$\mathbf{3.9 \pm 0.2}$	$\mathbf{2.7 \pm 0.1}$
	VPM	1317	1317 ± 0	295.9 ± 14.6	300.8 ± 14.4	694.9 ± 16.8	1215.7 ± 7.8
	error	1.41	1.30	1.53	1.57	1.42	1.35
	time(ms)	**0.12**	**0.03927**	**0.070015**	**0.038238**	**0.07363**	**0.034061**
D: $\epsilon = 0.34$, $N = 495$	I	138.0 ± 1.0	138.1 ± 1.1	142.5 ± 1.4	136.7 ± 1.2	139.6 ± 1.1	136.7 ± 1.2
	K	5.8 ± 0.1	5.7 ± 0.1	10.1 ± 0.3	5.5 ± 0.1	5.8 ± 0.1	5.5 ± 0.1
	K_rej	2.0 ± 0.1	2.7 ± 0.0	2.7 ± 0.1	2.7 ± 0.0	2.0 ± 0.0	2.7 ± 0.0
	models	$\mathbf{3.0 \pm 0.1}$	$\mathbf{2.9 \pm 0.1}$	$\mathbf{6.5 \pm 0.2}$	$\mathbf{2.8 \pm 0.1}$	$\mathbf{3.0 \pm 0.1}$	$\mathbf{2.8 \pm 0.1}$
	VPM	495	495 ± 0	159.9 ± 5.7	482.4 ± 1.5	342.1 ± 5.9	482.4 ± 1.5
	error	5.72	5.74	5.40	5.87	5.70	5.87
	time(ms)	**0.05**	**0.01957**	**0.030765**	**0.016903**	**0.04241**	**0.016903**

Table 5: Performance result of Homography estimation for different verification methods.

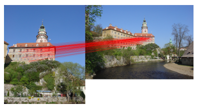
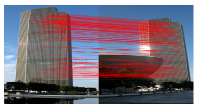
		PROSAC Maximality	PROSAC Ratio= 0.65	PROSAC Ratio= 0.75	PROSAC Non-randomness	**PROSAC** χ^2
A: $\epsilon = 0.46$,	I	1326.4 ± 2.4	1241.7 ± 3.5	1314.6 ± 2.3	969.6 ± 10.2	967.2 ± 10.4
$N = 2540$	K	62.7 ± 0.6	17.7 ± 0.7	36.8 ± 1.3	8.4 ± 0.2	8.7 ± 0.2
	K_rej	0.9 ± 0.0	0.1 ± 0.0	0.5 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	models	$\mathbf{61.8 \pm 0.5}$	$\mathbf{17.6 \pm 0.7}$	$\mathbf{37.2 \pm 1.3}$	$\mathbf{8.4 \pm 0.2}$	$\mathbf{8.7 \pm 0.2}$
	VPM	741.7 ± 5.6	1138.5 ± 14.3	970.2 ± 13.9	1193.5 ± 14.8	1192.7 ± 14.5
	error	2.06	2.11	2.00	2.27	2.66
	time(ms)	0.16379	0.09050	0.13150	**0.08103**	**0.05638**
B: $\epsilon = 0.15$, $N = 514$	I	74.1 ± 0.2	65.4 ± 0.3	70.4 ± 0.2	64.3 ± 0.4	72.5 ± 0.2
	K	2000 ± 0	21.8 ± 5.6	42.7 ± 9.8	13.6 ± 0.4	107.2 ± 17.8
	K_rej	1698.93 ± 0.7	10.3 ± 4.8	25.5 ± 8.3	3.0 ± 0.1	77.0 ± 15.2
	models	$\mathbf{301.1 \pm 0.7}$	$\mathbf{12.4 \pm 0.9}$	$\mathbf{18.3 \pm 1.5}$	$\mathbf{10.6 \pm 0.3}$	$\mathbf{30.3 \pm 2.6}$
	VPM	49.7 ± 0.2	289.4	281.1 ± 3.5	294.3 ± 3.6	267.0 ± 3.6
	error	1.94	2.38	2.29	2.4	3.11
	time(ms)	0.2250	0.02552	0.03190	**0.025110**	**0.036557**
C: $\epsilon = 0.23$,	I	296.1 ± 0.4	272.9 ± 0.6	295.4 ± 0.4	202.9 ± 1.4	202.6 ± 1.1
$N = 1317$	K	1773.7 ± 5.6	54.7 ± 6.3	708.5 ± 34.8	5.0 ± 0.1	4.6 ± 0.1
	K_rej	1049.7 ± 3.8	19.1 ± 3.6	393.7 ± 21.4	2.3 ± 0.1	2.1 ± 0.1
	models	$\mathbf{723.9 \pm 2.1}$	$\mathbf{36.6 \pm 2.8}$	$\mathbf{315.6 \pm 13.4}$	$\mathbf{2.7 \pm 0.1}$	$\mathbf{2.4 \pm 0.1}$
	VPM	119.6 ± 0.5	890.7 ± 12.6	381.4 ± 10.8	1215.7 ± 7.8	1244.9 ± 6.5
	error	1.40	1.51	1.40	1.35	1.28
	time(ms)	0.4878	0.093214	0.32967	**0.034061**	**0.018477**
D: $\epsilon = 0.34$, $N = 495$	I	179.7 ± 0.0	149.5 ± 0.8	160.1 ± 0.6	138.4 ± 1.1	138.4 ± 1.1
	K	263.8 ± 0.1	5.3 ± 0.1	6.8 ± 0.2	5.5 ± 0.1	5.5 ± 0.1
	K_rej	140.2 ± 0.3	2.8 ± 0.1	3.1 ± 0.1	2.7 ± 0.0	2.7 ± 0.0
	models	$\mathbf{123.6 \pm 0.3}$	$\mathbf{3.5 \pm 0.1}$	$\mathbf{4.6 \pm 0.1}$	$\mathbf{2.7 \pm 0.1}$	$\mathbf{2.7 \pm 0.1}$
	VPM	259.2 ± 0.8	477.2 ± 1.7	460.2 ± 2.4	484.3 ± 1.4	483.9 ± 1.4
	error	3.58	5.50	5.10	5.89	5.9
	time(ms)	0.1343	0.015431	0.017222	**0.76780**	**0.011761**

Table 6: Performance result of Homography estimation for different stopping criteria.

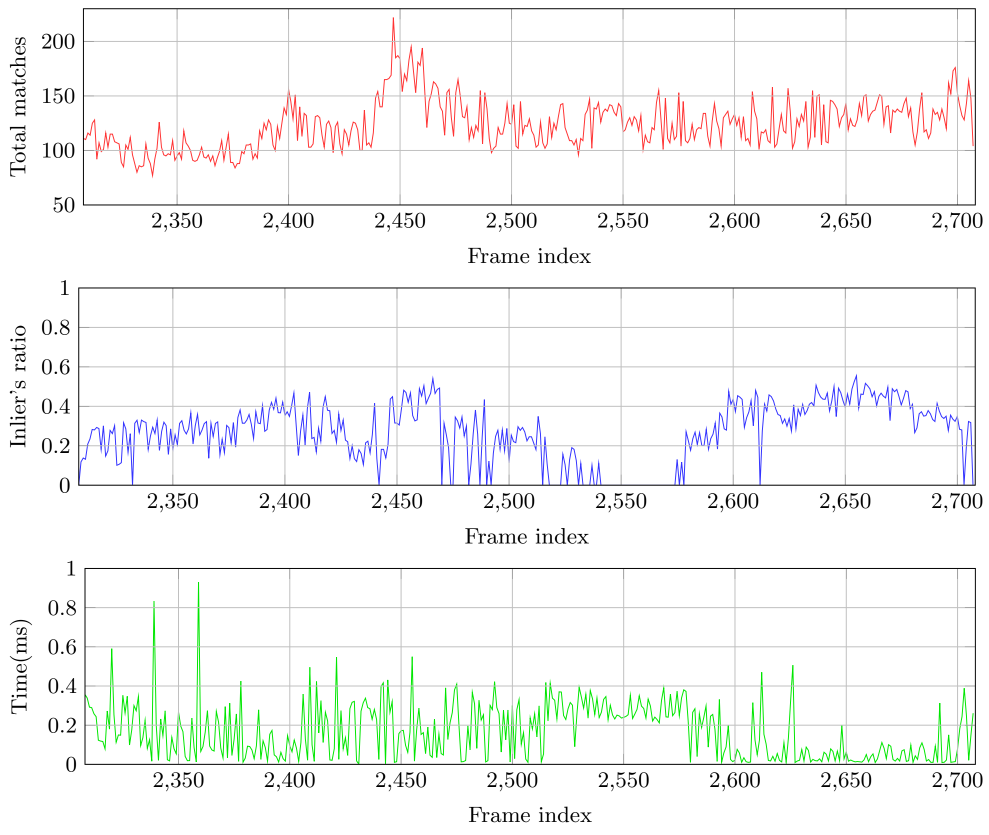


Fig. 7: Shows number of extracted matches, inliers ratio (Total inliers vs. total matches) and homography estimation time (ms) for each frame related to Fig. 8 experiment. Interested readers should refer to [22] for comparisons.

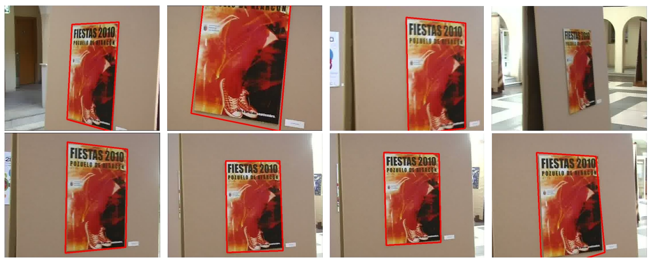


Fig. 8: Result of homography estimates (shown in red) for the sequence of images used in [22].

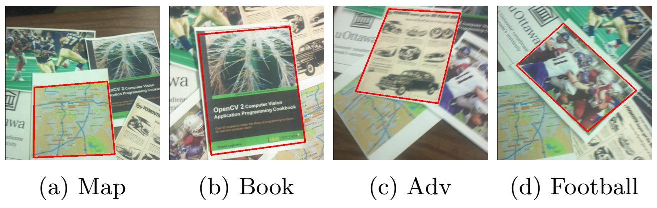


Fig. 9: Homography estimation for one frame of each of our test videos.

4.2.4 Non-randomness vs. Chi-squared Approximation

We have demonstrated in Table 6 the effect of different stopping conditions for still images. However, in the case of target recognition in a live video, the difference between computational timing of the non-randomness and Chi-squared approaches becomes more significant.

Therefore, we repeated the homography estimation test for one of the test sequences of Fig. 9. Table 7 shows, for each algorithm of Sect. 4.2.3, the number of inliers, the number of generated models, maximum positional error (in pixels) and the run-time (all reported on average). Except for the ratio test, a very similar accuracy is obtained for the rest of algorithms. Despite the significant reduction in the number of hypothesized models for the non-randomness algorithm, this one exhibits a higher execution time. This additional computational cost can be explained by the recursive process needed for computing binomial probabilities and optimal value of n^* (the minimum subset size satisfying Eq. (8)). This demonstrates the impact of the Chi-squared approximation that saves compute time by calculating n^* only once, prior to the PROSAC loop.

Figure 10 illustrates the maximal positional error associated with each frame as a measure of accuracy for

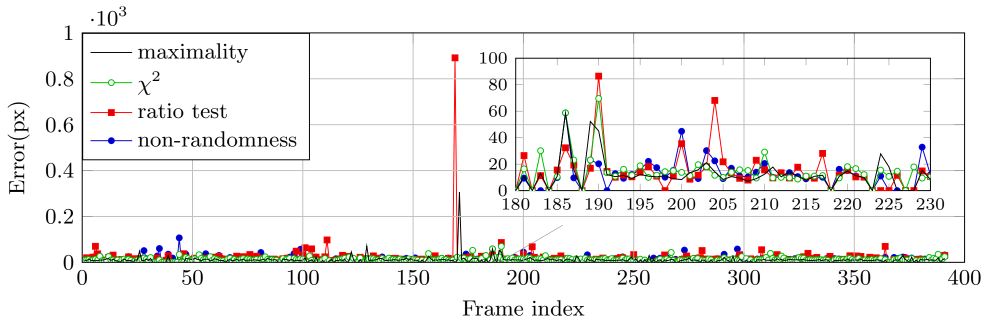


Fig. 10: Maximal positional error(px) for the homography estimation with different stopping criteria.

	Maximality	Non-randomness	χ^2	Ratio test
Inliers	43.56	38.6	39.3	31.2
Models	**160.8**	**70.4**	**72.2**	**111.5**
Recognition(%)	46.2	34.91	34.81	21.03
Error(pixel)	10.50	11.92	10.81	14.61
Time(ms)	0.1064	**0.1573**	**0.0648**	0.0814

Table 7: Performance results of the four sequences with different stopping criteria.

each algorithm. Although, a very similar range of errors is obtained, the non-randomness and χ^2 approaches exhibit better performances by reducing the peaks induced by the maximality and ratio test approaches. This closely corresponds to the standard deviation values reported in Table 7.

For different stopping criteria, we also evaluated per-frame execution time that is illustrated in Fig. 11. The first plot shows the computational timings for all aforementioned algorithms. It is evident from the second and the third plots that the χ^2 algorithm performs faster than the maximality and non-randomness algorithms. It can also be seen from the results that except for the non-randomness algorithm, other stopping conditions do not bring noticeable overhead to the PROSAC loop.

5 Conclusion

In this paper, we proposed a fast framework for robust homography estimation that can efficiently run under resource-constrained platforms. This framework profits from the non-uniform sampling approach of PROSAC and approximates the non-randomness stopping criterion using the χ^2 statistical test. The verification stage employs the Sequential Probability Ratio Test to improve the overall performance. Since the estimation step is repeated many times in the hypothesize-and-verify scheme, we presented an algebraic solution for plane-to-plane homography estimation relying on the well-known Gaussian elimination algorithm. We showed through experiments that this simplified approach significantly reduces the computational load for a real-time implementation, with an accuracy comparable to the one obtained by the more conventional SVD solution.

As a consequence of this work, individual recent developments made by different researchers are here considered together. Several adaptations to the existing algorithms are undertaken, yielding a unified framework for robust pose estimation. Out of consideration for the processing capabilities of modern CPUs, methods that can be optimized for throughput with SIMD vector instruction sets were strongly preferred. Although we do make use of SIMD vectorization, this is entirely optional; Our implemented optimization strategies are mostly algorithmic and are well-aligned with the computing capacity of all modern CPUs. The highly-optimized framework that we have presented leverages state-of-the-art algorithms and could be extended to an even broader range of fitting problems. The framework is now part of OpenCV 3.0, under the `cv::findHomography` function, method flag `RHO`.

References

1. Sharat Saurabh Akhoury and Robert Laganière. Training binary descriptors for improved robustness and efficiency in real-time matching. In Alfredo Petrosino, editor, *ICIAP 2013*, volume 8157 of *Lecture Notes in Computer Science*, pages 288–298. 2013.
2. A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517, June 2012.
3. A. Barclay and H. Kaufmann. Ft-ransac: Towards robust multi-modal homography estimation. In *Pattern Recognition in Remote Sensing (PRRS), 2014 8th IAPR Workshop on*, pages 1–4, Aug 2014.
4. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3): 346 – 359, 2008.
5. Jeffrey S Beis and David G Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE*

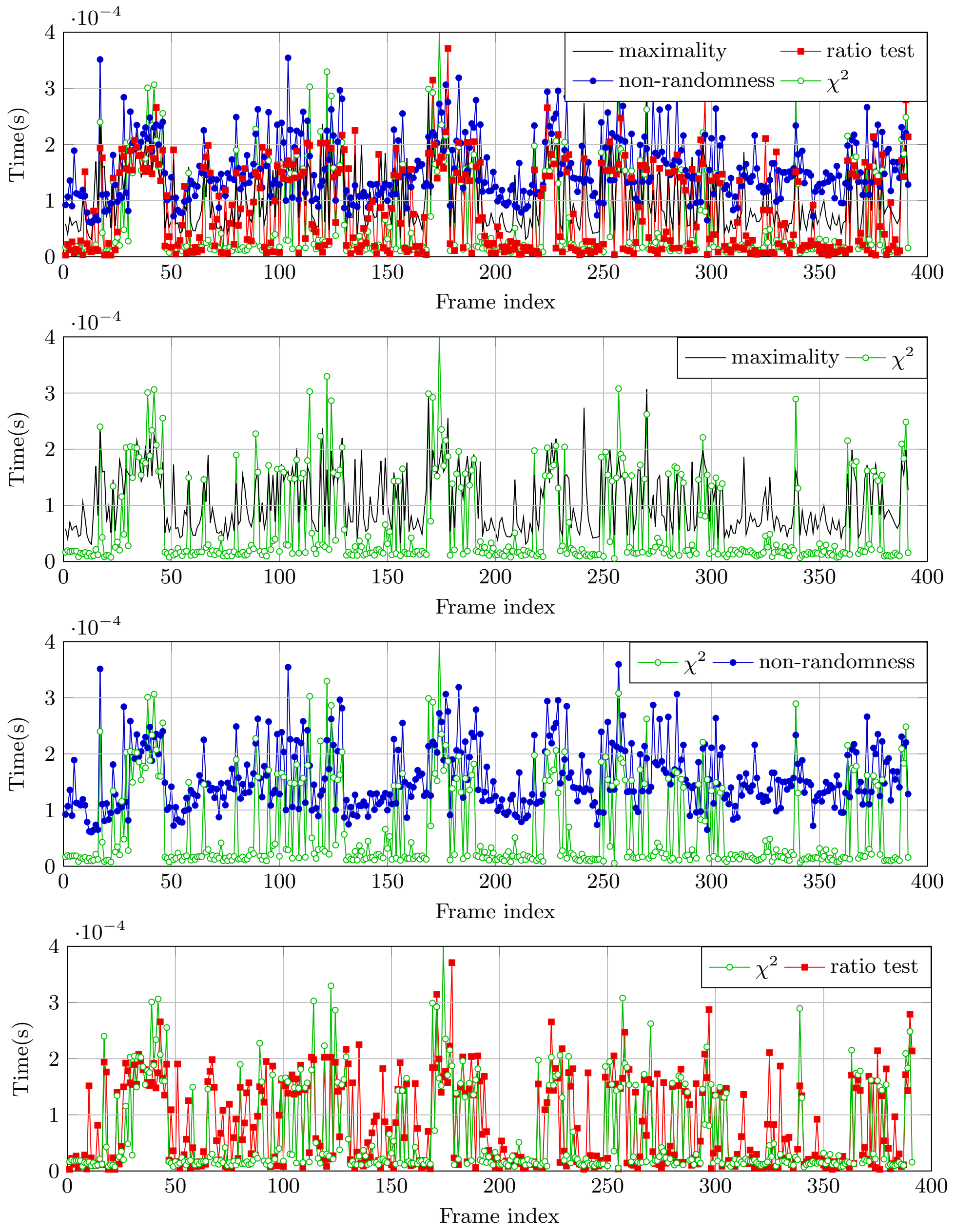


Fig. 11: Execution time for the homography estimation with different stopping criteria for one of the test sequences presented in Fig. 9.

Computer Society Conference on, pages 1000–1006. IEEE, 1997.

6. O. Bilaniuk, H. Bazargani, and R. Laganiere. Fast target recognition on mobile devices: Revisiting gaussian elimination for the estimation of planar homographies. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 119–125, June 2014.
7. Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *ECCV 2010*, volume 6314 of *Lecture Notes in Computer Science*, pages 778–792. Springer Berlin Heidelberg, 2010.
8. O. Chum and J. Matas. Randomized ransac with t d,d test. In *IMAGE AND VISION COMPUTING*, pages 448–457, 2002.
9. O. Chum and J. Matas. Matching with prosac - progressive sample consensus. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 220–226 vol. 1, June 2005.

10. O. Chum and J. Matas. Optimal randomized ransac. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(8):1472–1482, Aug 2008.
11. Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
12. Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.
13. Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM. ISBN 0-89791-962-9.
14. Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–506. IEEE, 2004.
15. V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1465–1479, Sept 2006.
16. V. Lepetit, J. Pilet, and P. Fua. Point matching as a classification problem for fast and robust object pose estimation. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–244–II–250 Vol.2, June 2004.
17. S. Leutenegger, M. Chli, and R.Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555, Nov 2011.
18. David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2): 91–110, 2004.
19. J. Matas and O. Chum. Randomized ransac with sequential probability ratio test. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1727–1732 Vol. 2, Oct 2005.
20. Jiri Matas and Ondrej Chum. Randomized ransac. In *VIENNA UNIVERSITY OF TECHNOLOGY*, pages 49–58, 2002.
21. I.F. Mondragòn, P. Campoy, C. Martinez, and M.A. Olivares-Mèndez. 3d pose estimation based on planar object tracking for uavs control. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 35–41, May 2010.
22. Pablo Mrquez-Neila, Javier Lpez-Alberca, JosM. Buenaposada, and Luis Baumela. Speeding-up homography estimation in mobile devices. *Journal of Real-Time Image Processing*, pages 1–14, 2013.
23. Hirotaka Niitsuma, Prasanna Rangarajan, and Kenichi Kanatani. High accuracy homography computation without iterations. In *Proc. 16th Symp. Sensing Imaging Inf.(SSII10)*, 2010.
24. M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(3):448–461, March 2010.
25. R Raguram, O. Chum, M. Pollefeys, J. Matas, and J.M. Frahm. Usac: a universal framework for random sample consensus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):2022–2038, 2013.
26. Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1508–1515 Vol. 2, Oct 2005.
27. Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 430–443. Springer Berlin Heidelberg, 2006.
28. E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571, Nov 2011.
29. I. Skrypnyk and D.G. Lowe. Scene modelling, recognition and tracking with invariant image features. In *Mixed and Augmented Reality, 2004. ISMAR 2004. Third IEEE and ACM International Symposium on*, pages 110–119, Nov 2004.
30. S. Taylor, Edward Rosten, and Tom Drummond. Robust feature matching in 2.3 μs. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 15–22, June 2009.
31. Simon Taylor and Tom Drummond. Binary histogrammed intensity patches for efficient and robust matching. *Int. J. Comput. Vision*, 94(2):241–265, September 2011.
32. P. H. S. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:2000, 2000.
33. Etienne Vincent and Robert Laganiere. An empirical study of some feature matching strategies. *Proc. Conf. Vision Interface, Calgary, Canada*, pages 139–145, 2002.
34. Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Pose tracking from natural features on mobile phones. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 125–134. IEEE Computer Society, 2008.
35. Abraham Wald. *Sequential Analysis*. John Wiley and Sons, 1st edition, 1947.

Appendix A Homography Estimation using Gaussian Elimination

This appendix shows precisely the row operations we use to reduce to reduced-row-echelon form the matrix shown in Eq. 4. The Gaussian Elimination is done in two parts; In the first, identical row operations are applied to the top and bottom halves of the matrix, while in the second, row operations are applied to the whole matrix.

$$\begin{pmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0X_0 & -y_0X_0 & X_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1X_1 & X_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -y_2X_2 & X_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3X_3 & -y_3X_3 & X_3 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0Y_0 & -y_0Y_0 & Y_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1Y_1 & -y_1Y_1 & Y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2Y_2 & -y_2Y_2 & Y_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3Y_3 & -y_3Y_3 & Y_3 \end{pmatrix}$$

Starting from the above, we subtract rows 2 and 6 from the rows 0, 1, 3 and 4, 5, 7 respectively, thus eliminating almost all 1's in column 2 and 5. Since we choose not to scale the rows containing said 1's, they will remain unaffected throughout the remainder of the computation and therefore no storage needs to be reserved for them.

$$\sim \begin{pmatrix} x_0-x_2 & y_0-y_2 & 0 & 0 & 0 & 0 & x_2X_2-x_0X_0 & y_2X_2-y_0X_0 & X_0-X_2 \\ x_1-x_2 & y_1-y_2 & 0 & 0 & 0 & 0 & x_2X_2-x_1X_1 & y_2X_2-y_1X_1 & X_1-X_2 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -y_2X_2 & X_2 \\ x_3-x_2 & y_3-y_2 & 0 & 0 & 0 & 0 & x_2X_2-x_3X_3 & y_2X_2-y_3X_3 & X_3-X_2 \\ 0 & 0 & 0 & x_0-x_2 & y_0-y_2 & 0 & x_2Y_2-x_0Y_0 & y_2Y_2-y_0Y_0 & Y_0-Y_2 \\ 0 & 0 & 0 & x_1-x_2 & y_1-y_2 & 0 & x_2Y_2-x_1Y_1 & y_2Y_2-y_1Y_1 & Y_1-Y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2Y_2 & -y_2Y_2 & Y_2 \\ 0 & 0 & 0 & x_3-x_2 & y_3-y_2 & 0 & x_2Y_2-x_3Y_3 & y_2Y_2-y_3Y_3 & Y_3-Y_2 \end{pmatrix}$$

We note here that at this stage, of the 72 potential floating-point values in the matrix, only 32 (excluding the two remaining 1's) are distinct and non-zero. This neatly fits in half of a vector register file with 16 4-lane registers, a common configuration in most modern architectures.

For brevity, after this point only the row operations are given. They were designed to delay the use of reciprocals as long as possible. And the first part is duplicated on both top and bottom half.

First we eliminate column 0 of rows 1 and 3:

$$\mathbf{R}_1 = r_{0,x} * \mathbf{R}_1 - r_{1,x} * \mathbf{R}_0, \qquad \textit{idem on } \mathbf{R}_5$$
$$\mathbf{R}_3 = r_{0,x} * \mathbf{R}_3 - r_{3,x} * \mathbf{R}_0, \qquad \textit{idem on } \mathbf{R}_7$$

We eliminate column 1 of rows 0 and 3.

$$\mathbf{R}_0 = r_{1,y} * \mathbf{R}_0 - r_{0,y} * \mathbf{R}_1, \qquad \textit{idem on } \mathbf{R}_4$$
$$\mathbf{R}_3 = r_{1,y} * \mathbf{R}_3 - r_{3,y} * \mathbf{R}_1, \qquad \textit{idem on } \mathbf{R}_7$$

We eliminate columns 0 and 1 of row 2.

$$\mathbf{R}_0 = \frac{1}{r_{0,x}} * \mathbf{R}_0, \qquad \textit{idem on } \mathbf{R}_4$$
$$\mathbf{R}_1 = \frac{1}{r_{1,y}} * \mathbf{R}_1, \qquad \textit{idem on } \mathbf{R}_5$$
$$\mathbf{R}_2 = \mathbf{R}_2 - (r_{2,x} * \mathbf{R}_0 + r_{2,y} * \mathbf{R}_1), \qquad \textit{idem on } \mathbf{R}_6$$

Columns 0-5 of rows 3 and 7 are zero, and the matrix now resembles this:

$$\begin{pmatrix} 1 & & & & & 0 & a_{06} & a_{07} & a_{08} \\ & \ddots & & & & & a_{16} & a_{17} & a_{18} \\ & & 1 & 0 & & & a_{26} & a_{27} & a_{28} \\ & & 0 & 0 & & & a_{36} & a_{37} & a_{38} \\ & & 0 & 1 & & & a_{46} & a_{47} & a_{48} \\ & & & & \ddots & & a_{56} & a_{57} & a_{58} \\ & & & & & 1 & a_{66} & a_{67} & a_{68} \\ 0 & & & & & 0 & a_{76} & a_{77} & a_{78} \end{pmatrix}$$

, which concludes the first part. We now cease treating the matrix as two independent 4×9 halves and now consider the rightmost three columns as one 8×3 matrix for the second part. We use the barren rows 3 and 7 to eliminate columns 6 and 7, thus:

First, we normalize row 7.

$$\mathbf{R}_7 = \frac{1}{r_{76}} * \mathbf{R}_7$$

We eliminate column 6 of rows 0-6.

$$\mathbf{R}_0 = \mathbf{R}_0 - r_{06} * \mathbf{R}_7 \qquad \mathbf{R}_1 = \mathbf{R}_1 - r_{16} * \mathbf{R}_7$$
$$\mathbf{R}_2 = \mathbf{R}_2 - r_{26} * \mathbf{R}_7 \qquad \mathbf{R}_3 = \mathbf{R}_3 - r_{36} * \mathbf{R}_7$$
$$\mathbf{R}_4 = \mathbf{R}_4 - r_{46} * \mathbf{R}_7 \qquad \mathbf{R}_5 = \mathbf{R}_5 - r_{56} * \mathbf{R}_7$$
$$\mathbf{R}_6 = \mathbf{R}_6 - r_{66} * \mathbf{R}_7$$

We normalize row 3.

$$\mathbf{R}_3 = \tfrac{1}{r_{37}} * \mathbf{R}_3$$

We eliminate column 7 of rows 0-2 and 4-6.

$$\mathbf{R}_0 = \mathbf{R}_0 - r_{07} * \mathbf{R}_3 \qquad \mathbf{R}_1 = \mathbf{R}_1 - r_{17} * \mathbf{R}_3$$
$$\mathbf{R}_2 = \mathbf{R}_2 - r_{27} * \mathbf{R}_3 \qquad \mathbf{R}_4 = \mathbf{R}_4 - r_{47} * \mathbf{R}_3$$
$$\mathbf{R}_5 = \mathbf{R}_5 - r_{57} * \mathbf{R}_3 \qquad \mathbf{R}_6 = \mathbf{R}_6 - r_{67} * \mathbf{R}_3$$

This leaves the last column of the matrix, which contains the homography, normalized to $h_{22} = 1$:

$$\sim \begin{pmatrix} 1 & & & & & & & h_{00} \\ & \ddots & & & & & & h_{01} \\ & & 1 & 0 & & & & h_{02} \\ & & 0 & 0 & & & 1 & h_{21} \\ & & 0 & 1 & & & & h_{10} \\ & & & & \ddots & & & h_{11} \\ & & & & & 1 & & h_{12} \\ 0 & & & & & 0 & 1 & h_{20} \end{pmatrix} \rightarrow \begin{pmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{pmatrix}$$

This concludes the second part.