New Conditional Sampling Strategies for Speeded-Up RANSAC

Tom Botterill[1, 2]
tom.botterill@grcnz.com
Steven Mills[2]
steven.mills@grcnz.com
Richard Green[1]
richard.green@canterbury.ac.nz

[1] Department of Computer Science
University of Canterbury
Christchurch 8140, NZ

[2] Geospatial Research Centre (NZ) Ltd.
Private Bag 4800
Christchurch 8140, NZ

Abstract

RANSAC (Random Sample Consensus) is a popular algorithm in computer vision for fitting a model to data points contaminated with many gross outliers. Traditionally many small hypothesis sets are chosen randomly; these are used to generate models and the model consistent with most data points is selected. Instead we propose that each hypothesis set chosen is the one most likely to be correct, conditional on the knowledge of those that have failed to lead to a good model. We present two algorithms, BaySAC and SimSAC, to choose this most likely hypothesis set. We show them to outperform previous improved sampling methods on both real and synthetic data, sometimes halving the number of iterations required. In the case of real-time essential matrix estimation, BaySAC can reduce the failure rate by 78% with negligible additional cost.

1 Introduction

RANSAC (Random Sample Consensus) [5] is a popular algorithm in computer vision for fitting a model to a set of data points contaminated by outliers. A minimal number of points necessary to generate a model is randomly selected (a hypothesis set), a model is generated from these points, and the number of other points consistent with this model are counted. This is repeated for many randomly selected hypothesis sets until a model consistent with a sufficient number of data points is found.

This paper describes our improvement to the hypothesis set selection stage of RANSAC. Traditionally hypothesis sets are chosen randomly. Instead (as suggested by Chum and Matas [3]) at each time we select the hypothesis set most likely to lead to a good model. This deterministic selection process uses information gained by having tried and failed to find a good model from previous hypothesis sets, together with any prior information we might have on inlier probabilities. This innovation reduces the number of iterations needed to find a good model, hence reducing the computational cost. For real-time applications where the number of iterations is bounded, probability of failing to find a model is reduced.

Section 2 describes RANSAC, its applications in Computer Vision, and previous attempts to improve hypothesis set sampling, Section 3 describes our new algorithms for gen-

BMVC 2009 doi:10.5244/C.23.33

erating hypothesis sets, Section 4 presents results on simulated and real data, and Section 6 discusses these results.

2 Background

The original RANSAC algorithm [5] proceeds as follows:

1. **Hypothesise:** Randomly select n data points, where n is the minimum number needed to generate the model we are fitting. Generate model from these points.

2. **Test:** Count the number of data points consistent with this model.

3. Repeat until a model consistent with a large number of data points is found. If the model is correct these data points are inliers.

In this paper we select hypothesis sets of size n from a set of D data points. $\mathbb{I}$ is the set of all inliers that we aim to find, and H_t the hypothesis set at iteration t. $\Pi = \{\pi_1, \pi_2, ..., \pi_D\}$ denotes the prior inlier probabilities of each data point. This notation is summarised in Table 1.

D	Number of data points
n	Hypothesis set size
$\mathbb{I}$	Set of all inliers
t	Time (number of sets that have been tested)
H_i	Hypothesis set of n data points used at time i
$\mathbb{H} = \{H_1, H_2, ..., H_t\}$	History of failed hypothesis sets
$\Pi = \{\pi_1, \pi_2, ..., \pi_D\}$	Prior inlier probabilities

Table 1: Notation

There are many applications for RANSAC, these include finding the homography relating two views of a planar scene for image mosaicing [1], fitting 3D models to 3D point clouds from a laser scanner [11], and linear regression [10] (for example to find a ground plane). One of the best known applications is for simultaneously finding the fundamental matrix (or essential matrix in the case of calibrated cameras) relating correspondences between two images, and to identify and remove bad correspondences [14]. We are interested in essential matrix estimation as one of the stages of real-time Visual Odometry: this is where a robot's motion is computed by calculating the trajectory of an attached camera, and the essential matrix is used to find the relative position and orientation of many pairs of frames. Using RANSAC to estimate the essential matrix is an expensive part of our implementation, accounting for around 30% of the total computation.

Several related algorithms are commonly used for similar problems. MLESAC [15] (Maximum Likelihood Estimation Sample Consensus—the model with the highest estimated likelihood is selected, rather than the one with the highest inlier count), and LMS [10] (Least-Median of Squares—the model minimising the median error of the data is selected) are both variations on the model selection stage of RANSAC. Results in this paper apply only to the hypothesis generation stage, so are equally valid for MLESAC and LMS.

The computational cost of RANSAC is proportional to the number of iterations, which is the number of hypothesis sets that are chosen before a good enough model is found. The original RANSAC hypothesis set sampling strategy assumes that all data points (and hence hypothesis sets) are equally likely, therefore the order in which they are considered is irrelevant. In practice however, we can often estimate the relative likelihoods of data points being inliers prior to carrying out RANSAC. With stereo matching for example, correspondences are less likely to be correct if there are other potential matches with points that appear almost as similar. This information can be used to assess the relative likelihood of hypothesis sets. To minimise the time taken to find an all-inlier set leading to a good model, hypothesis sets should be evaluated in order of decreasing likelihood. Alternatively, the same ordering will also maximise the probability of finding an all-inlier set for real-time applications when the time available is bounded.

The Guided-MLESAC algorithm [13] uses a function of feature correlation strength (derived by fitting a heuristic model to test data) as a measure of prior inlier probability. When choosing random hypothesis sets each correspondence is chosen with selection probability proportional to its prior probability. Together with other innovations this is shown to reduce the time to find a model.

Chum and Matas [3] present an alternative sampling strategy known as PROSAC (Progressive Sample Consensus) based on the same idea. Data points are sorted by some measure of inlier likelihood, avoiding the need to estimate actual probabilities. First the n most likely data points are chosen, then the point $n+1$ and the $\binom{n}{n-1} = n$ possible sets of $n-1$ data points of the first n in some random order, and so-on. This allows hypotheses to be chosen in approximate order of their prior likelihood. In some experiments with variable prior inlier likelihoods this method hugely reduces the number of sets that must be tested.

These sampling strategies are heuristic and are based on data point inlier probabilities rather than the likelihood of hypothesis sets consisting entirely of inliers. In Section 3 we describe our new sampling algorithms which combine inlier probabilities with information from previous outlier-contaminated hypothesis sets to allow the (usually) deterministic choice of the most likely hypothesis set, and the best order in which hypothesis sets should be tried.

3 Proposed Conditional Sampling Methods

Previous sampling methods fail to take into account the information gained by testing hypothesis sets and finding them to be contaminated by outliers. In this paper we propose two methods to take this into account based on the following observation: given a hypothesis set H that leads to a model that is consistent with few data points, we assume that this was because it contains one or more outliers (the possibility that it contains a degenerate configuration of inliers will be dealt with later). Trying the same set again is a waste of time (although this is unlikely to happen with any reasonable number of likely data points). However hypothesis sets with one or more data points in common with H are also now less likely, as they are likely to include the same outlier(s) that contaminated H. Note that this is true for any set of prior probabilities, in particular the original RANSAC random sampling algorithm which assumes constant prior probabilities.

Ideally at each time we will choose the hypothesis set that is *most likely to contain no outliers* based on the prior probabilities Π and the history of contaminated samples $\mathbb{H}$. Unfortunately a closed-form solution for this posterior probability is algebraically intractable and probably doesn't exist, even for the simplest non-trivial case $n=2$ (as every intersecting pair of contaminated hypothesis sets introduces covariances between the inlier probabilities of all of their data points). Instead we present two methods of approximating this probability, both of which are shown to work well in practice. The aim of each of these methods is to choose H_t at time t such that $P(H_t|\mathbb{H},\Pi) \geq P(H|\mathbb{H},\Pi)\,\forall H$. This strategy minimises the number of hypotheses that must be tested before finding one consisting entirely of inliers.

3.1 Naïve Bayes method—BaySAC

Our first proposed method, BaySAC, assumes independence between inlier probabilities of data points in the same hypothesis set. First select the n data points with the highest inlier probabilities as our hypothesis set. This is the most likely under our independence assumption. After testing a hypothesis set H_t and finding that it was contaminated with outliers we update the inlier probabilities of each data point using Bayes' rule. Repeat with our new inlier probabilities. Probabilities are updated as follows:

$$\begin{aligned}
P_t(i \in \mathbb{I}) &= \text{Inlier probability for data point } i \text{ at time } t \\
&= \begin{cases} \frac{P_{t-1}(i\in\mathbb{I})P(H_t\not\subseteq\mathbb{I}|i\in\mathbb{I})}{P(H_t\not\subseteq\mathbb{I})} & i \in H_t \\ P_{t-1}(i \in \mathbb{I}) & i \notin H_t \end{cases} && (1) \\
\text{where } P(H_t \not\subseteq \mathbb{I}) &= 1 - P(H_t \subseteq \mathbb{I}) = 1 - \prod_{j\in H_t} P_{t-1}(j \in \mathbb{I}) && (2) \\
P(H_t \not\subseteq \mathbb{I}|i \in \mathbb{I}) &= 1 - P(H_t \subseteq \mathbb{I}|i \in \mathbb{I}) = 1 - \prod_{j\in H_t : j\neq i} P_{t-1}(j \in \mathbb{I}) && (3) \\
&= 1 - P(H_t \subseteq \mathbb{I})/P_{t-1}(i \in \mathbb{I}) && (4)
\end{aligned}$$

$$\Rightarrow P_t(i \in \mathbb{I}) = \begin{cases} \frac{P_{t-1}(i\in\mathbb{I}) - P(H_t\subseteq\mathbb{I})}{1-P(H_t\subseteq\mathbb{I})} & i \in H_t \\ P_{t-1}(i \in \mathbb{I}) & i \notin H_t \end{cases} \quad (5)$$

BaySAC uses these equations to update inlier probabilities:

```
Do
        1. Choose $n$ data points most likely to be inliers
        2. Use Equations 2 and 5 to update the inlier probability of these points
until a sufficiently large inlier set is found
```

BaySAC is fast and works well when there are few large intersections between sets. It works poorly in some cases when there are a small number of data points of which many are of very similar or equal probability, as following a Bayes update they're still equiprobable, and are therefore likely to be selected in the same hypothesis set again. This is not a problem for larger D as long as equiprobable data points are selected at random when they make the hypothesis set choice ambiguous.

3.2 Simulation method—SimSAC

An alternative way to compute data point inlier probabilities is by simulation. Choose random inlier/outlier statuses for each data point, check whether these statuses are compatible with having failed to observe an inlier set so far, and if so accumulate a histogram of inlier counts for each of the D data points. The n highest peaks in this histogram form the most likely inlier set. SimSAC proceeds as follows:

1. Do
    - Simulate random inlier/outlier statuses $x_1 ... x_N \in \{0,1\}^D$ based on prior inlier probabilities (so that $P(x_i = 1) = \pi_i$).
    - If $\forall H \in \mathbb{H} \exists h \in H$ s.t. $x_h = 0$ then increment a counter for each simulated inlier.

   until we have found T compatible simulated sets.

2. Choose the n data-points that were most frequently inliers as our next hypothesis set.

SimSAC works by sampling from the prior distribution of inlier/outlier status vectors, then updating this sample conditional on observing hypothesis sets that contain outliers (indicating that many possible status vectors were not correct). Finding the peaks in a histogram from accumulating these status vectors gives us the n data points that are individually most likely to be inliers, conditional on the failed hypothesis set history. Assuming independence between inlier probabilities for the most likely inliers at time t (a much weaker assumption than the Naïve Bayes assumptions in BaySAC), this algorithm will give us the actual most likely hypothesis set at each time, as $T \to \infty$. To avoid this assumption we would have to choose the inlier/outlier status vector that is most frequent conditional on being drawn from the prior distribution, and being compatible with the observed history. This is so complex (taking time and memory $\Omega(exp(D))$—a large fraction of possible status vectors must be simulated many times) that it is infeasable even for the smallest problems.

While SimSAC is much slower than other approaches, and still has high complexity, for practical applications it may be considerably faster than model generation. For practical applications choosing $T = 10$ samples provides a method to choose one of the more likely hypothesis sets. A value such as $T = 1000$ provides a 'ground truth' reference–it is unlikely any feasible algorithm can perform substantially better than this.

3.3 N–M Correspondences

When searching for correspondences between two frames it is often found that multiple (N) features in one image appear sufficiently similar to multiple (M) features in the other that they could be the same feature. The standard approach to this situation is to limit the image distance between features in the two images, and/or to reject all correspondences with multiple similar-looking potential matches [4, 9, 14]. The problem with this is that in visually poor or self-similar environments too few good correspondences may remain for scene geometry to be recovered accurately, if at all. Also limiting track lengths introduces difficulties in registering frames during rapid camera motion or following loop closure. Figure 1 shows two images where 2–2 and 3–3 correspondences are essential to obtain enough inliers over a wide enough area to recover camera motion. As a result, visual navigation systems are rarely demonstrated in such self-similar or visually poor environments.

A better solution to this problem is to introduce correspondences between all possible pairs of points. This leads to $N \times M$ correspondences of which at most $min(N,M)$ are correct. If $N \leq M$, and the probability of one of these points i in the first image matching any of the M points in the second is p, then the probability that correspondence (i,j) is an inlier is p/M. In general, the probability that (i,j) is an inlier is $p/max(N,M)$. Our new sampling

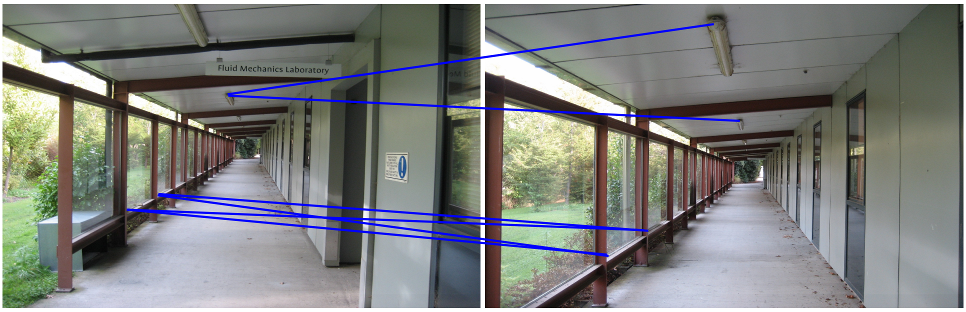

Figure 1: Frames from a video sequences used for essential matrix estimation with insufficient 1-1 correspondences for these to be used alone. Of the 6 potential correspondences shown at most 3 are correct.

algorithms, with a simple adaption to avoid choosing incompatible correspondences (if (i,j) is an inlier, (i,k) and (l,j) are not $\forall\, k,l$) can make use of these probabilities. Note that unlike other sampling strategies (RANSAC and Guided-MLESAC) it does not matter that many correspondences with low probabilities are introduced, as they will only be selected as part of hypothesis sets once sufficient evidence has been observed to indicate they are more likely to be correct than correspondences with higher prior probabilities.

If $(i,j) \in H_t$ and H_t is contaminated by outliers, the posterior probabilities of $\{(i,k),(l,j)\ \forall\, k,l\}$ are slightly increased (the probability of other correspondences that are incompatible with $\{(i,k),(l,j)\}$ should also be slightly reduced but this amount is negligibly small). We adapt Equation 5 as follows to take this into account:

$$\begin{aligned} P_t((i,k) \in \mathbb{I} | (i,j) \in H_t) &= P((i,k) \in \mathbb{I} | (i,j) \in \mathbb{I}) P((i,j) \in \mathbb{I} | (i,j) \in H_t) && (6)\\ &+ P((i,k) \in \mathbb{I} | (i,j) \notin \mathbb{I}) P((i,j) \notin \mathbb{I} | (i,j) \in H_t) \\ &= \frac{P_{t-1}((i,k) \in \mathbb{I})}{(1 - P_{t-1}((i,j) \in \mathbb{I}))} (1 - P_t((i,j) \in \mathbb{I})) && (7)\\ \text{because } (i,j) \in \mathbb{I} &\Rightarrow (i,k) \notin \mathbb{I} \ \Rightarrow\ P((i,k) \in \mathbb{I} | (i,j) \in \mathbb{I}) = 0 && (8) \end{aligned}$$

4 Results

We have validated our new sampling algorithms using both simulated data and real data (for the case of essential matrix estimation). When considering *N*–*M* correspondences all algorithms are modified to avoid selecting impossible combinations: correspondences are not selected if a previously selected correspondence shares an endpoint.

5 Results on Simulated Data

To test our new sampling algorithms we simulate the inlier/outlier statuses of D data points with some prior probabilities Π. We then sample repeatedly using one of our sampling strategies until a set consisting entirely of inliers is found. Table 2 shows the mean number of iter-

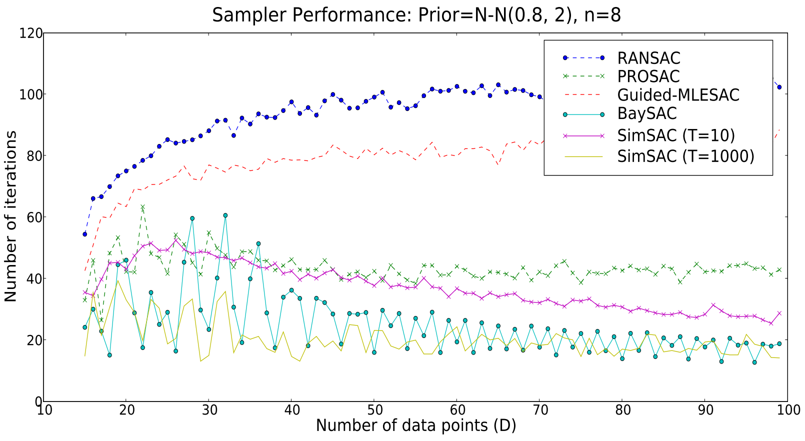

Figure 2: BaySAC and SimSAC with $T = 1000$ reduce the number of iterations needed to find an all-inlier hypothesis set.

ations (and a 99% confidence bound on this mean) needed to find a hypothesis set containing only inliers, for the values D (number of data points) $= 50$ and n (hypothesis set size) $= 5$.

Several prior probability distributions are used:

Constant(p) Original RANSAC—prior probabilities all equal a constant p ($p = 0.5$ is shown in Table 2, similar results are observed for $p = 0.25$ and $p = 0.75$).

Unif(a,b) Prior probabilities are uniformly distributed in the range (a,b). Uniform(0.25, 0.75) are shown in Table 2—the second case corresponds to feature matches being uncertain despite a very strong feature matcher response, together with a lower bound on allowed match quality. Very similar results are observed for probabilities from Uniform(0, 1).

N-N(p,N) Prior probability of one data point having a match is constant ($p = 0.6$ for these experiments), but sets of K points each have K match candidates ($P(\text{match}) = p/K$, $K \sim \text{Unif}[1,\ldots,N]$).

Additionally, as prior probabilities are hard to estimate in advance [3], simulations are carried out where true prior probabilities are uniformly distributed about the estimated values from 0.25 below to 0.25 above (subject to staying within $(0,1)$). There is also a possibility that degenerate configurations incorrectly assumed to contain outliers will cause a sampling strategy to fail. To simulate this inlier sets are rejected with probability 0.25. The number of iterations needed under these assumptions are shown in column 'Uncertain Π' of Table 2.

Table 2 only shows results for the values $D = 50$ and $n = 5$. Results are comparable for values of D from 15 to 1000 and n from 2 to 10; Figure 2 shows the consistently strong performance of BaySAC and SimSAC for $D = 15..100$ and $n = 8$, with prior probabilities from N-N$(0.8, 2)$. Note the periodicity in the BaySAC performance–when n divides D (or they have a high common factor) sets of n equiprobable data points remain equiprobable (and hence are selected again at the same time) following a Bayes update. Run times are given–these are only directly comparable for a particular number of iterations (before either succeeding or terminating) as different sampling strategies have different complexity, as discussed in Section 5.2. In these simulations RANSAC is terminated after 250 iterations if no inlier set is found (termination is essential for real-time applications), and the success rate is recorded.

The results presented include the mean number of samples drawn when a hypothesis set is found, a 99% confidence bound on this mean and the proportion of samples where no inlier set was found. SimSAC with $T = 1000$ outperforms all other methods, but is

Sampling algorithm	Prior probabilities	Mean iters. (99% bound)	Success (%)	Time ($\times 10^{-6}$s)	Mean iters. Uncertain Π
RANSAC	Constant(0.5)	43.28 ± 0.16	96	0.34	51.77 ± 0.18
PROSAC	Constant(0.5)	66.85 ± 0.29	53.2	0.447	70.1 ± 0.3
Guided-MLESAC	Constant(0.5)	43.14 ± 0.16	95.9	0.661	51.57 ± 0.18
BaySAC	Constant(0.5)	41.74 ± 0.16	96.2	1.04	50.41 ± 0.18
SimSAC ($T = 10$)	Constant(0.5)	42.76 ± 0.16	96	7.26	50.82 ± 0.18
SimSAC ($T = 1000$)	Constant(0.5)	$\mathbf{41.71 \pm 1.1}$	96.2	579	$\mathbf{47.93 \pm 3.9}$
RANSAC	Unif(0.25, 0.75)	43.34 ± 0.16	96	0.341	51.8 ± 0.18
PROSAC	Unif(0.25, 0.75)	30.96 ± 0.16	90.9	0.625	34.21 ± 0.17
Guided-MLESAC	Unif(0.25, 0.75)	32.39 ± 0.13	98.2	0.687	40.03 ± 0.15
BaySAC	Unif(0.25, 0.75)	18.99 ± 0.12	96.4	1.42	23.37 ± 0.14
SimSAC ($T = 10$)	Unif(0.25, 0.75)	21.51 ± 0.12	98.2	6.47	27.86 ± 0.14
SimSAC ($T = 1000$)	Unif(0.25, 0.75)	$\mathbf{16.47 \pm 0.68}$	99	479	$\mathbf{19.9 \pm 2.3}$
RANSAC	N-N(0.6, 3)	109.1 ± 0.43	29.5	0.386	112.7 ± 0.48
PROSAC	N-N(0.6, 3)	48.61 ± 0.29	40.6	0.687	51.86 ± 0.31
Guided-MLESAC	N-N(0.6, 3)	96.32 ± 0.35	44.5	1.07	101.2 ± 0.38
BaySAC	N-N(0.6, 3)	58.8 ± 0.38	40.9	3.59	57.61 ± 0.38
SimSAC ($T = 10$)	N-N(0.6, 3)	61.7 ± 0.29	56.3	10.1	65.82 ± 0.31
SimSAC ($T = 1000$)	N-N(0.6, 3)	$\mathbf{19.65 \pm 1.4}$	31.9	509	$\mathbf{28.79 \pm 6.1}$

Table 2: Results on simulated data with $D = 50$ and $n = 5$.

far more costly. BaySAC performs almost as well, and considerably better that Guided-MLESAC sampling, RANSAC sampling or PROSAC sampling in most cases. Inaccurate prior probability estimates have surprisingly little effect on results. Improvements from our new algorithms are marginal in the original case with uniform prior probabilities (although are more significant when D is small, e.g. $D = 20$), but large improvements are gained when prior probabilities can be estimated.

The case when BaySAC does not performs quite as well is with N-N correspondences. This is the one case where there is significant room for improvement—no other algorithm comes close to matching the 20 iterations needed by the simulated sampler with $T = 1000$.

Note that PROSAC sampling often performs poorly, particularly with constant prior probabilities where it performs considerably worse than random sampling. This is because it repeatedly samples from the same small set of data points, and when these are contaminated with outliers many iterations are wasted trying small variations on contaminated hypothesis sets. PROSAC does not take into account the information provided by previous failures.

5.1 Results on Real Data

In order to evaluate our new RANSAC variations on real data we use them to find the essential matrices from the N–M correspondences ($N, M \leq 3$) between consecutive pairs of frames from a 20 frame indoor and a 204 frame outdoor video sequence (Figure 3). The prior probability model described earlier is used, with $p = 0.5$. Each video was processed

Figure 3: Frames from the indoor and outdoor sequences used for essential matrix estimation

Sampling algorithm	Outdoors		Indoors	
	Success rate (s.d.)	Inliers	Success rate (s.d.)	Inliers
RANSAC	144.0/203 (1.0)	151	16.7/19 (0.5)	131
PROSAC	179.3/203 (5.0)	130	17.0/19 (0.0)	130
Guided-MLESAC	186.3/203 (3.4)	129	16.3/19 (2.6)	124
BaySAC	189.0/203 (0.0)	130	18.3/19 (1.1)	129
SimSAC (10)	185.0/203 (2.0)	130	17.7/19 (1.1)	122
SimSAC (1000)	**191.3/203** (1.9)	131	**18.7/19** (0.5)	131

Table 3: Essential matrix estimation—N–M correspondences from pairs of consecutive frames from indoor and outdoor video sequences

three times, and the mean number of times a good essential matrix was found (one with at least 15 inliers, and within a threshhold of the known camera motion) is given. The number of iterations is limited to 250, and in practice we rarely terminate much earlier than this (as inlier rates are not high enough). As a result, run times are almost identical (0.04s per frame pair). Table 3 shows that BaySAC finds the essential matrix more often than other methods, other than SimSAC with $T = 1000$; this takes 0.3s per frame pair so is too slow for real-time use. Note that ignoring prior probabilities entirely (original RANSAC) results in poor performance.

We find around 400 potential correspondences per frame with $N, M \leq 3$. Using only 1-1 correspondences results in around 50-150 correspondences, and regularly less than 25 inliers are found from these (an average of 49 are found in the indoor sequence). We have found this level too low to recover an accurate camera motion estimate from these videos.

5.2 Run-times and complexity

Execution times in this paper refer to C/C++ code compiled with gcc and running on one core of a 3Ghz Intel Core 2 Duo processor.

Drawing a hypothesis set of size n using BaySAC has $O(n\,log(D))$ complexity (maintain a sorted list of D probabilities, update around n at each iteration, and select the top n). There is an initial setup cost of $O(D\,log(D))$.

Simulating inlier/outlier statuses for D data points takes time of $O(D)$. Testing compatibility with a history of size t takes time of $O(tn)$. If prior probabilities truly reflect the inlier/outlier probabilities then few status sets need to be generated to find one compatible with having not found an all-inlier set so far. This would give SimSAC a total complexity of $O(Dtn)$. However overly optimistic inlier probabilities lead to large numbers of status sets being rejected, greatly increasing this complexity.

The RANSAC application we are particularly interested in is real-time essential matrix estimation for visual odometry. Stewénius *et al.* [12] developed an algorithm to compute the 2-10 essential matrices compatible with five points; our C++ implementation (using the Eigen matrix library [6]) takes an average of 1.4×10^{-4}s to estimate an average of 5 matrices compatible with a hypothesis set. Validating a model for one data point takes 2.5×10^{-8}s, and typically few data points need to be compared in practice [2, 8], making this a small component of the total cost. For this application BaySAC's increased sampling costs are insignificant compared to the cost of model generation and testing (taking about 1.5×10^{-6}s per sample with $D = 50$, $n = 5$—about 1% as long).

For applications where model generation and verification is more expensive (for example trifocal tensor estimation [7]) minimising the number of iterations required by using SimSAC may be worthwhile, however the improvement in the case of essential matrix estimation is not great enough to justify the increased cost of generating samples.

6 Conclusions

BaySAC is an effective way to speed up RANSAC sampling when prior probabilities can be estimated. In some cases the number of iterations required can be halved compared with simple RANSAC sampling, and it significantly outperforms other sampling algorithms that are based on estimated prior probabilities. Performance is good even when prior probability estimates are only approximate.

In the case of essential matrix estimation BaySAC and SimSAC can make effective use of $N-M$ correspondences between sets of multiple points. This allows improved performance in visually poor and self-similar environments, decreasing the probability of failing to find an essential matrix connecting two cameras by 78% in one example. BaySAC and SimSAC can also reduce the number of iterations needed when data points are assumed equally likely to be inliers, as in the original RANSAC algorithm, although the speed-up is marginal unless model generation is particularly expensive.

References

[1] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *Int J Comput Vision*, 74(1):59–73, 2007.

[2] D.P. Capel. An effective bail-out test for RANSAC consensus scoring. In *Proc British Machine Vision Conference*, pages 1–10, 2005.

[3] O. Chum and J. Matas. Matching with PROSAC - progressive sample consensus. In *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*, pages 220–226, Los Alamitos, USA, 2005. ISBN 0-7695-2372-2.

[4] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc IEEE Int Conf Comput Vis*, October 2003.

[5] 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, 1981. ISSN 0001-0782.

[6] Gaël Guennebaud and Benoît Jacob. Eigen 2 matrix library. URL `http://eigen.tuxfamily.org/`.

[7] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, second edition, 2003.

[8] Jiří Matas and Ondřej Chum. Randomized RANSAC with sequential probability ratio test. In *Proc IEEE Int Conf Comput Vis*, pages 1727–1732, New York, USA, October 2005.

[9] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1):3–20, 2006.

[10] P. J. Rousseeuw and A. M. Leroy. *Robust regression and outlier detection*. John Wiley & Sons, Inc., New York, NY, USA, 1987. ISBN 0-471-85233-3.

[11] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226, June 2007.

[12] H. Stewénius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60:284–294, June 2006.

[13] Ben J. Tordoff and David W. Murray. Guided-MLESAC: Faster image transform estimation by using matching priors. *IEEE Trans Pattern Anal Mach Intell*, 27(10):1523–1535, 2005. ISSN 0162-8828.

[14] P. H. S. Torr and D. W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *Int J Comput Vision*, 24(3):271–300, 1997.

[15] P. H. S. Torr and A. Zisserman. MLESAC: a new robust estimator with application to estimating image geometry. *Comput. Vis. Image Underst.*, 78(1):138–156, 2000. ISSN 1077-3142.