GMS: Grid-based Motion Statistics for Fast, Ultra-robust Feature Correspondence

JiaWang Bian*
Singapore University of Technology and Design

Wen-Yan Lin*
Advanced Digital Sciences Center

Yasuyuki Matsushita
Osaka University

Sai-Kit Yeung
Singapore University of Technology and Design

Tan-Dat Nguyen
Institute of Infocomm Research

Ming-Ming Cheng
Nankai University

* denotes joint first author

Abstract

Incorporating smoothness constraints into feature matching is known to enable ultra-robust matching. However, such formulations are both complex and slow, making them unsuitable for video applications. This paper proposes GMS (Grid-based Motion Statistics), a simple means of encapsulating motion smoothness as the statistical likelihood of a certain number of matches in a region. GMS enables translation of high match numbers into high match quality. This provides a real-time, ultra-robust correspondence system. Evaluation on videos, with low textures, blurs and wide-baselines show GMS consistently out-performs other real-time matchers and can achieve parity with more sophisticated, much slower techniques.

1. Introduction

Feature matching is the basic input of many computer vision algorithms. Thus its speed, accuracy and robustness are of vital importance. Currently, there is a wide performance gap between slow (but robust) feature matchers and the much faster (but often unstable) real-time solutions.

The central problem lies in the coherence constraints (neighboring pixels share similar motion) utilized in the more powerful feature correspondence techniques. Coherence is a powerful constraint but sparse features lack well defined neighbors. This causes coherence based feature correspondence [16, 42] to be both expensive to compute and complex to implement. This paper proposes *GMS* (Grid-based Motion Statistics), a means of encapsulating motion smoothness as a statistical likelihood of having a certain number of feature matches between a region pair. We show *GMS* can rapidly and reliably differentiate true and false matches, enabling high quality correspondence in Fig. 1.

Our paper draws inspiration from *BF* [16]. *BF* emphasized that the apparent lack of feature matches is not due to too few correct matches but in the difficulty of reliably separating the true and false matches. *BF* demonstrated the feasibility of achieving such a separation by using a coherence measure computed with a complex minimization. In practice, *BF* works very well (albeit slowly). However, it is primarily motivated by observations and intuitions. The lack of theoretical clarity makes improvements difficult as researchers must rely on empirical tests on image data influenced by many fluctuating variables.

We suggest the complex smoothness constraints used in *BF* and other similar techniques [30, 19, 42] can be reduced to a simple statement: Motion smoothness induces correspondence clusters that are highly unlikely to occur at random. Thus true and false matches can be differentiated

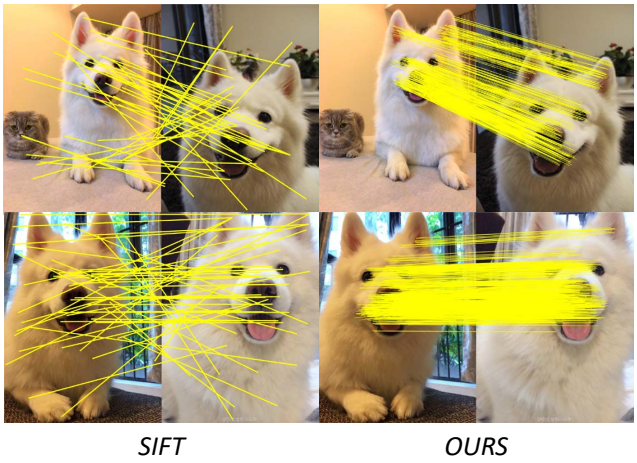


Figure 1. The highly respected SIFT [22] descriptor has difficulty on this scene because the dog's fur movement adds noise to local descriptors. Although we use weaker *ORB* descriptors, our *GMS* solution can leverage feature numbers to improve quality while maintaining real-time performance.

by simply counting the number of matches in their neighborhood. From the law-of-large-numbers, the partionablity of true and false scales to infinity with match numbers. The mathematical analysis is straight forward but the results are potentially paradigm shifting.

Previous feature matching papers [22, 35, 2, 47] assume that match quality primarily scales with improvements in feature invariance/distinctiveness. *GMS* reveals a new direction for improvement; Raw feature numbers can also impact quality. As finding more features is simpler than designing new descriptors, *GMS* potentially offers simple solutions to previously intractable matching problems, as illustrated in Fig. 1.

In summary, our contributions are:

- Converting the motion smoothness constraints into statistical measures for rejecting false matches. We show this constraint enables matching on previously intractable scenes;
- Develop an efficient Grid-based score estimator that can be incorporated into a real-time feature matcher;
- Demonstrate our *GMS* system is significantly better than traditional *SIFT* [22], *SURF* [2] and recent, CNN trained *LIFT* features [47] with standard ratio-test.

1.1. Related works

The foundational works on feature matching sought to increase the distinctiveness/invariance of feature descriptors and improve localization. Examples includes classic works like *SIFT* [22], *ORB* [35], *SURF* [2], *A-SIFT* [26], *Harris Corners* [9] and affine covariant region detectors [25]. Many of these works have GPU-acceleration [45, 40, 7] permitting real (or near-real) time performance. In addition, there are *FLANN* works for accelerating feature matching [14, 27, 28]. Such research is still on-going, the most recent example being *CNN* trained *LIFT* descriptors [47]. Together, these works form a core set of techniques that we build on.

The problem with sole reliance on descriptors is the difficulty in differentiating true and false matches. This results in the elimination of a large fraction of true matches to limit false matches [16]. *RANSAC* [10, 41, 5, 32, 6, 36, 15] can leverage geometric information to alleviate this problem. However, *RANSAC* itself requires most false matches to be pre-eliminated and cannot accommodate the sheer number of false matches in the set of all nearest-neighbor matches [17].

Recently, a number of techniques [30, 16, 17, 19, 42, 24] have focused on separating true and false matches using match distribution constraints. However, their formulations result in complex smoothness constraints, which are difficult to understand and expensive to minimize. Our approach is inspired by these works but uses a much simpler and more easily understood statistical matching constraint. This enables matching that is both robust and efficient.

More generally, our work is related to optical flow [13, 23, 4, 43, 33, 21, 1, 46], point based coherence techniques [48, 18, 29], patch-match based matchers [12] which directly use smoothness to help match estimation. These techniques can be very powerful. However, they are also much more complicated and expensive. Finally, we acknowledge the inspiration drawn from boosted learners like *AdaBoost* [11] which integrates multiple weak-learners into a powerful learner. *GMS* shares this design philosophy by using the smoothness constraint to integrate information from multiple matches to make high quality decisions.

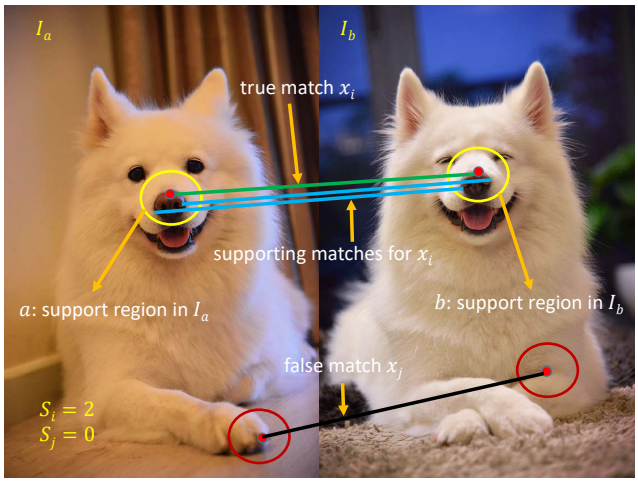


Figure 2. The neighborhood of match x_i is defined as $\{a, b\}$, a pair of small support regions around the respective features. We predict true match neighborhoods will have many more supporting matches than false match neighborhoods.

2. Our approach

Given a pair of images taken from different views of the same 3D scene, a feature correspondence implies a pixel (feature point) in one image is identified as the same point in the other image. If the motion is smooth, neighboring pixels and features move together. This allows us to make the following assumption:

Assumption 1. *Motion smoothness causes a (small) neighborhood around a true match to view the same 3D location. Likewise, the neighborhood around a false match views geometrically different 3D locations.*

Here neighborhood is defined as a pair of regions $\{a, b\}$ surrounding the respective image features shown in Fig. 2.

Assumption 1 implies that true match neighborhoods, view the same $3D$ region and thus share many similar features across both images. This results in the neighbor-

hood having many supporting matches. In contrast, false match neighborhoods view different $3D$ regions and have far fewer similar features. This reduces matching support. We encapsulate this intuition into a statistical framework called *GMS*, that reliably distinguishes between true and false matches. Sec. 3 introduces the grid algorithm for fast neighborhood score computation, while Sec. 4 presents results and comparisons.

2.1. Notation

Image pairs $\{I_a, I_b\}$ have $\{N, M\}$ features respectively. $\mathcal{X} = \{x_1, x_2, \ldots, x_i, \ldots, x_N\}$ is the set of all nearest-neighbor feature matches from I_a to I_b. $\mathcal{X}$ has cardinality $|\mathcal{X}| = N$. Our goal is to divide $\mathcal{X}$ into sets of true and false matches by analyzing the local support of each match.

Notation for the match neighborhood shown in Fig. 2 is as follows: The respective regions in $\{I_a, I_b\}$ are denoted as $\{a, b\}$, each with $\{n, m\}$ additional (excluding the original match pair) features respectively. $\mathcal{X}_i \subseteq \mathcal{X}$ is the subset of matches between regions $\{a, b\}$ of match x_i. S_i is a measure of neighborhood support:

$$S_i = |\mathcal{X}_i| - 1, \tag{1}$$

where -1 removes the original match from the sum.

2.2. Basic statistical constraints

As the regions are small, we restrict our considerations to idealized true and false region pairs, ignoring partially similar locations. Let f_a be one of the n supporting features in regions a. Given f_a has probability t of matching correctly, our goal is to derive the arrival rate of matches to regions $\{a, b\}$ for the cases when $\{a, b\}$ view the same/ different locations. Tab. 1 summarizes commonly used notations and events while Fig. 3 illustrates f_a's event space.

To make the problem tractable, we make the assumption

Assumption 2. *Given f_a matches wrongly, its nearest-neighbor match can lie in any of the M possible locations.*[1]

Assumption 2 arises because in general, there is no a-prior reason for a feature's wrong nearest neighbor to favor any image region. Assumption 2 gives

$$p(f_a^b | f_a^f) = \beta m / M \tag{2}$$

where m is the number of features in region b and β is a factor added to accommodate violations of assumption 2 caused by repeated structures like a row of windows.

Let $p_t = p(f_a^b | T^{ab})$ be the probability that, given $\{a, b\}$ view the same location, feature f_a's nearest neighbor is in region b. Thus,

$$\begin{aligned} p_t = p(f_a^b | T^{ab}) &= p(f_a^t | T^{ab}) + p(f_a^f, f_a^b | T^{ab}) \\ &= p(f_a^t | T^{ab}) + p(f_a^f | T^{ab}) p(f_a^b | f_a^f, T^{ab}) \\ &= p(f_a^t) + p(f_a^f) p(f_a^b | f_a^f) \\ &= t + (1 - t)\beta m / M \end{aligned} \tag{3}$$

Explanation: Fig. 3(i) shows event f_a^b occurs only if, f_a matches correctly, or it matches wrongly but coincidentally lands in region b. This gives equation (3)'s first line. The second line arises from Baye's rule. As features are pre-matched, $p(f_a^t)$, $p(f_a^f)$ are independent of T^{ab}. Due to assumption 2, $p(f_a^b | f_a^f)$ is also independent of T^{ab}. Dropping the conditioning T^{ab} and substituting values from Tab. 1 and equation (2) gives the final expression.

[1] Can also be $M - 1$ depending matching circumstances

(i) Notations	
x_i	the ith feature match
f_a	one of the n supporting features in region a
S_i	neighborhood score of match x_i
$\{p_t, p_f\}$	$\{p(f_a^b \| T^{ab}), p(f_a^b \| F^{ab})\}$ respectively
(ii) Events	
T^{ab}	regions $\{a, b\}$ view the same location
F^{ab}	regions $\{a, b\}$ view the different locations
f_a^t	f_a matches correctly, $p(f_a^t) = t$
f_a^f	f_a matches wrongly, $p(f_a^f) = 1 - t$
f_a^b	f_a's nearest-neighbor is a feature in region b
$\overline{f_a^b}$	f_a's nearest-neighbor is not in region b

Table 1. Table of notations and events.

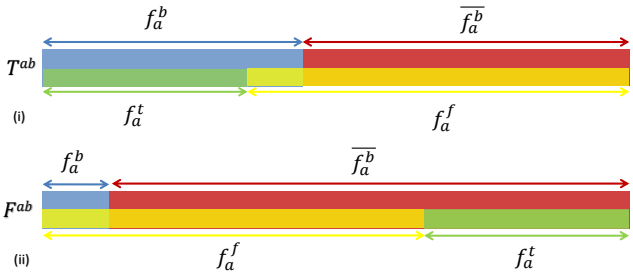


Figure 3. Event space for feature f_a. f_a's nearest-neighbor match either lands in region b (event f_a^b) or it does not (event $\overline{f_a^b}$). The match is either true (event f_a^t) or false (event f_a^f). (i) Given T^{ab}, f_a^b is the union of events f_a^t and some f_a^f events. (ii) Given F^{ab}, f_a^b is necessarily a subset of events f_a^f.

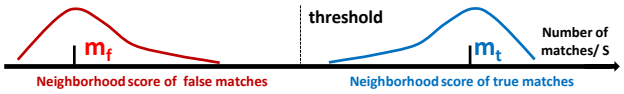


Figure 4. Neighborhood scores of true and false matches follow different distributions. If distribution means have sufficiently large separation relative to their standard deviations, true and false matches are potentially separable with score based thresholding.

Let $p_f = p(f_a^b|F^{ab})$. Similar to equation (3),

$$\begin{aligned} p_f = p(f_a^b|F^{ab}) =& p(f_a^f, f_a^b|F^{ab}) \\ =& p(f_a^f|F^{ab})p(f_a^b|f_a^f, F^{ab}) \\ =& p(f_a^f)p(f_a^b|f_a^f) \\ =& \beta(1-t)(m/M) \end{aligned} \tag{4}$$

Explanation: From Fig. 3(ii) we know event f_a^b is a subset of f_a^f. Thus, $p(f_a^b|F^{ab}) = p(f_a^f, f_a^b|F^{ab})$, giving the first line of equation (4). Similar to equation (3), the probabilities can be expanded by Bayes rule into sub-probabilities that are independent of F^{ab}. Substituting values from Tab. 1 and equation (2) gives the final expression.

As the matching of each feature is independent, using assumption 1 and equations (3),(4), we can approximate the distribution of S_i, the number of matches in a neighborhood of match x_i, with a pair of binomial distribution:

$$S_i \sim \begin{cases} B(n, p_t), & \text{if } x_i \text{ is true} \\ B(n, p_f), & \text{if } x_i \text{ is false} \end{cases} \tag{5}$$

While equations (5) seems complex, the important point is that true and false matches have neighborhood scores, S, that follow very different distributions. This means the overall pdf of S is potentially bimodal, making S score a useful indicator for differentiating true and false matches. This is illustrated in Fig. 4.

2.3. Multi-neighborhood generalization

Motion is often smooth over a large area. However, assumption 1 requires sufficiently small neighborhoods. In over-large neighborhoods, true match neighborhoods will include some false matching regions and vice-versa. This reduces the separability of true-false score distributions. Thus, generalizing assumption 1, we have:

Assumption 3. *If motion is smooth over a region, a true match allows prediction of multiple small region pairs that view the same 3D location. Using the same prediction function on a false match will result in geometrically different 3D locations.*

This gives a more generalized score

$$S_i = \sum_{k=1}^{K} |\mathcal{X}_{a^k b^k}| - 1 \tag{6}$$

where K is the number of disjoint regions which match i predicts move together. $\{a^k, b^k\}$ are predicted region pairs and $\mathcal{X}_{a^k b^k} \subseteq \mathcal{X}$ is the subset of matches that land on region pairs $\{a^k, b^k\}$.

Assuming each region pair has $\{n, m\}$ features, similar to equations (5), assumption 3 indicates a Binomial distributions of S_i

$$S_i \sim \begin{cases} B(Kn, p_t), & \text{if } x_i \text{ is true} \\ B(Kn, p_f), & \text{if } x_i \text{ is false} \end{cases} \tag{7}$$

The respective mean and standard deviation of S_i's distribution are

$$\begin{aligned} &\{m_t = Knp_t, s_t = \sqrt{Knt(1-p_t)}\} && \text{if } x_i \text{ is true} \\ &\{m_f = Knp_f, s_f = \sqrt{Knp_f(1-p_f)}\} && \text{if } x_i \text{ is false} \end{aligned} \tag{8}$$

Typically, when dealing with statistical events, we consider an event at x standard deviations from the mean as highly unlikely to happen. This is illustrated in Fig. 4 and can be quantified with a partionability score:

$$P = \frac{m_t - m_f}{s_t + s_f} = \frac{Knp_t - Knp_f}{\sqrt{Knp_t(1-p_t)} + \sqrt{Knp_f(1-p_f)}}. \tag{9}$$

Our goal is to design algorithms which maximize P.

2.4. Analysis

These derivations are simple. However they bring clarity to our intuition and permit derivation of some useful results.

Quantity-Quality equivalence: The key result of these derivations is:

$$P \propto \sqrt{Kn}. \tag{10}$$

This means, provided $m_t > m_f$, the partionability of true and false matches scales to infinity with n, the number of features. i.e. Even for difficult scenes with few true matches, if true matching locations have more matches than random locations, we can obtain as many matches as we desire, with perfect precision and recall, provided the number of features is sufficiently large. This forms a direct link from the assumptions 1, 2, 3 to the law-of-large-numbers. The results are interesting as most previous works assume the key to better correspondence is increasing feature distinctiveness/ invariance. Instead, equation (10) shows raw feature numbers can contribute to match quality. This makes it possible to solve challenging matching problems by increasing feature numbers! Fig. 5 shows an example.

Motion Prediction: Often, it is impractical to simply increase n. Equation (10) suggests an alternative is to increase K by predicting more jointly moving of image regions. This is the approach we utilize to build our real-time matching system.

Practical applicability: The *GMS* constraint is powerful given a sufficiently large n. The question is whether n is large enough in practice? Given $10,000$ evenly distributed

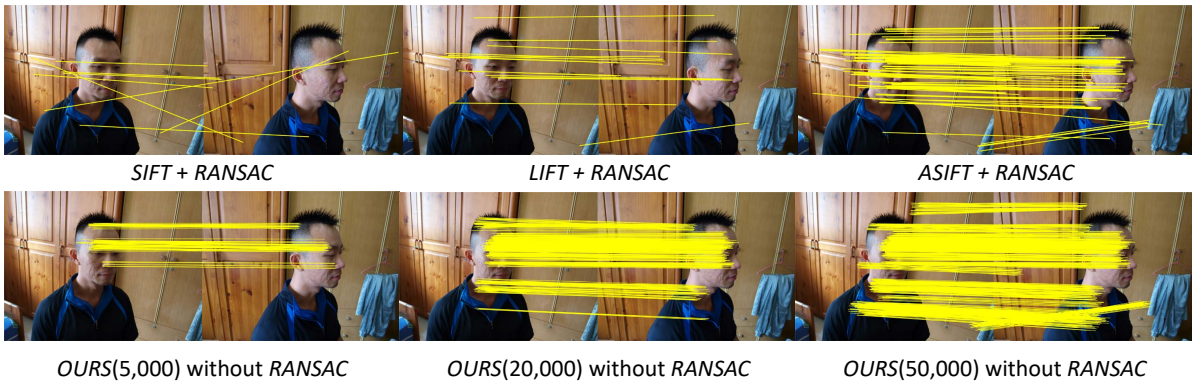


Figure 5. Top: Traditional feature matching focuses on increasing descriptor invariance. The most recent iteration is *LIFT* [47]. Thus, *A-SIFT* [26], which explicitly models large deformations, has the best wide-baseline performance. Bottom: *GMS* achieves the same effect with standard *ORB* [35] by simply increasing the number of features (indicated in brackets) . This makes *ORB* a viable wide-baseline feature!

features, some typical values would be $\{m = n = 25, \beta = 1, t = 0.5, K = 1\}$. Thus, the mean and standard deviations of S_i in the basic constraint of equation (8) are:

$$\{m_t = 12.5, s_t = 2.5\}, \quad \{m_f = 0.03, s_f = 0.176\}, \quad P = 4.7 \tag{11}$$

This is a wide separation. Note that $K = 1$. This demonstrates the basic *GMS* constraint is quite powerful at typical match numbers. It may also help explains why so many techniques with similar formulations achieve good results [30, 16, 17, 19, 42, 24].

Relationship to descriptors: The relationship of partionability with t, percentage match correctness is given by

$$\lim_{t \to 1} m_t \to Kn, \quad \lim_{t \to 1} m_f \to 0, \quad \lim_{t \to 1} P \to \infty. \tag{12}$$

As nearest neighbor matching tends to perfection, m_t tends to its maximum value, m_f to its minimum and partionablity to ∞. Thus, if a fixed threshold is set, *GMS*'s results will be better on easier scenes with high t. This is of practical importance as t is unknown and scene pair dependent. It also increases *GMS*'s generality by allowing it to scale with improvements in feature descriptor design.

3. Grid framework for fast scoring

This section transits the previous analysis into an effective real-time matching algorithm. Sec. 3.1 introduces the grid framework, addressing issues like: a) Efficient score computation through grid-cells; b) Which neighborhoods (grid-cells) to group together; c) How many grid-cells to use; d) How to compute an effectively threshold S. An implementational overview is given in algorithm 1 and fine details discussed in Sec. 3.2.

3.1. Griding the problem

a) Efficient score evaluation. The cost of scoring each feature match's neighborhood is $O(N)$, where N is the number of image features. This conflicts with our desire to use as many features as possible. We address it with a grid approximation that divides an image into $G = 20 \times 20$ non-overlapping cells. Scores of potential cell-pairs are computed only once. All matches between cell-pairs deemed true are accepted. This make score computation independent of feature numbers i.e. $O(1)$. In practice, many features lie on grid edges. To accommodate this, we repeat the algorithm 3 more times with gird patterns shifted by half cell-width in x, y and both x and y directions.

b) Grouping match neighborhoods (cell-pairs) for robustness. As shown in equations (6), (10), grouping cell-pairs increases partionability. We group cell-pairs using a smooth lateral motion assumption. Thus from equation (6), the score $\mathcal{S}_{ij}$, for cell-pair $\{i, j\}$ is :

$$\mathcal{S}_{ij} = \sum_{k=1}^{K=9} |\mathcal{X}_{i^k j^k}| \tag{13}$$

where $|\mathcal{X}_{i^k j^k}|$ is the number of matches between cells $\{i^k, j^k\}$ shown in Fig. 6. This solution is efficient but limits in-plane rotational invariance. *GMS* is competitive with the ratio-test at extreme rotations as shown in Fig. 8, however performance is better at lower rotations. In practice, rotation can often be estimated by other sensors and OpenCV 3.0 actually auto-rotates all read images to a canonical orientation using their EXIF information. Alternatively, we also provide a scale and rotational version of the algorithm by performing grid-selection over all potential scale-rotation pairs and choosing matches from the pair with the best results.

c) How many grid-cells should be used? More grid-cells improve match localization. However, it reduces n,

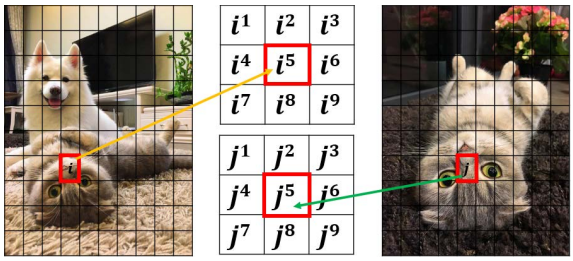


Figure 6. 9 regions around cells $\{i, j\}$ used in score evaluation.

the number of features in each cell and thus partionability. This can be compensated by increasing K but that affects computational time. Our theoretical and empirical results suggest $G = 20 \times 20$ cells for $10,000$ features. This gives n an average value of 25. More features permit finer cells.

d) Thresholding S_{ij} to divide cell-pairs into true and false sets $\{\mathcal{T}, \mathcal{F}\}$.

Fig. 4 shows the desired threshold can be given as $\tau = m_f + \alpha s_f$ where m_f is the mean and αs_f is an appropriate number of standard deviations to ensures most wrong grid-cells are rejected. In practice, m_f is small by design, while α is very large to ensure confident rejection of the large number of wrong cell-pairs. With a slight abuse of notation, the threshold can be approximated as $\tau \approx \alpha\sqrt{s_f} \approx \alpha\sqrt{n}$

This results in a single parameter thresholding function

$$\text{cell-pair } \{i,j\} \in \begin{cases} \mathcal{T}, & \text{if } S_{ij} > \tau_i = \alpha\sqrt{n_i} \\ \mathcal{F}, & \text{otherwise} \end{cases} \quad (14)$$

where $\alpha = 6$ is experimentally determined and n_i is the average (of the 9 grid-cells in Fig. 6) number of features present in a single grid-cell.

3.2. Implementation details

We use OpenCV ORB features. Feature number is fixed at $10,000$, the maximum number permissible for real-time performance. Large, well-textured images can have more than $10,000$ features. This causes features to be poorly distributed (clustered in a corner). Small less-textured images can have far fewer than $10,000$ distinct features. The former problem is addressed by resizing all images to 480×640. The latter by setting FAST [34] feature thresholds to zero. This enables matching in weakly textured environments (see accompanying video). Nearest-neighbors are discovered with brute-force Hamming distance comparisons on the GPU. This runs in parallel with CPU-based feature detection. These are the two most expensive modules. Finally, true matches are sieved out through *GMS* algorithm 1. This step takes $1ms$ in single thread CPU-time. Code is provided at the link below.[2].

[2] **CODE:** https://github.com/JiawangBian/GMS-Feature-Matcher

Algorithm 1. GMS Feature Matcher

Input: One pair of images
Initialization:
1: Detect feature points and calculate their descriptors
2: For each feature in I_b, find its nearest neighbor in I_a
3: Divide two images by G grids respectively
4: **for** $i = 1$ to G **do**
5: $\quad j = 1$;
6: $\quad$ **for** $k = 1$ to G **do**
7: $\quad\quad$ **if** $|\mathcal{X}_{ik}| > |\mathcal{X}_{ij}|$ **then**
8: $\quad\quad\quad j = k$;
9: $\quad\quad$ **end if**
10: $\quad$ **end for**
11: $\quad$ Compute $\mathcal{S}_{ij}, \tau_i$; ▷ Eq. (13)(14)
12: $\quad$ **if** $\mathcal{S}_{ij} > \tau_i$ **then**
13: $\quad\quad Inliers = Inliers \cup \mathcal{X}_{ij}$;
14: $\quad$ **end if**
15: **end for**
Iteration: Repeat from line 4, with gird patterns shifted by half cell-width in the x, y and both x and y directions.
Output: $Inliers$

4. Experiments

GMS is an efficient, effective alternative to the tradition ratio-test used to reject false matches. We evaluate *GMS* on two metrics: a) Its recall, precision and F-measure, $F = 2 \cdot (Precision \cdot Recall)/(Precision + Recall)$, relative to the ratio-test; b) *GMS*'s matchings effectiveness in improving performance of pose estimation used in SLAM [3] and Visual Structure from Motion [39]. We compare *GMS* to fast matchers like *SIFT* [22], *SURF* [2], *ORB* [35], *USAC* [31] and powerful matchers that are orders of magnitude slower, *BF* [16], *BD* [20], *DM* [44], *GAIM* [8], *LIFT* [47].

4.1. Datasets

We evaluate on four datasets, *TUM* [38], *Strecha* [37], *VGG* [25] and *Cabinet* [38], described in Tab. 2. *TUM* has six video sequences with challenging wide-baselines, low-texture and blur. Scenes are shown in Fig. 7. *Strecha* [37][3] and *VGG* [25] are standard datasets with significant wide-baselines and good ground-truth. *Cabinet* (top center in Fig. 7) is a subset of *TUM* which permits separate analysis on low-texture scenes.

Each *TUM* [38] video is divided into sets of 100 frames. The first frame is designated as reference. All frames from a set are matched to the reference provided their relative rotation is less than 30 degrees. A similar process is used on *Strecha*'s [37] dataset, except every image is a reference.

[3] Images are quarter size to accommodate slower algorithms.

Dataset	*TUM* [38]	*Strecha* [37]	*VGG* [25]	*Cabinet* [38]
Full name	RGB-D SLAM Dataset and Benchmark	Dense Multiview Stereo Dataset	Affine Covariant Regions Datasets	A subset of TUM dataset
Image pairs	3141	500	40	578
Ground truth	Camera pose, Depth	Camera pose, 3D model	Homography	Camera pose, Depth
Description	Including all image condition changes	Well-textured images	Viewpoint change, zoom+rotation,blur	Low-texture images with strong light

Table 2. Dataset details. *Strecha* [37] and *VGG* [25] are standard benchmarks. *TUM* [38] and *Cabinet* dataset are VGA resolution videos.

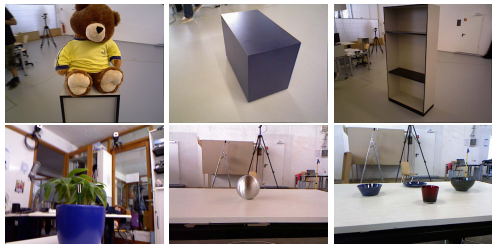


Figure 7. Scenes from the six *TUM* [38] videos. The dataset has many challenges like low texture, blur and strong lighting.

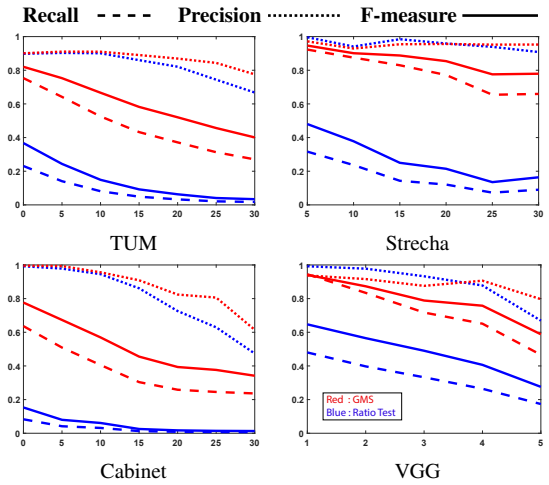


Figure 8. F-measure, Recall and Precision vs baseline (distance between image pairs). *GMS* is in red, *ratio-test* in blue. Baseline of *VGG* [25] represents image-pair order. For other datasets, baseline is represented by relative rotation in degrees. Note that *GMS*'s F-measure consistently outperforms the ratio-test by a large margin.

4.2. Results

We analyze *GMS*'s results against alternative matchers.

Precision & Recall: We compare *GMS* to the traditional *ratio-test* (threshold 0.66). Precision, Recall and F-measure are tabulated in Fig. 8. *GMS* is red and *ratio-test* blue. Each metric is denoted with a unique line style. *GMS*'s F-measure is consistently much higher than the *ratio-test*. This is even true on the *VGG* dataset which has significant in-plane rotation. Recall from Sec. 3.1, our formulation sacrifices rotational performance.

Performance vs speed: Matching speed is arguably as important as performance. Fig. 10 tabulates performance-speed trade-offs. Performance is quantified by the percentage of accurately estimated poses, while speed is measured as the log of time in milliseconds. A pose estimate is deemed correct if its rotation and translation errors are within 5 and 15 degrees respectively. Fig. 10 shows *GMS* (in red) maintains high speed and performance. *GMS*'s performance is much higher than other fast solutions and comparable to solutions like *BF* [16] and *Deep-Matching* [44] which are many orders of magnitude slower. In our experience, nearly all purely feature based techniques can reach real-time through GPU usage. With a computational cost of 1ms on CPU, *GMS* maintains this real-time performance. Full matching with *GMS* runs at 27.8 frames per second.

Full performance curves: To ensure the fairness of Fig. 10's threshold choice, Fig. 11 plots performance curves against different pose thresholds. Observe that *GMS*'s relative performance is unchanged.

Consistency: Prior experiments focus on average results. Fig. 12 illustrates performance variation across different *TUM* [38] scenes. Each box's central mark is the median. Box edges are the 25th and 75th percentiles. Whiskers show performance extrema. Most fast algorithms have poor consistency evidenced by low whiskers. *GMS* (in red) is the most consistent fast algorithm. Its consistency is comparable to much slower algorithms.

Video Results: *GMS* enables wide-baseline feature matching on video data. Fig. 9 provides screen-shots of videos in the supplementary.

5. Conclusion

We propose *GMS*, a statistical formulation for partitioning of true and false matches based on the number of neighboring matches. While this constraint has been implicitly employed by other techniques, our more principled approach enables development of simpler, faster algorithms with nearly equivalent performance. In addition, *GMS* suggests a link between feature numbers and match quality. This may prove an interesting research direction for handling previously intractable matching problems.

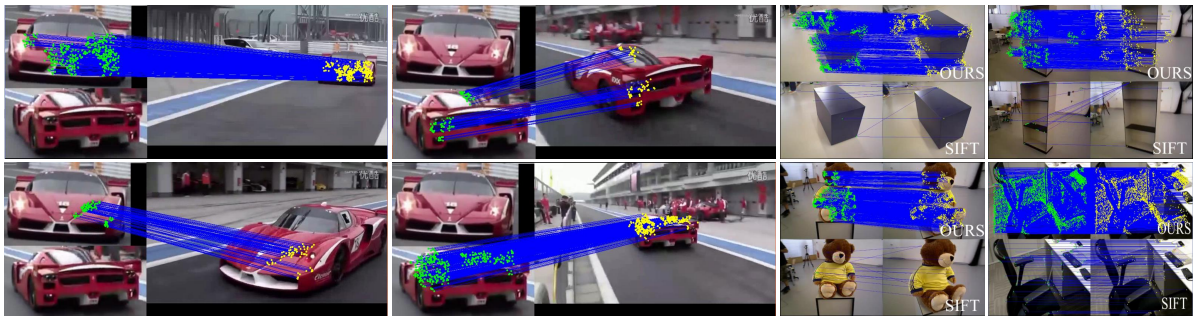
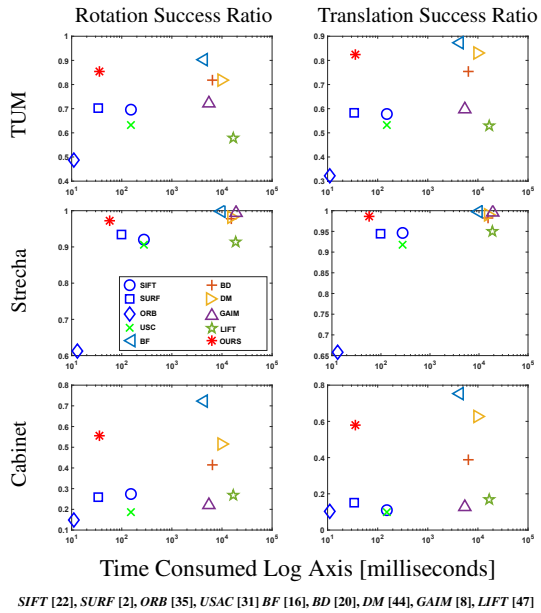


Figure 9. *GMS* enables real-time, wide-baseline matching on videos. These are screen-shots of videos in the supplementary.



SIFT **[22],** ***SURF*** **[2],** ***ORB*** **[35],** ***USAC*** **[31]** ***BF*** **[16],** ***BD*** **[20],** ***DM*** **[44],** ***GAIM*** **[8],** ***LIFT*** **[47]**

Figure 10. Performance vs Speed. Performance: Percentage of correctly estimated poses. Speed: log time. *GMS* (red star) is consistently in the top left, as it is efficient and has performance comparable to techniques many orders of magnitude slower.

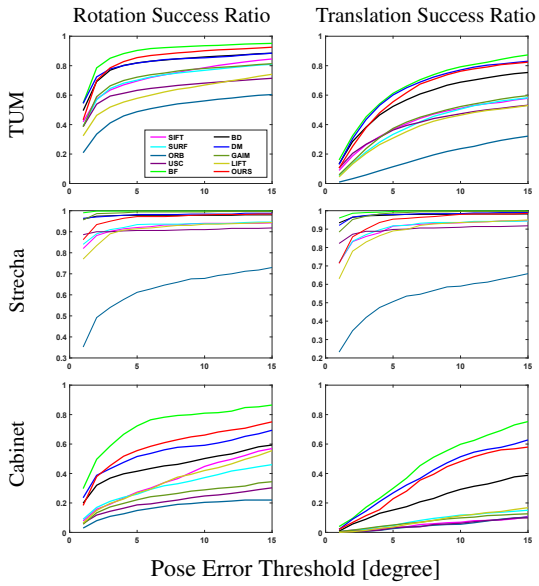


Figure 11. Percentage of accurately estimated poses at different thresholds. *GMS* (red) is consistently near the top. It is only slightly inferior to *BF* and *Deep-Matching* which are orders of magnitude slower.

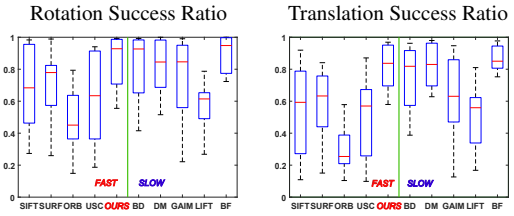


Figure 12. Consistency plots illustrate performance variations on *TUM* [38]. Each box's central mark is the median. Box edges are the 25th and 75th percentiles and whiskers show performance extrema. Relative to other fast algorithms, *GMS* has very short whiskers. This indicates consistency.

Acknowledgments: This work was supported by DSO grant for Highly Accurate Camera Pose by Effective Feature Matching; The research was also supported by NSFC (NO. 61572264, 61620106008), Huawei Innovation Research Program (HIRP), and CAST young talents plan. Sai-Kit Yeung is supported by Singapore MOE Academic Research Fund MOE2016-T2-2-154; Heritage Research Grant of the National Heritage Board, Singapore; SUTD Digital Manufacturing and Design (DManD) Centre which is supported by the National Research Foundation (NRF) of Singapore; the NRF, Prime Minister's Office, Singapore under its IDM Futures Funding Initiative and its Environmental & Water Technologies Strategic Research Programme and administered by the PUB, Singapores National Water Agency; This material is based on research/work supported in part by the National Research Foundation under Virtual Singapore Award No. NRF2015VSG-AA3DCM001-014.

References

[1] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision*,

56(3):221–255, 2004.

[2] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded up robust features. In *Proc. of European Conf. on Computer Vision*, pages 404–417, 2006.

[3] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 943–948. IEEE, 2004.

[4] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

[5] T.-J. Chin, J. Yu, and D. Suter. Accelerated hypothesis generation for multistructure data via preference analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2012.

[6] O. Chum, J. Matas, and J. Kittler. Locally optimized ransac. In *Joint Pattern Recognition Symposium*, pages 236–243. 2003.

[7] V. Codreanu, F. Dong, B. Liu, J. B. Roerdink, D. Williams, P. Yang, and B. Yasar. Gpu-asift: A fast fully affine-invariant feature extraction algorithm. In *Int. Conf. High Performance Computing and Simulation (HPCS)*, pages 474–481, 2013.

[8] T. Collins, P. Mesejo, and A. Bartoli. An analysis of errors in graph-based keypoint matching and proposed solutions. In *European Conference on Computer Vision*, pages 138–153. Springer, 2014.

[9] K. G. Derpanis. The harris corner detector. *York University*, 2004.

[10] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[11] Y. Freund and R. E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

[12] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. In *Proc. of ACM SIGGRAPH*, 2011.

[13] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[14] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[15] M. Kushnir and I. Shimshoni. Epipolar geometry estimation for urban scenes with repetitive structures. *IEEE transactions on pattern analysis and machine intelligence*, 36(12):2381–2395, 2014.

[16] W.-Y. Lin, M.-M. Cheng, J. Lu, H. Yang, M. N. Do, and P. Torr. Bilateral functions for global motion modeling. In *Proc. of European Conf. on Computer Vision*, pages 341–356, 2014.

[17] W.-Y. Lin, S. Liu, N. Jiang, M. N. Do, P. Tan, and J. Lu. Repmatch: Robust feature matching and pose for reconstructing modern cities. In *European Conference on Computer Vision*, pages 562–579. Springer, 2016.

[18] W.-Y. Lin, S. Liu, Y. Matsushita, T.-T. Ng, and L.-F. Cheong. Smoothly varying affine stitching. In *Proc. of Computer Vision and Pattern Recognition*, pages 345–352, 2011.

[19] Y. Lipman, S. Yagev, R. Poranne, D. W. Jacobs, and R. Basri. Feature matching with bounded distortion. *ACM Trans. on Graphics*, 33(3):26, 2014.

[20] Y. Lipman, S. Yagev, R. Poranne, D. W. Jacobs, and R. Basri. Feature matching with bounded distortion. *ACM Transactions on Graphics (TOG)*, 33(3):26, 2014.

[21] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. SIFT flow: Dense correspondence across different scenes. In *Proc. of European Conf. on Computer Vision*, pages 28–42, 2008.

[22] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int'l Journal of Computer Vision*, 60(2):91–110, 2004.

[23] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of Int. Joint Conf. Artificial Intelligence*, pages 674–679, 1981.

[24] J. Maier, M. Humenberger, M. Murschitz, O. Zendel, and M. Vincze. Guided matching based on statistical optical flow for fast and robust correspondence analysis. In *European Conference on Computer Vision*, pages 101–117. Springer, 2016.

[25] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International journal of computer vision*, 65(1-2):43–72, 2005.

[26] J. M. Morel and G. Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.

[27] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, 2(331-340):2, 2009.

[28] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2227–2240, 2014.

[29] A. Myronenko, X. Song, and M. A. Carreira-Perpinan. Non-rigid point set registration: Coherent point drift. In *Proc. of Advances in Neural Information Processing Systems*, pages 1009–1016, 2006.

[30] D. Pizarro and A. Bartoli. Feature-based deformable surface detection with self-occlusion. *Int'l Journal of Computer Vision*, 97(1):54–70, 2012.

[31] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J. Frahm. Usac: A universal framework for random sample consensus. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 35(8):2022–2038, 2013.

[32] R. Raguram, J.-M. Frahm, and M. Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. In *Proc. of European Conf. on Computer Vision*, pages 500–513, 2008.

[33] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1164–1172, 2015.

[34] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European conference on computer vision*, pages 430–443. Springer, 2006.

[35] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Proc. of Int'l Conf. on Computer Vision*, pages 2564–2571, 2011.

[36] T. Sattler, B. Leibe, and L. Kobbelt. Scramsac: Improving ransac's efficiency with a spatial consistency filter. In *Proc. of Int'l Conf. on Computer Vision*, pages 2090–2097, 2009.

[37] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. Ieee, 2008.

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

[39] C. Sweeney, T. Hollerer, and M. Turk. Theia: A fast and scalable structure-from-motion library. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 693–696. ACM, 2015.

[40] T. B. Terriberry, L. M. French, and J. Helmsen. Gpu accelerating speeded-up robust features. In *Proc. Int. Symp. on 3D Data Processing, Visualization and Transmission (3DPVT)*, pages 355–362. Citeseer, 2008.

[41] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Trans. on Computer Vision and Image Understanding*, 78:138–156, 2000.

[42] C. Wang, L. Wang, and L. Liu. Density maximization for improving graph matching with its applications. *IEEE Trans. on Image Processing*, pages 2110–2123, 2015.

[43] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *Proc. of Int'l Conf. on Computer Vision*, pages 1385–1392, Dec. 2013.

[44] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE Intenational Conference on Computer Vision (ICCV)*, Sydney, Australia, Dec. 2013.

[45] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu`, 2007.

[46] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1744–1757, 2012.

[47] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. Lift: Learned invariant feature transform. *European Conference on Computer Vision*, 2016.

[48] A. L. Yuille and N. M. Grzywacz. The motion coherence theory. In *Proc. of Int'l Conf. on Computer Vision*, pages 344–353, 1988.