GMS: Grid-based motion statistics for fast, ultra-robust feature correspondence

Jiawang BIAN

Wen-yan LIN

*Singapore Management University*, daniellin@smu.edu.sg

Matsushita YASUYUKI

Sai-Kit YEUNG

Tan-Dat NGUYEN

*See next page for additional authors*

Follow this and additional works at: [https://ink.library.smu.edu.sg/sis_research](https://ink.library.smu.edu.sg/sis_research)

Part of the Computer and Systems Architecture Commons

Citation

BIAN, Jiawang; LIN, Wen-yan; YASUYUKI, Matsushita; YEUNG, Sai-Kit; NGUYEN, Tan-Dat; and CHENG, Ming-Ming. GMS: Grid-based motion statistics for fast, ultra-robust feature correspondence. (2017). *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition: CVPR 2017, Honolulu, USA, July 21-26. 2828-2837*. Research Collection School Of Information Systems. *Available at*: [https://ink.library.smu.edu.sg/sis_research/4805](https://ink.library.smu.edu.sg/sis_research/4805)

This Conference Proceeding Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email library@smu.edu.sg.
Author
Jiawang BIAN, Wen-yan LIN, Matsushita YASUYUKI, Sai-Kit YEUNG, Tan-Dat NGUYEN, and Ming-Ming CHENG

This conference proceeding article is available at Institutional Knowledge at Singapore Management University:
https://ink.library.smu.edu.sg/sis_research/4805
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

Sai-Kit Yeung
Singapore University of Technology and Design

Yasuyuki Matsushita
Osaka University

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 [40, 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.
by simply counting the number of matches in their neighborhood. From the law-of-large-numbers, the partitionability of true and false scales to infinity with match numbers. The mathematical analysis is straightforward 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 [13,27,28]. 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 [13,23,43,33,21,11,46], 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. \( X = \{x_1, x_2, \ldots, x_i, \ldots, x_N\} \) is the set of all nearest-neighbor feature matches from \( I_a \) to \( I_b \). \( X \) has cardinality \( |X| = N \). Our goal is to divide \( X \) 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. \( X_i \subseteq X \) is the subset of matches between regions \( \{a, b\} \) of match \( x_i \). \( S_i \) is a measure of neighborhood support:

\[
S_i = |X_i| - 1, \quad (1)
\]

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 region \( a \). 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.

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

\[
p(f^b_a|f^b_a) = \beta m \big/ M \quad (2)
\]

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

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

\[
p_t = p(f^b_a|T^{ab}) = p(f^a_b|T^{ab}) + p(f^b_a|T^{ab}) = p(f^a_b|T^{ab}) + (1 - t)p(f^b_a|f^b_a) = t + (1 - t)\beta m \big/ M \quad (3)
\]

Explanation: Fig. 3(i) shows event \( f^b_a \) 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, \( p(f^a_b), p(f^b_a) \) are independent of \( T^{ab} \). Due to assumption 2 \( p(f^b_a|f^b_a) \) is also independent of \( T^{ab} \). Dropping the conditioning \( T^{ab} \) and substituting values from Tab. 1 and equation (2) gives the final expression.
From Fig. 3(ii) we know event of \( f_{S} \) overall pdf of \( S \) is that true and false matches have neighborhood scores, that are independent of line of equation (4). Similar to equation (3), the probabil-

\[
S_t \sim \begin{cases} 
B(K,n,p_t), & \text{if } x_i \text{ is true} \\
B(K,n,p_f), & \text{if } x_i \text{ is false} 
\end{cases}
\]  

(7)

The respective mean and standard deviation of \( S_t \)'s distribution are

\[
\{m_t = Kn p_t, s_t = \sqrt{Knt(1-p_t)}\} \text{ if } x_i \text{ is true} \\
\{m_f = Kn p_f, s_f = \sqrt{Knp(1-p_f)}\} \text{ if } x_i \text{ is false}
\]  

(8)

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

\[
P = \frac{m_t - m_f}{s_t + s_f} = \frac{Knp_t - Knp_f}{\sqrt{Knpt(1-p_t)} + \sqrt{Knpt(1-p_f)}}.
\]

(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:

\[
P \propto \sqrt{K/n}.
\]

(10)

This means, provided \( m_t > m_f \), the partitionability 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 de-
sire, with perfect precision and recall, provided the number of features is sufficiently large. This forms a direct link from the assumptions 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 in-
crease \( K \) 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
features, some typical values would be \(\{m = n = 25, \beta = 1, t = 0.5, K = 1\}\). Thus, the mean and standard deviations of \(S_t\) in the basic constraint of equation (8) are:

\[
\{m_t = 12.5, s_t = 2.5\}, \quad \{m_f = 0.03, s_f = 0.176\}, \quad P = 4.7
\]

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 partitionability with \(t\), percentage match correctness is given by

\[
\lim_{t \to 1} m_t \to Kn, \quad \lim_{t \to 1} m_f \to 0, \quad \lim_{t \to 1} P \to \infty. \quad (12)
\]

As nearest neighbor matching tends to perfection, \(m_t\) tends to its maximum value, \(m_f\) to its minimum and partionably to \(\infty\). 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.

3.1. 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 grid 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 \(S_{ij}\), for cell-pair \(\{i, j\}\) is:

\[
S_{ij} = \sum_{k=1}^{K=9} |\chi_{[i, j, k]}| \quad (13)
\]

where \(|\chi_{[i, j, k]}|\) 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 \(n\),
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 \( \{T, F\} \).

Fig. 4 shows the desired threshold can be given as \( \tau = m_f + \alpha_s f \) where \( m_f \) is the mean and \( \alpha_s f \) is an appropriate number of standard deviations to ensure most wrong grid-cells are rejected. In practice, \( m_f \) is small by design, while \( \alpha \) 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

\[
\text{cell-pair } \{i, j\} \in \begin{cases} T, & \text{if } S_{ij} > \tau_i = \alpha \sqrt{m_i} \\ F, & \text{otherwise} \end{cases}
\tag{14}
\]

where \( \alpha = 6 \) is experimentally determined and \( m_i \) 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 \( 640 \times 480 \) problem is addressed by resizing all images to

\[
\text{Performance of pose estimation used in SLAM} \[3\] and \text{Visuual Structure from Motion} \[39\]. We compare GMS to fast matchers like \text{SIFT} \[22\], \text{SURF} \[2\], \text{ORB} \[35\], \text{USAC} \[31\] and powerful matchers that are orders of magnitude slower, \text{BF} \[16\], \text{BD} \[20\], \text{DM} \[44\], \text{GAIM} \[8\], \text{LIFT} \[47\].

### 4.1. Datasets

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

Each \text{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 \text{Strecha}’s \[37\] dataset, except every image is a reference.

\begin{algorithm}
  \begin{algorithmic}[1]
    \caption{GMS Feature Matcher}
    \begin{itemize}
      \item [\hspace{1cm}] \textbf{Input:} One pair of images
      \item [\hspace{1cm}] \textbf{Initialization:}
        \begin{enumerate}
          \item Detect feature points and calculate their descriptors
          \item For each feature in \( I_b \), find its nearest neighbor in \( I_a \)
          \item Divide two images by \( G \) grids respectively
        \end{enumerate}
      \item [\hspace{1cm}] for \( i = 1 \) to \( G \) do
        \begin{enumerate}
          \item \( j = 1 \)
          \item for \( k = 1 \) to \( G \) do
            \begin{enumerate}
              \item if \( |X_{ik}| > |X_{ij}| \) then
                \begin{enumerate}
                  \item \( j = k \)
                \end{enumerate}
              \end{enumerate}
            \end{enumerate}
        \end{enumerate}
      \item [\hspace{1cm}] end if
      \item [\hspace{1cm}] end for
    \end{itemize}
    \begin{itemize}
      \item [\hspace{1cm}] \textbf{Iteration:} Repeat from line 4, with grid patterns shifted by half cell-width in the \( x \), \( y \) and both \( x \) and \( y \) directions.
      \item [\hspace{1cm}] \textbf{Output:} \text{Inliers}
    \end{itemize}
  \end{algorithmic}
\end{algorithm}

\[\text{Algorithm 1. GMS Feature Matcher}\]

\[\text{Input: One pair of images}\]

\[\text{Initialization:}\]

\[1: \text{Detect feature points and calculate their descriptors}\]

\[2: \text{For each feature in } I_b, \text{find its nearest neighbor in } I_a\]

\[3: \text{Divide two images by } G \text{ grids respectively}\]

\[4: \text{for } i = 1 \text{ to } G \text{ do}\]

\[5: \text{ } j = 1;\]

\[6: \text{ } \text{for } k = 1 \text{ to } G \text{ do}\]

\[7: \text{ } \text{if } |X_{ik}| > |X_{ij}| \text{ then}\]

\[8: \text{ } j = k;\]

\[9: \text{ } \text{end if}\]

\[10: \text{ } \text{end for}\]

\[11: \text{ } \text{Compute } S_{ij}, \tau_i; \quad \triangleright \text{Eq. (13)(14)}\]

\[12: \text{ } \text{if } S_{ij} > \tau_i \text{ then}\]

\[13: \text{ } \text{Inliers } = \text{Inliers } \cup X_{ij};\]

\[14: \text{ } \text{end if}\]

\[15: \text{ } \text{end for}\]

\[\text{Iteration:} \text{Repeat from line 4, with grid patterns shifted by half cell-width in the } x, y \text{ and both } x \text{ and } y \text{ directions.}\]

\[\text{Output: } \text{Inliers}\]

\[\text{4. Experiments}\]

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, \( F = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \), 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 \text{SIFT} \[22\], \text{SURF} \[2\], \text{ORB} \[35\], \text{USAC} \[31\] and powerful matchers that are orders of magnitude slower, \text{BF} \[16\], \text{BD} \[20\], \text{DM} \[44\], \text{GAIM} \[8\], \text{LIFT} \[47\].

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

\[\text{4.1. Datasets}\]

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

Each \text{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 \text{Strecha}’s \[37\] dataset, except every image is a reference.

\[\text{Images are quarter size to accommodate slower algorithms.}\]
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. 11’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.

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, Singapore’s 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,
[1] 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.

[2] Y. Freund and R. E. Schapire. A desicion-theoretic gener-alization of on-line learning and an application to boosting. In Proceedings of the thirtieth annual ACM symposium on Theory of Computing, pages 129–138. ACM, 1998.

[3] 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.

[4] W.-Y. Lin, S. Liu, N. Jiang, M. N. Do, and P. Torr. Bilateral functions for global motion modeling. In Proc. of European Conf. on Computer Vision, pages 341–356, 2014.

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

[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. Rep-match: Robust feature matching and pose for reconstructing modern cities. In European Conference on Computer Vision, pages 562–579. Springer, 2016.
[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 International 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](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.