Geometry-aware Feature Matching for Structure from Motion Applications

Rajvi Shah Vanshika Srivastava P J Narayanan
Center for Visual Information Technology, IIIT Hyderabad, India
{rajvi.shah@research., vanshika.srivastava@students., pjn@}iiit.ac.in

Abstract

We present a two-stage, geometry-aware approach for matching SIFT-like features in a fast and reliable manner. Our approach first uses a small sample of features to estimate the epipolar geometry between the images and leverages it for guided matching of the remaining features. This simple and generalized two-stage matching approach produces denser feature correspondences while allowing us to formulate an accelerated search strategy to gain significant speedup over the traditional matching. The traditional matching punitively rejects many true feature matches due to a global ratio test. The adverse effect of this is particularly visible when matching image pairs with repetitive structures. The geometry-aware approach prevents such preemptive rejection using a selective ratio-test and works effectively even on scenes with repetitive structures. We also show that the proposed algorithm is easy to parallelize and implement it on the GPU. We experimentally validate our algorithm on publicly available datasets and compare the results with state-of-the-art methods.

1. Introduction

Geometrically meaningful feature matching is crucial to many stereo-vision and structure from motion (SFM) applications. Recent methods in 3D reconstruction [15, 2, 5] begin with match-graph construction, which requires matching SIFT-like features [11] across several unordered images of a static scene. Typically, features between two images are matched by computing L2 distances between the descriptors of features in one image against that of the other image and finding the closest feature as a candidate match. However, L2 distance in descriptor space is not sufficiently meaningful by itself to indicate a match. Due to spurious detection and clutter, the closest feature may not be the true match. Hence, it is common to verify the candidate match by a *ratio test*. The ratio test compares the distance of a query feature from its closest neighbor (candidate) to its second closest neighbor in the target image. If the ratio of distances is below a threshold then the candidate is declared a match. The assumption is that features in an image are randomly distributed in descriptor space. In the absence of a true match, the candidate would be an arbitrary feature and the best distance would not be significantly better than the second-best distance; leading to a ratio close to 1 [11].

There are two main problems with this approach. First, images captured by high-resolution cameras have tens of thousands of features, making descriptor comparison for several thousand such image pairs a daunting task for SFM applications. Most large scale reconstruction methods [2, 5, 19] either downsample the images or restrict the number of features considered for matching to cope with this problem. Second, the assumption of randomly distributed features does not hold true for images with repetitive structures such as windows, arches, pillars, etc. in architectural images. The ratio test punitively rejects many correspondences for features on such repeating elements due to similar appearance. Previous efforts that propose tailored solutions to mitigate this problem involve extra processing steps and make additional assumptions [10, 20, 17].

We propose a geometry-aware approach for fast and generalized feature matching that also works for images with repetitive structures without any assumptions or complex processing. Our algorithm first estimates the epipolar geometry between two images by reliably matching small subsets of features from both images. This is followed by an efficiently formulated geometry-aware correspondence search, where each query feature is quickly compared only against a small number of features that lie close to the corresponding epipolar line. This simple trick reduces the false rejections on repetitive structures significantly by avoiding the duplicates from comparison and recovers denser matches as compared to global matching (see Figure 1).

We leverage the epipolar constraints to optimize several steps of geometry-guided correspondence search to make the proposed matching approach efficient. Our algorithm is significantly faster than commonly used Kd-tree based feature matching [12, 3] and has only slightly worse runtime as compared to the recently proposed cascade hashing based approach [8] while producing many folds more correspondences. We further show that the proposed algorithm is easy

to parallelize and implement it on a GPU. The GPU implementation of our matching also performs faster than the global matching on a GPU [1]. We validate our matching by manually verifying the matches on four image pairs of different scene types. We also use our algorithm to construct match-graphs for 3D reconstruction of three SFM datasets and show that our matching leads to denser and complete models as compared to the unguided feature matching.

2. Background and Related Work

In this section, we briefly revisit the feature matching pipeline in the context of SFM. We also discuss the commonly used techniques for efficient feature matching and overview prior efforts on matching and geometry estimation in the presence of repetitive elements.

Feature Matching for SFM applications: Geometrically consistent feature correspondences are established by *(i)* finding feature matches by descriptor comparison and ratio test; *(ii)* estimating fundamental matrix using robust estimators; and *(iii)* removing the outlier matches using epipolar constraint. If the end-goal is to reliably estimate the fundamental matrix, relatively few good feature matches would suffice. However, for many SFM applications it is desirable to find as many correspondences as possible so as to establish connections with newer images and to produce denser point clouds. Since large scale reconstruction involves feature matching for thousands of image pairs, efficiency of matching algorithm is also crucial.

Efficient Feature Matching: A brute-force approach exhaustively computes distances between all features, requiring $O(n^2)$ comparisons, where n is the average number of features in the image. Typical internet images of monuments have tens of thousands of features, making brute-force search impractical for matching thousands of image pairs for SFM. Kd-tree based approximate near-neighbor methods are most commonly used for SFM applications [3, 12]. These methods use efficient data structures to store features resulting in faster search. The average time complexity for these methods is $O(n \log n)$. Recently, hashing based approximate feature matching methods have also been proposed for SFM [16, 8]. These methods convert feature descriptors into compact binary codes and employ fast search strategies to compare binarized features. LDA-Hash [16] is a data-dependent algorithm; it uses linear discriminant analysis (LDA) for feature binarization and requires manual labeling for training. Recently proposed cascade hashing algorithm (CascadeHash) [8] uses data-independent, locality sensitive hashing (LSH) for binarization and has state-of-the-art runtime performance. We show that our algorithm is significantly faster than Kd-tree based methods and performs similarly to CascadeHash while producing more matches. For high-dimensional data, Kd-tree and hashing based approximate methods are difficult to parallelize due to the hierarchical nature of search; whereas our algorithm can easily be distributed on parallel platforms like GPUs for further speedup.

Repetitive Structures and SFM: Repetitive structures pose several problems for geometry estimation and reconstruction. The ratio test produces conservative matches only for distinctive and non-repeated featured. For images with severe repetition such as skyscrapers and urban buildings, this can lead to insufficient reliable matches for geometry estimation. In [10], this problem is handled for building facades by first clustering similar features and then matching clusters using fronto-parallel assumptions. [20, 17] introduce a-contrario frameworks for robust RANSAC that can deal with repetitive patterns while recovering underlying geometry. The focus of these methods is on recovering the correct geometry and not on establishing more correspondences. Our method can use the estimated geometry to establish further feature matches efficiently, crucial for SFM. [13, 9, 18] focus on eliminating the aliasing problems that arise while reconstructing scenes with large duplicate structures such as domes, towers, etc. Aliasing occurs when large number of incorrect matches on repeated elements form consistent sets. [9, 13] jointly optimize the correctness of a match and camera pose while using image cues such as missing correspondences and time-stamps as priors. [18] propose a graph-topological measure based on local visibility structures to evaluate goodness of a feature track. The focus of these approaches is on removing bad tracks/matches while post-processing and not on directly improving the feature matching. In this paper, we do not attempt to solve the specific problems posed by repetitive structures. Our approach is generalized and does not treat repetition as a special case. Since we use geometry-guided correspondence search, our approach can also handle images with repetitive structures.

3. Overview

Our Approach: Most matching methods perform a global ratio test followed by epipolar verification, preemptively rejecting many correspondences on repeating elements. We suggest that if epipolar constraints are used before the ratio test, many true matches on repetitive structures can be retained. Since estimating epipolar geometry requires feature matches, there is a cyclic dependency. We overcome this problem by a two-stage matching approach. The first stage selects a small subset of features from both images and performs Kd-tree based feature matching within the subsets. The fundamental matrix estimated using the matches from this stage is then used to perform an efficiently formulated guided correspondence search as ex-

plained in Section 4. For each query feature, only a small subset of features that lie close to the epipolar line are considered as *candidate matches*. Distance comparison and ratio test are performed only within this set.

Avoiding Preemptive Rejection: If repetition in the image is not along the direction of camera motion, the duplicates of a true match are excluded from the candidate matches as they don't lie close to the epipolar line. Hence, geometry-aware matching reduces the number of false negatives significantly by avoiding duplicates from being considered for ratio test. Figure 1 shows this effect on an example image pair. The corresponding features on repeating arches are marked by red points and highlighted by circles. The query feature (left image) and its corresponding epipolar line (right image) are marked in green. The query feature and its top two neighbors are shown above the target image for both: global matching vs. geometry-aware matching. It can be seen that the correspondence would only survive for geometry-aware matching.

Effect on spurious matches: The constrained ratio test would still be able to reject noisy matches since unlike structured repetition, spurious features are randomly distributed. For a subset of features sampled along the epipolar line, duplicate of a true correspondence would only be sampled when the line is along the direction of repetition. However, an arbitrary non-distinctive spurious candidate match is equally likely to encounter a similar feature in any randomly drawn subset.

Computational Overhead: The time required for initial matching and geometry estimation in first stage of our algorithm is relatively small since fewer features are involved. The second stage, i.e. geometry-guided matching, needs to find the set of candidate matches for each query feature. This requires computing distances of all feature points in target image from the epipolar line for each query. Moreover, since the candidate set is different for each query, a smaller but new Kd-tree needs to be created to find the two closest points for each query. A naïve solution would result in significant computational overhead. In the next section, we explain each step of our algorithm in detail and show that the guided search can be optimized to reduce the computational overhead by leveraging epipolar constraints.

4. Geometry-aware Feature Matching

Given an image pair to match, we first estimate the fundamental matrix by matching a small subset of features and use it to perform a fast correspondence search for the unmatched features. These correspondences can further be verified and pruned using triplet verification or closure check while forming feature tracks for SFM. We explain these steps in detail in the following subsections.

(a) Query Image (b) Target Image

Figure 1: Global vs. geometry-aware matching for repetitive structures: correct match is detected for both methods, but 2-NN for global matching would fail the ratio test due to similarity.

4.1. Geometry Estimation

We select the top 20% SIFT features from the query images in decreasing order of their scales and match them using the Kd-tree based technique. Our preference of top-scale features is based on the observations that top-scale SIFTs match well and they match well with other top-scale features [14, 19]. If at least 16 matches are found between the feature subsets, then these matches are used to estimate the fundamental matrix F between the images using RANSAC [4] and the 8-point algorithm [6] taking into account degenerate cases. The estimated geometry is considered reliable only if at least $>\frac{2}{3}$ of the initial matches are inliers; the image pair is not processed further otherwise. Once the F-matrix is computed, guided search is used to match the remaining features as explained next.

4.2. Geometry Guided Matching

Given $\mathrm{I_s}$ and $\mathrm{I_t}$, two $\mathsf{M} \times \mathsf{N}$ input images, their corresponding feature sets $\mathbb{S}$ and $\mathbb{T}$ are defined as,

$$\mathbb{S} = \{(\mathsf{x}, \mathsf{y}, \mathbf{v}) \mid \mathsf{x} \in [0, \mathsf{M}],\, \mathsf{y} \in [0, \mathsf{N}],\, \mathbf{v} \in \mathbb{R}^{\mathsf{k}}\} \quad (1)$$

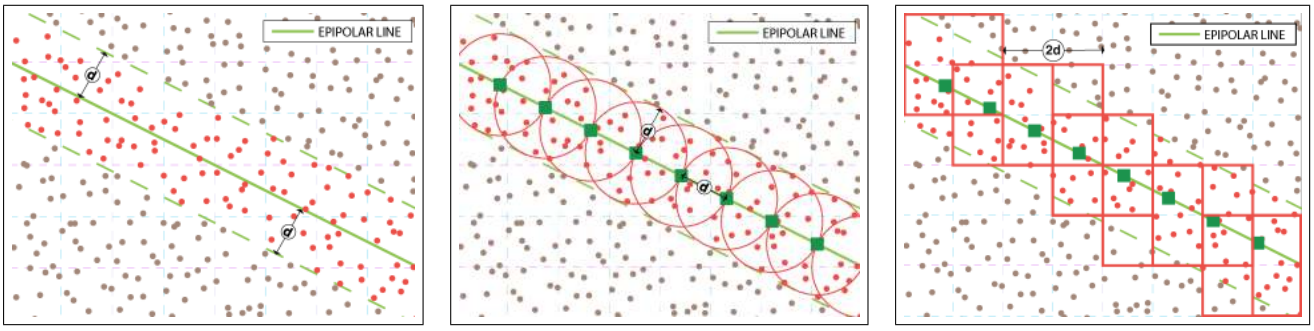
$$\mathbb{T} = \{(\mathsf{x}', \mathsf{y}', \mathbf{v}') \mid \mathsf{x}' \in [0, \mathsf{M}],\, \mathsf{y}' \in [0, \mathsf{N}],\, \mathbf{v}' \in \mathbb{R}^{\mathsf{k}}\} \quad (2)$$

where (x, y), $(\mathsf{x}', \mathsf{y}')$ denote the coordinates of features in image space and $\mathbf{v}$,$\mathbf{v}'$ denote the corresponding k-dimensional feature descriptors ($\mathsf{k} = 128$ for SIFT).

Epipolar constraints: For a query feature point $\mathsf{p_q} = (\mathsf{x_q}\ \mathsf{y_q}\ 1)$ in feature set $\mathbb{S}$ of image $\mathrm{I_s}$ the corresponding epipolar line $\mathsf{l_q} = (\mathsf{a_q}, \mathsf{b_q}, \mathsf{c_q})$ in image $\mathrm{I_t}$ is given by $\mathsf{l_q} = \mathsf{F} \cdot \mathsf{p_q}$. If $\mathsf{p'_q} = (\mathsf{x'_q}\ \mathsf{y'_q}\ 1)$ denotes the corresponding feature point in image $\mathrm{I_t}$ then as per the epipolar constraint $\mathsf{p'_q} \cdot \mathsf{F} \cdot \mathsf{p_q} = 0$, point $\mathsf{p'_q}$ must lie on the epipolar line i.e. $\mathsf{p'_q} \cdot \mathsf{l_q} = 0$. Due to inaccuracies in estimation, it is practical to relax the constraint to $\mathsf{p'_q} \cdot \mathsf{l_q} < \epsilon$. To find $\mathsf{p'_q}$, instead of considering all features in set $\mathbb{T}$, we limit our search to only those features which are close to the epipolar line $\mathsf{l_q}$. We define the set of candidate feature matches $\mathbf{C}$ as,

$$\mathbf{C} = \{\mathsf{p}' \mid \mathrm{dist}(\mathsf{p}', \mathsf{l_q}) \leq \mathrm{d}\} \quad (3)$$

$$\mathrm{dist}(\mathsf{p}', \mathsf{l_q}) = \frac{\mathsf{a_q x}' + \mathsf{b_q y}' + \mathsf{c_q}}{\sqrt{\mathsf{a_q}^2 + \mathsf{b_q}^2}} \quad (4)$$



(a) Linear Search $O(|\mathbb{T}|)$ (b) Radial Search $O(\mathsf{K}\log|\mathbb{T}|)$ (c) Grid based Search $O(1)$

Figure 2: Illustration of the geometry-aware feature search strategy. Search for points within distance d from the epipolar line (shown by red dots) can be approximated by radial search and more efficient grid based search. Red squares in (c) show the center-most cell of the overlapping grids selected for each equidistant points along the epipolar line.

Linear search for candidates: In Figure 2 the candidate feature matches (features in set $\mathbf{C}$) are marked by red dots. Finding these candidate matches using linear search would require computing the distances of all features in $\mathbb{T}$ from line l_q using equation (4). This search has a time complexity of $O(|\mathbb{T}|)$. Linear search can be approximated by a radial search algorithm of logarithmic complexity.

Radial search for candidates: In this search, first a Kd-tree of (x,y) coordinates of features in $\mathbb{T}$ is constructed. Then K equidistant points (at distance d) on the epipolar line l_q are sampled and each of these points is queried into the Kd-tree to retrieve feature points within radial distance d from the sampled point [12]. In Figure 2b dark green squares on the epipolar line mark the equidistant query points and red circles indicate coverage of *true* candidate matches when radial search is used. If line l_q intersects image I_t in points $\mathsf{p}_\mathsf{A}=(\mathsf{x}_\mathsf{A},\mathsf{y}_\mathsf{A})$ and $\mathsf{p}_\mathsf{B}=(\mathsf{x}_\mathsf{B},\mathsf{y}_\mathsf{B})$ then the coordinates $(\mathsf{x}_\mathsf{k},\mathsf{y}_\mathsf{k})$ of the equidistant points are given by,

$$\mathsf{x}_\mathsf{k}=\frac{\mathsf{k}\cdot\mathsf{x}_\mathsf{A}+(\mathsf{K}-\mathsf{k})\cdot\mathsf{x}_\mathsf{B}}{\mathsf{K}},\quad \mathsf{k}=0,1,2,\cdots,\mathsf{K} \tag{5}$$

$$\mathsf{y}_\mathsf{k}=\frac{\mathsf{k}\cdot\mathsf{y}_\mathsf{A}+(\mathsf{K}-\mathsf{k})\cdot\mathsf{y}_\mathsf{B}}{\mathsf{K}},\quad \mathsf{k}=0,1,2,\cdots,\mathsf{K} \tag{6}$$

where $\mathsf{K}=\sqrt{(\mathsf{x}_\mathsf{B}-\mathsf{x}_\mathsf{A})^2+(\mathsf{y}_\mathsf{B}-\mathsf{y}_\mathsf{A})^2}/\mathrm{d}$. This search has a complexity of $O(\mathsf{K}\cdot\log|\mathbb{T}|)$ where $\mathsf{K}\ll|\mathbb{T}|$.

Grid based Search for candidates: We further optimize the candidate search to $O(1)$ using a grid based approach. We first divide the target image I_t into four overlapping grids of cell size $2\mathrm{d}\times2\mathrm{d}$, depicted by the dotted lines in Figure 2c. The origins of these grids are respectively at $(0,0)$, $(0,\mathrm{d})$, $(\mathrm{d},0)$, and (d,d). We then bin all feature points of $\mathbb{T}$ into cells of the overlapping grids based on their image coordinates. Each feature point (x,y) would fall into four cells, the coordinates of centers of these cells are given by,

$$\mathsf{x}_{\mathsf{c}1}=\lfloor\frac{\mathsf{x}}{2\mathrm{d}}\rfloor\cdot\mathrm{d}+2\mathrm{d},\qquad \mathsf{y}_{\mathsf{c}1}=\lfloor\frac{\mathsf{y}}{2\mathrm{d}}\rfloor\cdot\mathrm{d}+2\mathrm{d} \tag{7}$$

$$\mathsf{x}_{\mathsf{c}2}=\lfloor\frac{\mathsf{x}}{2\mathrm{d}}-\frac{1}{2}\rfloor\cdot\mathrm{d}+2\mathrm{d},\ \mathsf{y}_{\mathsf{c}2}=\lfloor\frac{\mathsf{y}}{2\mathrm{d}}\rfloor\cdot\mathrm{d}+2\mathrm{d} \tag{8}$$

$$\mathsf{x}_{\mathsf{c}3}=\lfloor\frac{\mathsf{x}}{2\mathrm{d}}\rfloor\cdot\mathrm{d}+2\mathrm{d},\qquad \mathsf{y}_{\mathsf{c}3}=\lfloor\frac{\mathsf{y}}{2\mathrm{d}}-\frac{1}{2}\rfloor\cdot\mathrm{d}+2\mathrm{d} \tag{9}$$

$$\mathsf{x}_{\mathsf{c}4}=\lfloor\frac{\mathsf{x}}{2\mathrm{d}}-\frac{1}{2}\rfloor\cdot\mathrm{d}+2\mathrm{d},\ \mathsf{y}_{\mathsf{c}4}=\lfloor\frac{\mathsf{y}}{2\mathrm{d}}-\frac{1}{2}\rfloor\cdot\mathrm{d}+2\mathrm{d} \tag{10}$$

Given a query point p_q, we find its epipolar line l_q and the equidistant points $(\mathsf{x}_\mathsf{k},\mathsf{y}_\mathsf{k})$ as per equations (5) and (6). For each of the equidistant points on the epipolar line, we find the four overlapping cells that contain this point and find its Cartesian distance from centers of the four cells. We select the cell with the shortest such distance for each point and accumulate all feature points binned into these cells to obtain an approximate set of candidate matches $\mathbf{C}'$. Red squares in Figure 2c indicate the coverage of true candidate matches in set $\mathbf{C}$ by grid based approximate search. In practice, we use the larger grid size $(2\mathrm{d}\times2\mathrm{d})$ to account for misses due to the grid approximation. Since feature points are binned only once per image, the time complexity for searching candidate matches is $O(1)$ in grid based approach.

Finding the match: To finalize a match from the candidate set $\mathbf{C}'$, a Kd-tree of descriptors in $\mathbf{C}'$ is constructed, two features closest from the query are retrieved, and the ratio test is performed. The number of candidate feature matches $|\mathbf{C}'|$ is a small fraction of total points $|\mathbb{T}|$ (typically 200:1 in our experiments), reducing the size of the Kd-tree notably. Geometry-aware search reduces the number of operations required for two-image matching from $(|\mathbb{S}|\log|\mathbb{T}|)$ to $(|\mathbb{S}|\log|\mathbf{C}'|)$, with $|\mathbf{C}'|\log|\mathbf{C}'|$ overhead of constructing a small Kd-tree of size $|\mathbf{C}'|$ for each query feature. Though the asymptotic complexity of our algorithm is still $O(n\log n)$, we observe a significant speed up in practical runtime since we deal with much smaller constants.

Avoiding redundant Kd-tree construction: To reduce the overhead of redundant Kd-tree construction, we exploit the dual nature of epipolar lines; i.e. for all points that lie on line l in image $\mathrm{I_t}$, their corresponding points must lie on the dual line l' in image $\mathrm{I_s}$. We use this property, to group the query points in $\mathbb{S}$ whose epipolar lines intersect the boundaries of $\mathrm{I_t}$ in nearby points (within 2 pixels) and search for matches group by group. Since all feature points in a group have the same epipolar line and hence the same candidate matches, we avoid redundant construction of the small Kd-tree of size $|\mathbf{C}'|$ for all points in a group.

4.3. Verification of Feature Matches

The matches produced by our approach are mostly geometrically consistent. However, the selective ratio test can introduce a small number of false positives ($< 10\%$). Robust estimators like RANSAC can easily deal with such a small percentage of outliers. A bad initial estimate of fundamental matrix can sometimes lead to spurious correspondences. Though we seldom experienced this in our experiments, in practice this can be detected by re-estimating the F-matrix using all matches and checking the inlier ratio.

If strictly true matches are required, *triplet verification* or *closure check* can be performed while building feature tracks for SFM applications. Feature tracks are formed by finding connected-components in a graph where image-feature pair is a node and edges indicate matches between features. Triplet verification demands that if a feature A in image $\mathrm{I_A}$ is a match for a feature B in image $\mathrm{I_B}$ then the correspondence $\mathsf{A} \leftrightarrow \mathsf{B}$ must be verified by some feature C in a commonly connected image $\mathrm{I_C}$ such that $\mathsf{C} \leftrightarrow \mathsf{B}$ and $\mathsf{A} \leftrightarrow \mathsf{C}$ are also correspondences. This can be ensured by finding triangles in each connected-component (track) and pruning all nodes that are not part of any triangle. Closure check is less strict and demands that each feature track be a closed connected-component in the graph.

5. GPU Implementation

The proposed algorithm is well suited for parallel computation on the GPU. We use high performance CUDA parallel primitives like sort and reduce from the Thrust [7] library and obtain a significant speedup over CPU. Exploiting parallelism in Kd-tree based hierarchical search on high-dimensional data is difficult. GPU flann [12] works only for up to 3-dim data. On the contrary, exact near-neighbor (ExNN) search can be performed efficiently by using parallel matrix multiplication for pairwise descriptor distance computation on a GPU [1]. Hence, we use ExNN approach to find initial matches between feature subsets on the GPU and estimate F-matrix on the CPU. The guided-search steps are made parallel by leveraging point/group independence as follows.

Step 1. Grid computation and feature binning We launch a grid of threads on the GPU with one thread processing one feature each. Each thread computes the cell centers and the cell indices for all features in $\mathbb{T}$ in parallel. A fast parallel sort using cell indices as keys brings all features belonging to the same cell together. Using a fast parallel prefix scan yields the starting index into each cell of the sorted array.

Step 2. Epipolar line based feature clustering In this step also a grid of threads is launched where each thread computes the epipolar line and its intersection with the image boundaries for each feature in $\mathbb{S}$. To cluster the features in $\mathbb{S}$ based on their epipolar lines, we perform a parallel sort of all the features using the coordinates of the line-image intersection points as the keys. To assign sorted feature points to clusters, we again launch a grid of threads where each thread writes 0 or 1 to a stencil array if its epipolar line differs from the previous one by more than 2 pixels. We then perform a fast parallel scan on this stencil array to provide us with the individual cluster indices.

Step 3. Finding the set of candidate matches We use one CUDA block per cluster of features in $\mathbb{S}$ to find its corresponding set of candidate matches $\mathbf{C}'$ in the target image $\mathbb{T}$. Each thread in the block takes one equidistant point on the epipolar line, computes its corresponding cell indices and retrieves the features binned into these cells in step 1. The candidate matches corresponding to each cluster of query features are stored in the GPU global memory.

Step 4. Finding the true match from candidates For every query point, we need to find its two nearest points in the respective candidates set derived in step 3. Each query point is handled by one CUDA thread block and each thread within the block computes the L2 distance between the query feature and one feature from the candidate set in parallel. A parallel block-wise minimum and next-minimum is computed and followed by a ratio test to declare a match.

6. Experiments and Results

We evaluate our algorithm by: *(i)* matching four images of different scene types shown in Figure 3 and manually verifying the matches; *(ii)* matching features for 3D reconstruction of three SFM datasets. For 3D reconstruction, we use Bundler [15] code and publicly available datasets: *(i)* Tsinghua School[1] (193 images), *(ii)* Barcelona Museum[2] (191 images), *(iii)* Notre Dame[3] subset (99 of 715 images). We cannot evaluate our algorithm on the popular Oxford dataset, as it only consists of different viewpoints of a planar (wall) scene where the F-matrix would degenerate.

[1] http://vision.ia.ac.cn/data/index.html

[2] http://www.inf.ethz.ch/personal/acohen/papers/datasets/barcelona.zip

[3] http://phototour.cs.washington.edu/datasets/

(a) Monument [$|\mathbb{S}|$ = 24K, $|\mathbb{T}|$ = 30K] (b) Indoor [$|\mathbb{S}|$ = 4K, $|\mathbb{T}|$ = 3K] (c) Plaza [$|\mathbb{S}|$ = 8K, $|\mathbb{T}|$ = 7K] (d) Desk [$|\mathbb{S}|$ = 16K, $|\mathbb{T}|$ = 5K]

Figure 3: Image pairs of different scene types with hand-verified feature matches

	Num. Features		Kd-tree		CasHash		Our		Kd-tree	CasHash	Our-CPU	Our-GPU	SIFTGPU
Image Pair	$\lvert\mathbb{S}\rvert$	$\lvert\mathbb{T}\rvert$	#match	#TP	#match	#TP	#match	#TP	sec.	sec.	sec.	sec.	sec.
Monument	24K	30K	189	184	386	376	**1099**	**1083**	2.87	0.81	0.91	0.046	–
Indoor	4K	3K	366	359	444	439	445	440	1.39	0.10	0.14	0.009	0.095
Plaza	8K	7K	234	229	314	306	469	458	0.69	0.13	0.19	0.011	0.127
Desk	16K	5K	120	114	114	109	**537**	**468**	1.59	0.21	0.29	0.018	0.148

Table 1: Number of matches (#match), number of correct matches (#TP) and matching time for image pairs shown in Figure 3.

			Kd-tree		CasHash		Our		Kd-tree	CasHash	Our (CPU)
Dataset	#Images	#Feat. (avg)	#PTS	#PTS3+	#PTS	#PTS3+	#PTS	#PTS3+	sec.	sec.	sec.
Notre Dame Paris	99	21K	85K	46K	82K	43K	**109K**	**65K**	6504	1408	3702
Tsinghua School	193	26K	178K	112K	180K	111K	**204K**	**132K**	27511	8660	8965
Barcelona Museum	191	18K	39K	12K	40K	11K	**179K**	**77K**	18282	3662	5120

Table 2: Comparison of the number of 3D points in final reconstruction and run-time for match-graph construction by various methods for the three SFM datasets. #PTS shows the number of total 3D points and #PTS3+ shows the number of points with feature track length 3 or higher. Our algorithm produces significantly denser point clouds.

In Table 1, we compare the performance of our algorithm with Kd-tree based matching [3, 12] and cascade hashing based matching [8] for the image pairs shown in Figure 3. We compare the number of matches, both total (#match) and true positives (#TP), as well as the run-time for all three methods. The reported timings for pairwise matching also include the time for Kd-tree construction, hash construction and initial geometry estimation for the respective methods. We also compare the time taken by the GPU implementation of our algorithm with global matching on GPU using the publicly available SIFT GPU library [1].

For the global Kd-tree based matching, we limit the maximum number of points visited to 400 (common practice in SFM) for fair comparison of timing. For CascadeHash, we use 8-bit hash codes and 6 bucket groups, provided as default in the authors' code. All CPU experiments are run on a single Intel 2.5GHz core with 4GB memory and GPU experiments are run on a machine with Intel core-i7 (2.67GHz) CPU and Nvidia K40 GPU.

It can be seen that our geometry-aware method retrieves more matches for image pairs of all scene categories. The limitation of global ratio test based matching methods for scenes with repetitive structures is strongly reflected in the number of matches for the monument image pair. Figure 4 shows a side by side, visual comparison of geometry-aware matches vs. global Kd-tree based matches for monument and desk image pairs. Figure 5 depicts the effectiveness of geometry-aware matching for repeating features on central rose window of the Notre Dame Cathedral (the monument image). The runtime of our algorithm for pairwise matching is comparable to that of CascadeHash and it is significantly better than global Kd-tree based matching. The GPU implementation of our algorithm also clearly outperforms global matching on the GPU (SIFT GPU).

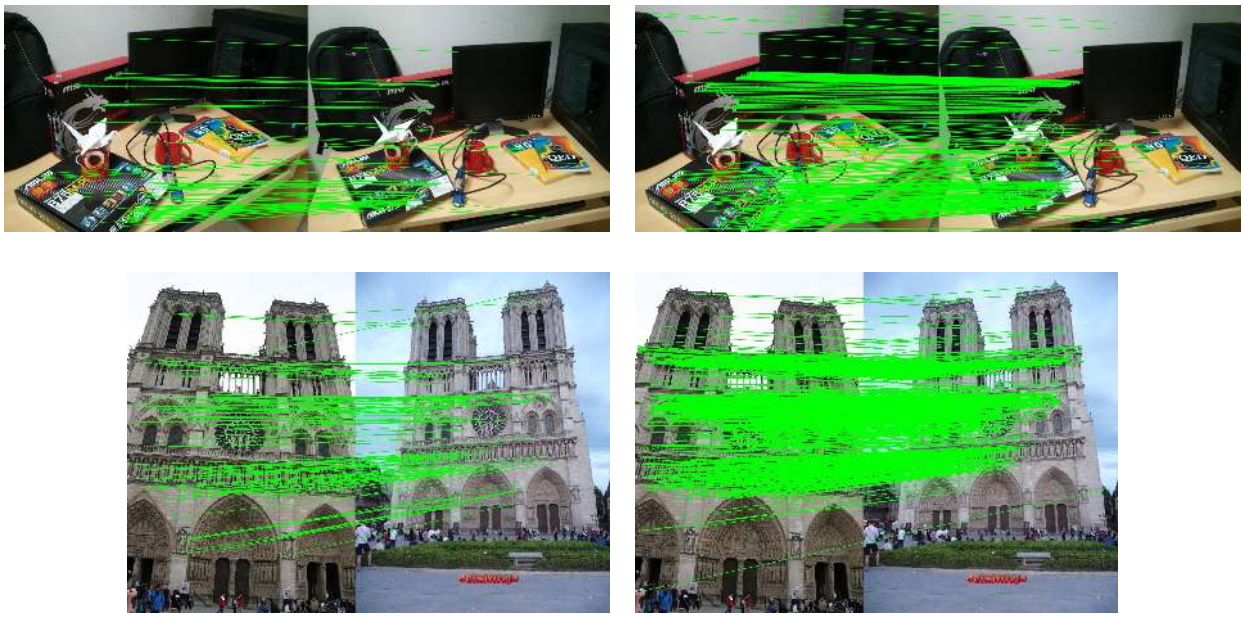


Figure 4: Feature matches for the desk image pair (top) and the monument image pair (bottom) using the global Kd-tree based method (left) and the geometry-aware method (right). A significantly higher number of good matches are retained by the geometry-aware approach for both pairs. Only bi-directional matches are shown for the sake of clarity.

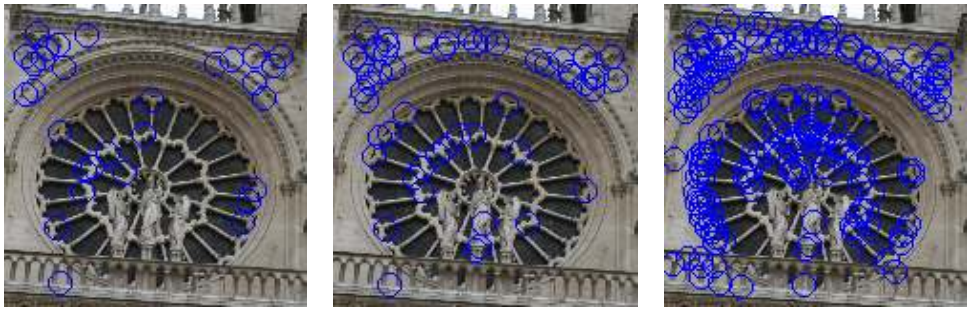
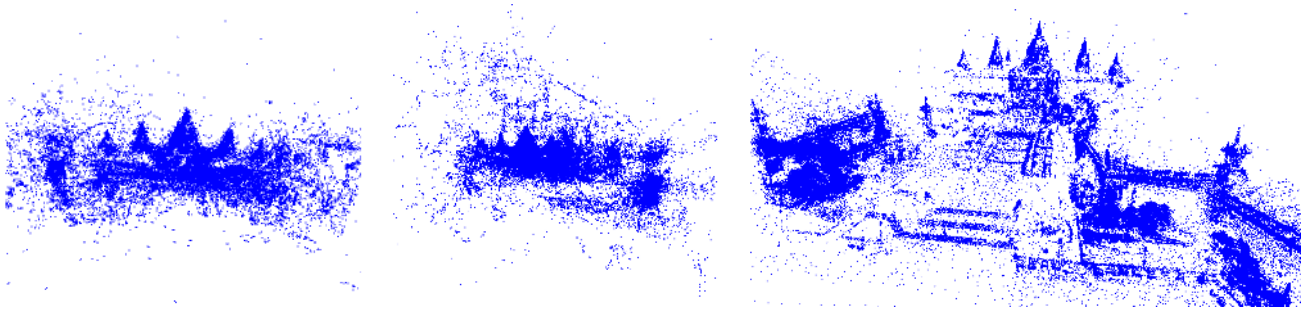


Figure 5: Central rose window of the Notre Dame Cathedral with repetitive structures. Successfully matched features are highlighted by blue circles for global Kd-tree based method (left), CascadeHash (center), and geometry-aware method (right).



(a) Kd-tree based Matching (b) Cascade hashing based Matching (c) Geometry-aware Matching

Figure 6: Point clouds for Barcelona Museum Reconstruction using match-graphs of different methods. While global methods recovers only partial structure, geometry-aware method (ours) performs significantly better.

Table 2 compares the match-graph construction time for 3D reconstructions of different datasets using the three feature matching methods. It also shows the number of 3D points (#PTS) in the final reconstruction for each matching method. In the absence of ground truth, we compare the number of reconstructed 3D points with feature tracks of length 3 or more (#PTS3+) as they are considered more reliable. Our method clearly outperforms other methods by producing significantly denser point clouds overall and also when only the points with track length $>= 3$ are considered. Reconstruction of Barcelona Museum dataset using the match-graphs of Kd-tree matching and CascadeHash could register only 119 and 136 images respectively, whereas reconstruction using our matching could register 181 out of 191 total images. Consequently, the recovered structure using our match-graph is also more complete as compared to other match-graphs as shown in Figure 6.

The run-time for match-graph computation using our method is significantly less than the Kd-tree based method and only slightly worse than CascadeHash. However, since we are able to find many times more correspondences, time per matched point for our method is much less compared to CascadeHash. Our GPU algorithm can further accelerate the match-graph computation significantly; at $\sim$ 20ms per image it can compute the match graph for a 100 image dataset only in a couple of minutes.

Limitations: Our geometry-aware matching depends on global methods for initial geometry estimation. For scenes with severe repetition such as urban skyscrapers, global matching can fail to find any reliable matches. As a result, our method would not be able to find reliable F-matrix to search further matches. Our method is based on the assumptions of a rigid scene and epipolar relations between the images. For scenes with large non-rigid dynamic objects (such as humans, faces), our matching would not produce meaningful correspondences.

7. Conclusions and Future Work

We proposed a simple and generalized geometry-aware algorithm for fast and efficient feature matching of static scenes and demonstrated its application for SFM reconstruction. We showed that the geometry-guided search also works well for images with repetitive structures and as a direct result, we get significantly denser feature correspondences and point clouds for architectural image datasets. We also showed that the proposed method is easy to parallelize and outlined a GPU algorithm for the same. In future, we would like to improve our method to handle severe repetition and also extend it for planar scenes.

Acknowledgement We thank the IDH project of DST and Google India PhD fellowship for financial support.

References

[1] `http://cs.unc.edu/~ccwu/siftgpu/`.

[2] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *Commun. ACM*, 54(10), 2011.

[3] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6), 1998.

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

[5] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day. In *Proceedings ECCV*, 2010.

[6] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[7] J. Hoberock and N. Bell. Thrust: A parallel template library, 2010. Version 1.7.0.

[8] C. Jian, L. Cong, W. Jiaxiang, C. Hainan, and L. Hanqing. Fast and accurate image matching with cascade hashing for 3d reconstruction. In *Proceedings IEEE CVPR*, 2014.

[9] N. Jiang, P. Tan, and L.-F. Cheong. Seeing double without confusion: Structure-from-motion in highly ambiguous scenes. In *Proceedings IEEE CVPR*, 2012.

[10] M. Kushnir and I. Shimshoni. Epipolar geometry estimation for urban scenes with repetitive structures. In *Proceedings ACCV*, 2012.

[11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2), 2004.

[12] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36, 2014.

[13] R. Roberts, S. N. Sinha, R. Szeliski, and D. Steedly. Structure from motion for scenes with large duplicate structures. In *Proceedings IEEE CVPR*, 2011.

[14] R. Shah, A. Deshpande, and P. J. Narayanan. Multistage sfm : Revisiting incremental structure from motion. In *3DV-Conference*, 2014.

[15] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Trans. Graph.*, 25(3), 2006.

[16] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua. LDAHash: Improved Matching with Smaller Descriptors. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 34(1), 2012.

[17] F. Sur, N. Noury, and M.-O. Berger. Image point correspondences and repeated patterns. Rapport de recherche RR-7693, INRIA, 2011.

[18] K. Wilson and N. Snavely. Network principles for sfm: disambiguating repeated structures with local context. In *Proceedings of ICCV*, 2013.

[19] C. Wu. Towards linear-time incremental structure from motion. In *3DV-Conference*, 2013.

[20] W. Zhang and J. Kosecka. Generalized ransac framework for relaxed correspondence problems. In *Proceedings 3DPVT*, 2006.