Scalable object instance recognition based on keygraph matching

Estephan Dazzi\textsuperscript{a,**}, Teofilo de Campos\textsuperscript{b}, Adrian Hilton\textsuperscript{c}, Roberto M. Cesar - Jr.\textsuperscript{a}

\textsuperscript{a}Instituto de Matemática e Estatística - Universidade de São Paulo, São Paulo, Brasil.
\textsuperscript{b}Universidade de Brasília - DF, Brasil.
\textsuperscript{c}Center for Vision, Speech and Signal Processing - University of Surrey, Guildford, UK.

ABSTRACT

We propose a generalisation of the local feature matching framework, where keypoints are replaced by $k$-keygraphs, i.e. isomorphic directed attributed graphs of cardinality $k$ whose vertices are keypoints. Keygraphs have structural and topological properties which are discriminative and efficient to compute, based on graph edge length and orientation as well as vertex scale and orientation. Keypoint matching is performed based on descriptor similarity. Next, 2-keygraphs are calculated; as a result, the number of incorrect keypoint matches reduced in 75% (while the correct keypoint matches were preserved). Then, 3-keygraphs are calculated, followed by 4-keygraphs; this yielded a significant reduction of 99% in the number of remaining incorrect keypoint matches. The stage that finds 2-keygraphs has a computational cost equal to a small fraction of the cost of the keypoint matching stage, while the stages that find 3-keygraphs or 4-keygraphs have a negligible cost. In the final stage, RANSAC finds object poses represented as affine transformations mapping images. Our experiments concern large-scale object instance recognition subject to occlusion, background clutter and appearance changes. By using 4-keygraphs, RANSAC needed 1% of the iterations in comparison with 2-keygraphs or simple keypoints. As a result, using 4-keygraphs provided a better efficiency as well as allowed a larger number of initial keypoint matches to be established, which increased performance.

© 2018 Elsevier Ltd. All rights reserved.

1. Introduction

Many problems in computer vision involve finding correspondences of robust invariant local features (i.e., keypoints), for both 2D images and 3D point clouds generated using depth images. For example, object instance recognition applied to visual search, augmented reality or object manipulation by robots. In such contexts, a query image is matched against a (possibly large) set of model images by individually matching each query feature against the set of model features. Such recognition strategy based on local feature matching provides three main advantages. First, by employing indexing techniques, it is possible to efficiently compare the query features against the model features. Second, local feature matching is effective against problems caused by occlusions and background clutter, leading to a good performance in “real-world” object detection. Third, it is possible to obtain geometrically precise detections, since fine object structures are matched.

The extraction of local features involves two stages: detection and description. In the detection stage, keypoints presenting rich local information are identified. In the description stage, the method assigns to each keypoint local shape information (e.g., scale and orientation) as well as a descriptor representing local visual content. After keypoint extraction, the next stage finds correspondences between keypoints representing the same parts in different images subject to large changes in viewpoint, scale and appearance. Traditionally, correspondences are established between individual keypoints based on descriptor similarity only, e.g. the original SIFT approach of Lowe (2004) or the method of Hsiao et al. (2010). Other authors employed spatial information in keypoint neighbourhoods in order to improve the overall quality of keypoint matching. For instance, the approach of Li et al. (2015) which matches keypoint pairs or the SCRAMSAC method of Sattler et al. (2009) that examines consistency in keypoint neighbourhoods.

We propose a generalisation of the traditional keypoint...
framework, by replacing keypoints for keygraphs, i.e. isomorphic directed attributed graphs whose vertices are keypoints, in order to explore structural and topological properties. A keygraph constitutes a semi-local descriptor that maintains robustness to geometric deformations provided by its vertex (keypoint) features. Each image is represented by a set of keygraphs and correspondences are established between keygraphs. We refer to a keygraph with $k$ vertices as a “$k$-keygraph”. In this paper, we consider 2-keygraphs, 3-keygraphs and 4-keygraphs.

Figure 1 illustrates the keygraph matching pipeline. Correspondences are established between query image keypoints and model image keypoints based on descriptor similarity; this yields a large number of both correct and incorrect keypoint matches. Next, matches of keygraphs of cardinality $k$ are calculated based on keygraphs of cardinality $k - 1$. The computational cost of the stage that obtains 2-keygraphs can be limited to a small fraction (say, 20%) of the cost of the keypoint matching stage, while the stages that obtain 3-keygraphs or 4-keygraphs have a negligible cost. As a result of the keygraph matching phase, the vast majority of incorrect initial keypoint matches are filtered out while the correct ones are preserved. Finally, RANSAC is applied to find object poses represented as affine transformations mapping images. By using matches of 4-keygraph vertices, RANSAC requires very few iterations, since a large fraction of correct matches is available.

This paper presents two fundamental contributions. First, we propose structural and topological properties of keygraphs with two, three or four vertices. Keygraph properties involve graph edge length and orientation as well as vertex scale and orientation, being fundamental to the success of the proposed approach. Second, we introduce an efficient method to calculate keygraph matches. The proposed method filters out the vast majority of incorrect keypoint matches, which provides two main advantages: (1) efficiency, since RANSAC needs to perform few iterations and (2) increased recognition performance, as the strategy allows a large number of keypoints matches to be established in the initial stage which results in a large number of correct keypoint matches.

Our experiments consider an object instance recognition problem in which a query image is matched against a large dataset of model objects. Query images are subject to realistic occlusions and illumination, viewpoint transformations. In this paper, we use keygraphs whose vertices are SIFT keypoints, since SIFT is widely known and has a good matching performance, as shown by Lowe (2004). As experimentally evaluated by Lowe (2004), SIFT features are robust against problems caused by occlusions, background clutter and illumination changes. And, since transforming keypoint matches into keygraph matches does not eliminate correct keypoint matches, the proposed method also presents robustness against occlusion, background clutter and illumination changes. The model dataset is composed of $10^5$ images, which generated a large set of $10^9$ SIFT keypoints.

Experiments showed that obtaining 2-keygraphs filtered out 75% of the incorrect initial keypoint matches (while correct keypoint matches were not eliminated); next, obtaining 3-keygraphs reduced in 99% the number of incorrect remaining keypoint matches; then, obtaining 4-keygraphs filtered out a moderate fraction of remaining incorrect matches. As a result, RANSAC required very few iterations. In contrast, if simpler 2-keygraphs were used, RANSAC needed to perform two orders of magnitude more iterations, leading to a total computational at least 25% larger than the cost of the method based on 4-keygraphs. An even worse result was obtained by using simple “1-keygraphs” (i.e., the initial keypoint matches). We also present results for the SCRAMSAC method of Sattler et al. (2009). SCRAMSAC has a similar computational cost as the proposed method while being based on a different use of spatial properties – namely, consistency in keypoint neighbourhoods. As shown by experiments, the proposed method achieved superior results than SCRAMSAC.

The remainder of this paper is organised as follows. Section 2 discusses related works in the literature. Section 3 presents mathematical definitions regarding keygraph matching. Section 4 describes an efficient implementation of the concepts presented in Section 3. Section 5 presents experimental evalu-
2. Related work

An advantage gained from using local features is a natural geometric precision in detection, since fine object structures are matched. This property is fundamental in tasks such as Structure-from-Motion, wide-baseline stereo, augmented reality or object manipulation by robots, as discussed by Loncomilla (2016). In the present paper, we consider the problem of large-scale object instance recognition. Since our goal is introducing the keygraphs method, we employ a relatively simple, hand-crafted descriptor (SIFT), instead of using CNN-based descriptors that are tailored for image matching applications, such as the descriptors proposed by Dong and Soatto (2015) and Han et al. (2015). Moo Yi et al. (2016) employ a CNN in order to assign a canonical orientation to a keypoint; in contrast, SIFT uses the dominant orientation. Buoncompagni et al. (2015) propose an efficient method to rank and select keypoints based on their saliency.

In order to achieve a better performance, local features conveying complementary information should be employed. Tombari et al. (2013a) proposed a keypoint detector and descriptor that operates in textureless regions. The result of the keypoint detection stage is a sparse keypoint set extracted from visually rich regions, which yields accurate keypoint matches. A different strategy relies on extracting a dense keypoint set, e.g., one keypoint from each pixel. Choy et al. (2016) employ CNN’s in order to learn a feature space where dense local descriptors are compared.

Keypoints can be extracted from 3D point clouds created using depth images generated by sensors such as LIDAR. 3D keypoint detectors and descriptors are evaluated by Tombari et al. (2013b) and Guo et al. (2016), respectively. The Fast Point Feature Histograms (FPFH) proposed by Rusu et al. (2009) calculates a 3D keypoint descriptor as a histogram of angles between surface normals, measured in the surrounding keypoint neighbourhood. Kim and Hilton (2014) improve the performance of FPFH by separately employing neighbourhoods of different sizes and then combining the matching outputs. Kim et al. (2016) propose a framework for registration of visual data acquired from various 2D and 3D sensing modalities.

After the keypoint matching phase, a subset of matches agreeing on an object pose instantiation is found; in this context, the Random Sample Consensus (RANSAC) method proposed by Fischler and Bolles (1981) is a standard solution. However, if a large fraction of incorrect matches is used, RANSAC needs to perform a large number of iterations, leading to a prohibitive computational cost. In order to filter out incorrect matches aiming to enable the use of RANSAC, in the approach of Pang et al. (2014), a candidate keypoint match \((p, q)\) is filtered out in case the local geometric structure of \(p\) is different from the one of \(q\); such local geometric structure is calculated using an optimization process that reconstructs the keypoint from its three neighbours in the image. The approach of da Camara Neto and Campos (2010) obtains a coarse global registration between a pair of images, which constrains the keypoint correspondence space; however, in case the fraction of correct keypoint matches is low, the estimation of coarse global registration is likely to fail.

Sattler et al. (2009) proposed the Spatial Consensus RANSAC (SCRAMSAC) method. For each tentative keypoint match \((p, q)\), a minimum fraction of the keypoints in a neighbourhood of \(p\) is required to match keypoints in a neighbourhood of \(q\) otherwise the match is filtered out. Such an approach presents two main drawbacks in comparison to ours. First, a large keypoint neighbourhood is employed in order to decide whether a candidate keypoint match is valid, thus making the approach susceptible to the fraction of correct keypoint matches. In contrast, the proposed method considers small keypoint neighbourhoods (up to four keypoints). Second, SCRAMSAC employs limited spatial information (keypoint distance only), while the proposed method considers orientation, scale and position. The computational cost of SCRAMSAC is similar to the cost of the proposed method: both methods are quadratic in the number of keypoint matches.

Previous work investigated the idea of finding correspondences of small keypoint sets instead of correspondences of individual keypoints. In particular, using matches of keypoint pairs demonstrated good results. The method of Carneiro and Jepson (2004) checks for consistency in changes in keypoint scale and orientation as well as changes in length and orientation of a vector connecting each keypoint pair. Li et al. (2015) additionally employ a Hough transform in order to filter out a number of keypoint matches. Hao et al. (2013) detect 3D object models (created using Structure-from-Motion) in 2D query images; a candidate match of 2D-3D keypoint pairs is evaluated by back-projecting the 2D positions into the camera coordinates and then checking whether both 3D distances are similar. Avrithis and Tolias (2014) propose a method with a linear cost in the number of matches that explores simple pairwise relations. Zhang et al. (2011) employ the co-occurrence statistics of visual words within some local image regions.

Instead of using keypoint pairs, previous work used matches of keypoint triples or quadruples, which allows exploring richer structural information. Zitnick et al. (2007) extract all keypoint triples from query and model images, which are then mapped to a canonical space where keypoint descriptors are calculated; then, keypoint triples are matched based on descriptor similarity. Kalantidis et al. (2011) use Delaunay triangulations in order to select a subset of keypoint triples from both query and model images. Hao et al. (2012) detect 3D object models in 2D images; Delaunay triangulations generate keypoint triples from the models, and a valid match presents consistent changes in keypoint scales and distances of 2D projections of 3D points. Hinterstoisser et al. (2007) extract keypoint quadruples and quintuples from the model images, which can be used to instantiate a 3D object model onto a 2D query image. Hashimoto and Cesar Jr (2009) introduced the concept of keygraphs: all possible keygraphs with three vertices are extracted from a model image, and Fourier coefficients of keygraph edges are used as local descriptors; then, during matching time, a Delaunay triangulation generates keypoint triples from a query image.

A drawback of the discussed methods based on matches of keypoint triples or quadruples is relying on storing pre-
calculated structures of model keypoints in working memory, which presents a scalability issue. Our previous approach (Dazzi et al. (2016)) avoids that scalability problem: Delaunay triangulations are employed in order to select several keypoint triples in the query image. Next, each query keypoint triple (i.e. 3-keygraph) is matched against a model image in case all its three constituent keypoint matches exist and the candidate 3-keygraph match satisfies the keygraph properties. Unfortunately, it is possible that a set of three correct keypoint matches do not form a 3-keygraph match, which would occur in case those three keypoint matches are not selected by a Delaunay triangulation for composing the same keypoint triple in the query image. In the present paper, we introduce an efficient strategy that matches all possible keypoint triples or quadruples during the matching phase, thus preserving correct keypoint matches.

A different graph-based approach for image matching models an image as a global graph whose vertices are keypoints. As shown by Zhang et al. (2016), graph matching techniques can be used to find similar images. McAuley et al. (2010) consider that graph similarity encompasses three aspects: keypoint descriptors, distances of keypoint pairs and inner angles of keypoint triples. Park et al. (2014) also propose a method that is based on similarity of inner angles of keypoint triples. In contrast, our method relies on efficient comparisons of changes occurring in individual graph edges and/or vertices.

3. Definitions

Table 1 summarises the symbols and important concepts adopted in this paper.

3.1. Keypoints

A keypoint is a locally distinct feature. Each keypoint \( p \) is detected using SIFT and assigned a scale \( \sigma_p \), an orientation \( \theta_p \), a position \( x_p = (x_p, y_p) \) and a descriptor. Let \( \mathbb{P}_T \) be the set of keypoints extracted from an image \( I \); each keypoint \( p \in \mathbb{P}_T \) is assigned an unique, random integer label \( L(p) \) ranging from 1 to \( |\mathbb{P}_T| \).

A keypoint match is a pair \( i = (p, q) \), where keypoint \( p \) is in a query image \( I_Q \) and keypoint \( q \) is in a model image \( I_M \). A set \( \mathcal{M}_T \) has the initial keypoint matches between a pair of images.

Each keypoint match \((p_i, q_i)\) is associated to a change in keypoint scale, 

\[
\Delta\sigma_i = \frac{\sigma_{q_i}}{\sigma_{p_i}},
\]

and a change in keypoint orientation, measured as a (signed) difference between a pair of 2D orientations (i.e., a 2D angle):

\[
\Delta\theta_i = \theta_{q_i} - \theta_{p_i} .
\]

3.2. Keygraphs

A keygraph \( G = (V_G, E_G) \) is a directed attributed graph whose vertices are keypoints, with vertex set \( V_G \) (composed of keypoints in the same image) and graph edge set \( E_G \).

A keygraph match is a triple \( \mu = (G, H, f) \), where \( G = (V_G, E_G) \) is a keygraph in a query image and \( H = (V_H, E_H) \) is a keygraph in a model image (with \( G \) and \( H \) being isomorphic), and \( f : V_G \to V_H \) is a bijection mapping \( V_G \) and \( V_H \).

### Table 1. Summary of symbols and concepts adopted in the paper.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_Q, I_M )</td>
<td>Query image ( I_Q ) and model image ( I_M ).</td>
</tr>
<tr>
<td>( p, \sigma_p, \theta_p, x_p )</td>
<td>Keypoint ( p ) with scale ( \sigma_p ), orientation ( \theta_p ), and 2D position ( x_p ).</td>
</tr>
<tr>
<td>( L(p) )</td>
<td>Keypoint label in an image.</td>
</tr>
<tr>
<td>( \mathbb{F}_T )</td>
<td>Set of keypoints extracted from an image.</td>
</tr>
<tr>
<td>( E, E_q, \Theta_E )</td>
<td>Keygraph edge ( E ) with length ( E_{\ell} ) and orientation ( \Theta_E ).</td>
</tr>
<tr>
<td>( \ell_{\min}, \ell_{\max} )</td>
<td>Minimum ( \ell_{\min} ) and maximum ( \ell_{\max} ) allowed edge length in a query image.</td>
</tr>
<tr>
<td>( E_q )</td>
<td>Set of keygraph edges in a query image ( I_Q ).</td>
</tr>
<tr>
<td>( G = (V_G, E_G) )</td>
<td>Keygraph ( G ) with vertex set ( V_G ) and edge set ( E_G ).</td>
</tr>
<tr>
<td>( G_{2v}, G_{3v}, G_{4v} )</td>
<td>2-keygraph, 3-keygraph and 4-keygraph.</td>
</tr>
<tr>
<td>( i = (p, q) )</td>
<td>Keypoint match ( i ) between keypoints ( p ) and ( q ).</td>
</tr>
<tr>
<td>( \mu = (G, H, f) )</td>
<td>Match of keygraphs ( G ) and ( H ) with bijection mapping vertices ( f ).</td>
</tr>
<tr>
<td>( \Delta\sigma, \Delta )</td>
<td>Change in keypoint scale ( \Delta\sigma ) and change in keygraph edge length ( \Delta ).</td>
</tr>
<tr>
<td>( \nabla_{\sigma}(\cdot) )</td>
<td>Dissimilarity of changes in keypoint scale and/or edge length.</td>
</tr>
<tr>
<td>( \Delta\theta, \Delta\Theta )</td>
<td>Change in keypoint orientation ( \Delta\theta ) and change in keygraph edge orientation ( \Delta\Theta ).</td>
</tr>
<tr>
<td>( \nabla_{\theta}(\cdot) )</td>
<td>Dissimilarity of changes in keypoint orientation and/or edge orientation.</td>
</tr>
<tr>
<td>( \mathcal{M}_T )</td>
<td>Set of initial keypoint matches between a pair of images.</td>
</tr>
<tr>
<td>( \mathcal{M}<em>{2v}, \mathcal{M}</em>{3v}, \mathcal{M}_{4v} )</td>
<td>Set of matches of 2-keygraphs ( \mathcal{M}<em>{2v} ), 3-keygraphs ( \mathcal{M}</em>{3v} ), and 4-keygraphs ( \mathcal{M}_{4v} ).</td>
</tr>
<tr>
<td>( \nabla_{\theta} )</td>
<td>Set of matches of vertices of 4-keygraphs.</td>
</tr>
</tbody>
</table>

3.2.1. Keygraph edges

A keygraph edge is defined as an ordered pair \( E_{ij} = (p_i, p_j) \), where \( p_i \) and \( p_j \) are keypoints in the same image. In case \( E_{ij} \) is an edge in a query image, the keypoint labels are such that \( \ell_{E_{ij}} < \ell_{E_{pq}} \); that is, we arbitrarily determine an edge’s direction to be such that the edge leaves the keypoint with the smaller label and enters the keypoint with the larger label (i.e. from \( p_1 \) to \( p_2 \)). Determining edge direction is a necessary step in order to assign edge orientation, which is obtained as the regular angle of a 2D vector with the horizontal axis (as illustrated in Figure 3-c). In a 2D image, a vector \( \mathbf{v}_{ij} = x_{p_i} - x_{p_j} \) is associated to \( E_{ij} \); the length \( \ell_{E_{ij}} = |\mathbf{v}_{ij}| \) and the orientation \( \Theta_{E_{ij}} \) of \( \mathbf{v}_{ij} \) are assigned to edge \( E_{ij} \).

A match between edges \( E_{ij} = (p_i, p_j) \) and \( F_{ij} = (q_i, q_j) \) with a bijection \( f = [(p_i, q_i), (p_j, q_j)] \) has a change in edge length:

\[
\Delta\ell_{ij} = \frac{\ell_{E_{ij}}}{\ell_{F_{ij}}} ,
\]

and a change in edge orientation, measured as a (signed) difference between a pair of 2D orientations (i.e., a 2D angle):

\[
\Delta\Theta_{ij} = \Theta_{F_{ij}} - \Theta_{E_{ij}} .
\]
a keygraph edge with length \( l \) in a model image is mapped to a query image in which a change in scale \( s \) has occurred, its length becomes \( sl \). Thus, the minimum edge length in a query image \( l_{\text{min}} \) is related to how much the area of an object in a query image can be reduced in comparison to the area of that object in a model image. As for the maximum edge length in a query image \( l_{\text{max}} \), it should be set sufficiently large to occupy the image of the considered object in a query image but not unnecessarily large to generate unnecessary computational cost.

### 3.2.2. Dissimilarity between changes in keygraph attributes

A keygraph match is associated to changes in keypoint scale \( \Delta r \) (Equation 1) and changes in edge length \( \Delta l \) (Equation 3). Let \( \Delta \phi, \Delta \phi' \) be a pair of changes in keypoint scale and/or edge length; a dissimilarity \( \psi(\Delta \phi, \Delta \phi') \) between them is measured as a ratio. We define a pair of changes \( \Delta \phi, \Delta \phi' \) to be similar if the largest one is at most twice the smaller one:\(^1\)

\[
0.5 \leq \frac{\Delta \phi}{\Delta \phi'} \leq 2.0.
\]

(5)

A keygraph match is also associated to changes in keypoint orientation \( \Delta \theta_1 \) (Equation 2) and changes in edge orientation \( \Delta \Theta \) (Equation 4). Let \( \Delta \alpha, \Delta \alpha' \) be a pair of changes in keypoint orientation and/or edge orientation; a dissimilarity \( \varphi(\Delta \alpha, \Delta \alpha') \) between them is measured as an angle. We define a pair of changes \( \Delta \alpha, \Delta \alpha' \) to be similar if the absolute value of the smaller angle between them is at most 60\(^\circ\):\(^2\)

\[
\arccos(\cos(\Delta \alpha - \Delta \alpha')) \leq 60^\circ.
\]

(6)

### 3.2.3. 2-keygraphs

A 2-keygraph \( G_{2v} = (V_{2v}, E_{2v}) \) has two vertices and one edge. Given a set \( E_Q \) of keygraph edges in a query image \( I_Q \) (Section 3.2.1), a set \( M_{2v} \) of 2-keygraph matches between \( I_Q \) and a model image \( I_M \) is obtained by matching each edge \( E \in E_Q \) against \( I_M \). Let triple \( \mu_{2v} = (G_{2v}, H_{2v}, f_{2v}) \) represent a candidate 2-keygraph match, where the query image keygraph \( G_{2v} \) has edge \( E_{12} = (p_1, p_2) \) and the bijection mapping vertices is \( f_{2v} = \{(p_1, q_1), (p_2, q_2)\} \) (with the keypoint matches in \( f_{2v} \) being determined in the initial keypoint matching stage). This candidate 2-keygraph match is established if it presents similar changes in attributes: the changes in keypoint scale \( \Delta r_1, \Delta r_2 \) and the change in edge length \( \Delta l_{12} \) must be pairwise similar.

1 In an image subjected to a zoom of factor \( \kappa \), each keygraph edge length or SIFT scale changes in the same factor \( \kappa \). Sattler et al. (2009) showed that in an image subjected to a change in viewing angle of \( \psi \) degrees, a unit circle becomes an ellipse whose longer and shorter axes have length 1 and \( \cos\psi \), respectively. For successful matching, this change in viewing angle must be below 60\(^\circ\), since SIFT features lose reliability when \( \psi > 60^\circ \), as shown by Lowe (2004). When \( \psi = 60^\circ \), the length of the transformed ellipse’s shorter axis divided by the original circle’s diameter is \( \cos 60^\circ / 1 = 0.5 \). Based on preliminary experiments, we used this value for the parameter, thus making keygraph matching invariant to changes in viewing angle of at most 60\(^\circ\).

2 Rotating an image by \( \theta \) degrees changes the orientation of every keypoint and keygraph edge in \( \theta \) degrees. Under a moderate change in viewing angle, not every keypoint and edge rotates in the same \( \theta \) degrees, although very distinct changes in orientation cannot occur. We allow a maximum difference in rotations of 60\(^\circ\), according to preliminary experiments.

### 3.2.4. 3-keygraphs

A 3-keygraph \( G_{3v} = (V_{3v}, E_{3v}) \) is composed of three vertices and three edges. A set \( M_{3v} \) of 3-keygraph matches between a pair of images contains all combinations of three 2-keygraph matches yielding a valid 3-keygraph match. Let \( \mu_{3v} = (G_{3v}, H_{3v}, f_{3v}) \) represent a candidate 3-keygraph match, where the query image keygraph \( G_{3v} \) has edges \( V_{3v} = \{(p_1, p_2), (p_1, p_3), (p_2, p_3)\} \) and the bijection mapping vertices is \( f_{3v} = \{(p_1, q_1), (p_2, q_2), (p_3, q_3)\} \). In order for this candidate 3-keygraph match to be established, similar changes in keygraph attributes are required: the changes in keypoint scale \( \Delta r_1, \Delta r_2, \Delta r_3 \) and the changes in edge length \( \Delta l_{12}, \Delta l_{13}, \Delta l_{23} \) must be pairwise similar (Equation 5), which yields \( (\frac{\Delta \phi}{\Delta \phi'}) \leq 2.0 \). In order for this candidate 3-keygraph match to be established, similar changes in keygraph attributes are required: the changes in keypoint scale \( \Delta r_1, \Delta r_2, \Delta r_3 \) and the changes in edge length \( \Delta l_{12}, \Delta l_{13}, \Delta l_{23} \) must be pairwise similar (Equation 5), which yields \( (\frac{\Delta \phi}{\Delta \phi'}) \leq 2.0 \).

### 3.2.5. 4-keygraphs

A 4-keygraph \( G_{4v} = (V_{4v}, E_{4v}) \) has four vertices and five edges. A set \( M_{4v} \) of 4-keygraph matches between a pair of images is composed of all combinations of two 3-keygraphs sharing a 2-keygraph (Figure 2-Right). Both constituent 3-keygraph matches are consistent w.r.t. the changes in the shared edge and vertices. Obtaining 4-keygraphs eliminates disconnected 3-keygraphs as well as 3-keygraphs sharing one vertex only.

### 3.3. Final matches of keygraph vertices

Given a set \( M_{k} \) of 4-keygraph matches between a pair of images, the set \( N_{k} \) of 4-keygraph vertex matches is

\[
N_{4v} = \{(p, q) : (p, q) \in f_{4v} \text{ and } (G_{4v}, H_{4v}, f_{4v}) \in M_{4v}\}.
\]

(7)
4. Methodology and implementation

The proposed method is composed of a learning phase and a matching phase. During the learning phase, keypoints are extracted from all model images and then indexed in the descriptor space. During the matching phase, keypoints are extracted from a query image \( I_Q \); then, object matching between \( I_Q \) and the dataset of model images follows five stages: keypoint matching and then matching of 2-keygraphs, 3-keygraphs and 4-keygraphs, followed by RANSAC using matches of 4-keygraph vertices. This pipeline is illustrated in Figure 1-b.

4.1. First stage: keypoint matching

In order to index model keypoint descriptors, we use a modified version of the hierarchical \( K \)-means tree proposed by Muja and Lowe (2014). Given all model keypoints, \( K \)-means splits the descriptor space, recursively; a region with less than \( K \) descriptors then becomes a leaf node. When a query image keypoint \( p \) traverses a tree, in an intermediate tree level, \( p \)’s descriptor is assigned to the nearest cluster center. In a leaf node, the distance from \( p \)’s descriptor to the cluster center is calculated and represents the similarity between \( p \) and each one of the keypoints in this leaf; then, \( p \) restarts the traversal from the next most similar cluster mean (in an intermediate level). When \( p \) examines a total of \( L \) stored descriptors, the traversal stops. As a result, \( p \) establishes up to one keypoint match with each model image \( I_M \) (where \( n \) stands for the \( n \)-th model image in the dataset); during the traversal, if \( p \) is compared to more than one keypoint from the same model image, only the match with the highest similarity between descriptors is maintained. We also performed experiments employing at most two keypoint matches per model image, instead of one; however, using 4-keygraphs, this approach provided only a small gain in performance, thus we opted to use only one match since it allowed a more fair comparison of the keygraphs method with methods based on simple keypoints.

4.2. Second stage: matching of 2-keygraphs

Let \( M_{1v} \) be a set of initial keypoint matches between a query image \( I_Q \) and a model image \( I_M \). Each pair of keypoint matches in \( M_{1v} \) is checked for constituting a match of 2-keygraphs (Section 3.2.3). This generates a set \( M_{2v} \) of 2-keygraph matches between images \( I_Q \) and \( I_M \). The set \( M_{2v} \) is implemented as a (sparse) matrix of lists \( T_{2v} \), with one list \( L_{p,q} \) for each keypoint match \( (p,q) \) in the set of matches of vertices of 2-keygraphs \( \mathcal{N}_{2v} = \{(p,q) : (p,q) \in f_{2v} \land (G_{2v}, H_{2v}, f_{2v}) \in M_{2v} \} \). A list \( L_{p,q} \) contains all elements \((p’, q’)\) such that there is an established match of 2-keygraphs whose edge in the query image is \( (p, p’) \) and whose bijection mapping vertices is \((p, q), (p’, q’)\). Each list \( L_{p,q} \) stores its elements in sorted order, with an element \((p’, q’)\) being assigned a sort key value consisting of a pair \((L(p’), L(q’))\).

For each 2-keygraph match that is established (i.e., that satisfies the 2-keygraph properties shown in Section 3.2.3), the calculated changes in edge and vertices are stored in the matrix \( T_{2v} \) as well, in order to be employed in the next stage.

4.3. Third stage: matching of 3-keygraphs

Let \( \mu_{2v} = (G_{2v}, H_{2v}, f_{2v}) \) be a match of 2-keygraphs in \( M_{2v} \) with edge in the query image \( E_{12} = \{(p_1, p_2)\} \) and bijection mapping vertices \( f_{2v} = \{(p_1, q_1), (p_2, q_2)\} \). In a matrix of lists \( T_{2v} \), let lists \( L_{p_1,q_1} \) and \( L_{p_2,q_2} \) be associated to keypoint matches \((p_1, q_1)\) and \((p_2, q_2)\), respectively (Section 4.2). Since each list stores its elements in sorted order, finding elements which are common to both lists has a linear complexity in the total number of elements. Each element \((p_1, q_1)\) which is present in both lists \( L_{p_1,q_1} \) and \( L_{p_2,q_2} \) yields a candidate match of 3-keygraphs \( \mu_{3v} = (G_{3v}, H_{3v}, f_{3v}) \) with edges in the query image.

Fig. 3. Structural attributes of 3-keygraphs. (a) Edge length. (b) Vertex scale. (c) Edge orientation. (d) Vertex orientation.

Fig. 4. Consistency in changes of 3-keygraph attributes. (a) Consistency among a change in vertex scale (changing from \( \sigma_{p_1} \) to \( \sigma_{p_2} \)) and a change in edge length (from \( l_{p_1,p_3} \) to \( l_{p_2,p_3} \)); \( 0.5 \leq \frac{\sigma_{p_1}}{\sigma_{p_2}} < 2.0 \). (b) Consistency among a change in vertex orientation (from \( \theta_{p_1} \) to \( \theta_{p_2} \)) and a change in edge orientation (from \( \theta_{p_1,p_3} \) to \( \theta_{p_2,p_3} \)); \( \arccos((\theta_{p_1} - \theta_{p_2}) - (\theta_{p_1} - \theta_{p_2})) \leq 60^\circ \).
\[ \mathcal{E}_{G_{3v}} = \{(p_1, p_2), (p_1, p_3), (p_2, p_3)\} \] and bijection mapping vertices \( f_{3v} = \{(p_1, q_1), (p_2, q_2), (p_3, q_3)\} \). Then, the method checks whether this 3-keygraph match is valid (Section 3.2.4). Since edge changes \( \Delta\Theta_i \), \( \Delta_i \), and vertex changes \( \Delta\nu_i \), \( \Delta\sigma_i \) were previously calculated and stored, the current stage only needs to evaluate the pairwise dissimilarities (Equations 5 and 6). Also, since it is known that the changes involved in each individual 2-keygraph match are pairwise similar, they do not need to be re-evaluated; e.g., it is known that the changes in edge length and vertex scale \( \{\Delta\tau_1, \Delta\sigma_1, \Delta\sigma_2\} \) are pairwise similar.

Each 2-keygraph match in \( M_{3v} \) is checked for being involved in 3-keygraph matches, as illustrated in Figure 5.

A set \( M_{3v} \) of 3-keygraph matches between a pair of images is implemented as a (sparse) matrix of lists \( T_{3v} \). In \( T_{3v} \), list \( L_{p_1,q_1,p_2,q_2,q_3} \), which is associated to a pair of keypoint matches \( \{(p_1, q_1), (p_2, q_2), (p_3, q_3)\} \), contains all elements \( (p', q') \) such that there is a 3-keygraph match whose bijection mapping vertices is \( f = \{(p_1, q_1), (p_2, q_2), (p_3, q_3)\} \); thus, in the case the 3-keygraph match associated with \( f \) is established, three elements are inserted into matrix \( T_{3v} \): an element \( (p', q') \) inserted into list \( L_{p_1,q_1,p_2,q_2,q_3} \) inserted into \( L_{p_1,q_1,p_2,q_2,q_3}^{p',q'} \) and \( (p_1, q_1) \) inserted into \( L_{p_1,q_1,p_2,q_2,q_3}^{p',q'} \).

4.4. Fourth stage: matching of 4-keygraphs

Let \( \mu_{3v} = (G_{3v}, H_{3v}, f_{3v}) \) be a 3-keygraph match in \( M_{3v} \) with edges in the query image \( \mathcal{E}_{G_{3v}} = \{(p_1, p_2), (p_1, p_3), (p_2, p_3)\} \) and bijection mapping vertices \( f_{3v} = \{(p_1, q_1), (p_2, q_2), (p_3, q_3)\} \). The method verifies whether there are other 3-keygraph matches sharing an edge with \( \mu_{3v} \) by individually checking each constituent 2-keygraph match. That is, for the edge \( E_{12} = (p_1, p_2) \), the method checks in the matrix of lists \( T_{3v} \) whether list \( L_{p_1,q_1,p_2,q_2,q_3}^{p',q'} \) has any inserted element; for each keypoint match \( (p', q') \) in list \( L_{p_1,q_1,p_2,q_2,q_3}^{p',q'} \), if \( L(p') > L(p_2) \), then a match of 4-keygraphs \( \mu_{4v} = (G_{4v}, H_{4v}, f_{4v}) \) is established, where keygraph \( G_{4v} \) in the query image is the union of both 3-keygraphs, with edges \( \mathcal{E}_{G_{4v}} = \mathcal{E}_{G_{3v}} \cup \{(p_2, p'), (p_3, p')\} \) and bijection mapping vertices \( f_{4v} = f_{3v} \cup \{(p', q')\} \). Similarly, for the edge \( E_{13} = (p_1, p_3) \), list \( L_{p_1,q_1,p_2,q_2,q_3}^{p',q'} \) is verified, while for the edge \( E_{23} = (p_2, p_3) \), list \( L_{p_1,q_1,p_2,q_2,q_3}^{p',q'} \) is verified.
identical to a model image: all candidate 2-keygraph matches would be established, yielding $O(n \cdot d_s^2)$-keygraph matches, $O(n \cdot d_s^3)$-3-keygraph matches and $O(n \cdot d_s^4)$-4-keygraph matches, where $d_{vs} = O(n)$. Thus, an effective approach to control the combinatorial complexity limits the number $d_{vs}$ of edges leaving a vertex as well as the number $d_{vs}$ of 3-keygraphs sharing an edge i.e., makes $d_{vs} = O(1)$ and $d_{vs} = O(1)$. As a result, the cost of the stage that finds 2-keygraphs dominates the cost of the following stages.

5. Experiments and results

We consider an object instance recognition problem where images are subject to realistic viewpoint, scale and appearance changes, as well as occlusion and background clutter. We employed the CMU10 dataset made available by Hsiao et al. (2010). This dataset contains ten types of model objects, for a total of 250 model images with resolution $640 \times 480$ or $1600 \times 1200$ pixels. There are 500 query images with resolution $640 \times 480$ pixels. The dataset collected by Hsiao et al. (2010) considers a natural setting, consisting of common household objects in real, cluttered environments under different lighting conditions, occlusions and viewpoints (examples of images are presented in Figures 1 and 9). Ten different objects are considered: clam chowder can, diet coke can, juice box, orange juice carton, pot roast soup, rice pilaf box, rice tuscan box, soy milk can, soy milk carton and tomato soup can. Some objects present relatively few visual features (e.g. diet coke can and soy milk can) while other objects present a larger number of features (e.g. clam chowder can and rice pilaf box). We decided to average out the final accuracy measure over all the ten object classes in order to better focus on the main result of the present paper, namely, the gain in performance and efficiency provided by using $k$-keygraphs with $k = 3$ or $k = 4$ in comparison to using $k = 1$ or $k = 2$ (we experimentally observed that all the ten considered object categories presented such gains in performance and efficiency). In order to simulate a large-scale scenario, we also considered an additional set of $10^5$ “distractor” model images, obtained from the Oxford Buildings dataset introduced by Philbin et al. (2007). Their model keypoints are indexed together with the true model keypoints from the CMU10 dataset. The distractor images consider a very large range of different visual contexts, effectively simulating a “real-world” scenario in which many distinct model objects are considered. This yielded a total of $10^9$ SIFT keypoints which were indexed in a hierarchical $K$-means tree using $K = 16$. The VLFeat library (Vedalid and Fulkerson (2008)) was used for keypoint extraction.

For each detected object in a query image, we used the recovered affine transformation to project the model image’s ground-truth segmentation onto the query image, which yields a region $A$: a detection is correct if $(A \cap A_{gt})/(A \cup A_{gt}) > 0.4$, where $A_{gt}$ is the ground-truth in the query image. For each of the ten model objects, we plotted a precision/recall curve and then calculated the area underneath the curve; this is denoted as Average Precision (AP). The average AP over all ten AP values summarises the results; we denote this average value as “AP”.

Object matching can be divided into three stages: keypoint matching, keypoint match filtering and RANSAC which finds affine transformations. We evaluated five strategies for the matching process. (1) Not filtering out keypoint matches, i.e. RANSAC uses the initial keypoint matches; this is similar to a traditional approach that relies only on descriptor similarity and does not consider spatial information, such as the original SIFT method of Lowe (2004) which uses simple, individual keypoints. (2) Using 2-keygraphs, which presents similarities with the method of Li et al. (2015) that finds matches of keypoint pairs. (3) Using 3-keygraphs. (4) Using 4-keygraphs. And (5) using the SCRAMSAC method of Sattler et al. (2009).

The maximum and minimum allowed keygraph edge length in a query image was set as $l_{\text{max}} = 256$ and $l_{\text{min}} = 8$ pixels, for images of resolution $640 \times 480$; those values were set empirically, following the discussion in Section 3.2.1. RANSAC considers a keypoint match as correct if the distance, in the query image, between the true keypoint position $(x, y)$ and the mapped model keypoint’s position $(x', y')$ is lower than three pixels. During image traversal, each query keypoint $p$ investigated $L = 4000$ model keypoints, leading to $p$ to establish up to 4000 initial keypoint matches (with at most one match with each model image). Next, each query keypoint retained only its $N < L$ matches with highest similarity between descriptors.

Figure 7 presents the AP per maximum number $N$ of initial matches of a query keypoint. Limiting $N$ is useful in order to control the quadratic complexity of the keygraph matching.

---

3In matching methods based on 3D object models (e.g. the method of Hsiao et al. (2010)), training keypoints obtained from multiple viewpoints can be combined to generate a putative object pose. This enables the projected 3D object to have a larger area than in case matching is performed by using a single 2D warped training image that only considers a frontal view of the object (which is the case of the present paper). Therefore, a threshold of 0.5 on the overlap criterion (intersection divided by union of segmentation masks) is too tight for image-to-image matching methods in Hsiao et al. (2010)’s dataset. Based on preliminary experiments, we found that setting the threshold as 0.4 allows a more fair comparison of the investigated methods. As an illustration, Figures 9 and 10 shows cases in which keypoint correctly matched the query images but, because of the way the training segmentation is projected, their overlap criterion fell under the 0.5 threshold.
phase. Experiments show that using a small $N$ generated few initial matches in total. In this case, simple keypoints achieved a similar AP as keygraphs; this demonstrates that transforming keypoints into keygraphs did not eliminate a significant number of correct keypoint matches. Simple keypoints benefited moderately from using a larger number of initial matches: the AP improved from .39 to .44 when $N$ increased from $10$ to $10^2$ (however, a large number of RANSAC iterations, $R = 10^6$, was used). Then, as $N$ increased even more, so did the number of initial matches and the fraction of incorrect correspondences; in this case, simple keypoints achieved a poor performance, as a consequence of an infeasible number of RANSAC iterations becoming necessary. In contrast, 4-keygraphs benefited from using a larger number of initial matches: the AP improved from .37 to .53 when $N$ increased from $10$ to $10^3$. This demonstrates that using a larger $N$ did yield a larger number of correct keypoint matches, even though those additional matches had a lower descriptor similarity than the matches obtained by using a small $N$ only. Interestingly, 2-keygraphs achieved a consistently lower AP than 3- or 4-keygraphs, even when much more RANSAC iterations were used; this is a consequence of 2-keygraphs not filtering out as many incorrect matches as 3- or 4-keygraphs do. 2-keygraphs achieved its best performance (an AP of .50) by using a moderate number of initial matches, $N = 500$, and a large number of RANSAC iterations, $R = 10^6$.

Figure 8 shows the estimated AP per number of RANSAC iterations. We used a moderately large number of initial matches $(N = 10^3)$; nevertheless, the stage which finds 2-keygraphs presented a small computational cost in comparison to the keypoint matching stage. 4-keygraphs achieved a high AP even using very few RANSAC iterations. 3-keygraphs performed slightly worse than 4-keygraphs when very few RANSAC iterations were employed. In case of 2-keygraphs, much more RANSAC iterations were necessary; for instance, 2-keygraphs achieved an AP of .36 and .47 for $R = 10^5$ and $R = 10^6$, respectively, while 4-keygraphs achieved .51 and .53 for $R = 10^4$ and $R = 10^6$, respectively.

We evaluated the computational cost of a single-thread C implementation of the object recognition pipeline. By using 4-keygraphs and $N = 10^3$, each query image required approximately eight seconds (not considering SIFT feature extraction).

Table 2. Average AP and average number of keypoint matches between a query image and each model image (before RANSAC), using $N = 10^3$ initial matches per query keypoint. The method based on 2-keygraphs, which employ $R = 10^6$ RANSAC iterations, require a total computational cost approximately 25% larger than the methods based on 3- or 4-keygraphs.

<table>
<thead>
<tr>
<th>Method</th>
<th>Avg. number of keypoint matches</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keypoints, $R = 10^6$</td>
<td>44.7</td>
<td>.35</td>
</tr>
<tr>
<td>2-keygraphs, $R = 10^6$</td>
<td>10.1</td>
<td>.47</td>
</tr>
<tr>
<td>3-keygraphs, $R = 10^5$</td>
<td>0.12</td>
<td>.52</td>
</tr>
<tr>
<td>4-keygraphs, $R = 10^6$</td>
<td>0.06</td>
<td>.53</td>
</tr>
<tr>
<td>SCRAMSAC, $R = 10^6$</td>
<td>1.5</td>
<td>.48</td>
</tr>
</tbody>
</table>

Figure 8. Average AP versus number $R$ of RANSAC iterations, using $N = 10^3$ initial matches per query keypoint.
9-b. From Figure 9-a to 9-b, two incorrect 2-keygraph matches were filtered out. Figures 9-c and 9-d show the same query image as before, but considering a different model image, presenting a larger number of established keypoint matches: Figure 9-c shows 4-keygraph matches, while Figure 9-d shows the keypoint matches after RANSAC as well as the affine transformation mapping images. Figure 9-e presents incorrect 3-keygraph matches that were eliminated when 4-keygraphs were obtained. Figure 9-f shows another example of keypoint matches after RANSAC and the calculated affine transformation.

We compared 4-keygraphs with our previous method (Dazzi et al. 2016), which finds matches of keypoint triples that are generated by using several Delaunay triangulations. A valid keypoint triple is required to satisfy similar spatial properties as 3-keygraphs. Such an approach is particularly effective in a context where a large number of initial keypoint matches is established, since they are filtered out by using triangulations.

In this experiment, the original dataset adopted by Dazzi et al. (2016) is used. It is composed of the 250 images of the CMU10 dataset employing a K-means tree with $K = 16$. During tree traversal, a query image keypoint is compared against $L = 50$ stored model keypoints. In case of 4-keygraphs, a maximum of $N = 5$ initial keypoint matches is retained for each query keypoint, while the method of Dazzi et al. (2016) uses $N = 50$. We also investigated the performance of the “ratio test” proposed in Lowe (2004), in which each query keypoint matches a single model keypoint. In this experiment, $R = 10^3$ RANSAC iterations were used, which was sufficiently large for all methods. Table 3 shows the results. Lowe’s ratio test and Dazzi et al. (2016)’s method achieved an AP of .53 and .55, respectively, while 4-keygraphs achieved a significantly higher AP (.68). This result demonstrates that the methods of Dazzi et al. (2016) and Lowe’s ratio test did eliminate correct keypoint matches, since they achieved a lower AP than 4-keygraphs.

### Table 3. AP results obtained on the dataset of Dazzi et al. (2016).

<table>
<thead>
<tr>
<th>Method</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ratio test of Lowe (2004)</td>
<td>.53</td>
</tr>
<tr>
<td>Multiple triangulations of Dazzi et al. (2016)</td>
<td>.55</td>
</tr>
<tr>
<td>4-keygraphs, proposed in this paper</td>
<td>.68</td>
</tr>
</tbody>
</table>

Keygraphs of cardinality $k$ are defined based on keygraphs of cardinality $k - 1$. Keypoint matches are transformed into 2-keygraphs, which involves calculating changes in length, orientation and scale. In our experiments, obtaining 2-keygraphs reduced 75% of the incorrect keypoint matches; this operation had a small computational cost in comparison to the cost of keypoint matching. Next, 3-keygraphs are obtained at a negligible computational cost, which yielded a reduction of 99% of the remaining incorrect keypoint matches. As a consequence, the method based on 3-keygraphs achieved a high performance even using 1% of the RANSAC iterations in comparison to the method based on 2-keygraphs. Our experiments also showed that using all the initial keypoint matches, as well as SCRAMSAC, performed worse than using 3-keygraphs.

We proposed 4-keygraphs, that are generated from a pair of 3-keygraphs sharing an edge. Such method to obtain 4-keygraphs can be employed to find keygraphs of cardinality larger than four. However, since a valid match of $k$-keygraphs requires $k$ correct keypoint matches, one drawback associated to using $k > 4$ is a reduced probability of detecting small or occluded objects, which present few keypoint matches. In this paper, we used 4-keygraphs, that provided the best results.

The keygraphs method has few intrinsic parameters. The two most important ones are the thresholds on attribute changes (scale/length and orientation). The values were selected in order to allow a large range of viewpoint change between images.

In this paper, we employed SIFT features, chosen mostly due to their popularity. Keygraph matching uses scale, orientation and position computed by the keypoint detector. SIFT could be readily replaced by other descriptors which generate scale, orientation and position information with a similar precision as SIFT. Examples of methods proposed more recently than SIFT are SURF (Speeded-Up Robust Features) and ORB (Oriented FAST corner detector and Rotated BRIEF features). The main improvement that they brought is in terms of computational efficiency, while the improvement in terms of performance is not as substantial. CNN-based local descriptors have been proposed more recently and could also replace SIFT, as long as they are designed to produce local scale and orientation information. However, CNN-based descriptors require substantially more computational power than SIFT, SURF and ORB. It is worth noting that the proposed keygraphs method tolerates an elevated amount of noise in the estimations of scale, orientation and position of keypoints, being robust to wide variations in viewpoint. Its focus is on the structure of small sets of local features rather the quality of individual matches. A benchmark evaluation of descriptors is therefore beyond the scope of this paper and constitutes a suggestion for future work.

Another direction for future work is an extension of our keygraphs method for 3D point clouds, using 3D keypoint detectors and descriptors for applications such as 3D object retrieval and 3D scene understanding.

### Acknowledgment

The authors are grateful to FAPESP grant #2011/50761-2, #14/50769-1, 2015/22308-2, CNPq grant #502472/2014-6, CAPES, NAP eScience - PRP - USP.
Fig. 9. Matches between query (left) and model (right) images. Keygraph matches before RANSAC: (9-a) 2-keygraphs; (9-b) and (9-c) 4-keygraphs; (9-e) 3-keygraphs. (9-d) and (9-f) show final keypoint matches (after RANSAC) and the calculated affine transformations.

References


