MIT-CSAIL-TR-2010-024 May 8, 2010

SIFT Flow: Dense Correspondence across Scenes and its Applications

Ce Liu, Jenny Yuen, Antonio Torralba, and William T. Freeman

SIFT Flow: Dense Correspondence across Scenes and its Applications

Ce Liu, Jenny Yuen, Antonio Torralba, and William T. Freeman

Abstract—While image alignment has been studied in different areas of computer vision for decades, aligning images depicting different scenes remains a challenging problem. Analogous to optical flow where an image is aligned to its temporally adjacent frame, we propose *SIFT flow*, a method to align an image to its nearest neighbors in a large image corpus containing a variety of scenes. The SIFT flow algorithm consists of matching densely sampled, pixel-wise SIFT features between two images, while preserving spatial discontinuities. The SIFT features allow robust matching across different scene/object appearances, whereas the discontinuity-preserving spatial model allows matching of objects located at different parts of the scene. Experiments show that the proposed approach robustly aligns complex scene pairs containing significant spatial differences. Based on SIFT flow, we propose an alignment-based large database framework for image analysis and synthesis, where image information is transferred from the nearest neighbors to a query image according to the dense scene correspondence. This framework is demonstrated through concrete applications, such as motion field prediction from a single image, motion synthesis via object transfer, satellite image registration and face recognition.

Index Terms—Scene alignment, dense scene correspondence, SIFT flow, coarse-to-fine, belief propagation, alignment-based large database framework, satellite image registration, face recognition, motion prediction for a single image, motion synthesis via object transfer

1 Introduction

Image alignment, registration and correspondence are central topics in computer vision. There are several levels of scenarios in which image alignment dwells. The simplest level, aligning different views of the same scene, has been studied for the purpose of image stitching [51] and stereo matching [45], *e.g.* in Figure 1 (a). The considered transformations are relatively simple (*e.g.* parametric motion for image stitching and 1D disparity for stereo), and images to register are typically assumed to have the same pixel value after applying the geometric transformation.

The image alignment problem becomes more complicated for dynamic scenes in video sequences, *e.g.* optical flow estimation [12], [29], [38]. The correspondence between two adjacent frames in a video is often formulated as an estimation of a 2D flow field. The extra degree of freedom transitioning from 1D in stereo to 2D in optical flow introduces an additional level of complexity. Typical assumptions in optical flow algorithms include brightness constancy and piecewise smoothness of the pixel displacement field [3], [8].

Image alignment becomes even more difficult in the object recognition scenario, where the goal is to align different instances of the same object category, as illustrated in Figure 1 (b). Sophisticated object representations [4], [6], [19], [57] have been developed to cope with the variations of object shapes and appearances. However, these methods still typically require objects to be salient, similar, with limited background clutter.

In this work, we are interested in a new, higher level of image alignment: aligning two images from different 3D scenes but sharing similar scene characteristics. Image alignment at the scene level is thus called *scene alignment*. As illustrated in Figure 1 (c), the two images to match may contain object instances captured from different viewpoints, placed at different spatial locations, or imaged at different scales. The two images may also contain different quantities of objects of the same category, and some objects present in one image might be missing in the other. Due to these issues the scene alignment problem is extremely challenging.

Ideally, in scene alignment we want to build correspondence at the semantic level, *i.e.* matching at the object class level, such as buildings, windows and sky. However, current object detection and recognition techniques are not robust enough to detect and recognize all objects in images. Therefore, we take a different approach for scene alignment by matching local, salient, and transform-invariant image structures. We hope that semantically meaningful correspondences can be established through matching these image structures. Moreover, we want to have a simple, effective, object-free model to align image pairs such as the ones in Figure 1 (c).

Inspired by optical flow methods, which are able to produce dense, pixel-to-pixel correspondences between two images, we propose *SIFT flow*, adopting the computational framework of optical flow, but by matching SIFT descriptors instead of raw pixels. In SIFT flow, a SIFT descriptor [37] is extracted at each pixel to characterize local image structures and encode contextual information. A discrete, discontinuity preserving, flow estimation algorithm is used to match the SIFT descriptors between two images. The use of SIFT features allows robust matching across different scene/object appearances and

• *Ce Liu is with Microsoft Research New England, One Memorial Drive, Cambridge, MA 02142. Email: celiu@microsoft.com.*
Jenny Yuen, Antonio Torralba and William T. Freeman are with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139. Email: {jenny,torralba,billf}@csail.mit.edu

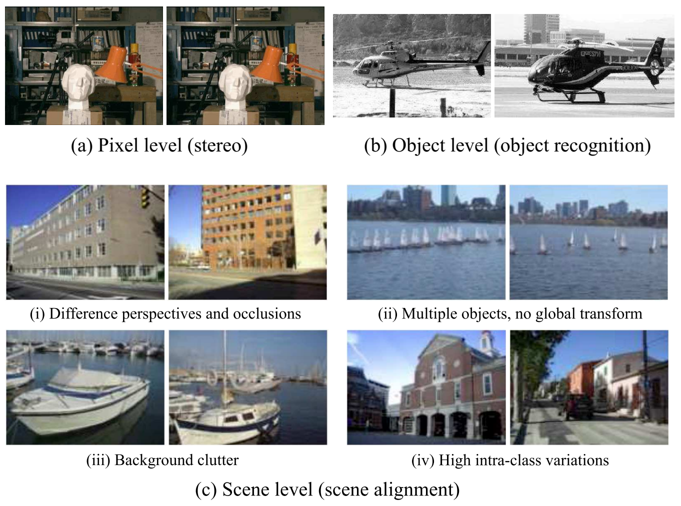

(a) Pixel level (stereo)
(b) Object level (object recognition)
(i) Difference perspectives and occlusions
(ii) Multiple objects, no global transform
(iii) Background clutter
(iv) High intra-class variations
(c) Scene level (scene alignment)

Fig. 1. **Image alignment resides at different levels**. Researchers used to study image alignment problems at the pixel level, where two images are captured from the same scene with slightly different time or at different perspective [45] (a). Recently, correspondence has been extended to the object level (b) for object recognition [6]. We are interested in image alignment at the scene level, where two images come from different 3D scene but share similar scene characteristics (c). SIFT flow is proposed to align the examples in (c) for scene alignment.

the discontinuity-preserving spatial model allows matching of objects located at different parts of the scene. Moreover, a coarse-to-fine matching scheme is designed to significantly accelerate the flow estimation process.

Optical flow is only applied between two adjacent frames in a video sequence in order to obtain meaningful correspondences; likewise, we need to define the *neighborhood* for SIFT flow. Motivated by the recent progress in large image database methods [28], [43], we define the neighbors of SIFT flow as the top matches retrieved from a large database. The chance that some of the nearest neighbors share the same scene characteristics with a query image increases as the database grows, and the correspondence obtained by SIFT flow can be semantically meaningful.

Using SIFT flow, we propose an alignment-based large database framework for image analysis and synthesis. The information to infer for a query image is transferred from the nearest neighbors in a large database to this query image according to the dense scene correspondence estimated by SIFT flow. Under this framework, we apply SIFT flow to two novel applications: *motion prediction from a single static image*, where a motion field is hallucinated from a large database of videos, and *motion transfer*, where a still image is animated using object motions transferred from a similar moving scene. We also apply SIFT flow back to the regime of traditional image alignment, such as satellite image registration and face recognition. Through these examples we demonstrate the potential of SIFT flow for broad applications in computer vision and computer graphics.

The rest of the paper is organized as follows: after reviewing the related work in Sect. 2, we introduce the concept of SIFT flow and the inference algorithm in Sect. 3. In Sect. 4, we show scene alignment examples using SIFT flow with evaluations. In Sect. 5, we show how to infer the motion field from a single image, and how to animate a still image, both with the support of a large video database and scene alignment. We further apply SIFT flow to satellite image registration and face recognition in Sect. 6. After briefly discussing how SIFT flow fits in the literature of image alignment in Sect. 7, we conclude the paper in Sect. 8.

2 Related Work

Image alignment, *a.k.a.* image registration or correspondence, is a broad topic in computer vision, computer graphics and medical imaging, covering stereo, motion analysis, video compression, shape registration, and object recognition. It is beyond the scope of this paper to give a thorough review on image alignment. Please refer to [51] for a comprehensive review on this topic. In this section, we will review the image alignment literature focusing on

(a) *What* to align, or the features that are consistent across images, *e.g.* pixels, edges, descriptors;
(b) *Which* way to align, or the representation of the alignment, *e.g.* sparse vs. dense, parametric vs. nonparametric;
(c) *How* to align, or the computational methods to obtain alignment parameters.

In addition, correspondence can be established between two images, or between an image and image models such as in [15]. We will focus on the correspondence between two images.

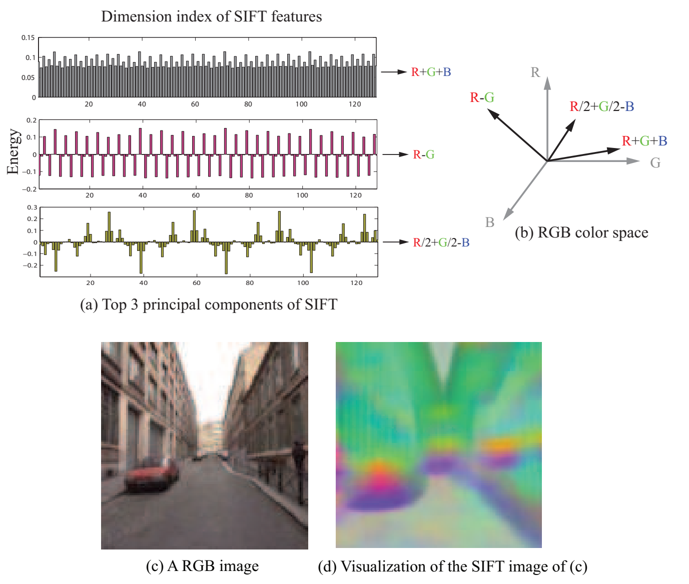
In image alignment we must first define the features based on which image correspondence will be established: an image measurement that does not change from one image to another. In stereo [26] and optical flow [29], [38], the brightness constancy assumption was often made for building the correspondence between two images. But soon researchers came to realize that pixel values are not reliable for image matching due to changes of lighting, perspective and noise [25]. Features such as phase [21], filter banks [30], mutual information [54] and gradient [11] are used to match images since they are more reliable than pixel values across frames, but they still fail to deal with drastic changes. Middle-level representations such as scale-invariant feature transform (SIFT) [37], shape context [5], [6], histogram of oriented gradients (HOG) [17] have been introduced to account for stronger appearance changes, and are proven to be effective in a variety of applications such as visual tracking [1], optical flow estimation [10] and object recognition [37]. Nevertheless, little has been investigated for exploring features to establish correspondences at the scene level.

The representation of the correspondence is another important aspect of image alignment. One can utilize the information of every pixel to obtain a dense correspondence, or merely use sparse feature points. The form of the correspondence can be pixel-wise displacement such as a 1-D disparity map (stereo) and a 2-D flow field (optical flow), or parametric models such as affine and homography. Although a parametric model can be estimated from matching every pixel [7], and a dense correspondence can be interpolated from sparse matching [56], typically, pixel-wise displacement is obtained through pixel-wise correspondence, and parametric motion is estimated from sparse, interest point detection and matching [46]. In between the sparse and dense representation is correspondence on contours [33], [55], which has been used in tracking objects and analyzing motion for textureless objects. The fact that the underlying correspondence between scenes is complicated and unclear, and detecting contours from scenes can be unreliable, leads us to seek for dense, pixel-wise correspondence for scene alignment.

Estimating dense correspondence between two images is a nontrivial problem with spatial regularity, *i.e.* the displacements (flow vectors) of neighboring pixels tend to be similar. When the feature values of the two images are close and temporally smooth, this displacement can be formulated as a continuous variable and the estimation problem is often reduced to solving PDE's using Euler-Lagrange [11], [29]. When the feature values are different, or other information such as occlusion needs to be taken into account, one can use belief propagation [22], [49] and graph cuts [9], [31] to optimize objective functions formulated on Markov random fields. The recent studies show that optimization tools such as belief propagation, tree-reweighted belief propagation and graph cuts can achieve very good local optimum for these optimization problems [52]. In [47], a dual-layer formulation is proposed to apply tree-reweighted BP to estimate optical flow fields. These advances in inference on MRF's allow us to solve dense scene matching problems effectively.

Scene retrieval, parsing and recognition has become an important research direction to understand images at the scene level [39], [53]. Image representations, such as color histograms [50], texture models [23], segmented regions [14], GIST descriptors [39], bag of words [18] and spatial pyramids [32], have been proposed to find similar images at a global level. Common to all these representations is the lack of meaningful correspondences across different image regions, and therefore, spatial structural information of images tends to be ignored. Our interest is to establish dense correspondences between images across scenes, an alignment problem that can be more challenging than aligning images from the same scene and aligning images of the same object category since we wish all the elements that compose the scene to be aligned. Our work relates to the task of co-segmentation [41] that tried to simultaneously segment the common parts of an image pair, and to the problem of shape matching [5] that was used in the context of object recognition.

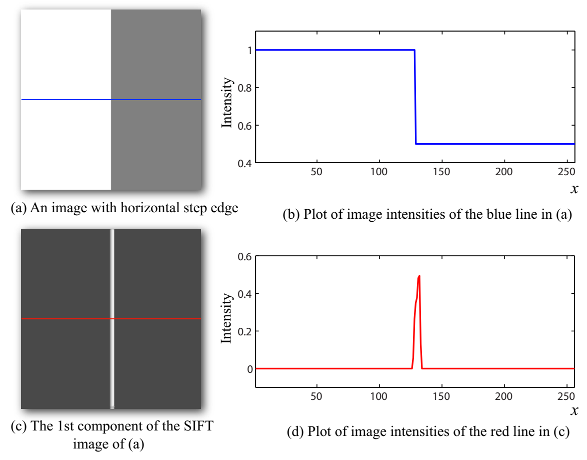
Inspired by the recent advances in image alignment and scene parsing, we propose SIFT flow to establish the correspondence between images across scenes. An early version of our work was presented in [36]. In this paper, we will explore the SIFT flow algorithm in more depth and will demonstrate a wide array of applications for SIFT flow.



(a) Top 3 principal components of SIFT (b) RGB color space

(c) A RGB image (d) Visualization of the SIFT image of (c)

Fig. 2. **Visualization of SIFT images**. To visualize SIFT images, we compute the top three principal components of SIFT descriptors from a set of images (a), and then map these principal components to the principal components of the RGB space (b). For an image in (c), we compute the 128-d SIFT feature for every pixel, project the SIFT feature to 3d color space, and visualize the SIFT image as shown in (d). Intuitively, pixels with similar colors share similar structures.



(a) An image with horizontal step edge (b) Plot of image intensities of the blue line in (a)

(c) The 1st component of the SIFT image of (a) (d) Plot of image intensities of the red line in (c)

Fig. 3. **The resolution of SIFT images**. Although histograms are used to represent SIFT features, SIFT images are able to capture image details. For a toy image with a horizontal step-edge in (a), we show the 1st component of the SIFT image in (c). We plot the slice of a horizontal line in (a) (blue) and (c) (red) in (b) and (d), respectively. The sharp boundary in (d) suggests that SIFT images have high resolutions.

3 The SIFT Flow Algorithm

3.1 Dense SIFT descriptors and visualization

SIFT is a local descriptor to characterize local gradient information [37]. In [37], SIFT descriptor is a sparse feature representation that consists of both feature extraction and detection. In this chapter, however, we only use the feature extraction component. For every pixel in an image, we divide its neighborhood (*e.g.* 16×16) into a 4×4 cell array, quantize the orientation into 8 bins in each cell, and obtain a $4\times4\times8=128$-dimensional vector as the SIFT representation for a pixel. We

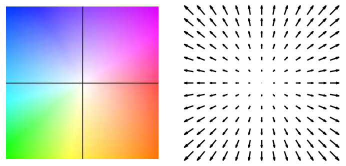


Fig. 4. **The visualization of flow fields**. We follow the way in [2] to visualize a flow field: each pixel denotes a flow vector where the orientation and magnitude are represented by the huge and saturation of the pixel, respectively.

call this per-pixel SIFT descriptor *SIFT image*.

To visualize SIFT images, we compute the top three principal components of SIFT descriptors from a set of images, and then map these principal components to the principal components of the RGB space, as shown in Figure 2. Through projecting a 128D SIFT descriptor to a 3D subspace, we are able to compute the SIFT image from an RGB image in Figure 2 (c) and visualize it in (d). In this visualization, the pixels that have similar color may imply that they share similar local image structures. Note that this projection is only for visualization; in SIFT flow, the entire 128 dimensions are used for matching.

Notice that even though this SIFT visualization may look blurry as shown in Figure 2 (d), SIFT images indeed have high spatial resolution as suggested by Figure 3. We designed an image with a horizontal step-edge (Figure 3 (a)), and show the 1st component of the SIFT image of (a) in (c). Because every row is the same in (a) and (c), we plot the middle row of (a) and (c) in (b) and (d), respectively. Clearly, the SIFT image contains a sharp edge with respect to the sharp edge in the original image.

Now that we have per-pixel SIFT descriptors for two images, our next task is to build dense correspondence to match these descriptors.

3.2 Matching Objective

We design an objective function similar to that of optical flow to estimate SIFT flow from two SIFT images. Similar to optical flow [11], [12], we want SIFT descriptors to be matched along the flow vectors, and the flow field to be smooth, with discontinuities agreeing with object boundaries. Based on these two criteria, the objective function of SIFT flow is formulated as follows. Let $\mathbf{p}=(x,y)$ be the grid coordinate of images, and $\mathbf{w}(\mathbf{p})=(u(\mathbf{p}),v(\mathbf{p}))$ be the flow vector at $\mathbf{p}$. We only allow $u(\mathbf{p})$ and $v(\mathbf{p})$ to be integers and we assume that there are L possible states for $u(\mathbf{p})$ and $v(\mathbf{p})$, respectively. Let s_1 and s_2 be two SIFT images that we want to match. Set ε contains all the spatial neighborhoods (a four-neighbor system is used). The energy function for SIFT flow is defined as:

$$E(\mathbf{w}) = \sum_{\mathbf{p}} \min\Big(\big\|s_1(\mathbf{p}) - s_2(\mathbf{p}+\mathbf{w}(\mathbf{p}))\big\|_1, t\Big) + \quad (1)$$

$$\sum_{\mathbf{p}} \eta\Big(|u(\mathbf{p})| + |v(\mathbf{p})|\Big) + \quad (2)$$

$$\sum_{(\mathbf{p},\mathbf{q})\in\varepsilon} \min\Big(\alpha|u(\mathbf{p}) - u(\mathbf{q})|, d\Big) + \min\Big(\alpha|v(\mathbf{p}) - v(\mathbf{q})|, d\Big), \quad (3)$$

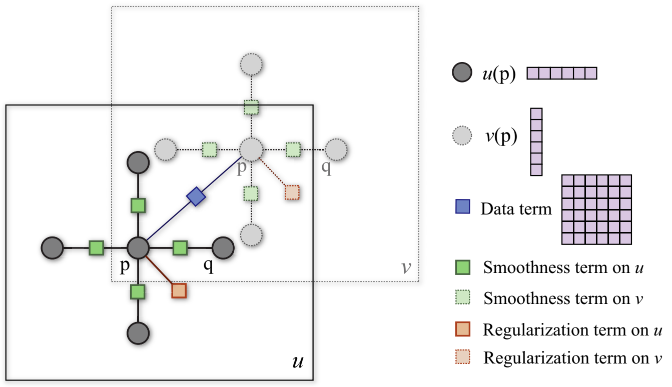


Fig. 5. **Dual-layer Belief Propagation**. We designed the objective function of SIFT flow to be decoupled for horizontal (u) and vertical (v) components.

which contains a *data term*, *small displacement term* and *smoothness term* (*a.k.a.* spatial regularization). The *data term* in Eqn. 1 constrains the SIFT descriptors to be matched along with the flow vector $\mathbf{w}(\mathbf{p})$. The *small displacement term* in Eqn. 2 constrains the flow vectors to be as small as possible when no other information is available. The *smoothness term* in Eqn. 3 constrains the flow vectors of adjacent pixels to be similar. In this objective function, truncated L1 norms are used in both the data term and the smoothness term to account for matching outliers and flow discontinuities, with t and d as the threshold, respectively.

We use a dual-layer loopy belief propagation as the base algorithm to optimize the objective function. Different from the usual formulation of optical flow [11], [12], the smoothness term in Eqn. 3 is decoupled, which allows us to separate the horizontal flow $u(\mathbf{p})$ from the vertical flow $v(\mathbf{p})$ in message passing, as suggested by [47]. As a result, the complexity of the algorithm is reduced from $O(L^4)$ to $O(L^2)$ at one iteration of message passing. The factor graph of our model is shown in Figure 5. We set up a horizontal layer u and vertical layer v with exactly the same grid, with the data term connecting pixels at the same location. In message passing, we first update *intra*-layer messages in u and v separately, and then update *inter*-layer messages between u and v. Because the functional form of the objective function has truncated L1 norms, we use distance transform function [20] to further reduce the complexity, and sequential belief propagation (BP-S) [52] for better convergence.

3.3 Coarse-to-fine matching scheme

Despite the speed up, directly optimizing Eqn. (3) using this dual-layer belief propagation scales poorly with respect to image dimension. In SIFT flow, a pixel in one image can literally match to any pixels in the other image. Suppose the image has h^2 pixels, then $L \approx h$, and the time and space complexity of this dual-layer BP is $O(h^4)$. For example, the computation time for 145×105 images with an 80×80 search window is 50 seconds. It would require more than two hours to process a pair of 256×256 images with a memory usage of 16GB to store the data term.

To address the performance drawback, we designed a coarse-to-fine SIFT flow matching scheme that significantly

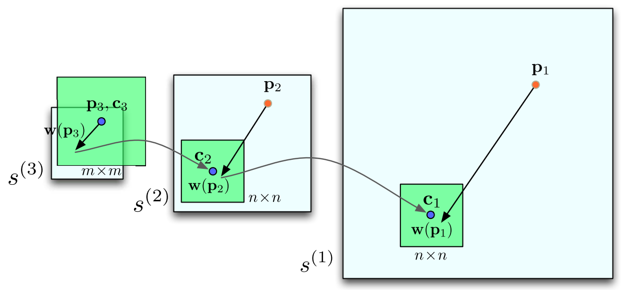


Fig. 6. An illustration of coarse-to-fine SIFT flow matching on pyramid. The green square is the searching window for $\mathbf{p}_k$ at each pyramid level k. For simplicity only one image is shown here, where $\mathbf{p}_k$ is on image s_1, and $\mathbf{c}_k$ and $\mathbf{w}(\mathbf{p}_k)$ are on image s_2. See text for details.

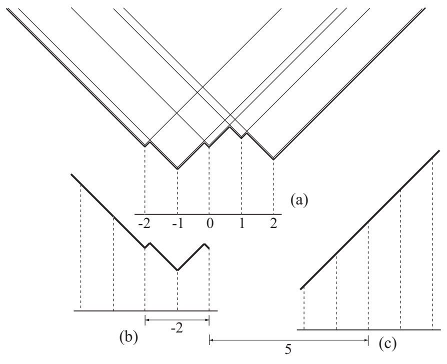


Fig. 7. We generalized the distance transform function for truncated L1 norm [20] to pass messages between neighboring nodes that have different offsets (centroids) of the searching window.

improves the performance. The basic idea is to roughly estimate the flow at a coarse level of image grid, then gradually propagate and refine the flow from coarse to fine. The procedure is illustrated in Figure 6. For simplicity, we use s to represent both s_1 and s_2. A SIFT pyramid $\{s^{(k)}\}$ is established, where $s^{(1)} = s$ and $s^{(k+1)}$ is smoothed and downsampled from $s^{(k)}$. At each pyramid level k, let $\mathbf{p}_k$ be the coordinate of the pixel to match, $\mathbf{c}_k$ be the offset or centroid of the searching window, and $\mathbf{w}(\mathbf{p}_k)$ be the best match from BP. At the top pyramid level $s^{(3)}$, the searching window is centered at $\mathbf{p}_3$ ($\mathbf{c}_3 = \mathbf{p}_3$) with size $m\times m$, where m is the width (height) of $s^{(3)}$. The complexity of BP at this level is $O(m^4)$. After BP converges, the system propagates the optimized flow vector $\mathbf{w}(\mathbf{p}_3)$ to the next (finer) level to be $\mathbf{c}_2$ where the searching window of $\mathbf{p}_2$ is centered. The size of this searching window is fixed to be $n\times n$ with $n = 11$. This procedure iterates from $s^{(3)}$ to $s^{(1)}$ until the flow vector $\mathbf{w}(\mathbf{p}_1)$ is estimated. The complexity of this coarse-to-fine algorithm is $O(h^2 \log h)$, a significant speed up compared to $O(h^4)$. Moreover, we double η and retain α and d as the algorithm moves to a higher level of pyramid in the energy minimization.

When the matching is propagated from an coarser level to a finer level, the searching windows for two neighboring pixels may have different offsets (centroids). We modify the the distance transform function developed for truncated L1 norm [20] to cope with this situation, with the idea illustrated in Figure 7. To compute the message passing from pixel $\mathbf{p}$ to its neighbor $\mathbf{q}$, we first gather all other messages and data term, and apply the routine in [20] to compute the message from $\mathbf{p}$ to $\mathbf{q}$ assuming that $\mathbf{q}$ and $\mathbf{p}$ have the same offset and range. The function is then extended to be outside the range by increasing α per step, as shown in Figure 7 (a). We take the function in the range that $\mathbf{q}$ is relative to $\mathbf{p}$ as the message. For example, if the offset of the searching window for $\mathbf{p}$ is 0, and the offset for $\mathbf{q}$ is 5, then the message from $\mathbf{p}$ to $\mathbf{q}$ is plotted in Figure 7 (c). If the offset of the searching window for $\mathbf{q}$ is -2 otherwise, the message is shown in Figure 7 (b).

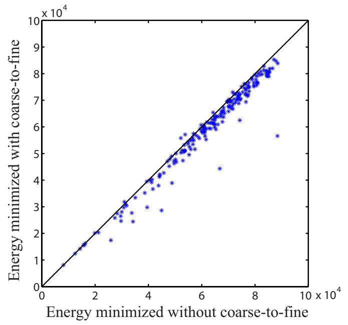


Fig. 8. Coarse-to-fine SIFT flow not only runs significantly faster, but also achieves lower energies most of the time. Here we compare the energy minimized using the coarse-to-fine algorithm (y-axis) and using the single-level version (x-axis) by running them on 200 pairs of examples. The coarse-to-fine matching achieves lower energy compared to the ordinary matching algorithm most of the time.

Using the proposed coarse-to-fine matching scheme and modified distance transform function, the matching between two 256×256 images takes 31 seconds on a workstation with two quad-core 2.67 GHz Intel Xeon CPUs and 32 GB memory, in a C++ implementation. Further speedup (up to 50x) can be achieved through GPU implementation [16] of the BP-S algorithm since this algorithm can be parallelized. We leave this as future work.

A natural question is whether the coarse-to-fine matching scheme can achieve the same minimum energy as the ordinary matching scheme (using only one level). We randomly selected 200 pairs of images to estimate SIFT flow, and check the minimum energy obtained using coarse-to-fine scheme and ordinary scheme (non coarse-to-fine), respectively. For these 256×256 images, the average running time of coarse-to-fine SIFT flow is 31 seconds, compared to 127 minutes in average for the ordinary matching. The coarse-to-fine scheme not only runs significantly faster, but also achieves lower energies most of the time compared to the ordinary matching algorithm as shown in Figure 8. This is consistent with what has been discovered in the optical flow community: coarse-to-fine search not only speeds up computation but also leads to better solutions.

3.4 Neighborhood of SIFT flow

In theory, we can apply optical flow to two arbitrary images to estimate a correspondence, but we may not get a meaningful correspondence if the two images are from different 3D scenes. In fact, even when we apply optical flow to two adjacent frames in a video sequence, we assume dense sampling in time so that there is significant overlap between two neighboring frames. Similarly, in SIFT flow, we define the neighborhood of an image as the nearest neighbors when we query a large database with the input. Ideally, if the database is large and dense enough to contain almost every possible image in the world, the nearest neighbors will be close to the query image, sharing similar local structures. This motivates the following analogy with optical flow:

Dense sampling in time : optical flow ::
Dense sampling in the space of all images : SIFT flow

As dense sampling of the time domain is assumed to enable tracking, dense sampling in (some portion of) the space of world images is assumed to enable scene alignment. In order to make this analogy possible, we collect a large database consisting of 102,206 frames from 731 videos, mostly from street scenes. Analogous to the time domain, we define the "adjacent frames" to a query image as its N nearest neighbors in this database. SIFT flow is then established between the query image and its N nearest neighbors.

For a query image, we use a fast indexing technique to retrieve its nearest neighbors that will be further aligned using SIFT flow. As a fast search we use spatial histogram matching of quantized SIFT features [32]. First, we build a dictionary of 500 visual words [48] by running K-means on 5000 SIFT descriptors randomly selected out of all the video frames in our dataset. Then, histograms of the visual words are obtained on a two-level spatial pyramid [24], [32], and histogram intersection is used to measure the similarity between two images.

Other scene metrics such as GIST [39] can also be used for retrieving nearest neighbors [35]. It has been reported that various nearest matching algorithms do not result in significant difference in obtaining nearest neighbors for matching [42].

4 Experiments on Video Retrieval

4.1 Results of video retrieval

We conducted several experiments to test the SIFT flow algorithm on our video database. One frame from each of the 731 videos was selected as the query image and histogram intersection matching was used to find its 20 nearest neighbors, excluding all other frames from the query video. The SIFT flow algorithm was then used to estimate the dense correspondence (represented as a pixel displacement field) between the query image and each of its neighbors. The best matches are the ones with the minimum energy defined by (3). Alignment examples are shown in Figure 11–13. The original query image and its extracted SIFT descriptors are shown in columns (a) and (b). The minimum energy match (out of the 20 nearest neighbors) and its extracted SIFT descriptors are shown in columns (c) and (d). To investigate the quality of the pixel displacement field, we use the computed displacements to warp the best match onto the query image. The warped image and warped SIFT descriptor image are shown in columns (e) and (f). The visual similarity between (a) and (e), and (b) and (f) demonstrates the quality of the matching. Finally, the

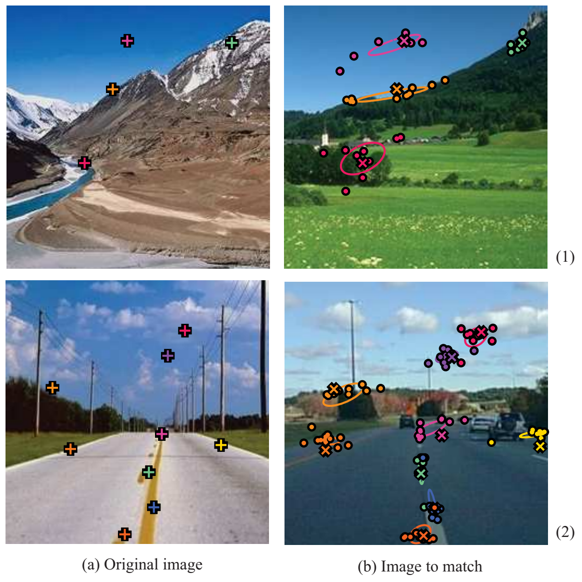


Fig. 9. For an image pair such as row (1) or row (2), a user defines several sparse points in (a) as "+". The human annotated matchings are marked as dot in (b), from which a Gaussian distribution is estimated and displayed as an ellipse. The correspondence estimated from SIFT flow is marked as "×" in (b).

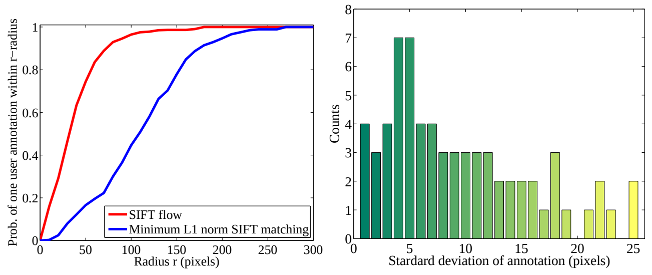


Fig. 10. The evaluation of SIFT flow using human annotation. Left: the probability of one human annotated flow lies within r distance to the SIFT flow as a function of r (red curve). For comparison, we plot the same probability for direct minimum L1-norm matching (blue curve). Clearly, SIFT flow matches human perception better. Right: the histogram of the standard deviation of human annotation. Human perception of scene correspondence varies from subject to subject.

displacement field, visualized using the color-coding in Figure 4 [2], is shown in column (g).

Figure 11 shows examples of matches between frames coming from exactly the same (3D) scene, but different video sequences. The reasonable matching in (1) and (2) demonstrates that SIFT flow reduces to classical optical flow when the two images are temporally adjacent frames in a video sequence. In (3)–(5), the query and the best match are more distant within the video sequence, but the alignment algorithm can still match them reasonably well.

Figure 12 shows more challenging examples, where the two frames come from different videos while containing the same type of objects. SIFT flow attempts to match the query image by reshuffling the pixels in the candidate image. Notice significant changes of objects between the query and the match in examples (8), (9), (11), (13), (14) and (16). The large amount of discontinuities in the flow field are due to: (i) the coefficient

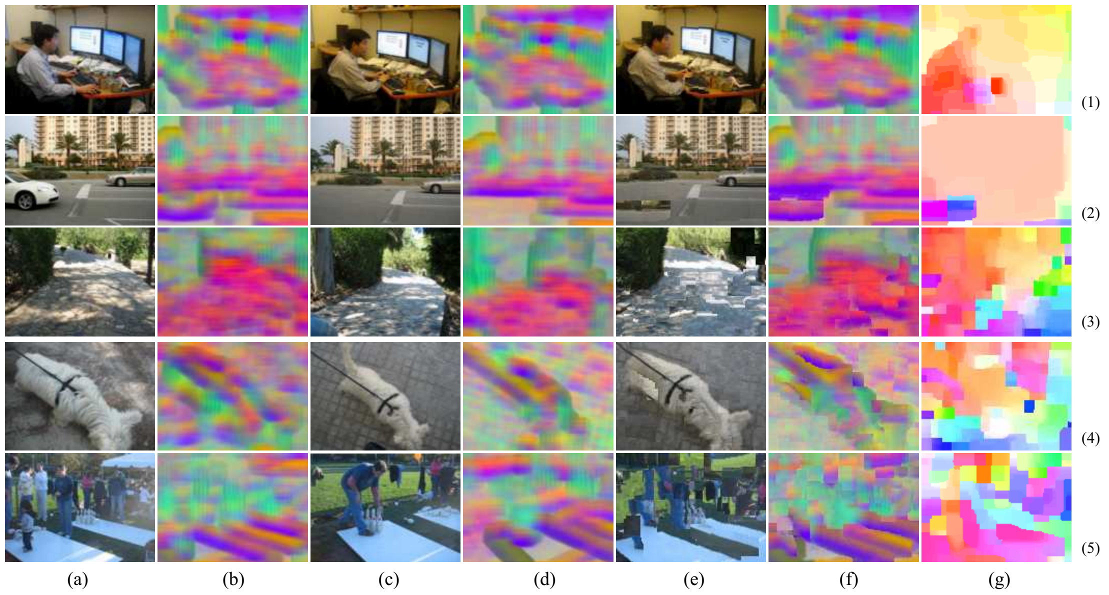


Fig. 11. SIFT flow for image pairs depicting the same scene/object. (a) shows the query image and (b) its densely extracted SIFT descriptors. (c) and (d) show the best (lowest energy) match from the database and its SIFT descriptors, respectively. (e) shows (c) warped onto (a). (f) shows the warped SIFT image (d). (g) shows the estimated displacement field with the minimum alignment energy shown to the right.

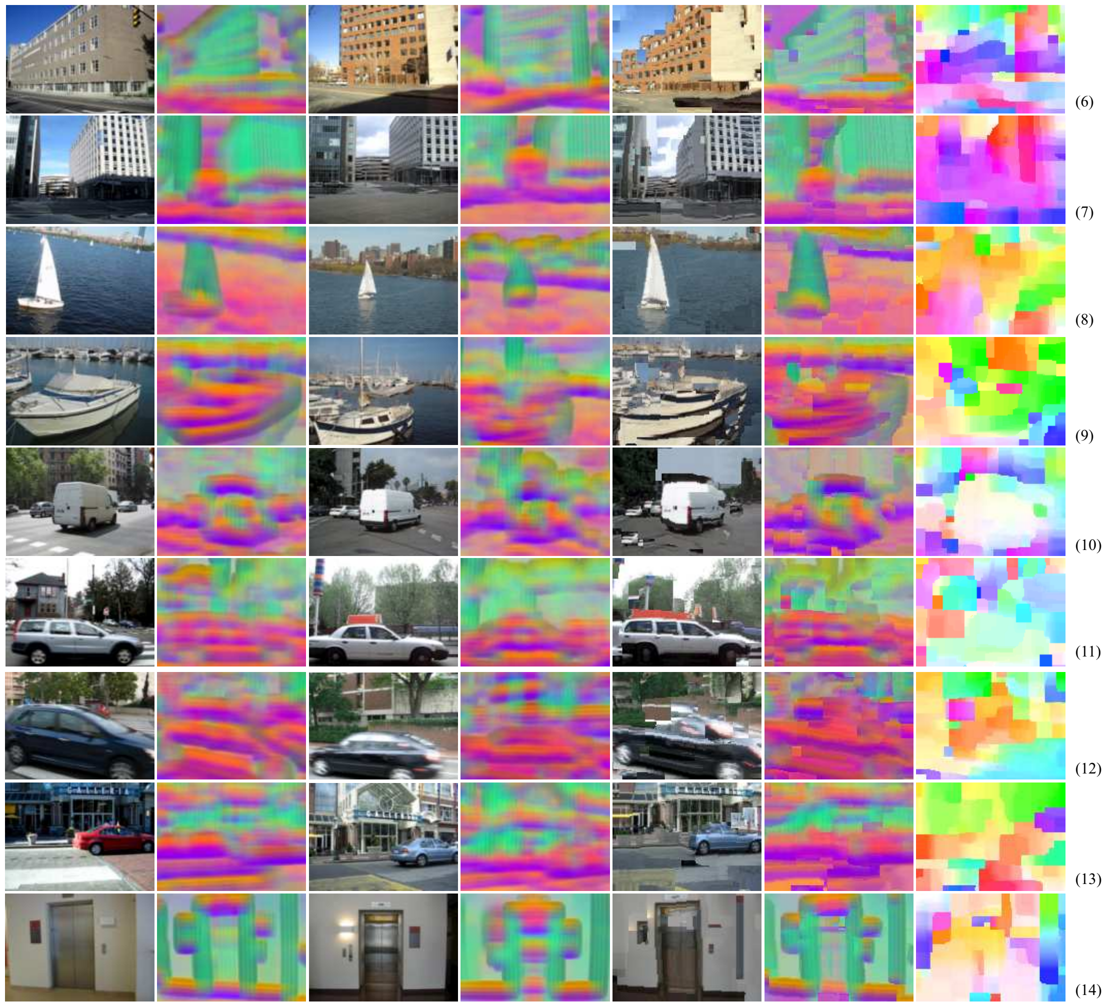


Fig. 12. SIFT flow computed for image pairs depicting the same scene/object category where the visual correspondence is obvious.

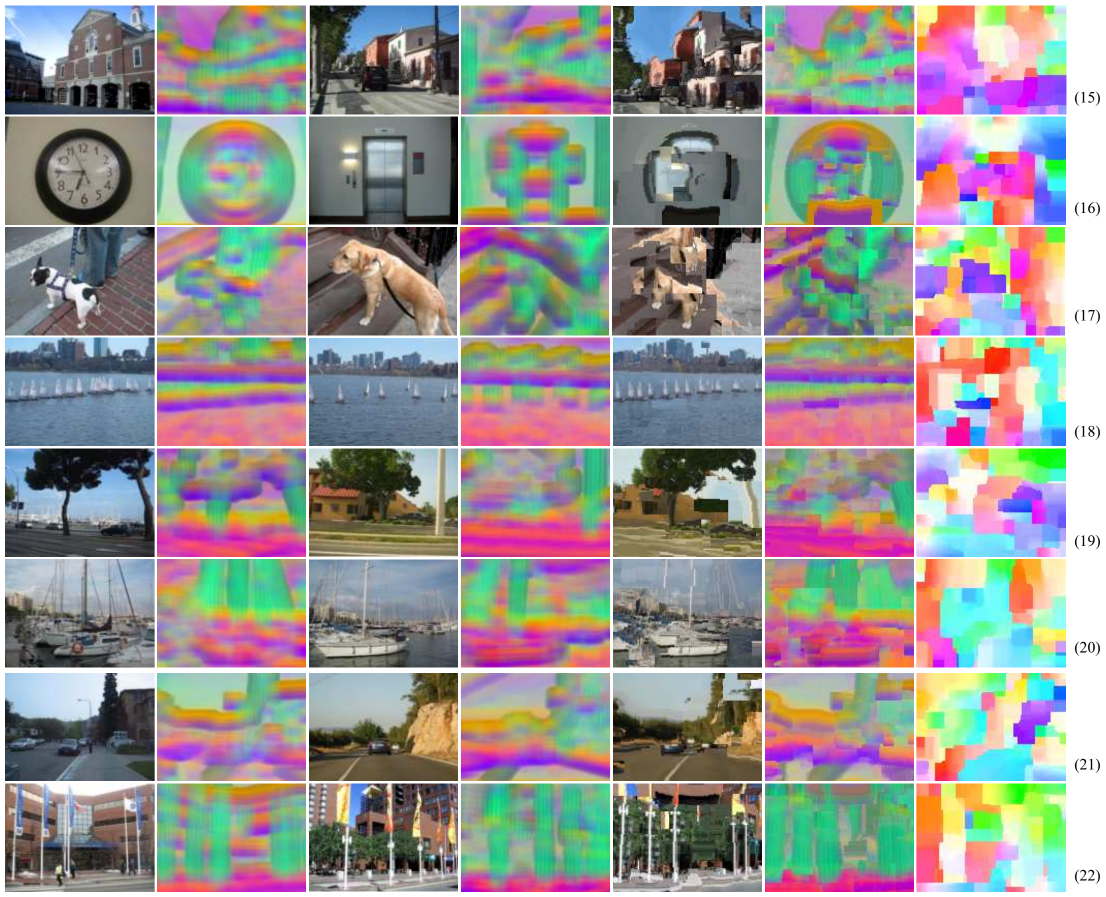


Fig. 13. SIFT flow for challenging examples where the correspondence is not obvious.

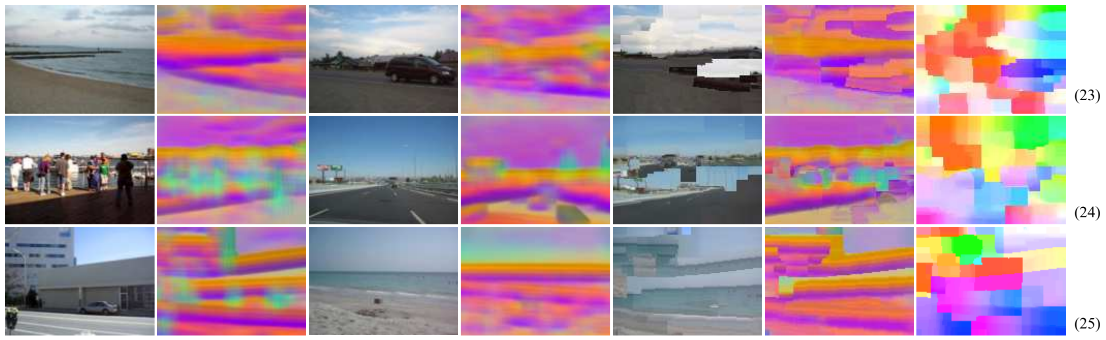


Fig. 14. Some failure examples with semantically incorrect correspondences. Although a SIFT flow field is obtained through minimizing the objective function, the query images are rare in the database and the best SIFT flow matches do not belong to the same scene category as the queries. However, these failures can be overcome through increasing the size of the database.

on spatial regularization α is small, and (ii) the content of the two images are too different to produce smooth matches (compare to example (1) and (2) in Figure 11. The square shaped discontinuities are a consequence of the decoupled regularizer on the horizontal and vertical components of the pixel displacement vector.

Figure 13 shows alignment results for examples with no obvious visual correspondences. Despite the lack of direct visual correspondences, SIFT flow attempts to rebuild the house (15), change the shape of the door into a circle (16) or reshuffle boats (18).

Some failure cases are shown in Figure 14, where the correspondences are not semantically meaningful. Typically, the failure are caused by the lack of visually similar images in the video database for the query image. It shows that our database is not dense enough in the space of images.

We find that SIFT flow improves the ranking of the K-nearest neighbors retrieved by histogram intersection, as illustrated in Figure 15. This improvement demonstrates that image similarities can be better measured by taking into account displacement, an idea that will be used for later applications of SIFT flow.

4.2 Evaluation of the dense scene alignment

After showing some examples of scene alignment, we want to evaluate how well SIFT flow performs compared to human perception of scene alignment. Traditional optical flow is such a well-defined problem that it is straightforward for humans to

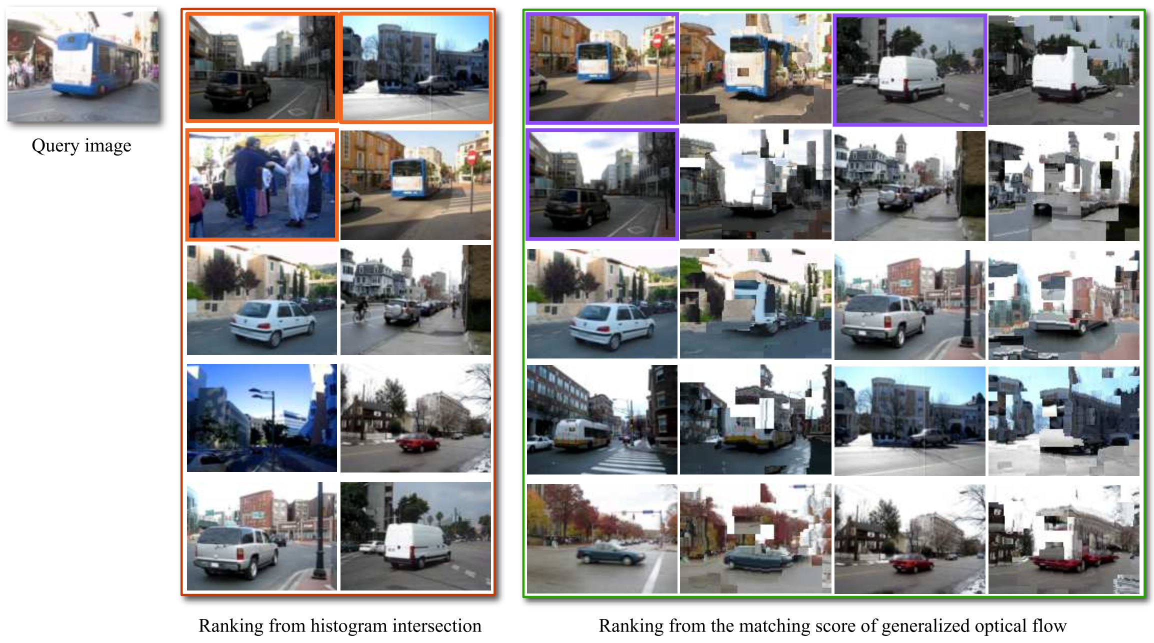

Query image
Ranking from histogram intersection
Ranking from the matching score of generalized optical flow

Fig. 15. **SIFT flow typically improves ranking of the nearest neighbors**. Images enclosed by the red rectangle (middle) are the top 10 nearest neighbors found by histogram intersection, displayed in scan-line order (left to right, top to bottom). The top three nearest neighbors are enclosed by orange. Images enclosed by the green rectangle are the top 10 nearest neighbors ranked by the minimum energy obtained by SIFT flow, and the top three nearest neighbors are enclosed by purple. The warped nearest neighbor image is displayed to the right of the original image. Note how the retrieved images are re-ranked according to the size of the depicted vehicle by matching the size of the bus in the query.

annotate motion for evaluation [34]. In the case of SIFT flow, however, there may not be obvious or unique pixel-to-pixel matching as the two images may contain different objects, or the same object categories with very different instances.

To evaluate the matching obtained by SIFT flow, we performed a user study where we showed 11 users image pairs with 10 preselected sparse points in the first image and asked the users to select the corresponding points in the second image. This process is explained in Figure 9. The corresponding points selected by different users can vary, as shown on the right of Figure 10 . Therefore, we use the following metric to evaluate SIFT flow: for a pixel $\mathbf{p}$, we have several human annotations $\mathbf{z}_i$ as its flow vector, and $\mathbf{w}(\mathbf{p})$ as the estimated SIFT flow vector. We compute $\Pr\big(\exists\mathbf{z}_i, \|\mathbf{z}_i - \mathbf{w}(\mathbf{p})\| \leq r|r\big)$, namely the probability of one human annotated flow is within distance r to SIFT flow $\mathbf{w}(\mathbf{p})$. This function of r is plotted on the left of Fig. 10 (red curve). For comparison, we plot the same probability function (blue curve) for minimum L1-norm SIFT matching, *i.e.* SIFT flow matching without spatial terms. Clearly, SIFT flow matches better to human annotation than minimum L1-norm SIFT matching.

5 Dense Scene Alignment Applications

As illustrated in the previous section, we are able to find dense scene correspondences between two images using SIFT flow, and the correspondence can be semantically meaningful if the two images contain similar objects. In this section, we will introduce two novel applications for dense scene alignment: motion field prediction from a single image, and motion synthesis via transfer of moving objects common in similar scenes.

5.1 Predicting motion fields from a single image

We are interested in predicting motion fiends from a single image, namely to know which pixels could move and how they move. This adds potential temporal motion information onto a singe image for further applications, such as animating a still image and event analysis.

A scene retrieval infrastructure is established to query still images over a database of videos containing common moving objects. The database consists of sequences depicting common events, such as cars driving through a street and kids playing in a park. Each individual frame was stored as a vector of word-quantized SIFT features, as described in Sect. 3.4. In addition, we store the temporal motion field estimated using [12] between every two consecutive frames of each video.

We compare two approaches for predicting the motion field for a query still image. In the first approach, using the SIFT-based histogram matching in Sect. 3.4, we retrieve nearest neighbors (similar video frames) that are roughly spatially aligned with the query, and directly transfer the motion from the nearest neighbors to the query. In the second approach, dense correspondences are estimated between the query and nearest neighbors using SIFT flow, and the temporally estimated motion of the nearest neighbors are warped to the query according to the estimated SIFT flow fields . Figure 16 shows examples of predicted motion fields directly transferred from the top 5 database matches and the warped motion fields.

A still image may have multiple plausible motions: a car can move forward, back up, turn, or remain static. This is handled by retrieving multiple nearest neighbors from the video database. Figure 18 shows an example of 5 motion fields predicted using our video database. All the motions fields are

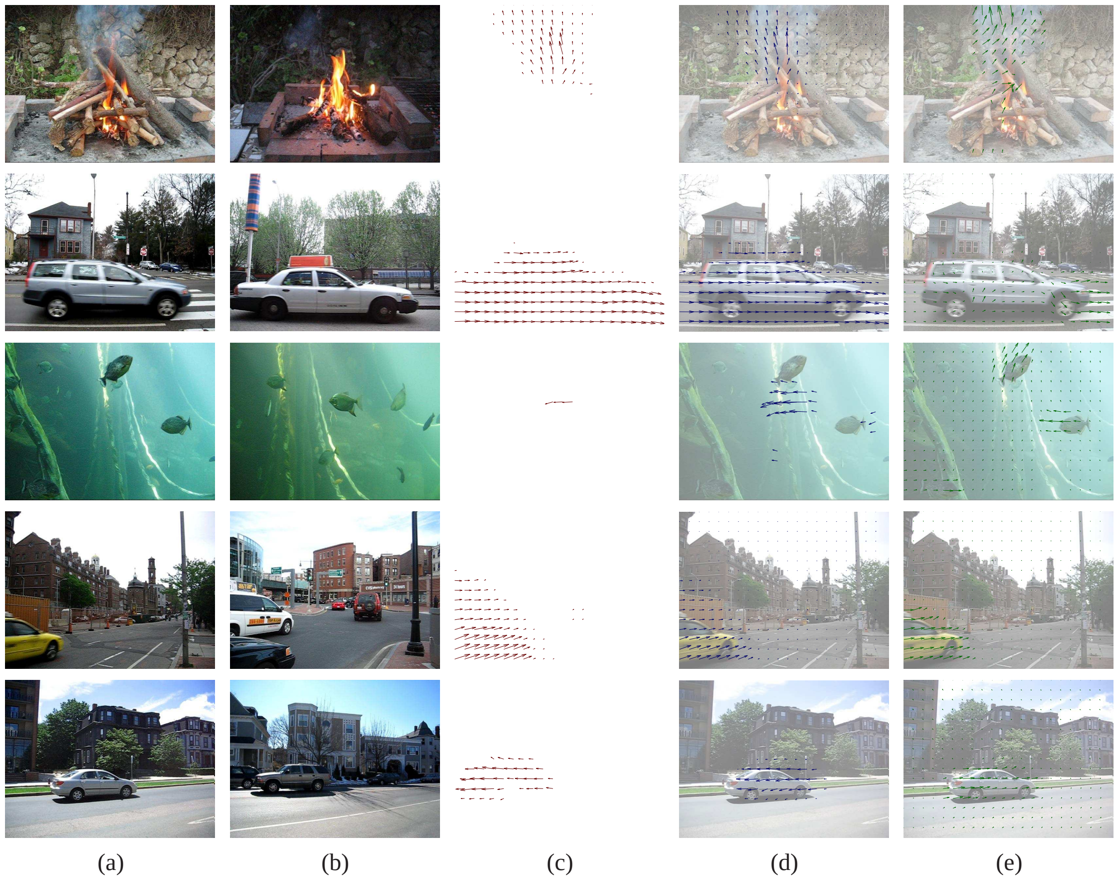


Fig. 16. **Motion from a single image**. (a) Original image; (b) bast match in the video database; (c) temporal motion field of (b); (d) warped motion of (c) and superimposed on (a), according to the estimated SIFT flow; (e) the “ground truth” temporal motion of (a) (estimated from the video containing (a)). The predicted motion is based on the motion present in other videos with image content similar to the query image.

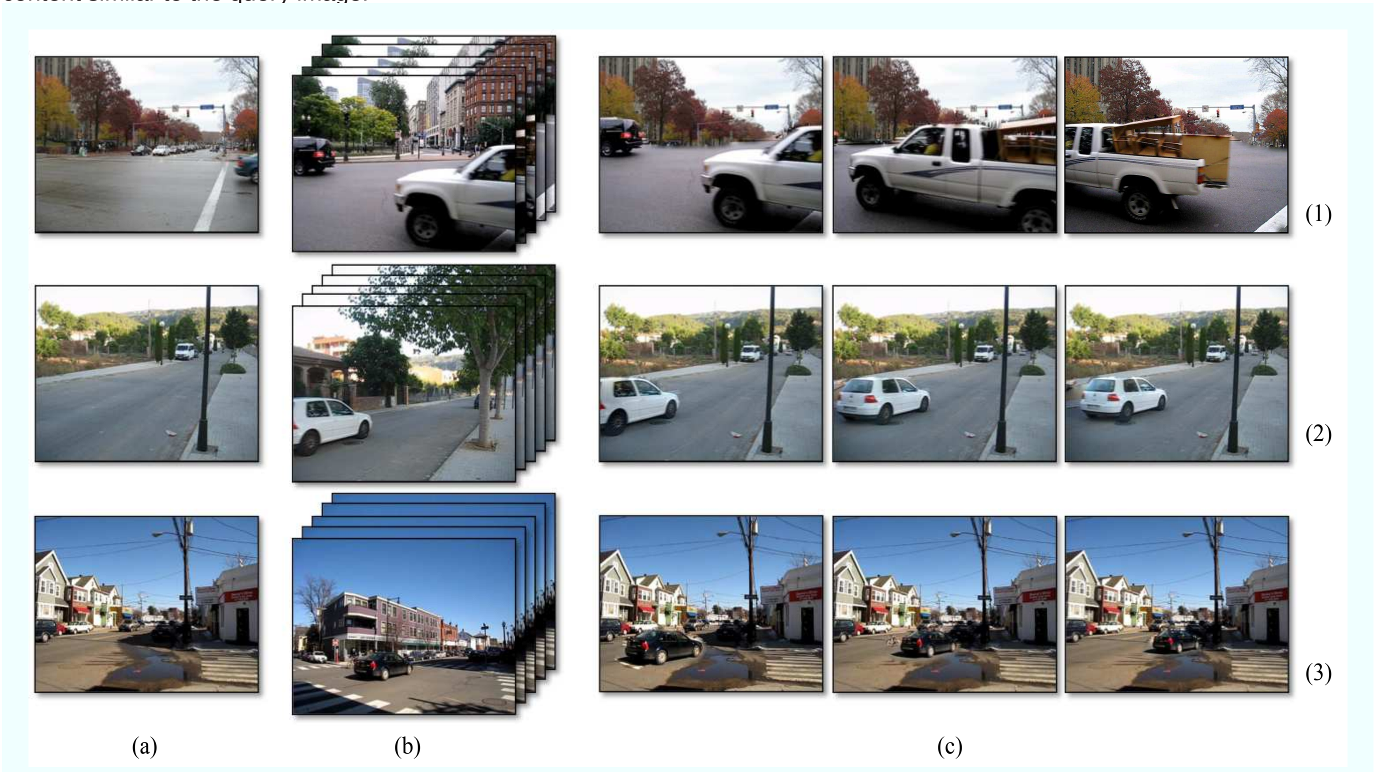


Fig. 17. **Motion synthesis via object transfer.** Query images (a), the top video match (b), and representative frames from the synthesized sequence (c) obtained by transferring the moving objects from the video to the still query image.

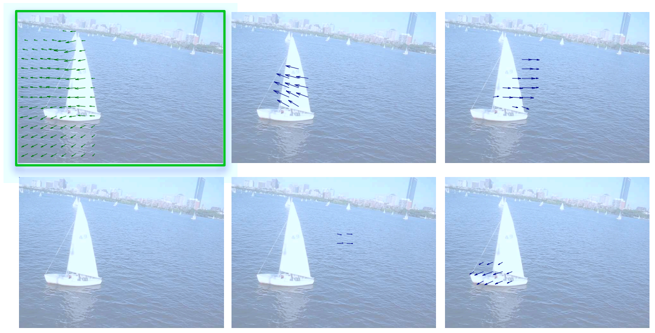


Fig. 18. **Multiple motion field candidates**. A still query image with its temporally estimated motion field (in the green frame) and multiple motion fields predicted by motion transfer from a large video database.

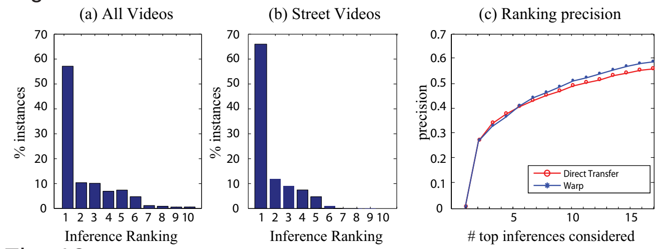


Fig. 19. **Evaluation of motion prediction**. (a) and (b) show normalized histograms of prediction rankings (result set size of 15). (c) shows the ranking precision as a function of the result set size.

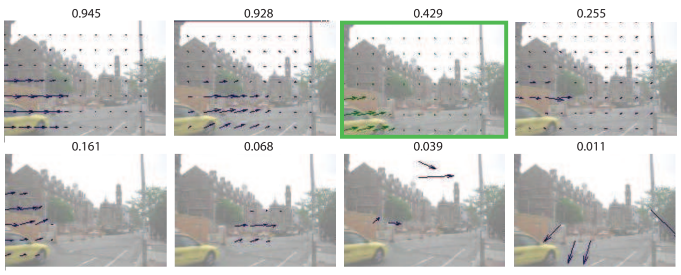


Fig. 20. **Motion instances where the predicted motion was not ranked closest to the ground truth**. A set of random motion fields (blue) together with the predicted motion field (green, ranked 3rd). The number above each image represents the fraction of the pixels that were correctly matched by comparing the motion against the ground truth. In this case, some random motion fields appear closer to the ground truth than our prediction (green). However, our prediction also represents a plausible motion for this scene.

different, but plausible.

5.2 Quantitative evaluation of motion prediction

Due to the inherent ambiguity of multiple plausible motions for a still image, we design the following procedure for quantitative evaluation. For each test video, we randomly select a test frame and obtain a *result* set of top n inferred motion fields using our motion prediction algorithm. Separately, we collect an *evaluation* set containing the temporally estimated motion for the test frame (the closest to a ground truth we have) and 11 random motion fields taken from other scenes in our database, acting as distracters. We take each of the n inferred motion fields from the result set and compute their similarity (defined below) to evaluation set. The rank of the ground truth motion with respect to the motion of random distracters is an indicator of how close the predicted motion is to the true motion estimated from the video sequence. Because there are many possible motions that are still realistic, we perform this comparison with each of the top n motion fields within the result set and keep the highest ranking achieved. This evaluation is repeated ten times with a different randomly selected test frame for each test video, and the median of the rank score across the different trials is reported.

There are a number of ways of comparing temporal motion (optical flow) fields [2], such as average angular error (AAE). For our experiment, we want to compare two motion fields at a coarse level because careful comparisons such as AAE would not be meaningful. In fact, we care more for *which* pixels move than *how* they move. For this evaluation, therefore, we represent each motion field as a regular two dimensional motion grid filled with 1s where there is motion and 0 otherwise. The similarity between two motion fields is defined as

$$S(\mathbf{M},\mathbf{N}) \overset{\text{def}}{=} \sum_{(x,y)\in G} \Big(\mathbf{M}(x,y) = \mathbf{N}(x,y)\Big), \qquad (4)$$

where $\mathbf{M}$ and $\mathbf{N}$ are binary motion fields $\mathbf{M},\mathbf{N} \in \{0,1\}$. Notice that this formula indeed compares the segmentation of motion fields.

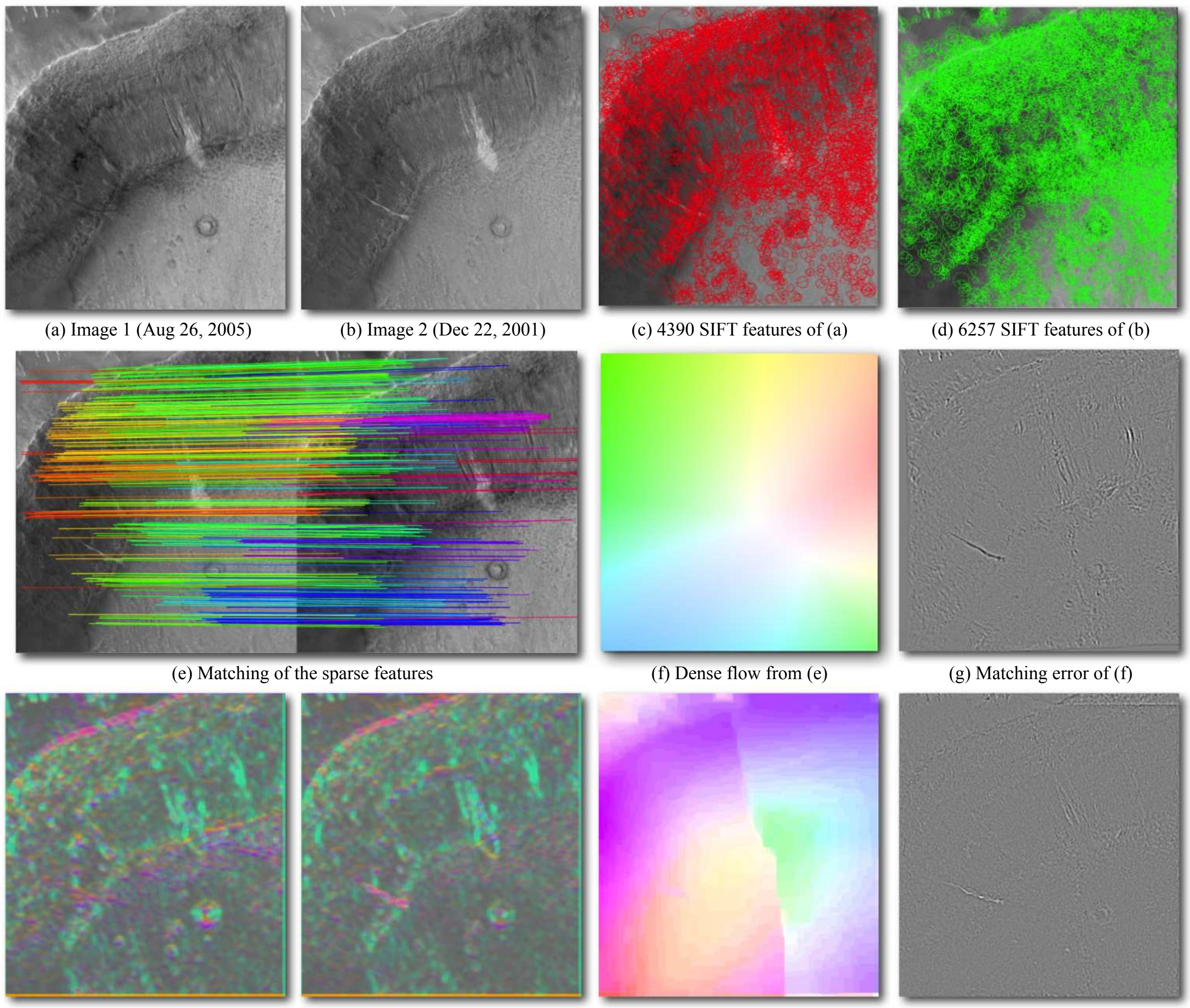
Figure 19 (a) shows the normalized histogram of these rankings across 720 predicted motion fields from our video data set. Figure 19 (b) shows the same evaluation on a subset of the data that includes 400 videos with mostly streets and cars. Notice how, for more than half of the scenes, the inferred motion field is ranked the first suggesting a close match to the temporally-estimated ground truth. Most other test examples are ranked within the top 5. Focusing on roads and cars gives even better results with 66% of test trials ranked 1st and even more test examples ranked within the top 5. Figure 19 (c) shows the precision of the inferred motion (the percentage of test examples with rank 1) as a function of the size of the result set, comparing (i) direct motion field transfer (red circles) and (ii) warped motion field transfer using SIFT flow (blue stars).

While histograms of ranks show that the majority of the inferred motions are ranked 1st, there are still a number of instances with lower rank. Figure 20 shows an example where the inferred motion field is not ranked top despite the reasonable output. Notice that the top ranked distracter fields are indeed quite similar to our prediction, indicating that predicted motion can still be realistic.

5.3 Motion synthesis via object transfer

Besides predicting possible temporal motions for a still image, we want to infer moving objects that can appear in a single image and make a plausible video sequence. For example, a car moving forward can appear in a street scene with an empty road, and a fish may swim in a fish tank scene.

The goal of motion synthesis is to transfer moving objects from similar video scenes to a static image. Given a still image q that is not part of any videos in our database D, we identify and transfer moving objects from videos in D into q as follows:



(a) Image 1 (Aug 26, 2005) (b) Image 2 (Dec 22, 2001) (c) 4390 SIFT features of (a) (d) 6257 SIFT features of (b)

(e) Matching of the sparse features (f) Dense flow from (e) (g) Matching error of (f)

(h) Dense SIFT of image 1 (i) Dense SIFT of image 2 (j) SIFT flow field (k) Matching error of (j)

Fig. 21. **SIFT flow can be applied to aligning satellite images**. The two Mars satellite images (a) and (b) taken four years apart, show different local appearances. The results of sparse feature detection and matching are shown in (c) to (g), whereas the results of SIFT flow are displayed in (h) to (k). The mean absolute error of the sparse feature approach is 0.030, while the mean absolute error of SIFT flow is 0.021, significantly lower. Please refer to http://people.csail.mit.edu/celiu/SIFTflow/NGA/ for the animations showing the warping.

1) Query D using the SIFT-based scene matching algorithm to retrieve the set of closest video frame matches $F = \{f_i | f_i \text{ is the } i\text{th frame from a video in } D\}$ for the query image q.
2) For each frame $f_i \in F$, synthesize a video sequence G in which the ith frame g_i is generated as follows:
    a) Estimate temporal motion field m_i from frame f_i and f_{i+1};
    b) Perform motion segmentation and obtain the mask of moving pixels: $z_i = |m_i| > T$, where T is a threshold;
    c) Treat q as background, f_i as foreground, z_i the mask of foreground, and apply Poisson editing [40] to obtain g_i: $g_i = \text{PoissonBlend}(q,\ f_i,\ z_i)$.

Examples of motion synthesis via object transfer are shown in Figure 17. Given a still query image (a), we use histogram-intersection and SIFT flow to find its retrieved video sequences (b) from our database, and synthesize a new video sequence (some representative frames are shown in (c)) by transferring the moving objects from (b) into the still image (a). Notice the variety of region sizes transferred and the seamless integration of objects into the new scenes. Although it is challenging to estimate the correct size and orientation of the objects introduced to the still image, our framework inherently takes care of these constraints by retrieving sequences that are visually similar to the query image.

6 Experiments on Image Alignment and Face Recognition

We have demonstrated that SIFT flow can be effectively used for retrieval and synthesis purposes. In this section we show that SIFT flow can be applied to traditional image alignment scenarios to handle challenging registration problems.

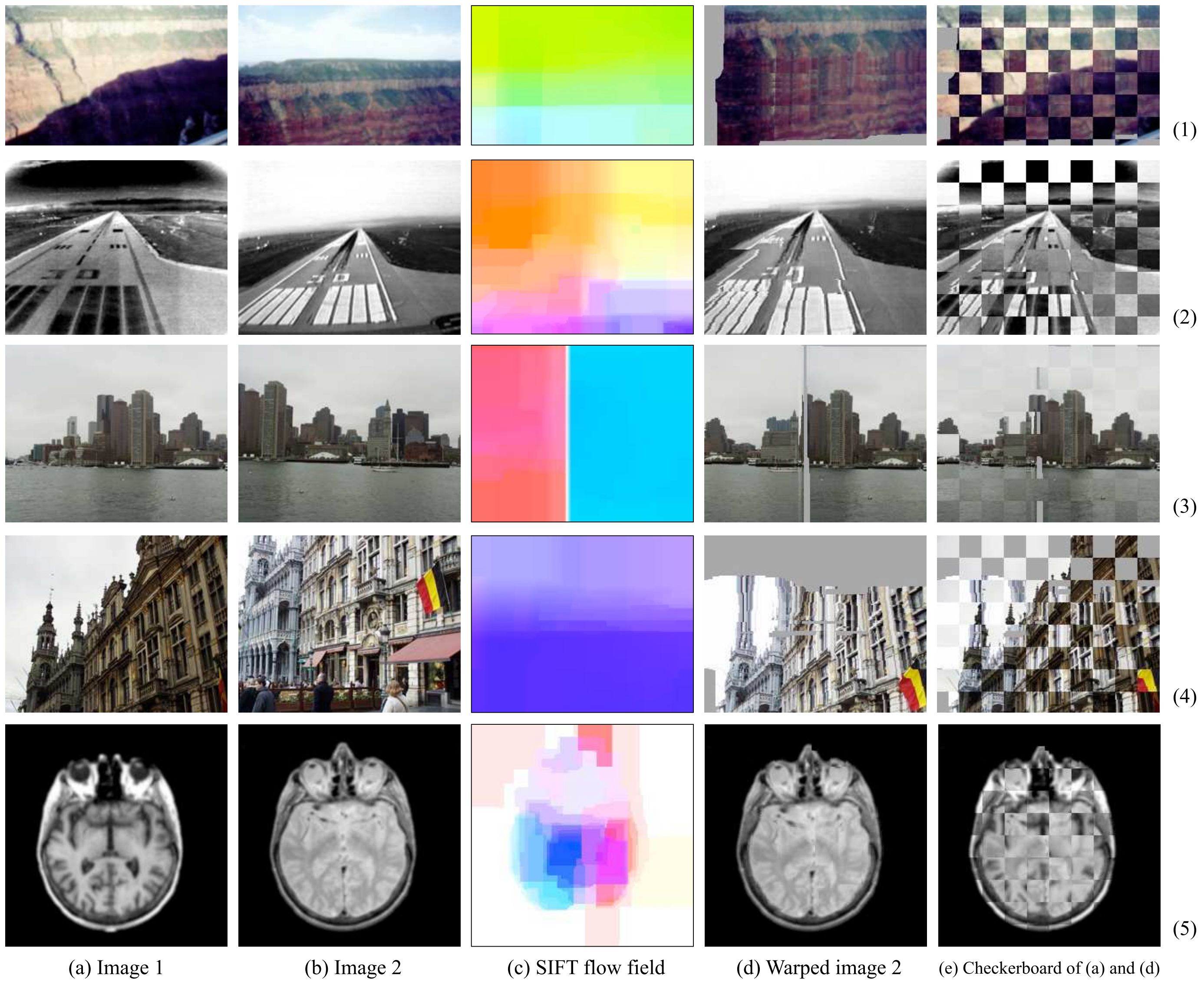


Fig. 22. **SIFT flow can be applied to same-scene image registration but under different lighting and imaging conditions**. Column (a) and (b) are some examples from [58]. Column (c) is the estimated SIFT flow field, (d) is the warped image 2. In (e), we follow [58] to display (a) and (d) in a checkerboard pattern. Even though originally designed for scene alignment, SIFT flow is also able to align these challenging pairs. Please refer to http://people.csail.mit.edu/celiu/SIFTflow/NGA/ for the animations of the warping.

6.1 Same-scene image registration

Image registration of the same 3D scene can be challenging when there is little overlap between two images, or drastic appearance changes due to phenomena such as changes of seasons and variations of imaging conditions (angle, lighting, sensor), geographical deformations, and human activities. Although sparse feature detection and matching has been a standard approach to image registration of the same scene [51], [58], we are curious how SIFT flow would perform for this problem.

Take a look at two satellite images[1] of the same location in Mars, as shown in Figure 21 (a) and (b). Because they were taken four years apart, image intensities vary drastically between the two images. For our experiment, we first use sparse SIFT feature detection [37] to detect SIFT feature points on both images ((c) and (d)), and a sparse correspondence is established through minimum SSD matching on SIFT features (e). This sparse correspondence is further interpolated to form a dense flow field as shown in (f). To investigate the quality of this dense correspondence, we warp image (b) to image (a) according to the dense flow field and display the pixel-wise matching error in (g). The mean absolute error of this correspondence is 0.030 (the pixel value is between 0 and 1). Clearly, the underlying correspondence between these two Mars images are not captured by this sparse correspondence approach.

We now apply SIFT flow to align these two images. The SIFT flow field is displayed in (j), and the pixel-wise matching error of the SIFT flow field is displayed in (k). The mean absolute error decreases to 0.021 for SIFT flow, and visually we can see that misalignment has been significantly reduced. To our surprise, there is a fracture in the estimated SIFT flow

1. Image source: http://www.msss.com/mars_images/moc/2006/12/06/gullies/sirenum_crater/index.html

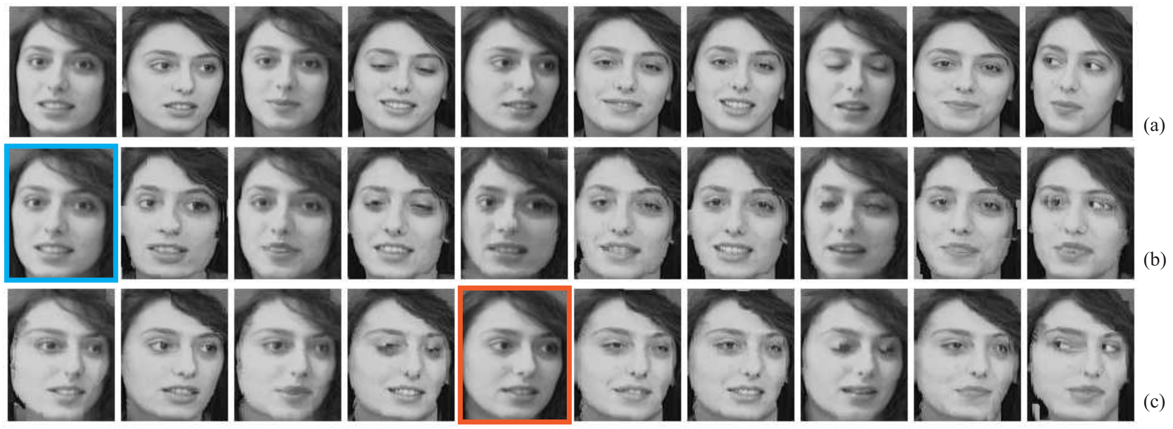


Fig. 23. SIFT flow can account for pose, expression and lighting changes for face recognition. (a): Ten samples of one subject in ORL database [44]. Notice pose and expression variations of these samples. (b): We select the first image as the query, apply SIFT flow to align the rest of the images to the query, and display the warped images with respect to the dense correspondence. The poses and expressions are rectified to that of the query after the warping. (c): The same as (b) except for choosing the fifth sample as the query.

field in (j), which could be caused by some stitching artifact in the satellite images. This is automatically discovered by SIFT flow.

We further apply SIFT flow to align some challenging examples in [58] (the algorithm proposed in [58] is able to handle all these examples well) and the results are displayed in Figure 22, where column (a) and (b) are pairs of images to align. The correspondences between some pairs, *e.g.* rows (1), (3), and (4) are not obvious to human visual systems. The dense correspondences estimated from SIFT flow are displayed in column (c). For visualization purposes, we warp image 2 to image 1 according to the flow field and display the warped image 2 in column (d). To inspect the quality of the flow, we superimpose warped image 2 to image 1 on a checkerboard, as shown in column (e). From these results, the reader can see that SIFT flow is able to handle challenging image registration problems despite drastically different image appearances and large displacement.

6.2 Face recognition

Aligning images with respect to structural image information contributes to building robust visual recognition systems. We design a generic image recognition framework based on SIFT flow and apply it to face recognition, since face recognition can be a challenging problem when there are large pose and lighting variations in large corpora of subjects.

We use the ORL database [44] for this experiment. This database contains a total of 40 subjects and 10 images with some pose and expression variation per subject. In Fig. 23, a female sample is selected as an example to demonstrate how dense registration can deal with pose and expression variations. We first select the first image as the query, apply SIFT flow to align the rest of the images to the query, and display the warped images with respect to the SIFT flow field in Fig. 23 (b). Notice how the poses and expressions of other images are rectified to that of the query. We can also choose a different sample as query and align the rest of the images to this query, as demonstrated in (c). Distances established on images after

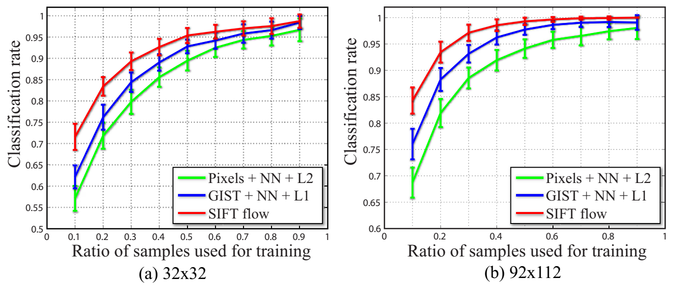


Fig. 24. **SIFT flow is applied for face recognition.** The curves in (a) and (b) are the performance plots for low-res and high-res images in the ORL face database, respectively. SIFT flow significantly boosted the recognition rate especially when there are not enough training samples.

the alignment will be more meaningful than the distances directly on the original images.

In order to compare with the state of the art, we conducted experiments for both original size (92×112) and downsampled (32×32) images. We randomly split a γ ($\gamma\in(0,1)$) portion of the samples for each subject for training, and use the rest $1-\gamma$ portion for testing. For each test image, we first retrieve the top nearest neighbors (maximum 20) from the training database using GIST matching [39], and then apply SIFT flow to find the dense correspondence from the test to each of its nearest neighbors by optimizing the objective function in Eqn. (3). We assign the subject identity associated with the best match, *i.e.* the match with the minimum matching objective, to the test image. In other words, this is a nearest neighbor approach where the SIFT flow score is as the distance metric for object recognition.

The experimental results are shown in Figure 24. We use the nearest neighbor classifier based on pixel-level Euclidian distance (Pixels + NN + L2) and nearest neighbor classifier using the L1-norm distance on GIST [39] features (GIST + NN + L1) as the benchmark. Clearly, GIST features outperform raw pixel values since GIST feature is invariant to lighting changes. SIFT flow further improves the performance as SIFT flow is able to align images across different poses. We observe

Test errors	1 Train	2 Train	3 Train
S-LDA [13]	N/A	17.1 ± 2.7	8.1 ± 1.8
SIFT flow	28.4 ± 3.0	16.6 ± 2.2	8.9 ± 2.1

TABLE 1
Our face recognition system using SIFT flow is comparable with the state of the art [13] when there are only few (one or two) training samples.

that SIFT flow boosts the classification rate significantly especially when there are not enough samples (small γ). We compare the performance of SIFT flow with the state of the art [13], where facial components are explicitly detected and aligned. The results of few training samples are listed in Table 1. Our recognition system based on SIFT flow is comparable with the state of the art when there are very few samples for training.

7 Discussions

I. Scene alignment

We introduced a new concept of image alignment, *scene alignment*, to establish dense correspondences between images across scenes, as illustrated by the examples in Figure 1 (c). The concept of scene alignment advances image alignment from pixel and object levels to a new, scene level. Although seemingly impossible at a first glance, we showed in the paper that dense scene alignment can be obtained by matching in large database using SIFT flow. We also demonstrated through many examples that scene alignment can be a very useful tool to many computer vision problems.

II. Sparse vs. Dense correspondence

There have been two schools of thought for image alignment: dense and sparse correspondence. In the sparse representation, images are summarized as feature points such as Harris corners [27], SIFT [37], and many others [46]. Correspondence is then established by matching these feature points. The algorithms based on the sparse representations are normally efficient, and are able to handle lighting changes and large displacements. In the dense representation, however, correspondence is established at the pixel level in the two images, *e.g.* optical flow field for motion analysis and disparity field for stereo. Because of spatial regularities (*i.e.* the flow vectors of neighboring pixels are similar), estimating flow fields is reduced to optimization in Markov random fields (MRF's). Despite the challenges in optimizing MRF's, via dense correspondence we can easily warp one image to the other, and this warping can be very useful in many applications.

SIFT flow inherits the merits of both the dense representation by obtaining pixel-to-pixel correspondences, and the sparse representation by matching transform-invariant feature of SIFT. In Sect. 4, we demonstrated that SIFT flow is able to align images across scenes, a task that cannot be achieved by traditional optical flow. In Sect. 6.1, we showed that SIFT flow outperforms traditional sparse feature matching

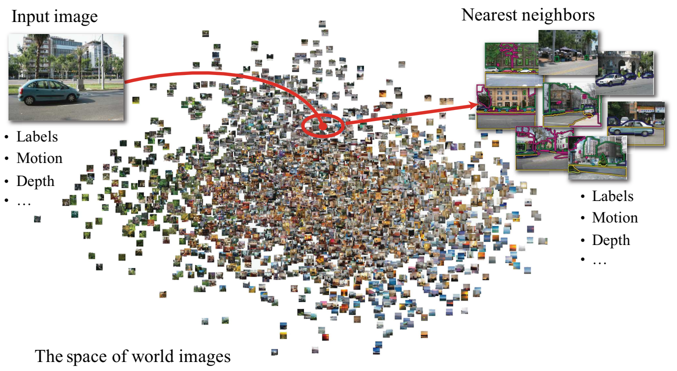


Fig. 25. **An alignment-based large database framework for image analysis and synthesis**. Under this framework, an input image is processed by retrieving its nearest neighbors and transferring their information according to some dense scene correspondence.

in aligning satellite images of the same scene but different appearances. Therefore, SIFT flow becomes a suitable tool for scene alignment.

An important direction for improving SIFT flow is speed. The current system cannot be used for real-time image or video retrieval and matching. One potential approach is the GPU implementation [16] of the BP-S algorithm, which can get up to 50x speedup. However, we feel that there could be essential speedup from the sparse matching. The bottleneck of SIFT flow is the large search window size as the locations of objects may change drastically from one image to the other. The sparse, independent matching provides good, approximate matching for sparse points, and this correspondence can be propagated by abiding by the spatial regularities.

III. SIFT flow vs. optical flow

Although derived from optical flow, SIFT flow is drastically different from optical flow. In SIFT flow, correspondence is built upon pixel-wise SIFT features instead of RGB or gradient that was used in optical flow [11]. We formulated a discrete optimization framework for SIFT flow, whereas often a continuous optimization is incurred for optical flow as sub-pixel precision is required. Even if optical flow is formulated in a discrete manner, the search window size in SIFT flow is much larger than that in optical flow as we want to handle large location changes of objects in SIFT flow.

However, the similarity between SIFT flow and optical flow can be helpful. Inspired by the coarse-to-fine scheme in optical flow, we also designed a coarse-to-fine matching scheme for SIFT flow that improves both speed and quality. Similar to the dense temporal sampling as the foundation for obtaining meaningful optical flow fields, we proposed dense sampling in the space of world images for obtaining potentially semantically meaningful SIFT flow fields, namely correspondences are establishes between objects of the same categories.

Can we apply SIFT flow to analyze temporal motion? On one hand, SIFT flow can be complementary to optical flow. Example (1) and (2) in Figure 11 and the results on satellite image alignment in Sect. 6.1 suggest the possibility. Recently, matching image features such as SIFT has been integrated into the traditional optical flow estimation framework to improve temporal motion estimation [10]. On the other hand, the continuous optical flow model can achieve sub-pixel accuracy, but discrete-matching based SIFT flow can only achieve pixel-level accuracy. Therefore, we do not feel that optical flow can be replaced by SIFT flow.

IV. An alignment-based large database framework for image analysis and synthesis

Using SIFT flow as a tool for scene alignment, we designed an alignment-based large database framework for image analysis and synthesis, as illustrated in Figure 25. For a query image, we retrieve a set of nearest neighbors in the database and transfer information such as motion, geometry and labels from the nearest neighbors to the query. This framework is concretely implemented in motion prediction from a single image (Sect. 5.1), motion synthesis via object transfer (Sect. 5.3) and face recognition (Sect. 6.2). In [35], the same framework was applied for object recognition and scene parsing. Although large-database frameworks have been used before in visual recognition [42] and image editing [28], the dense scene alignment component of our framework allows greater flexibility for information transfer in limited data scenarios.

8 Conclusion

We introduced the concept of dense scene alignment: to estimate the dense correspondence between images across scenes. We proposed SIFT flow to match salient local image structures with spatial regularities, and conjectured that matching in a large database using SIFT flow leads to semantically meaningful correspondences for scene alignment. Extensive experiments verified our theory, showing that SIFT flow is capable of establishing dense scene correspondence despite significant differences in appearances and spatial layouts of matched images. We further proposed an alignment-based large database framework for image analysis and synthesis, where image information is transferred from the nearest neighbors in a large database to a query image according to the dense scene correspondence estimated by SIFT flow. This framework is concretely realized in motion prediction from a single image, motion synthesis via object transfer and face recognition. We also applied SIFT flow to traditional image alignment problems. The preliminary success on these experiments suggested that scene alignment using SIFT flow can be a useful tool for various applications in computer vision and computer graphics.

The SIFT flow code package can be downloaded at http://people.csail.mit.edu/celiu/SIFTflow/.

9 Acknowledgments

Funding for this work was provided by Royal Dutch/Shell Group, NGA NEGI-1582-04-0004, MURI Grant N00014-06-1-0734, NSF Career award IIS 0747120, NSF contract IIS-0413232, a National Defense Science and Engineering Graduate Fellowship, a Xerox fellowship, Foxconn, and gifts from Microsoft and Google.

References

[1] S. Avidan. Ensemble tracking. *IEEE TPAMI*, 29(2):261–271, 2007.

[2] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *Proc. ICCV*, 2007.

[3] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Systems and experiment performance of optical flow techniques. *IJCV*, 12(1):43–77, 1994.

[4] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *NIPS*, 2000.

[5] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE TPAMI*, 24(4):509–522, 2002.

[6] A. Berg, T. Berg., and J. Malik. Shape matching and object recognition using low distortion correspondence. In *CVPR*, 2005.

[7] J. R. Bergen, P. Anandan, K. J Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *ECCV*, pages 237–252, 1992.

[8] M. J. Black and P. Anandan. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, January 1996.

[9] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE TPAMI*, 23(11):1222–1239, 2001.

[10] T. Brox, C. Bregler, and J. Malik. Large displacement optical flow. In *CVPR*, 2009.

[11] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, pages 25–36, 2004.

[12] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/Kanade meets Horn/Schunk: combining local and global optical flow methods. *IJCV*, 61(3):211–231, 2005.

[13] D. Cai, X. He, Y. Hu, J. Han, and T. Huang. Learning a spatially smooth subspace for face recognition. In *CVPR*, 2007.

[14] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Color-and Texture-Based Image Segmentation Using EM and Its Application to Image Querying and Classification. *IEEE TPAMI*, 24(8):1026–1038, 2002.

[15] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *ECCV*, volume 2, pages 484–498, 1998.

[16] N. Cornelis and L. V. Gool. Real-time connectivity constrained depth map computation using programmable graphics hardware. In *CVPR*, pages 1099–1104, 2005.

[17] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[18] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *cvpr*, volume 2, pages 524–531, 2005.

[19] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1), 2005.

[20] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, 70(1):41–54, 2006.

[21] D. J. Fleet, A. D. Jepson, and M. R. M. Jenkin. Phase-based disparity measurement. *Computer Vision, Graphics and Image Processing (CVGIP)*, 53(2):198–210, 1991.

[22] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *IJCV*, 40(1):25–47, 2000.

[23] M. M. Gorkani and R. W. Picard. Texture orientation for sorting photos at a glance. In *ICPR*, volume 1, pages 459–464, 1994.

[24] K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features. In *ICCV*, 2005.

[25] W. E. L. Grimson. Computational experiments with a feature based stereo algorithm. *IEEE TPAMI*, 7(1):17–34, 1985.

[26] M. J. Hannah. *Computer Matching of Areas in Stereo Images*. PhD thesis, Stanford University, 1974.

[27] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[28] J. Hays and A. A Efros. Scene completion using millions of photographs. *ACM SIGGRAPH*, 26(3), 2007.

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

[30] D. G. Jones and J. Malik. A computational framework for determining stereo correspondence from a set of linear spatial filters. In *ECCV*, pages 395–410, 1992.

[31] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *ICCV*, pages 508–515, 2001.

[32] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, volume II, pages 2169–2178, 2006.

[33] C. Liu, W. T. Freeman, and E. H. Adelson. Analysis of contour motions. In *NIPS*, 2006.

[34] C. Liu, W. T. Freeman, E. H. Adelson, and Y. Weiss. Human-assisted motion annotation. In *CVPR*, 2008.
[35] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. In *CVPR*, 2009.
[36] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. SIFT flow: dense correspondence across different scenes. In *ECCV*, 2008.
[37] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, Kerkyra, Greece, 1999.
[38] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.
[39] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.
[40] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM SIGGRAPH*, 22(3):313–318, 2003.
[41] C. Rother, T. Minka, A. Blake, and V. Kolmogorov. Cosegmentation of image pairs by histogram matching - incorporating a global constraint into mrfs. In *CVPR*, volume 1, pages 993–1000, 2006.
[42] B. C. Russell, A. Torralba, C. Liu, R. Fergus, and W. T. Freeman. Object recognition by scene alignment. In *NIPS*, 2007.
[43] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008.
[44] F. Samaria and A. Harter. Parameterization of a stochastic model for human face identification. In *IEEE Workshop on Applications of Computer Vision*, 1994.
[45] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1):7–42, 2002.
[46] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *IJCV*, 37(2):151–172, 2000.
[47] A. Shekhovtsov, I. Kovtun, and V. Hlavac. Efficient MRF deformation model for non-rigid image matching. In *CVPR*, 2007.
[48] J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *ICCV*, 2003.
[49] J. Sun, N. Zheng, , and H. Shum. Stereo matching using belief propagation. *IEEE TPAMI*, 25(7):787–800, 2003.
[50] M. J. Swain and D. H. Ballard. Color indexing. *IJCV*, 7(1), 1991.
[51] R. Szeliski. Image alignment and stiching: A tutorial. *Foundations and Trends in Computer Graphics and Computer Vision*, 2(1), 2006.
[52] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE TPAMI*, 30(6):1068–1080, 2008.
[53] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE TPAMI*, 30(11):1958–1970, 2008.
[54] P. Viola and W. Wells III. Alignment by maximization of mutual information. In *ICCV*, pages 16–23, 1995.
[55] Y. Weiss. Interpreting images by propagating bayesian beliefs. In *NIPS*, pages 908–915, 1997.
[56] Y. Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *CVPR*, pages 520–527, 1997.
[57] J. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *ICCV*, pages 756–763, 2005.
[58] G. Yang, C. V. Stewart, M. Sofka, and C.-L. Tsai. Registration of challenging image pairs: Initialization, estimation, and decision. *IEEE TPAMI*, 29(11):1973–1989, 2007.