Effective Visualization of Sparse Image-to-Image Correspondences

C. Andujar[1], A. Chica[1], M. Comino[1]

[1]VirVIG, Computer Science Department, Universitat Politecnica de Catalunya, Jordi Girona 1-3, Barcelona, Spain

Abstract

Finding robust correspondences between images is a crucial step in photogrammetry applications. The traditional approach to visualize sparse matches between two images is to place them side-by-side and draw link segments connecting pixels with matching features. In this paper we present new visualization techniques for sparse correspondences between image pairs. Key ingredients of our techniques include (a) the clustering of consistent matches, (b) the optimization of the image layout to minimize occlusions due to the super-imposed links, (c) a color mapping to minimize color interference among links (d) a criterion for giving visibility priority to isolated links, (e) the bending of link segments to put apart nearby links, and (f) the use of glyphs to facilitate the identification of matching keypoints. We show that our technique substantially reduces the clutter in the final composite image and thus makes it easier to detect and inspect both inlier and outlier matches. Potential applications include the validation of image pairs in difficult setups and the visual comparison of feature detection / matching algorithms.

CCS Concepts

• ***Computing methodologies*** → ***Reconstruction;*** *Image segmentation;*

1. Introduction

Feature detection and feature matching are essential steps in a number of Computer Vision applications, including photogrammetry, structure-from-motion [SF16], multi-view stereo [SZPF16], image-based localization [LSXK15, NLH*19], content-based image retrieval [ZYT18] and motion field prediction [LYT10].

A number of keypoint detectors and feature descriptors have been proposed, being SIFT, SURF, ORB and CNN-based descriptors the most popular ones (see [KPS17, ZYT18] for recent reviews). Although these descriptors have been designed to handle common image differences, including affine transformations, intensity variations and viewpoint changes, large image differences often cause wrong matches. Challenging image pairs arise e.g. in photogrammetry when reconstructing 3D scenes with occluding objects, moving objects, mirror-reflective surfaces, and self-similar structures. Other applications requiring the alignment of images from different 3D scenes are even more challenging, since the two images to align may contain different object instances, from different viewpoints, and at different locations [LYT10].

State-of-the-art photogrammetry pipelines include tools to visualize feature matches. Traditionally, keypoint matches are shown by placing the two images side-to-side and drawing link segments connecting pixels with matching features (Figure 1). Unfortunately, this approach tends to produce cluttered images where individual matches are hard to identify (Figure 1-left) and potential outliers are not apparent at all. An alternative option, for small viewpoint changes, is to draw these segments not between images but within each image (Figure 1-right). This makes it easy to spot potential outliers, but since one of the endpoints of each segment is not over a true keypoint, the image contents around matching keypoints is not readily available for comparison.

Figure 1: *Traditional feature match visualization (VisualSFM), using a vertical layout. Left: matches are represented as line segments joining matching keypoints. Right: the segments represent the displacement of the matches, within each individual image.*

In this paper we present new techniques for visualizing sparse correspondences between image pairs. We wish the techniques to produce images able to facilitate two user tasks. First, users should be able to identify individual matches, i.e. given an arbitrary keypoint on one image, finding the matching keypoint on the other image. Second, users should be able to see the image contents at each

DOI: 10.2312/evs.20201064

keypoint, as well as the surrounding context. For example, users should be able to check if matches between two building windows refer to the same window instance (same floor, same column). This means that visual overlays should be as little invasive as possible.

Instead of showing all matches at once, we use a *hierarchical clustering* approach to group consistent matches. At the top level, only a few aggregated segments are shown. Although this image already provides a great picture of the matches, we allow the user to selectively explore matches at finer levels by either clicking at a segment or at an image region. Segments representing large clusters (large number of matches) are drawn using thicker lines, and small clusters with thinner lines. Since isolated matches often correspond to outliers, we give visibility priority to small clusters.

We also *optimize the layout* of the two images. Instead of using plain vertical or horizontal layouts, we automatically compute an optimal placement for the images by finding the relative 2D translation that minimizes the impact of overlaid segments.

For images captured from similar viewpoints, most link segments are approximately parallel and thus segments are prone to large overlaps. This hinders the task of visually tracing segments from one keypoint to its matching keypoint. We thus propose different techniques to minimize confusion between neighboring segments, including the use of different colors from a high-contrast palette, and the bending of the segments through quadratic curves. We also uniquely identify keypoints using colored glyphs to further speed up visual tracing and to prevent keypoint mismatch errors.

2. Previous work

We first discuss related work specific to feature matches between image pairs. As stated above, the most common approach is to place the images side-by-side and to draw straight-line segments joining a keypoint on one image with its matching keypoint on the other image (Figure 1-left). This approach is adopted in state-of-the-art photogrammetry applications, e.g. COLMAP [SF16]. This is also the most common approach in feature matching literature, e.g. [DSRO10, MJZ*18].

An alternative approach is to draw segments joining matching keypoint locations within each image [LYT10] instead of between images (Figure 1-right). This is useful for illustrating e.g. motion prediction [LYT10], but not for checking for match correctness nor for finding out an explanation for wrong matches.

Since drawing all matches leads to cluttered images, a few approaches use multiple colors instead of a single one [KPS17], but these colors are assigned randomly and thus some neighboring segments share similar or identical colors [KPS17].

Besides the specific techniques discussed above, the problem at hand can be seen as a particular instance of *graph visualization* [Gov19]. Graph drawing algorithms attempt to optimize certain quality measures such as the total length of the edges and the number of edge crossings [Pur97]. However, most graph layout methods are mostly concerned with the placement of graph nodes [DLM19]. In our case, node positions correspond to specific keypoints within each image and therefore are fixed with respect to the images. This reduces layout options to image transformations (translation, rotation, scaling, shear) affecting each image. Our technique focuses on translations to define the relative placement of one image with respect to the other.

Another common quality measure for graph layout methods is the complexity of edge shapes, to make it easier for the eye to follow them. Such complexity can be measured as the number of bends (polygonal edges) or the number of control points (spline curves). Examples of approaches focusing on changing the course of the edges include [WC07, Hol06, HVW09].

In our setup, a major issue is to minimize the occlusion caused by overlaid edges on the underlying images. This reduces the range of options to straight-line edges or low-curvature spline edges. Edge bundling methods combine geometrically close edges into bundles, which reduce edge clutter [Hol06, HVW09]. Force-directed methods work by attaching spring and electrostatic forces to segmentation points of the edges. Ambiguity in bundles can be avoided if only edges with either a common origin or a common destination are bundled. Bundling approaches use edge angles and lengths as compatibility criteria and thus in our scenario they would require a specific image layout prior to perform the bundling.

A related problem is the generation of origin-destination flow maps [YDJ*18]. These approaches reduce clutter in output maps by following well-known design principles [JSM*17, YDJ*18] such as curving flows, minimizing overlaps among flows and between flows and nodes, and avoiding crossings of nearly parallel flows.

We propose a simpler clustering and bending criteria (just adding one additional control point per edge, and using quadratic Splines) that benefits from the fact that there exists one separating axis (e.g. a horizontal line in a vertical image layout) such that it must be crossed by all edges going from one image to the other.

For further facilitating the task of following edges, the use of highly-contrasted color palettes has been explored for example in the context of transport maps. A related problem is finding a number of distinct colors that can be used to identify the different routes on a map without risk of confusion [GA10]. We benefit from proposed color palettes [Kel65] by a simple approach for assigning different color entries to neighboring edges.

3. Our approach

The input of our algorithm is an arbitrary pair of RGB images A, B along with detected feature matches $\{(x_i, y_i) \in A\}$, $\{(x'_i, y'_i) \in B\}$. Our approach is independent of the feature detection algorithm; for our experiments we used SIFT features. Although the image pair can be arbitrary, in typical applications the images will refer to identical or similar scenes, and will be captured from similar viewpoints. An interactive application shows the two images plus a collection of segments representing either individual matches or aggregated matches.

3.1. Edge clustering

Our edge clustering approach has the same goal as edge bundling techniques, i.e. reduce clutter. A major difference though is that our edge compatibility criterion does not depend on the relative placement of the two images and thus can be performed once before the layout optimization step.

A match is represented by its endpoints $(x_i, y_i) \in A$ and $(x'_i, y'_i) \in B$. For measuring the *distance between matches*, we just compute the Euclidean distance between the points in 4D space $\{(x_i, y_i, x'_i, y'_i)\}$. We thus favor the clustering of edges with both endpoints at nearby locations. Notice that using e.g. edge lengths and edge orientations would require deciding an image layout for A and B beforehand, which we try to avoid.

For clustering the edges, we use a hierarchical clustering approach. The obvious benefit is that it allows for trivially going from an overall map depicting major matches, to finer matches within a cluster (or a region of interest) through an interactive application.

The clustering algorithm proceeds bottom-up: each match starts in its own cluster, and clusters are successively merged together until a desired number of clusters is achieved. We explored three metrics for measuring the compatibility of two clusters C_1, C_2, using resp. the *min*, *avg* or *max* distance between a match in C_1 and a match in C_2. Figure 2 shows the output for these strategies. The best metric to use depends on to which extent we wish potential outliers to be easily recognized by retaining their own cluster. Assuming A and B have similar viewpoints, potential outliers are likely to correspond to 4D points with no nearby matches.

Since inliers matches tend to form large dense regions in 4D space, the *min* metric (also known as single linkage) tends to generate large clusters representing inlier matches, and keeps isolated matches (potential outliers) in their own clusters. This metric is suitable when the major task is the inspection of potential outliers.

In contrast, the *max* metric (complete linkage) keeps clusters from growing too much in extent. For a fixed number of clusters, this means that outlier matches are likely to be merged into other clusters and thus fade out in the final map. This metric is mostly suitable when we wish to neglect outliers. The *avg* metric provides a trade-off between inlier vs outlier attention and thus this is our default option.

In all cases, clusters are represented by segments. Each segment endpoint is computed as the centroid of the keypoints represented by the cluster. The endpoints themselves are represented as circles. The thickness of the segments and the radius of the circles represent the number of matches within the cluster.

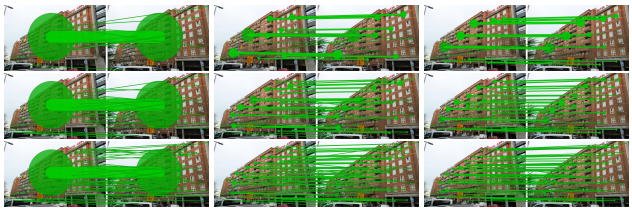

Figure 2: *Clustering to N=12, 25 and 50 clusters (top, middle, bottom), with min, avg, and max metrics (left, middle, right). Compare to Figure 1-left.*

3.2. Image layout optimization

Our problem lies in-between graph layout methods (where nodes are free to move) and origin-destination flow maps (where nodes are fixed). A match with endpoints $(x_i, y_i) \in A$ and $(x'_i, y'_i) \in B$ is fixed only after the layout of the two images has been fixed. This means that we can optimize the relative position of the images. Rotations, scales and shears might hide the true transform relationship between the images and thus are not explored.

Given that we will only allow for translations, we may consider that image A is fixed and image B is translated by a vector $t = (t_x, t_y)$. We use as optimization criterion the sum of squared distances between matches,

$$E(t) = \sum_i \|a_i - b_i - t\|^2$$

We find the vector t that minimizes $E(t)$, subject to the constraint that the translated image B does not overlap A. See supplemental material for full details. Figure 3 shows the optimal layout for different image pairs.

Figure 3: *Optimal layout found for three different image pairs.*

3.3. Color coding

Using different colors for neighboring matches facilitates the visual following of segments at crossings and roughly parallel sections. In the context of image feature matching, we are only aware of trivial approaches assigning random colors to edges [KPS17]. Instead, we use a highly-contrast palette and attempt to assign different entries to neighboring segments. In our experiments we used (a subset of) Kelly's 22 colors of maximum contrast [Kel65].

The assignment of color entries to segments should minimize the probability of neighboring segments to share the same entry. We use a simple algorithm that exploits the fact that all segments must go from one image to the other, and thus must cross either a horizontal axis (vertical layout of A and B) or a vertical axis (horizontal layout). We chose the axis corresponding to the border of A shared with B. We then compute the intersection of all segments with the axis, and sort them according to their intersection point along the axis. Let r_i the rank order of the i-th segment. The color index c_i for such a segment is computed simply as $c_i = r_i \bmod N_c$, where N_c is the number of palette colors to be used.

Figure 4 compares different color mappings. Our simple strategy succeeds in assigning different colors to roughly parallel segments, which are harder to follow (notice that these segments often have different lengths, and thus confusing them might lead to endpoints significantly far apart). Segments crossing at larger angles might be assigned the same color, but these crossings do not pose any visual challenge because of the low edge complexity of our edges (a single straight line or a single quadratic Bezier curve, see below).

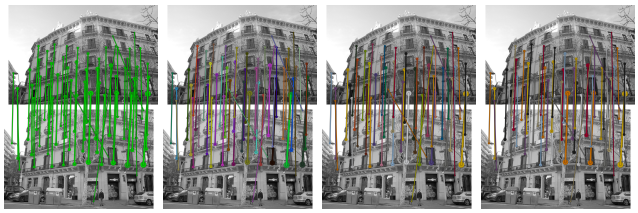

Figure 4: *Different color mappings: single color, random color, and our approach with 22 and first 9 Kelly's colors. These first 9 colors are maximally different for people with defective color vision as well as for people with normal vision [GA10].*

3.4. Segment bending

We let users bend segments to further move nearly-parallel segments apart. This is convenient when A and B have similar viewpoints and many segments are roughly parallel (Figure 5). The bending replaces straight line segments by a quadratic Bezier curve, by adding a single control point. The control point is initially located at the axis-segment intersection. We use the (median-centered) rank order to move the control points towards the image borders, so that the higher the rank, the higher the deviation and thus the bending of the segment. This behavior is similar to that of repulsion-force approaches, which would bend nearly-parallel edges towards the sides of the map.

Figure 5: *Segments drawn with increasing bending factors.*

3.5. Glyph assignment

The use of colored glyphs for labeling a few matching image features is common practice, see e.g. [LYT10]. We also adopt this technique. Since we usually visualize 25-50 segments, we just use lowercase and uppercase letters from the English alphabet as glyphs, which are drawn next to nodes, with the same color as the segment (Figure 6). Since letters are assigned using the rank order, neighboring nodes sharing the same letter are highly unlikely.

Figure 6: *Final output with N=25, 50 and 100 clusters.*

4. Results

We tested our algorithm using Python. Images had 10-25 Mpixels, and 1K-10K matches. Running times on a commodity PC were always below one second, including all steps except image I/O. Segments and glyphs might overlap, so the order in which they are drawn matters. We sort segments by thickness (cluster size) and render first (lowest visibility priority) thicker segments, then thinner segments, and finally all glyphs (highest priority). Due to clustering, node-to-node overlaps are not common and thus glyphs are unlikely to overlap (Figure 6). Figure 7 compares our results with a baseline [SF16] enhanced to include our layout optimization, with green (left) or random color (right) lines. In our output images, aggregated segments are easy to follow, outlier matches are apparent, and the image content is mostly preserved, allowing users to check matches. The baseline approach hides a large part of the image content and, although some main directions are apparent, individual matches can be hardly followed and only a few outlier matches can be distinguished. Compared to traditional feature matching visualization techniques, our approach greatly reduces image clutter and makes it easier for the eye to follow the different paths. *See additional material for further results.*

Figure 7: *Results compared with a baseline approach. Left: wrong matches 'x', 'y' are easy to spot and verify in our output, whereas in the baseline only 'x' is easy to follow. Right: our output shows some outlier matches (e.g. 'P', 'U') that cannot be checked in the baseline images. Random colors facilitate the detection of some outlier directions, but the image content is too occluded to allow for any checking.*

5. Conclusions and future work

We have presented a visualization technique for depicting matches between image pairs. We combine hierarchical clustering, layout optimization, high-contrast color coding, bending and glyphs to provide images much less cluttered than traditional ones. The use of hierarchical clustering allows users to choose the appropriate aggregation level. Potential applications include the validation of matches for challenging pairs (e.g. self-similar objects, different views, different scenes) as well as identifying and understanding outlier matches. Our tool can be used also for a visual comparison of feature detection / feature matching algorithms. As future work, we plan to explore transformations beyond translations, considering image contents to optimize the image layout, and the extension to interactive 3D/immersive applications.

Acknowledgements This work has been partially funded by the Spanish Ministry of Economy and Competitiveness and FEDER Grant TIN2017-88515-C2-1-R.

References

[DLM19] DIDIMO W., LIOTTA G., MONTECCHIANI F.: A survey on graph drawing beyond planarity. *ACM Comput. Surv. 52*, 1 (Feb. 2019). doi:10.1145/3301281. 2

[DSRO10] DRAGON R., SHOAIB M., ROSENHAHN B., OSTERMANN J.: Nf-features–no-feature-features for representing non-textured regions. In *European Conference on Computer Vision* (2010), Springer, pp. 128–141. 2

[GA10] GREEN-ARMYTAGE P.: A colour alphabet and the limits of colour coding. *JAIC-Journal of the International Colour Association 5* (2010). 2, 4

[Gov19] GOVE R.: Force-directed graph layouts by edge sampling. In *2019 IEEE 9th Symposium on Large Data Analysis and Visualization (LDAV)* (Oct 2019), pp. 1–5. doi:10.1109/LDAV48142.2019.8944364. 2

[Hol06] HOLTEN D.: Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on visualization and computer graphics 12*, 5 (2006), 741–748. 2

[HVW09] HOLTEN D., VAN WIJK J. J.: Force-directed edge bundling for graph visualization. In *Computer graphics forum* (2009), vol. 28, Wiley Online Library, pp. 983–990. 2

[JSM*17] JENNY B., STEPHEN D. M., MUEHLENHAUS I., MARSTON B. E., SHARMA R., ZHANG E., JENNY H.: Force-directed layout of origin-destination flow maps. *International Journal of Geographical Information Science 31*, 8 (2017), 1521–1540. 2

[Kel65] KELLY K. L.: Twenty-two colors of maximum contrast. *Color Engineering 3*, 26 (1965), 26–27. 2, 3

[KPS17] KARAMI E., PRASAD S., SHEHATA M.: Image matching using SIFT, SURF, BRIEF and ORB: Performance comparison for distorted images, 2017. arXiv:1710.02726. 1, 2, 3

[LSXK15] LU G., SEBE N., XU C., KAMBHAMETTU C.: Memory efficient large-scale image-based localization. *Multimedia Tools and Applications 74*, 2 (2015), 479–503. 1

[LYT10] LIU C., YUEN J., TORRALBA A.: SIFT flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence 33*, 5 (2010), 978–994. 1, 2, 4

[MJZ*18] MA J., JIANG J., ZHOU H., ZHAO J., GUO X.: Guided locality preserving feature matching for remote sensing image registration. *IEEE Transactions on Geoscience and Remote Sensing 56*, 8 (Aug 2018), 4435–4447. doi:10.1109/TGRS.2018.2820040. 2

[NLH*19] NIU Q., LI M., HE S., GAO C., GARY CHAN S. H., LUO X.: Resource-efficient and automated image-based indoor localization. *ACM Trans. Sen. Netw. 15*, 2 (Feb. 2019). doi:10.1145/3284555. 1

[Pur97] PURCHASE H.: Which aesthetic has the greatest effect on human understanding? In *Graph Drawing* (Berlin, Heidelberg, 1997), DiBattista G., (Ed.), Springer Berlin Heidelberg, pp. 248–261. 2

[SF16] SCHÖNBERGER J. L., FRAHM J.-M.: Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 1, 2, 4

[SZPF16] SCHÖNBERGER J. L., ZHENG E., POLLEFEYS M., FRAHM J.-M.: Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)* (2016). 1

[WC07] WONG N., CARPENDALE S.: Supporting interactive graph exploration using edge plucking. In *Visualization and Data Analysis 2007* (2007), vol. 6495, International Society for Optics and Photonics, p. 649508. 2

[YDJ*18] YANG Y., DWYER T., JENNY B., MARRIOTT K., CORDEIL M., CHEN H.: Origin-destination flow maps in immersive environments. *IEEE transactions on visualization and computer graphics 25*, 1 (2018), 693–703. 2

[ZYT18] ZHENG L., YANG Y., TIAN Q.: SIFT meets CNN: A decade survey of instance retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence 40*, 5 (May 2018), 1224–1244. doi:10.1109/TPAMI.2017.2709749. 1