Supplementary Material for “Match Graph Construction for Large Image Databases”

Kwang In Kim[1], James Tompkin[1,2,3],
Martin Theobald[1], Jan Kautz[2], and Christian Theobalt[1]

[1]Max-Planck-Institut für Informatik, Campus E1 4, 66123 Saarbrücken, Germany
[2]University College London, Malet Place, WC1E 6BT London, UK
[3]Intel Visual Computing Institute, Campus E2 1, 66123 Saarbrücken, Germany

This appendix presents additional discussion on several aspects of the proposed algorithm. Sec. A presents a modification of our algorithm which enables users to reflect local connectivity in link prediction. The remaining sections focus on the label propagation application. Sec. B and C discuss functionalities of active label acquisition and adding new images to the match graph while Sec. D discusses our error correction schemes for label propagation, which rely on an external database. Label propagation in videos is discussed in Sec. E. Finally, Sec. F briefly discuss evaluation of label propagation and future work.

A Density-dependent prediction

The design principle for our incremental graph construction algorithm is to densify the overall graph as quickly as possible. This is equivalent to maximizing the hit ratio of the predicted potential links, which is reflected in our energy functional (Eq. 1 from the main paper restated):

$$\mathcal{O}(F) = \frac{1}{2}\left(\lambda tr[F^\top LF] + \frac{1}{l}\|F - T\|^2_{\mathcal{F}}\right). \tag{1}$$

As discussed in the main paper, this criterion is objective and measurable and facilitates many potential applications. However, in certain applications, it might be desirable to increase the connectivity such that the *density* of the examined nodes in the graph stays roughly equalized. This gives high priorities to potential links which connect two nodes with small degrees (the number of attached examined edges), even though they are not as strongly supported by the context than other links connecting nodes which already have high degrees.

To facilitate this, one can introduce a measure of local connectivity in our algorithm. We regard the *inverse density* or *weight* w_{ij} of a link which joins two nodes I_i and I_j as a function which is inversely proportional to the degrees of I_i and I_j, respectively. This local density information can be reflected in the prediction energy functional e.g., by multiplying w_{ij} with k_{ij} during the construction of the graph Laplacian or, more straightforwardly, by multiplying w_{ij} with the predicted confidence F_{ij}.

Figure 1 illustrates the second alternative with an example. In this example, w_{ij} is defined as:

$$w_{ij} = \left(\frac{(N-1-D_i)^2 + (N-1-D_j))^2}{2N^2}\right)^C, \tag{2}$$

where D_i is the degree of l_i, N is the number of nodes in the graph, and C is a parameter (set at 2) which trades between the hit ratio (which is related to the global connectivity) and the uniformity of local densities. The ground truth link assignments are such that all five nodes are connected to each other. The solid lines indicate examined links which are assigned uniform weights of 1. The dashed lines correspond to the unexamined candidate links where the corresponding numbers show the confidences estimated by minimizing the objective functional (1) with the regularization parameter λ, set at 0.1. The gray lines show candidates with the highest confidences, which indicates that these links are strongly supported by the context. In this example, these links have high density. The magenta lines show the links with the highest estimated confidences when they are weighted. These candidate links are not only the links with the lowest densities: the green link has the same density. However, the magenta links have much stronger support from the context and accordingly their final confidences are higher.

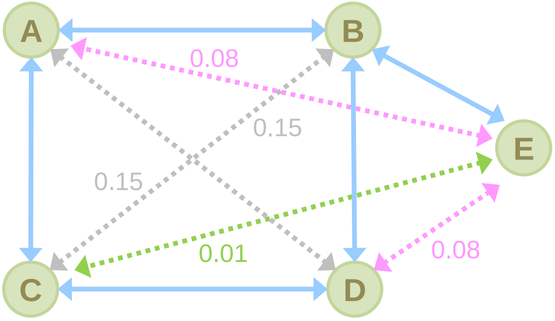


Fig. 1. A toy example of scaling the predicted confidences based on the *density*.

B Active label acquisition

We assume that, initially, labels are provided by users. However, it is beneficial for the system to identify 'interesting' sets of images and *actively* ask users to provide tags for labels. While the definition of interestingness is a non-trivial problem and is a research topic by itself, we use the simple criteria of node degree in the match graph. This implies that we define interesting images as those having the potential to propagate labels to many images.

For each connected component in the match graph, the highest degree node is selected as the label image and presented to the user for tagging. After a label is provided and propagated in the graph, the corresponding nodes and the edges attached to them are removed. This process is repeated until a desired number of interesting images are selected. Figure 2a shows examples of identified interesting images.

C Adding images to the graph

Our graph construction procedure is inherently incremental and, as such, adding a new node a posteriori is simple. For a new image, the filtering phase is followed by the matching phase in which the randomly selected initial matches are augmented based on our confidence measure. The only difference to the batch mode of graph construction (when all images are assessed at once) is that chosen candidates are selected only from those links connected to the new node.

A related system functionality is to match a given image to the database: In applications, one might want to know from where a given photograph is taken or what reviews a given establishment has been given (by looking through the comments attached to the matching images). In principle, this can be implemented with our incremental graph construction procedure; however, this may be computationally too demanding for applications like image search. Instead, we propose matching with only the subset of images in the database which contains *representative views* of labeled objects. For each label, a representative image is selected by considering the number of feature points contained in the corresponding box and the location and size of the box. Specifically, we firstly filter out images in which the number of feature points in the corresponding boxes is below the 60th percentile. From the remaining candidates, the representative view is selected as the maximizer of the cost functional:

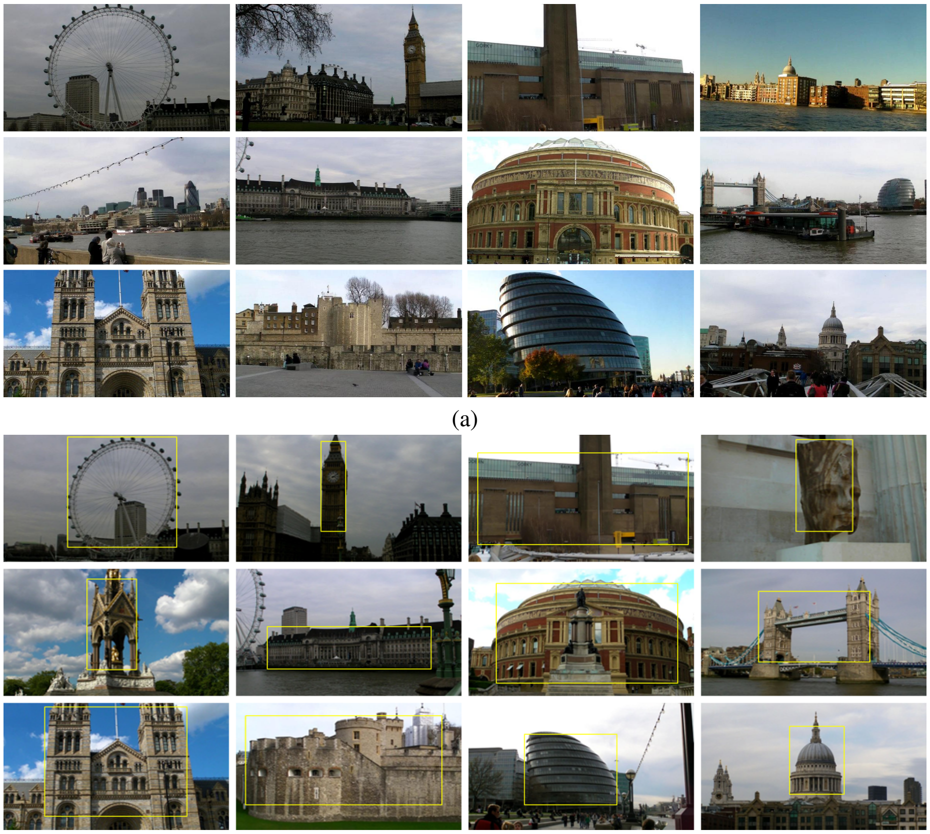
$$|0.6 * S_I - S_B|/\sqrt{S_I} + \|C_I - C_B\|/\sqrt{S_I}, \qquad (3)$$

where S_I and S_B are the size of the image and the corresponding box, respectively, while C_I and C_B are the center of the image and of the box, respectively. This implies that we prefer objects of interest which appears in the center of the image and at a size which is close to 60% of the size of the image. When we seek an image where the size of the object of interest is close to 100% of the image, the resulting representative image is likely to show only a portion of the object. Figure 2b shows examples of representative images. A more advanced method of identifying representative views (which would also benefit from our graph construction procedure) can be found in [1].

D Error correction in label propagation

The results of LP may contain errors, e.g., 'Big Ben' could be mistakenly labeled as the 'Tate Modern'. These errors can be caused either by mistakes in user-provided labels or by incorrect links established during graph construction (see Fig. 3 for an example).

Improving pair-wise image matching so that occurrences of incorrect links are reduced is an interesting problem but is beyond the scope of this paper. In our demonstrated examples, this would involve improving upon standard 3D geometry-based image matching. Furthermore, in general, achieving 100% accuracy with any kind of matching is very difficult. Instead, we focus on removing incorrect links by relying on human operators (or users). The error correction process can be decomposed into two phases: 1) error identification, and 2) error removal. Here, we propose two specific application scenarios.



(a)

(b)

Fig. 2. Examples of automatically found interesting images in our B data set: (a) Images used for active label acquisition. Each image does not necessarily correspond to a representative view of a certain object. (b) Images corresponding to representative views of objects whose labels are marked with the corresponding bounding boxes.

In the *passive scenario*, error identification is performed explicitly by users. When a user finds a label mistake in an image, he or she can report it to the system.[1] An operator then checks the corresponding user-provided label. If the error is caused by the provided label then it is fixed immediately by the operator. Otherwise, the chain of propagated labels is sequentially followed back to its source image and the incorrect link is removed. To facilitate this process, each label contains the index of the original provided label and the parent label.

In the *active scenario*, the system actively generates a certain set of candidate mistakes and asks the operator to investigate. This can be performed either by finding mistakes in the objective labels (e.g., a name of a landmark) or by finding spurious links. For the first case, we use a database containing GPS locations for spatial landmarks that are taken from Wikipedia. The database retains multiple name entries for each landmark such that certain label ambiguities are tolerated (e.g., 'Big Ben' and 'Bigben' are identified as the same). The system then alerts when objects appearing in an image are more than a certain spatial distance apart (0.5km). For the second case, we adopt the link confidence criteria (1) as discussed in the main paper.

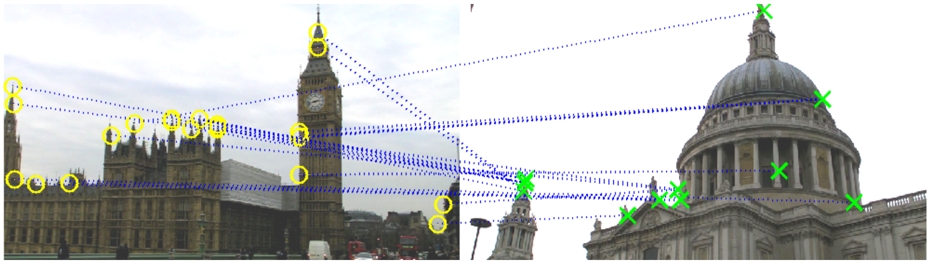


Fig. 3. An example of a mistakenly established link between two images (left and right). Due to a high number of feature point correspondences (20), thresholding cannot be used to rule out this case.

E Label propagation in videos

We can regard a video as a set of images by focusing on static objects. However, the image matching approach discussed in the main paper cannot be directly applied to videos. Even for a small number of videos ($\approx 1,000$), the number of total frames (l') can easily reach billions. Fortunately, videos exhibit a high degree of redundancy which can be exploited to reduce the computational complexity significantly. Instead of propagating labels through all frames in the video database, we pre-select key frames as a representative set $\mathcal{I} = \{I_1, \ldots, I_l\}$. The results of label propagation on the subset $\mathcal{I}$ are then propagated through the entire video database based on simpler feature correspondences

[1] Finding link errors directly is less practical as it requires joint inspection of pairs of images. This is not typical in photo or video browsing interfaces.

(e.g., linear interpolation across intermediary frames, KLT tracking at correspondence points, or fast optical flow techniques).

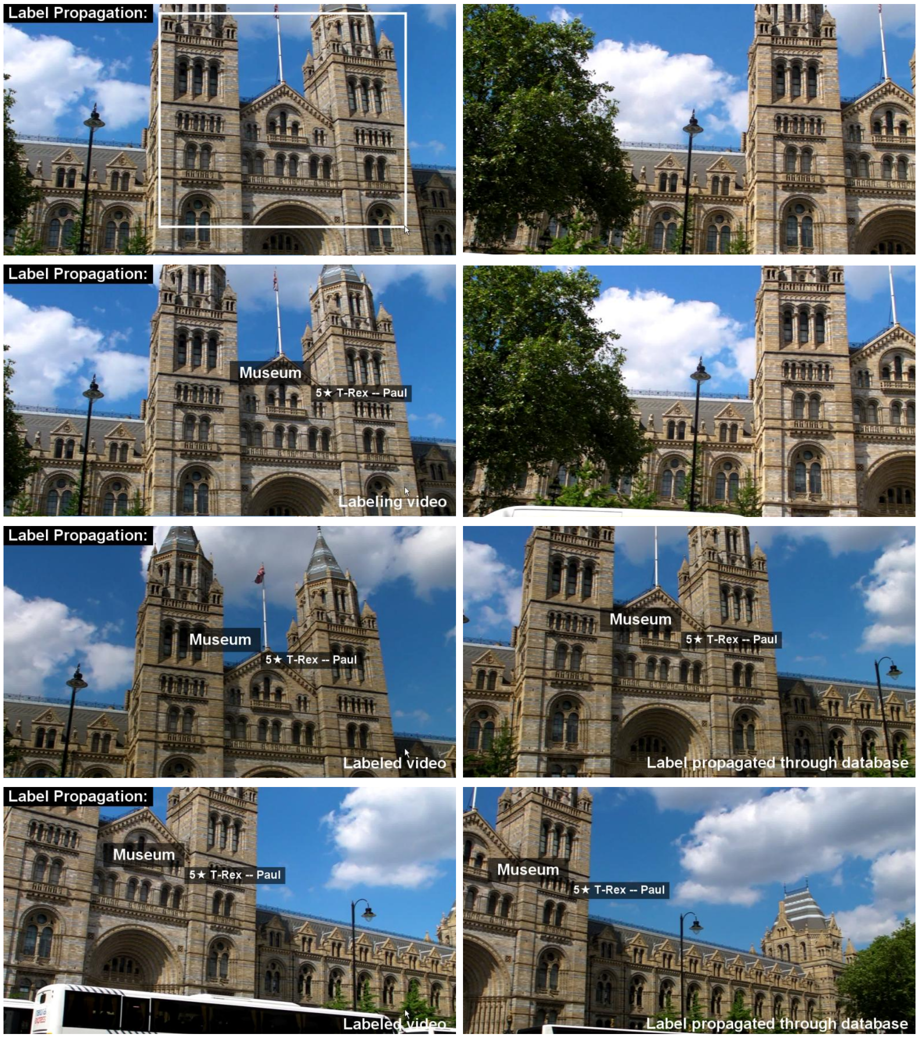
The frequency of key frames needed to provide drift-free interpolation depends on the video content and the specific feature, though in our experience the use of simple bounding boxes to represent object boundaries increases the perceptual tolerance to tracking error. A thorough key frame sampling strategy might perform a pre-process using optical flow to calculate motion-aware key frames, with a conservative strategy being to take a new key frame every time the accumulated flow reaches 25% of the image size – fast GPU optical flow solvers make this at least possible for large video databases. We use the approach of Eisemann *et al.* [2] to compute these flow fields, conservatively pre-process to select good key frames, and finally to track bounding boxes between key frames. In computationally constrained situations, a regular sampling of key frames and linear interpolation still produces acceptable results as the bounding box represents only a figurative object boundary. Figure 4 shows an example of label propagation in videos. Additional results can be found in [3].

F Evaluation and discussion

Systematic evaluation of label propagation is a difficult problem due to the size of the databases involved. We would need to visually evaluate tens or hundreds of thousands of propagated labels to produce statistically meaningful precision results, and to visually evaluate a similar number of images for each label (of which there may be hundreds or thousands) to generate ground truth for recall evaluation.

As such, we performed a smaller precision experiment to suggest the accuracy of our system. We constructed labels for approximately 200 objects for each of our two databases, and propagated these labels to all images. Then, we visually inspected each label. Since the label transfer process is conservative and link error is very rare, the propagated labels were all correct for each of the 200 objects. This is exemplified in Fig. 3 of the main paper and Fig. 2b. In general, LP should not be 100% error free; our supplementary material contains an example of the kind of incorrect link which may be produced by 3D matching. Estimating recall is very difficult even for a few labels since all images need to be inspected. However, in general, we believe that recall is strongly correlated to the connectivity of the corresponding match graph given high precision. As match graph connectivity is directly maximized by our graph construction algorithm, we believe our construction would produce higher recall rates than existing graph construction algorithms. Due to the described difficulty, systematic evaluation is left as future work.

For the application of label propagation, unlike previous approaches, our algorithm is very well suited to large datasets in which the same objects are only visible in a sparse subset of frames. One of the major differences between our algorithm and existing content-based annotation algorithms is that our objective is not to pre-cluster images or to identify important objects (i.e., landmarks). These services can be easily provided with the match graph constructed by our algorithm. Our primary goal is to connect as many images as possible such that any label on any object can be propagated through these connections, not just the most popular ones. In our system, one could propagate



(a) source video (b) target video

Fig. 4. An example of label propagation between videos: the user can create a label for any frame in a source video (a), which is propagated to consecutive frames in the same video through key frames. The label of a key frame in the source video is transferred to individual key frames of other videos (e.g., (b)) using the label propagation algorithm discussed in the main paper. These key frames are then propagated through the remaining frames in each video.

a subjective description label for an unpopular place (e.g., “Latte macchiato in this out-of-the-way cafe is the best!”). This information is retained and can be shared by our system. Our algorithm may not be so useful for fully recovering a large-scale, densely connected structure. However, even in this case, it can quickly identify a sparse subset – this may already be useful to many applications.

Future work should include: a) automatic acquisition of labels, e.g., by using Wikipedia articles [4] or using travel guide articles [5], b) improving bounding box generation for labeled regions, e.g., by refining feature point correspondences similarly to [4], or by using any number of object segmentation algorithms with correspondences as seed points, and c) improving image matching by exploiting *spatial context*, e.g., by extending [6].

References

1. Weyand, T., Leibe, B.: Discovering favorite views of popular places with iconoid shift. In: Proc. ICCV. (to appear)
2. Eisemann, M., De Decker, B., Magnor, M., Bekaert, P., de Aguiar, E., Ahmed, N., Theobalt, C., Sellent, a.: Floating Textures. Computer Graphics Forum **27**(2) (April 2008) 409–418
3. Tompkin, J., Kim, K.I., Kautz, J., Theobalt, C.: Videoscapes: exploring sparse, unstructured video collections. ACM TOG (Proc. SIGGRAPH) (2012) 68:1–12
4. Gammeter, S., Bossard, L., Quack, T., Gool, L.V.: I know what you did last summer: object-level auto-annotation of holiday snaps. In: Proc. ICCV. (2009) 614–621
5. Zheng, Y.T., Zhao, M., Song, Y., Adam, H., Buddemeier, U., Bissacco, A., Brucher, F., Chua, T.S., Neven, H.: Tour the world: building a web-scale landmark recognition engine. In: Proc. IEEE CVPR. (2009) 1085–1092
6. Philbin, J., Sivic, J., Zisserman, A.: Geometric latent Dirichlet allocation on a matching graph for large-scale image datasets. IJCV **95**(2) (2011) 138–153