PhotoSketch: Internet Image Montage

Tao Chen[1] Ming-Ming Cheng[1] Ping Tan[2] Ariel Shamir[3] Shi-Min Hu[1]
[1]TNList, Department of Computer Science and Technology, Tsinghua University
[2]National University of Singapore [3]The Interdisciplinary Center

(a) (b) (c) (d)

Figure 1: A simple freehand sketch is automatically converted into a photo-realistic picture by seamlessly composing multiple images discovered online. The input sketch plus overlaid text labels is shown in (a). A composed picture is shown in (b); (c) shows two further compositions. Discovered online images used during composition are shown in (d).

Abstract

We present a system that composes a realistic picture from a simple freehand sketch annotated with text labels. The composed picture is generated by seamlessly stitching several photographs in agreement with the sketch and text labels; these are found by searching the Internet. Although online image search generates many inappropriate results, our system is able to automatically select suitable photographs to generate a high quality composition, using a filtering scheme to exclude undesirable images. We also provide a novel image blending algorithm to allow seamless image composition. Each blending result is given a numeric score, allowing us to find an optimal combination of discovered images. Experimental results show the method is very successful; we also evaluate our system using the results from two user studies.

1 Introduction

A picture is said to be worth a thousand words. Very often, people compose pictures to convey ideas. A common approach is to sketch a line drawing by hand, which is flexible and intuitive. An informative sketch, however, requires some artistic skill to draw, and line drawings typically have limited realism. An alternative approach known as *photomontage* uses existing photographs to compose a novel image to convey the desired concept. Many commercial photo editing packages, such as Adobe Photoshop and Pixel Image Editor, can seamlessly compose multiple digital images. However, it is up to the user to provide suitable images, and the quality of the final composition depends on the consistency of these images. Composing images with large illumination or texture differences often causes undesirable artifacts. Hence, the main drawback to the use of photomontage is the difficulty of obtaining a set of images suitable for composition. With the prevalence of digital cameras and photo-sharing, billions of images are available online. These online images form an enormous pool for image selection for photomontage.

We propose the combination of sketching and photomontage for realistic image synthesis. Figure 2 provides an overview of our system. The user provides a simple freehand sketch, where each scene item is tagged with a text label. Our goal is to convert this sketch into a photorealistic image. To achieve this, we search online for each scene item, and the background, using the text label. The discovered results are filtered to exclude undesirable images. During filtering, each image is segmented to find scene elements matching items in the sketch. We then optimize the combination of the filtered images to seamlessly compose them, using a novel image blending technique. Several compositions are automatically generated and ranked according to estimated quality. The user can then select among these results and follow up with interactive refinement.

In general, all these stages, including image search, image segmentation and image composition, are well studied, and difficult, problems. Here, we do not claim to solve these challenging problems in general. Rather, we seek an effective solution tailored for our application. The key observation is that certain images are more 'algorithm-friendly' than others, and we rapidly discard any images whose automatic processing is likely to give unreliable results. Firstly, we only retain images with a clear and simple background, which greatly simplifies subsequent image analysis steps. This is achieved by the saliency filtering to filter out images with a cluttered background, for which automatic segmentation would be less reliable. The retained images are typically close-up pictures captured with a large focal length, or with a simple background. Secondly, we consider both content and contour consistency to further discard unsuitable images. While such filtering may cause us to discard perfectly good images, this does not matter due to the wealth of images available on the Internet. Finally, we use a carefully designed algorithm which can cope with large texture and color differences between images being composited. A better composition algorithm enlarges the search space for candidate images and hence provides a better chance of producing a high quality result.

The main contribution of this work is a complete system for semantic image composition whose success relies on two key factors:

stringent filtering of less suitable images, and a carefully designed image composition algorithm.

2 Related work

Segmenting a scene item from a source image and seamlessly pasting it into another target image is a well studied problem in computer graphics. Rother et al. [2004] and Li et al. [2004] used graph-cut based optimization for interactive image segmentation. Wang and Cohen [2007] and Levin et al. [2008] used an alpha matte to accurately segment transparent and partial objects. Segmented objects can be seamlessly inserted into other images by alpha blending. Alpha blending, however, can generate artifacts when different illumination conditions are used to produce these images. To reduce composition artifacts, Pérez et al. [2003] composed images by solving a Poisson equation in the gradient domain. Jia et al. [2006] further improved this method by optimizing the blending boundary. More recently, Farbman et al. [2009] achieved similar composition results without solving an expensive Poisson equation. All these works aim to minimize composition artifacts for given source and target images. In contrast, our work also seeks to *choose* suitable images which *facilitate* composition. During the search for such images, we apply segmentation and blending to various choices of input images, and evaluate the composition quality to find the optimal combination.

Composing images by specifying their content, our work also belongs to the field of *content-aware image synthesis*. Several pioneer works exist in this emerging area. These works apply a whole spectrum of algorithms making use of varying amounts of semantic information during image selection. Hays and Efros [2007] used millions of online photographs to complete the missing parts in an image. Image consistency was checked by low-level scene descriptors. The user cannot control the content of the completed region, since high-level semantic information is not used. Similarly, Eitz et al. [2009] designed a sketch interface for image retrieval and composition from a database of millions of images. Images with similar gradients as the sketches are selected for composition. Their method does not make use of text labels. Hence, relatively more user interaction are required in sketching and image selection. Further, their gradient domain composition contains artifacts. Diakopoulos et al. [2004] and Johnson et al. [2006] specified scene content in some regions of an image or an empty canvas. These regions were filled by images with specified content from an automatically labeled and segmented database. The quality of their results relies on the quality of the database, which in their case often has imprecise segmentation. Lalonde et al. [2007] employed a database with high quality segmentation. They further estimated camera poses and illumination conditions to select physically consistent images for composition. However, building such a database of images with a wide range of illumination conditions and camera poses for a large number of scene items is challenging. From [Hays and Efros 2007] to [Lalonde et al. 2007], increasing amounts of semantic information are used for image selection. Generally speaking, methods employing more semantic information produce better results, as more information is used to select appropriate images. On the other hand, methods using less semantic information potentially allows more suitable candidate images to be found, because image selection is less restricted.

Our method uses more semantic information than [Johnson et al. 2006], but less than [Lalonde et al. 2007]. We compute accurate segmentations, but do not estimate camera poses and illumination conditions. This is because a correct segmentation is critical to avoid incomplete image objects, however, illumination inconsistencies can be tolerated by a suitable image blending method. Hence, our method combines the benefits of both generating a high-quality composition ([Lalonde et al. 2007]) while being less restricted during image selection ([Johnson et al. 2006]). Unlike all previous works, we combine the use of user sketched scene elements *and* text labels during image selection. Element contours are very effective in excluding inappropriate search results, as illustrated later (Table 1). Furthermore, contours provide an intuitive user-interface, and ensures a final composition which agrees well with the user's wishes. Another unique feature of our method is to search for candidate images from the Internet rather than from a large, but still limited, database. The advantage of using online images is that many more images are available. Hence, we can obtain enough 'algorithm-friendly' images to facilitate filtering.

Our work is also related to content-based image search. Smeulders et al. [2000] gave a comprehensive survey of this field. More recent progress can be found in [Fergus et al. 2005]. Among all these works, we highlight two [Jacobs et al. 1995; Rajendran and Chang 2000] that also utilize user drawn sketches for image search. Unlike either of these content-based image search works, we do not seek to achieve accurate understanding of every discovered image. In fact, our problem is much easier, because we only need to keep one image we do understand amongst the search results.

3 User Input

The user sketches on screen to specify a scene. By default, each scene item is represented by an ellipse. The user can drag these ellipses to change the target image layout, and scale them to adjust the sizes of scene items. The user may also optionally draw a shape contour for each item. Each item is also given a text label, which is later used for search. This label could be a noun, such as 'dog' or 'apple', or a noun with an adjective or verb, such as 'dog jump', 'red apple', to target a more careful search. Discovered images are shown to the user immediately, allowing the user to refine the text label and search again if necessary. The background is an empty canvas with a text label. Generally, we require the background to be a landscape image. A typical text label could be 'meadow', 'desert', 'beach', 'mountain', etc. By default, a horizon is placed in the background at the mid-height. The user can adjust the horizon line if desired. Instead of using discovered images, the user may also specify given image components, both for scene items and the background, allowing the user to re-use existing photographs.

4 Candidate Image Selection

Once the sketch is finalized, the system starts to compose the picture by searching for candidate images matching the provided text labels. Web search often generates inappropriate image results. We use a novel filtering scheme to select images amongst these results, giving a small set of candidate images for each scene item and the background, typically 100 for each item and 20 for the background.

4.1 Background image selection

We limit the background to a landscape to make selection easier. Thousands of images can be retrieved using the target text label. We choose candidate images amongst them according to two criteria. First, the image content should be consistent with the query text label. Second, the image should be uncluttered and provide open space in which to perform composition.

Content consistency filtering Our filtering for content consistency is inspired by [Ben-Haim et al. 2006]. Background images with the same content often have similar appearance. For example, beach images often have yellow sand and blue sky; meadow images have green grass. If we cast the discovered images into an appearance feature space, images with similar content typically cluster together. We assume the biggest cluster is formed by images with consistent content, matching the label. In our implementation, we use histograms in LUV color space as image features. Mean shift clustering [Georgescu et al. 2003] is employed to find clusters in feature space. Content consistency is computed as the normalized

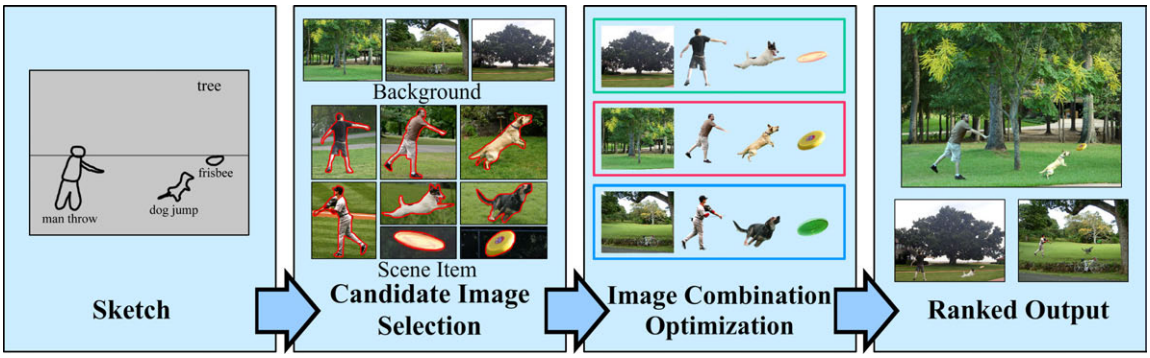


Figure 2: Pipeline. The input is a user-drawn sketch giving a text label for each scene item. We search the Internet for images matching the text labels and select discovered images with contents matching sketched contours; at the same time, each scene item is segmented. Then we optimally combine elements of the candidate images. Several compositions are generated from which the user may make a selection.

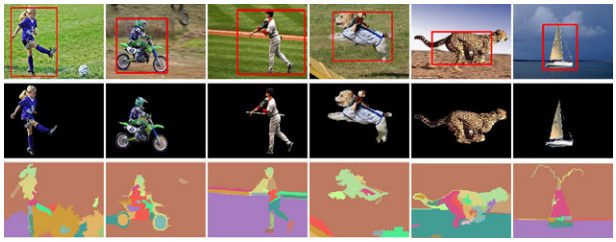


Figure 4: Saliency-based segmentation of scene item images. Top row: scene item images (after saliency filtering), saliency regions marked by a red rectangle. Second row: results of saliency based segmentation. Third row: results using a general image segmentation algorithm [Felzenszwalb and Huttenlocher 2004].

Mahalanobis distance to the largest cluster (the distance is linearly mapped so that the largest and smallest distances are 1 and 0 respectively). Out of 1000 initial images, we choose the 100 images having smallest distance for the next stage.

Uncluttered region filtering The background image should be aligned with the canvas so that the horizon line is in the same position. This alignment is used to determine the position of scene items on different background images. To perform alignment, we estimate a horizon line for each candidate background image using a ground plane computed as in [Saxena et al. 2008]. Images with large differences ($> 30\%$ of image height) in horizon line to the canvas are discarded. From the remaining images we perform further selection, retaining images with large uniform regions which form a suitable background for scene items. We segment each image and count the number of segments covered by the convex hull of all scene items. A lower count corresponds to a more uniform background. Any standard segmentation algorithm may be used; we used the method in [Felzenszwalb and Huttenlocher 2004]. The segment count is normalized to take values in $[0, 1]$. This normalized count is linearly combined using a weight of 0.3 with the content filtering score, the Mahalanobis distance, to rank the retained background images. The top 20 are selected as final candidates. Our background candidate image selection is illustrated in Figure 3 (a).

4.2 Scene item image selection

Scene item images are first filtered to exclude images whose automatic analysis is unreliable. Then both shape and content consistency are checked to further refine the set of selected images.

Saliency filtering Some of the discovered images have a clear simple background. Automatic analysis is much more reliable for such images, so we discard any images with complicated backgrounds. We note that in images with a clear background, the scene item draws clear visual attention to itself. Thus, we compute the high-saliency region for each discovered image. Various saliency detection algorithms exist [Liu et al. 2007] and [Hou and Zhang 2007]. We choose the former for its accuracy, though the latter has better runtime efficiency. As done for background filtering, we segment each image and count the number of segments in a narrow band (of 30 pixels width) surrounding the high-saliency region. If there are more than 10 segments in this band, we consider the image too complicated and discard it.

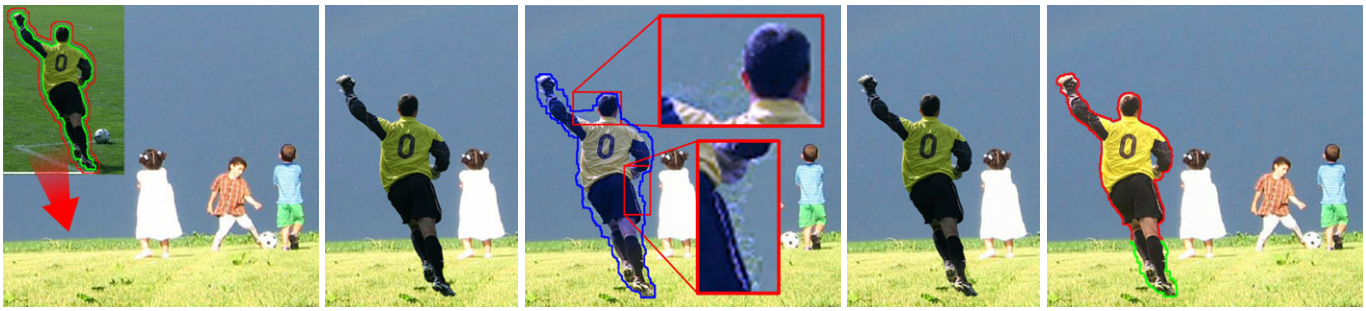
Scene item segmentation In each retained image, we segment out the scene item using the grab-cut algorithm [Rother et al. 2004]. We expand the high-saliency region by morphological dilation and apply grab-cut to this expanded region. Sometimes, this expanded region does not cover the complete scene item. We thus iteratively apply dilation followed by grab-cut until the segmentation boundary does not change or a maximum number of iterations (20) is reached. In Figure 4, the top row shows various scene item images with the salient region marked by red rectangles. The second row indicates the item extracted using saliency-based segmentation. As a comparison, we also show the result of a general segmentation in the third row. Our method often generates better segmentation, which facilitates subsequent filtering. This success is due to the fact that we only process 'algorithm-friendly' images.

Contour consistency filtering If the scene item has a silhouette specified by the user, we further use shape matching technique to filter remaining images. We measure the consistency between the user-drawn contour and the scene item contour (extracted by our saliency-based segmentation). Because the segmentation often generates closed regions, we convert the user drawn outline to closed regions by the morphological close operator. We employ the shape context proposed in [Belongie et al. 2002] to measure the consistency between the two contours. We first sample a set of points at both contours and compute a shape descriptor at each sample point. Given a one-to-one sample point correspondence among the two contours, a score is computed as the summed difference of the shape descriptors at corresponding samples. This score is minimized over all possible point correspondences [1]. The minimum value is regarded as the final contour consistency. This consistency score is normalized to a value between $[0, 1]$, where the most/least consistent image has a score of 0/1. The segmented scene items are ranked by this score, and those ranked below 500 (out of 3000) are discarded.

Content consistency filtering A content consistency filtering approach similar to that used for the background image is applied here. We cast the extracted scene items into a feature space for mean-shift clustering. Instead of relying solely on the largest cluster, we consider all clusters which contain more than 5% of images as having consistent content, because scene items typically have

[1] This minimization could lead to a flip of the searched image, as in the second row of Figure 9.

Figure 3: Filtering of background and scene items. (a) Background filtering. Top to bottom: discovered images for the keyword 'tree', images after content consistency filtering, images after uncluttered region filtering (final background candidates). (b)–(d) Scene item filtering. Top to bottom: discovered images for the keywords 'man throw', 'frisbee' and 'dog jump', images after saliency filtering, images after contour consistency filtering and images after content consistency filtering (final scene item candidates). (e) is the image filtering result for the keyword 'dog' and the same sketch as (d), illustrating usefulness of text labels in search.



(a) (b) (c) (d) (e)

Figure 5: Results using different blending methods. (a) Input images. The red curve indicates the blending region. (b) Simultaneous Matting and Compositing [Wang et al. 2007]; (c) Drag&Drop Pasting [Jia et al. 2006]: the blue curve is the optimized blending boundary; (d) Photo Clip Art [Lalonde et al. 2007]; (e) Our result: the red curve indicates Γ_2 and the green curve Γ_1.

more diverse appearance than backgrounds. Each segmented scene item is assigned a content consistency score, its normalized Mahalanobis distance to its clustering center in the feature space. We linearly combine this score and the contour consistency score to select candidate scene item images. The default combination weight is set to 0.5. The user may adjust this weight to emphasize the contour (for example, we give a weight of 0.8 to contour consistency, when filtering scene items with less consistent color like 'man throw', 'dog jump') or alternatively the appearance. The effect of this filtering is demonstrated in Figure 3 (b)–(e).

4.3 Filtering performance

Filtering images discovered from the Internet is a challenging problem. We do not solve this general problem in our system. Instead, we design an application-specific solution. We apply strict criteria (e.g. having a simple background, and consistent contour and content) to select a small set of candidate images with a low false positive rate. Here we give some statistics of our filtering. We manually check the suitability of images selected for the key word 'dog jump'. Among the first 100 images returned by the search engine, only 35% have a desirable item (a dog with specified contour). In other words, the false positive rate is 65%. This false positive rate becomes 66%, 21% and 15%, after saliency filtering, contour consistency filtering and content consistency filtering in turn. Note how contour consistency filtering is very effective in reducing the false positive ratio. Similar statistics can be found in Table 1 for other scene items.

We wish to highlight two features of our system. Firstly, we note that it is often helpful to include appropriate verbs, e.g. 'throw' and 'jump', to constrain the filtering. As a comparison, we used just the keyword 'dog' with the same sketched jumping dog. The corresponding filtering result is shown in the righthand column of Table 1 and Figure 3(e). After our filtering processes, the false positive rate is still very high (68%). Secondly, as can be seen in Table 1, saliency filtering is not very effective in reducing the false positive rate. However, it is important to the success of our filtering, because it guarantees a good segmentation can be obtained by discarding images with complicated background. This segmentation is used during contour consistency filtering, which can significantly reduce the false positive rate. Thus, saliency filtering serves to select 'algorithm-friendly' data rather than discarding false positive data. We have tried disabling saliency filtering in our experiments, and the false positive rate increases to more than 40% (which is less than 30% when the saliency filtering is enabled). Note that the majority of online images are not 'algorithm-friendly'. A manual check showed that only about 1/3 of 'sheep' images and 1/15 'motorcycle-rider' images have a simple background. However, due to the large number of online images, we can always find sufficient data to proceed.

5 Hybrid Image Blending

With a set of candidate images for each scene item and the background, we optimize the combination of these images into a final picture. In principle, we might choose any existing image blending method and optimize the combination accordingly. However, a simple blending method might not result in a suitable combination. Our novel blending method suited to an optimization-based approach contains two steps. First, we optimize the blending boundary and assign each pixel within the boundary to a set M_1 or M_2, indicating whether the texture and color at that pixel is consistent or not. Second, we compute the blending result by combining improved Poisson blending and alpha blending. Before describing our method, we briefly review existing blending techniques to motivate our method.

Drawbacks of previous methods There are primarily two types of methods for seamless image composition, namely alpha blending and Poisson blending. (A further novel recently introduced blending method [Farbman et al. 2009] has similar effects to Poisson blending but better efficiency). Generally speaking, alpha blending cannot handle illumination changes between images; on the other hand, Poisson blending can suffer from texture or color differences. These problems are exemplified in Figure 5, where the two pictures in (a) are blended. Figure 5 (b) shows the alpha blending result using our implementation of the method in [Wang and Cohen 2007]. The result looks artificial because of illumination inconsistency for different people (the man appears in a darker environment and the children in a brighter light). Figure 5 (c) highlights the 'texture mixing' and discoloration artifacts caused by Poisson blending. As shown in the zoomed regions, 'texture mixing' is caused by pasting the grass next to sky, which has different texture. The discoloration, i.e. the character gets a blue hue, is caused by the large color difference between the sky and the character. Lalonde et al. [2007] reduce discoloration by requiring the blended result to be close to the source image. However, this method suffers from illumination inconsistencies, as exemplified in Figure 5 (d). When illumination conditions in the source and target images differ, requiring the result to be close to the source image will cause illumination inconsistencies similar to those arising in alpha blending, in (b).

5.1 Blending boundary optimization

We first optimize the blending boundary. We apply morphological expansion 20 times to the scene item segmentation contour to obtain an initial blending region Ω_0. The blending boundary is then optimized within Ω_0. This optimization amounts to: 1) decide an optimal blending region $\Omega \subset \Omega_0$; 2) assign each pixel within Ω to either M_1 or M_2. M_1 consists of pixels where texture and color are consistent, and M_2 consists of the other pixels.

Our optimization operates on super-pixels for efficiency. We employ an over-segmentation to break the source and target images into super-pixels. We aim to form an optimal closed chain of super-

	man throw	dog jump	frisbee	sailboat	moto rider	wedding kiss	seagull	sheep	kid ski	dog
IS (%)	83	65	79	71	86	77	72	73	70	97
SF (%)	81	66	80	61	81	78	70	69	67	97
CF1 (%)	30	21	31	35	27	28	29	31	24	77
CF2 (%)	29	15	27	27	24	19	23	20	21	68

Table 1: False positive rate at different stages of filtering. IS: images returned from the internet search. SF: images after saliency filtering. CF1: images after contour consistency filtering. CF2: images after content consistency filtering.

pixels enclosing the scene item. The chain should pass through super-pixels with greater blending suitability, measured by a blending cost computed from the texture and color consistency. Texture consistency is measured by the difference of Gabor feature vectors [Manjunath and Ma 1996] between the source and target images. Color consistency is computed as the summed pixelwise difference of the UV color components. The overall consistency within a super-pixel i is

$$\mathcal{F}_i^p \propto w_1 \frac{\|\Delta G_i\|^2}{\sigma_g^2} + (1-w_1)\frac{\|\Delta U_i\|^2}{\sigma_u^2}.$$

Here, $\Delta G_i, \Delta U_i$ are the Gabor feature difference and summed pixelwise color difference. σ_g, σ_u are the variances of $||\Delta G_i||$, $||\Delta U_i||$ over all super-pixels. w_1 is a combination weight, set as 0.7 in all examples shown in this paper. If this cost is smaller than a threshold T_1 (set to 0.5 in our experiments), we consider Poisson blending is safe at that super-pixel and use $\mathcal{F}_i^p$ as its blending cost. In this case, we also tentatively assign the super-pixel to M_1. Otherwise, the super-pixel is tentatively assigned to M_2, and its blending cost is measured by the feasibility of matting, since matting is computed within it instead (see Sec. 5.2). Matting is difficult in highly textured regions. It is also hard to matte objects having similar color to their neighborhood. Thus, the cost of matting can be measured by

$$\mathcal{F}_i^m \propto w_2 \overline{\|\nabla^2 f_i^s\|} + \frac{1-w_2}{\|\Delta H_i^s\|^2}.$$

Here, $\overline{\|\nabla^2 f_i^s\|}$, indicating the texture complexity, is the average gradient magnitude of the source image in super-pixel i. ΔH_i^s is the L_2 distance between the color histograms of the super-pixel i and of the segmented scene item. It indicates the color similarity between the item and and this super-pixel. w_2 is set to 0.5 in our experiments. The blending cost at a super-pixel i is then defined as

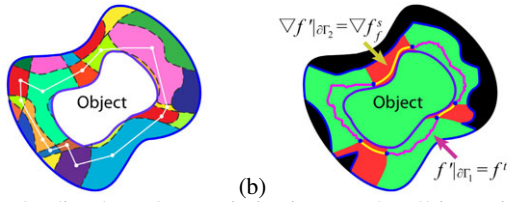
$$\mathcal{F}_i = \begin{cases} \mathcal{F}_i^p & \text{if} \quad \mathcal{F}_i^p \le T_1 \quad \text{(i.e. super-pixels in } M_1\text{)} \\ \mathcal{F}_i^m + T_1 & \text{otherwise.} \end{cases}$$

Now, each super-pixel is associated with a blending cost. For any closed chain Φ, we may compute its overall cost as the weighted sum of the cost of all super-pixels it contains, i.e.

$$C_{chain} = \sum_{i \in \Phi} c_i * \mathcal{F}_i.$$

Here, the weight c_i is the angle super-pixel i spans with respect to the scene item center. We use dynamic programming to compute an optimal chain. To make sure the chain encloses the item, we optimize the chain within the narrow band between the saliency based segmentation contour and $\partial\Omega_0$. This band is illustrated in Figure 6 (a), where cells indicate super-pixels and the white chain indicates the optimized boundary.

After an optimal chain has been computed, we put all enclosed super-pixels into M_1 and exclude outside ones from blending. The assignment to M_1 and M_2 is unchanged for super-pixels on the chain. This assignment is illustrated in Figure 6 (b), where black regions are super-pixels excluded from blending, red indicates super-pixels in M_2, and green indicates those in M_1. A pixel-wise blending boundary is then computed within each super-pixel over the chain. For those super-pixels within M_1, we apply the method described in [Jia et al. 2006] to optimize the blending boundary. For those super-pixels within M_2, the boundary is set at pixels with small alpha matte value, e.g. $\alpha = 0.001$.



(a) (b)

Figure 6: Blending boundary optimization. Each cell in (a) indicates a super-pixel. A chain of super-pixels (shown in white) is computed by minimizing blending cost. Super-pixels are assigned to M_1 (green) or M_2 (red) according to the optimized chain and their texture and color consistency. A pixelwise blending boundary is computed in each super-pixel belonging to the chain.

5.2 Hybrid blending

Now, we perform composition determined by the blending region Ω and M_1, M_2. First, we compute an intermediate result f' by an improved Poisson blending operation in the gradient domain. Second, f' is blended again with the target image by alpha blending.

Improved Poisson blending Conventional Poisson blending can be safely applied to the pixels within M_1. However, it can cause artifacts (e.g. 'texture mixing' and discoloration) within M_2. We thus improve the Poisson solver at pixels within M_2. We apply matting to separate the foreground and background layer of the source image within M_2, and use the foreground layer for blending. At the boundary of M_2, we require the gradient of f' to be equal to that of the matted foreground layer. In summary, the intermediate result f' is computed from

$$\min_{f'} \int_{p \in \Omega} |\nabla f' - v|^2 dp,$$

where

$$v(p) = \begin{cases} \nabla f^s & \text{if} \quad p \in M_1 \\ \nabla f_f^s & \text{if} \quad p \in M_2, \end{cases}$$

with the boundary condition

$$f'|_{\partial\Gamma_1} = f^t \quad \text{and} \quad \nabla f'|_{\partial\Gamma_2} = \nabla f_f^s.$$

Here, f^s, f^t indicate source and target images respectively, f_f^s is the matted foreground layer from the source image, and $\Gamma_i = \partial\Omega \cap M_i, i = 1, 2$, is the blending boundary within M_i.

Alpha blending After f' has been computed, the final blending f is then computed as

$$f(p) = \begin{cases} f' & \text{if} \quad p \in M_1 (\alpha = 1) \\ \alpha f'(p) + (1-\alpha) f^t(p) & \text{if} \quad p \in M_2. \end{cases}$$

α is the alpha matte computed in M_2.

A result from this hybrid method is shown in Figure 5 (e). The blending boundary is overlaid on the image. The red section of the boundary indicates Γ_2, where the gradient $\nabla f'$ is fixed. The green section of the boundary indicates Γ_1, where the value f' is fixed. Our method does not suffer from 'texture mixing' or discoloration, and the pasted character appears under plausible illumination similar to that in the target image.

6 Image Combination Optimization

Even with our hybrid composition method, not al pairs of images can be blended without artifacts. Images with similar texture and color are more suitable for blending. In this section, we optimize

Figure 7: Blending a similar scene with, from left to right, composition costs of 0.2, 0.4, 0.6 and 0.8 respectively. Lower cost is clearly associated with better blending.

the selection of candidate images for composition. We use the minimized cost C_{chain} (see Sec. 5.1) as a measurement of the feasibility of blending two given images. To verify this measurement, Figure 7 shows multiple blended images of a similar scene with different blending cost. From left to right, we show 4 compositions with blending cost 0.2, 0.4, 0.6 and 0.8. It is clear that a smaller cost results in less noticeable blending artifacts. In principle, we wish to check all combinations of candidate images and select the one with minimum cost. Generally, we have 100 candidate images for each item and 20 background candidate images. Exhaustively searching all combinations requires running our blending boundary optimization 20×100^K times, where K is the total number of scene items. Fortunately, scene items often do not overlap in the image and can be optimized independently. This reduces the combination number to $100 \times K \times 20$. We may exhaustively search all these combinations and rank them according to cost. The ten top ranked compositions are displayed for user selection. (We also require each candidate image to appear at most twice among these ten images, to provide some variation in results.)

7 Interactive refinement

Our system automatically composes multiple images ranked by their composition costs. The user then selects a composition amongst them and interactively improves it. The user interaction includes: 1) selection of a composition where all scene items are acceptable; 2) refinement of the automatic segmentation. The first step is necessary, because some automatically composed images may contain incorrect scene items due to errors in image filtering. Furthermore, the saliency based segmentation also sometimes needs improvement. For example, elongated parts of items may be cut off by automatic segmentation. The user can interactively refine the segmentation with methods like those in [Li et al. 2004] or [Rother et al. 2004]. Figure 8 shows all 10 automatic compositions for the example in the first row of Figure 9. Two (shown with a blue frame) have incorrect scene items. Interactive refinements performed in three of them (shown with a red frame) are circled.

8 Experiments and results

We have tested our system with several examples. Using a casually drawn input sketch, our system can composes a variety of photo-realistic pictures. To generate the results in this paper, our system automatically downloads 3000 images from flickr.com, Google.com and Yahoo.com for each scene item, and 1000 images for the background. Among them, 100 and 20 candidate images are selected for each scene item and the background respectively, to reduce the set of images to a manageable size. We start to compute the salient region and perform segmentation while downloading. Computations for different images are processed in parallel. Typically, it takes about 15 mins to process (including image downloading and filtering) each scene item, and about 3–4 mins to process the background. The image combination optimization is also perfromed in parallel, and takes about 1 min. The overall processing time, including downloading and analysis, for generating the results shown in this paper is about $15n + 5$ mins, where n is the number of scene items. All our experiments were performed on two PCs with 2.66 GHz Quad core CPUs and 6 GB RAM.

	Subjects:	A	Pro	B	C
Task I	AS:	2.6	4.8	4.3	4.3
	TT (min):	61	18	64	45
	IT (min):	48+13	12+6	59+5	4.3 + 1.5
Task II	AS:	2.0	4.0	3.0	4.6
	TT (min):	29	22	29	28
	IT (min):	18+11	15+7	25+ 4	2.3 + 1.7

Table 2: Statistics from user study 1. Each column contains data for one group. 'Pro' indicates the professional artist. The interaction time for group A and B includes the time spent on manual image search and on interactive composition. For group C, it includes the time spent on contour sketch and interactive refinement. AS: average score; TT: total time; IT: interaction time.

Composition results Figure 1 shows a wedding picture composed by our system, where (a) shows the input user-drawn sketch and text labels and (b) is the user refined composition. Several selected candidate images are shown for each scene item in (d). All these images have content consistent with the text label and a similar contour to the sketch. However, the illumination conditions and backgrounds for these images vary significantly. The strength of our hybrid blending is that it enables us to compose these challenging images, which ensures a good chance of finding suitable images for composition. Two additional compositions with low blending cost are shown in (c).

Further examples are included in Figure 9. Here we highlight the first row, which contains plasible interaction among scene items. As discussed in Section4, if the scene item has a potentially wide range of shape such as man and dog, it is better to include additional words to constrain the scene item, like 'man throw', 'dog jump'. The additional words can make it harder to draw a suitable contour. In practice, the user can first search with the text label alone, and then draw a contour based on some of the returned images. For this example, all candidate images and the top 10 auto-compositions are provided in Figure 3 and Figure 8. It is interesting to note that combination optimization also helps to exclude some incorrect candidate images. For example, the pigeon image in the candidate background images and the sun image in the candidate frisbee images are both excluded because they cannot be blended well. We give here the number of images containing incorrect scene items among the top 10 auto-compositions. The example in Figure 1 has 4 compositions with incorrect scene items, and from top to bottom, the examples in Figure 9 have 2, 3, 5, 6 and 3 compositions with incorrect scene items respectively.

User study I We designed two user studies to evaluate our system. In the first user study, we tested the efficiency and composition quality of our system. Ten subjects were selected. Nine of them were novices to photomontage (to both our system and Adobe Photoshop). The other was an artist with professional experience of Adobe Photoshop. We split the nine novices into three groups of equal size. Group A were provided with Adobe Photoshop; group B were provided with our hybrid blending tools, using a drag & drop interface for interactive composition; group C were provided with our complete system. The artist was also provided with the Adobe Photoshop. Each group of subjects was given 20 minutes instruction on use of the tools.

The study consisted of two tasks, where the subjects had to generate an image according to a verbal description. In the first task, the subjects were required to generate the best result they could. In the second task, the subjects were given 30 minutes to generate a result. The time spent on searching and composing were recorded separately. Each composition was presented to five evaluators (not se-

Figure 8: The top 10 automatic compositions. Compositions with incorrect scene items have a blue frame. The three compositions chosen for interactive refinement have a red frame; the refinement is circled.

Figure 10: Example composite images from user study I. Compositions generated with our system (average score 4.7) are shown in the left column. Compositions (without our system) of highest (average score 4.4) and lowest (average score 1.6) score are shown in the middle and right column.

lected as subjects) who gave them a subjective score (ranging from 1 to 5, higher score indicating better quality).

Some of the composed images are shown in Figure 10. We summarize the results of the user study in Table 2. We first compared group A and group B to evaluate our hybrid blending. With our novel blending method, group B constantly generated better compositions than group A in both tasks. As indicated by the interaction time (IT), group B also spent less time on image composition. Secondly, we compared group B and the professional artist (Pro) with group C to evaluate the candidate image search and filtering. In task I, they all produced high quality compositions. Group C spent 45 minutes to generate the results, while group B used 64 minutes to acheive similar quality. Within the 45 minutes spent by group C, the actual user interaction time was less than 6 minutes. In comparison, group B spent 59 minutes to find the right images and another 5 minutes for composition. The professional artist achieved a similar quality in 18 minutes. Although the artist spent less overall time than group C, his interaction time was longer. In task II, the blending quality of group C (average score 4.6) is higher than group B (average score 3), because group B could not find suitable images within the alloted time. The professional artist achieved similar quality to group C with 22 minutes of interaction. However, group C needed only 4 minutes of interaction. We also observed from the user studies that our system is quite robust to variations in user input. For example, the sketches for 'dog jump' varied significantly among subjects, but our system always reduces the false positive rate to less than 20%, ensuring a successful composition.

User study II Next, we tested if our system can successfully generate novel scenes. Four subjects were selected, all novices to our system. After 20 minutes of instruction, one of the subjects was

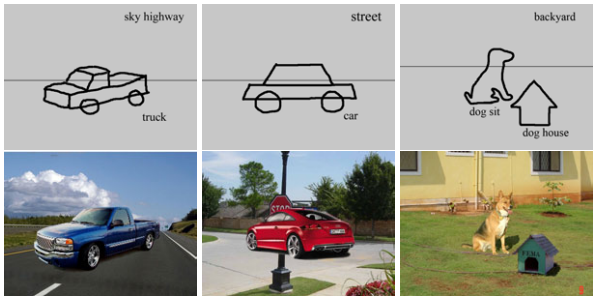


Figure 12: Failures of our system. Left to right: incorrect perspective, incorrect occlusion and incorrect relative scene item size.

asked to provide 15 image composition tasks. The other three subjects used our system to generate these compositions. Each composition was evaluated in the same way as before. Among all the 45 compositions (15 tasks $\times$ 3 subjects) , 35 (77.8%) were considered to be successful (average score ≥ 3). Some of the compositions are shown in Figure 11.

9 Limitations

Content based image synthesis is a challenging problem. Here, we elaborate a few factors that might prevent our system from giving satisfactory results. Firstly, our image filtering is still limited. Background filtering works well for landscape images, but often fails in indoor scenes. Scene item filtering is also limited. Composition may fail if the false positive rate is too high for some scene items. Placing more items in one scene increases the chance of failure. Typically, if 2–3 scene items are included in one scene then 2–6 among the 10 top ranked compositions contain incorrect scene items. The user can adjust the keywords or sketch to improve the results. However, the system cannot automatically recover from a failure. A potential way to improve our image filtering is to adopt the sketch-vs-image descriptor in [Eitz et al. 2009] which match sketched gradient with those in images. However, it will also require a more careful sketching. Secondly, sometimes compositions have significant artifacts due to different scene objects being projected differently—we do not take camera pose into account for image selection. This problem is exemplified on the left in Figure 12. The perspective of the car is different from that of the road. Such failures can be avoided if camera pose is estimated as in [Lalonde et al. 2007]. Thirdly, the composition can contain incorrect occlusion effects between scene items. This happens when the background image contains some thin objects in front, e.g. the lamp post in the middle of Figure 12. Covering them by scene items causes incorrect occlusion. Finally, the relative scales of scene items are manually specified by the user, and can be physically implausible.

(a) (b) (c) (d)

Figure 9: Composing a photo-realistic picture from a casual sketch: (a) user drawn sketch, (b) final output, (c) two additional compositions with low blending cost, (d) images selected to compose these results.

Figure 11: Composition results from user study II. The image with a red frame is a failed example.

For example, in Figure 12 (c), the dog is too large for its kennel.

10 Conclusion

We have proposed a method for generating photo-realistic pictures from a casually drawn sketch with added text labels. There are two key contributions to achieve this goal. First, we use a novel filtering scheme to select images with simple backgrounds to exclude undesirable discovered images. Second, we use a novel hybrid blending approach to provide improved image composition results. The latter also gives a numerical measurement of composition quality allowing automatic selection of optimal image combinations.

Acknowledgements

We thank all the reviewers for their helpful comments. This work was supported by the National Basic Research Project of China (Project Number 2006CB303106), the National High Technology Research and Development Program of China (Project Number 2009AA01Z330) and FDCT, Macau (Project Number 008/2008/A1). Tan Ping is supported by Singapore FRC Grant R-263-000-477-112.

References

BELONGIE, S., MALIK, J., AND PUZICHA, J. 2002. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 4, 509–522.

BEN-HAIM, N., BABENKO, B., AND BELONGIE, S. 2006. Improvingweb-based image search via content based clustering. In *Proc. of CVPR Workshop*.

DIAKOPOULOS, N., ESSA, I., AND JAIN, R. 2004. Content based image synthesis. In *Proc. of International Conference on Image and Video Retrieval (CIVR)*.

EITZ, M., HILDEBRAND, K., BOUBEKEUR, T., AND ALEXA, M. 2009. Photosketch: A sketch based image query and compositing system. In *SIGGRAPH 2009 Talk Program*.

FARBMAN, Z., HOFFER, G., LIPMAN, Y., COHEN-OR, D., AND LISCHINSKI, D. 2009. Coordinates for instant image cloning. *SIGGRAPH 2009*.

FELZENSZWALB, P. F., AND HUTTENLOCHER, D. P. 2004. Efficient graph-based image segmentation. *Int. J. of Comput. Vision 59*, 2, 167–181.

FERGUS, R., FEI-FEI, L., PERONA, P., AND ZISSERMAN, A. 2005. Learning object categories from google's image search. In *Proc. of ICCV*.

GEORGESCU, B., SHIMSHONI, I., AND MEER, P. 2003. Mean shift based clustering in high dimensions: A texture classification example. In *Proc. of ICCV*.

HAYS, J. H., AND EFROS, A. A. 2007. Scene completion using millions of photographs. *SIGGRAPH 2007*.

HOU, X., AND ZHANG, L. 2007. Saliency detection: A spectral residual approach. In *Proc. of CVPR*.

JACOBS, C., FINKELSTEIN, A., AND SALESIN, D. 1995. Fast multiresolution image querying. In *SIGGRAPH 1995*.

JIA, J., SUN, J., TANG, C.-K., AND SHUM, H.-Y. 2006. Drag-and-drop pasting. *SIGGRAPH 2004*.

JOHNSON, M., BROSTOW, G. J., SHOTTON, J., ARANDJELOVIĆ, O., KWATRA, V., AND CIPOLLA, R. 2006. Semantic photo synthesis. *Proc. of Eurographics*.

LALONDE, J.-F., HOIEM, D., EFROS, A. A., ROTHER, C., WINN, J., AND CRIMINISI, A. 2007. Photo clip art. *SIGGRAPH 2007*.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2008. A closed-form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 2, 228–242.

LI, Y., SUN, J., TANG, C.-K., AND SHUM, H.-Y. 2004. Lazy snapping. *SIGGRAPH 2004*.

LIU, T., SUN, J., ZHENG, N.-N., TANG, X., AND SHUM, H.-Y. 2007. Learning to detect a salient object. In *Proc. of CVPR*.

MANJUNATH, B. S., AND MA, W. Y. 1996. Texture features for browsing and retrieval of image data. *IEEE Trans. Pattern Anal. Mach. Intell. 18*, 8, 837–842.

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *SIGGRAPH 2003*.

RAJENDRAN, R., AND CHANG, S. 2000. Image retrieval with sketches and compositions. In *Proc. of International Conference on Multimedia & Expo (ICME)*.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. "grabcut": interactive foreground extraction using iterated graph cuts. *SIGGRAPH2004*.

SAXENA, A., CHUNG, S. H., AND NG, A. Y. 2008. 3-d depth reconstruction from a single still image. *Int. J. of Comput. Vision 76*, 1, 53–69.

SMEULDERS, A., WORRING, M., SANTINI, S., GUPTA, A., AND JAIN, R. 2000. Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell. 22*, 12, 1349–1380.

WANG, J., AND COHEN, M. 2007. Simultaneous matting and compositing. In *Proc. of CVPR*, 1–8.