

Hidden Images

Qiang Tong*
Tsinghua University

Song-Hai Zhang†
Tsinghua University

Shi-Min Hu‡
Tsinghua University

Ralph R. Martin§
Cardiff University

Abstract

A *hidden image* is a form of artistic expression in which one or more secondary objects (or scenes) are hidden within a primary image. Features of the primary image, especially its edges and texture, are used to portray a secondary object. People can recognize both the primary and secondary intent in such pictures, although the time taken to do so depends on the prior experience of the viewer and the strength of the clues. Here, we present a system for creating such images. It relies on the ability of human perception to recognize an object, e.g. a human face, from incomplete edge information within its interior, rather than its outline. Our system detects edges of the object to be hidden, and then finds a place where it can be embedded within the scene, together with a suitable transformation for doing so, by optimizing an energy based on edge differences. Embedding is performed using a modified Poisson blending approach, which strengthens matched edges of the host image using edges of the object being embedded. We show various hidden images generated by our system.

CR Categories: I.4.9 [Image Processing and Computer Vision]: Applications;

Keywords: Hidden Images, Shape Matching, Image Blending, Poisson Blending

1 Introduction

Hidden images are a form of artistic expression, in which one or more secondary objects (or scenes) are hidden within a primary image. When people look at such an image, they readily see the main scene at a first glance. However, by looking more closely at features of the scene, and reinterpreting it, they recognize that another object (or scene) is also hidden in the same image. Such objects require a reinterpretation of the image features. Figure 1 shows two examples of hidden images created by surrealist artists who have experimented with such techniques.

According to the feature integration theory proposed by [Treisman and Gelade 1980], human vision and perception follow two main phases. The first is a largely parallel feature search, and the second, a serial and slow conjunction search [Treisman 1988; Wolfe 1994]. By removing the characteristic features used in feature search, the viewer cannot immediately recognize the embedded object, and instead is forced to use conjunction search to find and understand the embedded object by integrating clues from multiple features [Huang and Pashler 2007; Huang et al. 2007]. Dif-



Figure 1: *Hidden images:* (a) “Dürer in the Forest” by I. Orosz. (b) An eagle with several running horses.

ferent artistic styles of hidden images require a range of skills for their production, including shape recognition, spatial perception, visual coordination, memory, concentration, creativity and imagination [Ball 2008].

People often use prior experience to decipher a complex object from its components—they can recognize familiar objects, especially human faces, just from incomplete *interior* details, even without a bounding contour. For example, viewers imagine a face in Figure 1(a), even though it is not a complete face, as it contains features which look like eyes, a nose and a mouth. For most objects, the contours are strong edges, while interior details tend to be weak edges. This observation can be used as a basis to hide an object within an image: we take into account both weak and strong edges of the object, and find a location at which its edges make a good match with edges of the host image. Strengthening the matched edges within the host image helps the viewer to recognize the object in the final hidden image: the viewer recognizes these weak edges while mentally adding any missing edges (including the outer contour if necessary) to complete the object.

We use these ideas to give an algorithm for generating *hidden images*. It starts by extracting the edges (with their intensities) in the host image and the object to be hidden, and then matches edge intensities to find an appropriate position in the host image at which to embed a transformed version of the object. Embedding is done using a modified Poisson blending approach, by adjusting the strengths of edges in the background image which are a good match with those of the transformed object, to form the hidden image. We demonstrate our system with several examples, showing that it is potentially a useful creative tool for artists.

2 Related Work

Various previous work has already considered how to hide extraneous objects in a background image. [Yoon et al. 2008] applies a stylized line drawing method to both background image and object images, to find suitable places to hide small and simple objects, generating hidden-picture puzzles. This work requires a pre-constructed object database which contains contours of objects, and it cannot work on large, complex objects with rich texture detail. [Mitra et al. 2009] generates emergent images of 3D objects using a synthesis method which enables objects to be easily detected by humans, but which are much more difficult for an automatic algorithm to recognize. [Chu et al. 2010] presents a texture synthesis technique for generating camouflaged images that use object seg-

*e-mail: tong-q04@mails.tsinghua.edu.cn

†e-mail:shz@tsinghua.edu.cn

‡e-mail:shimin@tsinghua.edu.cn

§e-mail:Ralph.Martin@cs.cardiff.ac.uk

ments and their topological relations as clues that humans detect. Texture synthesis techniques produce natural-looking effects in the final result, but require both foreground objects and background images to contain detailed textures. Our paper also embeds objects into a host image, but we use relatively sparse edges of the embedded objects as clues for the viewer, using an approach based on shape matching and Poisson blending.

Shape matching in images broadly uses either brightness-based methods or feature-based methods [Veltkamp and Hagedoorn 2001]; several variants exist of each. Brightness-based methods treat the intensity of each pixel in the image as the feature descriptor, although these values may be affected by factors such as the poses of objects and illumination changes [Cootes et al. 1995; Vetter et al. 1997]. On the other hand, feature-based methods describe the shapes of objects in an image in the spatial domain, using a much smaller number of features such as the well-known SIFT descriptors [Lowe 2003]. Other papers use the relations between contours in an image as shape descriptors [Latecki et al. 2000]; Hausdorff distance is used to measure the distance between a pair of images [Huttenlocher et al. 1993]. The shape context method [Belongie et al. 2002] is translation and scale invariant, and has been widely used in shape matching, especially for character recognition. [Berg et al. 2005] introduced geometric blurring for robust template matching under affine distortion. While all of these feature-based shape matching methods are currently more popular, most of them do not take into account the different importance of different feature points. Here, we not only match locations of edge points, but also their strengths, using this information to derive an energy for shape matching.

Various methods of image blending exist for embedding an image within another, the general aim being to minimize composition artifacts. The classical method for this problem is alpha blending. However, it may generate artifacts when different illumination conditions prevail. [Pérez et al. 2003] instead composes images by solving a Poisson equation in the gradient domain, which accurately handles transparent and partial objects while reducing composition artifacts. [Jia et al. 2006] shows how to optimize the blending boundary to further improve this method. [Farbman et al. 2009] achieves similar composition results without the need to solve an expensive Poisson equation.

Other work has used texture synthesis methods to achieve similar effects to image blending. [Fang and Hart 2004] applies recovered surface normals from one image to another to create an embossed effect. [Ritter et al. 2006] presents a system for texture painting which combines strokes of texture and produces natural-looking boundary effects. [Oliva et al. 2006] present a method for producing a hybrid image by superimposing two images at two different spatial scales, giving two interpretations when the image is viewed at two different distances. [Chu et al. 2010] gives a texture synthesis method for hiding objects in a background image by removing the original subtle texture details of foreground objects, and replacing them by those of the surrounding background. Our method hides objects by strengthening the edges of the host image which are matched with the edges of the objects. We apply a modified Poisson blending operation to depict the object edges as well as to preserve the original details of the host image in the final results. This allow us to hide untextured objects in the host image.

3 Approach

We now present our automatic system to generate a *hidden image*. We define a hidden image as a new image which combines a host image with one or more secondary objects (or scenes); the latter are embedded so as to seem to be natural parts of the host image.

Here, we simply consider embedding a *single object* into the host image—multiple objects can be handled by repeated application of the method. Our approach comprises two steps. The first is a shape matching process, which finds a best match between the host image and a transformed version of the object we want to hide, measured in terms of differences in edge locations and edge intensities. The second performs object embedding, hiding the transformed object into the background, controlled by intensities of matched edges.

The input to our system comprises two images, which are first pre-processed: the background (or host) image, and the image containing the object of interest, which must first be cut out from its background. The user interactively draws a mask Ω to mark out the object. As humans are much more sensitive to changes in the luminance channel than to changes in color difference channels [Wandell 1995], we extract the object’s luminance channel O for downstream operations (i.e. the Y channel of YCrCb color space, where Cr and Cb channels are chrominance components). We do the same for the host image, denoting its luminance channel by H .

We start by extracting edges of H and O . The first step when generating a hidden image is to find a match between H and some transformed version of O , to provide a suitable place to hide the object. We use a multi-resolution approach to solving this problem, based on matching edge intensities of background and object. We find a location for the object at which the edges of the object naturally occur as certain edges of the background. Then we apply Poisson blending to perform object embedding, which incorporates texture details from both H and O and strengthens the matched edges of the object to make it better join in with the original image. Merging the object into the background image is done in luminance space, and the Cr and Cb channels of the original background image are added in to give the final synthesized image. From here on, we only consider luminance in the discussion unless otherwise stated.

4 Shape Matching

Using the edge information of the background image to depict the embedded object requires finding a good match between the background image H and the object O . Typical object matching methods are based on exterior object contours, whereas here we try to find a match by considering weaker interior edges too. Even if the result does not match the exterior contour so well, the interior edges may still provide an acceptable match. If we embed an object based on such a match, viewers still can recognize the object, using their imagination to supply missing edges of the object based on prior experience. The goal is thus to find a transformation T for the object O which minimizes an energy based on the total difference in edge intensities between the transformed object and the background. We use the Sobel operator to extract edges from both images, giving edge intensities from 0 to 255.

The energy \mathbb{E}_T , relating the background image H and the transformed object O_T , depends on the transformation T . Let e^H denote the edge image derived from H , and e_T^O denote the edge image derived from O_T , as shown in Figure 2(a). To provide a little flexibility in edge positions when matching, these edge images are dilated using a simple 3×3 rectangular element, giving E^H from e^H , and E_T^O from e_T^O —see Figure 2(b).

The total energy \mathbb{E}_T measuring the difference between these dilated edge images is defined as follows:

$$\mathbb{E}_T = \frac{1}{S(\Omega)} \sum_{p \in \Omega} \mathbb{E}_T(p), \quad (1)$$

where $S(\Omega)$ denotes the area of the mask and the matching energy



Figure 2: Left: affine transformed edges of the object. Right: after dilation.

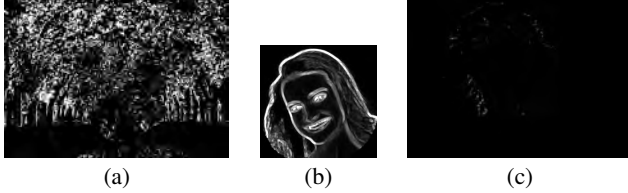


Figure 3: Optimizing matches of edge images to give a combined edge image: (a) edges of dilated host image, (b) edges of dilated transformed object, (c) matching energy — dark means low energy,

at pixel p is

$$\mathbb{E}_T(p) = \frac{|E^H(p) - E_T^O(q)|}{E^H(p) + E_T^O(q)} \frac{255}{\max_p(E^H(p), E_T^O(q))} \sin(\gamma). \quad (2)$$

Here $q \in E_T^O$ is the object pixel corresponding to $p \in E^H$, and $E^H(p)$ and $E_T^O(q)$ are edge strengths of pixels in each dilated edge image. The angle between the edge directions at p in E^H and q in E_T^O is γ .

To allow us to flexibly hide the object, we let the homogeneous transformation T be a combination of translation, rotation and scaling:

$$T = \begin{pmatrix} s \cos \theta & -s \sin \theta & \Delta x \\ s \sin \theta & s \cos \theta & \Delta y \\ 0 & 0 & 1 \end{pmatrix},$$

where s is a scale factor, θ is a rotation angle, and Δx and Δy represent a translation. By considering a set of discrete values for s , θ , Δx and Δy , we find $\mathbb{E}_{\min} = \min(\mathbb{E}_T)$ to determine the best match between E^H and E_T^O . We record the best matching dilated object edge image as E_{best}^O .

Searching this four-dimensional parameter space is potentially costly, but the search can be restricted to a certain extent. For example, when embedding a human face, viewers are unlikely to recognize it if it is rotated by a large angle from vertical, so we restrict θ to lie in the range -30° to 30° , and we also use a coarse step size of 15° . Furthermore, if the object is too small, it will be hard to see in the background, while if it is too big, it may also go unnoticed, so we also restrict the range of s according to the relative scale of H and O , and typically use a step size of 0.2. (Actual ranges and step sizes used in practice are shown later in Table 1). Iterating Δx and Δy pixel by pixel across the background while keeping the transformed object within the host image would be very costly. Instead, we apply a multi-level strategy to speed up the search. Setting E_T^O as $E_{T,0}^O$ and E^H as E_0^H , we construct edge image pyramids $E_{T,0}^O, E_{T,1}^O, \dots, E_{T,i}^O$ and $E_0^H, E_1^H, \dots, E_i^H$, where $E_{T,i}^O$ is an upsampling of $E_{T,i-1}^O$ by a factor of 2, and similarly for E_i^H .

Matching is first done at the coarsest scales, and propagated down to the finer scales—see Algorithm 1. Multiple candidates are passed between levels to avoid premature convergence to a local minimum. Figure 3 shows a matching result; Figure 3(c) illustrates the energy map of the best match between the background and transformed object.

Algorithm 1 Multi-level shape matching

```

for all  $s \in \text{range}(s)$  and  $\theta \in \text{range}(\theta)$  do
   $T \leftarrow \text{ConstructTransformMatrix}(s, \theta)$ 
   $E_T^O \leftarrow \text{Transform}(E^O, T)$ 
  for level =  $l : 0 : -1$  do
    if level =  $l$  then
       $\text{prevPos} \leftarrow E_l^H$ 
    else
       $\text{prevPos} \leftarrow \text{Downsample}(\text{pos})$ 
    end if
     $\text{Clear}(\text{pos})$ 
    for every  $p \in E_{\text{level}}^H \cap \text{prevPos}$  do
       $\mathbb{E}_T^p \leftarrow \text{ComputeDiffEnergy}(E^H, E_{T,\text{level}}^L)$ 
      if  $\mathbb{E}_T^p$  is one of the least 20 for the current level then
        Save  $p$  in  $\text{pos}$ 
      end if
    end for
  end for
   $\mathbb{E}_T \leftarrow \mathbb{E}_T^{\text{pos}}$ 
  if  $\mathbb{E}_T < \mathbb{E}_{\min}$  then
     $\mathbb{E}_{\min} \leftarrow \mathbb{E}_T$ 
     $T_{\text{best}} \leftarrow T$ 
  end if
   $(s, \theta) \leftarrow \text{Next}(s, \theta)$ 
end for
 $E_{\text{best}}^O \leftarrow E_{T_{\text{best}}}^O$ 

return  $E_{\text{best}}^O$ 

```

5 Object Embedding

After finding the best position and orientation at which to hide the embedded object within the host image, we now use a Poisson blending process to produce the final hidden image. If straightforward Poisson blending were used, we would solve Equation 3 to find new luminance values $L(p)$ for pixels inside the mask Ω of O_T :

$$|N_p|L(p) - \sum_{q \in N_p \cap \Omega} L(q) = \sum_{q \in N_p \cap \partial\Omega} H(q) + \sum_{q \in N_p} v_{pq}, \quad (3)$$

where N_p are the neighboring pixels for pixel p , and $|N_p|$ is the number of neighbors. $\partial\Omega$ is the boundary of Ω , and $v_{pq} = O_T(p) - O_T(q)$ is the gradient from O_T .

However, traditional Poisson blending is not directly suitable for our problem. As it uses v_{pq} to control the gradient value in the mask, the blended image would have the same gradient as the object image. If the host image contained rich texture details and the object contained little texture detail, traditional Poisson blending method would produce an image with edges of the embedded object inside the mask only. Furthermore, shape matching finds the best *overall* match between host image and object based on edge intensities, but the object embedding process needs to ignore any badly-matched strong edges of the object image and enhance the matched edges, as after blending, these would break up the background image in a very visible way.

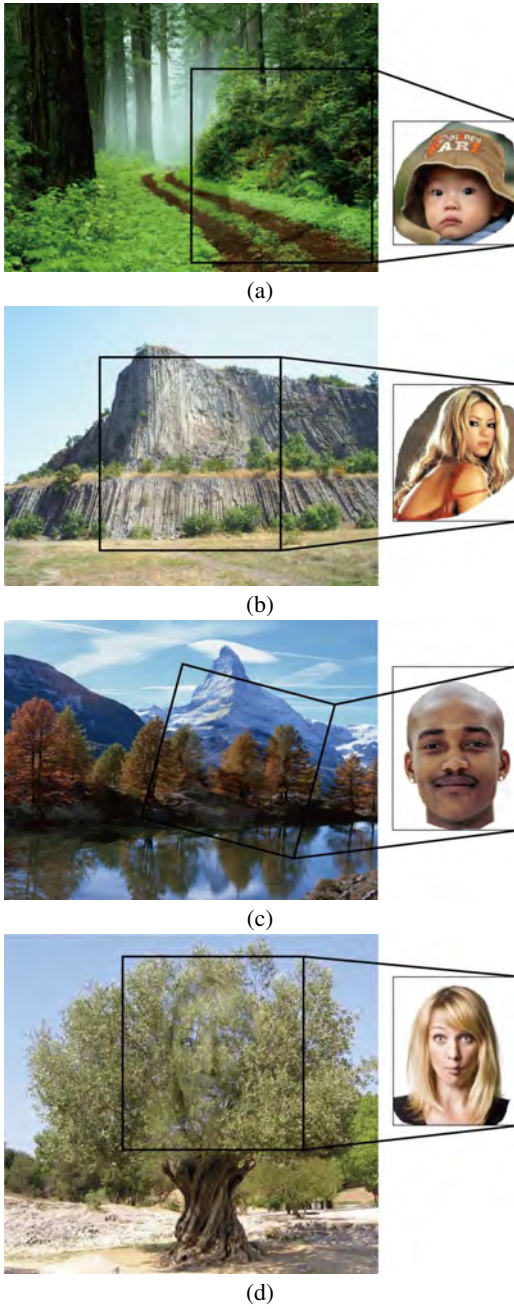


Figure 4: Some results. Left: hidden images created by our system. Right: the embedded objects.

To resolve the issues above, we apply a modified form of Poisson blending to perform object embedding. This is done by considering gradients of the host image H , the object O_T and object edges E_T to determine v'_{pq} in the Poisson equation, as below.

$$v'_{pq} = (1 - \alpha)(H(p) - H(q)) + \alpha\beta(O_T(p) - O_T(q)) + \alpha(1 - \beta)(E_T(p) - E_T(q)) \quad (4)$$

where α in the range $[0, 1]$ is used to control the degree of hiding of the embedded object; low α means it is harder to detect the embedded object. If $\alpha = 0$, the object would completely disappear; if $\alpha = 1$, the texture details of the host image would be replaced by those of the object. β in the range $[0, 1]$ controls relative impor-

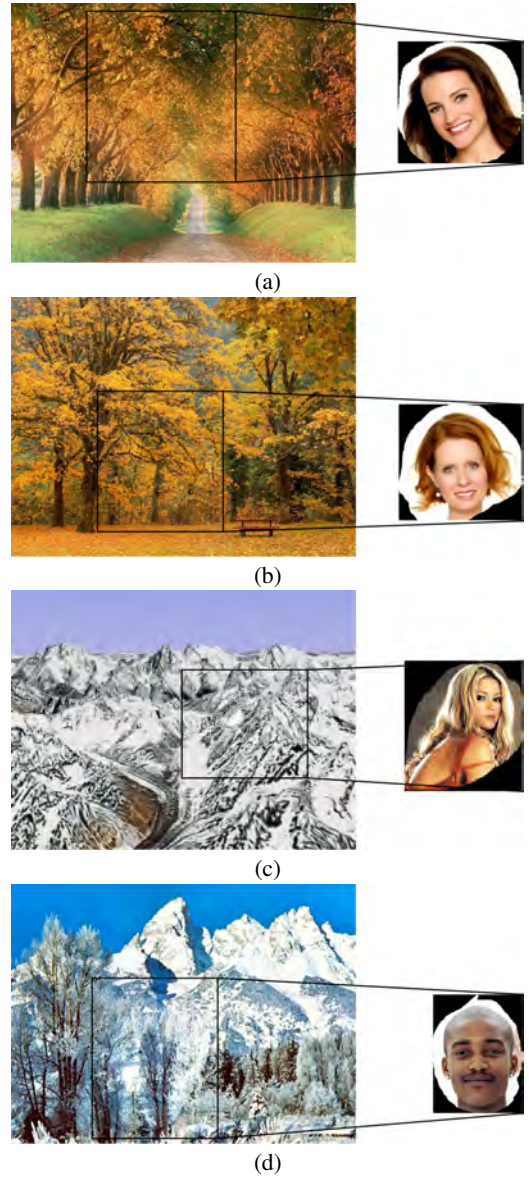


Figure 5: Further results. Left: hidden images created by our system. Right: the embedded objects.

tance of shape and edges; higher β means edges are more important shapes. The user has considerable flexibility in choosing α and β to give the most pleasing results (see Figure 6). Since v'_{pq} combines gradients of both H and O_T , it merges texture details of the object with texture details of the host image.

To further reduce the discrimination of the object and preserve the content of the background image, we set v_{pq} as the gradient of the background image if it is bigger than that of the object. The final v_{pq} is defined as follows:

$$v_{pq} = \begin{cases} H(p) - H(q), & |H(p) - H(q)| > |O_T(p) - O_T(q)| \\ v'_{pq}, & |H(p) - H(q)| \leq |O_T(p) - O_T(q)| \end{cases} \quad (5)$$

We solve the resulting Poisson blending equation for the luminance channel using the above v_{pq} . Afterwards we copy the Cr and Cb channels of the original background image to give the final synthesized image.



Figure 6: Hidden image as in Fig. 4(b), using different values of control parameters α and β .

Table 1: Algorithm performance.

Figure	Host Image	Object	θ (range, step)	s (range, step)	Matching Time	Embedding Time	Total Time
Fig. 4(a)	1024 × 768	273 × 272	$[-30^\circ, 30^\circ], 15^\circ$	[1.0, 3.0], 0.2	27s	5s	32s
Fig. 4(b)	1023 × 767	249 × 274	$[-30^\circ, 30^\circ], 15^\circ$	[1.0, 3.0], 0.2	32s	3s	35s
Fig. 4(c)	700 × 525	189 × 249	$[-30^\circ, 30^\circ], 15^\circ$	[0.6, 2.0], 0.2	11s	1s	12s
Fig. 4(d)	644 × 586	256 × 307	$[-30^\circ, 30^\circ], 15^\circ$	[0.8, 2.4], 0.2	9s	1s	10s
Fig. 5(a)	1014 × 718	263 × 256	$[-30^\circ, 30^\circ], 15^\circ$	[1.0, 3.0], 0.2	26s	5s	31s
Fig. 5(b)	1280 × 960	261 × 244	$[-30^\circ, 30^\circ], 15^\circ$	[1.0, 3.0], 0.2	29s	3s	32s
Fig. 5(c)	1022 × 679	249 × 274	$[-30^\circ, 30^\circ], 15^\circ$	[1.0, 3.0], 0.2	23s	1s	24s
Fig. 5(d)	1024 × 768	189 × 249	$[-30^\circ, 30^\circ], 15^\circ$	[1.0, 3.0], 0.2	23s	1s	24s

6 Results and Discussion

Even skilled artists take a long time to create successful and interesting hidden images. To generate such an image, an artist must first sketch the background and the embedded objects, and then paint or draw them as a coherent whole. While adding detail, the artist must be careful to achieve a good balance between hiding and revealing the embedded object. Our synthesis algorithm provides a way of automating this process. Figures 4 and 5 show several examples created by our system.

All results were produced using an Intel Core2 2.10GHz PC with 2GB memory. Computation times are shown in Table 1, as are the ranges of θ and s , and their step sizes. Unsurprisingly, shape matching takes the most time. Object embedding is relatively fast as it is based on Poisson blending. Shape matching is slow as we need to traverse many possible positions for the object in the host image. This could be reduced somewhat by the user suggesting a localised area in which to place the embedded object.

Figure 6 shows the results using different values of embedding control parameters α and β with the same host image and object: as α

Table 2: Recognition times and success rates for three difficulty levels of hidden images without and with user prior experience, and user quality ratings.

Figure	α, β			Unseen			Seen			Average Quality
	Difficult	Medium	Easy	Difficult	Medium	Easy	Difficult	Medium	Easy	
Fig. 4(a)	0.35, 0	0.40, 0	0.50, 0	17.5s (4)	10.1s (8)	6.2s (10)	13.2s (5)	6.3s (8)	1.7s (10)	3
Fig. 4(b)	0.30, 0	0.40, 0	0.50, 0	15.0s (1)	16.8s (4)	8.5s (10)	17.6s (5)	10.3s (7)	2.2s (10)	4.2
Fig. 4(c)	0.40, 0	0.50, 0	0.60, 0	19.0s (1)	18.3s (3)	17.7s (7)	17.3s (3)	14.0s (6)	12.1s (9)	3.4
Fig. 4(d)	0.30, 0	0.35, 0	0.40, 0	15.7s (3)	16.5s (6)	14.4s (9)	14.3s (3)	15.3s (8)	9.7s (10)	4.5
Fig. 5(a)	0.70, 1	0.80, 1	0.90, 1	20.0s (0)	13.3s (3)	7.1s (9)	20.0s (0)	14.3s (6)	10.3s (8)	4
Fig. 5(b)	0.70, 1	0.80, 1	0.90, 1	20.0s (0)	13.5s (4)	14.5s (8)	20.0s (0)	10.4s (5)	10.1s (9)	4
Fig. 5(c)	0.60, 0.8	0.70, 0.8	0.80, 0.8	15.3s (4)	9.3s (7)	5.1s (9)	16.2s (5)	14.0s (8)	6.0s (10)	4.5
Fig. 5(d)	0.70, 1	0.80, 1	0.90, 1	19.0s (1)	18.5s (4)	14.4s (7)	14.3s (3)	15.2s (6)	9.1s (8)	4.5

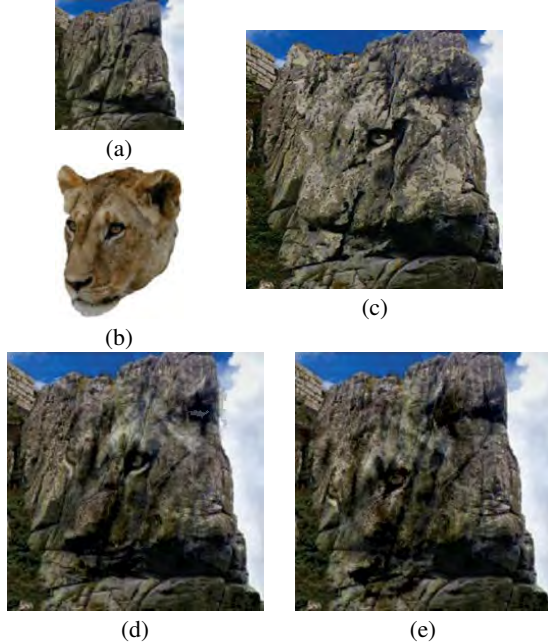


Figure 7: Comparison of Chu's approach and ours: (a) background, (b) object, (c) result using Chu's method, (d) our result embedding the object at the same position as in (c), (e) our result using automatic positioning.

decreases the embedded object becomes more difficult to detect; as β increases the embedded objects are depicted more by edges. Figure 6(c) shows that in this case the embedded object almost completely disappears for $\alpha < 0.4$, while Figure 6(d) shows that for $\alpha > 0.9$, the embedded object is too obvious.

We conducted a user study with 20 participants using 4 generated hidden images. Each hidden image was generated using three different α values given in columns 2 to 4 of Table 2, producing hidden images which we rated as difficult, medium, or easy to see. 10 participants were shown the hidden images, and 10 other participants were shown them together with the original embedded object images. The participants were given 20 seconds to study each hidden image. In each case, they were first told the difficulty level and asked to indicate the position of the hidden object, and what that object was, if they had not already been shown it. If the user could not recognize the object, we continued with the same image at an easier level (after showing them a blank frame in between); we assumed that participants could have identified easier versions if they had successfully recognized a more difficult level. Each par-

ticipant was also asked to rate the quality of each image according to the similarity of the embedded object to the original object and how well the object and host image had been combined, using a score from 1 to 5. Columns 5 to 7, and 8 to 10 of Table 2 show the average recognition times and the number of participants who successfully recognized the hidden image, for varying difficulty levels, either not having, or having, seen the original embedded objects. Column 11 shows the average quality score. The results show that the objects were well hidden in the host image, but prior knowledge of the object greatly increased the recognition rate.

We also compare our results to those in [Chu et al. 2010]. Although they also apply techniques to embed an object into a background image, our approach is quite different. [Chu et al. 2010] creates camouflage images by removing the texture details of the object, and synthesizing new texture according to the texture of the background and the structure of the object. When looking at camouflage images produced by [Chu et al. 2010], the viewer recognizes hidden objects by piecing together several neighboring color blocks. However, our approach produces hidden images by overlapping and discarding edges of objects. Viewers recognize embedded objects in our results by detecting edges and mentally complete the missing edges and supply the outer object contour. Figure 7 compares results produced by our object embedding approach with the texture synthesis method in [Chu et al. 2010]. Note that in Chu's method, the object embedding position is chosen manually. To enable a full comparison, we produce two images using our method, one by applying our object embedding approach at that same position (i.e. we skip the search for embedding location), as well as one based on our automatic shape matching approach.

7 Conclusions and Future Work

This paper has given an automatic system for generating *hidden images*. By applying techniques for shape matching and object embedding, our system can produce pleasing results which embed objects into a background image by considering edges. Viewers can see the embedded object while recognizing the background scene. Our system can produce good results with both textured and non-textured objects, but like many other computer tools, it cannot substitute for the creative work of artists. We believe our approach can provide an effective tool for amateurs to create hidden images, or for skillful artists to plan their designs before execution.

Further work is needed to improve our system. Our approach minimizes an edge matching energy, and there is scope for both improving its speed, and the edge matching process used. [Yoon et al. 2008] presents a simple rotation-invariant method for defining shape contexts. As it compares the distributions of sampled points, its effectiveness depends on how to choose sampled points in a complex and large object. We plan to investigate variants of the

shape context method for representing intensity edges in an object. We also hope to further improve our system by considering other recent progress in human perception.

Currently the best edge matching is not always adequate to provide an acceptable hiding result for an arbitrary object image and host image given by users. An alternative idea would be for the object image to be given, and to search for a host image from a database or the Internet to achieve a good hiding result. This is an easy extension of our methods.

References

- BALL, L., 2008. Liz's hidden picture puzzles. <http://www.hiddenpicturepuzzles.com>.
- BELONGIE, S., MALIK, J., AND PUZICHA, J. 2002. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, 4, 509–522.
- BERG, A. C., BERG, T. L., AND MALIK, J. 2005. Shape matching and object recognition using low distortion correspondences. In *Proc. Computer Vision and Pattern Recognition*, 26–33.
- CHU, H.-K., HSU, W.-H., MITRA, N. J., COHEN-OR, D., WONG, T.-T., AND LEE, T.-Y. 2010. Camouflage images. *ACM Transactions on Graphics* 29, 3, 51.
- COOTES, T. F., TAYLOR, C. J., COOPER, D. H., AND GRAHAM, J. 1995. Active shape models: their training and application. *Computer Vision and Image Understanding* 61, 1, 38–59.
- FANG, H., AND HART, J. 2004. Textureshop: texture synthesis as a photograph editing tool. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 23, 3, 354–359.
- FARBMAN, Z., HOFFER, G., LIPMAN, Y., COHEN-OR, D., AND LISCHINSKI, D. 2009. Coordinates for instant image cloning. *ACM Transactions on Graphics* 28, 3 (August), 67.
- HUANG, L., AND PASHLER, H. 2007. A boolean map theory of visual attention. *Psychological Review* 114, 599–631.
- HUANG, L., TREISMAN, A., AND PASHLER, H. 2007. Characterizing the limits of human visual awareness. *Science* 317, 5839, 823.
- HUTTENLOCHER, D. P., KLANDERMAN, G. A., AND RUCKLIDGE, W. A. 1993. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15, 9, 850–863.
- JIA, J., SUN, J., TANG, C.-K., AND SHUM, H.-Y. 2006. Drag-and-drop pasting. *ACM Transactions on Graphics* 25, 3, 631–637.
- LATECKI, L. J., LAKAMPER, R., AND ECKHARDT, U. 2000. Shape descriptors for non-rigid shapes with a single closed contour. In *Proc. Computer Vision and Pattern Recognition*, 424–429.
- LOWE, D. 2003. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 20, 91–110.
- MITRA, N. J., CHU, H.-K., LEE, T.-Y., WOLF, L., YESHURUN, H., AND COHEN-OR, D. 2009. Emerging images. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 28, 5, 1–8.
- OLIVA, A., TORRALBA, A. B., AND SCHYNS, P. G. 2006. Hybrid images. *ACM Transactions on Graphics* 25, 527–532.
- PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 22, 3, 313–318.
- RITTER, L., LI, W., CURLESS, B., AGRAWALA, M., AND SALESIN, D. 2006. Painting with texture. In *Eurographics Symposium on Rendering*.
- TREISMAN, A., AND GELADE, G. 1980. A feature-integration theory of attention. *Cognitive Psychology* 12, 1 (January), 97–136.
- TREISMAN, A. 1988. Features and objects: the fourteenth bartlett memorial lecture. *Quarterly Journal of Experimental Psychology* 40A, 201–236.
- VELTKAMP, R. C., AND HAGEDOORN, M. 2001. *State of the art in shape matching*. Springer.
- VETTER, T., JONES, M. J., AND POGGIO, T. 1997. A bootstrapping algorithm for learning linear models of object classes. In *Computer Vision and Pattern Recognition*, 40–46.
- WANDELL, B. 1995. *Foundations of Vision*. Sinauer Associates Inc.
- WOLFE, J. M. 1994. Guided search 2.0: A revised model of visual search. *Psychonomic Bulletin and Review* 1, 2, 202–238.
- YOON, J., LEE, I., AND KANG, H. 2008. A hidden-picture puzzles generator. *Computer Graphics Forum* 27, 1869–1877.