EUROGRAPHICS 2006 / E. Gröller and L. Szirmay-Kalos
(Guest Editors)

Volume 25 (2006), Number 3

Semantic Photo Synthesis

M. Johnson, G. J. Brostow, J. Shotton, O. Arandjelovic, V. Kwatra[1], and R. Cipolla

University of Cambridge
[1]UNC-Chapel Hill

Abstract

Composite images are synthesized from existing photographs by artists who make concept art, e.g., storyboards for movies or architectural planning. Current techniques allow an artist to fabricate such an image by digitally splicing parts of stock photographs. While these images serve mainly to "quickly" convey how a scene should look, their production is laborious. We propose a technique that allows a person to design a new photograph with substantially less effort. This paper presents a method that generates a composite image when a user types in nouns, such as "boat" and "sand." The artist can optionally design an intended image by specifying other constraints. Our algorithm formulates the constraints as queries to search an automatically *annotated image database. The desired photograph, not a collage, is then synthesized using graph-cut optimization, optionally allowing for further user interaction to edit or choose among alternative generated photos. An implementation of our approach, shown in the associated video, demonstrates our contributions of (1) a method for creating specific images with minimal human effort, and (2) a combined algorithm for automatically building an image library with semantic annotations from any photo collection.*

1. Introduction

Paintings, illustrations, and photographs in particular are very efficient for conveying information. But efficiency depends on both the audience's reception and the author's expended effort. Our primary contribution is a new method for creating images of specific things and people, with minimal human effort. While a photographer might opportunistically capture a desired image, we enable the average person to design a photograph semantically, as if explaining to another human that "it looks like this." Such images are desirable as concept art that is normally carefully prepared for team productions such as movies and architectural planning. Beyond aesthetic value, such art has very practical uses. The layout of elements in a planning image conveys the creator's vision, which can then be interpreted by others and revised. Alfred Hitchcock's storyboards are famous for their impact on the scene-by-scene look of his movies, though they are somewhat utilitarian. The industry of stock photography is organized around themes of images, so that customers can more easily locate each image of an intended category, though combining of categories usually requires some Photoshop^{TM} talent.

Figure 1: *An image generated by our system. The query was "water" in front of Taj Mahal image (also see Figure 10).*

The goal of our proposed method is to simplify image creation by leveraging raw image archives. Given an empty canvas to start, the user can enter words such as "sky" and "car" in the caption or at desired locations (See Figure 2).

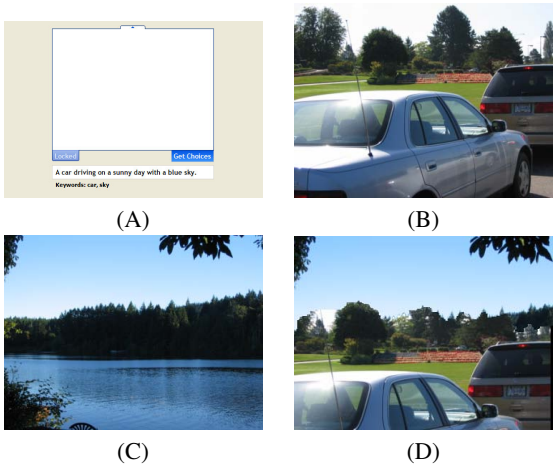


Figure 2: *In (A), the user specifies only a semantic caption. (B) and (C) show the automatically selected source images used to synthesize the designed photograph (D).*

The canvas also accepts other similar semantic labels such as proper names of people, and can be supplemented with copy-pasted islands of pixels for categories that lack semantic labels, such as logos or scanned designs (Figure 1 shows a result for which the query was a combination of text and literal patches). Our algorithm treats the empty parts of the canvas as don't-cares, and searches a specified image library to find candidate stock images. Different combinations of these candidate images are then stitched together using a graph cuts based seam optimization algorithm to generate realistic photographs for each combination. The best combination is then picked as the one with the minimum seam cost. Our secondary contribution is a parallel algorithm for automatically building an image library of semantic annotations from any photo collection, that can then be used for image creation as described above.

2. Background

The existing Content Based Image Synthesis work of [DEJ04] also seeks a high-level interface for performing effective image manipulation, and operates by recombining image regions from a database with semantic labels. They define the problem in terms of semiotics, and explain the three combinations of possible variations: (1) vary the paradigm (meaning) and fix the syntagm (spatial positioning), (2) fix the paradigm and vary the syntagm, and (3) vary both. They considered only the first combination and were able to alter the mood of a photo by replacing, for example, a clear sky with a stormy one. We instead consider the third case, generating our semantic labels automatically instead of manually; our user's image design is populated by drawing from 160,000 regions and 16,000 images, compared to their 100 regions and 50 images. In addition, their interface is meant for detailed manipulation of a manually specified region, while our method allows for very fast design of new photographs. The Semanticons work [SABGG05] parses files for their overall meanings and then automatically merge icons which represent those meanings to create synthesized icons for the files. These icons form an intuitive visual clue to the contents of the file, and were found in user studies to raise user productivity.

Interfaces Annotated data has been employed to generate different graphical elements based on simple user instructions. [AFO03] showed how annotated motion capture data can be employed to synthesize animations with various desired behaviors. A special high-level scripting language was developed in [CDM*02] to simplify the process of solid modeling, while a beautiful interface that inferred the third dimension from pen-strokes was presented in the surface modeling of [IMT99]. Except for [HJO*01] (who address combination (2) above), there has been little progress for the high level design of photographs.

Synthesis Novel image synthesis from example source images has been of great research interest in recent times. The work most closely related to our own is that of Graphcut Textures [KSE*03] in which a large texture image is synthesized from a much smaller example image by intelligent placement and stitching of multiple copies of the input example. The stitching is performed using a graph cuts based optimization algorithm to determine the *least noticeable* seam (or transition boundary) that passes through a set of overlapping input patches. Graphcut textures also support low-level user control for fusing together arbitrary input images – not just textures – where the user can constrain certain pixels to copy from a certain input before performing the graph cut seam optimization. We use graphcut textures as the backbone of our synthesis framework but allow user control at a much higher semantic level, where the constraints are specified as annotations in the form of *words* and not mere pixels. Other related techniques include Image Analogies [HJO*01], where example pairs of input-output images are used as guides to learn filtering and painting operations, that are then applied to new inputs. They also demonstrate pixel-level user control in their texture-by-numbers application. In Digital Tapestry [RKKB05], multiple images from a photo collection are used to automatically construct a collage by choosing informative and spatially compatible regions from the collection. They attempt to pack as much information as possible from the input collection into the synthesized image, thereby creating a summary of the collection. In our work, on the other hand, we extract only as much information from the database as necessary to satisfy the user specification, thereby culling away most of the database.

Semantic Labeling of Image Libraries Semantic labeling of images is an especially active area of research for the field of Computer Vision. It encompasses the problems of object detection, recognition, and segmentation, expanding the range of relevant semantic labels. Numerous approaches exist because so far, all have significant failure modes, but

the representative works of [KH05,SWRC06] are among the newest algorithms that consider image regions in their context of the rest of the image. Clever approaches for retrieving images from automatically classified image libraries include [TS00] and [FB03]. The former has the original semantic vocabulary used in [DEJ04], and the latter allows for refinement of queries by allowing a user boolean control over large sets of small category images. Interestingly, several websites (Facebook.com and Flickr.com) dispense with automation as being too unreliable, and maintain only labeling information entered by people visiting these sights to view their friends' photo albums. However, our method can tolerate the shortcoming of current automatic category and face labeling methods by virtue of having a human making the final decision among results produced by the system.

We now proceed to explain how our system automatically builds an image library, and our algorithm for allowing a user to very quickly and easily build specific new photos.

3. Image-Library Preprocessing

A user is essentially allowed to paint using semantic labels, so our method relies on the quantity and variety of available image regions that have those labels. While images on the internet have contextual information, most digital photos have no more than a timestamp. Even newspaper image archives have little more than the text from each caption. The task of automatically assigning semantic labels to pixel regions is a long standing vision problem that we do not claim to have solved. However, we are able to consolidate the complimentary algorithms that currently lead the field for their respective label categories. The result is an *automatic* means of generating a broad range of semantic labels for thousands of raw images.

3.1. Automatic Semantic Image Labeling

Our implementation uses the TextonBoost algorithm [SWRC06] of Shotton et al. to accurately segment test images into contiguous semantically labeled regions. We provide a brief overview of this algorithm here. The segmentation result is created by evaluating a learned classification function at every pixel; we first describe the image features used by the classifier, and then its form.

The algorithm takes an L*a*b* image and convolves it with a bank of Gaussians, derivatives of Gaussians, and Laplacian filters, over the three color channels. L*a*b* space is used as it is a perceptually uniform color space and has been shown to work well for the purposes of textonization. Each pixel is then assigned a texton [MBSL99, VZ02] number according to a nearest neighbor search within a learned texton dictionary for the filter response vector at that pixel, producing a texton map. The features used in the classification function, known as 'shape filters', are each computed as a count of pixels with a given texton number within a particular sub-window of the texton map.

The classification function is an additive model learned

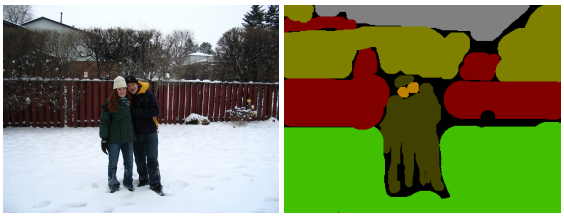


Figure 3: *A sample training image with corresponding ground truth which has been hand-labelled through a 'paint' interface offline.*

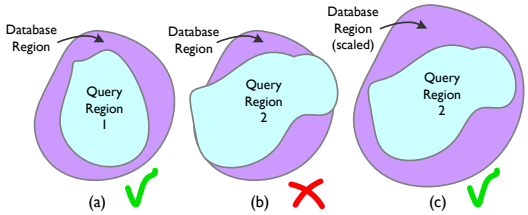


Figure 4: *Examples of matching query regions to a database region. (a) Query region 1 can be fully contained within the database region at some translation. (b) Query region 2 is too large to fit within the database region at any translation. (c) However, scaling up the database region allows query region 2 to be fully contained. This match will be given a higher cost since we desire minimum translation and scaling of the database regions.*

using the Joint Boosting algorithm of [TMF04]. It combines a number M of weak classifiers h_m additively as

$$H_i(c,\mathbf{x}) = \sum_{m=1}^{M} h_{i,m}(c,\mathbf{x}) \tag{1}$$

at pixel i, for category label c and input image $\mathbf{x}$. A posterior category probability can then be computed trivially for each category c with a soft-max function

$$P_i(y_i = c|\mathbf{x}) = \frac{\exp H_i(c,\mathbf{x})}{\sum_c \exp H_i(c,\mathbf{x})} \tag{2}$$

where y_i is the random variable representing the category label at pixel i. Note that the learning algorithm requires roughly labeled training data, as seen in Figure 3, and so 300 example images were hand-labeled in about 3hrs.

For the purposes of this paper, a maximum *a-posteriori* (MAP) estimate of the category labels is taken at each pixel, $\hat{y}_i = \arg\max_c P_i(y_i = c|\mathbf{x})$, and combined to give the MAP image, $\mathbf{y}$. The code for this labelling is available online at: `http://mi.eng.cam.ac.uk/~jdjs2/`.

Semantic-Driven Search The MAP image is computed for each database image offline. Each MAP image is then divided into a set of contiguous semantically labeled regions, by means of a standard connected-component analysis. To

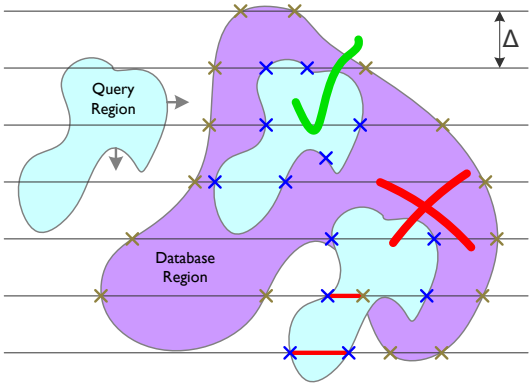


Figure 5: *To determine matching locations, the query region is scanned across the database region. The matching algorithm determines whether all scans of the translated query region (marked with blue crosses) lie wholly within scans of the database region (marked with brown crosses), for all scan-lines at interval Δ. An example, matching, relative translation is indicated with a green check-mark, and another, non-matching translation, with a red cross. The horizontal red lines indicate the parts of query scans that fail to be contained within the database scans. Note that the search in scale has been omitted from this figure for clarity.*

avoid spurious matches and increase efficiency, only regions with a certain minimum area are kept.

Our search algorithm takes a query region, semantically labeled through our user-interface, and attempts to match it against all regions of the same category label in the database, such that for a particular relative shift and scaling, the database region completely covers the query region (see Figure 4). A search over all possible shifts and a range of scales can be done extremely efficiently by comparing sets of spans (start and end x-coordinates) for each scan-line in the query (see Figure 5). These spans can be pre-computed.

3.2. Automatic Face Recognition

The inherent difficulties of face recognition and those specifically in the context of image and video retrieval [BBE*04, SEZ05] are well appreciated in the literature. Lighting, facial expressions, and head pose drastically change the appearance of faces, as do occlusions, blur and small resolution. The recognition algorithm we employ is based on the adaptive filtering framework of [AC06]. We start with the face detector of [VJ04]. We then recognize faces from single photos by synthetically generating a face set from each single input and then performing set-to-set matching.

Assuming that the face to image space embedding function is smooth, geodesically close images correspond to small changes in imaging parameters (e.g. yaw or pitch). Hence, the face motion manifold is locally topologically similar to the *affine warp* manifold. We generate a set of face images from a single face by applying stochastic affine warp perturbations to the face (see [Mar02] and [SP98]). Similarity between two sets of images is computed as the weighted average of the first three canonical correlations [BG73] between these sets. That similarity measures the variation of subspaces spanned by the two sets.

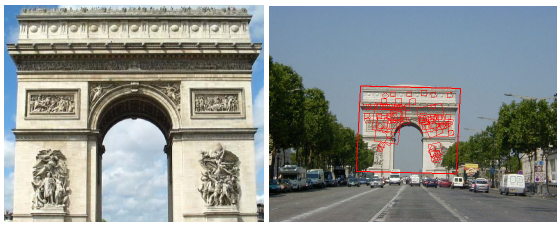


Figure 6: Sample Retrieval Result. *On the left is a sample Literal Patch, placed by the user in the canvas as a constraint. The image on the right shows an intermediate step, where an image with matching feature points has been found in the database. Literal patch matching is necessary because the Arc de Triomphe and many other objects are not yet recognized with semantic labels.*

3.3. Literal Patches

A good deal of recent work uses interest points and robust, local descriptors for image retrieval and matching [Low04, KSH04, OM05, Kad02, LSP03, TG04]. The majority of these methods incorporate a nearest neighbor search, where the closest match for an interest point descriptor in the query image is found for the ultimate purpose of discovering which image or images in a database match the query. Groups of these matches are verified, often using geometric constraints, and then evaluated, the final result being a ranked list of possible database images which appear in some form in the query. A sample result from such a system is shown in Figure 6.

Naturally, this process can be reversed such that all images in a database which contain a particular object are returned in ranked order, and it is in this context that the technique becomes interesting for use in image synthesis. There are certain circumstances when the user will want to specify that a particular patch of an image must be present in the synthesized result, as is the case with a particular landmark, or a particular building facade. In these cases, asking for a generic region from the database will not suffice, and so the user is able to add a "literal" image patch to the canvas. In order to find matching image patches to use in synthesis, we use a modified version of PCASIFT [KS04] using the database structure from [KSH04] along with a center clustering technique for pose prediction to query the database.

4. Synthesis Framework

Our method for enabling an artist to create images of specific scenes benefits from an implementation with a clean

user interface. While other interfaces can be built around the algorithm, our implementation resembles a simple painting program. A blank box in the middle serves as a canvas which the user annotates with descriptions of their desired elements. The descriptions are usually in the form of text, placed on the canvas to specify relative locations where the requested categories should be generated.

The descriptions (or constraints) specified by the user are converted into queries, which are pixel masks paired with categories, then handed over to the search algorithm. The procedure for query formulation from constraint specification is explained in Section 4.1. The database search returns a set of real photographs for each constraint specified by the user. This set is then sorted into a list based on category-specific heuristics and the magnitude of shifting and scaling required to align the query and the database photograph. Note that we can search for all user constraints in parallel since they are considered independently of each other.

Once we have a sorted list of photographs for each user constraint, we synthesize a set (or selection list) of output photographs, where each output photograph is constructed from a combination of database photographs. Each combination picks one photograph for each constraint by sampling from its sorted list of photographs. For each combination, the photograph is synthesized using the graph cut optimization algorithm to find seams (transition boundaries) between the database photographs in the combination[†]. Typically, we search the top few combinations exhaustively and then sample the remaining greedily by favoring those with less costly graph-cut seams. The synthesized photographs are eventually ranked based on their graph cut seam cost and returned to the user for selection. See Algorithm 1 for pseudo-code.

Algorithm 1 Synthesis Pipeline

```
for all constraint ∈ user_constraints_list do
  query ← ParseConstraint(constraint)
          {query ≝ pair<mask, category>}
  db_images ← SearchDataBase(query)
  db_images_list[constraint] ← Sort(db_images)
end for {for loop executed in parallel}
synthesized_list ← empty
repeat
  source_images ← NextCombination(db_images_list)
  synthized_image ← GraphCutsStitch(source_images)
  synthesized_list.Add(synthesized_image)
until TestedEnoughCombinations()
```

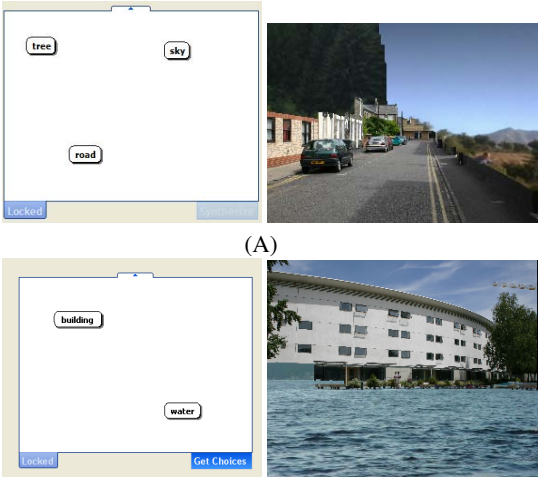


Figure 7: *Shown are examples of user's text specification and corresponding synthesized images: "tree," "road," and "sky" in (A) and "building" and "water" in (B).*

4.1. Easy Formulation of Queries

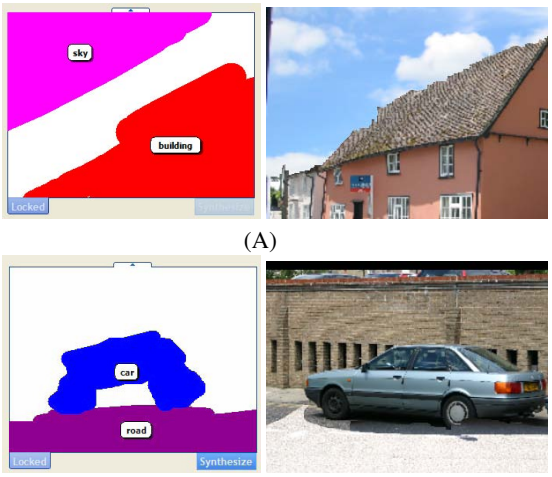
Text by Itself The user enters text after clicking within the canvas. Once one or more such constraint words have been entered, the "synthesize" button converts the constraints into mask images. If the word "car" was typed in the bottom right, then that word is sent to our recognition database with a query mask featuring a circle of *on* pixels in the bottom right. The query proceeds as described in the Semantic-Driven Search of Section 3.1 (see Figure 7 for examples).

Text with Paint A user may wish to design a photograph in which some of the categories appear in specific shapes and sizes. The paint-tool is used for this case, and each connected component of painted pixels is associated with a typed constraint word (see Figure 8). Once the desired photo has been designed, clicking "synthesize" generates location mask queries, performs the search, and returns a collection of synthesized images for the user to choose from.

Location-free Text in Image Captions When the user enters constraining text without locating their preference of object locations in the canvas, our application samples a location prior that has been learned from the labeled training images. The prior is learned by creating a normalized histogram for each category label of location. See Figure 9. To sample from this prior we use a simple and efficient Gibbs sampler. Figure 2 shows the result of entering scene direction for a movie script as the caption.

Literal Patches For situations where the desired photo should contain an object from an unknown category, the user can drag-drop an example of the object, assuming it has distinct image features, into the canvas as one of the constraints (see Figure 10).

[†] The expansion move algorithm [BVZ01, KZ04] is used to stitch the candidate images together. Combining ideas from Graph Cut Textures [KSE*03] and Grab Cut [RKB04], a color model is learned for each user specified region, and used to assign a cost to all pixels within each source image. These unary potentials are combined with pairwise potentials based on a contrast sensitive Potts model [BJ01]. The expansion move algorithm finds an (approximate) minimum energy which can compose multiple layers together simultaneously.



(A)

(B)

Figure 8: *The user may both specify text and paint desired regions to synthesize a photo. (A) and (B) show the query for "sky" and "building"; (C) and (D) show "car" and "road."*

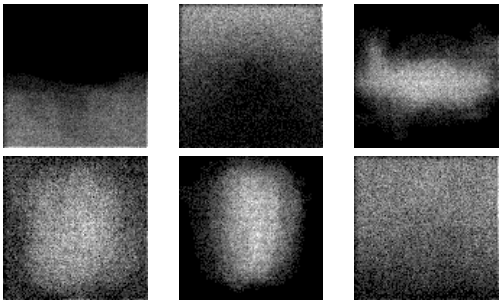
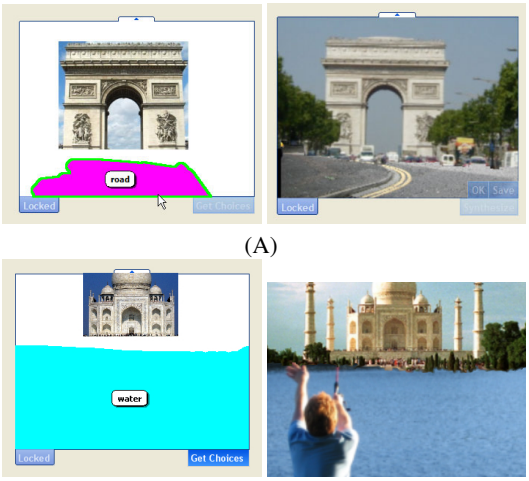


Figure 9: *Whiteness of dots indicates the learned prior probability of that category occurring at that image location; image intensities are displayed here with a gamma = 1.75 enhancement. Shown are example categories (left to right, top row first): snow, sky, airplane, flower, face, tree.*

5. Results & Conclusions

Our method for designing photographs has successfully enabled non-experts to quickly create a variety of images using only text, text and some markup of a canvas, or using example pixel patches. The average image takes a few typed words and 15 to 45sec. to synthesize, where most of that time is spent waiting for the graph-cut optimization to try out more combinations. The system returns an array of results like those shown in Figure 11 for the user to choose from to find the one which is most aesthetically pleasing, or appropriate to the task at hand.

The system does have distinct failure modes, as depicted in Figure 12. They are related to known limitations of the automatic semantic labeling and image stitching technologies, however, and as those systems improve so will ours. To deal with the variety of images available in any significant photo archive, the interface allows the user to revise results.



(A)

(B)

Figure 10: *The drag-dropped example pixels are treated as part of the image's design. The synthesized image contains pixels from a matching object, even though there is not yet a semantic label for this category.*

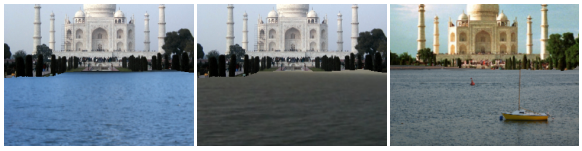


Figure 11: *The system returns a wide array of results that combine the input images in many different ways. These images along with the one in Figure 10B are the top four results returned for the Taj Mahal and "water" query shown in Figure 10B.*

To replace a stitched region within a result, the user picks an alternate same-class source image from other most-highly compatible candidates. Generally, so little human effort is sufficient only because we were able to automate the approximate but broad image-database category labeling. With fewer images, user requests for specific layouts would be harder to synthesize, though image-flipping and automatic color correction are known technologies that could be employed.

(A) (B) (C)

Figure 12: *The system has three main failure modes: (A) the stitching algorithm chooses incorrect seams and severs an object; (B) the automatic semantic labeling fails and gives an incorrect result for a semantic label; (C) scale differences result in obvious resolution change across a border.*

6. Acknowledgements

This work was funded by a Microsoft Research Studentship, the Louise T. MacBain Cambridge Studentship, and funding from the Cambridge-MIT Institute. We are grateful to Jim Davies for helpful discussions.

References

[AC06] ARANDJELOVIĆ O., CIPOLLA R.: A new look at filtering techniques for illumination invariance in automatic face recogition. *In Proc. IEEE Conference on Automatic Face and Gesture Recognition* (April 2006).

[AFO03] ARIKAN O., FORSYTH D. A., O'BRIEN J. F.: Motion synthesis from annotations. *ACM Trans. Graph. 22*, 3 (2003), 402–408.

[BBE*04] BERG T. L., BERG A. C., EDWARDS J., MAIRE M., WHITE R., YEE WHYE TEH ERIK LEARNED-MILLER D. A. F.: Names and faces in the news.

[BG73] BJÖRCK Å., GOLUB G. H.: Numerical methods for computing angles between linear subspaces. *Mathematics of Computation 27*, 123 (1973), 579Ű–594.

[BJ01] BOYKOV Y., JOLLY M.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *Proc. of ICCV '01* (2001), vol. 1, pp. 105–112.

[BVZ01] BOYKOV Y., VEKSLER O., ZABIH R.: Efficient approximate energy minimization via graph cuts. *IEEE transactions on PAMI 20*, 12 (November 2001), 1222–1239.

[CDM*02] CUTLER B., DORSEY J., MCMILLAN L., MÜLLER M., JAGNOW R.: A procedural approach to authoring solid models. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), pp. 302–311.

[DEJ04] DIAKOPOULOS N., ESSA I. A., JAIN R.: Content based image synthesis. In *CIVR* (2004), pp. 299–307.

[FB03] FAUQUEUR J., BOUJEMAA N.: Logical query composition from local visual feature thesaurus. In *Third International Workshop on Content-Based Multimedia Indexing (CBMI'03)* (2003).

[HJO*01] HERTZMANN A., JACOBS C. E., OLIVER N., CURLESS B., SALESIN D. H.: Image analogies. *Proceedings of SIGGRAPH 2001* (August 2001), 327–340. ISBN 1-58113-292-1.

[IMT99] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: a sketching interface for 3d freeform design. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), pp. 409–416.

[Kad02] KADIR T.: *Scale, Saliency, and Scene Description*. PhD thesis, University of Oxford, 2002.

[KH05] KUMAR S., HEBERT M.: A hierarchical field framework for unified context-based classification. In *Proc. ICCV* (October 2005).

[KS04] KE Y., SUKTHANKAR R.: Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of CVPR '04* (2004).

[KSE*03] KWATRA V., SCHÖDL A., ESSA I., TURK G., BOBICK A.: Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics, SIGGRAPH 2003 22*, 3 (July 2003), 277–286.

[KSH04] KE Y., SUKTHANKAR R., HUSTON L.: Efficient near-duplicate detection and sub-image retrieval. In *Proc. ACM Multimedia* (2004).

[KZ04] KOLMOGOROV V., ZABIH R.: What energy functions can be minimized via graph cuts? *IEEE Transactions on PAMI* (February 2004).

[Low04] LOWE D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (2004), 91–110.

[LSP03] LAZEBNIK S., SCHMID C., PONCE J.: Affine-invariant local descriptors and neighborhood statistics for texture recognition. In *Proceedings of ICCV '03* (2003), pp. 649–656.

[Mar02] MARTINEZ A. M.: Recognizing imprecisely localized, partially occluded and expression variant faces from a single sample per class. *IEEE Trans. Pattern Analysis and Machine Intelligence 24*, 6 (2002), 748–763.

[MBSL99] MALIK J., BELONGIE S., SHI J., LEUNG T.: Textons, contours and regions: Cue integration in image segmentation. In *Proc. ICCV '99 - Volume 2* (1999), p. 918.

[OM05] OBDRŽÁLEK S., MATAS J.: Sub-linear indexing for large scale object recognition. In *Proceedings of BMVC '05* (2005).

[RKB04] ROTHER C., KOLMOGOROV V., BLAKE A.: Grabcut - interactive foreground extraction using iterated graph cuts. *Proc. ACM Siggraph* (2004).

[RKKB05] ROTHER C., KUMAR S., KOLMOGOROV V., BLAKE A.: Digital tapestry. In *CVPR (1)* (2005), pp. 589–596.

[SABGG05] SETLUR V., ALBRECHT-BUEHLER C., GOOCH A., GOOCH B.: Semanticons: Visual metaphors for files. In *Proc. of Eurographics '05* (2005).

[SEZ05] SIVIC J., EVERINGHAM M., ZISSERMAN A.: Person spotting: video shot retrieval for face sets.

[SP98] SUNG K. K., POGGIO T.: Example-based learning for view-based human face detection. *IEEE Trans. Pattern Analysis and Machine Intelligence 20*, 1 (1998), 39–51.

[SWRC06] SHOTTON J., WINN J., ROTHER C., CRIMINISI A.: Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV (to appear)* (2006).

[TG04] TUYTELAARS T., GOOL L. V.: Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision 59*, 1 (2004), 61–85.

[TMF04] TORRALBA A., MURPHY K. P., FREEMAN W. T.: Sharing features: efficient boosting procedures for multiclass object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (June 2004), vol. 2, pp. 762–769.

[TS00] TOWN C., SINCLAIR D.: *Content based image retrieval using semantic visual categories*. Tech. rep., 2000.

[VJ04] VIOLA P., JONES M.: Robust real-time face detection. *International Journal of Computer Vision 57*, 2 (2004), 137–154.

[VZ02] VARMA M., ZISSERMAN A.: Classifying images of materials: Achieving viewpoint and illumination independence. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark* (May 2002), vol. 3, pp. 255–271.