# Creating and Exploring a Large Photorealistic Virtual Space

Josef Sivic[1]*, Biliana Kaneva[2], Antonio Torralba[2], Shai Avidan[3] and William T. Freeman[2]

[1]INRIA, WILLOW Project, Laboratoire d'Informatique de l'Ecole Normale Superieure, Paris, France

[2]CSAIL, Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.

[3]Adobe Research, Newton, MA, U.S.A.

`The supplementary video can be viewed at: http://people.csail.mit.edu/josef/iv08/video.avi`

## Abstract

*We present a system for exploring large collections of photos in a virtual 3D space. Our system does not assume the photographs are of a single real 3D location, nor that they were taken at the same time. Instead, we organize the photos in themes, such as city streets or skylines, and let users navigate within each theme using intuitive 3D controls that include move left/right, zoom and rotate. Themes allow us to maintain a coherent semantic meaning of the tour, while visual similarity allows us to create a "being there" impression, as if the images were of a particular location. We present results on a collection of several million images downloaded from Flickr and broken into themes that consist of a few hundred thousand images each. A byproduct of our system is the ability to construct extremely long panoramas, as well as image taxi, a program that generates a virtual tour between a user supplied start and finish images. The system, and its underlying technology can be used in a variety of applications such as games, movies and online virtual 3D spaces like Second Life.*

## 1. Introduction

Exploring large collections of images requires one to determine how to organize images and how to present them. A leading approach to organizing photo collections is to use tags. The user navigates the collection by typing a query tag and reviewing the retrieved images. Typically, the results are shown as a page of thumbnails or a slide show, which is a practical but not engaging way to browse images. Moreover, automatic image tagging is still an unsolved problem and manual image tagging is a time consuming process that does not always occur in practice.

Of particular interest are geo-referenced images that can be automatically and accurately tagged with GPS data. This allows users to explore the collection based on location, but more importantly offers a new way to navigate. Instead of showing the retrieved images as thumbnails, one can display images in a 3D context, enhancing the user experience. Instead of typing query tags, the user can use simple and intuitive 3D controls to navigate. Indeed, the Street View approach offered by several companies allows users to virtually wander the streets of a city in a 3D like fashion. This was underscored by the success of the PhotoTourism interface [27] which put all images of a given scene in a common 3D space using bundle adjustment. Users browse the image collection by moving in 3D space.

If all images are collected from the same point of view then a big Gigapixel image can be constructed and viewed interactively using familiar 2D image controls [16]. Alternatively, if the image dataset consists of a random collection of images of different scenes, taken at different times, one can construct an AutoCollage [25]. This gives visually pleasing collages, but fails to scale to large collections where thousands or millions of images are involved.

Another option considered in the past is to treat images as points in a high dimensional space, compute the distance between them and use multi-dimensional scaling to display them in the image plane for the user to navigate through [26]. However, there is no effort to create a virtual 3D world and as a result there is no sense of "being there".

In this work, we use intuitive 3D navigation controls to explore a virtual world created from a large collection of images, as illustrated in figure 1. This side-steps the problem of image tagging altogether, relying instead on an automatic image similarity measure. In our approach the user is free to move from one image to the next using intuitive 3D controls such as move left/right, zoom and rotate. In response to user commands, the system retrieves the most visually similar images, from a data set of several million images, and displays them in correct geometric and photometric alignment with respect to the current photo. This gives the illusion of moving in some large virtual space. We preprocess the collection ahead of time, allowing the system to respond interactively to user input.

There are a number of applications to this technology.

*This work was carried out whilst J.S. was at MIT, CSAIL

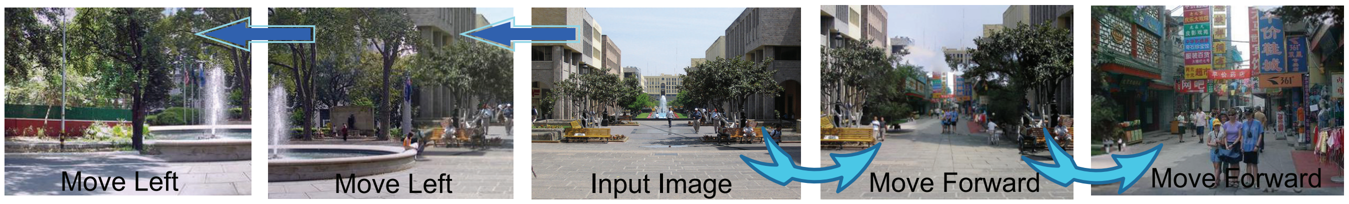

Figure 1: Given the user supplied starting image, our system lets the user navigate through a collection of images as if in a 3D world.

First, imagine large 3D virtual online worlds like Second Life, where users are free to roam around. We make it possible for them to tour a photorealistic environment, instead of a computer generated one. Alternatively our method can be used to construct large photorealistic environments for games. Other applications include movies or advertisements, where very wide backgrounds for a particular scene may be needed.

A byproduct of our method is the ability to create different types of panoramas such as infinitely long panoramas or an image taxi. Image taxi allows the user to specify the first and last images of a tour and the program will automatically generate a virtual tour through the image collection that starts and ends at the user specified images. This extends the large body of work on multi-perspective panoramas [22, 24] where a large number of images taken from different view points are stitched together. Our method does not assume images to be of the same scene or taken at the same time.

In a similar fashion we can create infinite zoom effects that resemble the "Droste effect"[1]. The effect is named after a particular image that appeared on the tins and boxes of the Dutch Droste cocoa powder. It displays a nurse carrying a serving tray with a cup of hot chocolate and a box of Droste cocoa. The recursive effect first appeared in 1904, and eventually become a household notion. This effect has been used by various artistic groups to generate infinite zooms[2].

To achieve these goals we use a number of components. First, we use a large collection of images, which defines the image space the user can explore. Second, we use a convenient interface to break images into themes to ensure the semantic coherence of the tour. Third, we use an image similarity metric that allows us to retrieve images similar to a transformed version of the current image (e.g. to support a "move left/right" operation we need to shift the current image, before retrieving similar images). Finally, we show how to combine two images in a visually pleasing manner to create the illusion of moving in a single, coherent 3D space.

## 2. Constructing the image space

In this section we describe how we collect and arrange a large collection of images into a virtual 3D space for the user to explore as if walking through the pictures. The images do not need to correspond to a real unique 3D space as in Phototourism [27]. Instead, our images are expected to correspond to unrelated places.

[1]http://en.wikipedia.org/wiki/Droste_effect

[2]http://zoomquilt2.madmindworx.com/zoomquilt2.swf

We collected a dataset of more than 6 million images from Flickr by querying for suitable image tags and relevant groups. We queried for particular locations in conjunction with tags such as 'New York street', or 'California beach', and also downloaded images from Flickr groups returned by search on terms relevant to particular themes, such as 'Landscape' or 'Forest'. The typical resolution of the downloaded images is $500 \times 375$ pixels. One million jpeg compressed images takes about 120GB of hard-disk space.

Traditional image retrieval deals with the task of finding images that are similar to a particular query image. Here we want to extend this query by also allowing transformations between images. For instance, we want to find images similar to a photo taken if we rotate the camera by 45 degrees with respect to a query image. We hope that the candidate set contains images that can be seamlessly blended with the query image after applying the appropriate camera transformation. We show how a simple planar image model can be used to retrieve images that are related to the query image by 3D transformations.

### 2.1. Geometric scene matching

Our goal is to extend a given image using images from the theme database to give an impression of a particular camera motion. We consider three camera motions: (i) translation left/right, (ii) horizontal rotation (pan), and (iii) zoom (forward motion). The camera motions are illustrated in figure 2. First, we synthesize a new view of the current image as seen from the new desired camera location. Camera translation is approximated by a translation in the image plane, ignoring parallax effects, horizontal camera rotation is achieved by applying appropriate homography transformation to the image and zoom is approximated by scaling the image.

Now, we find semantically similar images in the database coarsely matching the geometry of the observed portion of the synthesized view. For example, when the camera rotation changes a fronto-parallel view of a building to a view with a strong perspective (as shown in the middle column of figure 2), we find most retrieved images depict scenes looking down a street.

### 2.2. Image representation

A key component of our approach is finding a set of semantically similar images to a given query image. For example, if the query image contains a cityscape in a sunset with a park in the foreground, we would like to find a candidate set

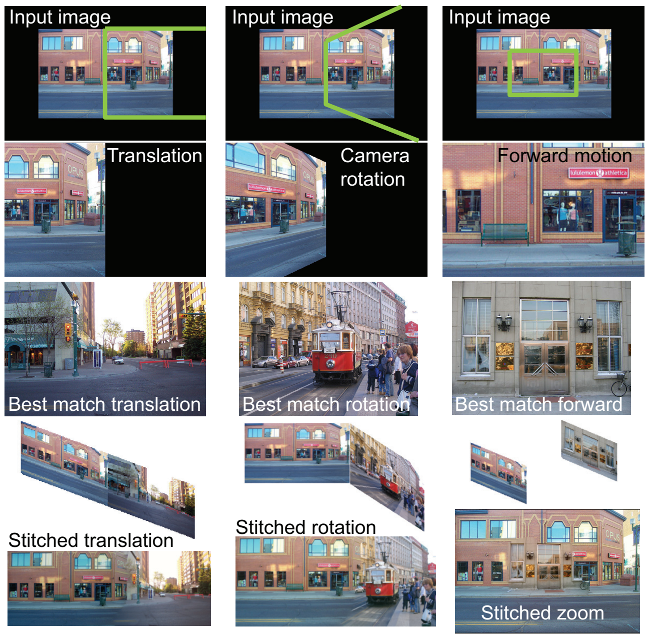

Figure 2: Scene matching with camera view transformations. First row: The input image with the desired camera view overlaid in green. Second row: The synthesized view from the new camera. The goal is to find images which can fill-in the unseen portion of the image (shown in black) while matching the visible portion. The third row shows the top matching image found in the dataset of street scenes for each motion. The fourth row illustrates the induced camera motion between the two pictures. The final row shows the composite after Poisson blending.

of images with similar objects, camera viewpoint and lighting. We describe the image features we use to achieve this and how to extract semantic scene description and camera properties from them.

Semantic scene matching is a difficult task but some success has been recently shown using large databases of millions of images [13, 29]. In this paper we show that we can also induce camera transformations without an explicit model of the 3D geometry of the scene. Matching results are sometimes noisy; for example, a river is sometimes mismatched for a road. In a recent work on image completion [13] this issue was addressed by relying on user interaction, essentially allowing the user to select a visually pleasing result among a set of candidate completions. We wish to find matching images automatically or with minimal user interaction. To reduce the difficulty of scene matching we train classifiers to pre-select images of particular scene types or themes from the entire image collection. The user can then navigate in a visual space that is built only from images of a particular theme or a combination of themes. Images are represented using the GIST descriptor, which was found to perform well for scene classification [21]. The GIST descriptor measures the oriented edge energy at multiple scales aggregated into coarse spatial bins. In this work we use three scales with (from coarse to fine) 8, 8 and 4 filter orientations aggregated into $6 \times 4$ spatial bins, resulting in a 480 dimensional image descriptor. In addition we represent a rough spatial layout of colors by down-sampling each of the RGB channels to $8 \times 8$ pixels. We normalize both GIST and the color layout vectors to have unit norm. This makes both measures comparable.

As illustrated in figure 2, not all pixels are observed in the transformed image and hence only descriptor values extracted from the observed descriptor cells in the query image are considered for matching. In this case, the GIST and the color layout vectors are renormalized to have unit norm over the visible region.

The distance between two images is evaluated as the sum of square differences between the GIST descriptors and the low-resolution color layout of the two images. We set the weight between GIST and color to 0.5, which we found is a good compromise between matching on the geometric structure of the scene captured by GIST and the layout of colors. Currently we consider only images in the landscape format with an aspect ratio close to 4:3. This represents about 75% of all collected images.

Figure 3 shows an example of a query image, the bins used to compute the image descriptor and the closest matching images from a dataset of 10,000 street images. The images returned belong to similar scene categories and have similar camera properties (same perspective pattern and similar depth).

### 2.3. Alignment and compositing

Images returned by the scene matcher tend to contain similar scenes, but can still be misaligned as the GIST descriptors matches only the rough spatial structure given by the $6 \times 4$ grid of cells. For example, in the case of city skylines, the horizon line can be at a slightly different height.

To fine tune the alignment, we apply a standard gradient descent alignment [20, 28] minimizing the mean of the sum of square pixel color differences in the overlap region between the two images. To do this, we search over three parameters: translation offset in both the $x$ and $y$ direction and scale. The alignment is performed on images down-sampled to $1/6$ of their original resolution. In the case of translations and zoom, the alignment search is initialized by the synthesized view transformation. In the case of camera rotation we initialize the alignment with a translation in the x direction matching the image overlap induced by the rotation, e.g. half the image width for the example in middle column of figure 2. As a result, the camera rotation is approximated by a translation and scale in the image domain. The camera rotation is only used in the scene matching to induce a change in the geometry of the scene.

The aligned images are blended along a seam in their re-

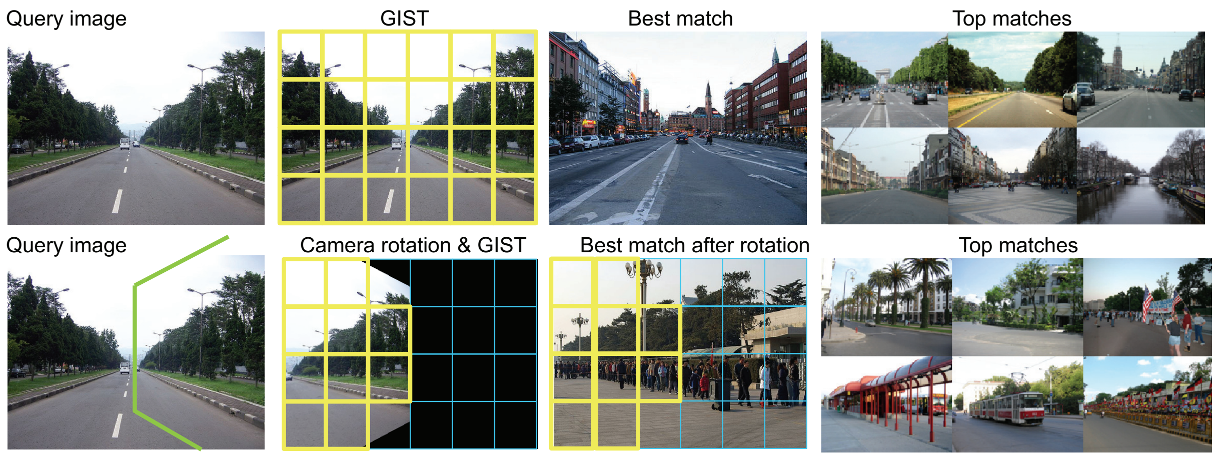

Figure 3: Each row shows the input image, the $6 \times 4$ spatial bins used to compute the GIST description, the best match on a dataset of 10,000 images, and the next 6 best matches. Top row: we look for images that match the full GIST descriptor. Bottom row: Result of a query after simulating a camera rotation. The returned images contain a new perspective, similar to the one that we will have obtained by rotating the camera 45 degrees to the right.

gion of overlap using Poisson blending [23]. In the case of camera translation and rotation we find a seam running from the top to the bottom of the overlap region minimizing the sum of absolute pixel color differences by solving a dynamic program [10]. In the case of zoom, where images are within each other, we find four seams, one for each side of the overlap region. In addition, to preserve a significant portion of the zoomed-in image, we constrain each seam to lie close to the boundary of the overlap region. Finally, images are blended in the gradient domain using the Poisson solver of [2].

To obtain a sequence of images with a particular camera motion the above process is applied iteratively using the most recently added image as a query.

## 2.4. Estimation of camera parameters

Although we do not perform a full estimation of the 3D geometry of the scene structure, we show that the estimation of some simple camera parameters can improve the quality of results when navigating through the collection. Specifically, we show how to estimate 3D camera properties from a single image using the GIST descriptor. Here we follow [18, 30] in which the estimation of the horizon line is posed as a regression problem. As opposed to [7, 9] in which camera orientation is estimated by an explicit model of the scene geometry, we use machine learning to train a regressor. The main advantage of our method is that it works even when the scene lacks clear 3D features such as a vanishing point. We collected $3,000$ training images for which we entered the location of the horizon line manually (for pictures taken by a person standing on the ground, the horizon line can be approximated by the average vertical location of all the faces present in the image). We use weighted mixture of linear regressors [12] to estimate the location of the horizon line

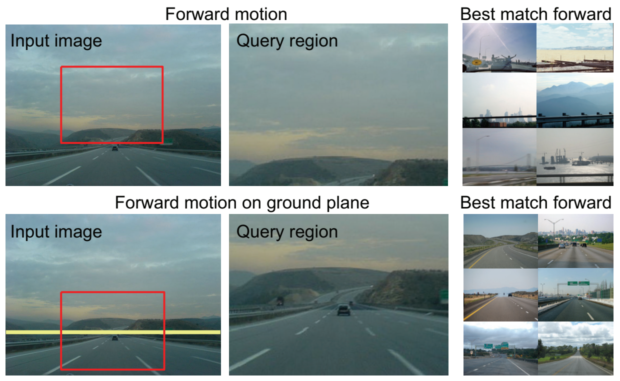

Figure 4: The top row shows the queried region when we take the central image portion. The bottom row shows the results obtained when centering the queried region on the horizon line. The retrieved images contain roads with similar perspective to the input image.

from the GIST features as described in [30].

Figure 4 illustrates the importance of estimating the location of the horizon line. In this example, we want forward motion to represent a person walking on the ground plane. As shown in the top row, if we zoom into the picture by using the image center as the focus of expansion, we move into the sky region. However, if we zoom in on the horizon line we simulate a person moving on the ground plane. In order to generate a motion that simulates a person moving on the ground plane it is important to keep the horizon line within the queried region.

## 2.5. Organizing pictures into themes

When performing a query after a camera motion, the information available for matching is weaker than the original GIST descriptor (due to the smaller image overlap after the transformation). This results in semantically incoherent

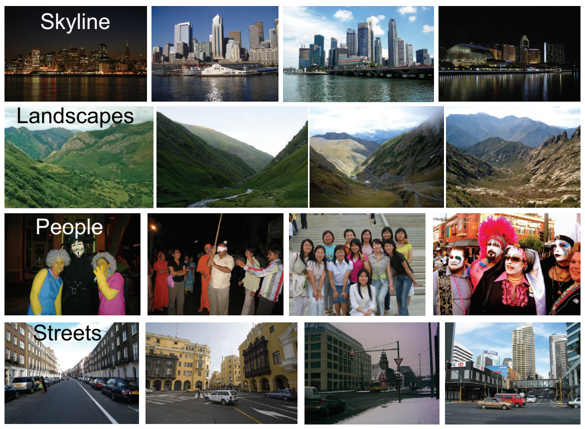

Figure 5: Example of images belonging to different scene themes. Partitioning a large collection of images improves the quality of the results. The classification is done automatically. When navigating through the image collection, it is important to keep the themes constant when moving from one picture to another to avoid undesired transitions. The user can also allow transitions across themes.

transitions between images when creating long sequences (in fact, out of 10 sequences none of them produced satisfactory results when using a random set of 0.5 million images). In order to enhance the GIST descriptor, we use the theme as an additional constraint. The theme of an image is, in most cases, invariant with respect to camera motions. Therefore, we can use this as an additional constraint by only matching pictures that belong to the same theme. Using the themes results in dramatically improved results. It also lowers the memory and CPU requirements as only part of the database needs to be searched. Examples of themes we consider in this work are shown in figure 5.

To obtain a visually and semantically coherent set of images for each theme we train theme classifiers from manually labeled training data in a manner similar to [5, 19]. For each theme we train 1-vs-all linear SVM classifier from about 1,000 positive and 2,000 negative training images. We have developed a suitable labeling interface so that the classifier can be trained interactively. At each iteration the user can visually assess the classifier performance and label additional images if needed. At each iteration, the most uncertain images are shown to the user for labeling.

## 3. Navigating the virtual 3D space

We process the image database and create a graph where the nodes represent images and there are different types of (directed) edges, corresponding to different types of motion, connecting the nodes. The weight of each edge corresponds to the matching cost between the two images and the particular camera motion associated with this edge. We typically retain only the top 10 matches for every particular motion (Figure 6). This gives us a sparse graph that allows our system to tour the virtual space efficiently. The main bottleneck

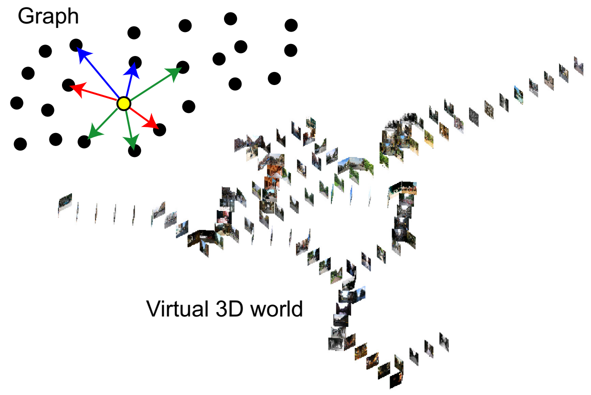

Figure 6: In the original graph, each image is a node and there are different types of edges (color coded in the upper left graph) that correspond to different camera motions. For each motion, there are many good matches and we typically keep just the top 10 for each edge type. All the images of a collection can by organized in a virtual world by greedily assigning to each image the best neighbors and camera motions that minimize the matching cost.

in processing the database is to compute the GIST descriptors for every image, which we do at about 5 images per second. Once the image database is processed and the graph is constructed the user can start navigating in it using intuitive 3D controls. There are different modes of operation, all relying on the same basic machinery. Given a query image, the user can interactively tour the virtual space, ask for a predefined path or generate a tour between two images using the image taxi metaphor. It takes about a second to retrieve the matching images, in our matlab implementation, and a couple of seconds to align and blend the retrieved images with the query image.

### 3.1. Sequences with different camera transformations

The user selects an image that serves as his gateway to the virtual 3D space and starts navigating interactively in a particular theme using intuitive 3D commands (move left/right, zoom, rotate). The system currently can either choose the top matching image automatically or let the user choose the best one from the top 5 matches that were found. The tour can be recorded and then shown as a video, seamlessly moving between different images.

In some cases, users are interested in a particular type of tour. For example, the user might be interested in generating a tour that starts with a short forward motion, followed by a long left translation and concluding with a right rotation. The system will then query the graph and construct a video sequence that satisfies the user's request. In addition, the user can choose the image taxi option. In this case, the user only specifies the first and last images of the tour and the image taxi takes the user along the shortest path in the graph connecting these two images.

Figures 7, 8(a) and 8(b) show image sequences for camera zoom, translation and rotation respectively obtained from a 'street' theme. Note that the left/right translation motion tends to preserve the camera orientation with respect to the scene (the scene remains roughly fronto-parallel), while the rotation motion induces a change in perspective between consecutive images (Fig 9). The translation and rotation sequence were produced fully automatically. The zoom-in sequence was produced interactively by letting the user specify the zoom-in direction - toward the horizon of the image.

The reader is encouraged to view the accompanied video that presents the different tours generated by our system. All the sequences in the video were generated automatically, except for the "Hollywood" and "Windows" sequence that had the user choose the best match out of the top 5 candidates.

To generate the motion videos we produce the composite (including Poisson blending) of each image of the sequence with its neighboring images. The intermediate frames of the motion are synthesized from these composited images by temporal blending to hide small color differences resulting from Poisson blending subsets of images independently. In the case of translation and rotation we look at only one image to each side of the current image. In the case of zoom, a particular composited image can contain pixels from several images ahead, as illustrated in figure 7(d). The translation is simulated by translating with constant per pixel speed between consecutive images of the sequence (also applying small scale changes if the consecutive images are of different size). The case of rotation is similar to translation but, in addition, we display the resulting images on a cylinder. In the case of zoom we apply constant scale change between consecutive frames of the video to create an impression of a person walking at a constant speed in the 3D world. The zoom is performed toward the center of the next image in the sequence, which can result in (typically small) changes in zoom direction between consecutive images.

**Infinite panoramas and the infinite street**: As a byproduct of our system we can construct infinite panoramas by automatically moving the camera at fixed intervals. Fig. 7 gives an example of an infinite perspective street generated with our system. Figure 10 shows examples of long panoramas created for various themes.

**The image taxi: finding a path between two images**: The image taxi allows the user to specify start and end images and let the system take him on a tour along the shortest path between them. Given two input images, we first connect them to the graph and then find the shortest path between them using the Dijkstra algorithm [6]. We then follow the path creating a tour based on the different edges along the way.

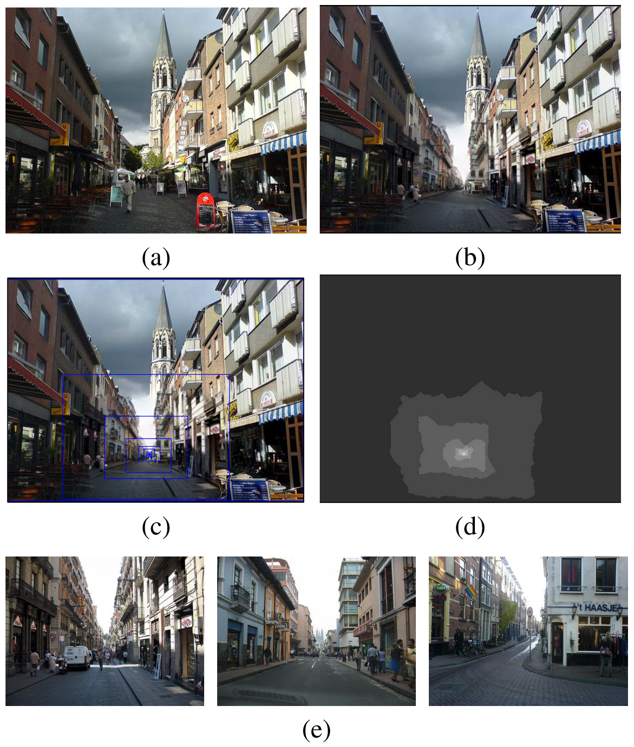

Figure 7: Zoom sequence of 16 images. (a) The query image. (b) The query image composited with the consecutive images in the sequence. (c) Image boundaries of the following images in the sequence overlaid over the composited image (b). (d) Masks indicating areas in (b) coming from different images. (e) Images 2–4 used to generate the zoom sequence. Each consecutive image was obtained by inducing camera zoom as illustrated in figure 2. The zoom direction was indicated interactively by clicking on the horizon line.

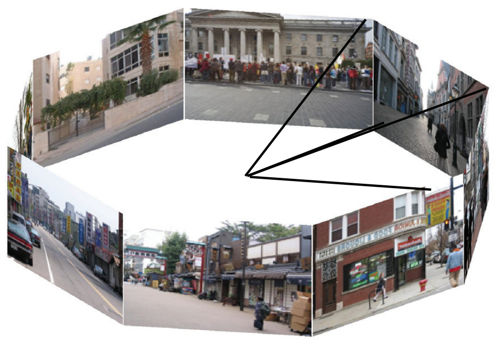

Figure 9: A camera rotation can be used to generate from a single picture a good guess of the surrounding environment not covered by the camera. Here, the system is hallucinating what could be behind of the camera (original image marked with a frustrom). Note that the back of the camera is also a perspective street, aligned with the camera view.

## 4. Limitations and Conclusion

In the course of developing the system we have learned about some of its limitations. First, the system is as good

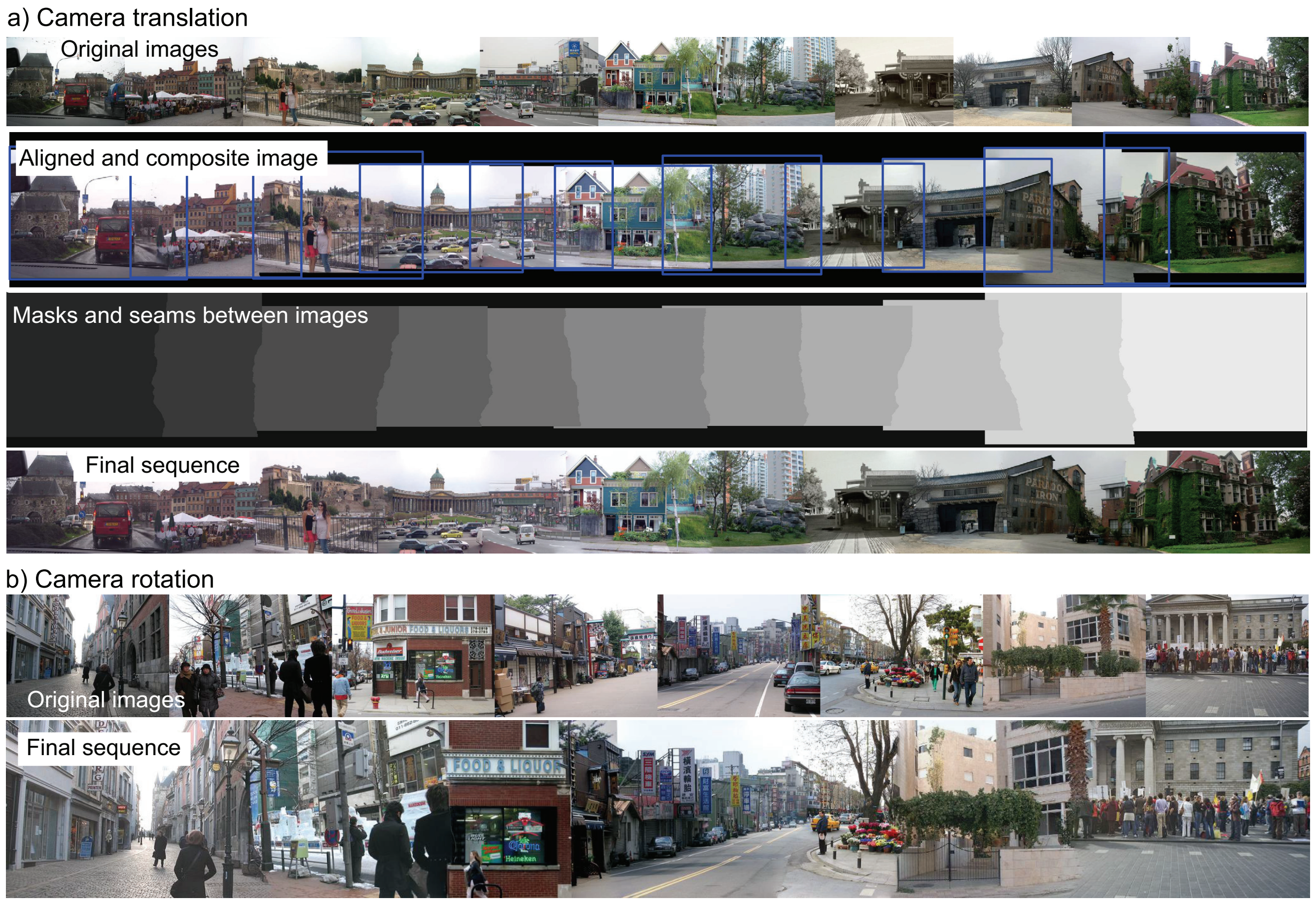

Figure 8: This figure illustrates sequences generated from a large database of street images by inducing two camera motions: a) Camera translation. Each consecutive image was obtained by inducing camera translation by half the image size as illustrated in figure 2. b) Camera rotation. Each consecutive image was obtained by inducing camera rotation of 45 degrees as illustrated in figure 2. This produces changes in the perspective between consecutive images.

as the underlying database is. The larger the database, the better the results. We also found that there are three typical failure modes. The first is when a semantically wrong image is retrieved. This is mitigated by the use of themes but still remains a problem for future research. Second, compositing two distinct images is always a challenge and at times, the seam is quite visible. Finally, there are cases in which the seam runs through important objects in the image which produces noticeable artifacts.

Nevertheless, the proposed system offers an intuitive 3D-based navigation approach to browsing large photo collections. This can be used to create photorealistic visual content for large online virtual 3D worlds like Second Life, computer games or movies. Our system arranges the images into themes and relies on image content to retrieve the next image. Another novelty of our system, as opposed to existing image retrieval systems, is the use of transformed image retrieval. We first transform the query image, before performing the query, to simulate various camera motions. Our infrastructure also allows us to create infinite panoramas or use the image taxi to generate tailor-made tours in the virtual 3D space. These applications can find use in games, movies and other media creation processes.

## References

[1] A. Agarwala, M. Agrawala, M. Cohen, D. Salesin, and R. Szeliski. Photographing long scenes with multi-viewpoint panoramas. *ACM Trans. Graph.*, 2006.

[2] A. Agrawal, R. Raskar, and R. Chellappa. What is the range of surface reconstructions from a gradient field? In *Proc. European Conf. Computer Vision*, 2006.

[3] S. Avidan and A. Shamir. Seam carving for content-aware image retargeting. *ACM Trans. Graph.*, 2007.

[4] S. Bae, S. Paris, and F. Durand. Two-scale tone management for photographic look. *ACM Transactions on Graphics*, 25(3):637–645, 2006.

[5] A. Bosch, A. Zisserman, and X. Munoz. Scene classification using a hybrid generative/discriminative approach. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 30(4), 2008.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, 2001.

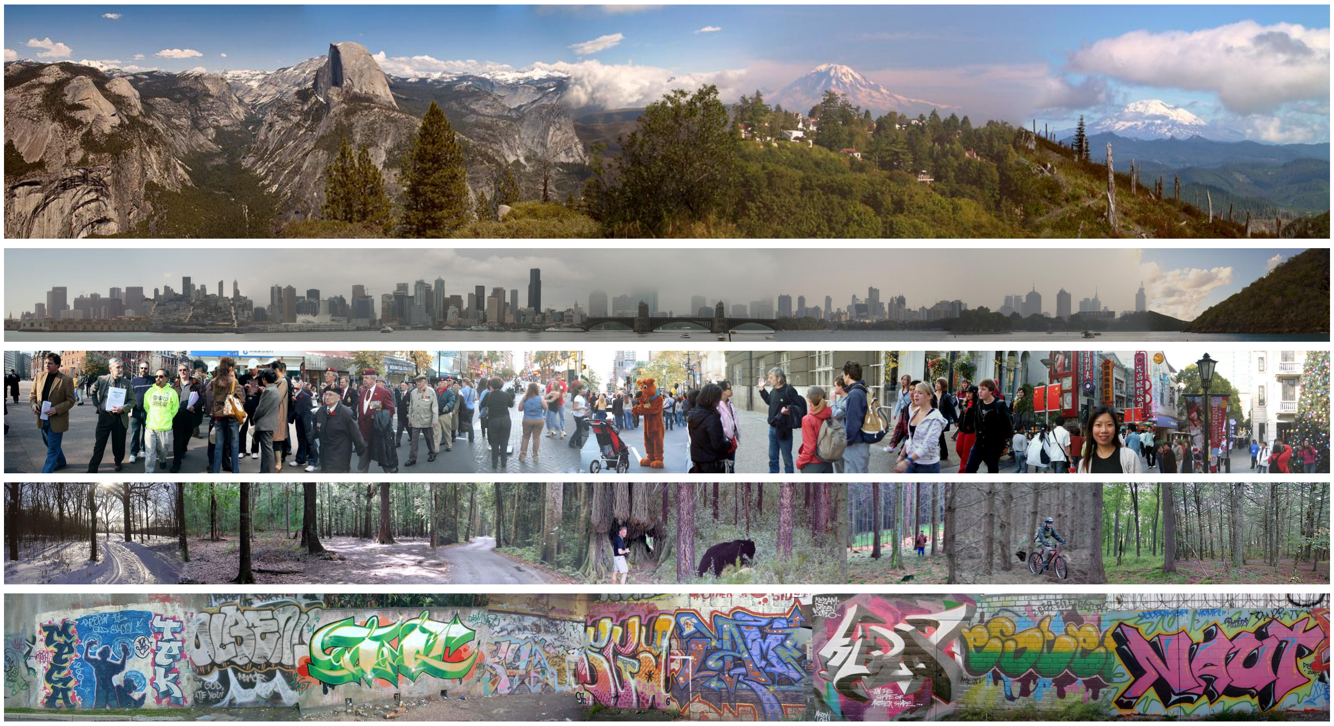

Figure 10: Various panoramas created by our system. The top two panoramas were created automatically from the 'landscape' and 'skyline' themes respectively. The bottom three panoramas were created interactively from the 'people' 'forest' and 'graffiti' themes respectively.

[7] J. Coughlan and A. L. Yuille. Manhattan world: Orientation and outlier detection by bayesian inference. *Neural Computation*, 15:1063–1088, 2003.

[8] J. S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proc. ACM SIGGRAPH*, 1997.

[9] J. Deutscher, M. Isard, and J. MacCormick. Automatic camera calibration from a single manhattan image. In *Proc. European Conf. Computer Vision*, pages 175–188, 2002.

[10] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proc. ACM SIGGRAPH*, 2001.

[11] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics & Applications*, 22(2):56–65, 2002.

[12] N. Gershenfeld. *The nature of mathematical modeling*. Cambridge University, 1998.

[13] J. Hays and A. A. Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics*, 26(3):4:1–4:7, 2007.

[14] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proc. ACM SIGGRAPH*, pages 327–340, 2001.

[15] H. D. J. and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *Proc. ACM SIGGRAPH*, 1995.

[16] J. Kopf, M. Uyttendaele, O. Deussen, and M. Cohen. Capturing and viewing gigapixel images. *ACM Transactions on Graphics*, 26(3), 2007.

[17] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: image and video synthesis using graph cuts. In *Proc. ACM SIGGRAPH*, 2003.

[18] J. F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi. Photo clip art. *ACM Transactions on Graphics*, 26(3):3:1–3:10, 2007.

[19] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006.

[20] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the 7th Int. Joint Conf. on Artificial Intelligence*, pages 674–679, 1981.

[21] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *Int. Journal of Computer Vision*, 42(3):145–175, 2001.

[22] S. Peleg, B. Rousso, A. Rav-Acha, and A. Zomet. Mosaicing on adaptive manifolds. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(10):1144–1154, 2000.

[23] P. Perez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. Graph.*, 22(3):313–318, 2003.

[24] P. Rademacher and G. Bishop. Multiple-center-of-projection images. *ACM Trans. Graph.*, 1998.

[25] C. Rother, L. Bordeaux, Y. Hamadi, and A. Blake. Autocollage. In *Proc. ACM SIGGRAPH*, 2006.

[26] Y. Rubner, C. Tomasi, and L. J. Guibas. Adaptive color-image embeddings for database navigation. In *'Asian Conference on Computer Vision*, pages 104–111, 1998.

[27] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *Proc. ACM SIGGRAPH*, 2006.

[28] R. Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends in Computer Graphics and Computer Vision*, 2006.

[29] A. Torralba, R. Fergus, and W. T. Freeman. Tiny images. Technical Report MIT-CSAIL-TR-2007-024, MIT, 2007.

[30] A. Torralba and P. Sinha. Statistical context priming for object detection. In *Proc. IEEE Int. Conf. Computer Vision*, pages 763–770, 2001.

[31] Y. Weiss and W. Freeman. What makes a good model of natural images? In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007.

[32] D. N. Wood, A. Finkelstein, J. F. Hughes, C. E. Thayer, and D. Salesin. Multiperspective panoramas for cel animation. *ACM Trans. Graph.*, 1997.