Saliency-Assisted Navigation of Very Large Landscape Images

Cheuk Yiu Ip and Amitabh Varshney, *Fellow, IEEE*

Abstract—The field of visualization has addressed navigation of very large datasets, usually meshes and volumes. Significantly less attention has been devoted to the issues surrounding navigation of very large images. In the last few years the explosive growth in the resolution of camera sensors and robotic image acquisition techniques has widened the gap between the display and image resolutions to three orders of magnitude or more. This paper presents the first steps towards navigation of very large images, particularly landscape images, from an interactive visualization perspective. The grand challenge in navigation of very large images is identifying regions of potential interest. In this paper we outline a three-step approach. In the first step we use multi-scale saliency to narrow down the potential areas of interest. In the second step we outline a method based on statistical signatures to further cull out regions of high conformity. In the final step we allow a user to interactively identify the exceptional regions of high interest that merit further attention. We show that our approach of progressive elicitation is fast and allows rapid identification of regions of interest. Unlike previous work in this area, our approach is scalable and computationally reasonable on very large images. We validate the results of our approach by comparing them to user-tagged regions of interest on several very large landscape images from the Internet.

Index Terms—Image Saliency, Very Large Scale Images, Scene Perception, Interactive Visualization, Anomaly Detection, Guided Interaction.

1 Introduction

We are seeing a significant growth in the interest and relevance of very large images. One of the reasons behind this trend is the development of systems that can automatically capture and stitch photographs to create images of unprecedented detail ranging from a few gigapixels [8, 27, 46] to even a few terapixels [12]. Recent advances in consumer-grade robotic image acquisition from companies such as Gigapan have further energized social network communities that are interested in building, sharing, and collectively exploring such large images. Some relevant work on processing of very large images includes a streaming multigrid solver for gigapixel scale out-of-core gradient-domain image processing by Kazhdan and Hoppe [22]. More recently, Summa *et al.* [40] present progressive processing of high-resolution images with interactive previews. Kopf *et al.* [27] discuss how to naturally display the stitched image by cylindrical projections. Luan *et al.* [32] adaptively annotate these very large images by text and audio according to the viewing position and scale. While these are very interesting first steps in computational processing and display of very large images, this paper addresses a different challenge for such large images – their effective visual navigation.

Consider a gigapixel image shown in Fig. 1 . The successive zooms give an indication of the level of detail in such images. When viewing such images, users typically pan at the coarse level and occasionally zoom in to see the fine details. Panning at the finest level of detail is too tedious and panning at the coarsest level of detail does not have enough information for the user to know where to zoom in. Just to convey the magnitude of the problem, let us consider some numbers. Imagine a user is visualizing a 4 Gigapixel image on a 2 Megapixel monitor. This would suggest that every monitor pixel is representing 2000 image pixels and the observable image on the monitor is a mere 0.05% of the total dataset. Further, if it takes a user just a couple of seconds to scan the monitor, it will take more than an hour to scan through the entire image.

In this paper, we leverage principles of visualization to ease the task of navigating very large landscape images. In visual exploration of very large images the biggest challenge involves identifying the most salient content and visually presenting this information to the user. Just as the transfer function design in traditional visualization uses opacity to identify what data to show and color to emphasize, we present data-driven techniques to identify and emphasize potential areas of interest in very large images. Our techniques are relevant for visualization of datasets when the data size is several orders of magnitude larger than what the display device can accomodate. We use techniques based on visual knowledge discovery to help in user navigation and adaptive context- and scale-dependent visual overlays to assist in spatial localization of salient detail.

Challenges: The interactive visual exploration of very high-resolution large-scale images presents three challenges:

Visual Scalability: The visual scalability challenge arises from the inability of the human visual system to take in all the details that are present in a very large image. This arises from a fundamental limitation of the retina as well as the display hardware which have not kept pace with our ability to acquire ever larger images. Fig. 2 shows the growth in resolution of the mainstream consumer-grade LCD displays against camera sensors in recent years. The display resolutions correspond to the highest-resolution monitors sold by a mainstream vendor (such as Dell and Apple) and the camera-sensor sizes correspond to the highest-resolution entry-level SLR cameras manufactured by Canon, Nikon, or Sony. The resolution growth of these off-the-shelf cameras has clearly outpaced the resolution of the display monitors.

Information Scalability: The challenge here is to design effective computational algorithms to identify nuggets of useful visual information that hide in large-scale images. In very large images, most of the image data is innocuous and unimportant and even considering it wastes precious time and resources. Often relatively small regions in such very large images are accorded a very high information value by human observers. Identifying informative regions in very large images that match human expectations is an ambitious challenge.

Data Scalability: The sheer data size of these images poses a computational challenge. Processing such large images along with their auxiliary data structures often necessitate out-of-core methods as well as designing of algorithms that are cache- and memory-efficient. Even routine image-processing operations for very large images require a careful mapping to the many-core and multi-core processor architectures for any reasonable performance.

• *Cheuk Yiu Ip and Amitabh Varshney are with the Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, E-mail:{ipcy,varshney}@cs.umd.edu.*

Manuscript received 31 March 2011; accepted 1 August 2011; posted online 23 October 2011; mailed on 14 October 2011.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

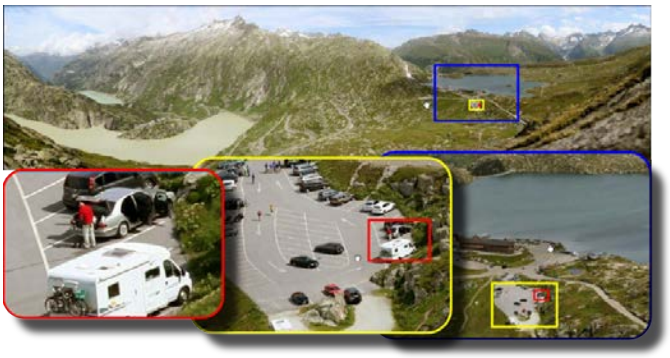


Fig. 1. A very large image contains fine details. We progressively zoom into the blue, yellow, and red regions in the panorama. There are interesting regions at different scales: The overview panorama shows the landscape. The blue region shows the hotel and parking lot. The yellow region shows the cars. The red region shows a human. Note the red region is less than one pixel at the overview scale.

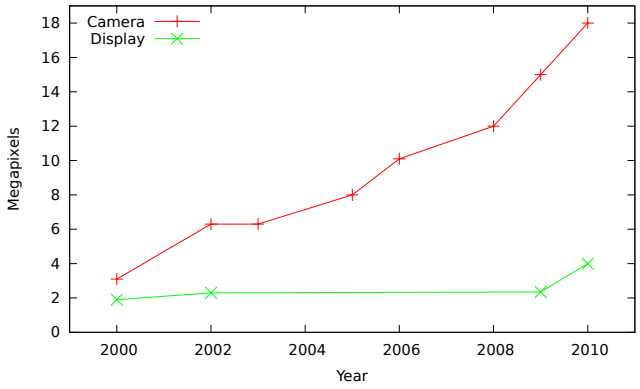


Fig. 2. The growth of camera sensor resolution vs display resolution.

Contributions: In this paper we present the first steps towards addressing the above challenges for interactive visual exploration of very large images. We outline our main contributions next.

1. We extend classical computational image saliency to very large images. Users often navigate across three or more orders of magnitude scale differences – from the overview to the finest-scale views while viewing a very large image. Classical algorithms for multi-scale image saliency break down at handling such a large span of visual scales. We discuss the issues involved and present a solution in Section 4.

2. As discussed above in the information scalability challenge, it is important to identify the regions of interest in very large images that characterize areas of high information value to users. The question of how to effectively characterize visual information content is still far from settled. There are a number of measures of visual information content and often the definition depends on the task at hand. Our goal is not to provide a definitive characterization of the visual information content of a region of an image, which is a very deep question related to the issues of task semantics and knowledge. Instead we present here a fairly general information discovery algorithm in Section 5, that can serve as a framework for further research with other measures of information content.

3. Interactive visual exploration of very large images requires a careful balancing of computational analysis and user preferences. Too much reliance on automatic intelligence-extraction algorithms is currently not feasible since it is often very difficult to codify semantics of what a user is looking for. At the same time a purely interactive visual exploration without any computational assistance proves to be tedious and overwhelming due to the sheer scope of the data that is being visualized. We present an interactive visual exploration and information discovery system in Section 6.

4. Data scalability is an important issue when dealing with very large images and we present advances in this area in Section 7.

5. We present and compare our results with those from a social community of gigapixel image enthusiasts in Section 8.

2 Related Work

Over the years a number of techniques have evolved to address limitations of the display medium with respect to the visual data. If the input data has more bits of color information than those that can be displayed, we use tone mapping to approximate the appearance of high-dynamic range images [10]. This involves mapping the color space from a high-dimensional input to a low-dimensional output, while preserving most of the salient content. Similarly, video summarization techniques [9] typically involve extraction of the salient key-frames with some context that results in warping of the temporal space. Recent research has also looked at how to carry out image warping so that images that are larger than the display can be adaptively resized to preserve their most salient content [1, 37]. Most recently, Laffont *et al.* [28] present content-aware zooming on images up to 16 megapixels. While such image re-targeting techniques could be effective for images that are an order of magnitude larger than the displays, they do not scale well to three orders of magnitude or more that we desire. In this paper we present distortion-free navigation of very large images, assisted by identification of their most salient content, to allow viewing of very large images on displays of modest sizes. Unlike the previously discussed methods, visualization by navigation maintains the metric fidelity of the image. The pan and zoom interactions allow the users to follow the local image context naturally.

2.1 Scene Analysis and Image Saliency

Image saliency has been used in modeling visual attention. Top-down and bottom-up models for building a saliency map have been introduced by Tsotsos *et al.* [42], Itti *et al.* [17] and several others. Oliva *et al.* [35] apply the top-down model to object detection.

Recent approaches include work by Hou and Zhang [14] that computes image saliency by the difference of the image's original and smoothed log-Fourier spectrum. Bruce *et al.* [4] learn a set of sparse code from example images to evaluate saliency of new images. Wang *et al.* [44] use random graph walk on image pixels to compute image saliency. Goferman *et al.* [11] consider visual organization and high level features such as human faces in saliency computation.

In spite of the impressive advances in the recognition of specific objects, such as buildings, cars, and humans, general scene understanding remains a hard problem [13]. State-of-the-art systems [7, 31, 38] train on web-scale databases of small images and then extract regions of interests for test scenes in a supervised manner. More recently, Kim and Torralba [23] use alternating optimization on sets of unlabeled images to extract one to three regions of interest per image. Our system needs a more general approach since we expect a variety of regions and objects of interest in a very large image.

Although there has been extensive previous work in identifying salient regions using several methods, such techniques typically extract a very small number of regions from a relatively small image. For instance, Goferman *et al.*'s [11] program requires 74 seconds to process a 250×142 image, and Bruce *et al.*'s [4] program requires 30 minutes to process a 3000×1500 image. These timings are on a current state-of-the-art workstation described in Section 8. Assuming their running time scales linearly and memory swapping does not become an issue, these approaches will take hundreds of hours to process gigapixel-sized images. The purpose of this is not to downplay

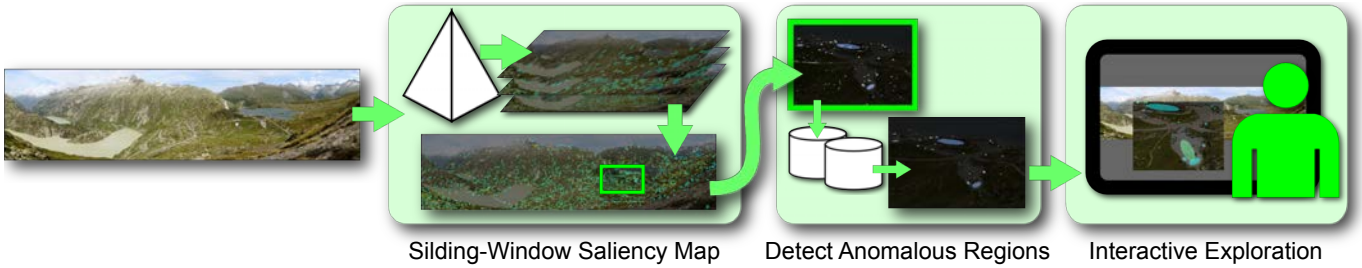


Fig. 3. Our system discovers regions of interests in very large images and assists user exploration. In the first step we build a saliency map by augmenting the traditional multi-scale image saliency approach with a sliding window over scales. In the second step we carry out information discovery by using color descriptors to identify the most unique regions. In the third step we facilitate rapid elimination of false positives through user interaction during visualization.

the achievements of these approaches, which are actually quite impressive in what they are targeting, rather to highlight the difference in their approaches from ours.

We believe user interaction in the visualization process can greatly assist in rapid culling of false positives and can greatly enhance the overall computational efficiency of the resulting algorithms. Our approach adopts a three-step process of progressive culling of potential regions of interest. We believe this provides a better balance of accuracy and computational efficiency with a user in the middle than a purely computational approach.

2.2 Visual Data Analysis

There is a rich history of data analysis for visual summarization and identification of information-rich subspaces for effective visual presentation. We next present a few example of how salience is defined and used for visual datasets to enhance their depiction. Notable advances include defining saliency for polygonal meshes [30], comparing it to human eye movements [26], and illuminating meshes based on saliency [29]. Howlett *et al.* [15] use eye-tracking data to identify salient features on meshes and carry out user studies to validate their findings. Kim *et al.* present and validate saliency-based enhancement operators to guide visual attention in volume visualization [24] and geometric meshes [25].

Machiraju *et al.* [33] present a system to detect contextually significant multiscale features in very large datasets directly in the wavelet domain and visualize them progressively. Bordoloi and Shen [3] select informative views of volumetric data based on saliency defined using entropy measures. Viola *et al.* [43] determine the most expressive view for a selected region of interest in a volume using mutual information. Bruckner and Möller [5] use isosurface similarity maps based on mutual information to automatically select the most salient isosurfaces. Saliency-based summarization of time-varying datasets has been carried out for videos [9] and molecular dynamics simulations [36]. Wiebel *et al.* [45] identify salience with the vortices that originate from walls in three-dimensional time-dependent vector fields and track their evolution using generalized streak lines. Unlike much of the previous work on volumetric or time-varying data, the focus of this paper is on visualization of very large images.

3 Overview of Our Approach

The goal of our paper is to carry out computationally-assisted navigation of very large images. We leverage the principles of visual saliency and statistical similarity to outline a three-step approach. Fig. 3 shows an overview of our approach.

Our first goal is to identify regions of interesting detail in very large images. The challenges here are in dealing with the dramatic span of visual scales and the sheer amount of data as well as information. Our approach addresses each of them.

Visual Scalability: Traditional algorithms for multi-scale image saliency work well for small images up to a few megapixels but do not scale up well to gigapixels and beyond. To address this we augment the traditional multi-scale image saliency approach with a sliding window over scales to effectively work with very large images. Our approach only requires Gaussian convolutions on images. It is highly parallelizable and scales linearly with the size of the image. The sliding-window saliency map phase of our approach discovers thousands of locally salient regions from billions of pixels.

Information Scalability: Typical landscape images comprise of a large number of natural elements such as clouds, rocks, grass, and trees. In this paper we assume that such repeating scene elements are not of interest to the viewers. We characterize all image regions using automatic color-structure descriptors. We then argue that the most interesting regions are the ones that are the most different from their k nearest neighbors (k-NN) in such color-structure feature space. We have empirically observed that this definition works well for landscape images. Other descriptors may be found to be more suitable for other datasets. We use a spatial index to accelerate the k-nearest-neighbor queries. This indexing and querying process grows as $O(n \log n)$, where n is the number of salient regions identified by the sliding-window saliency step. For gigapixel images n is typically of the order of a few tens of thousands. We refer to this step as *anomaly detection*.

Interactive Visual Exploration: We have developed an interactive visualization environment to assist in exploration of very large images. We facilitate users to be aware of details that are a fraction of the screen pixel by ensuring that the overlays for such regions are large enough to be visible at every scale. The users can then explore and inspect all such regions interactively. We also have an automatic mode of the system in which the users are led through a smooth camera fly-through over all the informative regions in the image in order of their uniqueness as determined by the Visual and Information Scalability stages of the algorithm. If a user comes across a region of the image that the system claims is important, but the user finds to be unimportant, the user can identify all similar regions (using a slider) and discard them interactively. The spatial index structure built to address the Information Scalability stage of the algorithm allows this operation to occur within a few milliseconds.

Data Scalability: The size of the very large images together with their auxiliary data-structures imposes a significant computational and storage burden. We have used a number of approaches to ameliorate this problem including out-of-core computation, efficient storage of salient regions, and building the image viewer using a clipmap-based tiled approach.

4 Sliding-Window Saliency

A very high resolution image is particularly interesting because it exceeds what the human eye can see at a given spot. In a very large image we can zoom from seeing the big picture to scrutinizing the finest details. Figure 4 shows three different views of an image at three different scales. We note that the cars seen in Figure 4(c) are not discernible in Figure 4(a).

4.1 Traditional Image Saliency

A saliency map shows which part of an image is likely to attract the most attention of the low-level human visual system. Itti *et al.* [17]

(a) (b) (c)

Fig. 4. We zoom into a very large image (a) to see the lake and the nearby terrain, (b) to discover the hotel and parking lot (b) and then go still further (c) to see the cars in the parking lot.

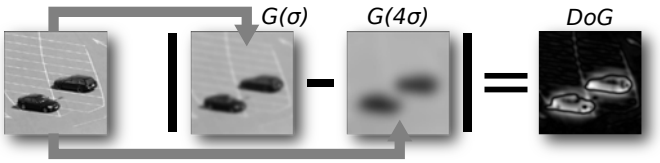


Fig. 5. Computational saliency mimics the contrast detection mechanism in the human retina. This image shows how Difference of Gaussians (*DoG*) operator detects the high contrast cars instead of the low contrast parking grids.

have proposed a computational model of visual saliency by using multiscale image processing. Multiscale image processing techniques analyze an image at different scales to simulate the retinal receptive fields. Their image saliency model aggregates the results from three features of an image – intensity, color opponencies, and orientation. We have found that the use of the orientation features decreases the quality of our results as it ends up enhancing naturally occurring structures with strong edges such as cracks in rocks or trees, that end up becoming too salient. In this paper we only consider the intensity and the two color-opponency attributes for computing image saliency. The intensity (F_I) is the average of primary colors, red, green, and blue. The color opponency attribute contains two sub channels, Red-Green(F_R) and Blue-Yellow(F_B). The details on computation of these attributes can be found in [17].

We next briefly review the traditional algorithm for computing the saliency map $\mathscr{S}$ of an image.

$$
\begin{aligned}
F_i &\leftarrow \text{Image}, \quad i \in \{I,R,B\} \\
G_{i,j} &= \mathscr{G}(j) \otimes F_i, \quad j \in \sigma, 2\sigma, 4\sigma, 8\sigma, 16\sigma \cdots \\
D_{i,j,k} &= |G_{i,j} - G_{i,k}|, \quad k \in \{4j, 8j\} \\
N_I &= \sum_k \sum_j \mathscr{N}(D_{I,j,k}) \\
N_C &= \sum_k \sum_j [\mathscr{N}(D_{R,j,k}) + \mathscr{N}(D_{B,j,k})] \\
\mathscr{S} &= \frac{1}{2}[\mathscr{N}(N_I) + \mathscr{N}(N_c)] \qquad (1)
\end{aligned}
$$

We first extract the intensity feature, F_I, and color features F_R, F_B from an image. Then, we convolve the feature images F_i with Gaussian kernels, $\mathscr{G}$, at different scales j. We find contrasting regions by computing the difference of Gaussians (*DoG*) images at each scale, $G_{i,j}$. We compute the *DoG* images at scales $\{\sigma, 4\sigma\}, \{\sigma, 8\sigma\}$. The *DoG* operation mimics the contrast detecting receptive fields of retinal ganglion cells. Fig. 5 shows the *DoG* operation extracts contrasting cars from the background. $\mathscr{N}$ is a normalization function that promotes the peak salient regions [16]. The saliency map, $\mathscr{S}$, is an aggregation of the normalized *DoG* images. We use $\sigma = 2.0$ in our experiments.

4.2 Sliding Window Aggregation

To see the difficulties introduced by the traditional saliency method, consider the saliency map for the parking lot image in Fig. 4 computed by Itti's *et al.* [17] method. If we aggregate the saliency at all the levels of detail, we obtain the salient region in Fig. 6(a). We observe that mostly the center of the parking lot has been included, while most of the surrounding cars have been excluded. It is interesting to note that if we analyze the saliency maps at each scale we find that cars are salient at fine scales $\sigma, 2\sigma, 4\sigma$ and the parking lot is salient at scales $16\sigma, 32\sigma, 64\sigma$. This can be seen in Figures 6(b)–(d). Therefore even though the saliency maps at individual scales were able to correctly identify the constituent salient elements, the overall aggregation ended up suppressing a number of them. The reason behind this is that if the salient regions at two different scales overlap, this overlap tends to disproportionately promote the overall salience of that region. Such events are infrequent or otherwise are not of great concern when the image sizes are relatively small. However, this no longer holds true in very large images. As shown in Fig. 4, an observer would recognize the parking lot and the cars quite independently of each other at different scales. Therefore we believe that it is inappropriate to aggregate the saliency of the cars and the parking lot together since they are detected by *DoG* filters that are 16 scales of difference apart. The key observation here is that while simulating the multiscale capabilities of the human visual system, we have to be aware that our eyes have a finite resolution. We should limit the number of scales in multiscale image processing based on the limits of the human visual system.

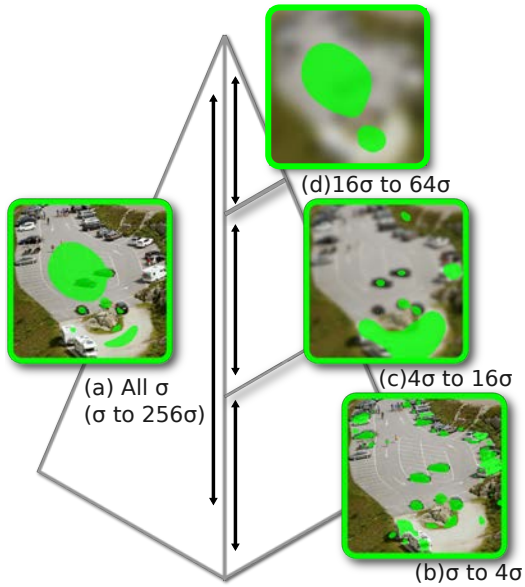


Fig. 6. This shows a comparison of the thresholded Itti *et al.*'s [17] method in (a) and our sliding window saliency maps in (b)-(d). (a) The center cars and the parking lot are salient while the surrounding cars are suppressed (σ to 256σ). (b) All cars are salient (σ to 4σ). (c) The lower part of the parking lot becomes salient with a few cars (4σ to 16σ). (d) Only the parking lot is salient (16σ to 64σ).

To address the above, we introduce a sliding-window approach to build saliency maps at multiple scales with limited aggregation. In this approach by limiting the scales of saliency aggregation we produce multiple maps that simulate the zooming operation. This allows views of drastically different scales to be analyzed virtually independently of each other. For example, we can extract cars from the aggregation of normalized maps at scales $\{\sigma, 2\sigma, 4\sigma\}$. Fig. 6(b) shows the resulting saliency map. It highlights most of the cars without highlighting the parking lot. We separate salient regions at widely different scales by limiting the aggregation procedure. We limit the aggregation of normalized maps from scale j to scale $j+\delta$. We modify the aggregation procedure in equation (1) to equation(2). This sliding window approach ensures overlapping regions from drastically different scales do not interfere with one another.

$$N_{I,j} = \sum_k \sum_j^{j+\delta} \mathcal{N}(D_{I,j,k})$$
$$N_{C,j} = \sum_k \sum_j^{j+\delta} [\mathcal{N}(D_{R,j,k}) + \mathcal{N}(D_{B,j,k})]$$
$$\mathcal{S}_j = \frac{1}{2}[\mathcal{N}(N_{I,j}) + \mathcal{N}(N_{C,j})] \quad (2)$$

We seek a scale difference δ that truly reflects the human visual system. We use $\delta = 4j$ in this paper. We next use arguments from the human visual system theory to suggest why this may be appropriate.

Human Visual System Considerations: We would like to use the scales in multiscale image processing based on the sensitivity difference between foveal and peripheral vision. Perceptual studies have shown that the fovea is the most sensitive region of the retina and the sensitivity drops as the view angle increases. Let us assume that we are viewing an image on a 30-inch monitor at 1 meter. In this case, the view angle subtended from the edge of the monitor to the center of the screen is about 20°. At 20°, our retina retains approximately $1/5^{th}$ of the foveal resolution. Therefore, we have decided to compute the saliency maps with images within the 4 scales ($\sigma - 4\sigma$).

Fig. 7 shows the saliency map resulting from our sliding-window-scale approach for an example image. The computational saliency model detects 18 thousand salient regions. The computational saliency model pre-processes the data efficiently and reduces our quest for interesting and meaningful detail from billions of pixels to thousands of regions. Yet this is still too many regions for a user to manually inspect. We next discuss a more discriminating anomaly detection procedure to refine this initial pool of candidate regions.

5 Information Discovery

It is difficult to quantify the visual information content of a region. Some very interesting advances have been made in this field recently. Jänicke *et al.* [19, 20, 21] have extended the idea of local statistical complexity to measure the local information content of a region. This provides an application-independent, purely mathematical measurement of information. More recently, a very interesting and expansive treatment of how saliency and information theory can be adapted and adopted for visualization has been carried out by Chen and Jänicke [6, 18]. In this paper we wish to slightly side-step the deeply intriguing topic of how to quantify visual information content of a region in the general case, and instead talk about what we have found to work well for large-scale landscape images. We hope that further advances in the field of visual information quantification will seamlessly replace the method that we outline next to work with a wide variety of very large image databases beyond just landscape images.

In this paper we have decided to adopt the approach that the most important regions of an image are those that are the most different from every other region. In other words, we are interested in identifying the outliers, or the anomalies, in a very large image. As seen in Fig. 7, the sliding-window saliency aggregation step identifies a large number of salient regions. They range from patches of grass on the ground, to cracks between the rocks in the mountains.

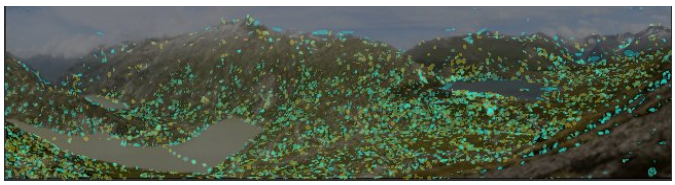


Fig. 7. The saliency map of our example image. The sliding-window saliency map detects 18k regions. (The salient regions are enlarged for visibility)

5.1 Image Region Descriptors

To be able to quantify differences amongst different image regions we need to first identify what we mean by an image *region* and then we need to settle on the space in which such differences will be measured.

To identify an image region, we fit oriented ellipses to the salient regions that were detected in the previous section. We then fit bounding boxes to the ellipses, pad them by a few extra pixels (we have used 20 pixels for all the examples in this paper) to ensure that the padded bounding boxes fully enclose the salient regions. These bounding boxes then represent the image regions of interest.

To achieve rotational invariance, we use histograms of the color-space of the pixels belonging to the region of interest. We have tested a number of color spaces – RGB, HSV, CIELab and found that neither of them were very discriminative. We also experimented with shape and orientation descriptors and found that they were excessively discriminative. This search led us towards a descriptor that would represent both color as well as statistical structural information of an image region and would have a discriminating ability that would lie between the two extremes (color-based and edge-based descriptors).

The MPEG-7 color-structure image descriptor represents both color and structural information. The MPEG-7 is an ISO standard for describing multimedia content data and facilitates information retrieval. It consists of generic descriptors that cover many basic visual features, such as color, texture, and shape. The MPEG-7 color-structure descriptor embeds color structure information into the descriptor by counting color frequencies in a moving window of 8×8 pixels. Color values are represented in the double-coned HMMD color space, which is quantized non-uniformly into 64 bins. The range of histogram is normalized to $0 - 255$. The resulting descriptor is a 64-dimensional vector. We follow the recommendation of MPEG-7 standard and compare them using the Euclidean L_2 norm distance.

$$D(p,q) = \|p - q\|_2 \quad (3)$$

p and q are the descriptors of two image patches and $D(p,q)$ is the distance in between the descriptors. We compute the MPEG-7 color-structure descriptor for each image patch using the software provided by the `BilVideo-7` [2] video indexing and retrieval system.

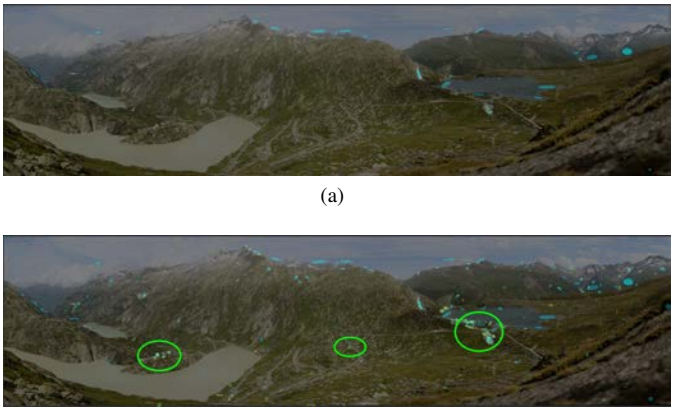
5.2 k-Nearest-Neighbors Anomaly Detection

We compute the uniqueness of each region by considering its k-nearest-neighbors (k-NN). For each image region, we search for its k-nearest-neighbor regions:

$$U(p) = \sum_{i=1}^{k} \frac{D(p,q_i)}{k} \quad \text{where} \quad p, q_i \in P, p \neq q_i$$
$$D(p,q_1) \leq D(p,q_2) \leq \ldots \leq D(p,q_k) \leq D(p,q_{k+1}) \ldots \quad (4)$$

P is a set of image patch descriptors. The uniqueness of image patch p, $U(p)$, is its average distance to its k-nearest neighbors, $q_1 \ldots q_k$. Repetitive regions with many close neighbors have a low average distance. Regions such as humans, signs, or vehicles should be distinct from the other regions and have a high average distance. We identify the unique regions of interest by their high average distances. We select the top 3% of salient regions as the regions of interest in our experiments.

Approximate nearest-neighbor data structures accelerate the k-nearest-neighbor search [39]. The biggest overhead in the k-nearest-neighbors anomaly detection is the need to retrieve k-nearest-neighbors for each region. Linear search of the k-nearest-neighbor queries is computationally expensive. Research in computational geometry provides many spatial data structures to facilitate this nearest neighbor querying. The approximated nearest-neighbors index significantly accelerates this search process. The sum of distances to the top k-approximated nearest-neighbors provides a reliable uniqueness estimate of each region. Our implementation uses a randomized KD-Tree index in the `flann` library [34] through the `OpenCV` library.



(a)

(b)

Fig. 8. We visualize the detected regions with adaptive scaling. (a) Small detected regions are too small to be seen in the macro view. (b) Adaptive scaling ensures the unique regions are enlarged and visible.

Fig. 8(b) shows an image with detected regions overlaid after anomaly detection phase. This process reduces the 18 thousand salient regions to just about 500 anomalous regions.

6 Interactive Visualization

We guide users to explore the image through the detected regions and interactive visualization. We visualize the detected regions, provide automatic fly-through, and allow interactive user refinement. We highlight three features to assist large image exploration.

- Adaptive scaling ensures the regions of interest are visible.
- Automatic exploration guides the user to discover the unique regions of the image.
- User interaction refines the computed detections.

6.1 Visualizing the Detected Regions

We want the users to see the detected regions from the macro view of the image and also allow them to zoom-in to inspect. We believe maintaining the zooming procedure gives the users a much more natural context. Small regions are invisible at the macro view, therefore the corresponding overlay regions are also too small to be seen. We need to visualize these regions more effectively.

We adaptively scale the overlay tags on the detected regions to ensure the most unique regions are visible. We compute the scale, λ, as follows:

$$\lambda(r) = \begin{cases} \frac{s_m}{s_r}\left(1 - \frac{\text{rank}(r)}{\#\text{regions}}\right) & \text{if } s_r < s_m \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

$\lambda(r)$ aims to enlarge the overlay tag for the region r according to the viewing scale and the uniqueness of region r. s_r is region r's size in pixels on the screen. s_m is a user-defined target screen size for the overlay tag. $Rank(r)$ is the relative order of the uniqueness of region r as determined by equation 4 and # *regions* is the number of regions identified at the end of the anomaly detection phase. Thus, the greater the uniqueness of a region r, the greater the $\lambda(r)$ would be. As we zoom into an image, s_r increases and $\lambda(r)$ gradually decreases, and the overlay tags will be shown in their actual size once $s_r \geq s_m$. We color the overlay tags from cyan to magenta (the most unique) according to their rank of uniqueness (equation 4).

Fig. 8 shows the example image and the overlay tags on the detected regions. We see very few detected regions in Fig. 8(a) because they are too small to be seen. We scale the detected regions in Fig. 8(b) by $\lambda(r)$. Fig. 8 (b) shows two groups of regions on the left and on the right. The center region corresponds to a single salient car on the road.

6.2 Automatic Exploration

Our system guides the users to fly through the detected regions. This helps the users to start exploring the image when they know little about it. It smoothly pans and zooms into the detected regions according to their uniqueness. We achieve this by sorting the regions in descending order of their uniqueness (equation 4). This forms a natural exploration sequence that starts from the most unique regions. If users come across misidentified regions during the fly-through, they can suppress the regions by interactive refinement.

6.3 Interactive Refinement

Automated systems recognize a lot of regions of interest but they may also make mistakes. Our system allows the users to refine the results by selecting misidentified regions and deleting them. This mechanism allows the users to refine the automatic result but it can be tedious when users need to delete multiple regions. We provide interactions in our system to batch delete misidentified similar regions.

The user may delete a batch of misidentified regions that are similar in a single interaction. Fig. 9 illustrates this mechanism. This select-slide-delete mechanism can remove many misidentified similar regions at once. In the first step the user identifies a set of regions that in the opinion of the user have been misidentified by the system. Amongst these regions, the user selects one representative region in the second step. This is highlighted by the system. In the third step the user adjusts a slider control to change the similarity distance from the one misidentified representative region to others that are like it. The system interactively highlights other regions that it detects to be similar to the misidentified region. Once the highlighted regions match the user's intent, they can be deleted all at once.

The spatial index and color-structure descriptors of regions provide the interactive search capability. The anomaly detection spatial index is used in this step to provide fast similarity queries for regions. The system expands the region selection by querying the spatial index for regions that are similar to the user selection. The slider thresholds the number of regions retrieved. The spatial index performs a fast nearest-neighbor query to retrieve similar regions. The system includes these regions in the selection and performs these operations at an interactive speed. Fig. 10 shows our results after a few user interactions. The number of detected regions reduces from 500 to about 300 with just three such interactions.

7 Data Scalability

Processing many gigabytes of image data requires out-of-core algorithms and techniques. We pay special attention to memory constraints when we implement our saliency computation, storage of results, and our interactive viewer.

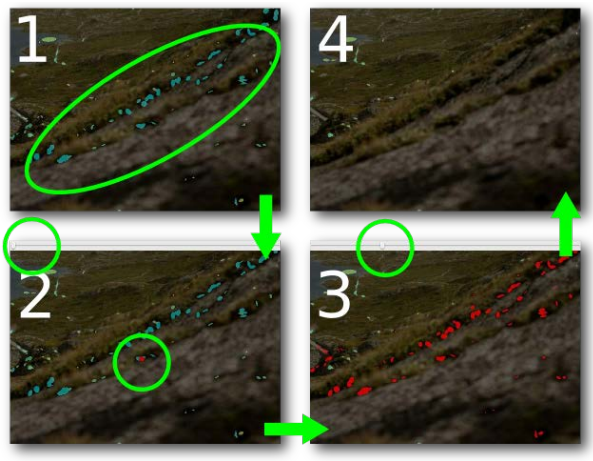


Fig. 9. Users refine the results by deleting misidentified regions in batch. 1. Locate similar misidentified regions. 2. Select one representative. 3. Adjust the slider to cover the desired regions. 4. Delete the selection.

Fig. 10. A few user-refinement interactions remove most of the misidentified regions. The number of regions reduces from 500 to about 300 with three select-slide-delete interactions.

7.1 Out-of-core GPU Saliency Computation

We use GPUs to accelerate image saliency computation. The required operations for Difference of Gaussians (*DoG*): image filtering, addition, subtraction, and resizing are highly parallelizable and suitable for GPU implementation.

We compute a Gaussian pyramid to approximate the Gaussian blurred images at different scales. We repeatedly downsize the image by a scale of two and convolve it with a fixed Gaussian kernel of scale σ. We store the $x\sigma$ scale Gaussian image at level $\log_2(x)$ of the pyramid. The *DoG* images can be computed by simply finding the difference of Gaussians images between two levels. The non-linear normalization function $\mathcal{N}$ iteratively applies the *DoG* operation [16] to promote the peak regions. We compute each of these normalized maps once and use them to compose sliding-window saliency maps .

A significant problem in processing very large images (which currently are a few gigapixels) is that the entire image will not fit in the GPU or main memory. To address this, we divide each image into small tiles (256×256). We load the image tiles into the GPU independently for addition, subtraction, and resizing operations.

Gaussian filtering requires information on image boundaries. Filtering each tile without overlap results in loss of information at the tile boundaries. To address this we load these tiles into the GPU with overlaps for filtering. For every sub-image that is to be filtered we load two extra rows of tiles (top and bottom) and two extra columns of tiles (left and right) that surround the sub-image. We fill the GPU memory with the largest possible sub-image (with additional surrounding overlapping tiles) to ensure efficient processing and minimize re-filtering of overlapping tiles. Depending on the loading order, either one row or one column will be used for filtering the next consecutive set of tiles. The ability to independently filter these tiles allows parallelization.

In our implementation, we use the NVIDIA performance primitives for GPU Gaussian filtering, resizing, addition, and subtraction.

7.2 Salient Regions Storage

We store ellipses to approximate salient regions to ease storage. The sliding-windows saliency maps incur a storage burden. Similar to storing mipmaps, it takes $1\frac{1}{3}$ times the image size to store the saliency maps at all levels. Although it is a constant factor increase, doubling the storage of the already very large images poses a challenge.

To address this, we first threshold the sliding-window saliency maps and locate continuous regions, also sometimes referred to as *blobs*. The open-source library `cvblob` provides blob detection in our implementation. We carry out local Principal Component Analysis (PCA) to fit ellipses to these regions. This allows us to reduce the storage of each region from hundreds of pixels to a few parameters (center, orientation, and principal axes intercepts of each ellipse). The threshold of our experiments is 0.25.

7.3 Tiled Image Viewer

Very large images presented in this paper do not fit in the GPU or main memory for display. Each gigapixel in RGB format takes three gigabytes of memory in an uncompressed format for rendering. Images with even a few gigapixels exceed the GPU or main memory of a workstation. We have implemented an out-of-core tiled-image viewer. Our viewer fetches only what can be seen at an appropriate scale from the disk. This is similar in spirit to the concept of a clipmap [41]. To carry this out we build a mipmap pyramid of the image. We divide the images on each level into tiles of 256×256. We load the required image tiles according to the viewing parameters. We pre-fetch two extra rows and columns of image tiles surrounding the viewing region to provide smooth panning. Loading images from the immediately nearby scales prepares for the zooming operation. We store these images in a texture array. This array is independent of the display arrangement of the textures. We map each texture to an array location by a hash function. The loaded texture can be reused on different views without any memory movement. The memory usage of the viewer is independent of the size of the image. It is related to the size of the viewing window on the screen only. Our viewer needs 350 megabytes for viewing a five gigapixel image with a few hundred overlay regions.

8 Results

We evaluate our approach on four multi-gigapixel images. We report the regions identified by our system and compare them against web community tags. We also report timings of our experiments. We perform our experiments on the Linux platform with one Intel E5420 CPU, 4GB RAM, and one NVIDIA GeForce GTX 295 GPU (895MB).

8.1 Datasets

Digital image stitching and consumer grade robotics have made panoramic photography very popular. Compact digital cameras can stitch multiple consecutive pictures into a panorama. Robotic devices such as the Gigapan EPIC can take hundreds of pictures automatically for image stitching. These products allow consumers to create images of several gigapixels. Community panorama websites such as Gigapan and HDView have gained much popularity on the Internet. Users around the world upload their panoramic pictures to these websites. The web community of panoramic photography enthusiasts then explore, tag, and comment on interesting regions in these images. We downloaded four gigapixel images from the Gigapan website as shown in Fig. 11 and discussed below.

Grimsel Pass: The Grimsel Pass is a high mountain pass in Switzerland. It connects the valley of Rhone River in the canton of Valais and the Haslital in the canton of Bern. The picture shows an overview of the mountain area, the lake, and the road network. There are distinctive areas with building constructions, hotels, and numerous cars on the road.

Royal Gorge Bridge: The Royal Gorge Bridge is the highest suspension bridge in the world. This tourist attraction is located in a theme park near Canon City, Colorado, shown in the top right of the picture. The Royal Gorge Route Railroad is a heritage railroad offering scenic and historical train-rides, shown at the bottom left of the picture. The picture shows the valley of Arkansas River with the Royal Gorge Bridge and a small town on the right.

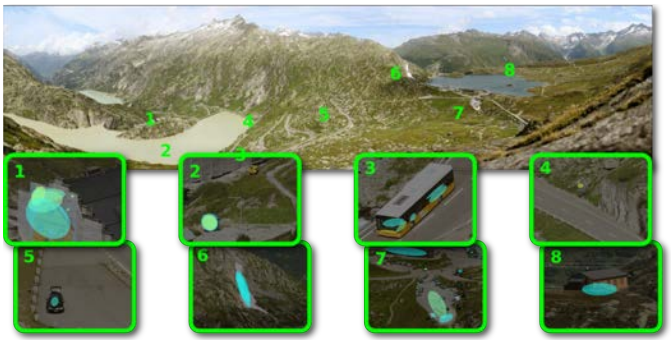
Cacti: This picture shows a cactus field in Arizona. A few hikers are hidden among thousands of cacti.

Main Mt. Whitney Trail: This is a trail in the Sequoia National Park, California. This image of mountains and lakes includes many hikers. They all took the challenge to hike the highest peak (14,497') in the lower 48 states.

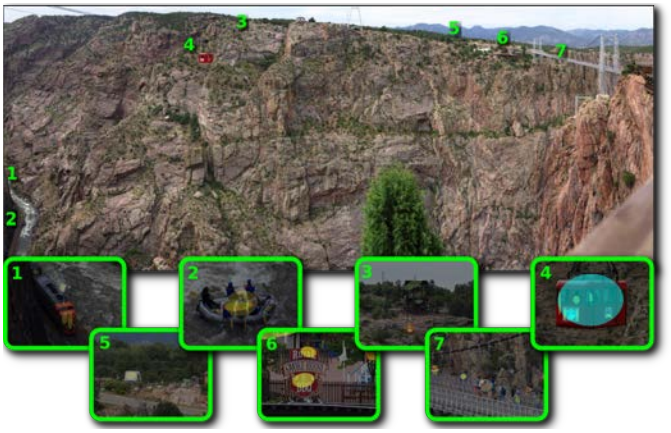
8.2 Evaluation

We show a sample of the detected regions in Fig. 11. We tag the locations with numbers and show the detected region in the corresponding thumbnails. We overlay the ellipses onto detected regions. Our system identifies a variety of regions at multiple scales. It locates humans, vehicles, buildings and even special features of the landscape such as a glacier. This shows the generality of our approach.

In the Grimsel Pass picture (Fig. 11(a)), the system detects buildings in thumbnails 1 and 8. Thumbnail 2 shows a blue Swiss Tardis. Cars, coaches, and road signs are found in thumbnails 3, 4, and 5. Thumbnail 6 shows a glacier in the mountain. Thumbnail 7 gives a view of the parking lot example in Section 4.

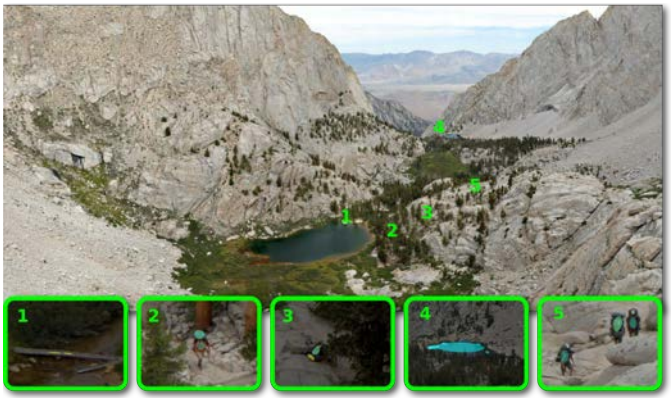


(a) Grimsel Pass (1.3 gigapixels)
http://www.gigapan.org/gigapans/30463/



(b) Royal Gorge Bridge (1.4 gigapixels)
http://www.gigapan.org/gigapans/7295/

(c) Cacti (4.0 gigapixels)
http://www.gigapan.org/gigapans/14937/



(d) Mt. Whitney (5.0 gigapixels)
http://www.gigapan.org/gigapans/44272/

Fig. 11. Gigapan picture datasets with samples of numbered detected regions shown in the thumbnails.

Image	SW Saliency	k-NN	Interactions
Grimsel Pass	18k (64)	525(64)	400(64), 325(62)
Royal Gorge	19k (49)	567(49)	226(49), 121(45)
Cacti	50k(10)	1.5k(10)	761(10), 513(7)
Whitney	40k(15)	1069(15)	604(14), 259(12)

Table 1. This table shows the quality of results after each step of processing. The SW Saliency column shows the results after sliding-window saliency map. The k-NN column shows the top 3% regions selected by anomaly detection. The Interaction column shows the results after three and five user interactions. In each of these columns, the number shows the count of computer-selected regions and the number in parenthesis is the count of the detected objects of interest. An object of interest may contain multiple detected regions.

In the lower left of the Royal Gorge Bridge picture (Fig. 11(b)), the system finds a train and a river rafting boat along the Arkansas river in thumbnails 1 and 2. We see two cable cars in thumbnails 3 and 4; the one in 3 has just departed the station whereas the one in 4 is much closer to the camera. Thumbnails 5 and 6 show a caravan and a restaurant sign around the town. Thumbnail 7 shows tourists and flags.

We found a few hikers among the cacti in Fig. 11(c). Thumbnail 1 shows the back of a hiker; the system has identified him by his jeans. Thumbnail 2 shows a double image of two hikers, who probably moved and were captured at two instances. The hiker in thumbnail 3 was using a camera. We find a man sitting in thumbnail 4. Two other hikers are detected in thumbnail 5.

Although Mt. Whitney (Fig. 11(d)) is a popular hiking spot, we detect more than the hikers. Thumbnail 1 and 4 show a bridge and the Alpine lake. We found 4 hikers in thumbnails 2 and 5. A hiking backpack is shown in thumbnail 3. We are able to detect 12 out of 13 hikers in this picture.

Table 1 shows a summary of the detected number of regions after each step. We inspect the results and count the number of detected meaningful objects, such as humans, vehicles, and buildings, after each processing step. These are shown in parentheses. Many detected regions may map to different parts of the same meaningful object.

The number of detected objects remains the same until the user-guided interactive refinement step. This shows our system is conservative and generally does not remove positive results. Our system finds a few hundred regions from images of a few gigapixels.

Gigapan Community Tags: Internet users tag and comment on these very large images on Gigapan's website. Table 2 shows our system is able to detect a majority of the gigapan community tags. Fig. 12 compares the tags with our user-refined results. These tags contain high-level semantic information. For example, the pattern we detected in thumbnail 1 of Fig. 11(a) is a child's drawing for a construction accident prevention campaign. Although a large number of tags represent regions of interest, such as vehicles or humans, not all interesting regions are tagged. Fig. 13 shows two examples of tags that contain semantics beyond general visual information. In Fig. 13 (a), a user has tagged a common cactus as being the *original one*. Another user has tagged some buffelgrass in Fig. 13 (b) because it grew after a fire. Clearly such tags require high-level semantic knowledge that a purely low-level perception-based system such as ours is unable to detect. We show the number of tags without semantic information in the second column of Table 2.

8.3 Performance

We report timings for different stages of our system. Table 3 shows the pre-processing times for the sliding-windows saliency computation, k-nearest neighbors anomaly detection, and user-guided interactive refinement.

The running time for the sliding-window saliency computation averages to 2.5 hours per gigapixel. Since the saliency computation routines are parallelizable, we expect a cluster of machines can easily process each gigapixel within a few minutes. k-nearest neighbor anomaly detection takes less than a minute. This is considerably faster than the

Image	Gigapan Tags		Detected by our system
	All	Non-Semantic	
Grimsel Pass	35	32	25
Royal Gorge	35	30	24
Cacti	24	17	15
Whitney	11	11	10

Table 2. This table compares our results with Gigapan community tags. Our system detects most of the Giganpan community tags. The second column shows the count of tags without semantics as discussed in Section 8.2. There is a difference between the count of tags and detected objects. Each tag may cover multiple objects and not all meaningful objects are tagged. Some of the tags also overlap with one another.

Image	Preprocessing		Interaction	
	SW Sal.	k-NN	w/o k-NN	w/ k-NN
Grimsel Pass	3.2h	5s	1193ms	9ms
Royal Gorge	4.5h	6s	1247ms	9ms
Cacti	10.3h	25s	2029ms	20ms
Whitney	11.1h	23s	1968ms	17ms

Table 3. This table shows the running time of our system. The preprocessing time includes the sliding-window saliency (SW Sal. column) and anomaly detection (k-NN columns). The interaction column show timings for interactive user refinements. This select-slide-delete refinement cannot be interactive without the k-nearest neighbors (k-NN) anomaly detection step.

recent image saliency techniques in Section 2.1. This pre-processing can be computed offline.

The k-nearest neighbor anomaly detection reduces thousands of salient regions to a few hundreds while it also enables the interactive refinement process. The interactive select-slide-delete refinement process (searching, and redrawing) takes only tens of milliseconds. In contrast, when we try to interactively refine thousands of salient regions, each interaction takes several seconds.

9 Conclusions and Discussion

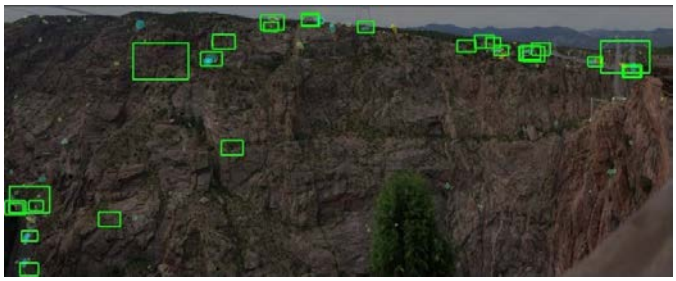
In this paper, we show how to use visualization and computation to assist the exploration of very high-resolution large-scale landscape images. We address the visual scale challenge by introducing the sliding-window computational saliency model. Anomaly detection automatically discovers information while visualization allows the users to explore the image interactively. Our system implementation is scalable to large datasets by using out-of-core methods. We show our system can detect interesting details from various landscape images. The detections largely match community user tags.

In this paper we have focused on large-scale landscape images that have been acquired by stitching together of a large number of photographs. However, very large scale images are finding use in a number of different areas. For example, semiconductor wafer manufacturers are using terapixel images for quality inspection of their chips for detecting circuit anomalies. As another example, latest generation microscopes stitch together volumes of very high resolution, multi-slice imagery that depicts the entire life-cycle of a number of parasites. As yet another example, astronomers are exploring the farthest reaches of the universe through the use of terapixel imagery. Our system is currently targeted for analyzing landscape images using color and appearance. The key to analyzing other domain-specific images is to design appropriate descriptors that characterize similarity across regions of interests. The descriptors should allow fast discovery of locally distinct regions as well as accurate identification of the globally unique regions. We believe the field of visualization will be considerably strengthened by incorporating analysis and visualization of very large images as a first-class data primitive next to volumes and meshes. This paper presents some of the first steps towards that goal.

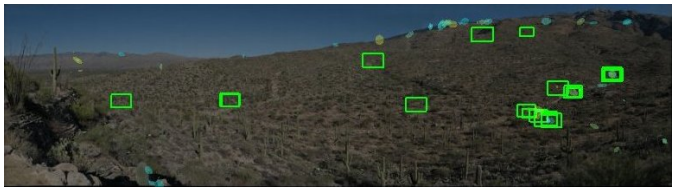
Acknowledgments

We thank Derek Juba for the discussions and constructive feedback that led to substantial improvements of the ideas in this paper. We also thank the anonymous reviewers for their tremendously constructive comments and suggestions that have greatly improved the presentation of this work. This work has been supported in part by the NSF grants: CCF 05-41120, CMMI 08-35572, CNS 09-59979 and the NVIDIA CUDA Center of Excellence. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the research sponsors.

(a) Grimsel Pass



(b) Royal Gorge Bridge



(c) Cacti

(d) Mt. Whitney

Fig. 12. This figure shows the Gigapan community tags and our detected regions. The rectangles are the Gigapan tags and the ellipses are our user-refined detected regions.

Fig. 13. Gigapan community tags: (a) The original cactus. (b) Some buffelgrass after a fire. These tags contain semantics beyond general appearance.

REFERENCES

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Trans. Graphics*, 26, 2007. doi:10.1145/1276377.1276390. 2

[2] M. Baştan, H. Çam, U. Güdükbay, and O. Ulusoy. An MPEG-7 compatible video retrieval system with integrated support for complex multimodal queries. *IEEE Multimedia*, 17(3):62–73, 2009. doi:10.1109/MMUL.2009.74. 5

[3] U. Bordoloi and H.-W. Shen. View selection for volume rendering. In *IEEE Visualization*, pages 487 – 494, Oct 2005. doi:10.1109/VISUAL.2005.1532833. 3

[4] N. D. B. Bruce and J. K. Tsotsos. Saliency, attention, and visual search: An information theoretic approach. *Journal of Vision*, 9(3):1–24, 2009. doi:10.1167/9.3.5. 2

[5] S. Bruckner and T. Möller. Isosurface similarity maps. *Computer Graphics Forum*, 29(3):773–782, 2010. doi:10.1111/j.1467-8659.2009.01689.x. 3

[6] M. Chen and H. Jänicke. An information-theoretic framework for visualization. *IEEE Trans. Visualization and Computer Graphics*, 16(6):1206–1215, 2010. doi:10.1109/TVCG.2010.132. 5

[7] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *IEEE CVPR*, pages 1 –8, June 2007. doi:10.1109/CVPR.2007.383050. 2

[8] C. S. Co, M. A. Duchaineau, and K. I. Joy. Streaming aerial video textures. In *Scientific Visualization: Advanced Concepts*, volume 1 of *Dagstuhl Follow-Ups*, pages 336–345, 2010. doi:10.4230/DFU.SciViz.2010.336. 1

[9] G. Daniel and M. Chen. Video visualization. In *IEEE Visualization*, pages 409–416, 2003. doi:10.1109/VISUAL.2003.1250401. 2, 3

[10] K. Devlin, A. Chalmers, A. Wilkie, and W. Purgathofer. STAR: Tone reproduction and physically based spectral rendering. In *State of the Art Reports, Eurographics 2002*, pages 101–123, September 2002. 2

[11] S. Goferman, L. Zelnik-Manor, and A. Tal. Context-aware saliency detection. In *IEEE CVPR*, pages 2376 –2383, June 2010. doi:10.1109/CVPR.2010.5539929. 2

[12] D. Guo and C. Poulain. Terapixel: A spherical image of the sky. In *Microsoft Environmental Research Workshop*, 2010. 1

[13] A. Hanjalic, R. Lienhart, W.-Y. Ma, and J. R. Smith. The holy grail of multimedia information retrieval: So close or yet so far away? *Proceedings of the IEEE*, 96(4):541 –547, 2008. doi:10.1109/JPROC.2008.916338. 2

[14] X. Hou and L. Zhang. Saliency detection: A spectral residual approach. In *IEEE CVPR*, pages 1–8, 2007. doi:10.1109/CVPR.2007.383267. 2

[15] S. Howlett, J. Hamill, and C. O'Sullivan. Predicting and evaluating saliency for simplified polygonal models. *ACM Trans. Applied Perception*, 2:286–308, 2005. doi:10.1145/1077399.1077406. 3

[16] L. Itti and C. Koch. A comparison of feature combination strategies for saliency-based visual attention systems. *Journal of Electronic Imaging*, 10:161–169, 2001. doi:10.1117/1.1333677. 4, 7

[17] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. PAMI*, 20(11):1254–1259, 1998. doi:10.1109/34.730558. 2, 3, 4

[18] H. Jänicke and M. Chen. A salience-based quality metric for visualization. *Computer Graphics Forum*, 29(3):1183–1192, 2010. doi:10.1111/j.1467-8659.2009.01667.x. 5

[19] H. Jänicke and G. Scheuermann. Visual analysis of flow features using information theory. *Computer Graphics and Applications, IEEE*, 2009. doi:10.1109/MCG.2009.135. 5

[20] H. Jänicke and G. Scheuermann. Measuring complexity in lagrangian and eulerian flow descriptions. *Computer Graphics Forum*, 29(6):1783–1794, 2010. doi:10.1111/j.1467-8659.2010.01648.x. 5

[21] H. Jänicke, A. Wiebel, G. Scheuermann, and W. Kollmann. Multifield visualization using local statistical complexity. *IEEE Trans. Visualization and Computer Graphics*, 13(6):1384 –1391, 2007. doi:10.1109/TVCG.2007.70615. 5

[22] M. Kazhdan and H. Hoppe. Streaming multigrid for gradient-domain operations on large images. *ACM Trans. Graphics*, 27:1–10, 2008. doi:10.1145/1360612.1360620. 1

[23] G. Kim and A. Torralba. Unsupervised Detection of Regions of Interest using Iterative Link Analysis. In *NIPS*, 2009. 2

[24] Y. Kim and A. Varshney. Saliency-guided enhancement for volume visualization. *IEEE Trans. Visualization and Computer Graphics*, 12(5):925–932, 2006. doi:10.1109/TVCG.2006.174. 3

[25] Y. Kim and A. Varshney. Persuading visual attention through geometry. *IEEE Transactions on Visualization and Computer Graphics*, 14:772–782, July 2008. doi:10.1109/TVCG.2007.70624. 3

[26] Y. Kim, A. Varshney, D. W. Jacobs, and F. Guimbretière. Mesh saliency and human eye fixations. *ACM Trans. Applied Perception*, 7:12:1–12:13, 2010. doi:10.1145/1670671.1670676. 3

[27] J. Kopf, M. Uyttendaele, O. Deussen, and M. Cohen. Capturing and viewing gigapixel images. *ACM Trans. Graphics*, 26(3), 2007. doi:10.1145/1276377.1276494. 1

[28] P.-Y. Laffont, J. Y. Jun, C. Wolf, Y.-W. Tai, K. Idrissi, G. Drettakis, and S.-e. Yoon. Interactive content-aware zooming. In *Proceedings of Graphics Interface 2010*, pages 79–87, 2010. 2

[29] C. H. Lee, Y. Kim, and A. Varshney. Saliency-guided lighting. *IEICE Trans. Information and Systems*, E92.D(2):369–373, 2009. doi:10.1587/transinf.E92.D.369. 3

[30] C. H. Lee, A. Varshney, and D. W. Jacobs. Mesh saliency. *ACM Trans. Graphics*, 24:659–666, July 2005. doi:10.1145/1073204.1073244. 3

[31] T. Liu, J. Sun, N.-N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. In *IEEE CVPR*, pages 1 –8, June 2007. doi:10.1109/CVPR.2007.383047. 2

[32] Q. Luan, S. M. Drucker, J. Kopf, Y.-Q. Xu, and M. F. Cohen. Annotating gigapixel images. In *UIST*, pages 33–36, 2008. doi:10.1145/1449715.1449722. 1

[33] R. Machiraju, J. E. Fowler, D. Thompson, and a. B. S. W. Schroeder. Evita: Efficient visualization and interrogation of tera-scale data. In *Data mining for scientific and engineering applications*. Kluwer, 2001. 3

[34] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, pages 331–340, 2009. 5

[35] A. Oliva, A. Torralba, M. Castelhano, and J. Henderson. Top-down control of visual attention in object detection. In *IEEE ICIP*, pages 253–6, Sept 2003. doi:10.1109/ICIP.2003.1246946. 2

[36] R. Patro, C. Y. Ip, and A. Varshney. Saliency guided summarization of molecular dynamics simulations. In *Scientific Visualization: Advanced Concepts*, volume 1 of *Dagstuhl Follow-Ups*, pages 321–335, 2010. doi:10.4230/DFU.SciViz.2010.321. 3

[37] M. Rubinstein, D. Gutierrez, O. Sorkine, and A. Shamir. A comparative study of image retargeting. *ACM Trans. Graphics*, 29:1–160, 2010. doi:10.1145/1882261.1866186. 2

[38] B. Russell, W. Freeman, A. Efros, J. Sivic, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *IEEE CVPR*, pages 1605 – 1614, 2006. doi:10.1109/CVPR.2006.326. 2

[39] J. Sankaranarayanan, H. Samet, and A. Varshney. A fast all nearest neighbor algorithm for applications involving large point-clouds. *Computers & Graphics*, 31(2):157 – 174, 2007. doi:10.1016/j.cag.2006.11.011. 5

[40] B. Summa, G. Scorzelli, M. Jiang, P.-T. Bremer, and V. Pascucci. Interactive editing of massive imagery made simple: Turning Atlanta into Atlantis. *ACM Trans. Graphics*, 30:1–13, 2011. doi:10.1145/1944846.1944847. 1

[41] C. C. Tanner, C. J. Migdal, and M. T. Jones. The clipmap: a virtual mipmap. In *SIGGRAPH*, pages 151–158, 1998. doi:10.1145/280814.280855. 7

[42] J. K. Tsotsos, S. M. Culhane, W. Y. K. Wai, Y. Lai, N. Davis, and F. Nuflo. Modeling visual-attention via selective tuning. *Artificial Intelligence*, 78(1-2):507–545, 1995. doi:10.1016/0004-3702(95)00025-9. 2

[43] I. Viola, M. Feixas, M. Sbert, and M. Gröller. Importance-driven focus of attention. *IEEE Trans. Visualization and Computer Graphics*, 12(5):933–940, 2006. doi:10.1109/TVCG.2006.152. 3

[44] W. Wang, Y. Wang, Q. Huang, and W. Gao. Measuring visual saliency by site entropy rate. In *IEEE CVPR*, pages 2368 –2375, June 2010. doi:10.1109/CVPR.2010.5539927. 2

[45] A. Wiebel, X. Tricoche, D. Schneider, H. Jänicke, and G. Scheuermann. Generalized streak lines: Analysis and visualization of boundary induced vortices. *IEEE Trans. Visualization and Computer Graphics*, 13(6):1735–1742, 2007. doi:10.1109/TVCG.2007.70557. 3

[46] R. Zonenschein and L. Velho. Visorama 2.0: a platform for multimedia gigapixel panoramas. In *SIGGRAPH ASIA 2010 Posters*, 2010. doi:10.1145/1900354.1900424. 1