Interactive Content-Aware Zooming

Pierre-Yves Laffont[1,2,3,*] Jong Yun Jun[2] Christian Wolf[3] Yu-Wing Tai[2] Khalid Idrissi[3]
George Drettakis[1] Sung-eui Yoon[2,†]

[1] REVES / INRIA Sophia-Antipolis [2] KAIST [3] Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205, F-69621

ABSTRACT

We propose a novel, interactive content-aware zooming operator that allows effective and efficient visualization of high resolution images on small screens, which may have different aspect ratios compared to the input images. Our approach applies an image retargeting method in order to fit an entire image into the limited screen space. This can provide global, but approximate views for lower zoom levels. However, as we zoom more closely into the image, we continuously *unroll* the distortion to provide local, but more detailed and accurate views for higher zoom levels. In addition, we propose to use an adaptive view-dependent mesh to achieve high retargeting quality, while maintaining interactive performance. We demonstrate the effectiveness of the proposed operator by comparing it against the traditional zooming approach, and a method stemming from a direct combination of existing works.

Index Terms: I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms;

1 INTRODUCTION

Recent years have seen widespread use of very high resolution images, either directly taken from high resolution cameras or specialized setups [15], or stitched together from several photographs to create panoramas [24, 4]. At the same time, recent technological advances have resulted in a proliferation of small-screen devices, such as mobile phones or e-book readers, which often have a limited resolution. A typical contemporary user will use these low resolution devices to navigate in the aforementioned high resolution images. In addition, these devices will frequently have very different aspect ratios compared to the original images, which are typically very wide.

In such a navigation session, we assume that a user specifies a *view center*, e.g., a landmark or important object in an image, and a *zoom level*, which indicates how far the user wishes to zoom in or out. Our overall goal is to maintain the two following properties during the navigation session: 1) maximize the *proportion* and *resolution* of the original image that is displayed on the screen, while fully utilizing available screen space, and 2) minimize the *distortion*, such as edge stretching and deformation of local shapes, that is introduced into the final displayed image.

Current solutions for such navigation suffer from multiple limitations. Most often, the traditional zooming approach (i.e., uniform scaling combined with cropping) is used. This results in a waste of screen space when zooming out, with "black bands" appearing above and below the image, along with a severe loss of resolution (see the first row of Fig. 1). Also, when zooming in, the context of the image is rapidly lost, severely hindering the navigation task in images that often include multiple points of interest.

*e-mail: pierre-yves.laffont@sophia.inria.fr
†e-mail: sungeui@kaist.edu

In parallel with large images and the spread of small screen devices, content-aware image retargeting methods [1, 26, 21] have received significant attention in the past few years. However, most previous work concentrates on retargeting an *entire* image or video, from a source format to a different output aspect ratio and resolution. In addition, these techniques have been applied on fairly small images, with resolutions that rarely exceed 1 Megapixels. To our knowledge, retargeting has not been previously used as a way to effectively *navigate* in large images, with typical resolutions higher than 5 Megapixels.

Given a desired zoom level and a view center, there is no evident manner to determine the image region that should serve as input to the retargeting process. This is possibly one of the reasons why retargeting has not been previously used for interactive image navigation. A reasonable direct adaptation of previous retargeting methods for interactive zooming would involve first retargeting the source image to match the aspect ratio of the target output device, and then performing traditional zooming on the retargeted image. In the following we call this the *Retargeting+Zooming* approach; as we shall see, the resulting images can be very distorted (second row in Fig. 1).

In this work, we introduce an *Interactive Content-Aware Zooming* operator, leveraging the power of retargeting methods to develop an efficient and effective solution for interactively navigating in large images. In particular, this new operator provides a *continuous* zooming experience in high resolution images. Our method is effective, since it can provide either a global, approximate view that utilizes the entire screen space when users zoom out, or a local and accurate view when they zoom in, as well as continuous transitions between different zoom levels. Efficiency is achieved with a two-level mesh warping process and a view-dependent adaptive mesh. Also, users can control the amount of distortion either by zooming into the image in an intuitive way, or by editing the *significance map* at runtime to select image features that should suffer from minimal distortion.

Our contributions can be summarized as follows:

- The introduction of our novel *interactive content-aware zooming* operator, which allows effective and continuous navigation in large images on a small display.
- An adaptive view-dependent mesh and a two-level mesh warping process, which provide an interactive performance for our new operator.
- The use of an irregular triangle mesh, with an appropriate deformation energy and line constraints, for retargeting with a warp-based iterative approach.

As we can see in Fig. 1, our method shows significant improvements compared to both traditional zooming and *Retargeting+Zooming*, since we optimize the use of screen space while avoiding distortions in higher zoom levels.

2 Related Work

Most recent image retargeting techniques are composed of two steps: first, a *significance map* is built to represent the visual importance of each pixel in the image, by combining a saliency measure [30] and high-level features such as detected faces; then, the image content is rearranged to fit the target size, while minimizing the induced distortion. Content-aware retargeting methods can be further classified as seam-based, warp-based, and patch-based approaches, while hybrid methods combine several of these techniques together.

Seam-carving [1] defines a seam to be a connected path of low energy pixels from one image boundary to another. By successively removing/inserting seams, it can automatically reduce/extend image size. This method is simple to implement and shows very convincing results for a variety of images, as long as the image contains a sufficient number of low-importance pixels for removal/insertion. Subsequent techniques such as [20, 14] have been proposed to improve the speed or quality of seam-based approaches. Seam-based approaches, however, often fail at preserving structural content such as straight lines.

Warp-based techniques [11, 26, 27, 29, 25] consider content-aware image resizing as a mesh warping problem. These techniques apply a mesh onto the original image and deform it non-homogeneously, according to the importance of each individual polygon of the mesh. The retargeted image is obtained by using texture mapping to map the image contents from the original mesh to the deformed mesh. Warp-based techniques allow a better control of the retargeting process, for example by considering line constraints [12, 16], and thus tend to better preserve the global image layout. This explains why this work uses an image retargeting method based on mesh warping.

Patch-based approaches [23, 8, 2, 19] assume the image to be composed as a set of small patches, and perform retargeting by reorganizing these patches to accommodate the desired target size. These methods can exploit redundancy of image patterns by mapping repetitive patches in the source image, to a few representative patches in the target image. However, these approaches are often computationally expensive; in addition, they are inappropriate in the context of *continuously* navigating in an image, since image objects can be removed or moved around during patch reorganization, leading to severe popping artifacts.

Most recently, hybrid techniques [21, 9] have been introduced to combine several different operators (i.e., scaling, cropping and seam carving) in order to achieve faster and better-looking results.

Compared to the aforementioned prior work, our method is unique in its focus on developing an interactive zooming operator for high resolution image visualization on small display devices. Our method is a hybrid approach that combines a warp-based retargeting technique and the traditional zooming operator. Also, to our knowledge the proposed method is the first to use a view-dependent mesh representation [17, 28], in order to improve the performance and quality of retargeting.

3 Overview

Our goal is to provide an interactive and effective visualization method for high resolution images in a limited screen space, which may have an aspect ratio different from those of the original images. We use an image retargeting method to effectively visualize large images in such a limited screen space: we employ a mesh warping technique (Sec. 4) that combines an irregular triangular mesh [12], an effective shape distortion energy [26], as well as line constraints [6] to prevent the deformation of straight lines. Building on top of this method, we introduce the following two main contributions:

Interactive content-aware zooming (Sec. 5.1): A retargeting method can fit an entire image into a limited screen space by shrinking and deforming insignificant image parts, while preserving significant regions. However, this process induces distortions, such as altered local shapes or twisted lines. In addition, users may want to take a closer look at particular regions by zooming into them. Retargeting the original image to modify its aspect ratio, and then applying traditional zooming on the retargeted image results in obvious distortion at higher zoom levels. Furthermore, it does not allow to recover details from insignificant parts that were blended together during the retargeting process. The results of such an operation, which we call *Retargeting+Zooming*, are shown in the second row of Fig. 1.

To address these issues, we introduce a novel *Interactive Content-Aware Zooming* (ICAZ) operator, which is based on the following simple, but intuitive assumption. When we zoom out, we want to obtain an overall, global view of the image. Therefore, we apply mesh warping and display the entire warped image, fully utilizing the screen space to display details of significant image regions. Then, as we zoom into a particular region, we want to observe *more details* of a *smaller zoomed region* of the image, with less distortion. We provide a local, but more accurate view of the zoomed region by *continuously unrolling* the distortion, which is performed by applying retargeting with a target aspect ratio closer to that of the original image. As a result, when users zoom in, visual artifacts caused by image retargeting vanish and the displayed image becomes closer to the original one (see Fig. 1, third row).

Mesh warping using a view-dependent mesh (Sec. 5.2): In order to visualize high resolution images at interactive rates with our method, we accelerate the performance of ICAZ by using a view-dependent mesh, which is an adaptive multi-resolution mesh refined according to the viewing information (i.e., zoom level and view center). As we zoom into a region of an image at runtime, we refine triangles that are within the viewing area and use large triangles outside the viewing area. Therefore, the distortion caused by the retargeting process is distributed smoothly over the fine triangles of the image portion being observed, while we keep the computation time of our retargeting method nearly constant by controlling the total number of vertices in the mesh. To efficiently refine triangles within the view area, we propose to use a multi-resolution hierarchy that is constructed from an initial coarse mesh at precomputation time.

4 Retargeting using Mesh Warping

In this section, we describe the image retargeting method based on mesh warping, that our proposed operator will use in Sec. 5. Inputs to the mesh warping process are 1) the original image, 2) a significance map that has the same resolution as the image, and 3) a set of line constraints provided by the user. The significance map can be either automatically computed by using a combination of the gradient norm and a saliency measure as in [26], or manually specified by users. In addition, our system allows users to adjust the significance map at runtime, through a simple visual interface.

4.1 Mesh Construction

While most retargeting methods based on mesh warping use an initial mesh composed of regular quads, we use irregular triangles for our initial mesh. This choice was mainly motivated by the fact that irregular triangles allow us to place vertices arbitrarily in the mesh; however, the proposed method remains valid and could be implemented with a few modifications in the case of quad-based meshes. We use a conforming Delaunay triangulation to construct an initial triangular mesh, where some triangle edges lie on the constrained lines. We control the size of triangles by specifying their maximum areas in the triangulation process, and the mesh quality by setting a minimal angle constraint [22]. Once the mesh has been constructed, we compute the significance value of each triangle by averaging its per-pixel significance values.

Figure 1: Top: traditional zooming; middle: *Retargeting+Zooming*; bottom: our *Interactive Content-Aware Zooming* method. We use the "power station" image (7475×1999 pixels) and varying zoom levels. Traditional zooming wastes a lot of screen space at lower zoom levels, with black bands covering most of the screen and a significant loss of resolution (first row, column **a**). The *Retargeting+Zooming* approach better utilizes the screen space (second row, column **a**) but suffers from heavy distortion at higher zoom levels (second row, column **d**). In comparison, our method shows global, but approximate views for lower zoom levels, and local, but accurate views for higher zoom levels (third row).

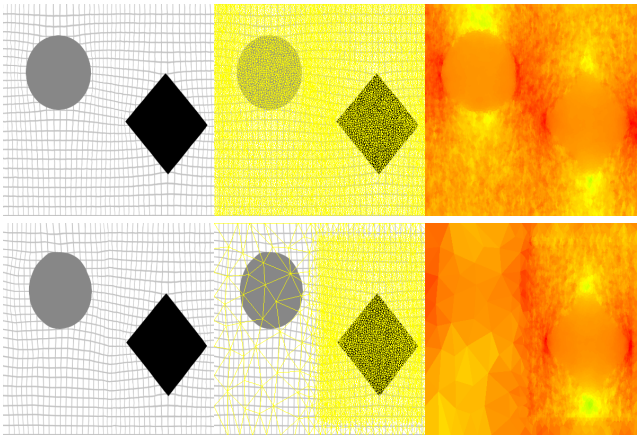

Figure 2: Influence of mesh quality on the results of mesh warping. The top row uses a uniform fine mesh whereas the bottom row uses a non-uniform mesh, with a coarse resolution on the left half and a fine resolution on the right half. The left, middle, and right columns show the input image, overlaid meshes, and scaling factors.

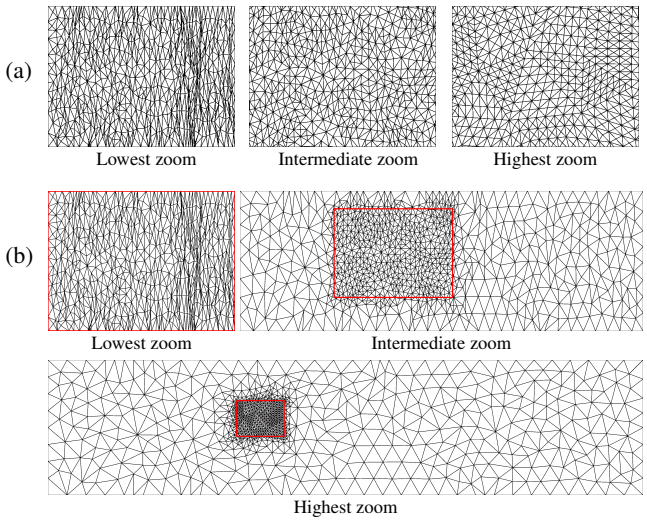

Figure 3: Evolution of the target aspect ratio and viewing area size, with the zoom level. (a) The part of the deformed mesh contained in the viewing area, which is mapped into the screen space, at different zoom levels. (b) A view of the entire mesh after warping; the red rectangle corresponds to the viewing area, and the zoom levels correspond to those used in **a**. As we zoom in, we modify both the target aspect ratio to control the amount of distortion, and the size of the viewing area to control the portion being observed.

For the rest of this section, we define $\mathbf{V}$ as the set of mesh vertices, $\mathbf{F}$ as the set of mesh triangle faces, and $\mathbf{L}$ as the set of input line constraints. We use $\mathbf{v}_i = (x_i; y_i)$ to denote the coordinates of the i^{th} vertex in $\mathbf{V}$, and $\mathbf{v}'_i = (x'_i; y'_i)$ to denote its coordinates in the deformed mesh. $\mathbf{e}_{ij}$ corresponds to the edge from $\mathbf{v}_i$ to $\mathbf{v}_j$.

4.2 Mesh Deformation Energy

Given an initial mesh constructed from the input image, the mesh warping process computes a deformed mesh by minimizing a global energy function, which is designed to measure the amount of perceived distortion. In our case the energy function consists of two components: shape distortion energy D_{Shape} and line constraint energy D_{Line}.

We use the shape distortion energy proposed by Wang et al. [26]. This energy penalizes all transformations other than isotropic scaling, in order to represent the distortion of mesh faces. Our energy is defined as a weighted sum of per-triangle terms, and is given as follows:

$$D_{Shape} = \sum_{\mathbf{f}\in\mathbf{F}} w_f \sum_{\mathbf{e}_{ij}\in\mathbf{E}(\mathbf{f})} \left\| (\mathbf{v}'_i - \mathbf{v}'_j) - s_f(\mathbf{v}_i - \mathbf{v}_j) \right\|^2,$$

where $\mathbf{E}(\mathbf{f})$ is the set of edges associated to triangle $\mathbf{f}$, w_f is the weight of triangle $\mathbf{f}$ and corresponds to its normalized significance, and s_f is the *scaling factor* of $\mathbf{f}$. This energy measures how far the deformed triangle is from a uniformly scaled version of the original triangle.

Our line constraint energy D_{Line} is expressed as one of two terms D_{Line_D} and D_{Line_O} [6]. Both terms penalize misalignment of vertices associated with a line constraint, by measuring the distance from such a vertex $\mathbf{v}'_i$ to line $\mathbf{l} \in \mathbf{L}$:

$$D_{Line_D} = \sum_{\mathbf{l}\in\mathbf{L}} \sum_{\mathbf{v'}_i\in\mathbf{V}'(\mathbf{l})} \left\| (\mathbf{v'}_i - \mathbf{v'}_{start(\mathbf{l})}) - r_{i(\mathbf{l})}(\mathbf{v'}_{end(\mathbf{l})} - \mathbf{v'}_{start(\mathbf{l})}) \right\|^2,$$

$$D_{Line_O} = \sum_{\mathbf{l}\in\mathbf{L}} \sum_{\mathbf{v'}_i\in\mathbf{V}'(\mathbf{l})} ((\mathbf{v'}_i - \mathbf{v'}_{start(\mathbf{l})})^T \mathbf{n}_{(\mathbf{l})})^2,$$

where $\mathbf{V}'(\mathbf{l})$ is the set of vertices that are associated with a constrained line $\mathbf{l}$ in the deformed mesh, $\mathbf{v}'_{start(\mathbf{l})}$ and $\mathbf{v}'_{end(\mathbf{l})}$ are the endpoints of line $\mathbf{l}$ in the deformed mesh, $\mathbf{n}_{(\mathbf{l})}$ is the *unit normal* of line $\mathbf{l}$, and $r_{i(\mathbf{l})}$ is a scalar representing the *normalized projection* of $\mathbf{v'}_i$ on line $\mathbf{l}$. We refer readers to [6] for a more detailed description of these terms.

The difference between D_{Line_D} and D_{Line_O} arises when minimizing the global energy with our iterative solver, which will be described in Sec. 4.3. Using D_{Line_O} as the line constraint energy term fixes the line normal but allows associated vertices to slide along the line, whereas using D_{Line_D} allows the line orientation to change but fixes the normalized projections of vertices on the line. For constrained horizontal and vertical lines, we simply use D_{Line_O} as our line constraint energy, since we want to keep the normals of these lines fixed. For other constrained lines, we alternate between using D_{Line_D} and D_{Line_O} in the iterative solver described in Sec. 4.3.

Note that our line constraint energy D_{Line} is different from the *line bending* energy introduced in [26]. The latter was originally designed by Wang et al. to reduce the distortion of image features occupying multiple adjacent mesh faces, and to prevent foldovers; however, in practice and as noted in [29], it affects *all the mesh edges* and therefore limits the degree of freedom of mesh vertices. On the other hand, the line constraints we use only consider a small, meaningful *subset of edges in the image*, and ensure those lines remain straight to preserve the global layout of the image. Note that these line constraints are typically set by the person who created the image and not the user navigating in it, although our system also allows editing constraints interactively.

Also, as mentioned before, we take line constraints into account in the initial step of mesh construction. Therefore, we can directly place vertices along constrained lines during the triangulation: this simplifies both the formulation and implementation of line constraints, as we do not need to add the virtual vertices used in [6].

4.3 Iterative Solver

Combining the shape distortion energy D_{Shape} and the line constraint energy D_{Line}, our complete deformation energy becomes:

$$D = D_{Shape} + \alpha\, D_{Line},$$

subject to boundary constraints that define the final dimensions of the mesh after the mesh warping process. The factor α weights the line constraint energy term compared to the shape distortion term. As viewers are very sensitive to twisted lines, this weight is usually large, typically 100 in our examples.

Since the resulting energy is non-quadratic in the position of the vertices in the deformed mesh, we minimize it with an iterative solver. At each iteration, the parameters s_f, and either $r_{i(\mathbf{l})}$ or $\mathbf{n}_{(\mathbf{l})}$ (depending on which line constraint energy is used for the current iteration) are evaluated according to the current vertex positions. Once these parameters are fixed, the total energy D becomes quadratic, and can be easily minimized by solving a sparse linear system. The position of the vertices is updated from the solution vector, and the system iterates until convergence.

Once the energy has been minimized, we reconstruct the target image from the final deformed mesh, by using standard texture mapping with anisotropic filtering.

4.4 Discussion

The quality of the retargeted image depends on the quality of the input mesh, in particular triangle sizes. Figure 2 shows a comparison of the results when retargeting a synthesis image, with different initial meshes generated by a Delaunay triangulation. The initial image contains a filled circle and a diamond, superimposed on a regular grid; its significance map has been manually edited to mark the circle and diamond as prominent. The shape of the diamond is preserved in both cases, thanks to the fine mesh covering the object, whereas the boundary of the circle is altered in the bottom results. The coarse mesh used in the bottom row does not allow to sufficiently discriminate between significant and insignificant regions near the circle boundary. The right column of Figure 2 also shows the final scaling factors after retargeting: using a finer mesh better distributes the distortion according to the importance of each image region. This observation leads us to employ a view-dependent adaptive mesh in our work, as will be explained in Sec. 5.2.

5 Interactive Content-Aware Zooming

In this section we describe our main contribution, the *Interactive Content-Aware Zooming* (ICAZ) operator, and a view-dependent mesh to improve the quality and performance of our method.

5.1 Definition of the Operator

Image retargeting techniques try to achieve two contradictory goals: 1) maximize the amount of visual information on the screen (i.e. the amount and resolution of significant image portions displayed), and 2) minimize the amount of distortion introduced during the retargeting process. While prior retargeting methods [21, 9] aim at finding the optimal balance between these two objectives, we propose to let users control the tradeoff through an intuitive and interactive process.

To formalize our method, we define two parameters that depend on zoom level z_l: the image *target aspect ratio* and the size of the *viewing area*. On the one hand, the target aspect ratio influences the distortion induced by the mesh warping process described in

Sec. 4: a value close to the input image aspect ratio will produce low distortion, while more artifacts will appear as the aspect ratio change becomes more aggressive. On the other hand, the viewing area corresponds to the region of the warped image that will be mapped to the screen space: a higher viewing area size means that a larger portion of the image will be displayed on the screen, with a lower resolution.

In our system, these two parameters depend on zoom level z_l: as described in Sec. 3, zooming into the image will produce a more local and accurate view, by decreasing the size of the viewing area and setting a target aspect ratio closer to that of the input image. This is illustrated in Fig. 3-(b), where the viewing area (represented as a red rectangle) and the size of the target mesh are modified according to the zoom level.

In order to assign values to these parameters, we simply interpolate between their two extremal values, which are known. At the lowest level $z_l = 0\%$, the size of the viewing area equals that of the warped image, while the target aspect ratio equals the aspect ratio of the screen space; the displayed image corresponds to the result of directly applying mesh warping on the entire image. At the highest level $z_l = 100\%$, the size of the viewing area equals that of the screen space, while the target aspect ratio corresponds to the input image aspect ratio; the warped image contains no distortion and we have a one-to-one mapping between the pixels of the viewing area and those of the screen. Therefore, the displayed image corresponds to a cropped version of the input image. Several interpolation functions, or *evolution curves*, can be used. However, we found that simple linear interpolation for both parameters works well in practice.

The viewing area has a fixed aspect ratio which equals that of the target screen, and a size that depends on the zoom level z_l. However, its position depends on user interaction: during a panning process, the user defines a new view center v_c, which will be used to find the center of the viewing area in the deformed mesh. The view center v_c is expressed as a pair of coordinates in the original image; after a zooming operation, we find the point in the new deformed mesh that corresponds to v_c in the original image, then center the viewing area on it. With this approach, we can ensure that the center of interest (i.e., the image object clicked by a user during a panning process) remains at the center of the screen, irrespective of the zoom level.

A straightforward first implementation of the operator described above would simply retarget the entire image with a new target aspect ratio, whenever the zoom level changes. The region covered by the viewing area would then be cropped, and mapped into the screen space. While such an implementation may run interactively for small images, it would not scale well to larger inputs, as later demonstrated in Sec. 6.2.

5.2 ICAZ using a View-Dependent Mesh

The main technical challenge for the ICAZ operator is to provide high retargeting quality, while maintaining an interactive response to user actions. As shown in Sec. 4.4, finer triangles are necessary to provide high retargeting quality when we zoom in. A naive approach would increase the resolution of the mesh with the zoom level. However, using such an approach severely degrades the performance of the mesh warping process since the number of vertices, and thus, the computation time, increases dramatically with finer meshes.

Instead, we propose to use a view-dependent adaptive mesh, which contains fine triangles only in the portion of the mesh within the viewing area, and coarse triangles outside. This method allows us to achieve high quality results while maintaining a nearly constant computation time, since the number of vertices in our view-dependent adaptive mesh can be controlled to remain nearly constant. This approach, however, raises an important issue: we need

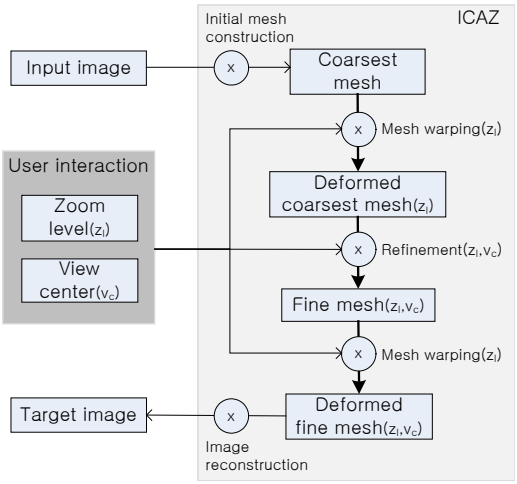

Figure 4: Overall process of the *Interactive Content-Aware Zooming* (ICAZ) operator with the two-level mesh warping procedure, for a given zoom level z_l and view center v_c.

to refine the portion of the mesh inside the viewing area *prior to applying mesh warping*. But, in order to determine which triangles are within the viewing area in the final mesh, we need to *apply mesh warping first*! This causes a typical Chicken and Egg problem.

In order to address this issue, we employ an *adaptive mesh warping* process that consists of two steps: a coarse-level mesh warping with the initial coarsest mesh, and a fine-level mesh warping with a refined version of the mesh. This process is illustrated in Fig. 4. Our method first performs the coarse-level mesh warping on the coarsest level of the mesh, to compute a deformed coarsest mesh. The deformed coarsest mesh serves as an approximation of the final mesh, to determine which triangles are within the viewing area. These triangles are then subdivided, and we obtain a refined view-dependent mesh. After performing fine-level mesh warping on this refined mesh, the image portion within the viewing area can be reconstructed, and finally mapped into the screen space.

5.3 Multi-Resolution Mesh and Refinement

In order to efficiently detect and refine the triangles that are within the viewing area, we propose to use a multi-resolution hierarchy. We construct the multi-resolution hierarchy from the initial coarsest mesh that was computed from the conforming Delaunay triangulation (Sec. 4.1), by applying a coarse-to-fine subdivision scheme [10]. We subdivide the longest edge of each triangle, i.e., a new vertex is added in the middle of this edge. This refinement method is simple to implement, and provides a reasonable mesh quality for the subsequent mesh warping process.

We recursively subdivide each triangle until its area becomes smaller than a threshold, which corresponds to the smallest area a triangle can reach in the runtime refinement process. The computed multi-resolution hierarchy is stored as a *forest* data structure, where each root of the forest corresponds to a triangle in the initial coarsest mesh (see Fig. 5). In addition, each node of the hierarchy corresponds to a triangle, and its two child nodes correspond to the two refined triangles obtained by subdividing the longest edge of this triangle. We construct this multi-resolution hierarchy in a preprocessing step, given the size of the input image and a set of line constraints. This construction method is performed only once and takes less than 300ms for our largest image, which contains about 15 Megapixels, with the settings described in Sec. 6.1.

Runtime refinement: We maintain a level-of-detail (LOD) cut in the multi-resolution representation that contains the current view-

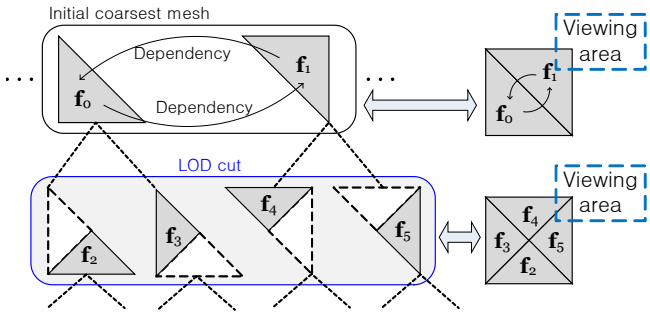

Figure 5: A simple adaptive mesh example. Left: our multi-resolution data structure, stored as a forest (with two root nodes in this example); right: the corresponding mesh. In this example, triangle $\mathbf{f}_1$ should be subdivided in the refinement process, as it is partly covered by the viewing area (top row). Since triangle $\mathbf{f}_0$ shares its longest edge with $\mathbf{f}_1$, it should be subdivided as well to avoid T-junctions, leading to the LOD cut shown in the bottom row.

dependent mesh. The LOD cut is initialized with root nodes of the forest data structure and thus represents the coarsest mesh. Whenever the viewing area is changed due to user interaction, the adaptive mesh has to be refined, and the LOD cut is updated accordingly: we traverse the LOD cut and refine triangles that are within the viewing area and whose areas are larger than a threshold, which corresponds to the *maximum triangle size in the viewing area*. This approach ensures that we have a nearly constant number of triangles covering the viewing area, for any zoom level and view center.

In addition, we have to ensure that subdividing a triangle does not produce T-junctions in the view-dependent mesh. Therefore, when we decide to refine a triangle $\mathbf{f}_1$ by subdividing its longest edge, we also refine the triangle $\mathbf{f}_0$, which shares the same longest edge and is thus a *dependent triangle* for $\mathbf{f}_1$. An example of dependencies in a simple mesh is shown in Fig. 5.

Geomorphing: Depending on the user actions while exploring an image, the LOD cuts may be drastically different between two sequential frames. This can cause popping artifacts that can be unpleasantly noticeable to users. In order to reduce popping artifacts and provide smooth transitions between different frames, we propose to use geomorphing [13], a well-known technique in the field of view-dependent rendering. To perform geomorphing for our ICAZ method, we maintain two separate LOD cuts of the current and next frames. Since the mesh structures (vertices and mesh connectivities) of these LOD cuts can be quite different, we refine these two LOD cuts until they share the same vertices and mesh structure; this can be easily done based on our multi-resolution hierarchy. Then, we perform a simple linear interpolation on the vertex coordinates of the two LOD cuts, in order to provide a smooth transition between the two sequential frames.

6 Results

In this section we present the results generated using the proposed ICAZ operator. We qualitatively compare the results of our method against those obtained with traditional zooming and *Retargeting+Zooming*. We also compare the running time of our method with and without the proposed view-dependent mesh. In addition, we refer the reader to the accompanying video that illustrates the interactive performance of our operator.

We implemented our ICAZ operator in C++ with the CHOLMOD library [7] for the direct sparse linear solver, an automatic saliency detection method [5], and with a conforming Delaunay triangulation [22] for the initial mesh construction. In all our experiments, we use a 2.66Ghz Xeon CPU with 4GB RAM. Our current implementation is single-threaded.

6.1 Visual Quality Analysis

Fig. 6 shows results of our method tested with various images. We compare our method with two other approaches: 1) traditional zooming, which combines cropping and uniform scaling, and 2) the *Retargeting+Zooming* method, which first applies mesh warping on the input image to modify its aspect ratio, followed by traditional zooming on the retargeted image. To be fair, we include the same line constraints in *Retargeting+Zooming* and our approach, by using the mesh warping method described in Sec. 4.

We use various input images listed in Table 1; for each of them, we automatically compute a significance map that we allow to be adjusted at runtime. In addition, we use the following fixed parameters for all tested images: the *maximum triangle area during initial triangulation* is set to 10K pixels, and the *maximum triangle area on screen space* at runtime is set to 500 pixels. To provide a fair comparison, the resolution of the uniform mesh used in the *Retargeting+Zooming* approach corresponds to the finest resolution of the adaptive mesh used with our method.

As can be seen from our results (Fig. 6), our ICAZ operator is effective at capturing significant image contents in the zoomed-out images (Fig. 6, column **c**), due to the use of a warp-based retargeting technique. Also, compared to traditional zooming (column **a**), ICAZ better utilizes the screen space to display details of significant regions and gets rid of the unattractive "black bands". As the user zooms into the image, our ICAZ operator unrolls the distortion *continuously* by modifying the boundary constraints of retargeting, which results in a less distorted image (columns **d**, **e**). Our result with 100% zoom level (column **f**) shows no distortion and corresponds to a cropped portion of the original image, whereas the *Retargeting+Zooming* approach contains obvious visual artifacts and suffers from a loss of detail at this level (column **b**).

Therefore, the ICAZ operator provides a global approximate view that utilizes the entire screen space when users zoom out, a local and accurate view when they zoom in, and continuous transitions in between. In the intermediate levels the result images show minor distortion, while significant regions are better preserved. This *Interactive Content-Aware Zooming* process can help users to navigate in large images with multiple points of interest; we refer readers to the **accompanying video** for more results.

6.2 Performance Analysis

To evaluate the performance of our ICAZ operator, we measure its running time on different benchmark images, with the same fixed parameters as mentioned in Sec. 6.1, and a target display of resolution 480×360 pixels. For each benchmark image, we record the *viewing path* of a user exploring the image, i.e. the sequence of panning and zooming operations; each of the viewing paths covers the entire range of zoom levels. We then compute the average time required for each panning or zooming action (see Fig. 8).

The first three tested images in Fig. 8 do not include line constraints. On such images, it takes less than 130ms to perform any panning or zooming operation using ICAZ; most of the running time of our method is spent in the fine-level mesh warping process, which has a larger vertex count. The coarse-level mesh warping process typically takes less than 20ms: as the connectivity of the coarsest mesh is fixed, we can compute the *Cholesky factorization* of the corresponding linear system matrix once, and then use fast *back-substitutions* [18] to efficiently solve the system.

In case of line constraints as defined in Sec. 4.2, the system matrix has to be updated at each iteration of the iterative solver. Therefore, its factorization is not fixed and the linear system cannot be efficiently solved. This explains the increased running time of ICAZ on the "power station" image, which includes line constraints.

In all the cases, the overhead spent on mesh refinement is low, typically around 5ms per operation. In addition, please note that panning operations do not require coarse-level mesh warping: since

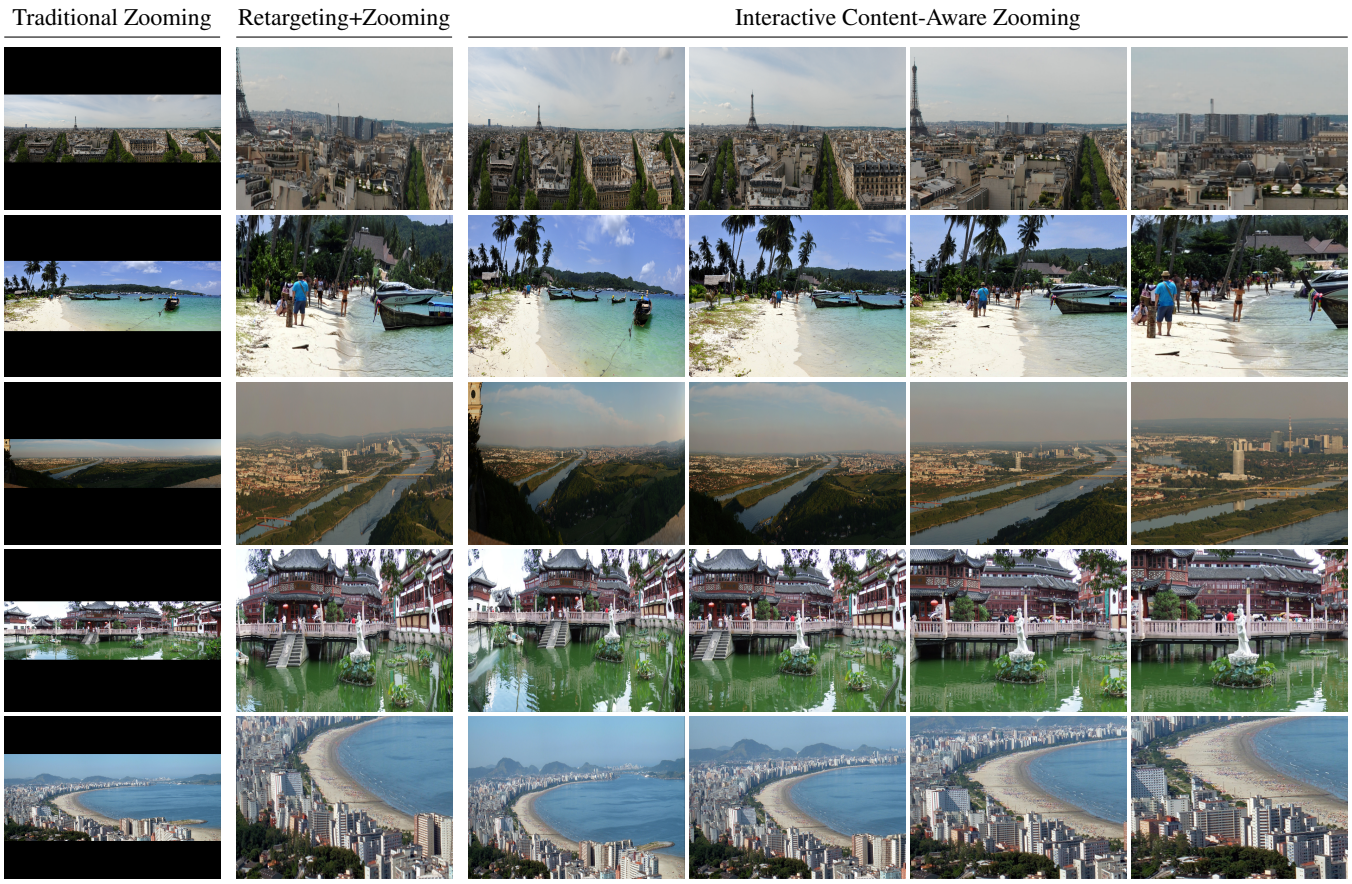

Figure 6: Visual quality comparison of the results obtained by (a) traditional zooming, (b) *Retargeting+Zooming*, (c-f) *Interactive Content-Aware Zooming* operators, on a 640×480 target display (400×300 for the "santos" image in the bottom row). For each input image, the results obtained at different zoom levels all share the same view center. Please zoom into the images to see them in higher resolution.

Figure 7: Results of two retargeting approaches in an extreme deformation case (the "bigyuyuan" image has been reduced to aspect ratio 4/3): (a) a result of the warping method described in Sec. 4, with our single line constraint (purple line on the balcony), (b) a seam carving result computed by the "Content-Aware Scaling" tool of Adobe Photoshop CS4.

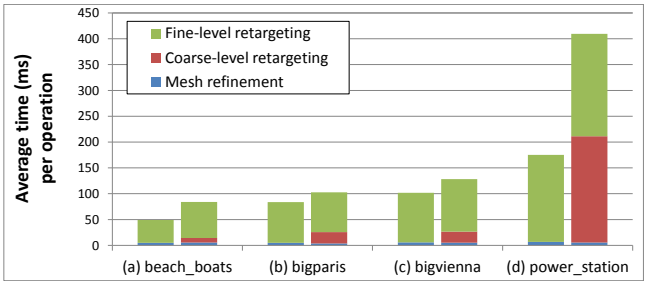

Figure 8: Average running time of individual panning (left) or zooming (right) operations, on different input images. The first three test images do not include line constraints. In all cases, the time required for the image reconstruction step is negligible, and is therefore not included.

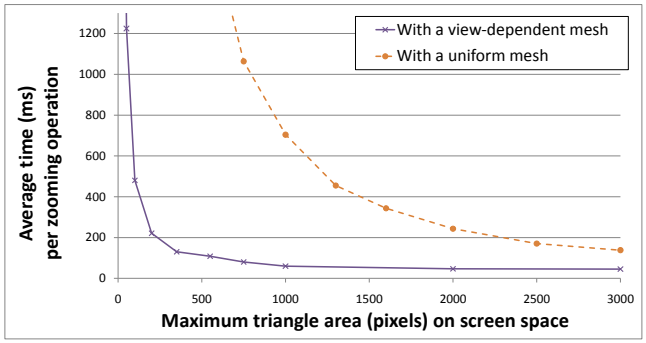

Figure 9: Average running time per zooming operation on the "bigvienna" image, using either the proposed view-dependent mesh or a uniform triangle mesh. Each sample on the graph corresponds to a value of the *maximum triangle area in screen space* parameter, which affects the visual quality of final results: finer triangles better distribute the distortion, as illustrated in Fig. 2. In our current system, we use a value of 500 pixels to provide an interactive performance for the ICAZ operator.

the zoom level does not change, the deformed coarsest mesh is already known from the previous frame.

Influence of the view-dependent mesh: In order to analyze the influence of mesh resolution on the performance of our ICAZ operator, we measure its average running time per zooming operation. We use different values of the *maximum triangle area on screen space* parameter, which controls the number of triangles in the screen space and viewing area, and thus influences the total vertex count and the result quality. We compare the running time of ICAZ with and without the proposed adaptive view-dependent mesh and the two-step mesh warping method; in the latter case, for each zoom level we use a uniform mesh that does not depend on the viewing area. As shown on Fig. 9, the ICAZ operator combined with a view-dependent mesh is significantly faster, for all the tested parameter values.

In addition to these results, we noticed that the running time of ICAZ without our view-dependent mesh is highly dependent on the zoom level. Even without using a view-dependent mesh, ICAZ shows interactive performance for panning operations, and zooming operations at a low zoom level. However, as we zoom in, a finer mesh has to be used in order to maintain a high retargeting quality; thus, it can take several seconds to compute a highly zoomed-in version of the image. On the other hand, with our view-dependent mesh the time required for each operation remains nearly constant during a navigation session, irrespective of the zoom level since we maintain a nearly constant number of vertices in the mesh. Therefore, using the settings described in Sec. 6.1, we can show interactive response to any action performed by users.

Nested iteration scheme: Each step of our two-level mesh warping process requires an initial guess for the iterative solver. A first approach would use the positions of vertices in the initial mesh, scaled to fit the target size. Instead, we use a *nested iteration scheme* [3], where the result of the coarse-level mesh warping is used as the initial guess for the fine-level mesh warping. We found that this approach reduces the number of iterations of the fine-level solver by a factor of two on average.

6.3 Discussion

The numbers reported in Fig. 8 and Fig. 9 refer to the computation time required to deform the view-dependent mesh after a single action from the user, without geomorphing. While our system does not achieve interactive speed in terms of *frames per second*, its performance should rather be evaluated in terms of *delay* between a user action and the system response, since users will spend most of the time observing stills. In this context, our system does provide *interactive response* to user actions. The smooth transitions, produced by geomorphing, reinforce this impression of interactivity (as shown in the accompanying video). However, the performance of our system could still be improved in a number of ways, for example by replacing the direct sparse linear system solver of our fine-level mesh warping by an iterative solver (with an initial guess resulting from the coarse-level mesh warping) or using a Full Multi-Grid approach [3], which would take advantage of the hierarchical representation of our mesh.

Image name	Resolution	Line constraints?
power station	7475×1999	yes
bigparis	7203×2247	no
beach boats	5530×1799	no
bigvienna	8000×1811	no
bigyuyuan	2230×598	yes
santos	1280×515	no

Table 1: Benchmark images shown in our results. We use "power station" in Fig. 1, and following images in Fig. 6.

Since ICAZ uses a retargeting method based on mesh warping, it inherits some limitations of such approaches. In particular, when the input image has a very different aspect ratio from that of the target screen, the zoomed-out image might look highly distorted if the image does not contain sufficient insignificant areas to divert the distortion into. An example is shown in the fourth row of Fig. 6: our results at low zoom levels show a distorted balcony on the right of the statue (column **c**). However, these results are usually better than those of seam carving, which tends to break structures and introduce discontinuities (Fig. 7, column **b**). In addition, our approach is effective at unrolling the distortion as users zoom in, since artifacts are hardly noticeable at higher zoom levels (Fig. 6, columns **e, f**). Alternative retargeting methods such as patch-based approaches [23, 8, 2, 19] might help to reduce the distortion at low zoom levels, but at the expense of higher computation costs and potential popping artifacts when zooming into the image.

Another limitation concerns the significance map, which has to be fairly accurate to sufficiently discriminate between significant and insignificant regions. Automatically computed significance maps often fail at detecting truly context-significant regions, and different users might consider varying image parts as significant. Therefore, we let them adjust the significance map at runtime, since our multi-resolution mesh allows propagating significance changes in a hierarchical manner.

Lastly, a few line constraints can help preserve the global image layout: on the examples where they were used we defined only 1 to 2 line constraints, either on the horizon line or on important structural features such as the balcony in "bigyuyuan" (Fig. 7, column **a**). However, including line constraints clearly increases the computational cost of mesh warping (Fig. 8). In addition, using several line constraints may sometimes over-constrain the system, which results in highly distorted images in the case of aggressive aspect ratio changes. Therefore, we encourage adjusting the importance of image regions instead, as this leads to fair results on most tested images.

7 Conclusions and Future Work

We have proposed an interactive content-aware zooming operator which makes it possible to efficiently visualize high resolution images on small screen spaces with potentially very different aspect

ratios. We retarget the original image to make it fit into the small screen space and then, when the user zooms in, unroll the distortion. The proposed operator combines the advantages of a global, but approximate view at lower zoom levels, and local, but accurate views at higher zoom levels. Interactive performance on large images is achieved through an adaptive view-dependent mesh with finer triangles in the viewing area only, while providing a high retargeting quality and maintaining a near-constant runtime performance.

We showed the effectiveness of our approach by comparing it with traditional zooming, and a method that naively zooms into the retargeted version of an input image. To our knowledge, the proposed method is the first to take advantage of image retargeting for interactively navigating in high resolution images.

Future work will investigate the design of a significance-aware construction method for the initial mesh: using smaller triangles in regions that have high gradients on the significance map might help distribute the distortion more smoothly. Since our ICAZ operator is flexible regarding the choice of the mesh warping method used, integrating recent works such as [29] might also improve the quality of our results, at a low implementation cost.

We plan to extend our work towards two orthogonal directions. First, to allow exploring *gigapixel images* with the help of our ICAZ operator; retargeting such images raises a number of issues, since they require a larger vertex count, even with a very coarse mesh. We would also like to design a light-weight version of our approach, specifically tailored to allow visualizing images interactively on hand-held devices, such as recent smartphones.

ACKNOWLEDGEMENTS

The authors wish to thank Tong-Yee Lee, Yu-Shuen Wang, Hui-Chih Lin, Robert Carroll and Yanwen Guo for sharing some of their results, as well as SeungYong Lee and the anonymous reviewers for their useful comments.

We also thank Bernhard Vogl (“bigvienna”), Wikimedia user MarcusObal (“bigparis”), Flickr users Destinys Agent (“power station”), motorito (“beach boats”), andrewstern (“bigyuyuan”) and Diego_3336 (“santos”), for letting us use their images under cc or cc-by-nc licence.

This work was supported in part by MKE/MCST/IITA [2008-F-033-02], MKE/IITA u-Learning, and KRF-2008-313-D00922.

REFERENCES

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Trans. Graph.*, 26(3):10, 2007.

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), Aug. 2009.

[3] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A multigrid tutorial (2nd ed.)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[4] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vision*, 74(1):59–73, 2007.

[5] N. J. Butko, L. Zhang, G. W. Cottrell, and J. R. Movellan. Visual saliency model for robot cameras. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2398–2403, 2008.

[6] R. Carroll, M. Agrawala, and A. Agarwala. Optimizing content-preserving projections for wide-angle images. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2009)*, 28(3), 2009.

[7] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Trans. Math. Softw.*, 35(3):1–14, 2008.

[8] T. S. Cho, M. Butman, S. Avidan, and W. Freeman. The patch transform and its applications to image editing. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[9] W. Dong, N. Zhou, J.-C. Paul, and X. Zhang. Optimized image resizing using seam carving and scaling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2009)*, 29, 2009.

[10] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, C. Aldrich, and M. B. Mineev-Weinstein. Roaming terrain: real-time optimally adapting meshes. In *VIS '97: Proceedings of the 8th conference on Visualization '97*, pages 81–88, Los Alamitos, CA, USA, 1997. IEEE Computer Society Press.

[11] R. Gal, O. Sorkine, and D. Cohen-Or. Feature-aware texturing. In *Proceedings of Eurographics Symposium on Rendering*, pages 297–303, 2006.

[12] Y. Guo, F. Liu, J. Shi, Z.-H. Zhou, and M. Gleicher. Image retargeting using mesh parametrization. *IEEE Transactions on Multimedia*, 11(5):856–867, 2009.

[13] H. Hoppe. View-dependent refinement of progressive meshes. In *SIGGRAPH*, pages 189–198, 1997.

[14] H. Huang, T. Fu, P. L. Rosin, and C. Qi. Real-time content-aware image resizing. *Science in China Series F: Information Sciences*, 52:172–182, 2009.

[15] J. Kopf, M. Uyttendaele, O. Deussen, and M. F. Cohen. Capturing and viewing gigapixel images. *ACM Trans. Graph.*, 26(3):93, 2007.

[16] P. Krahenbuhl, M. Lang, A. Hornung, and M. Gross. A system for retargeting of streaming video. *ACM Trans. Graph.*, 28(5), 2009.

[17] D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner. *Level of Detail for 3D Graphics*. Morgan-Kaufmann, 2002.

[18] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 2007.

[19] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *ICCV'09*, Kyoto, Sep-Oct 2009.

[20] M. Rubinstein, A. Shamir, and S. Avidan. Improved seam carving for video retargeting. In *SIGGRAPH*, pages 1–9, New York, NY, USA, 2008. ACM.

[21] M. Rubinstein, A. Shamir, and S. Avidan. Multi-operator media retargeting. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2009)*, 28(3), 2009.

[22] J. R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In M. C. Lin and D. Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer-Verlag, May 1996.

[23] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani. Summarizing visual data using bidirectional similarity. In *CVPR*. IEEE Computer Society, 2008.

[24] R. Szeliski. Image alignment and stitching: a tutorial. *Found. Trends. Comput. Graph. Vis.*, 2(1):1–104, 2006.

[25] Y.-S. Wang, H. Fu, O. Sorkine, T.-Y. Lee, and H.-P. Seidel. Motion-aware temporal coherence for video resizing. *ACM Trans. Graph. (Proceedings of ACM SIGGRAPH ASIA)*, 28(5), 2009.

[26] Y. S. Wang, C. L. Tai, O. Sorkine, and T. Y. Lee. Optimized scale-and-stretch for image resizing. In *SIGGRAPH Asia '08: ACM SIGGRAPH Asia 2008 papers*, pages 1–8, New York, NY, USA, 2008. ACM.

[27] L. Wolf, M. Guttmann, and D. Cohen-Or. Non-homogeneous content-driven video-retargeting. In *Proceedings of the Eleventh IEEE International Conference on Computer Vision (ICCV-07)*, 2007.

[28] S.-E. Yoon, E. Gobbetti, D. Kasik, and D. Manocha. *Real-Time Massive Model Rendering*. Morgan & Claypool Publisher, 2008.

[29] G.-X. Zhang, M.-M. Cheng, S.-M. Hu, and R. R. Martin. A shape-preserving approach to image resizing. *Computer Graphics Forum, Special issue of Pacific Graphics 2009*, Vol. 28(7):1897–1906, 2009.

[30] L. Zhang, M. H. Tong, T. K. Marks, H. Shan, and G. W. Cottrell. SUN: A Bayesian framework for saliency using natural statistics. *J. Vis.*, 8(7):1–20, 12 2008.