HAL
open science

Image-Based Rendering by Joint View Triangulation

Maxime Lhuillier, Long Quan

To cite this version:

Maxime Lhuillier, Long Quan. Image-Based Rendering by Joint View Triangulation. IEEE Transactions on Circuits and Systems for Video Technology, 2003, 13, pp.1051-1063. hal-00118513

HAL Id: hal-00118513

https://hal.science/hal-00118513

Submitted on 5 Dec 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Image-Based Rendering by Joint View Triangulation

Maxime LHUILLIER[1] and Long QUAN[2]

[1] LASMEA-UMR 6602 UBP/CNRS, France

[2] Hong Kong University of Science and Technology, Hong Kong

lhuillie@lasmea.univ-bpclermont.fr, quan@cs.ust.hk

Abstract

The creation of novel views using pre-stored images or image-based rendering has many potential applications, such as visual simulation, virtual reality and telepresence, for which traditional computer graphics based on geometric modeling would be unsatisfactory particularly with very complex 3D scenes. This paper presents a new image-based rendering system that tackles the two most difficult problems of image-based modeling: pixel matching and visibility handling. We first introduce the joint view triangulation (JVT), a novel representation for pairs of images that handles the visibility and occlusion problems created by the parallaxes between the images. The joint view triangulation is built from matched planar patches regularized by local smooth constraints encoded by plane homographies. Then, we introduce an incremental edge-constrained construction algorithm. Finally, we present a pseudo-painter's rendering algorithm for the joint view triangulation and demonstrate the performance of these methods experimentally.

Key words: image-based rendering, interpolation, triangulation, occlusions, matching.

1 Introduction

Recently, there has been much interest in computer vision and graphics in image-based rendering (IBR) methods [28], which generate new views of scenes from novel viewpoints, using a collection of images as the underlying scene representation. Compared with classical rendering [47] based on explicit geometric and photometric models, the images produced by image-based rendering systems are often more realistic, and both on-line rendering and off-line processing are independent of the geometric and photometric complexity of the underlying scenes. This paper describes an approach to image-based rendering that we have developed over the past few years.

Reconstruction-based methods One natural approach to IBR is classical computer vision based 3D reconstruction with rendering by texture mapping. Arbitrary views of the scene can be synthesized by re-projecting the reconstructed 3D model. Typical examples of this approach are [30, 15, 42] among others. More recent work has shown that multi-image matching constraints —the fundamental matrix for two views and the trifocal tensor for three can be used to synthesize new views without explicit 3D reconstruction. Laveau and Faugeras [17] use fundamental matrices and Avidan and Shashua [1] use trifocal tensors. These matching tensor methods are essentially equivalent to implicit 3D reconstruction methods, and as explicit reconstruction methods, they require rigid 3D scenes.

Interpolation-based methods In computer graphics, image-based rendering is often viewed as a problem of interpolation from a collection of images, inspired by techniques that generate smooth transitions between reference images by simply interpolating each pixel from the first to the second image value. For instance, Beier and Neely's morphing [5] uses line segments specified and matched by an animator. Lee et al. [19] study different warping strategies. Chen et al. [7, 6] and the QuickTimeVR products popularized the idea of direct pixel-by-pixel interpolation, however both originally assumed that the pixel correspondences in the basis images were given, as the basis images were computer rendered. Seitz and Dyer [34, 35] investigate view interpolation, but are mainly concerned with physically valid view generation via rectification of a perspective image pair following the linear combination method developed for object recognition of affine images. Like in the reconstruction-based methods, they also aim at rendering rigid scenes. A more abstract formulation on which a large amount of work has been based [28, 21, 13] is the light field or plenoptic function. This models all sets of rays seen from all points, considering each image as a set of rays. Image-based rendering is then about reconstructing this plenoptic function from the available images. The major challenge is the very high dimensionality of such plenoptic functions. Many simplifying assumptions that limit the underlying viewing space have been introduced: 5D plenoptic modeling [28], 4D Lightfield/Lumigraph [21, 13], 3D concentric mosaics [38] and 2D panorama [31, 6, 41].

All of these image-based rendering methods can be viewed as different sampling methods of the plenoptic function. Explicit and implicit reconstruction-based methods use sparse sampling while interpolation-based ones use dense sampling. Dense sampling requires huge numbers of pre-stored images to render new views. Sparse sampling needs fewer images, but it must face two difficult problems: pixel correspondence to recover depth information and occlusion analysis to handle the parallax between images. In the computer graphics community, these problems are usually avoided by using either a human animator [5] or computer-generated range data [7]. In computer vision, most of the currently available systems are based on the classical stereo algorithms [30, 15]. In this paper, we describe an image-based rendering approach that explic-

itly tackles the correspondence and occlusion problems. The first step is a reliable automatic matching algorithm called quasi-dense matching, which starts by selecting reliable seed matches and propagates these to neighboring pixels using a region-growing technique [25]. The result is a quasi-dense disparity map. The second step applies a homographic piece-wise smoothness constraint to construct robustly matched planar patches between pairs of images. To represent pair-image visibility and occlusion constraints, we propose a new representation called the joint view triangulation (JVT). This is constructed using a robust algorithm to separate matched areas from unmatched ones and to handle the partially occluded areas. Finally, we develop a pseudo-painter rendering algorithm from the joint view triangulation to synthesize new images. The paper therefore contributes to almost all aspects of image-based rendering systems: matching, pair-view representation and rendering. Early versions of this work appeared in conference papers [22, 23].

The paper is organized as follows. Section 2 reviews the quasi-dense matching algorithm based on propagation although it is not the subject of this paper. Section 3 and 4 describe, respectively, how to build a joint view triangulation and the pseudo-painter's rendering algorithm. Section 5 demonstrates the system with intensive examples and Section 6 gives some concluding remarks and suggestions for future research directions.

2 Review of quasi-dense disparity map construction

In different images, matching either high-level image primitives such as feature points and line segments or just pixels is probably the most difficult problem. This problem has been particularly studied for a stereo rig in which the relative orientation reduces the search space from the 2D image plane to 1D along epipolar lines [16, 10]. Meanwhile, the state of the art on matching does not yet give very satisfactory general results. In fact, almost all matching algorithms have trouble with either occlusion or untextured areas. This is not surprising as there is not enough information available in these areas for rendering decisions to be made. This has motivated the development of quasi-dense matching [25]. The key remark is that the disparity map could never be dense everywhere. The best we can hope is that only a set of sparsely distributed dense regions. This quasi-dense disparity map defined as such is therefore a more realistic goal.

The construction of the quasi-dense disparity map starts from matching some points of interest that have the highest textureness as seed points to bootstrap a region-growing type algorithm to propagate the matches in its neighborhood from the most textured (therefore the most reliable) pixels to the less textured ones. The algorithm could therefore be described in two steps: Seed selection and propagation, which are illustrated in Figure 1 between the first and the twentieth images of the flower garden sequence.

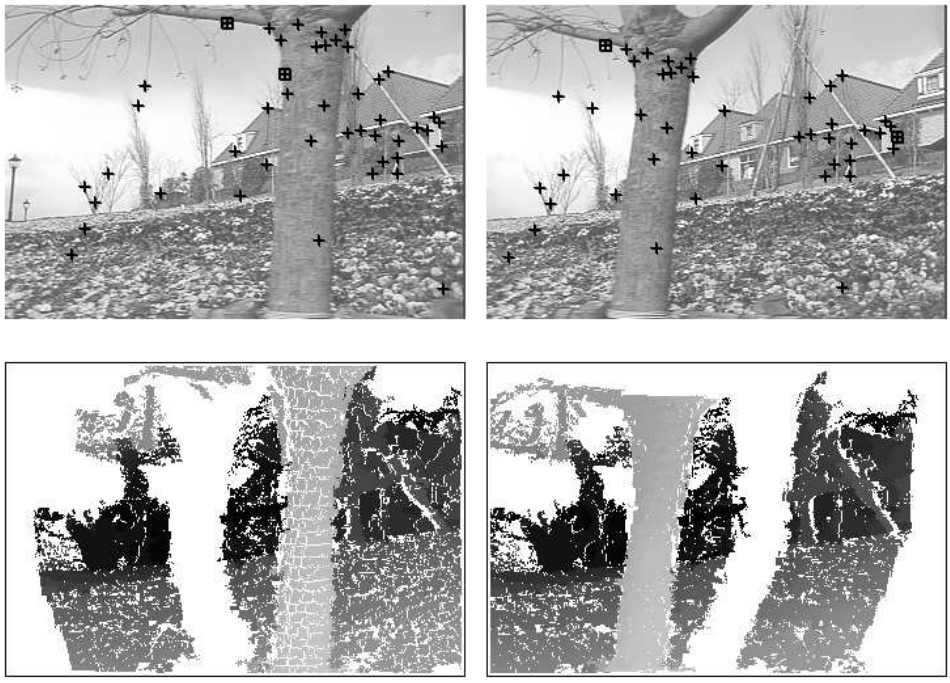

Figure 1: Top: initial seed matches between the first and twentieth image of the flower garden image sequence (two false matches are marked by a box instead of a cross for the correct ones). Bottom: movement after propagation without enforcing the epipolar constraint.

Seed selection Points of interest [26, 14, 37] are naturally good seed point candidates, as points of interest are by their very definition image points that have the highest textureness, i.e., the local maxima of the auto-correlation function of the signal.

We first extract points of interest from two original images, then a correlation method is used to match the points of interest across the two images, followed by a cross validation for the pair of images. This gives the initial list of correspondences sorted by the correlation score. We use the Zero-mean Normalized Cross-Correlation (ZNCC) defined at point $\mathbf{x} = (x, y)^T$ with shift $\Delta = (\Delta_x, \Delta_y)^T$ by

$$ZNCC_{\mathbf{x}}(\Delta) = \frac{\sum_{\mathbf{i}}(I(\mathbf{x}+\mathbf{i}) - \bar{I}(\mathbf{x}))(I'(\mathbf{x}+\Delta+\mathbf{i}) - \bar{I}'(\mathbf{x}+\Delta))}{(\sum_{\mathbf{i}}(I(\mathbf{x}+\mathbf{i}) - \bar{I}(\mathbf{x}))^2 \sum_{\mathbf{i}}(I'(\mathbf{x}+\Delta+\mathbf{i}) - \bar{I}'(\mathbf{x}+\Delta))^2)^{1/2}}$$

where $\bar{I}(\mathbf{x})$ and $\bar{I}'(\mathbf{x})$ are the means of pixel luminances for the given windows centered at $\mathbf{x}$.

Propagation From the current seed list, which is initialized by the first step, at each step, we pull the best match from the list of seeds. Then we look for additional matches in the neighborhood of the best seed. The neighbors of a seed point are taken to be all pixels within the 5×5 window centered on the seed point. For each neighboring pixel of the first image, we first construct in the second image a list of tentative match candidates that consist of all pixels

of a 3×3 window in the neighborhood of its corresponding location in the second image (see Figure 2).

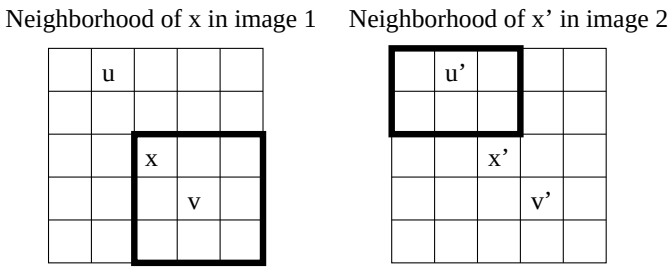

Figure 2: Possible matches $(\mathbf{u}, \mathbf{u}')$ and $(\mathbf{v}, \mathbf{v}')$ around a seed match $(\mathbf{x}, \mathbf{x}')$ come from its 5×5-neighbor $\mathcal{N}(\mathbf{x})$ and $\mathcal{N}(\mathbf{x}')$ as the smallest size for discrete 2D disparity gradient limit. The match candidates for $\mathbf{u}$ (resp. $\mathbf{v}'$) are within the 3×3 (black framed) centered at $\mathbf{u}'$ (resp. $\mathbf{v}$).

Such a match neighborhood enforces the continuity constraint and a disparity gradient limit of one pixel for the matching result. The matching criterion is still the ZNCC correlation score but within a smaller by half 5×5 window, therefore the "fattening" artifacts at the occluding contours are limited.

Finally, additional matches in the neighborhood of the current seed pair are added simultaneously in the match list and the seed list such that the match list is a one-to-one matching. The algorithm terminates when the seed list becomes empty.

This algorithm is efficiently implemented with a heap data structure for the seed list. Notice that as each time only the best match is selected, this drastically limits the possibility of bad matches. For instance, the seed selection step seems very similar to many existing methods [48, 44] for matching points of interest using correlation, but the crucial difference is that we need only to choose the most reliable ones rather than trying to match a maximum of them. In some extreme cases, only one good match of points of interest is sufficient to provoke an avalanche of the whole textured images. This makes our algorithm much less vulnerable. The same is true for propagation, the risk of bad propagation is considerably diminished by the best first strategy over all matched boundary points.

Rigid scenes and the aperture problem When the examples are rigid scenes, the epipolar geometry encoded by the fundamental matrix is integrated easily in the propagation step by the epipolar constrained propagation. This matrix is first estimated robustly from the unconstrained propagation result [24], or from the seed selection result [49].

More details about these matching steps are given in [25], including details on both rigid and non-rigid scenes. It is proposed in [25] that propagation walk and matching along edges in spite of the aperture problem be allowed. When the perspective distortion is moderate, the distance along the edges covered by the propagation is similar in both images, and we have found that

this behavior is very interesting for interpolation and morphing applications. Another choice would be to forbid propagation using one of the optical flow confidence measures [4].

3 Joint view triangulation—JVT

Triangulation is always necessary not only to remedy the sparseness of the disparity map, but also primarily to approximate images for rendering efficiency. The traditional independent triangulation operated on each individual image gives a good approximation when the occluded areas are negligible in the rendering view-field, but it becomes insufficient when the occlusions are apparent as illustrated in the garden flower sequence. In this section, we propose an original pairwise image description structure that we call *joint view triangulation*, which triangulates different images simultaneously and consistently (the term *consistency* will be precisely defined later). It can handle the most difficult occlusion problems between different images.

This new pair-image representation has been inspired by range view mesh integration approach in [45, 40], impostors for rendering graphics views [39] and layers [3, 46, 20, 2]. The layers representation is a very interesting one, but not all scenes can be approximated by layers and very often layers are created interactively. The JVT representation gives a much richer description than layers do and could be easily converted into appropriate layers depending on the underlying scenarios. Finally, another work about triangulation for rendering [29] does not include edge constraints and explicit modeling for artificial rectilinear objects contrary to our method described below.

We will first describe an intermediate step, match re-sampling, then define JVT and finally give an incremental algorithm for its implementation.

3.1 Match re-sampling

As quasi-dense matching gives an irregular distribution of matches, the matched pixels between two images have to be re-sampled. This re-sampling could equally be motivated by the necessity of post-match regularization to improve the match reliability by taking into account the local geometric constraints. We assume that the scene surface is at least piece-wise smooth. Therefore, instead of using global geometric constraints encoded by a fundamental matrix or a trifocal tensor, we could use local geometric constraints encoded by planar homography. The quasi-dense matching is thus regularized by locally fitting these plane homographies. As a by-product, this match re-sampling constructs also all regularized visible patches in two images.

The first image plane is initially divided into a regular grid of 8×8 pixel squares. The scale is a trade-off between the sampling resolution and the regularization stability. To increase the

resolution, a super-sampling grid is introduced by shifting the first grid by a half-size of the grid cell width as illustrated in Figure 3.

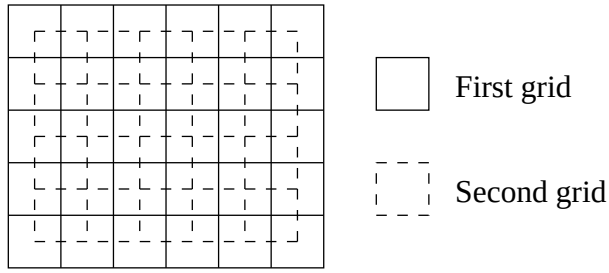

Figure 3: Two overlapping regular grids to subdivide the first image into square patches.

From each square patch, all matched points inside it from the disparity map are obtained. A plane transformation is tentatively fitted to these matched points. The most general linear plane transformation is a homography represented by a homogeneous 3×3 non-singular matrix. Four matched points, no three of them collinear, are sufficient to estimate a plane homography. Notice that an affine transformation encoded by 6 d.o.f. rather than a homography could also be used if the perspective distortion is mild between images.

Because a textured patch is rarely a perfect planar facet except for manufactured objects, the putative transformation for a patch cannot be estimated by standard least squares estimators. Robust methods have to be adopted. They provide a reliable estimate of the homography even if some of the matched points of the square patch are not actually lying on the common plane on which the majority lies. The Random Sample Consensus (RANSAC) method originally introduced by Fischler and Bolles [11] is used for robust estimation of the homography. RANSAC has been successfully used for robust computation of the geometric matching tensors in [43, 48]. If the consensus for the transformation reaches 75%, the square patch is considered as a valid planar patch. The location of the corresponding planar patch in the second image is defined by mapping the four corners of the grid in the first image with the estimated homography. This process of fitting the square patch to a homography is first repeated for all square patches of the first image for the two sampling grids. It turns out all matched planar patches at the end. Notice that the planar patches so constructed may overlap in the second image. To reduce the number of the overlapped planar patches, but not solve the problem, the corners of the adjacent planar patches are forced to coincide at an average point if they are sufficiently close. This is illustrated in Figure 4.

Each valid planar patch will be divided into 2 triangles along one of its diagonals for further processing. From now on, the meaning of a matched patch is more exactly a matched *planar* patch, as we will only consider the matched patches which succeed in fitting a homography.

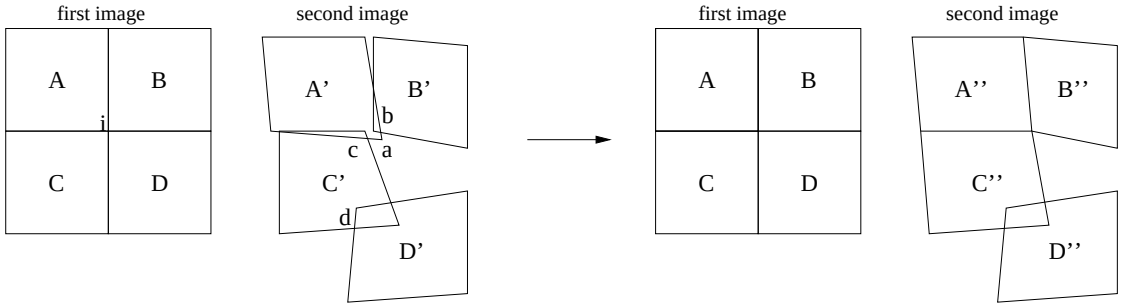

Figure 4: The patches A', B', C' and D' recomputed by the homographies in the second image correspond to the regular patches A, B, C and D in the first image. Because the corners a, b and c of different patches are very close, they are made to coincide in one common and averaged point. Note that this only improves but does not solve the overlapping problem of the patches, e.g., the patch C' and D' remain overlapped after this procedure.

3.2 Definition of the joint view triangulation (JVT)

The joint view triangulation is a "consistent" and constrained triangulation built on the resampled matched points. The "consistency" for joint view triangulation is defined as follows.

1. There is one-to-one correspondence between the matched image points;

2. There is one-to-one constrained Delaunay edge correspondence in two images. There are three types of Delaunay edge constraints:

   - **Boundary edges of matched areas** The matched areas from the quasi-dense disparity map represent the common visible visual events of the scenes. Their boundaries should not be crossed by any other triangles for later rendering. They should be preserved as the most natural constraints.

     However, if only boundary edges of matched areas were used, though it simplifies the implementation, it may give a poor approximation for the visual events. Natural outdoor scenes could be nicely handled, but the algorithm often produces undesirable artifacts for artificial objects. Two most typical rendering artifacts are the "broken line" artifact illustrated in Figure 5 and the "ghosting" artifact (double image) for thin structures like electric posts. To tackle these problems, two more constraints are introduced.

   - **Line segments** The most direct remedial measure for the broken line artifact is to integrate image contour points as constraints. The line segments are taken to be a polygonal approximation of linked contour points.

   - **Artificial rectilinear objects** Artificial rectilinear objects such as electric posts, trunk of trees, etc. are very frequent in outdoor scenes. They are often mishandled

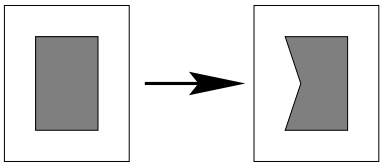

Figure 5: The "broken line" artifact: the original image of a gray box on a white background is broken on the border during the rendering step due to non-appropriate triangulation.

due to the size of the sampling grid, which is limited by stability considerations of the fitting procedure. When their detection and matching is possible, they are also integrated into Delaunay constraints. An explicit modeling of these objects is presented below.

3. The triangulation in each image is a constrained Delaunay triangulation.

   Delaunay triangulation is a good choice because of its minimal roughness property [33]. Recall that a constrained Delaunay triangulation [32] is a Delaunay triangulation in which the circum-circle of each triangle does not contain in its interior any other "visible points". Two points are said to be visible if they are not separated by a constraint edge.

3.3 Algorithm implementation

Putting all these constraints together, an edge-constrained JVT is implemented as an incremental insertion algorithm in both images simultaneously, consisting of five major steps. Figure 6 illustrates the evolution of the construction algorithm.

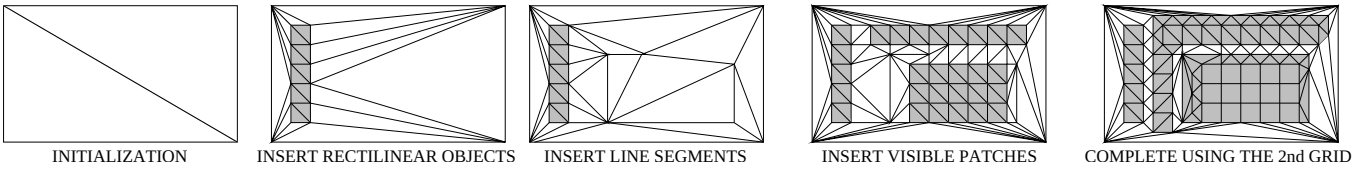

Figure 6: Illustration of the different steps of the edge-constrained joint view triangulation.

1. **Initialization**

   The initial empty images are triangulated by the diagonal from the top left corner to the bottom right one as illustrated in Figure 6.

2. **Detection of visible patches**

   This is the by-product provided by the match re-sampling section.

3. **Detection and insertion of rectilinear structures**

The vertical rectilinear structures are modeled as sets of connected visible patches. First, all vertical and connected triplets of patches are detected; then, all overlapping triplets are merged into connecting groups. Finally, each group is completed by the adjacent and vertical square containing a line segment. The whole procedure is illustrated in Figure 7.

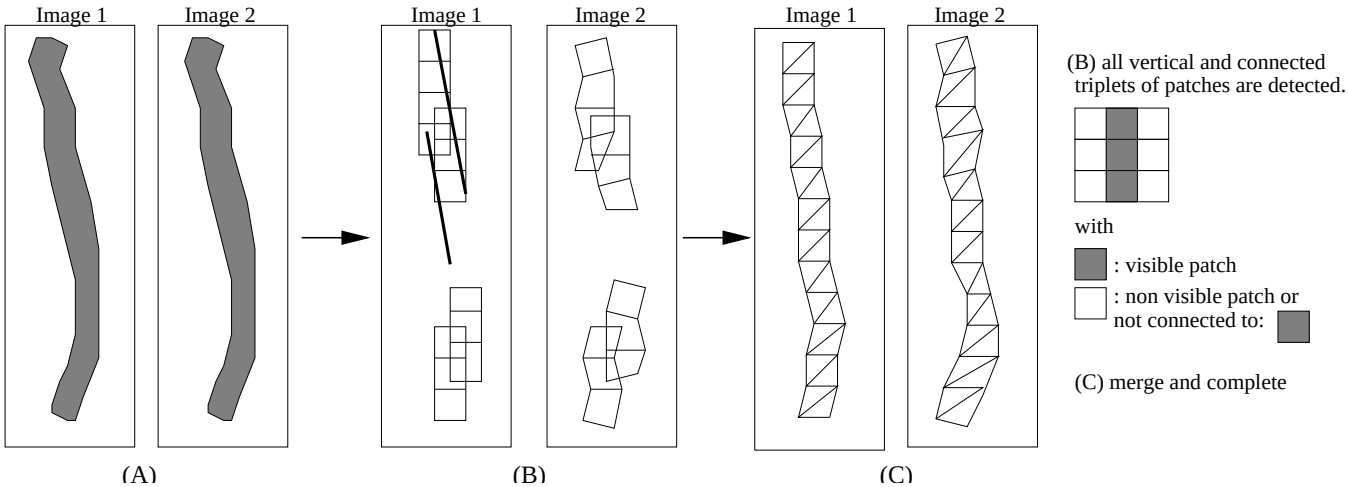

Figure 7: Illustration of detection of a (gray) rectilinear object whose width is about the size of the actual sampling grid. Middle: All corresponding vertical triplets of visible patches without left and right patch neighbors are first detected. Right: All overlapping triplets of patches form a connecting group. The vertical squares, which share a line segment with the group, are attached to the group.

These groups of connected patches are inserted into the current joint triangulation.

A real example for detecting a small wood post on the foreground and the consequent triangulation is shown in Figure 8.

4. **Detection and insertion of line segments**

The contour points as local maxima of gradient edge points are first detected by a Canny-like detector [9]; then, the connected contour points are linked and approximated by line segments. This procedure is performed only on the first image. We do not wish to match directly the line segments in two images as such a procedure is hardly stable. A deduction method illustrated in Figure 9 is implemented. This method deduces the corresponding line segment in the second image from the disparity map.

For any line segment detected in the first image, all corresponding points in the neighborhood of the end-points of the line segment are obtained from the disparity map. A tentative 1D transformation on the line segment in two images is estimated. This 1D

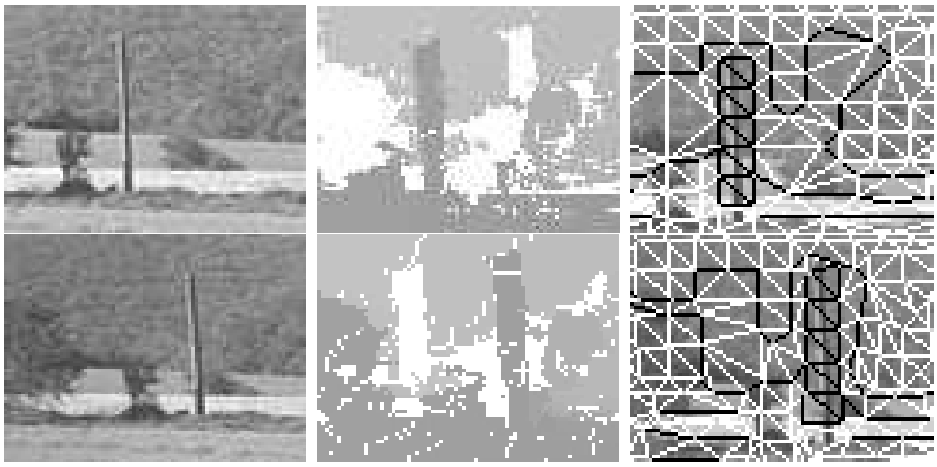

Figure 8: Left: Two views of a small wood post. Middle: The disparity map after the constrained propagation. Right: The modeling of the small post and the final JVT in which constraint edges are drawn in black.

transformation might be projectively encoded by a 2×2 homography by taking two end-points and one midpoint obtained from the epipolar geometry. If the epipolar geometry is not available, an affine transformation defined by two end-points is estimated instead.

The second stage of the algorithm re-samples the line segment by regularly dividing it into smaller parts. Each part is validated by taking into account the number of the match points around it from the disparity map. The final line segment is selected as the one that maximizes the number of matched segment parts.

The corresponding line segment is inserted in the current joint triangulation while not violating the existing constraints. If it crosses any existing constraint edge, the line segment is further split for inserting only its non-intersecting parts.

One example of the flower garden images is given in Figure 10.

5. **Insertion of visible patches from first sampling grid**

   All visible patches from the first sampling grid which do not intersect with any existing constraint edges are inserted into the current joint view triangulation.

6. **Boundary refinement from the super-sampling grid**

   The goal of this step is to refine and complete the polygonal boundaries of visible areas using the additional half-size-shifted sampling grid. First, all vertices of the visible patches from the shifted grid are inserted if they are outside the current visible patches in the current JVT. Each triangle touching the current boundaries is retained in the JVT if its surface is not too big and if an affine transformation can be fitted to the matched pixels within it. This step is illustrated in Figure 11.

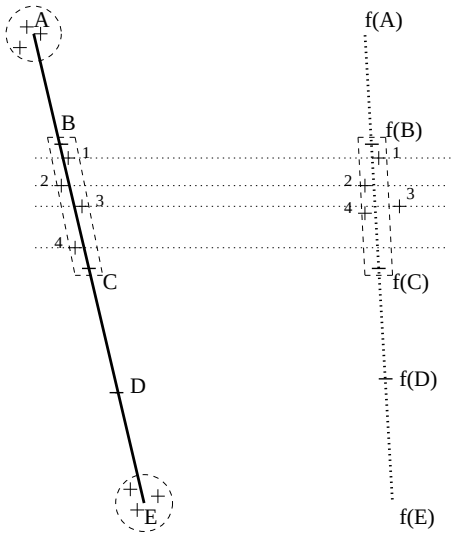

Figure 9: Illustrative figure for the deduction of corresponding line segments. AE is a line segment detected in the first image. A tentative corresponding line segment, f(A)f(E), is defined by the neighboring points of A and E such that a 1D homography f is fitted to all contour points along the segment. The segment AE is regularly divided into smaller parts, AB,BC,CD,DE, and each one is validated by counting the number of matches that satisfy f in their neighborhood. For instance, matches 1 and 2 satisfy the homography while 3 and 4 do not. The segment f(A)f(E), which maximizes the number of accepted smaller parts, is accepted to define the validated parts, for instance BC and f(B)f(C).

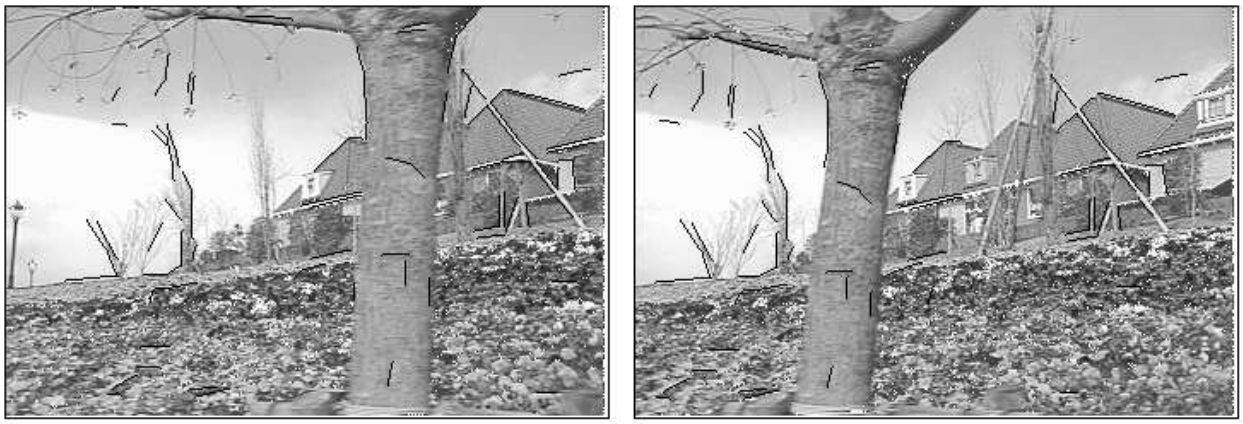

Figure 10: Detection of line segments as edge constraints by deduction from the disparity map. The corresponding line segments are displayed in black.

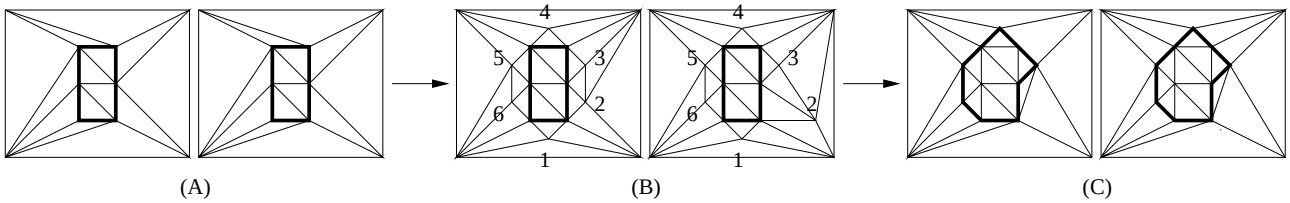

Figure 11: A: The current JVT with only the first grid. B: All vertices of visible patches from the second grid are inserted into the current JVT if they are outside the matched areas. C: Each triangle touching the current boundaries is tested. The triangle with vertex 1 is rejected as it fails fitting. The triangle with vertex 2 is also rejected as the triangular surface is too big in the second image. Finally, vertex 1 and 2 are removed from the JVT.

Results on the garden flower images (Figure 10) are shown in Figure 12. For the JVT without edge constraints, we can notice three broken line artifacts: one on the upper part of the tree; one on the middle of the tree; and one on the white oblique post. Most of these artifacts are removed with the edge-constrained JVT except the one in the middle of the tree, as there is no contour point detected in the first image.

It is also expected that this result is improved by using more than two shifted sampling grids and by combining the matched edges by the deduction method from the first to the second and from the second to the first image.

4 Rendering from JVT

This section describes how we use JVT to render novel views. If the depth information were available, rendering could be carried out by eliminating the hidden surfaces by using the Z-buffer or the painter's algorithm [12]. In the absence of depth information for the actual image-based approach, we develop a pseudo-painter's algorithm, in reference to how a painter draws final details over initial uncertain layers.

While the painter's algorithm with depth information is almost straightforward, the pseudo-painter's algorithm is much less straightforward as it lacks the critical depth information. The reliable and accurate depth information is difficult even with carefully lab-calibrated cameras. For image synthesis purpose, however, only a relative and qualitative depth order in the image space is sufficient. This order will be heuristically deduced from the JVT, as it approximates the images in triangular meshes while keeping the consistent correspondence information between the images.

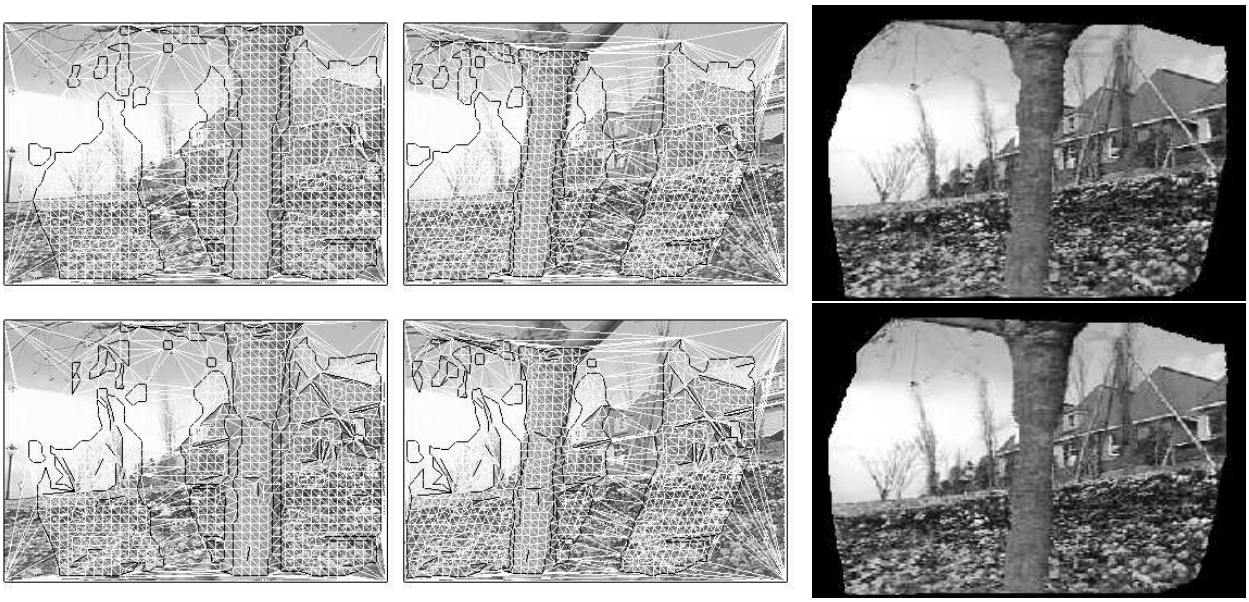

Figure 12: Top: The JVT without constrained edges of the flower garden image pair. The edges in black are the constrained edges and those in white are Delaunay ones. As only the borders of the matched areas are constrained, broken line artifacts appear on the upper and middle part of the front tree and in the middle of the white oblique post on the background. Bottom: The JVT with edge constraint. Two of the three above mentioned artifacts are removed, only the artifact on the middle part of the front tree persists.

4.1 Rendering order for matched pixels

It is relatively easy to deduce a depth order for pixels within matched triangles with JVT to deal with overlapping patches in the newly synthesized image. As the depth is related to the disparity, the depth order could be deduced from the epipolar geometry encoded by the fundamental matrix [18, 27, 8] on the assumption that the relative orientation between the virtual camera and the original camera is topologically known. For instance, it is sufficient to know that the virtual camera is located on the right, left, in front of or behind the original camera. In absence of any information, we apply the heuristic that an increasing disparity norm implies decreasing depth. This is what happens in a dominant lateral translational motion.

4.2 Rendering order for unmatched triangles

It is much more difficult to deduce a rendering order for pixels within unmatched triangles. Without any matching information, the missing disparity has to be somehow interpolated by relying on the approximating JVT.

When a pixel is unmatched, it usually comes from either an untextured area or an occluded area. For pixels from untextured areas, their disparities could simply be interpolated from those

of the surrounding boundaries. For pixels from half-occluded areas, two common cases should be distinguished as illustrated in Figure 13. The first one describes a big foreground motion, which produces a complete obstructed hole on the background in the other image. Once again, the disparities for the obstructed pixels could be interpolated from those of the surrounding matched pixels. The second case describes a relatively smaller foreground motion, which creates half-occluded areas between the fast-moving foreground object and the slowly moving background. The simple interpolation strategy would have created a undesirable quickly shrinking artifact, which is particularly undesirable for dynamic rendering. It is more appropriate to extrapolate background disparities to these half-occluded areas. This extrapolation is implemented via the virtual vertices.

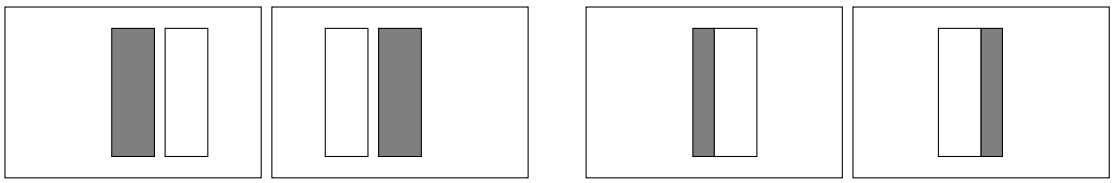

Figure 13: A scene is composed of a small vertical rectangle (as a trunk) in front of an infinite vertical plane. The left (resp. right) two columns show two views of the scene with a big (resp. small) camera translation and the corresponding half occluded areas in gray.

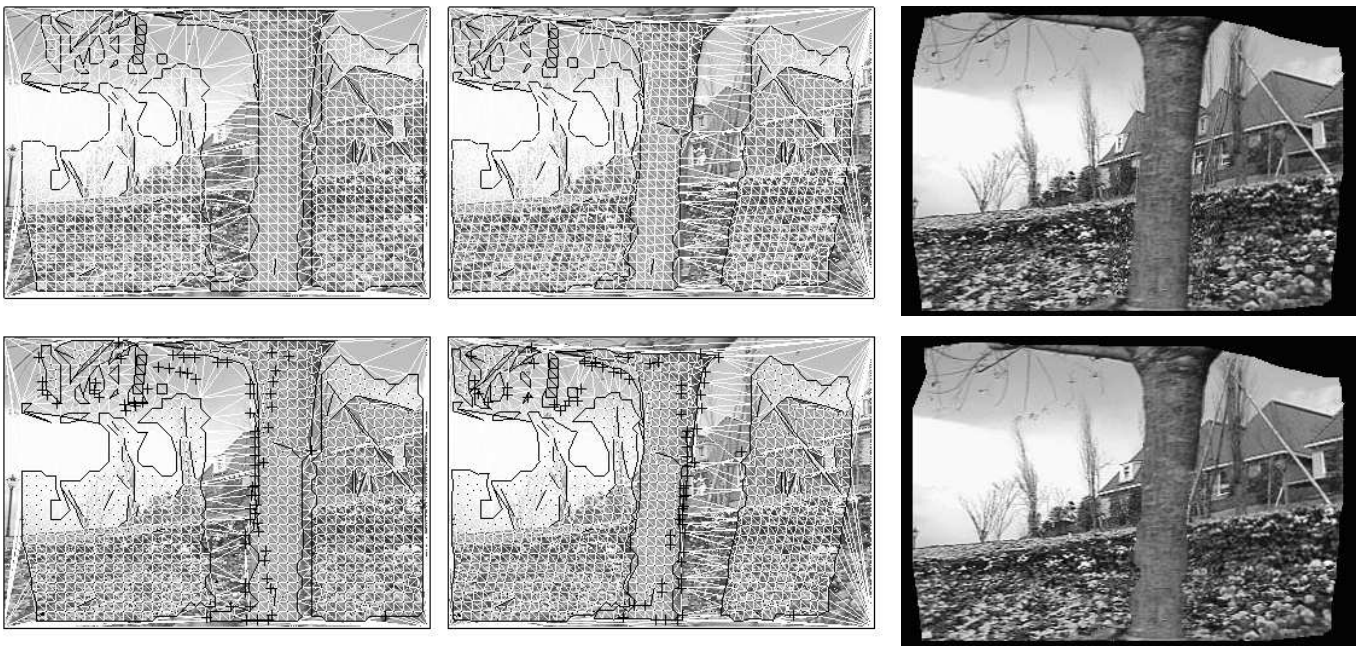

Figure 14: The first and tenth images of the flower garden sequence are used. Top: The JVT without virtual vertices (left) and the resulting rendering (right). Bottom: The JVT with virtual vertices (left) and the resulting rendering (right). We notice that virtual vertices reduce greatly the compressed background in the neighborhood of the trunk borders.

Now we describe how the virtual vertices are created and inserted into the current JVT. We look for all Delaunay edges shared by two unmatched triangles in both images. Then we retain those that have been sufficiently distorted in different images. The degree of distortion is measured for the corresponding edge by their relative lengths (not more than 1.3) and the angle between them (not more than 20 degrees). For the retained edge, one virtual vertex is created in the immediate neighbor of the fast-moving end-point, i.e., the end-point that has the larger disparity. This virtual vertex is inserted into the first triangulation of the JVT. Its correspondence in the second image remains implicit, is defined by the slow-moving end-point, i.e., the end-point that has the smaller disparity. The same is done for the second image. The JVTs with and without virtual vertices and their resulting renderings are shown in Figure 14 for the first and the tenth images of the flower garden sequence.

Finally, to define a rendering order for the pixels within unmatched triangles, a key observation is that the reliability of a given triangular patch is closely related to the degree of its distortion in different images. The more distorted the triangle, the less useful the texture it conveys for rendering. This suggests a heuristic rendering order for unmatched pixels to follow its bounding triangle distortion: most distorted triangles are drawn first and mildly transformed triangles are drawn last. A criterion is defined to measure the distortion degree in the algorithm. Obviously, all triangles containing at least one virtual vertex are painted first before all others.

4.3 The algorithm

A four-step algorithm that implements the above principle can be given as follows:

1. Create each triangle vertices in the synthesized image

   The position in the synthesized image, $\mathbf{u}''$, is transferred or interpolated from each corresponding vertex, $\mathbf{u}$ and $\mathbf{u}'$, of the triangles according to the trajectory specification of the virtual camera.

   Each virtual vertex is mapped onto the synthesized image from the virtual vertex position in the first (resp. second) triangulation and its implicit correspondence in the second (resp. first) one.

   Then, the following two steps are applied to each original image, $I(\mathbf{u})$ and $I(\mathbf{u}')$, to create two warped images, $\tilde{I}(\mathbf{u}'')$ and $\tilde{I}'(\mathbf{u}'')$.

2. Warp pixels within an unmatched triangle

   A weight w is assigned to each warped triangle to measure its distortion degree. The weight w is defined to be proportional to the ratio γ of the triangle surface in the first

image w.r.t. the second image bounded by 1, that is, $w = Min\{1, \gamma\}$ for the triangles of the first image and $w' = Min\{1, 1/\gamma\}$ for the triangles of the second image.

First, warp all unmatched triangles containing at least one virtual vertex; then, warp all other unmatched triangles in increasing distortion order w.

The triangles whose vertices are image corners are not considered.

3. Warp pixels within a matched triangle

   For each matched triangle $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2$ and $\mathbf{v}'_0, \mathbf{v}'_1, \mathbf{v}'_2$, the rendering order follows the maximum of disparity norms of its vertices: $Max\{||v_0 - v'_0||, ||v_1 - v'_1||, ||v_2 - v'_2||\}$.

   Warp all matched triangles in increasing disparity order.

   The last step recomposes the two warped images into the final synthesized image.

4. Color interpolation

   The final pixel color at $I''(\mathbf{u}'')$ for the synthesized image is obtained by blending the corresponding two weighted warped images, $\tilde{I}(\mathbf{u}'')$ and $\tilde{I}'(\mathbf{u}'')$:

$$I''(\mathbf{u}'') = \frac{(1-\lambda)w(\mathbf{u}'')\tilde{I}(\mathbf{u}'') + \lambda w'(\mathbf{u}'')\tilde{I}'(\mathbf{u}'')}{(1-\lambda)w(\mathbf{u}'') + \lambda w'(\mathbf{u}'')},$$

   where λ is the interpolation parameter to describe the path.

By simply using the in-between trajectory of two original garden flower images, the different steps of the rendering algorithm are shown in Figure 15. The rendered sequence is illustrated in Figure 16 in which some sample views are shown.

5 Experimental results

The whole working system described in this paper has been demonstrated on many real image pairs besides the garden flower example used throughout the paper as illustration.

Different camera trajectory As we have mentioned that although in-between interpolation is the easiest trajectory, any virtual camera trajectory could be specified by the so-called transfer equation [18, 36] or explicit move of the viewer's camera. A circular trajectory whose radius is the half of the unity between the two images and which is centered at the first camera is defined for the garden flower sequence. The rendered images are shown in Figure 17. We can notice more geometric distortion on the rendered images as the camera trajectory deviates far from the in-between path.

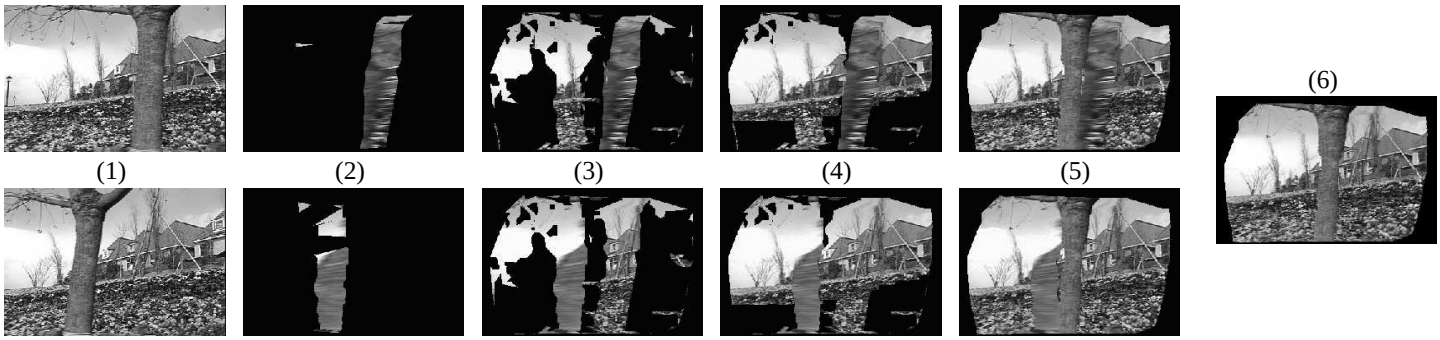

Figure 15: Some intermediate results of the interpolation for the middle frame of the garden flower pair to illustrate the rendering algorithm. The top row shows the drawing order for the first image and the bottom for the twentieth image. Frame 1 shows the original images. Frames 2 and 3 show the painting of the unmatched triangles. The most severely distorted ones are shown in Frame 2 and the remaining unmatched are in Frame 3. Frames 4 and 5 show the painting of the matched triangles. The matched triangles with lower disparities are shown in Frame 4 and those with higher disparities are in Frame 5. The final color blending of the two warped images is given in Frame 6. The final image has no inner black holes as all triangles (except those at the image borders) are rendered.

Figure 16: Some sample images of the interpolation: $\lambda = 0, 0.2, 0.4, 0.6, 0.8, 1$ from left to right.

Figure 17: Some samples of the rendered views for a circular trajectory.

Virtual vertices A typical usage of virtual vertices can be illustrated by the image sequence shown in Figure 18. We see how the half-occluded area connecting the foreground moving man and the background building is distorted due to different moving speeds. By introducing virtual vertices illustrated in Figure 19, the rendering results are considerably improved, though there are still slight artifacts due to finite approximating patches.

Figure 18: A typical usage of virtual vertices from the original image sequence. The fast-moving foreground man with slowly moving background buildings creates the half-occluded areas between the man and the building: The original images are on the left and right and the interpolated image is in the middle.

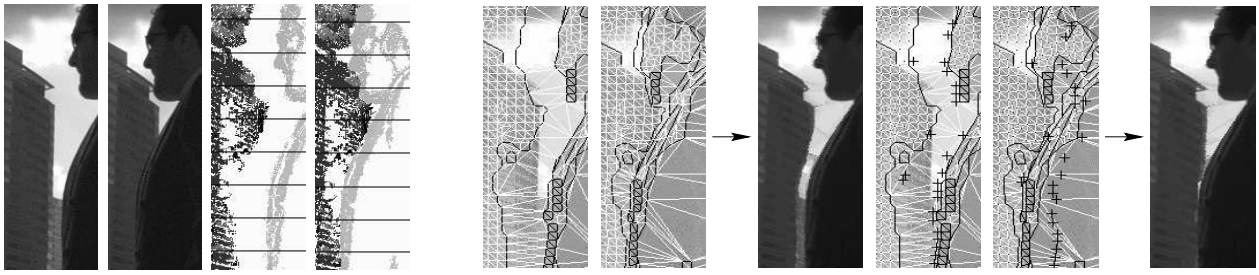

Figure 19: Illustration of the creation of virtual vertices and comparative results on rendering. Left: The disparity map for the sub-images. Middle and right: The rendering process without (middle) and with (right) virtual vertices. The rendered image with virtual vertices has no background distortion artifact.

Mixture of natural and man-made objects One typical example of a pair of outdoor building images is illustrated in Figure 20. This mixture of natural (trees) and man-made (street, building and posts) objects is difficult for several reasons: The leafs are hardly matched as the texture disparity is big and there is also a shortage of reliable seed matches; the epipolar lines are almost oriented the same way as the scene's horizontal direction; this makes the matching of vertical structures easier while it does not help matching the horizontal structures of the building. The matching error for the horizontal edges of the building and pavement may reach several pixels. The upper sky area and the lower road area are almost textureless. Good JVT results are obtained as shown in Figure 20. We can notice that in addition to the two

front posts, two false posts are also created in the background on the right side of the image, but it does not change the interpolation results illustrated in Figure 21.

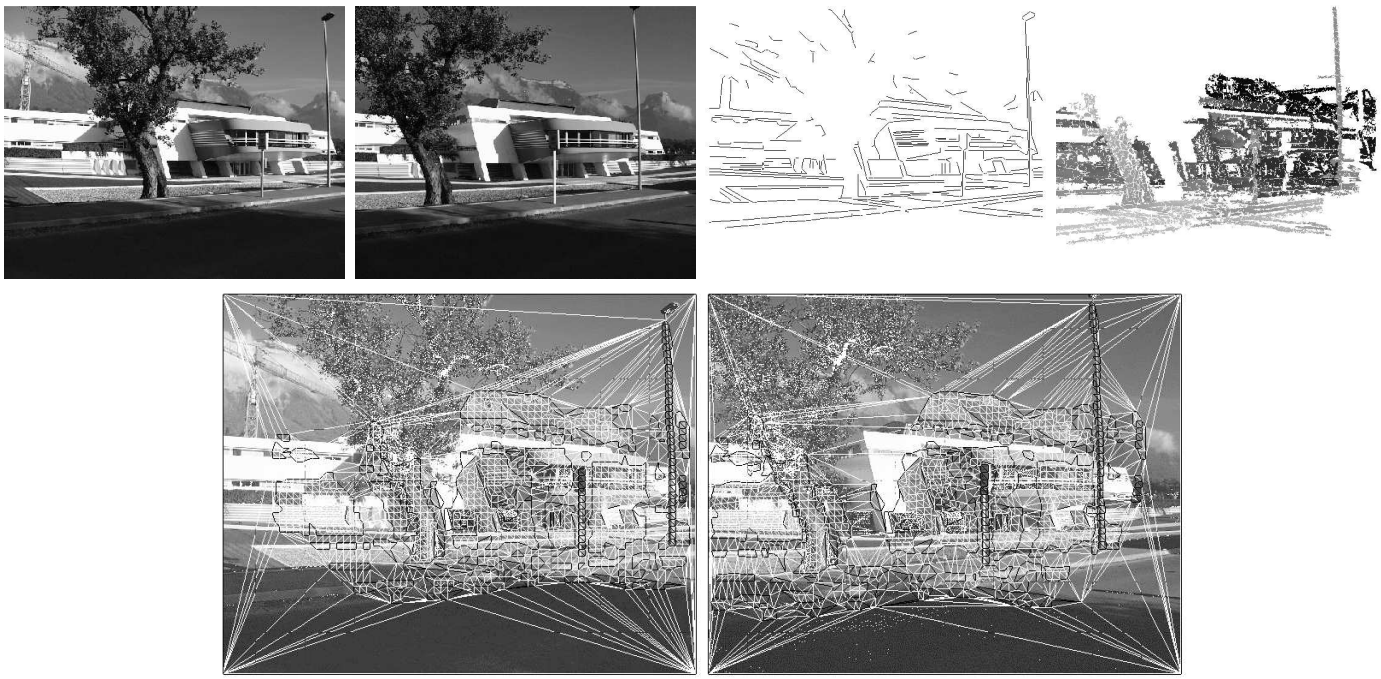

Figure 20: Top left: the original image pair. Top right: the left is the edge image of the first frame and the right is the disparity map. Bottom: the constrained joint view triangulation in the two images.

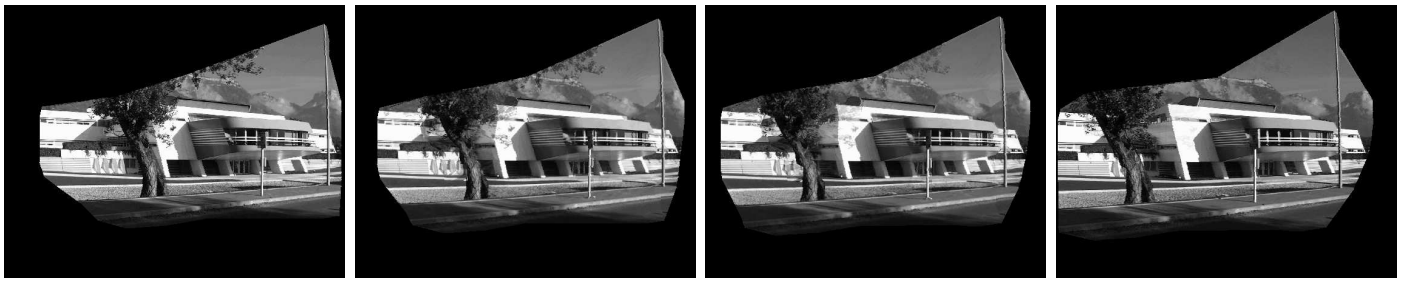

Figure 21: Some samples of the interpolated views at $\lambda = 0, 0.4, 0.6, 1$ from left to right.

Rendering from a triplet of images Though all the key components have been essentially described for pairs of images, the system could be extended to any number of images having a broad coverage of a view field. In Figure 22, we show some rendered images from three face images. Actually, all images within the circle going through the three original images are generated. The actual strategy is to interpolate/extrapolate all possible views from each pair of images, then a weighted blending procedure is applied for the three intermediate images from the three pairs of the images.

The direct construction of a JVT for a triangle of 3 views (or a tetrahedron of 4 views) by simultaneous insertions of edges and patches in the 3 (or 4) images would be more difficult to implement, although the corresponding rendering is expected to be faster than those of our current strategy. The JVT for 5 or more images is not useful in practice, since the set of JVTs may be organized as a 2D (or 3D) triangulation of the given view-points for rendering. Also the continuous transition between different triangles/tetrahedron is straightforward with our current strategy, not with the 3 or 4 view JVTs.

Figure 22: Rendered face images from a triplet of original images. The original three images are the smaller ones and the rendered images are the bigger ones.

Calculation times The calculation times on a Pentium III Mobile 500Mhz are as follows. For the first and twentieth images of the flower garden (360×240), it takes 2.3 seconds to generate the seeds, 1.3 seconds for one propagation and 5 seconds for edge-constrained JVT (2×2150 triangles). For the INRIA building images (562×450) in Figure 21, it takes 6.3 seconds to generate the seeds, 2.5 seconds for one propagation and 10.4 seconds for edge-constrained JVT (2×3000 triangles). For the man in New York images (768×512), it takes 2.6 seconds

to generate the seeds, 4 seconds for one propagation and 17 seconds for edge-constrained JVT (2×9900 triangles).

6 Conclusion and future work

We have presented a complete image-based rendering system. Based on re-sampling of robust quasi-dense matching, we have introduced a joint view triangulation that simultaneously triangulates two images while keeping consistent corresponding relationships. A pseudo painter's rendering algorithm has also been developed to create new images from the JVT structure. These techniques have produced visually convincing image sequences. Future studies include development of more advanced sub-sampling methods to refine the boundaries, integration of panorama views to have a complete rendering system and the simplification of the final JVT for better rendering efficiency.

Acknowledgement

The image sequence of the man in New York was provided by Dayton Taylor. The project was partly supported by RGC-HKUST6188/02E.

References

[1] S. Avidan and A. Shashua. Novel view synthesis in tensor space. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Puerto Rico, USA*, pages 1034–1040, June 1997.

[2] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 777–784, 1995.

[3] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Santa Barbara, California, USA*, pages 434–441, 1998.

[4] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[5] T. Beier and S. Neely. Feature-based image metamorphosis. In SIGGRAPH *1992*, pages 35–42, 1992.

[6] S.E. Chen. Quicktime VR - an image-based approach to virtual environment navigation. In SIGGRAPH *1995, Los Angeles, USA*, pages 29–38, 1995.

[7] S.E. Chen and L. Williams. View interpolation for image synthesis. In SIGGRAPH *1993*, pages 279–288. Computer Graphics, 1993.

[8] T.T. Wong C.W. Fu and P.A. Heng. Triangle-based view interpolation without depth-buffering. *Journal of Graphics Tools*, 3(4):13–31, 1998.

[9] R. Deriche. Using Canny's criteria to derive a recursively implemented optimal edge detector. *International Journal of Computer Vision*, 1(2):167–187, 1987.

[10] U.R. Dhond and J.K. Aggarwal. Structure from stereo – a review. IEEE *Transactions on Systems, Man and Cybernetics*, 19(6):1489–1510, November 1989.

[11] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing*, 24(6):381 – 395, June 1981.

[12] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, November 1991.

[13] S.J. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. The lumigraph. In *Proceedings of* SIGGRAPH*, New Orleans, LA*, pages 43–54, 1996.

[14] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 147–151, 1988.

[15] E. Izquierdo and S. Kruse. Image analysis for 3d modeling, rendering and virtual view generation. *Computer Vision and Image Understanding*, 71(2):231–253, August 1998.

[16] A. Koschan. What is new in computational stereo since 1989 : A survey on stereo papers. Technical report, Department of Computer Science, University of Berlin, August 1993.

[17] S. Laveau and O. Faugeras. 3D scene representation as a collection of images and fundamental matrices. Technical report, INRIA, February 1994.

[18] S. Laveau and O.D. Faugeras. 3D scene representation as a collection of images. In *Proceedings of the 12th International Conference on Pattern Recognition, Jerusalem, Israel*, volume 1, pages 689–691, 1994.

[19] S.Y. Lee, K.Y. Chwa, J. Hahn, and S.Y. Shin. Image morphing using deformation techniques. *The Journal of Visualization and Computer Animation*, 7:3–26, 1996.

[20] J. Lengyel and J. Snyder. Rendering with coherent layers. In *Proceedings of* SIGGRAPH*, Los Angeles, CA*, 1997.

[21] M. Levoy and P. Hanrahan. Light field rendering. In *Proceedings of* SIGGRAPH*, New Orleans, LA*, pages 31–42, 1996.

[22] M. Lhuillier and L. Quan. Image interpolation by joint view triangulation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado, USA*, volume 2, pages 139–145, 1999.

[23] M. Lhuillier and L. Quan. Edge-constrained joint view triangulation for image interpolation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Hilton Head Island, South Carolina, USA*, volume 2, pages 218–224, June 2000.

[24] M. Lhuillier and L. Quan. Robust dense matching using local and global geometric constraints. In *Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain*, volume 1, pages 968–972, 2000. ICPR'2000 Piero Zamperoni Best Student Paper Award.

[25] M. Lhuillier and L. Quan. Match propagation for image-based modeling and rendering. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1140–1146, 2002.

[26] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 1981.

[27] L. McMillan. Computing visibility without depth. Technical Report TR05-047, University of North Carolina, October 1995.

[28] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In SIGGRAPH *1995, Los Angeles, USA*, pages 39–46, 1995.

[29] D.D. Morris and T. Kanade. Image-consistent surface triangulation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Hilton Head Island, South Carolina, USA*, 2000.

[30] P.J. Narayanan, P.W. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. In *Proceedings of the 5th European Conference on Computer Vision, Freiburg, Germany*, pages 3–10, 1998.

[31] S. Peleg and J. Herman. Panoramic mosaics by manifold projection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Puerto Rico, USA*, pages 338–343, 1997.

[32] F. Preparata and M.I. Shamos. *Computational Geometry, An Introduction.* Springer-Verlag, Berlin, Germany, 1985.

[33] D. Rippa. Minimal roughness property of the delaunay triangulation. *Computer Aided Geometric Design*, 7:489–497, 1990.

[34] S.M. Seitz and C.R. Dyer. Physically-valid view synthesis by image interpolation. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 18–25, June 1995.

[35] S.M. Seitz and C.R. Dyer. View morphing. In *Proceedings of* SIGGRAPH*, New Orleans, LA*, pages 21–30, 1996.

[36] A. Shashua. Trilinearity in visual recognition by alignment. In J.O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 479–484. Springer-Verlag, May 1994.

[37] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, pages 593–600, 1994.

[38] H.Y. Shum and L.W. He. Rendering with concentric mosaics. In SIGGRAPH *1999, Los Angeles, USA*, pages 299–306, 1999.

[39] F. Sillion, G. Drettakis, and B. Bodelet. Efficient impostor manipulation for real-time visualization of urban scnenery. In *Eurographics'97, Barcelona, Spain*, 1997.

[40] M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(4):344–358, april 1995.

[41] R. Szeliski and H.-Y. Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of* SIGGRAPH*, Los Angeles, CA*, pages 251–258, 1997.

[42] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[43] P.H.S. Torr and D.W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision*, 24(3):271–300, 1997.

[44] P.H.S. Torr and A. Zisserman. Robust parameterization and computation of the trifocal tensor. In R.B. Fisher and E. Trucco, editors, *Proceedings of the seventh British Machine Vision Conference, Edinburgh, Scotland*, volume 2, pages 655–664. British Machine Vision Association, September 1996.

[45] G. Turk and M. Levoy. Zippered polygon meshes from range images. In SIGGRAPH *1994*, pages 311–318, 1994.

[46] J.Y.A. Wang and E.H. Adelson. Layered representation for motion analysis. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, New York, USA*, pages 361–366, 1993.

[47] A. Watt and M. Watt. *Advanced Animation and Rendering Techniques.* ACM Press, 1992.

[48] Z. Zhang, R. Deriche, O.D. Faugeras, and Q.T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1-2):87–119, 1994. Appeared in October 1995, also INRIA Research Report No.2273, May 1994.

[49] Z. Zhang and Y. Shan. A progressive scheme for stereo matching. In M. Pollefeys et al., editor, *LNCS 2018:3D Structure from Images - SMILE'2000*, Lecture Notes in Computer Science, pages 68–85. Springer-Verlag, 2001.

List of Figures

1 Seeds and propagations for the 1st and 20th images of the flower garden sequence 4
2 Match neighborhood for local propagation 5
3 Two overlapping regular grids for image subdivision 7
4 Merging close vertices of patches 8
5 The "broken-line" artifact 9
6 Main steps of JVT construction 9
7 Detection of a rectilinear object 10
8 Example of the small wood post 11
9 Deduction of the corresponding line segment 12
10 Example for the detection of corresponding line segment 12
11 How to refine the result of the first sampling grid using the second grid 13
12 Comparing JVT with and without constrained edges 14
13 Two kinds of half occluded areas 15
14 Comparing JVT without and with virtual vertices 15
15 Intermediates results of the pseudo-painter's algorithm 18
16 Interpolation of the flower garden image pair 18
17 Rendering of the flower garden image pair for a circular trajectory 18
18 Intermediate image for the man in New York using virtual vertices 19
19 Rendering of the man in New York with and without the virtual vertices 19
20 JVT for a natural scene with man-made objects 20
21 Interpolation for a natural scene with man-made objects 20
22 Rendering using a triplet of images 21