INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Towards Automatic Interpolation for Real and Distant Image Pairs

Maxime Lhuillier

N° 3619

Fevrier 1999

THÈME 3

Rapport de recherche

ISSN 0249-6399

Towards Automatic Interpolation for Real and Distant Image Pairs

Maxime Lhuillier

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet Movi

Rapport de recherche n°3619 — Fevrier 1999 — 32 pages

Abstract: Image-based rendering offers the advantage of being able to provide realistic output and at the same time to avoid the difficult problem of a complete geometric and photometric modeling of the real world. The method described here is able to deal with non rigid scenes and large camera motions.

We present in this report a three step algorithm for the interpolation of two views of a scene, from which we can for instance simulated a camera motion withing the given scene. The first step establishes pixel correspondences between the images and is the most difficult part. We justify the choice of region-growing based dense matching methods and we summarize their principle. Secondly, a robust algorithm converts these pixel correspondences to an adequate structure for the last step: image interpolation. This structure encompasses the transformation between the images using constrained and dependent triangulations in both of them, and handles the half-occluded areas. The implementation of the whole process is outlined and the process is demonstrated on real images.

Key-words: Image Matching, Correlation, Region Growing, Constrained Delaunay Triangulation, Visibility, Interpolation, Morphing, Image-Based Rendering

(Résumé : tsvp)

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN (France)
Téléphone : 04 76 61 52 00 - International: +33 4 76 61 52 00
Télécopie : 04 76 61 52 52 - International: +33 4 76 61 52 52

Vers l'interpolation automatique de paires d'images réelles et distantes

Résumé : Utiliser des images réelles pour rendre des images de synthèse offre l'avantage de contourner le difficile problème d'une modélisation géométrique et photométrique complète du monde réel. Ce rapport entre dans cette catégorie: il propose un algorithme d'interpolation de deux vues d'une même scène permettant ainsi des animations permettant par exemple de simuler des mouvements de caméras. Cet algorithme peut traiter les scènes rigides ou non ainsi que les grand déplacements de caméras. L'algorithme se décompose en trois étapes. La première étape est la plus difficile : il s'agit de définir les correspondances entre les pixels des deux images. Nous justifions le choix d'algorithmes d'appariements denses basés sur la croissance de régions et leur principe est résumé. Pendant la seconde étape, un algorithme robuste convertit ces correspondances en une structure adaptée à la troisième et dernière étape qu'est l'interpolation d'images. Cette structure décrit la transformation entre les deux images à l'aide de triangulations contraintes et dépendantes dans chacune d'elles, en tenant compte des zones partiellement occultées. L'implémentation de l'algorithme complet est décrite et des tests sur des images réelles sont finalement proposés.

Mots-clé : Appariement, Corrélation, Croissance de région, Triangulation de Delaunay Contrainte, Visibilité, Interpolation, Morphing, Rendu Basé Image

1 Introduction

Many methods have been proposed for view synthesis from real views. Their interest is to directly use informations from the real world in order to obtain photo-realistic results. Thus it avoids the difficult problem of a complete geometric and photometric modeling.

The first way to achieve this was proposed by the photogrammetry/vision fields: the goal is to automatically recover accurate 3D models from images. The most difficult part is to obtain enough reliable feature correspondences in the images. City modeling from aerial photos is for instance an important topic of photogrammetry (e.g. [RHM95],[LF96],[HSG96],[KM96]). Ones of the most successful systems in vision are obtained using turn-tables on which the object shape is recovered (e.g. [NB94],[TFZ96], [SD97]) or using reasonable user interactions for outdoor scenes (e.g. [DTM96]). A recent and interactive approach [LDR98] allows to modify the lights in a reconstructed scene taking into account of the shadows.

The second way is mainly proposed by the computer graphics field: in many years were proposed image based rendering techniques like image interpolation (e.g. [Wol90], [BN92], [Che95], [SD96], [LCH96]). It is shown for example that it is sufficient to manually set some salient features correspondences and obtain amazing results, even if there is no real geometric evidence for the geometric morphing proposed. So image interpolation has to be considered as a useful tool for image compression or visual simulation. More detail on related work is going to be presented in section 2.

This is the approach chosen here, and an algorithm to obtain interpolations from two distant views of a real scene is presented in section 3. Before reaching this point, we have to establish correspondence between to images; section 2.1 first justifies the choice of region-growing based dense matching methods we are presenting; The main reason is that this kind of algorithm can reasonably works with larger displacements (e.g a quarter of image size) than many others, and that it can work with or without the rigidity constraint. Secondly, a robust algorithm converts these pixel correspondences into an adequate structure for the last and view interpolation step. This structure encompasses the transformation between the images using constrained and dependent triangulations in both of them. In particular, half occluded areas are explicitly represented and unmatched areas (e.g. low textured ones) are filled. Third, we describe the view interpolation step. Like other morphing techniques, it uses heuristics to combine shape interpolation and texture blending.

2 Related Works

We shortly review related works for each of the three steps of our algorithm (matching, triangulations and rendering) and discuss their links with our method.

2.1 Matching

Matching is one of the major research area in the computer vision community. Its applications range from object recognition to 3D perception, and in areas ranging from robotics to video indexing. The broad categories of algorithms are identified upon matching primitives (sparse primitives like interest points and edges) or dense (all pixels) and the assumption of a rigid scene or of non rigid scene.

Using sparse or dense primitives for view synthesis is discuted by [BM97]. It is possible to recover a complete surface of an object using a sparse set of matched and then reconstructed points. Under the assumption that the matched points lies on planes, it is sufficient to compute a 2D Delaunay triangulation in one of the reference views with matches as vertices and obtain a surface by back-projection. Convincing results are obtained unless a single bad match occurs: in such a case the whole texture of neighboring triangles is wildly stretched when the viewer goes away from the reference view point. Further it does not model half-occluded areas which are important for large displacements of the camera. However, some manually but well selected feature matches like edges are sufficient to obtain dramatic image metamorphosis [BN92] if we combine generalized image warping and cross-dissolve between images elements [Wol90].

Thus for automatic methods, most people compute dense displacement mappings. For stereo images [DA89], [Kos93] whose epipolar geometry is known *a priori*, the search space can be reduced to a 1D along epipolar lines. The main problems are repetitive patterns, texture-less areas, depth discontinuity and half-occluded regions.

Another approach is optical flow estimation[BFB94], which handles non-rigid scenes but assuming instaneous small displacements. Hierarchical methods seems to be necessary to treat large displacement range. However, coarse to fine strategies might miss some texture details and fail at depth discontinuities. Especially in the non rigid case, the dense matching is ill-posed due to the aperture problem: the local displacement can only be recovered in the direction of the intensity gradient. Thus additional smoothness assumptions are needed usually to obtain displacements. Only a few algorithms are able to establish correspondences across images of two different scenes (e.g. to obtain morphings between different faces [Que97], [SK98]).

The requirements on a stereo algorithm for view synthesis are discussed by [Sch96a]. Although uniform areas are difficult to match, they usually do not create visual artifact if their boundaries are matched correctly. It is sufficient to interpolate the matching. Further, [SD95],[SD96] show that the resulting rendering is correct under the additional assumption of monotonicity along conjugated epipolar lines. Next we have to deal with partial occlusions: The best heuristic to assign displacements to such regions seems to assume a constant depth for the background [Sch96a].

If we are not interested by matching untextured areas, why not use an algorithm which match only reliable (i.e textured) areas but without strength constraints like epipolar constraint and displacement bound ? One can interpolate the other ones with a triangulation in the images. Region growing matchings are reasonably effective even without such constraints.

Region growing is a classic approach for segmentation [HS85],[Mon87]. In its simplest sense, region growing is the process of merging neighboring pixels (or collections of pixels) into larger regions based on homogeneity properties (cf. [HS85]). Region growing dense matchings are greedy algorithms and therefore they are simple and efficient: at each step, a new match is picked from the set of current matches and is used to detect other matches in its neighborhood [OC89], [Lhu98]. The growth of the matched area stops when its borders are stopped by image borders, occluded regions or untextured areas. A patch to patch region growing matching is designed by [OC89] to match two SPOT satellite images. Interesting results are obtained even for large displacements, little windows and without epipolar constraints for outdoor scenes [Lhu98]. Oddly, region growing matchings seems not to be commonly used.

2.2 Triangulations

Although graphic hardware needs usually triangulations to produce fast renderings, some authors display their dense displacement mappings using pixel by pixel rendering (e.g. [SD96], [Sch96a], [BM97]). The main advantage of this rendering is that any false match will be lost in the crowd and become unnoticeable [BM97]. The advantages to represent the displacement mapping with triangulation are: a reduced complexity and memory, explicit topological relations and easy interpolation for unmatched gaps.

In computer vision, triangulations are often associated with the problem of surface reconstruction. For such a case, the solutions obtained could provide a solution to our problem: rendering-oriented structure can be deduced from a reconstructed surface simply by projecting the surface triangulation into the views. However, it does not handle non rigid scenes for which no 3D information can be obtained, nor complex scenes (trees with leaves for instance) where the 3D reconstruction can never be achieved.

One class of surface algorithm generates triangulations from dense range images. For instance, [GSB97] present a fast adaptative triangulation. An intermediate adaptative quadrilateral mesh is generated from the depth curvature, then the diagonal edges which best agree with the depth gradients and discontinuities are chosen. Another method [Koc95] segments the range image using a surface orientation histogram and then spans each recovered smooth piece with independent triangulations. The discontinuities are thus preserved.

A second class of surface algorithms perturbs an initial surface so as to minimize the matching error. A robust method [FL94] combines diverse sources of information for the deformation: stereo and shape from shading data, 3D features and 2D silhouettes. A recent approach [FK98] guides a topology-variable surface using level set methods.

Our approach is closer to the adaptative triangulation method: A dense displacement is converted to a Joint View Triangulation (JVT). However there are two differences: The matching is validated or invalidated during the conversion; and we generate triangulations that correctly treat the half occluded regions in the two views.

A structure similar to the JVT was suggested as "future work" by [SDB97] in the computer graphics field. To obtain a real-time visualization of a complex urban scene, they represent nearly objects as classical 3D models and distant scenery as 'impostors'. An impostor is a pre-calculated view of a model projected onto a transparent polygon, which is drawn instead of the model to accelerate the display process. The suggested problem was to generate a structure to obtain smooth transitions between two improved impostors.

Other structures have been proposed to model visibility information in computer vision and computer graphics (e.g. aspect graphs [GCS91] and visibility skeleton [DDP97]), but they need a 3D model as input and are not designed for the same uses. In contrast to these, our structure is directly constructed from a displacement map of a (possibly non rigid) scene.

2.3 Rendering the Displacement Mapping

We distinguish geometry-based and heuristic renderings for a displacement mapping. Many solutions have been proposed in all cases.

If the scene is rigid, one can implicitly or explicitly recover the geometry of the scene. From weakly calibrated images, [LF94] generate new views applying a ray-tracing like algorithm in the displacement mapping along epipolar lines and [AS97] use the trifocal tensor. [SD95] and [Sch96a] add a prewarping rectification step and a post warping step to a linear interpolation of a complete displacement mapping. Hence the interpolation result coincides with a real camera movement between the two reference views. In particular, unnatural distortions of simple linear interpolation are avoided. Otherwise, the algorithms use strongly calibrated images. All corresponding points to the displacement are reconstructed and directly projected, or are first fitted by piece planes and next projected. This classical approach is however more fragile than the previous one, because the estimation of camera parameters is a fragile process, and it lacks of generality as cameras cannot always be calibrated.

The advantage of the heuristic renderings is their applicability for deformable objects (non rigid scenes or different objects like faces). Because of this generality, features are difficult to match automatically. Usually, a sparse set of salient matches are manually entered and then a displacement mapping is interpolated (e.g. [BN92], [LCH96]). The main drawback is that the allowed points of view are limited between the two initial ones.

We finally note that renderings are not limited to classic photos: one can merge them to obtain planar [Sze96] or cylindric [Che95], [MB95] mosaics and then interpolate them. An other way for rendering consists to organize the image contents using layers [BSA98], but it seems to be difficult to obtain a complete and automatic process.

3 Principle

We present in this section the principles of our three step method.

3.1 Overview

The first step (see Section 3.2) establishes dense correspondences for pixels between two images. Although uniform areas are difficult to match, they usually do not create visual artifact during image interpolation if their boundaries are matched correctly. It greatly simplifies our synthesis problem, because it suffices to match only the textured areas and interpolate the matching in surrounded and more uniform ones. We usually obtain surprisingly good results by applying unconstrained matching (no epipolar constraint or displacement bound) like correlation-based region growing matchings [Lhu98]. A best first matching strategy drastically limits the possibility of bad matches.

In the second step (see Section 3.3), a robust algorithm converts this unrefined matching to a rendering-oriented representation called *Joint View Triangulation* (JVT) which models visibility informations. It provides

- An image-based representation with a reduced set of primitives, which approximates the displacement map.
- For each view, a separation of matched and half occluded areas to allow different processes on them.
- A correspondence between primitives which represents the common (i.e. matched) areas of the pair. This ensures the global coherence of the data and allows non redundant processing during use.

This structure is finally used in the last step (see Section 3.4) to generate image interpolations. Like other morphing techniques, it uses heuristics to combine shape interpolation and texture blending. We tackle with the lack of depth information by drawing all triangles in both images in a painter-like [JDF91] back-to-front order.

3.2 Region Growing Matching

The matching has two main steps. The first step extracts and matches a sparse set of highly distinctive features like points of interest. The second step uses these initial matches to seed concurrent dense matching propagations, using a best first matching strategy. This extends the matches to include the textured areas of the images. The resulting matching is sufficient for image interpolation.

We use the ZNCC correlation measure (zero-mean normalized cross-correlation) in both steps for matching. The $ZNCC_k(ij, lm)$ measure for $(2k+1) \times (2k+1)$ windows centered at pixels (i,j) and (l,m) is

$$ZNCC_k(ij, lm) = \frac{\sum_{-k \leq dx, dy \leq k} (l_{i+dx, j+dy} - \bar{l}_{ij})(l_{l+dx, m+dy} - \bar{l}_{lm})}{\sqrt{\sum_{-k \leq dx, dy \leq k} (l_{i+dx, j+dy} - \bar{l}_{ij})^2} \sqrt{\sum_{-k \leq dx, dy \leq k} (l_{l+dx, m+dy} - \bar{l}_{lm})^2}}$$

where l_{ij} is the luminancy of pixel (i,j) and $\bar{l}_{ij} = \frac{1}{(2k+1)^2} \sum_{-k \leq dx, dy \leq k} l_{i+dx, j+dy}$. It is invariant to linear luminancy changes (i.e. changes of the form $l \leftarrow al + b$). However the use of such windows allows only image translation. Different approches exists which would allow important rotation changes, for instance using invariants like [Sch96b].

3.2.1 Seed Selection Step

Like many applications in computer vision, the first step of our method extracts and match interest points. We use a slightly modified version [SMB98] of Harris points [HS88], because of their good repeatability and informational content. They are the points in images which reach the local maxima of isotropic texture variation. Other seed features are possibles (e.g. regions [Lhu98]). The points are matched across the two images using $ZNCC_5$ and are the initial contain for the set of seed matches.

3.2.2 Propagation Step

All seed matches are starting points of simultaneous and concurrent propagations in the second step. At each step the match with the best $ZNCC_2$ is removed from the current set of seed matches. Then we look for new matches in its neighborhood and add them to the set of current seeds and to the set of accepted matches (the displacement mapping to build). The process terminates when the seed set is empty. Notice that the risk of bad propagation is greatly reduced by the choice of the best match at each time. For instance, a single good and initial seed match is sufficient to provoke an avalanche of correct matches. Thanks to the concurrent strategy between the initial matches, the bads are discarded by small or empty propagations. This makes our algorithm much less vulnerable than others [TZ96], [ZDF94] which try to match a maximum of interest points during the seed selection step. More details are given in Section 6.

3.3 Joint View Triangulation

This section explains how to convert robustly an unrefined matching between two images to a JVT (see figure 1). A precise definition is given in the following subsection.

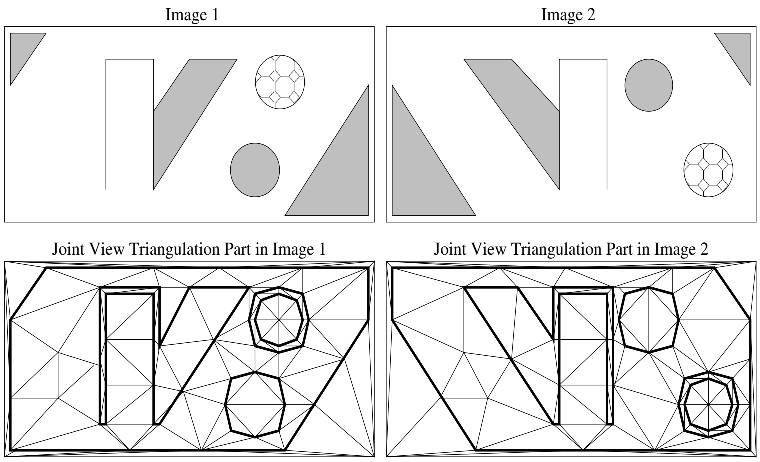


Figure 1: The first row represents two views of a non rigid scene, composed of a small vertical rectangle on an infinite horizontal plane and a falling ball. Half occluded areas (visible in only one image) are shaded gray. The second row shows a joint view triangulation for these two views. *Matched* (resp. *Unmatched*) *triangles* fill the matched (resp. unmatched) areas. The black edges represent the boundaries of matched areas and are forced to be edges of their respective triangulations.

3.3.1 What is a JVT?

As a JVT provides

1. An image-based representation with a reduced set of primitives;
2. For each view, a separation of matched and half occluded areas;
3. A correspondence between primitives which represents the matched areas in the image pair.

The joint view triangulation for two views is a pair of inter related image triangulations, one for each image, based on an underlying locally dense displacement map. The Delaunay triangulation is chosen because of its minimal roughness property [Rip90]. Triangulating in image space allows non rigid scenes to be handled.

We call *matched triangle* (resp. *unmatched triangle*) a triangle which covers a region of matched (resp. unmatched) pixels in its image. Matched and unmatched triangles are separated by constrained edges. If we assume that the displacement map is such that half occluded areas coincide with unmatched ones (ideal matching case), the condition 2 is satisfied. The *contours* are the sets of constrained edges which bound the sets of matched triangles in each image. We impose finally a one to one correspondence between each vertex and between each edges of the contour of different images to satisfy the condition 3.

3.3.2 Construction

This section describes a robust algorithm for JVT (more details are given in Section 7).

Because of noisy and/or bad matches, some caution is needed to convert the resulting and unrefined displacement map to a JVT. First we use the piecewise smooth assumption to regularise the dense matching by locally fitting planar patches and obtain a set of matched image patches. Second a JVT is defined by successive merges of the matched image patches in the two images simultaneously.

The first image is partitioned into a regular set of independent square patches, and for each one, we try to fit a matched patch in the second image. We represent the distortions with homographies or affine transformations, which are estimated using a RANSAC-like [FB81] procedure to inner matches (see Figure 2). If one is found, the relative coherence of these matches is checked.

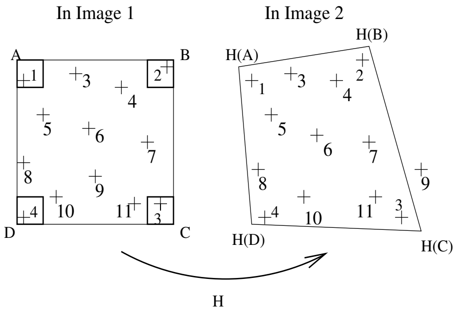


Figure 2: Points A,B,C, and D define a square patch P_1 in image 1. A sparse subset of the dense matching within is labeled from 1 to 11. The matches 1,2,3 and 4 (selected by a RANSAC trial) are respectively in the small framed 3×3 neighborhood of vertices A,B,C and D. They are used to accurately define a planar homography H, which maps the square patch P_1 in image 1 to the distorted square patch P_2 defined by transformed points $H(A), H(B), H(C)$ and $H(D)$ in image 2. All matches (except 9) are compatible with H.

However the image patches constructed in the second image are not exactly adjacent (see Figure 3). We next perturb the vertex locations in the second image to eliminate the small discontinuities and overlaps between patches. We do this by merging any of the four vertices (a, b, c and d) which are within a distance $s_{connect}$ of at least one other vertex, by averaging all vertices within this connected component. Note that this step improves but can not solve all cases of intersecting patches. The same result is obtained if a vertex does not exist or if it exists and is without the distance $s_{connect}$ of all others.

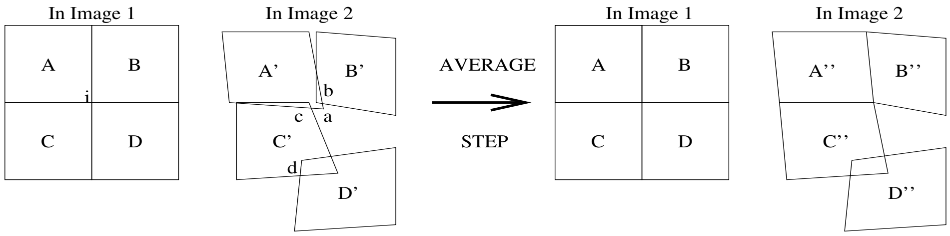


Figure 3: A,B,C and D are four patches of image 1 and A',B',C' and D' are their corresponding patches in image 2. Some patch vertices are forced to coincide if they are enough close from each others. A", B", C", and D" are the result of this averaging step. Note that it improves but can not solve all cases of intersecting patches (e.g. C" with D").

A JVT is then constructed by successive merges of the previous matched patches (see Figure 4). It starts from two unmatched triangles in each images. Each matched patch is merged to the current sets of matched triangles in both images, if its boundary edges do no intersect the current contour (the polygonal boundary of the set of matched triangles) at a single point in one of the images.

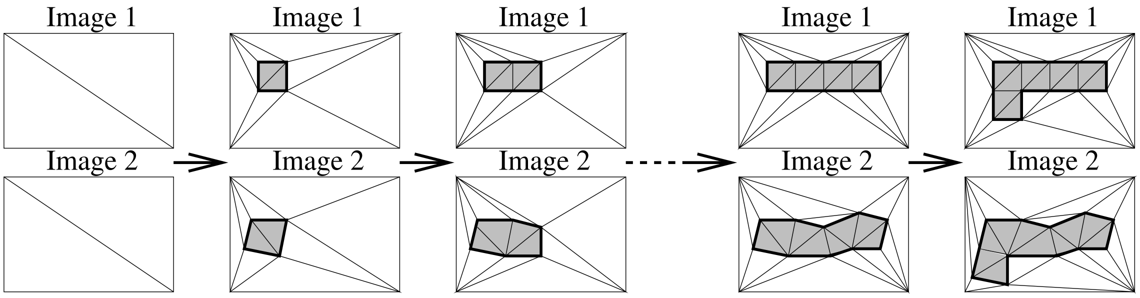


Figure 4: Gray (resp. white) triangles are matched (resp. unmatched) triangles. Black forced edges are constrained and form the contours. The sets of matched triangles grow simultaneously in the two images in a coherent way. Three insertions of patch matches of a complete merging step are shown.

This incremental scheme allows to merge robustly a non-intersecting subset of matched patches while maintaining the coherence between the two views (i.e. a one to one correspondence between all vertices and contour edges in the two images). It works row by row from the top to the bottom of the grid in the first image. The structure is improved in a last step by adding to the set of matched triangles each unmatched triangle for which we can fit an affin transformation using inner pixel matches. This improvement modifies the JVT by swapping constrained-unconstrained status on existing edges.

3.4 Image Interpolation

The two previous steps build a joint view triangulation from two images. We now describe how to use it to generate intermediate images. More details are given in Section 8.

A JVT provides a correspondence between all vertices of different images. Then all triangles are drawn using linear interpolation of their vertices, except those which have an image corner as vertex.

A modified version of the painters algorithm [JDF91] is used to deal with variable depth components of the scene. The classic painters algorithm consists of drawing all triangles in a back-to-front order. Thus, the foreground is drawn over the previously drawn background.

In the absence of any depth information, a heuristic warping order for each triangle of the JVT is deduced from its properties and the displacement of vertices. Unmatched triangles are drawn first because they contain occluded areas.

Secondly we choose to draw all matched triangles in increasing order of their vertex displacements, assuming that the fastest vertices are the closest to the viewer. Note that this drawing order is exact if the camera motion is a pure translation and the scene is rigid.

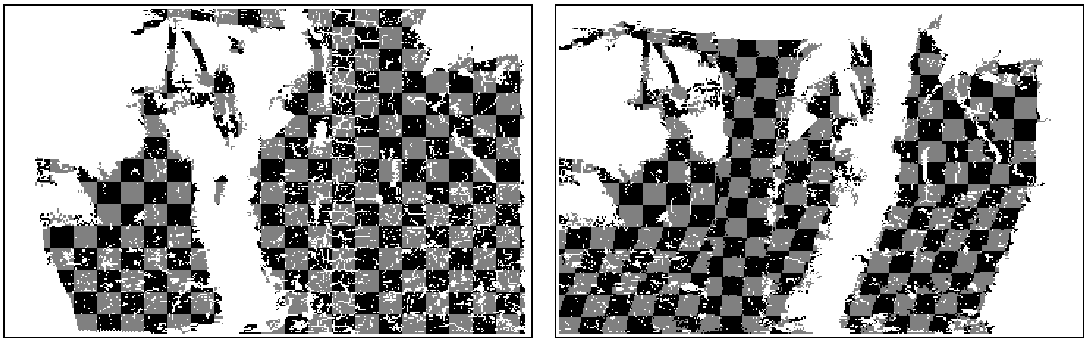
4 Results

Our method has been demonstrated on many real image pairs (see the Mpeg sequences at our web site http://www.inrialpes.fr/movi/people/Lhuillie/demo.html).

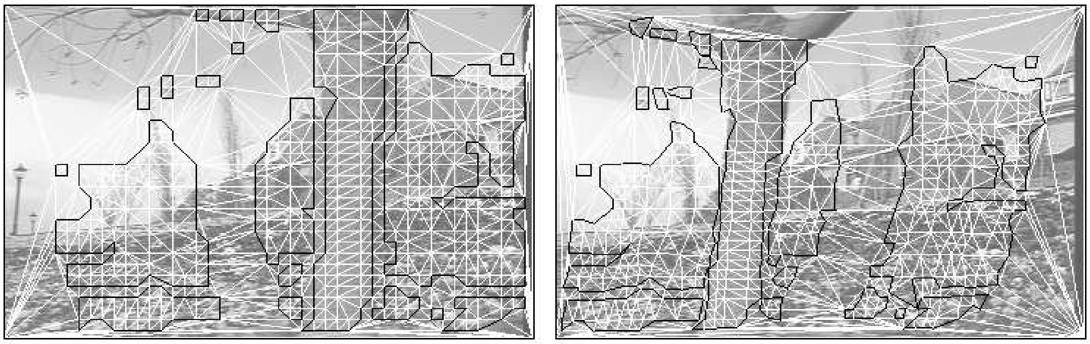
4.1 Details about a Complete Exemple

Summary: Figure 5 shows the intermediates results of the matching and JVT steps for one image pair in the *flower garden sequence*. First a sparse and initial set of Harris points are matched with correlation and winner take all (Figure 5.a). These seed matches are used in a correlation-based region growing step which propagates the matches in textured regions of the images (Figure 5.b). We have used the pixel to pixel region growing matching described in Section 6. Third the resulting pixelic dense matching is converted to a JVT (Figure 5.c) by fitting and merging planar patches in the image space. Finally all triangles of the JVT are warped by linear interpolation of their matched vertices to generate intermediates images (Figure 6). It should be stressed that we have not used or recovered the epipolar constraint. The algorithm runs quickly, for instance, these 360×240 images are matched in 4s (1s for seeds and 3s for propagation) and the joint view triangulation is constructed in 3s on a Ultra Sparc 300Mhz. In practice, we fit two sizes of square patches (16×16 and 8×8) to obtain more matched patches and accelerate the process.

(a)



(b)



(c)

Figure 5: Flower garden image pair and initial sparse matches (a), (non rigid) region growing dense matching (b), the resulting joint view triangulation (c). Matched pixels of the left image (b) are colored with a gray-black checker-board, and the corresponding pixels of the right image are colored with the same color. This makes it easy to visualize the match of each square and its distortion. Black edges (c) are constrained and form the final contours of matched regions. All vertices and the contour edges are matched in the two views. White edges are Delaunay edges and are not necessarily matched.

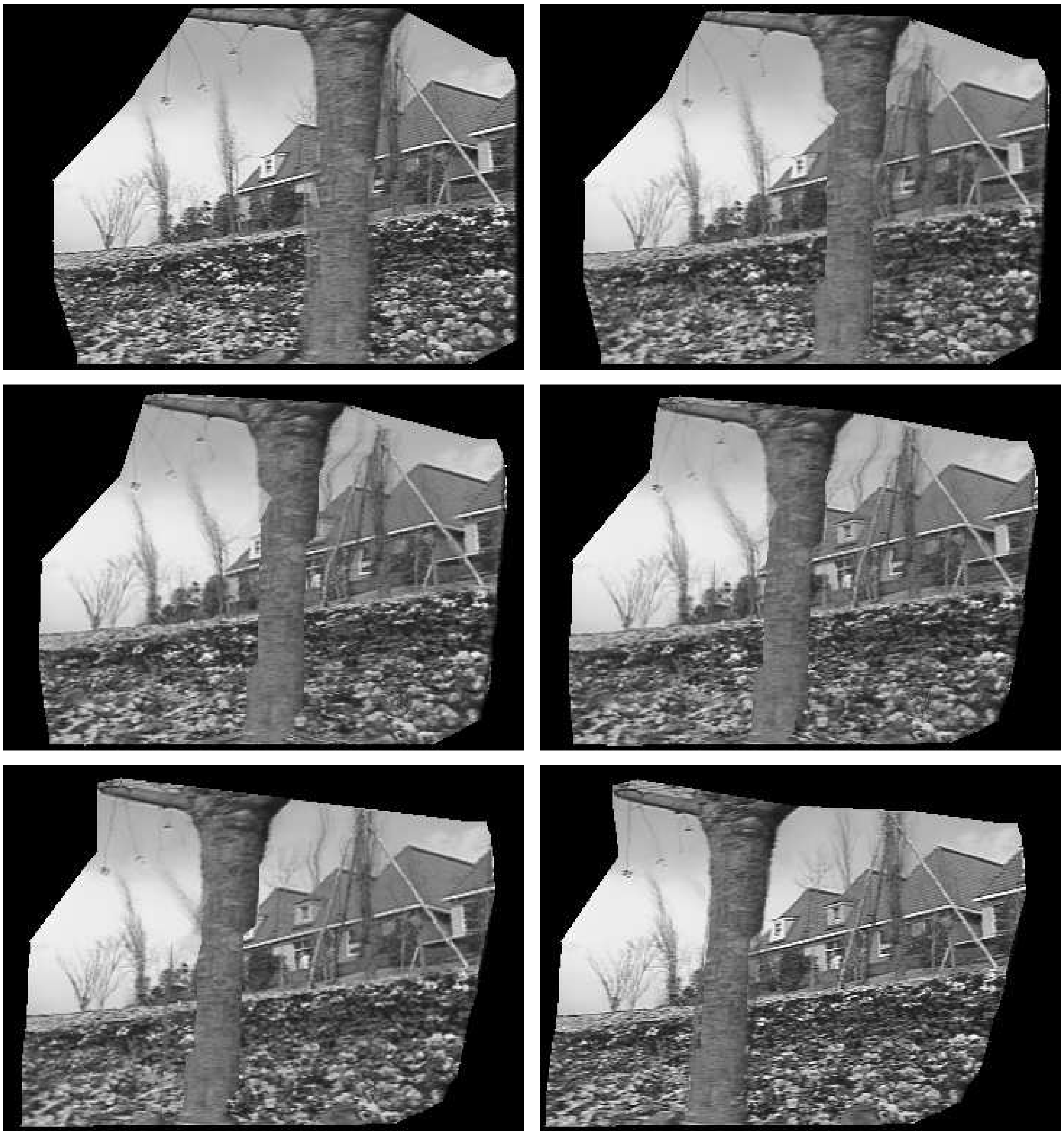


Figure 6: Some sample images of the interpolation: $\lambda = 0, 0.2, 0.4, 0.6, 0.8, 1$ from left to right and up to down (see this movie and many others at our web site http://www.inrialpes.fr/movi/people/Lhuillie/demo.html).

Unmatched areas: This first example is difficult to match and to morph: it exhibits large unmatched areas like the untextured sky and half occluded areas behind the trunk. Some heuristics criteria to distinguish them could be envisaged to improve the JVT but nothing such was done here. It does not affect the JVT coherence (i.e. one to one correspondences between vertices and between contour edges).

Precision: The global precision of our rendering structure is fixed by the patch size: roughly 8 pixels. Patch limitations (size and regular location in the left image) of our implementation and imprecise stops of propagation at occlusion contours are the causes of imprecise approximation and display of the trunk borders. The morphing result suffers of the lack of precision because the matching ordering constraint is violated. If patch size is too large, the approximation is too coarse. On the other hand, the patch fitting is unstable if it is to small. A more detailed discussion about patch size is given in subsection 7.5.

Stability: To illustrate the stability of our method, we propose a second experiment (see Figure 7) on the same image pair. We manually set four seed point matches with 1-2 pixel accuracy

1. in a bush at the left of the images.
2. in the flower at the right of the images
3. near the center of the trunk
4. at a house window near the center of the images

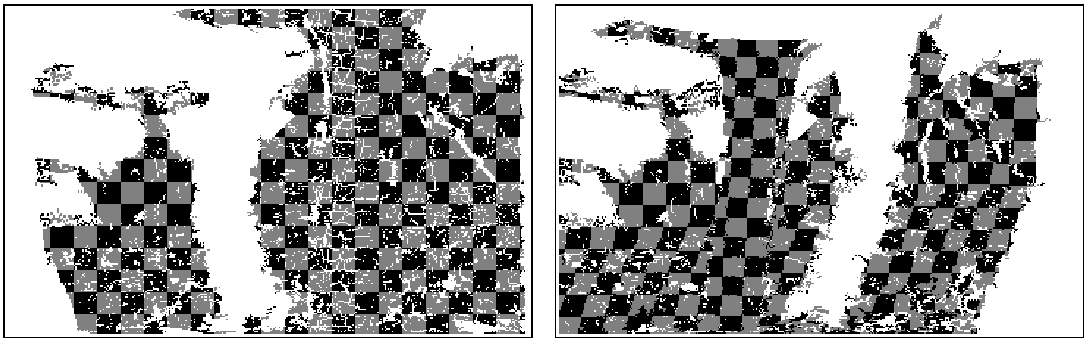
and use them as seed matches for a propagation. Each one is sufficient to provoke an avalanche of correct matches in each of the four isolated and textured region. The resulting JVT is nearly the same as in the first experiment and we have not observed noticeable blunders for the morphing.

4.2 Others examples

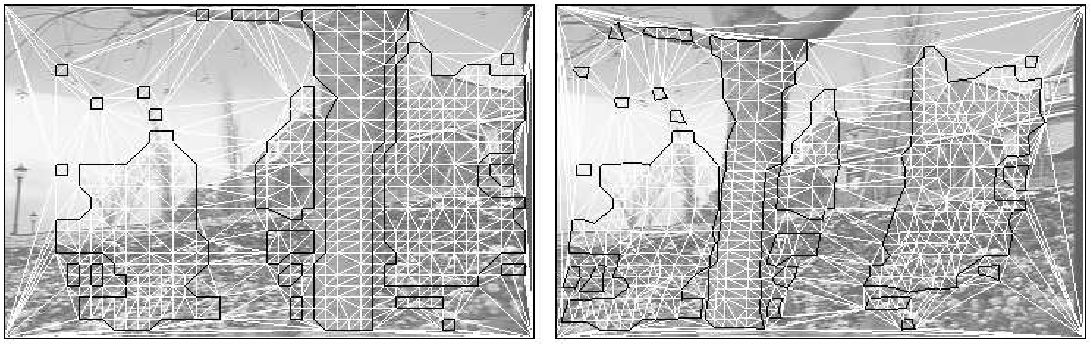
The following examples are obtained in the same conditions: pixel to pixel region growing, same parameters and no epipolar constraint.

The house image pair is shown in Figure 8. The matching is difficult in the fine and repetitive texture of the grass below the house, in low textured areas in the foggy background and in the bushes at the image bottom due to texture sampling and the lack of seed matches. Especially in the grass and bushes cases, the main goals of the JVT is the rejection/acceptance of matching and the interpolation of enclosed and unmatched areas. The morphing step is easier than the previous scene thanks to that the ordering constraint of the matches is preserved. These 768×512 images are matched in 28s (6s for seeds, 22s for propagation) and the JVT is constructed in 18s. Propagation can be done in 14s using small 3×3 windows, but the foggy background and a part of the grass can not be correctly matched.

(a)



(b)



(c)

Figure 7: Flower garden image pair and four manual seed matches (a), the resulting (non rigid) region growing dense matching (b) and joint view triangulation (c).

The street image pair is shown in Figure 9. These 768×512 images are matched in 24s (6s for seeds, 18s for propagation) and the JVT is constructed in 15s. Like in the previous grass case, only a sparse set of small matched areas are retained by the JVT in the fine texture of the road. The JVT fills the gaps at the morphing step. However bad matches are accepted by JVT near the right and top corner of the left image: the first in the repetitive texture of the roof, the second at the corner of the house window. The corner window match generates the main blunder while morphing; such an artifact is easily removed using the epipolar constraint in the propagation step after its recovery in the seed point matches step. The morphing result is shown in Figure 9 without the use of epipolar constraint.

5 Conclusion

A new method is proposed for the automatic image interpolation of two views, which can deals with non rigid scene and large camera motions. First, we present a greedy region growing based dense matching algorithm to obtain the displacement mapping in enough textured areas of the images. Second, this displacement is converted into a rendering-oriented representation which we call Joint View Triangulation. This conversion checks the matching coherence using local geometric constraints. A JVT models visibility information by separating matched and unmatched areas, and by matching primitives of common areas between the two images. Finally, our structure is used in the morphing step.

This approach improves existing ones in different ways: it does not assume rigidity of the observed scenes, even if rigidity constraint can improve the result quality. It provides an new image mesh structure which handles the matched and unmatched region. This structure allows simple standard display algorithms to run in realtime for animation purpose like camera motion simulation.

There still exists a number of ways to improve our method. First, the JVT accuracy is limited by the fixed patch size and locations of our implementation. We have seen that the morphing quality suffers of this lack of accuracy at the matched areas boundaries when the matching ordering constraint is violated (e.g. the trunk). In this case, a boundary refinement is necessary. Variable size patches would be a good way to represent matched regions like thin objects (e.g. electric-post) or large and untextured regions (e.g. manufactured object parts) with more precise borders.

Secondly, some unmatched gaps of the dense matching are filled using JVT by interpolation. These gaps are due to the untextured areas and matching errors, and the problem is to distinguish them from half occluded areas. We could detect them using heuristic criterions based on size, distortion between images, topological constraints and texture information. However careful experimentation has to be performed in order to see what real improvements these methods are bringing in. Finally JVT should also be used for surface reconstruction and this work should generalized for an arbitrary number of views. We are planning to work on these problems in a near future.

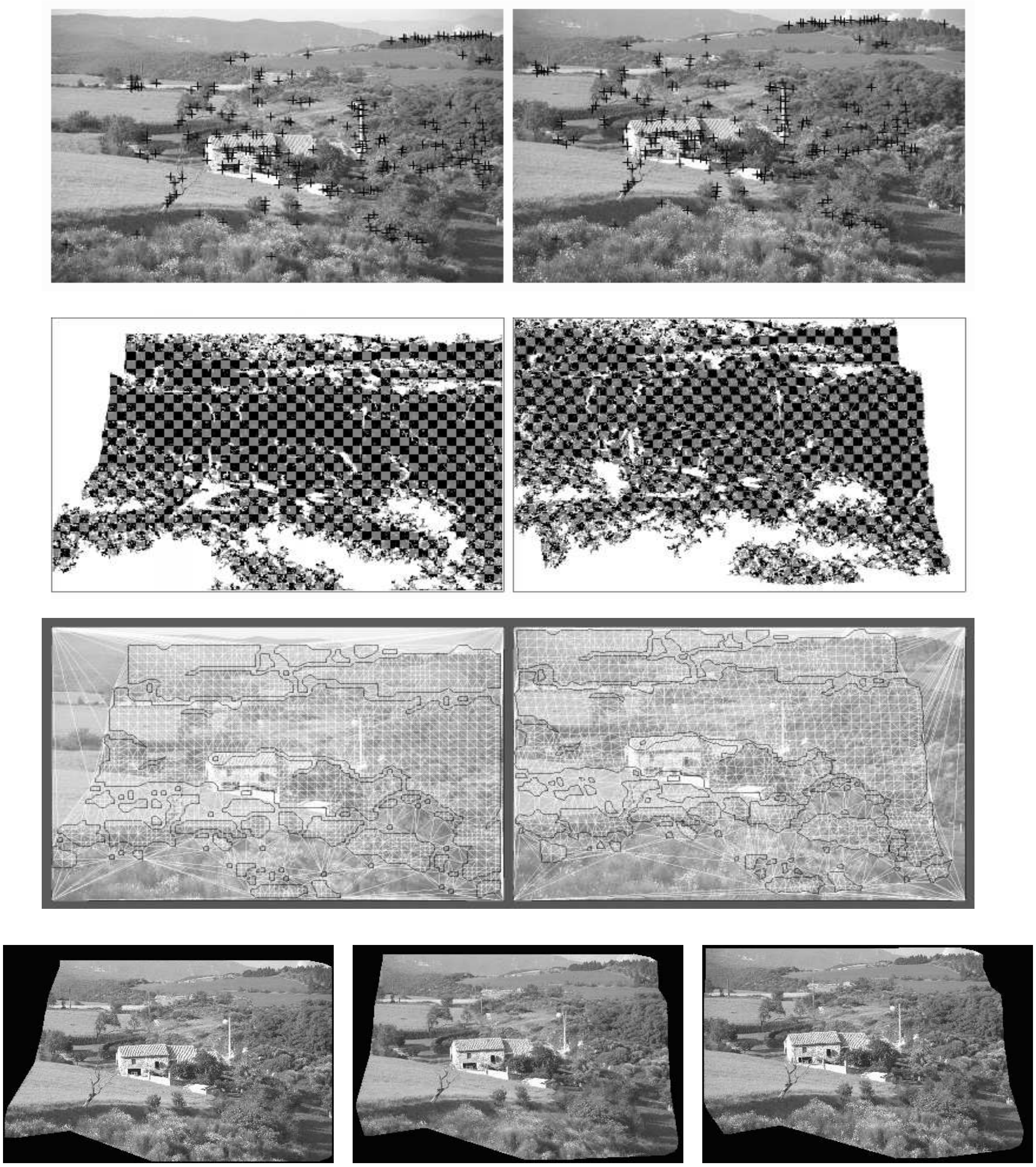


Figure 8: Seed matches, non rigid propagation, joint view triangulation and sample images of the interpolation $\lambda = 0, 0.5, 1$ for the house image pair.

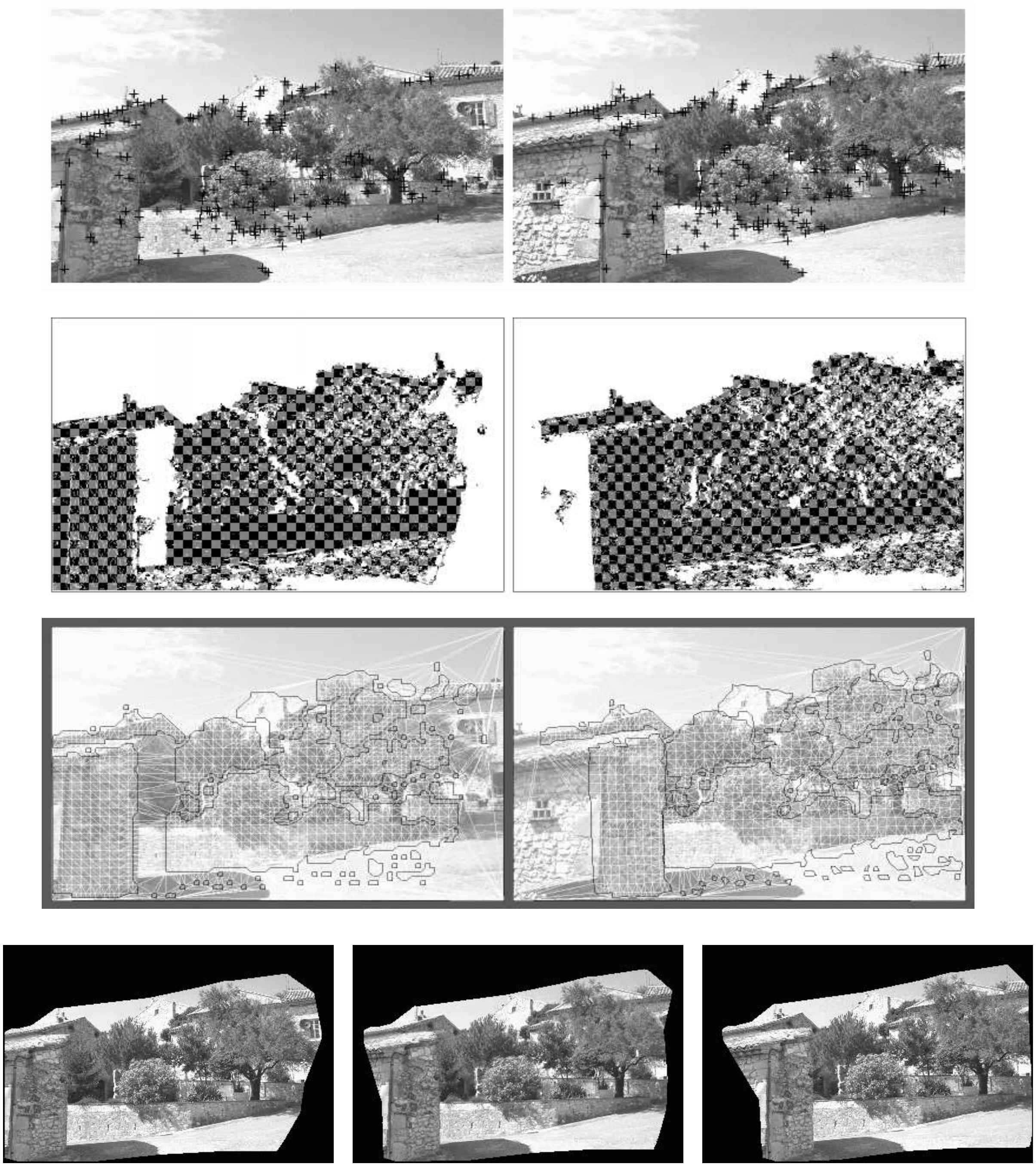


Figure 9: Seed matches, non rigid propagation, joint view triangulation and sample images of the interpolation $\lambda = 0, 0.5, 1$ for the street image pair.

6 Annex A: Pixel to Pixel Region Growing Matching

6.1 Principle

A displacement mapping Map stores the set of correct pixel matches. The algorithm consists of growing this set. Let $Seed$ be the set of current seeds matches near its boundaries. At each step we remove the match (a, A) from $Seed$ with the best correlation score. Match (a, A) is the seed for a local propagation: new matches in the neighborhood of (a, A) (see Figure 10) are added simultaneously to set $Seed$ and map Map. These new matches (b, B) are added only if neither pixel b nor B are already stored in Map.

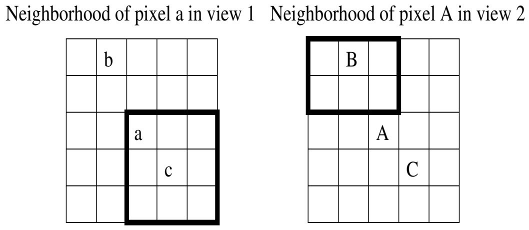


Figure 10: Definition of a match neighborhood. The neighborhood $\mathcal{N}(a, A)$ of a match (a,A) is a set of matches included in the two 5×5-neighborhood $\mathcal{N}_5(a)$ and $\mathcal{N}_5(A)$ of a and A. Possible matches for b (resp. C) are in the 3×3 black frame centered at B (resp. c). The complete definition of $\mathcal{N}(a, A)$ is

$$\{(b, B), b \in \mathcal{N}_5(a), B \in \mathcal{N}_5(A), (B - A) - (b - a) \in \{-1, 0, 1\} \times \{-1, 0, 1\}\}.$$

6.2 Algorithm

The result is a displacement mapping Map, which is maintained injective. We use a heap [AHU74] for the set $Seed$ of the potential seeds for local propagations and to select the best at each step. The complexity of propagation is then $O(nlog(n))$, where n is the number of matched pixels in the image at the end of the process. Notice that it is output sensitive (i.e. it is dependent only from the number of resulting matches, not from all pixels) and independent of any disparity bound.

Let $s(a)$ be some estimate of the luminancy roughness for pixel a and s_{min} be a lower threshold. We use $s()$ to forbid propagation into insufficiently textured areas $\{a, s(a) < s_{min}\}$.

Let $r(a, b)$ be a measure of reliability for pixel match (a, b) and r_{min} be a lower threshold. Matches with the best reliability are first considered for propagation.

Let l_{ij} be the luminance of pixel (i, j). The following definitions slightly differ from those presented in our previous work [Lhu98]:

$$
\begin{aligned}
s(ij) &= max\{|l_{i+dx,j+dy} - l_{ij}|, (dx,dy) \in \{(1,0),(-1,0),(0,1),(0,-1)\}\} \\
s_{min} &= 0.01 \\
ZNCC_k(ij,lm) &= \frac{\sum_{-k \le dx,dy \le k}(l_{i+dx,j+dy} - \bar{l}_{ij})(l_{l+dx,m+dy} - \bar{l}_{lm})}{\sqrt{\sum_{-k \le dx,dy \le k}(l_{i+dx,j+dy} - \bar{l}_{ij})^2}\sqrt{\sum_{-k \le dx,dy \le k}(l_{l+dx,m+dy} - \bar{l}_{lm})^2}} \\
r(ij,kl) &= ZNCC_2(ij,kl) \\
r_{min} &= 0.5
\end{aligned}
$$

The algorithm is

// ***** First, initialize the set $Seed$ with seed feature matches *****
Detect Harris points and match them using $ZNCC_5$ correlation and winner takes all.
Initialize $Seed$ with all these seed point matches.
// ******* Next, propagate *******
While $Seed \neq \emptyset$ do
. Pull from $Seed$ the match (a,b) which maximizes reliability $r(a,b)$
. Let $Local$ be a empty heap of pixel matches
. // **** Store in $Local$ the potential matches of local propagation from match (a,b) ****
. For each (c,d) in $\mathcal{N}(a,b)$ (see Figure 10) do
. . If c and d are not already matched and $s(c) > s_{min}$ and $s(d) > s_{min}$ and $r(c,d) > r_{min}$
. . Then store match (c,d) in the heap $Local$
. . End if.
. End for.
. // **** Store in $Seed$ and Map consistent matches of $Local$ with Map ****
. While $Local \neq \emptyset$ do
. . Pull from $Local$ the match (c,d) which maximizes $r(c,d)$
. . If c and d are not already matched in the disparity map Map
. . Then store match (c,d) in the disparity map Map and heap $Seed$
. . End if.
. End while.
End while.

If the scene is rigid, it is possible to add the epipolar constraint for match (c,d) in the line
if c and d are not already matched and $s(c) > s_{min}$ and $s(d) > s_{min}$ and $r(c,d) > r_{min}$

7 Annex B: Joint View Triangulation

7.1 Overview

We propose a five-step algorithm which robustly converts an imperfect displacement map to a JVT.

- **Fitting**: Partition the first image into a set of independent regular patches, and for each one, try to fit a matched patch in the second image using inner matches (see subsection 7.2).
- **Averaging**: Remove small discontinuities and overlaps between the matched patches in the second image, by slightly moving their vertices to averaged locations. (see subsection 7.3).
- **Merging**: This step is more delicate. It grows the regions of matched triangles in the two images simultaneously in a coherent and robust way, by merging patches which can be smoothly joined to the current region boundaries (see subsection 7.4).
- **Completion**: The three previous steps are not optimal because of the parameter choices, but produce a coherent JVT. We improve the structure by declaring each unmatched triangle to be matched, if we can fit a matched triangle to it. This step modifies the structure by swapping constrained-unconstrained status on existing edges and then it is easy.
- **Optimization**: This step depends on the application. One can improve the matching triangles accuracy by perturbing the vertices to optimize a criterion (e.g. correlation score, inlier rate, smoothness, epipolar errors...). Simplification is also possible by deleting some vertices. We have not used this step for the presented morphing application.

Finally, a discussion about two important parameters of this algorithm is described in subsection 7.5.

7.2 Fitting Step

Because of noisy and/or bad matches, some caution is needed to obtain a reliable estimate of a matched patch P_2 in the second image from dense matches in a square patch P_1 in the first one (see Figure 11).

7.2.1 Principle

We try to fit a plane homography H (see the next subsection) using a RANSAC-like [FB81] procedure from the dense matches within P_1. If one is found, the relative coherence of the matches is checked and P_2 is defined by $P_2 = H(P_1)$. However, we do not accept the patch match (P_1, P_2) if the distortion of P_2 is too large.

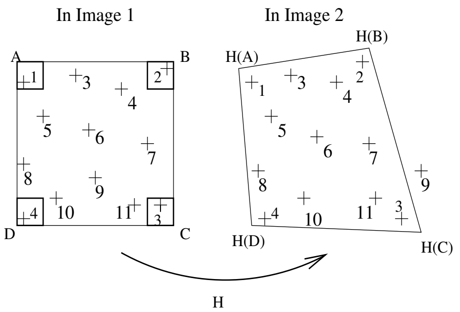


Figure 11: Points A,B,C, and D define a square patch P_1 in image 1. A sparse subset of the dense matching within is labeled from 1 to 11. The matches 1,2,3 and 4 (selected by a RANSAC trial) are respectively in the small framed 3×3 neighborhood of vertices A,B,C and D. They are used to accurately define a planar homography H, which maps the square patch P_1 in image 1 to the distorted square patch P_2 defined by transformed points $H(A), H(B), H(C)$ and $H(D)$ in image 2. All matches (except 9) are compatible with H.

For each RANSAC trial, four matches are selected in the square; this defines a trial homography (see the next subsection). These four matches are chosen from the neighborhood of the four corners to obtain a usable accuracy and to ensure a good match distribution. The second part of a RANSAC trial counts the number of matches in the square compatible with the current homography. The best homography maximizes the number of inliers.

7.2.2 Plane Homography

A point in an image is represented by its homogeneous coordinates $(\lambda x, \lambda y, \lambda)^\top$ or its Cartesian coordinates $(x, y)^\top$. A plane homography H is a one to one mapping which transforms a point $m_1 = (x_1, y_1, 1)^\top$ to a point $m_2 = (x_2, y_2, 1)^\top$ such that

$$\lambda \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}.$$

One can fit a plane homography from four matches (u_i, u'_i), no three of them are collinear. Indeed, each match provides 2 homogeneous linear equation in coefficients h_{ij} from relation $\lambda_i u'_i = H u_i$, and H counts only $8 = 9 - 1$ degree of freedom because its coefficients are defined up to a scale.

7.3 Averaging Step

The previous step provides a set of image patch matches (P_1, P_2), but the patches in the second image are not exactly adjacent (see Figure 12). We next perturb the vertex locations in the second image to eliminate the small discontinuities and overlaps between patches.

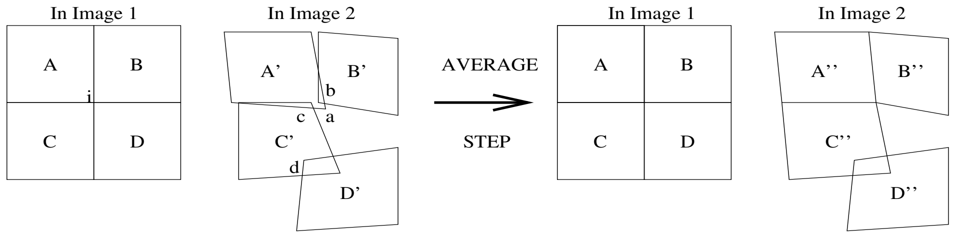


Figure 12: A,B,C and D are four patches of image 1 and A',B',C' and D' are their corresponding patches in image 2. The averaging step forces some patch vertices to coincide if they are enough close from each others. A", B", C", and D" are the result of the averaging step. Note that this step improves but can not solve all cases of intersecting patches (e.g. C" with D").

Let A, B, C, D be four connected squares (or nil if none), i their common vertex, and a (resp. b, c and d) the associated vertices of matched patches A' (resp. B', C' and D'). Let $s_{connect}$ be the threshold for the maximum distance between vertices. Let G be the unoriented graph whose vertices are a, b, c, d with edge ab (resp. ac, ad, bc, bd and cd) if $||ab||_2 < s_{connect}$ (resp. $||ac||_2, ||ad||_2, ||bc||_2, ||bd||_2 < s_{connect}$), where $||xy||_2$ is the Euclidean distance between vertices x and y. The averaging step consists to assign to a (resp. b, c, d) the average value of all the vertices a, b, c, d which are in the connected component of a (resp. b, c, d) of the graph G. Note that this step improves but can not solve all cases of intersecting patches (see caption of Figure 12). In this work, $s_{connect}$ is a constant equal to 3 pixels.

7.4 Merging Step

The previous steps produce a globally incoherent set of patch matches because of intersections. Many maximal and non-self intersecting subsets are possible. The merging step selects one of these and converts it to an incomplete but coherent JVT (the next step will complete it). The coherence between the two views is maintained at each stage of the merging step (i.e. a one to one correspondence between all vertices and contour edges of the JVT in the two images).

7.4.1 Principle

The JVT starts from two unmatched triangles in both images and the sets of matched triangles is grown simultaneously in the two images (see Figure 13) using the two operators $InsertMatchedTriangle$ and $ForceMatchedQuadrangle$ (see subsection 7.4.3). Each checks the current JVT to decide whether its operation is feasible, and if so greedily applies it. If it is not feasible, an unmatched gap (a set of unmatched triangles) is left in the final triangulation, unless the following completion step fills it. We explain a topological choice for JVT in subsection 7.4.2 and specify the merging step using the two operators in subsection 7.4.4.

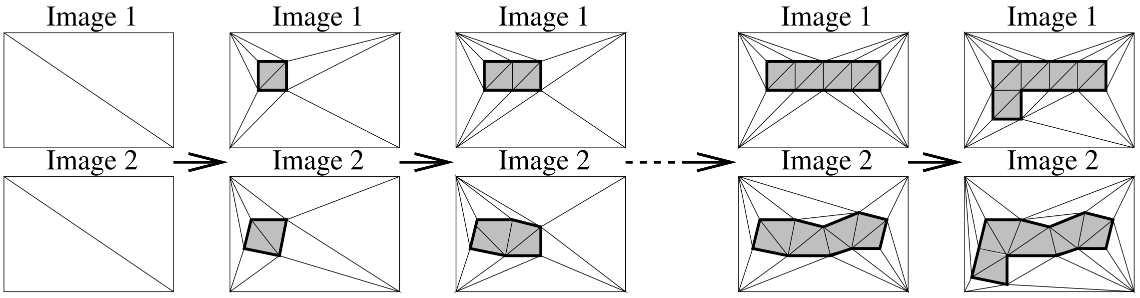


Figure 13: Gray (resp. white) triangles are matched (resp. unmatched) triangles. Black forced edges are constrained and form the contours. The sets of matched triangles grow simultaneously in the two images in a coherent way. Three insertions of patch matches (i.e. 3×2 $InsertMatchedTriangle$ operations) of a complete merging step are shown here.

7.4.2 Topological Limitation

We chose to limit the possible topologies of the set of matched triangles to simplify our algorithm: We do not accept the cases shown in Figure 14. A first concrete consequence is that the contour (the polygonal boundary of the set of matched triangles) has a simple representation. Each of its vertices has only two links: the next and the previous vertex. A second consequence is that algorithms which use the structure are simplified too. For instance, a simple walk using edge adjacencies suffices to cover a complete connected component of the set of matched triangles.

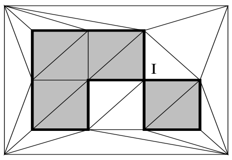


Figure 14: Gray (resp. white) triangles are matched (resp. unmatched). Black forced edges form the contour. More than two contour edges are adjacent to the same vertex I. We forbid a such case to simplify our algorithm and some others.

7.4.3 The operators $InsertMatchedTriangle$ and $ForceMatchedTriangle$

The $InsertMatchedTriangle$ operator takes the coordinates of three point matches in the two images and verifies that each of the three matches is *consistent* with the current structure. A new point match is *consistent* if its two points are corresponding vertices of a match of the contour, or are both outside of the set of matched triangles (see Figure 15). Second, it verifies that the resulting constrained edges would not intersect a contour in either image. The Figure 16 shows all good and some bad cases for edge configurations to obtain a success.

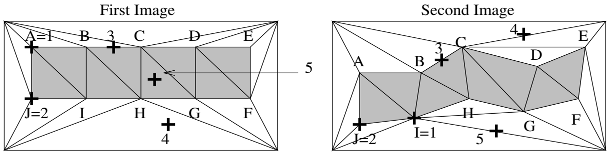


Figure 15: A,B...J are the current matched vertices of gray and matched triangles in the two images. A new point match is *consistent* if its two points are corresponding vertices of a match of the contour, or are both outside of the set of matched triangles. Matches 2 and 4 are *consistent* but 1,3 and 5 are not. Note that match 4 is *consistent* even though the matching ordering constraint is violated (see the ball in Figure 1).

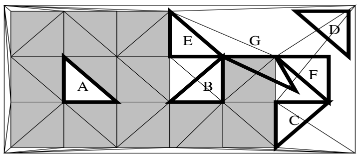


Figure 16: Gray triangles are matched triangles and the topology of their set is the same in the two images. Black framed triangles A,B,C and D (resp. E,F,G) are (resp. are not) correct inputs for the $InsertMatchedTriangle$ operator. Thus, triangle A,B,C can be added in both images to the current structure as matched triangles. Triangles E and F violate the topological restriction. One edge of Triangle G intersects an edge of the contour.

The $ForceMatchedQuadrangle$ operator takes coordinates of four point matches and verifies that each of the four matches coincides with a vertex match of the current structure. Second, it verifies that if the resulting constrained edges would not intersect a contour in either image. The Figure 17 shows the only three possible cases for a success. Note that we could not build a complete set of matched triangles homeomorphic to a planar ring without $ForceMatchedQuadrangle$ because of the topological restriction.

7.4.4 Merging Strategy

The two operators above are used in the merging step. A maximal set of coherent patch matches is converted into two dependent sets of matched triangles in the two images, row by row from the top to the bottom of the regular grid in the first image. Figure 18 shows a

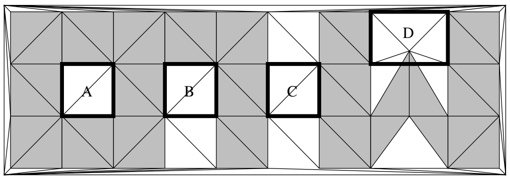


Figure 17: Gray triangles are matched triangles and the topology of their set is the same in the two images. Black framed quadrangles A,B,C (resp. D) are (resp. are not) correct inputs for the $ForceMatchedQuadrangle$ operator. Thus, Quadrangles A,B,C can be added in both images to the current structure as matched triangles. One edge of quadrangle D intersects an edge of the contour, and D is therefore rejected.

concrete example: the matched triangles of row n-1 were inserted in the current JVT (Figure 18.1) and impose topological constraints to the row n construction (Figure 18.3). To obtain the result (Figure 18.2), we use a two pass algorithm (Figure 18.4):

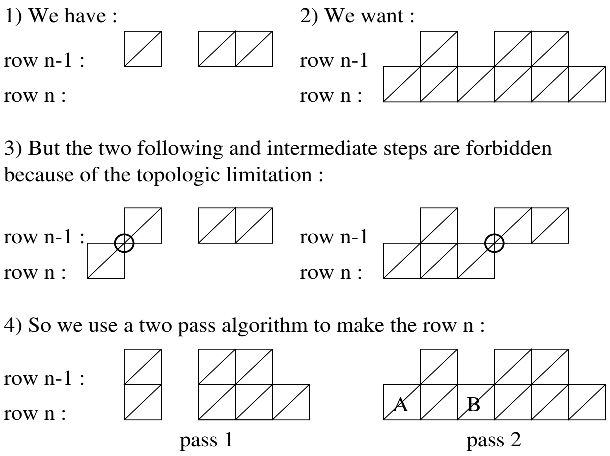


Figure 18: Only matched triangles are drawn and the topology of their set is the same in the two images. Part 1 shows the initial set, 2 the resulting set, 3 the problems to be avoided and 4 our solution.

1. use $InsertMatchedTriangle$ operator twice for each patch match from left to right avoiding topological limitations (Figure 13 showed a case without limitations).
2. complete some gaps in the current line by trying to grow the set of matched triangles using the operators $InsertMatchedTriangle$ (e.g. for patch A in Figure 18.4) and $ForceMatchedQuadrangle$ (e.g. for patch B in Figure 18.4).

7.5 Discussion on Main Parameters for JVT

Two important parameters of our algorithm are the size of the regular patches in the first image, and the connection radius $s_{connect}$ in the averaging step.

The patch size defines the global accuracy of the joint view triangulation. If it is too large, the approximation of the boundaries between matched and unmatched areas is too coarse. For instance, fine occluded areas are ignored and then recovered by matched triangles. Further, the fitting step can not define patch matches for matched but strongly curved areas: these areas are covered by unmatched triangles. On the other hand, the fitting step is unstable if the size is too small: some matched patches in the second image are too stretched. In practice, with our pixelic region growing matching, a good compromise seems to be a patch size between 8 and 16 pixels.

Parameter $s_{connect}$ is a maximum distance for merging patch vertices in the averaging step. Two vertices should coincide if their different location comes from matching inaccuracy but not if it comes from occlusion. If $s_{connect}$ is too large, fine occluded areas in the second image are ignored: patch vertices on both sides are merged. On the other hand, matched patches should be connected in the second image but are not if $s_{connect}$ is too small. In fact, the distinction between matching inaccuracy and thin occluded areas is unclear. We choose $s_{connect} = 3$ pixels for our tests.

Patch operations like detecting discontinuities, averaging and merging exist in the computer aided geometric design domain, for example the correction of CAD data with inconsistencies [UMH98].

8 Annex C: Image Interpolation

8.1 Morphing

A joint view triangulation is built from image pair I_0, I_1. We now describe how to use it to generate intermediate images $I(\lambda), \lambda \in [0,1]$ between I_0 and I_1 such that $I(0) = I_0, I(1) = I_1$.

To obtain $I(\lambda)$ from I_0, I_1, let $\tilde{I}_0$ and $\tilde{I}_1$ be two intermediate buffers. First, all triangles of the joint view triangulation of I_0 (resp. I_1) are warped into $\tilde{I}_0$ (resp. $\tilde{I}_1$) in a painter-like order. The following subsection explains how to warp a single triangle. Next, the final image $I(\lambda)$ is obtained by texture blending of $\tilde{I}_0$ and $\tilde{I}_1$.

For the warp from I_0 to $\tilde{I}_0$, unmatched triangles are drawn before matched ones because they usually contain occluded areas. The small unmatched triangles in I_0 with two vertices on different connected components (e.g. trunk and background) do not correspond to an object surface and should be drawn first. Thus, we choose to draw all unmatched triangles v_0^0, v_1^0, v_2^0 such that the value $Max\{||v_0^1 - v_1^1||, ||v_1^1 - v_2^1||, ||v_2^1 - v_0^1||\}$ decreases, where v_i^1 is matched with v_i^0. Matched triangles are then drawn. We assume that the vertices with the largest displacements between I_0 and I_1 are the closest to the viewer. Thus, we choose to draw all matched triangles v_0^0, v_1^0, v_2^0 in increasing order of $Max\{||v_0^1 - v_0^0||, ||v_1^1 - v_1^0||, ||v_2^1 - v_2^0||\}$.

For the final texture blending of $\tilde{I}_0$ and $\tilde{I}_1$, we use texture weights $s_0(x,y)$ and $s_1(x,y)$ for pixel (x,y) in $\tilde{I}_0$ and $\tilde{I}_1$, provided by the triangle warpings. The resulting value of pixel $I(\lambda)(x,y)$ is then

$$I(\lambda)(x,y) = \frac{(1-\lambda)s_0(x,y)\tilde{I}_0(x,y) + \lambda s_1(x,y)\tilde{I}_1(x,y)}{(1-\lambda)s_0(x,y) + \lambda s_1(x,y)}.$$

8.2 Warp a Triangle

We have to draw a 'source' triangle with vertices $v_0^0, v_1^0, v_2^0 \in I_0$ (resp. $v_0^1, v_1^1, v_2^1 \in I_1$) of a joint view triangulation. Let v_0^1, v_1^1, v_2^1 (resp. v_0^0, v_1^0, v_2^0) be their respective vertex matches in the other image. The vertices $v_0(\lambda), v_1(\lambda), v_2(\lambda)$ of the 'destination' triangle in image $\tilde{I}_0$ (resp. $\tilde{I}_1$) are defined by λ: $v_i(\lambda) = (1-\lambda)v_i^0 + \lambda v_i^1$. The texture of the source triangle is mapped on the destination triangle using linear interpolation. We do not draw the source triangle if the destination is reversed

A texture weight s is also estimated for each pixel of the destination triangle. It is used to weight the final blending between $\tilde{I}_0$ and $\tilde{I}_1$. Normal weight ($s = 1$) is assigned for non stretched triangles, and low weight ($0 < s < 1$) for stretched ones. We use $s = Min(1, \frac{||v_0^0 v_1^0 v_2^0||}{||v_0^1 v_1^1 v_2^1||})$ (resp. $s = Min(1, \frac{||v_0^1 v_1^1 v_2^1||}{||v_0^0 v_1^0 v_2^0||})$ in our tests, where $||abc||$ is the area of the triangle a, b, c.

ACKNOWLEDGMENTS Thanks to Long Quan, Roger Mohr and Bill Triggs for discussions and reading my papers.

References

[AHU74] A.V. Aho, J.E Hopcroft and J.D. Ullman. The design and analysis of computer algorithms. *Addison-Wesley, Reading, MA, USA*, 1974.

[AS97] 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.

[BFB94] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1): 43–77, 1994.

[BM97] J. Blanc and R. Mohr. Towards Fast and Realistic Image Synthesis from Real Views. In *10th Scandinavian Conference on Image Analysis, Finland*, pages 455–461, 1997.

[BN92] T. Beier and S. Neely. Feature-Based Image Metamorphosis *SIGGRAPH'92 Proceeding*, pages 35–42, 1992.

[BSA98] S. Baker, R. Szeliski and P. Anandan. A Layered Approach to Stereo Reconstruction. *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 434–441, 1998.

[Che95] S.E. Chen. Quicktime VR - an image-based approach to virtual environment navigation. In SIGGRAPH *1995, Los Angeles*, pages 29–38, 1995.

[DA89] U.R. Dhond and J.K. Aggarwal. Structure from stereo – a review. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6): 1489–1510, 1989.

[DDP97] F. Durand, G. Drettakis and C. Puech. The Visibility Skeleton: A Powerful And Efficient Multi-Purpose Global Visibility Tool. *Proceedings SIGGRAPH'97*, pages 89–100, 1997.

[DTM96] P.E. Debevec, C.J. Taylor and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Proceedings SIGGRAPH'96*, pages 11–20, 1996.

[FB81] 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, 1981.

[FK98] O. Faugeras and R. Keriven. Complete Dense Stereovision Using Level Set Methods. *Proceedings of 5th European Conference on Computer Vision*, volume 1, pages 379–393, 1998.

[FL94] P. Fua and Y.G. Leclerc, Using 3-Dimensional Meshes To Combine Image-Based and Geometry-Based Constraints. *Proceedings of 4th European Conference on Computer Vision*, pages 281–291, 1994.

[GCS91] Z. Gigus, J. Canny and R. Seidel. Efficiently Computing and Representing Aspect Graphs of Polyhedral Objects. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 13(6): 542–551, 1991.

[GSB97] M.A. Garcia, A.D. Sappa and L. Basafie. Efficient Approximation of Range Images Through Data-Dependent Adaptative Triangulation. *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 628–633, 1997.

[HS85] R.M. Haralick and L.G. Shapiro. Survey, image segmentation techniques. *Computer Vision, Graphics, and Image Processing*, 29: 100–132, 1985.

[HS88] C. Harris and M. Stephens. A combined Corner and Edge Detector. *Alvey Vision Conference*, pages 147–151, 1988.

[HSG96] O. Henricsson, A. Streilein, and A. Gruen. Automated 3-D reconstruction of buildings and visualization of city models. In *Proceedings of the Workshop on 3D City Models, Bonn, Germany*, 1996.

[JDF91] J.D. Foley, A. van Dam, S.K. Feiner and J.F. Hughes. Computer Graphics: Principles and Practice. *Addison-Wesley*, 1991.

[KM96] T. Kim and J.P. Muller. Automated urban area building extraction from high resolution stereo imagery. *Image and Vision Computing*, Volume 14, pages 115–130, 1996.

[Koc95] R. Koch. 3D Surface Reconstruction from Stereoscopic Image Sequences. *Proceedings of the 5th International Conference on Computer Vision*, pages 109–114, 1995.

[Kos93] A. Koschan. What is new in Computational Stereo Since 1989: A Survey on Current Stereo Papers. Technical Report 93-22, University of Berlin, 1993.

[LCH96] 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.

[LDR98] C. Loscos, G. Drettakis and L. Robert. Interactive Modification of Real and Virtual Lights for Augmented Reality. *Siggraph'98 technical sketch*, 1998.

[LF94] S. Laveau and O.D. Faugeras. 3D Scene representation as a collection of images. *Proceedings of 12th International Conference on Pattern Recognition*, pages 689–691, 1994.

[LF96] F. Lang and W. Forstner. Surface reconstruction of man-made objects using polymorphic mid-level features and generic scene knowledge. *International Archives of Photogrammetry and Remote Sensing*, XXXI, Part B3: 415–420, 1996.

[Lhu98] M. Lhuillier. Efficient Dense Matching for Textured Scenes Using Region Growing. *the Ninth British Machine Vision Conference*, pages 700–709, UK, 1998 (available at http://www.inrialpes.fr/movi/people/Lhuillie/demo.html), also Technical Report 3382, INRIA.

[MB95] L. McMillan ang G. Bishop. Plenoptic Modeling: An Image-Based Rendering System. In *SIGGRAPH'95*, pages 39–46, 1995.

[Mon87] O. Monga. An optimal region growing algorithm for image segmentation. *International Journal of Pattern Recognition and Artificial Intelligence*, 1(3): 351–375, 1987.

[NB94] W. Niem and R. Bushmann. Automatic Modelling of 3D Natural Objects from Multiple Views. in *Image Processing for Broadcast and Video Production.* Workshop in Computer Science Series (Springer-Verlag, 1994), ISBN 3-540-19947-0. (demo and paper availables at http://www.tnt.uni-hannover.de/project/3dmod/multview).

[OC89] G.P. Otto and T.K. Chau. A region-growing algorithm for matching of terrain images. *Image and Vision Computing*, 7(2): 83–94, 1989.

[PS85] F. Preparata and M.I. Shamos. *Computational Geometry, An Introduction.* Springer-Verlag, Berlin, Germany, 1985.

[Que97] G. Quenot. Computation of Optical Flow Using Dynamic Programming and Applications. Demo session of the *Conference on Computer Vision and Pattern Recognition*, page 5, 1997 (see http://www.limsi.fr/Individu/quenot).

[Rip90] D. Rippa. Minimal roughness property of the Delaunay triangulation. *Computer Aided Geometric Design*, 7: 489–497, 1990.

[RHM95] M. Roux, Y.C. Hsieh and D.M. McKeown. Performance analysis of object space matching for building extraction using several images. In McKeown and Dowman editors, *SPIE Integrating Photogrammetric Techniques with Scene Analysis and Machine Vision II, vol. 2486*, 1995.

[Sch96a] D. Scharstein. Stereo Vision for View Synthesis. *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 343–350, 1996.

[Sch96b] C. Schmid. Appariement d'images par invariants locaux de niveaux de gris. Thèse de doctorat, GRAVIR-IMAG-INRIA Rhône-Alpes, France, 1996 (available at ftp://ftp.imag.fr/pub/Mediatheque.IMAG/theses/96-Schmid.Cordelia/).

[SD95] S.M. Seitz and C.R. Dyer. Physically-valid view synthesis by image interpolation. *IEEE Workshop on Representation of Visual Scenes*, pages 26–33, 1995.

[SD96] S.M. Seitz and C.R. Dyer. View Morphing. *SIGGRAPH'96*, pages 21–30, 1996.

[SD97] S.M. Seitz and C.R. Dyer. Photorealistic Scene Reconstruction by Voxel Coloring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1067–1073, 1997 (paper available at http://www.cs.wisc.edu/~seitz).

[SDB97] F. Sillion,G. Drettakis and B. Bodelet. Efficient Impostor Manipulation for Real-Time Visualization of Urban Scenery. *Eurographics'97* (D. Fellner and L. Szirmay-Kalos, eds.), 16(3), 1997 (available at http://w3imagis.imag.fr/Publications/index.gb.html).

[SK98] Y. Shinagawa and T.L. Kunii. Unconstrained Automatic Image Matching Using Multiresolutional Critical-Point Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (D. Fellner and L. Szirmay-Kalos, eds.), 20(9): 994–1008, 1998.

[SMB98] C. Schmid, R. Mohr and C. Bauckhage. Comparing and Evaluating Interest Points. *Proceedings of the 6th International Conference on Computer Vision*, 1998 (Extended Version available at http://www.inrialpes.fr/movi/pub/Publications/en/publis.html).

[Sze96] R. Szeliski. Video Mosaics for Virtual Environments. *IEEE Computer Graphics and Applications*, pages 22–30, 1996.

[TFZ96] P.H.S. Torr, A. Fitzgibbon and A. Zisserman. Maintaining multiple motion model hypotheses over many views to recover matching and structure. *Proceedings of 6th International Conference on Computer Vision*, pages 727–732, 1998. (demo and paper availables at http://www.robots.ox.ac.uk/~vanguard).

[TZ96] 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.

[UMH98] A.E. Uva, G. Monno and B. Hamann. A New Method for the Repair of CAD Data with Discontinuities. *II Seminario Italo-Spagnolo on "Progettazione e Fattibilità dei Prodotti Industrial"*, Vico Equense - Naples, Italy, 1998, (available at http://muldoon.cipic.ucdavis.edu/~uva/papers.html).

[Wol90] G. Wolberg. Digital Image Warping. *IEEE Computer Society Press, Los Alamitos, California*, 1990.

[WW92] A. Watt and M. Watt. *Advanced Animation and Rendering Techniques.* ACM Press, 1992.

[ZDF94] Z. Zhang, R. Deriche, O. Faugeras, and Q.T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. Technical Report 2273, INRIA, May 1994.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399