BANDELET-BASED VIDEO INPAINTING

Aldo Maalouf, Philippe Carré, Bertrand Augereau, Christine Fernandez-Maloigne

XLIM Laboratory, UMR CNRS 6172, Department of Signal, Image and Communications
University of Poitiers
SP2MI-2 Bd Marie et Pierre Curie, PO Box 30179
86962 Futuroscope Chasseneuil, France
{maalouf, carre, augereau, fernandez}@sic.sp2mi.univ-poitiers.fr

ABSTRACT

In this work, a framework for inpainting missing parts of a video sequence is presented. The first step in the algorithm consists of finding the geometrical flow of each frame in the video sequence by using the Bandelet transform. Consequently, each frame is segmented into a quadtree where each dyadic square regroups pixels sharing the same geometrical flow direction. Then, two cases are considered. The first case is concerned with the removal of non-stationary objects that occlude stationary background. For this, the missing portions of the video sequence are filled by searching for pairs of dyadic squares in the successive frames quadtrees that are valid neighbors to the dyadic squares at the boundary of the inpainting zone. The search for valid neighbors squares is accomplished by the minimization of a cost measure that studies the geometrical similarity of each pair of squares. Once neighbors squares are identified, the corresponding squares in the original frame images are used to fill-in the missing areas. The second case involves filling-in moving objects when they are partially occluded. For this, we propose a priority scheme to first inpaint the occluded moving objects and then fill-in the remaining area with stationary background using the method proposed for the first case. Optical flow is used here, which tells if an undamaged pixel is moving or is stationary. We evaluate the performance of our algorithm based on a set of video sequences with different types of occlusions.

1. INTRODUCTION

A lot of research was recently done on so called image inpainting algorithms which perform the task of filling in missing or destroyed regions in images. The two main approaches thereby considered are partial differential equation (PDE) [1] [2] [5] [3] [4] and texture synthesis algorithms, usually presuming Markov Random Fields as the underlying image model [6] [7]. Unlike image inpainting, video inpainting has just recently started receiving more attention. It presents an entirely different set of challenges due to the temporally continuous nature of the data. One of the first efforts can be found in [8] where the PDE is applied spatially, and completes the video frame-by-frame. This does not take into account the temporal information that a video provides. The space-time video completion work in [9] extends the exemplar-based to video by posing the video inpainting as a global optimization problem. However, the results shown are for very low resolution video sequences, and the inpainted static background was different from one frame to another creating a ghost effect.

An interesting work for video inpainting has been reported in [10]. It consists of separating foreground objects from background, and then filling the background image using other image inpainting algorithm and steering the inpainting of the foreground object in the directions most consistent with the local motion. Although it showed good inpainting results, this method takes an inordinate amount of computation time. A simpler method was proposed in [11]. It uses background subtraction and object segmentation to extract a set of object templates and perform optimal object interpolation using dynamic programming to inpaint moving objects.

Although it requires less computation time than the approach presented in [10], the quality of the obtained results depends on the segmentation. Furthermore, it works well only in the case where the background texture is uniform and it does not provide smooth transition at the boundaries of the inpainting zone. In addition, moving objects may appear to be sliding from one frame to another.

In this work, the problem of video inpainting is addressed from a new angle. Instead of using background subtraction and object segmentation, we pose it as a problem of a geometrical similarity optimization.

First, the bandelet transform of each frame is computed. Consequently, each frame is segmented into multiple dyadic squares where the pixels in each square share the same geometrical properties. The motivation behind the usage of the bandelet transform is that the geometry of each frame in the video sequence is summarized with local clustering of similar geometric vectors, the homogeneous areas being taken from the quadtree structure. This will allow us to search for areas in the whole sequence that are geometrically similar to the zones around the inpainting domain and consequently these areas can be used to fill in the missing part.

Then, two cases are considered: the case of the removal of non-stationary objects that occlude stationary background and the case of filling-in moving foreground when they are partially occluded.

For the first case, we search for possible neighbors dyadic squares to the dyadic squares of the available parts of the background at the boundaries of the inpainting zone. The search is performed on the whole sequence. Consequently, for each square of the boundary of the available part, we search in the quadtrees of the previous and the succedent frames for possible neighbor dyadic squares. The search is based on the geometrical properties of the dyadic squares which are induced by the bandelet transform, i.e., the side length and the geometrical direction of each dyadic square. The search problem is accomplished by the minimization of a cost measure. After identifying all possible neighbor dyadic squares in the successive frames, the corresponding squares in the original frame images are used to fill in the correspond-

ing missing or damaged zone. That is, for each dyadic square used to fill in the inpainting zone, the corresponding area (or square) in the original frame image is used to fill in the missing part at the same position of dyadic square.
For the case of filling-in moving foreground when they are partially occluded,we assume the knowledge obtained from pre-computed optical flow [12] of whether a pixel is moving or not. Then, the static background is filled-in by using the method proposed for the first case and the moving foreground is completed by searching for the best neighbor dyadic squares in the moving parts from the undamaged parts in the whole video sequence.
The rest of the paper is organized as follows. In section 2, a review of the Bandelet transform is presented. In section 3, the bandelet-based video inpainting method is described. Experimental results are shown in section 4 and section 5 presents some concluding remarks.

2. BANDELET TRANSFORM

We only present here a brief review of the Bandelet transform. The reader can refer to [13] for a full detailed description of the Bandelet transform.
The bandelets are defined as anisotropic wavelets that are warped along the geometric flow, which is a vector field indicating the local direction of the regularity along edges. The dictionary of bandelet frames is constructed using a dyadic square segmentation and parameterized geometric flows. The ability to exploit image geometry, makes its approximation optimal for representing the images.
For image surfaces, the geometry is not a collection of discontinuities, but rather areas of high curvature. The Bandelet transform recasts these areas of high curvature into an optimal estimation of regularity direction. Figure 1 shows an example of bandelets along the geometric flow in the direction of edges. In real applications, the geometry is

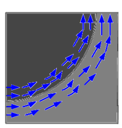


Figure 1: An illustration of bandelets with geometric flows in the direction of the edge. The support of the wavelets is deformed along the geometric flows in order to exploit the edge regularity.

estimated by searching for the regularity flow and then for a polynomial to describe that flow.

Implementation of the Bandelet Transform

The classical tensor wavelet transform of an image I is the decomposition of the latter on an orthogonal basis formed by the translation and dilation of three mother wavelets for the horizontal, vertical and diagonal directions. Once the wavelet transform is found, the quadtree is computed by dividing the image into dyadic squares with variable sizes (refer to [14] for more information on computing the quadtree). For each square in the quadtree the optimal geometrical direction is computed by the minimization of a lagrangian (refer also to [14]). Then a projection of the wavelet coefficients along the optimal direction is performed [14]. Finally a 1D discrete wavelet transform is carried on the projected coefficients.
On figure 2 one can see, after the bandelet decomposition, the quadtree, and a zoom on the orientation of the linear flow on each dyadic square. Notice that the quadtree

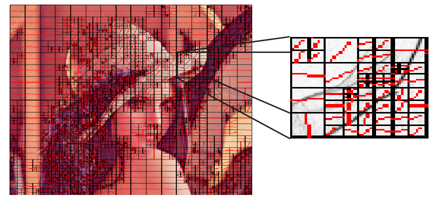


Figure 2: Lenna image quadtree segmented

segmentation performs very well in the area corresponding to edges.
The geometry encoded by the bandelet transform will be used in the definition of the cost measure. The role of this cost measure is to determine the geometrical similarity of all pairs of dyadic squares in the successive quadtrees of each frame image in the video sequence. Particularly, the size and the optimal geometrical direction of each square will be used as criteria to study the similarity. This idea is discussed in details in the following section.

3. BANDELET-BASED VIDEO INPAINTING

In this section, we describe our method for inpainting missing parts of a stationary background that is occluded by a moving object and the filling-in of moving objects that are partially occluded by stationary or moving foreground or object. This section will be divided into two parts treating each case separately.

3.1 Inpainting Static Background

In this subsection we describe our method of filling-in missing stationary background occluded by moving or stationary objects. We search for dyadic squares in the successive frames that are neighbors according to a cost measure to the dyadic squares at the boundaries of the available parts (Fig. 3). This defines a path in the space-time domain for each dyadic square at the boundary of the missing zone. This path links each of those dyadic squares with its possible neighbors in the previous and the following frames. A point in this path has two neighbor squares, one before it and one after it.
While the number of paths in the volume is exponentially large, we do not need to check all of them. The locality allows us to use dynamic programming to find the global optimum efficiently. For a dyadic square in one image we test several possible transitions (or paths) onto the next images. Each transition can be scored according to the geometrical similarities between the two squares at each end.
To find the optimal path we construct a graph where the nodes represent dyadic squares and the edges are the possible transitions. The leaves of this graph define the best neighbor square.
It is to be noted that the inpainting procedure is performed in an iterative manner. That is, we first define the inpainting zone (or damaged or missing zone) and then we search

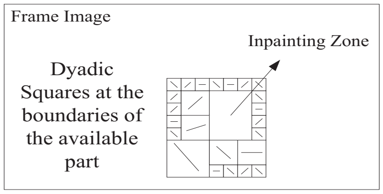


Figure 3: Inpainting Zone and Dyadic Squares at the boundaries of the available part, the inpainting procedure is performed by searching in the previous and succedent frames for neighbor dyadic squares to fill-in the inpainting region.

for each dyadic square at the boundary of this zone for neighbor squares. Then, the inpainting zone is redefined and the procedure is repeated until all the inpainting region is filled in. We represent in the following paragraphs the cost measure that is used to determine the neighbors squares.

Definitions

Let v be a video sequence, v is an ordered set of images $\{I_t\}_{t=1}^{N}$. After performing the bandelet transform, each image in turn contains an ordered set of dyadic squares $S_t^1, S_t^2 \ldots . S_t^K$ (where K is the total number of squares in the frame t). The dyadic squares in each quadtree are numbered from left to right and from top to bottom. The length of the side of the dyadic squares is made as small as possible (practically the maximum allowed length is fixed to 8 pixels).

We define the cost measure for any pair of dyadic squares $S_{t_1}^{k_1}$ and $S_{t_2}^{k_2}$ by:

$$D\left(S_{t_1}^{k_1}, S_{t_2}^{k_2}\right) = \min_{t} \min_{S_t^r S_t^s \subset I_t} \left| \left(\theta_{S_t^r}\theta_{S_t^s} - \theta_{S_t^{k_1}}\theta_{S_t^{k_2}}\right) \left(l_{S_t^r} l_{S_t^s} - l_{S_t^{k_1}} l_{S_t^{k_2}}\right) \right| \tag{1}$$

where, θ is the optimal geometrical direction in each dyadic square and l is the side length of the corresponding square.

That is, a pair of squares are valid neighbors if, when placed side-by-side, they are similar to some pair in one of the video images. We can get a good upper bound $D(.,.)$ by limiting the search to a window of neighbor squares in the preceding and succedent frames.

$$D\left(S_{t_1}^{k_1}, S_{t_2}^{k_2}\right) = \min\left\{ \left| \left(\theta_{S_{t_2}^{V_{k_2}}} - \theta_{S_{t_1}^{V_{k_1+1}}}\right)\left(l_{S_{t_2}^{V_{k_2}}} - l_{S_{t_1}^{V_{k_1+1}}}\right) \right|, \left| \left(\theta_{S_{t_1}^{V_{k_1}}} - \theta_{S_{t_2}^{V_{k_2-1}}}\right)\left(l_{S_{t_1}^{V_{k_1}}} - l_{S_{t_2}^{V_{k_2-1}}}\right) \right| \right\} \tag{2}$$

where V_{K_n} is the set of adjacent dyadic squares to the square number k_n.

The cost function (2) allows us to define a graph where the cost of each edge efficiently encodes the cost of considering the corresponding square as a valid neighbor to a square at the boundary of the inpainting zone.

Graph Construction

Let $G = (S, E)$ designates a graph where the nodes $S = \left\{S_t^k\right\}$ are the $K \cdot N$ image squares and the edges $E \subseteq S \times S$ encode the possible transitions between the squares. Each edge has an associated transition cost $D : E \to \mathbb{R}$ as defined in (2). Consider for example the two squares S_1^1 from the first frame and S_2^2 from the second frame, denoted by nodes in Fig. 4. The error of this placement is small if the pair S_1^3, S_2^4 appears together in some image. Instead of performing a costly search, we compute the similarity of S_1^3, S_2^3, and if they are similar, then S_2^4 can be placed next to S_1^3 (just like it is already placed next to S_2^3). Thus, the cost of a transition from node S_i^j to S_l^k is directly related to the similarity between S_i^j to S_l^{k-1} as given in equation (2).

Once we have found a neighbor for each square at the

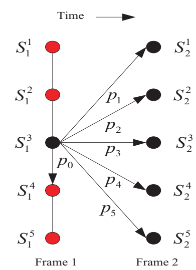


Figure 4: Graph construction. The graph encodes possible dyadic squares transitions. Each node denotes a square and the edges represent possible transitions between them. A square in Frame 1 has one natural transition $(p_0) = D\left(S_1^3, S_1^4\right)$ within the same frame and 5 possible transitions to the next frame $(p_1, \ldots, p_5) = (D(S_1^3, S_2^1), \ldots\ldots, D(S_1^3, S_2^5))$.

boundary of the inpainting zone by using this search procedure, the inpainting zone is redefined by the remaining missing parts and the same procedure is repeated for the dyadic squares at the boundaries of the new inpainting zone. Finally, the part from the original frame image corresponding to each of the neighbor squares is used to fill-in the inpainting zone.

We represent in the next subsection our method for filling-in moving foreground that is partially occluded by a stationary or a second moving object.

3.2 Inpainting Moving Foreground

If the target object occludes other moving objects, the moving objects need to be inpainted after the target object is removed. In this subsection, we describe a procedure similar to the one showed in the previous section that aims at providing natural object movement during the occlusion and smooth transition at the boundaries of the occlusion. To simplify the explanation, we assume that there is one single moving object that needs to be inpainted. If there are multiple objects, one can apply the same technique to each object.

The moving object is filled in a frame-by-frame procedure, each frame being completed using the following steps:

1. Identify moving objects by using the optical flow [12].
2. Search in the moving objects for neighbor dyadic squares to those in the available part of the moving object. The search procedure is performed by using the cost measure (2).
3. Search for neighbor dyadic squares in the non-moving regions of the video to fill-in the missing parts of the static

(a) Synthetic video sequence with an occluded moving red rectangle

(b) Inpainted video sequence

Figure 5: Synthetic example of object completion

background. This search procedure is performed as in the first case of the filling-in of static background by using equation (2).

In the final step, for each dyadic square the corresponding portion from the frame of the original sequence is used to fill-in the inpainting zone.

4. EXPERIMENTAL RESULTS

Figure 5 shows an inpainting example of a synthetic video sequence. The experiment consists of completing the moving red rectangle being occluded. This artificial experiment shows that our method can perfectly complete occluded moving objects.

Figure 6 shows a typical video surveillance sequence (only 8 frames are shown, the video consists of 1800 frames). The aim is to remove the woman passing the hall and entering her office. This is a typical example combining the two inpainting cases: the case of the removal of non-stationary objects that occlude stationary background (which is the case of filling-in the static hall parts that are occluded by the moving woman) and the case of filling-in moving objects when they are partially occluded (which is the case of the door being opened by the moving woman). The corresponding inpainted frames are shown in figure 7. The inpainted background is consistent throughout the video.

Figure 8 shows the results of both background image inpainting when the bord is removed and the occluded object filled-in. Our algorithm performs very well even when the region to be inpainted is very large.

5. CONCLUSION

In this work, a video inpainting technique based on the bandelet transform is proposed. It consists of searching between the quadtrees of each frame in the sequence for possible neighbors dyadic squares for the squares of the available parts of the video sequence. A cost measure is defined which studies the geometrical similarities between the pairs of dyadic squares and determines if they are valid neighbors. The proposed method showed a good visual quality in inpainting static background as well as moving objects.

There are issues that should be addressed in the future work, e.g., dealing with illumination changes along the sequence.

REFERENCES

[1] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles and Coloma Ballester, "Image Inpainting," *In Kert Akeley, editor, Siggraph 2000, Computer Graphics Proceedings*, pp. 417–424, 2000.

[2] Marcelo Bertalmio, "Strong-continuation, contrast-invariant inpainting with a third-order optimal pde," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 15, no. 7, pp. 1934–1938, 2006.

[3] S. Kang T. Chan and J. Shen, "Euler's elastica and curvature-based inpaintings," *SIAM J. Appl. Math.*, vol. 63, no. 2, pp. 564–592, 2002.

[4] H. Grossauer and O. Scherzer, "Using the complex ginzburgŰlandau equation for digital inpainting in 2d and 3d," *in Proc. Scale Space Methods in Computer Vision, Lecture notes in Computer Science*, vol. 2695, pp. 225–236, 2003.

[5] Selim Esedoglu Andrea L. Bertozzi and Alan Gillette, "Inpainting of binary images using the cahn-hilliard equation," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 16, no. 1, pp. 285–291, 2007.

[6] P. Harrison, "A non-hieararchical procedure for re-synthesis of complex textures," *In 9-th International Conference in Central Europe on Computer Graphics, Visualisation and Computer Vision*, pp. 190-197, 2001.

[7] I. Drori, D. Cohen-Or and H. Yeshurun, "Fragment-based image completion," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 303-312, 2003.

[8] M. Bertalmio, A. L. Bertozzi and G. Sapiro, "Navier-strokes fluid dynamics, and image and video inpainting," *In Proceedings of International Conference on Computer Vision and Pattern Recognition*, vol. I, pp. 355-362, 2001.

[9] Y. Wexler, E. Shechtman and M. Irani, "Space-time video completion," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*, vol. 1, pp.:120-127, 2004.

[10] K. A. Patwardhan, G. Sapiro and M. Bertalmio, "Video Inpainting of Occluding and Occluded Objects," *IEEE International Conference on Image Processing (ICIP05)*, vol. 2, pp.: 69-72, 2005.

[11] S. Cheung, J. Zhao and M. V. Venkatesh, "Efficient Object-Based Video Inpainting," *IEEE International Conference on Image Processing (ICIP06)*, pp.: 705-708, 2006.

[12] B. Augereau, B. Tremblais and C. Fernandez-Maloigne, "Vectorial Computation of the Optical Flow in Color Image Sequences," *Thirteenth International Color Imaging Conference*, pp.: 130-134, 2005.

[13] Peyre G., *Geometrie multi-échelles pour les images et les textures*, Phd thesis, Ecole Polytechnique, December 2005.

[14] G. Peyre and S. Mallat, "Surfave compression with geometrical bandelets," *ACM Transactions on Graphics*, vol. 14, no. 3, 2005.

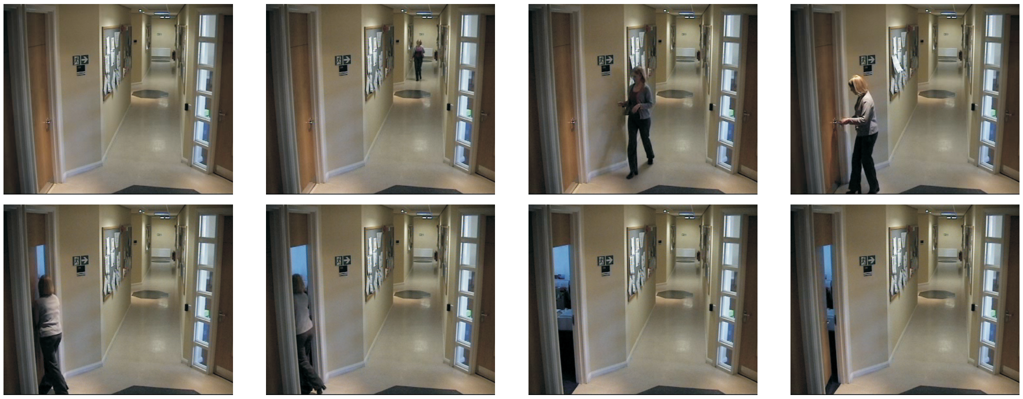


Figure 6: Part of the original frame sequence

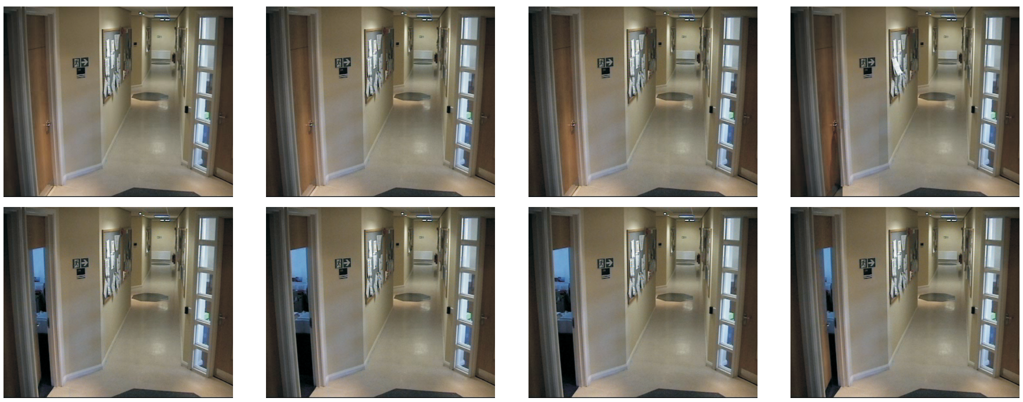
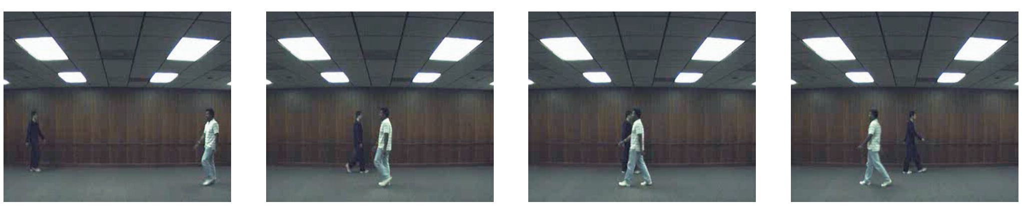
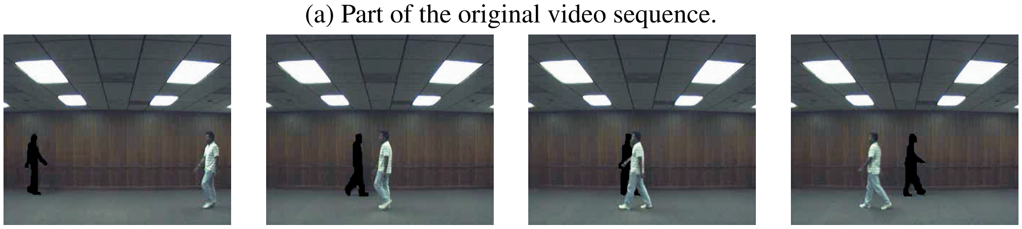


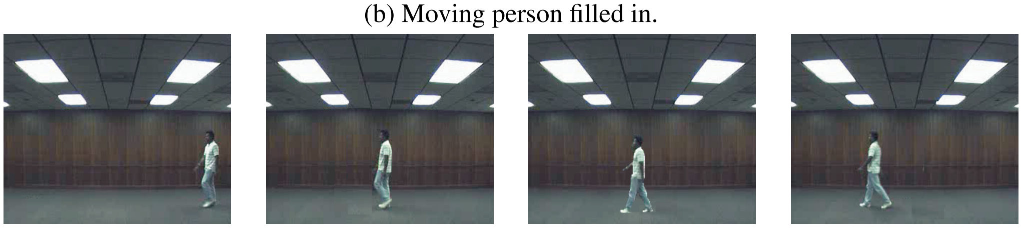
Figure 7: Corresponding inpainted frames



(a) Part of the original video sequence.



(b) Moving person filled in.



(b) Completely filled in sequence.

Figure 8: Corresponding inpainted frames