Clustering of Matched Features and Gradient Matching for Mixed-Resolution Video Super-Resolution

Renan U. Ferreira
Department of Electrical Engineering
University of Brasilia
Brasilia, Brazil
Email: renan@unb.br

Edson M. Hung
Electronic Engineering - UnB Gama Faculty
University of Brasilia - Gama Campus
Brasilia, Brazil
Email: mintsu@image.unb.br

Ricardo L. de Queiroz
Department of Computer Science
University of Brasilia
Brasilia, Brazil
Email: queiroz@ieee.org

***Abstract*—This work presents a novel technique for image reconstruction applied to mixed-resolution video super-resolution. We segment an image into patches defined by the clustering of a vector flow generated from matching SIFT features. We reconstruct the segmented image by applying image projective transformation to a reference image. By varying the number of clusters, we composed a sequence of reconstructed images, which are then used to compose a codebook, through gradient matching. This idea is extended to use low and high-resolution image pairs for super-resolution. Our results indicate a 1.4dB gain, on average, over the use of overlapped-block motion-compensation (OBMC).**

I. Introduction

Super-resolution (SR) is a process in which a single high-resolution (HR) image is composed either by several low-resolution (LR) images [1] or one low-resolution image and several low-high-resolution image pairs, known as example-based super-resolution [2]. In example-based super-resolution, a common approach is the use of pairs of patches for locally applying SR to an image [3].

In the field of video processing, SR has been used in a framework of mixed-resolution videos, i.e., videos that contain both LR or HR frames, in applications such as video coding with reversed complexity [4]. Since the similarities between LR and HR frames are intrinsic to the video scene, example-based SR is an approach that has shown good results. In this context, Song *et al.* [5] has proposed using sparsely existing high-resolution key-frames to super-resolve frames in a low-resolution sequence by using overlapped-block motion compensation (OBMC) and a dictionary training. Also, Hung *et al.* [6] proposed a SR technique through the use of codebooks, also derived from key-frames and with OBMC, achieving better results than those in [5].

We have presented, in a previous work [7], a technique based on SIFT features [8] and gradient matching for super-resolution of mixed-resolution video frames. One of its drawbacks was the need for manual parameter settings. In this work, we propose an improved approach that depends more on the images' characteristics, achieved through the clustering of a matching vector flow. Furthermore, we improved the gradient matching step. Our experiments show better results compared with works based on OBMC, specially for scenes that present more complex scene variations other than translation, such as scale, rotation, affine and perspective changes.

We present our work as follows: Section II presents the proposed image compensation technique based on the clustering of SIFT feature vector flow; Section III details the gradient matching with compensated images for image construction; Section IV shows how we combine the two previous steps for super-resolution; Section V presents our test conditions and results; and Section VI brings our conclusions to this work.

II. Image compensation from feature clustering

Let there be a current image $\boldsymbol{A}$ and a reference image $\boldsymbol{B}$, which are similar in the sense that they depict a similar scene. We assume these images share objects and a distinct set of features, i.e. some features in $\boldsymbol{A}$ are also present in $\boldsymbol{B}$. We build a compensated image $\boldsymbol{C}$ similar to $\boldsymbol{A}$, by using only pixels from $\boldsymbol{B}$, similarly to classical motion compensation (MC) [9], nevertheless exploring a larger variety of relative motions and scene variations between the two images, other than just translation. Figure 1 shows a diagram with the steps to compose the compensated image.

First, we select the set of shared features by matching those from both images $\boldsymbol{A}$ and $\boldsymbol{B}$ with Best-Bin-First and Nearest Neighbors techniques, as proposed by Lowe [8] and implemented by Hess et al. [10]. The difference between positions from each pair of matched feature produces a feature motion vector. We then compose a 4D vector in the form $[x, y, vx, vy]^T$, referred to as matching vector, in which x and y are the coordinates of a feature in image $\boldsymbol{A}$, while vx and vy are the feature motion vector components associated to the matched feature in image $\boldsymbol{B}$. The set of all the matching vectors compose a vector flow, which will be clustered. The use of 4D vectors is an usual approach in the clustering of vector flows, as in [11]. Figure 2 depicts an example of an image where the superimposed arrows represent the feature motion vectors. For this example, we used two frames from

978-1-4799-8391-9/15/$31.00 ©2015 IEEE

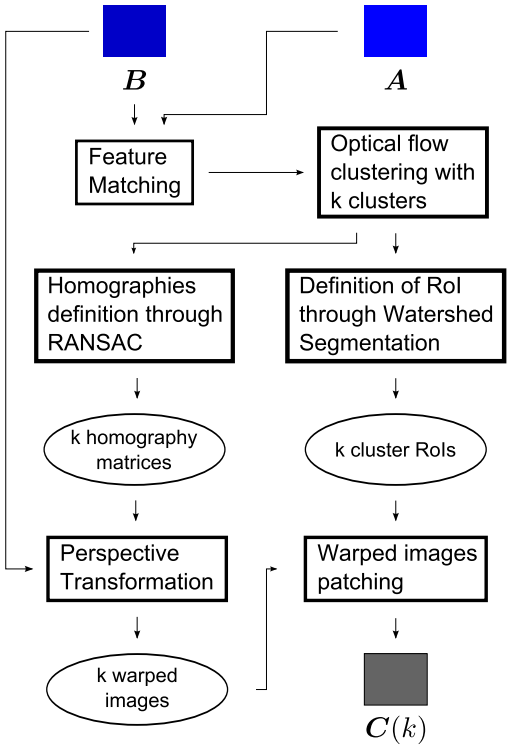

Fig. 1. Diagram of image composition through vector flow clustering.

video sequence *mobile*. The other superimposed markings will be explained in the following paragraphs.

To define the clusters, we use euclidean distance between each matching vector and inner squared distance between clusters, also known as Ward's method, which minimizes the total variance within each cluster [12]. These decisions were made empirically based on preliminary results. This clustering allows us to work with image's segments with arbitrary shapes, differently from the regular shaped blocks used in MC. Once the matching vectors are clustered, we must define two major aspects related to each cluster: a homography matrix and a region of influence (RoI).

Differently from the translation applied in MC, we perform a transformation in the reference image $\boldsymbol{B}$ according to the projective transformation matrix—linear transformation represented by a non-singular 3×3 matrix that can be calculated from four pairs of matched points in the 2D space [13], also known as homography—acquired from the feature motion vectors.

Since each cluster contains several matching vectors, we use RANSAC to determine the best homography matrix. The reference image $\boldsymbol{B}$ then suffers a projective transformation according to each derived matrix and is patched to its related RoI. For a small number of vectors, however, RANSAC may fail to produce a realistic homography, which leads to an erroneous compensated image. We will readdress this problem later on.

Each RoI is a patch in the compensated image $\boldsymbol{C}$ to be filled with the warped version of the reference image $\boldsymbol{B}$. In order to define the RoI for each cluster, we first acquire the clusters' convex hulls. Each convex hull is calculated according to the values of x and y of the matching vectors, i.e., all matching vectors in the same cluster are positioned inside the convex hull, but may point to an outside point.

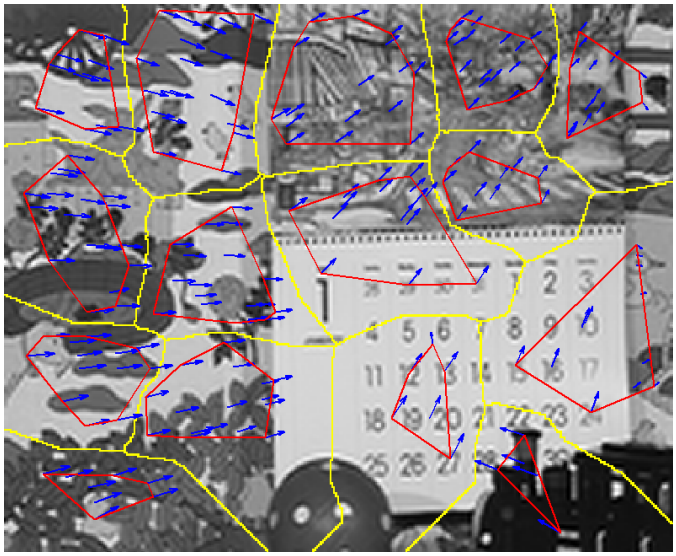

Fig. 2. Example of image segmented according to matching vectors separated in 15 clusters.

The convex hulls acquired from the clusters are contours of convex regions that do not span the entire image, which forbids them to be used as RoIs. These convex regions are then used to define the borders between all RoIs. We create a binary image in which all convex regions assume value 0, while the background assumes value 1. We then apply watershed segmentation [14] to this binary image, i.e., each convex region is a local minimum in the segmentation process. The segmentation process then defines the borders between catchment basins, which are our desired RoIs. In Figure 2, the red contours represent the convex hulls of the clustered vectors while the yellow lines indicated the borders of the RoIs.

Since the resulting compensated image depends on the number of clustered regions, we compose a set $\{\boldsymbol{C}(k)\}$ of distinct compensate images, where k indicates the number of clusters. The size of the set is defined by the maximum possible number of clusters, following a successive clustering approach. In this approach, we repeat the entire previously described image compensation process for each value of k clusters, in increasing order. However, if a cluster contains three or less vectors (in which case we could not calculate the homography), these vectors are excluded from the vector flow and the remaining vectors are re-clustered according to the current k value. The algorithm reaches a point at which it is no longer possible to cluster the remaining vectors for larger k values, and it stops. At this point, the value of k defines the number of elements in set $\{\boldsymbol{C}(k)\}$.

For the super-resolution application, we usually need several reference images $\boldsymbol{B}_n$. In the case, we get a family of indexed sets $\{\boldsymbol{C}(k)\}_n$.

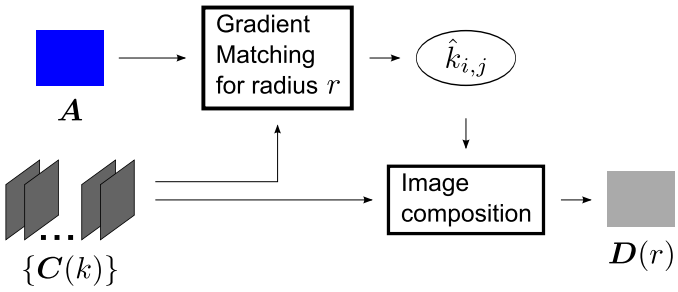

Fig. 3. Image composition through gradient matching.

III. Image construction from gradient matching

In the previous step, we generated a collection of compensated images $\{\boldsymbol{C}(k)\}$. As mentioned before, because of the use of RANSAC with only a few vectors, some of the images may have regions which are too distorted when compared to image $\boldsymbol{A}$. However, these images may still contain useful high-frequency information. For this, we use gradient matching to generate a new set of images

Let again $\boldsymbol{A}$ be the current image we want to reconstruct and $\{\boldsymbol{C}(k)\}$ be the set of compensated images. For a given pixel position (i, j), let $A_{i,j}$ be a pixel in $\boldsymbol{A}$, $C_{i,j}(k)$ be a pixel in $\boldsymbol{C}(k)$ and $\nabla A_{i,j}$ and $\nabla C_{i,j}(k)$ be their gradients, respectively. The best match is the index $\hat{k}_{i,j}$ that satisfies

$$\hat{k}_{i,j} = \underset{k}{\operatorname{argmin}} \|\nabla A_{i,j} - \nabla C_{i,j}(k)\|. \quad (1)$$

By finding these matches, we compose a new image $\boldsymbol{D}$, in which each pixel is given by $D_{i,j} = C_{i,j}(\hat{k}_{i,j})$. In the case of multiple reference images, the gradient matching may be carried using any of the indexed sets $\{\boldsymbol{C}(k)\}_n$ or even their union.

Instead of matching the gradient among pixels as in Eq.(1), it is found [7] that it is often beneficial to include a region R around pixel (i, j) in the match as in Eq.(2):

$$\hat{k}_{i,j} = \underset{k}{\operatorname{argmin}} \sum_{s\in R}\sum_{t\in R} \|\nabla A_{i+s,j+t} - \nabla C_{i+s,j+t}(k)\|, \quad (2)$$

The radius r of region R directly affects the final image result. Because of that, it is possible to build distinct $\boldsymbol{D}(r)$ images. These images are then combined to yield set $\{\boldsymbol{D}(r)\}$. We limit the maximum value of r, and the number of elements of $\{\boldsymbol{D}(r)\}$, to r_{max} so that region R entirely fits inside the smallest RoI found in the clustering step.

IV. Combining image compensation and gradient matching for super-resolution

As an application, the presented techniques can be used to build a codebook for super-resolution in substitution to OBMC, within the method presented in [6]. Our objective is to super-resolve low-resolution image $\boldsymbol{A}$ using high-resolution image $\boldsymbol{B}$. This is achieved by adding some high-frequency information $\boldsymbol{I}^{SR}$ (calculated using both $\boldsymbol{A}$ and $\boldsymbol{B}$) to the upsampled version of $\boldsymbol{A}$, $\boldsymbol{A}^u = u(\boldsymbol{A})$, where $u(.)$ is an interpolation kernel. Thus, we seek $\boldsymbol{A}^{SR} = \boldsymbol{A}^u + \boldsymbol{I}^{SR}$. Our technique must then be extended to calculate $\boldsymbol{I}^{SR}$.

In order to build the mentioned codebook, we need pairs of LR and HR images. We then calculate a LR version $\boldsymbol{B}^l$ of image $\boldsymbol{B}$ by applying a low-pass filter. This filtering process is done by decimation-interpolation operation, i.e., pre-filtering and downsampling $\boldsymbol{B}$ to the size of $\boldsymbol{A}$, followed by upsampling it back to its original size. This process is represented by $\boldsymbol{B}^l = u(d(\boldsymbol{B}))$, where $d(.)$ is the pre-filtering and downsampling kernel in the same scale factor as $u(.)$.

We acquire the vector flow from the matching of features from $\boldsymbol{A}^u$ and $\boldsymbol{B}$. $\boldsymbol{B}^l$ is not used because the resampling process produces artifacts, which may generate spurious features that may degrade feature matching. We now find the compensated images $\boldsymbol{C}(k)$, by warping image $\boldsymbol{B}$ according to the found homographies. For each cluster, we can also apply the homographies to image $\boldsymbol{B}^l$, giving us images $\boldsymbol{C}^l(k)$. Note that $\boldsymbol{C}^l(k) \approx u(d(\boldsymbol{C}(k)))$, except for the high-frequency borders of the patches.

The next step is to use image $\boldsymbol{A}^u$ and the sets $\{\boldsymbol{C}^l(k)\}$ and $\{\boldsymbol{C}(k)\}$ for gradient matching. The matching is then performed between image $\boldsymbol{A}^u$ and images in set $\{\boldsymbol{C}^l(k)\}$, since they are both approximately in the same resolution. The matching will again return $\hat{k}_{i,j}$. With the indexes found for each pixel position, we compose two images $\boldsymbol{D}^l$ and $\boldsymbol{D}$, where pixels in each image are $D^l_{i,j} = C^l_{i,j}(\hat{k}_{i,j})$ and $D_{i,j} = C_{i,j}(\hat{k}_{i,j})$, respectively, where $C^l_{i,j}(k)$ is a pixel in $\boldsymbol{C}^l(k)$ and $C_{i,j}(k)$ is a pixel in $\boldsymbol{C}(k)$. Considering the region R around each pixel position, we have sets $\{\boldsymbol{D}^l(r)\}$ and $\{\boldsymbol{D}(r)\}$. This pair is then used to build the codebook.

The codebook is populated by pairs of images in order to follow steps described in [6]. First, for each index r, the image $\boldsymbol{I}$ containing only high-frequency information is calculated by $\boldsymbol{I}(r)$ = $\boldsymbol{D}(r)$ - $\boldsymbol{D}^l(r)$. The other image in the pair is a LR image. Initially, we used image $\boldsymbol{D}^l(r)$. However, test have shown that the results could be improved by using images that are more similar to $\boldsymbol{A}^u$. Thus, we create new images $\boldsymbol{L}$, calculated for each index r as

$$\boldsymbol{L}(r) = u(d(\boldsymbol{A}^u + \boldsymbol{I}(r))), \quad (3)$$

The final high-frequency information added to image $\boldsymbol{A}^u$, which yields the super-resolved image, is a linear combination of images in set $\{\boldsymbol{I}(r)\}$, calculated in a block-wise fashion. Let $\boldsymbol{A}^u_s$ and $\boldsymbol{L}_s(r)$ be two collocated square blocks in images $\boldsymbol{A}^u$ and $\{\boldsymbol{L}(r)\}$, respectively. We calculate a distortion $\gamma(r) = dist(\boldsymbol{A}^u_s, \boldsymbol{L}_s(r))$ between these two blocks, where $dist$ is any desired distortion metric, like SAD or SSD, for example. Next, we calculate weights $\alpha(r)$, which are inversely proportional to the distortions $\gamma(r)$, given by:

$$\alpha(r) = \left(\frac{1}{\gamma(r)}\right)\left(\sum_{r=1}^{r_{max}} \frac{1}{\gamma(r)}\right)^{-1}. \quad (4)$$

Each $\boldsymbol{A}^u_s$ block is then super-resolved by a linear combination of blocks $\boldsymbol{I}_s(r)$ from $\boldsymbol{I}(r)$, i.e.

TABLE I
PSNR [dB] COMPARISON AMONG SR AND INTERPOLATION METHODS.

Sequence	Bicubic [5]	Lanczos	SR in [15]	MSR [5]	HSR [5]	Our Previous Work [7]	SR in [6]	**Ours**	**Ours** + OBMC
Container	27.9	27.4	30.7	31.9	33.2	35.0	36.0	**36.7**	***37.3***
Hall	29.1	28.2	32.6	37.4	38.0	40.3	41.1	**41.9**	***42.0***
Mobile	22.9	22.8	25.5	24.5	25.5	28.6	27.1	**29.6**	***30.1***
News	29.4	30.1	34.1	31.9	36.1	39.1	38.8	**39.5**	***39.9***
Mobcal	27.7	27.8	29.8	30.9	31.0	36.1	35.0	**38.1**	***38.2***
Shields	31.1	33.1	34.9	31.4	32.7	36.4	36.0	**36.5**	***37.1***

$$\boldsymbol{A}_s^{SR} = \boldsymbol{A}_s^u + \sum_{r=1}^{r_{max}} \alpha(r)\boldsymbol{I}_s(r). \quad (5)$$

The whole $\boldsymbol{A}^{SR}$ is calculated when the previous algorithm is applied to all s blocks in $\boldsymbol{A}^u$.

V. EXPERIMENTAL RESULTS

Our technique is compared to others under the same test conditions they have presented. We super-resolve a low-resolution version (original frame downsampled by a factor of 2) of the 16th frame of a video sequence, using the 1st and 31st frames as reference images. The tests are run on four CIF and two 720p sequences and the results are shown on Table I. All frames used are luminance only.

We compare our results to those directly reported in [5] and [6]. We also compare to the work [15], for which we ran new tests, where we composed 1000 patch-pairs with the reference frames as training sets. The training images where downsized by a factor of two with a bicubic filter, while the features extraction was performed using gradient and Laplacian filters. We also compare to our previous results.

In the entire process, both downsampling and upsampling were performed with a factor of 2 and used the Lancozs-3 filter kernel (using function *imresize* from MATLAB). In the optical flow clustering, the average number of maximum clusters, i.e., the size of sets $\{\boldsymbol{C}(k)\}$, was 17.25 for CIF sequences and 317.75 for 720p sequences. In the gradient matching process, the value of r_{max} varied from 9 to 18, depending on the sequence. In the codebook step, the square block sizes tested were 2×2, 4×4, 8×8 and 16×16. For CIF sequences, blocks of size 4×4 yielded the best results, while size 16×16 yielded the best results for 720p sequences.

In order to show our improvements over OBMC, we compose the codebook with our technique only, shown in column "Ours", in Table I. However, we also show the results for the combination of our codebook combined to that obtained by OBMC, shown in column "Ours+OBMC", in Table I. The overall results show that our technique not only surpasses previously presented results, but also works well along with OBMC, in the mixed-resolution video super-resolution application.

VI. CONCLUSIONS

We have proposed a technique for image composition, applied to mixed-resolution video super-resolution. The super-resolution is achieved with the construction of a codebook from the clustering of an optical flow of matched SIFT features followed by gradient matching. Our results have shown that this technique works very well alongside OBMC, or even replacing it. We also improved our previous work by allowing the algorithm to work without need for manual parameter setting. The average gains over OBMC of our work alone and combined with OBMC are of 1.4dB and 1.8dB, respectively.

This technique can be further improved, as a future work suggestion, in the sense of speeding up the process for larger video frames. Our experiments have shown that a dense optic flow may lead to a large number of cluster, which could turn the algorithm unfeasible for extremely large frame sizes. Any sort of early stop decision could bring good improvements.

REFERENCES

[1] T. S. Huang and R. Y. Tsai, "Multi-frame image restoration and registration," *Adv. Comput. Vis. Image Process.*, vol. 1, no. 2, pp. 317–339, 1984.

[2] W. Freeman, T. Jones, and E. Pasztor, "Example-based super-resolution," *IEEE Comput. Graph. Appl.*, vol. 22, pp. 56–65, March 2002.

[3] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, Anchorage, USA, Jun. 2008, pp. 1–8.

[4] D. Mukherjee, B. Macchiavelo, and R. de Queiroz, "A simple reversed-complexity Wyner-Ziv video coding mode based on a spatial reduction framework," in *Proc. IST/SPIE Symp. on Electronic Imaging, Visual Communications and Image Processing*, San Jose, USA, Jan 2007.

[5] B. Song, S.-C. Jeong, and Y. Choi, "Video super-resolution algorithm using bi-directional overlapped block motion compensation and on-the-fly dictionary training," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 3, pp. 274–285, March 2011.

[6] E. Hung, R. de Queiroz, F. Brandi, K. Oliveira, and D. Mukherjee, "Video super-resolution using codebooks derived from key frames," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 9, pp. 1321–1331, Sep. 2012.

[7] R. Ferreira, E. Hung, and R. de Queiroz, "Video super-resolution based on local invariant features matching," in *Proc. IEEE Intl. Conf. on Image Processing (ICIP'12)*, Orlando, USA, Sep.-Oct. 2012, pp. 877–880.

[8] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Intl. Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Jan 2004.

[9] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia*. New York, NY, USA: John Wiley & Sons, Inc., 2003.

[10] R. Hess, "An open source SIFT library," in *Proc. ACM Multimedia (ACMMM10)*, Firenze, Italy, Oct. 2010.

[11] G. Eibl and N. Brandle, "Evaluation of clustering methods for finding dominant optical flow fields in crowded scenes," in *Proc. IEEE Intl. Conf. on Pattern Recognition (ICPR'08)*. Tampa, FL: IEEE, Dec 2008.

[12] J. H. Ward, Jr., "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, p. 236244, 1963.

[13] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[14] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. NJ, USA: Prentice-Hall, 2006.

[15] R. Zeyde, M. Elad, and M. Protter, "On single image scaleup using sparse-representations," in *Curves & Surfaces*, June 2010.