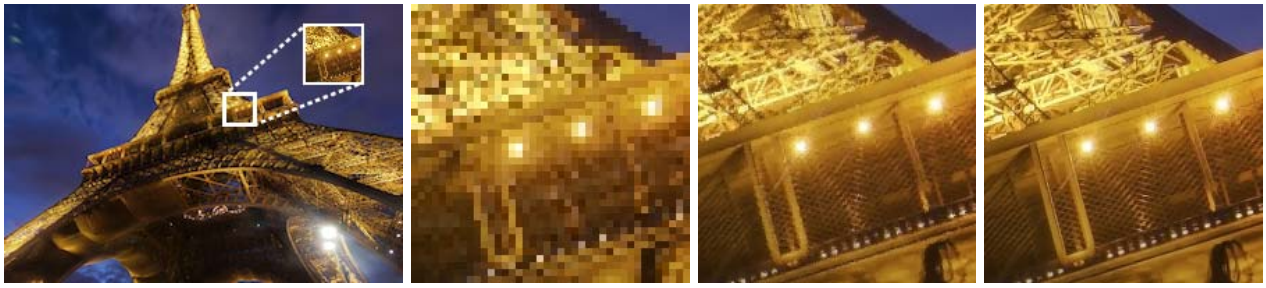
Fast and Effective Geometric K-Nearest Neighbors Multi-Frame Super-Resolution

Hilario Seibel Junior*†, Siome Goldenstein† and Anderson Rocha†
*Instituto Federal do Espírito Santo. Serra, Espírito Santo, Brazil. hsjunior@ifes.edu.br
†Institute of Computing. University of Campinas. Campinas, São Paulo, Brazil. {siome,anderson.rocha}@ic.unicamp.br



(a) (b) (c) (d)

Fig. 1. Super-resolution *versus* interpolation: (a) piece of an Eiffel Tower's low-resolution image; (b) same piece zoomed by a factor of 5 using a simple nearest neighbor interpolation; (c) the piece super-resolved by our algorithm using 35 low-resolution images as input and each pixel's nearest neighbor; and (d) equivalent piece in an HR target image.

***Abstract*—Multi-frame super-resolution is possible when there is motion and non-redundant information from a sequence of low-resolution input images. Remote sensors, surveillance videos and modern mobile phones are examples of devices able to easily gather multiple images of a same scene. However, combining a large number of frames into a higher resolution image may not be computationally feasible by complex super-resolution techniques. We discuss herein a set of simple and effective high-performance algorithms to fastly super-resolve several low-resolution images in an always-on low-power environment, with possible applications in mobile computing, forensics, and biometrics. The algorithms rely on geometric k-nearest neighbors to decide which information to consider in each high-resolution pixel, have a low memory footprint and run in linear time as we increase the number of low-resolution input images. Finally, we suggest a minimum number of input images for multi-frame super-resolution, considering that we expect a good response as fast as possible.**

***Keywords*-super-resolution; geometric k-NN; multi-frame; burst; mobile devices;**

I. Introduction

The basic premise for a multi-frame super-resolution (or SR) algorithm is the availability of multiple images, with small relative motion, captured from a scene. Subpixel motion provides new details for the super-resolved image that would not be found using a simple interpolation. Although such a requirement does not enable us to use SR in some situations, there are several cases wherein we can easily capture a sequence of images and subsequently super-resolve.

Recent mobile phones typically can take dozens of photos per second that could be combined into a higher resolution image as if it had been taken by a more powerful camera. The *Timeshift Burst* application available for *Android 4.2* can capture up to 30 images in a second, while *iPhone 6* can gather up to 10 pictures during this time. Sequences of frames in surveillance videos, usually with very low resolution, can also be combined to produce an image with more details. In such videos, many objects such as badges and license plates could help identifying a suspect, aiding forensic analysts in understanding a given event. Moreover, it may be very expensive to replace obsolete camera sensors in satellites, but a super-resolution technique could be an inexpensive solution via software to overcome this limitation.

The aforementioned applications often require quick responses. A person who takes a dozen photos in one second in a mobile device may not be willing to wait for minutes until a high-resolution image is generated. In addition, the application should have low memory footprint, because the person may simultaneously receive and make voice calls, send messages, take, receive and send photos and videos, listen to music and/or play games while waiting for the reconstructed image. In the forensic scenario, a crime suspect can escape while the forensic analyst is waiting for his identification if the algorithm takes hours to produce the output. Finally, recognizing, in time, an enemy aircraft visualized by satellite images can avoid an accident or even a terrorist attack.

For all these reasons, we investigate in this work multi-frame super-resolution algorithms in the spatial domain that provide effective results but still in a very fast response time. The methods use geometric k-nearest neighbors to decide which is/are the best candidate(s) to be placed (or combined)

1530-1834/15 $31.00 © 2015 IEEE
DOI 10.1109/SIBGRAPI.2015.47

in each pixel of the high-resolution image.

Contributions: We investigate five variations of a fast and simple, yet effective, algorithm for multi-frame super-resolution using geometric k-nearest neighbors to choose which details will appear in the resulting high-resolution picture. In our experiments, we exhaustively evaluate the algorithm using dozens of sequences as input, each one with up to 100 low-resolution images. In about one minute, for example, the solution can combine up to 30 images with $1,200 \times 800$ pixels into a $5\times$ higher resolution image with $6,000 \times 4,000$ pixels (higher than the recent mobile phones resolution). The results are promising both in qualitative and quantitative terms. The algorithms can be applied to always-on low-power environments, as they are fast and have a low memory footprint. Finally, we also discuss what should be a "magic number" of input images to be used in a super-resolution problem in this setup for a good visual quality and a small execution time.

II. Related work

The first multi-frame super-resolution algorithm was proposed by Tsai and Huang [1] in the *frequency domain*. The algorithm works on low-resolution (LR) images acquired by the Landsat 4 satellite, which produces a set of similar but globally translated images of the same area. Those shifts between consecutive images are taken into account by the shifting property of the Fourier transformation. Most frequency domain methods have problems with real-world applications, since they accept only a global displacement between the images. Therefore, the majority of SR algorithms have been developed in the spatial domain [2].

Spatial domain methods are based on interpolation over the LR images. A single image interpolation does not handle the SR problem well, since it may not produce those high-frequency components that were lost during the image acquisition process. But in multi-frame approaches, each LR observation provides an amount of additional information about the scene [3]. Such methods usually have a registration step, for aligning the LR images, and a reconstruction step, for producing the higher resolution image. Optionally, they can also include a deblurring step for enhancing the reconstructed high-resolution (HR) image produced in the second step.

Iterative Back Projection (IBP) algorithms [4], [5] are among the first methods for spatial-based SR. In these cases, each HR image pixel is estimated iteratively as a sum of different projections of the same LR image area, determined by the image blurring and displacement. IBP is simple, but might not yield a unique solution due to the ill-posed nature of the SR problem. Zomet et al. [6] proposed the *Robust Super Resolution*, a more robust version of the IBP algorithm using the median instead the mean. Papoulis [7] and Gerchberg [8], independently, demonstrated the method of iterative signal extrapolation. The classical *Papoulis-Gerchberg* (PG) method may not deliver good results in presence of blur and noise in the LR image. Vanderwalle et al. [9] modified PG to obtain SR images from multiple LR registered images.

Another group of iterative methods are those based on the concept of *Projection onto Convex Sets* [10]–[12]. These algorithms define an implicit cost function for solving the SR problem. In the POCS method, it is assumed that each LR image imposes an a priori knowledge on the final solution. POCS also does not give a unique solution and suffers from high computational costs.

Finally, one trend currently explored in the literature is the *Direct Methods* [13], [14]. Such methods simply align and scale the LR images to an HR grid, and then choose a filter to combine the LR pixels. Different filters can be used, such as mean and median filters [15], Adaboost classifier [16], and SVD-based filters [17]. In the *Structure-Adaptive Normalized Convolution* [18], the filter is an adaptive normalized averaging. Direct methods have been shown to be faster than IBP [2]. Recently, a set of non-parametric direct SR methods [19] combine the two steps of motion estimation and fusion. Such methods are shown to work well with video sequences.

Kanumuri et al. [20] proposed a SR reconstruction of mobile video using warped transforms and adaptive thresholding, but restricts itself to operate on single frames (just an interpolation problem). Shen and Xue [21], Chu [22] and Amanatiadis et al. [23] recently proposed SR solutions for mobile videos. However, as they are example-based solutions, they require a training process.

A good and up-to-date SR survey can be found in [2], which highlights that the state of the art for super-resolution algorithms is highly dependent on the application. This is mainly due to the different constraints that are imposed on the problem under different setups.

III. Geometric k-nearest neighbors multi-frame super-resolution

We investigate five variations of an algorithm which rely on geometric k-nearest neighbors in order to super-resolve a sequence of images in a small execution time. Such algorithm works in the spatial domain and is divided into two steps: registration and reconstruction.

A. Registration Step

To combine multiple images of a same scene into a higher resolution image, first we need to align the input frames with subpixel accuracy. Given a set of n LR images $I_1, I_2, \dots I_n$, we find keypoints and descriptors in all images and then match the keypoints in I_1 to the keypoints of I_k for each $k \in [2, n]$. The best matches between I_1 and each I_k are used to estimate a perspective transformation M_k. Thus, we can write I_k with respect to I_1, as follows:

$$\forall k \in [2, n] : I_k = M_k I_1 \tag{1}$$

We do design a new registration technique herein. Instead, we have evaluated three feature detectors to find the mentioned keypoints and descriptors: SIFT (Scale-invariant feature transform) [24], [25], SURF (Speeded Up Robust Features) [26] and ORB (Oriented FAST and Rotated BRIEF, purportedly a fast and efficient alternative to SIFT) [27]. The points are matched using k-NN and Flann [28], [29].

B. Reconstruction Step

To fastly reconstruct an HR image I_{HR} from a sequence of input images, we first focused on a very intuitive algorithm, as Figure 2 illustrates. Black squares and red dots in Figure 2a represent pixels from two LR images I_1 and I_2, aligned in the registration step. Then, we create a grid containing the HR pixels, as Figure 2b shows. In this example, blue triangles and black squares compose the grid (blue triangles are new pixels that will appear in the resulting image, and black squares are pixels in the HR image that coincide with I_1 pixels). In Figure 2c, we choose the nearest neighbor for each I_{HR} pixel among all pixels in I_1 and I_2. Finally, the final SR image is composed by pixels from both I_1 and I_2 in Figure 2d.

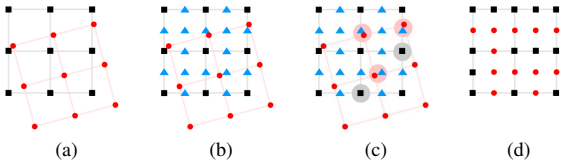


Fig. 2. (a) Two LR images already aligned in the registration step. (b) New pixels that will appear in the HR image. (c) For each pixel in the HR image, we find its geometric nearest neighbor among all pixels in the LR input sequence. (d) The HR image, composed by pixels from both input images.

As there are different possible policies to choose and combine nearest neighbors geometrically, we investigate five variations of this algorithm, aiming at improving the quality of the HR image. Such variations differ from each other in the choice of the best candidates for each pixel in the HR grid.

Let p be a pixel in the HR image and q_k be the nearest pixel from p in each LR image I_k. In the first algorithm variation, we chose each p in the HR image exactly as in Figure 2, using its closest neighbor among all possible values of q_k. In other words, we use k-nearest neighbors with $k = 1$ to find q_k from all LR images, and then another 1-NN to find the closest q_k with respect to p. We further refer to this variation as GSR_1 (Geometric k-NN Super-Resolution).

In the second variation (GSR_2), we first use 1-NN to find all q_k from the input sequence, and then a 3-NN to get three best values of q_k. The resulting pixel p is simply an average among its three closest neighbors. Similarly, GSR_3 finds the three best values of q_k, but combines them as a weighted average. We use static weights (60% for the nearest one and 20% for each other), but could also set them to be an inverse proportion to their distances.

Both algorithms GSR_2 and GSR_3 can produce bad results if the three closest values of q_k do not form a region which includes p. Figure 3 illustrates an example of this situation: it is possible to find a straight line that passes through p such that all the three nearest points q_k are on the same side of the line. In this case, the second and third nearest q_k may not add relevant information to p. In [30], the authors purpose a Delaunay triangulation for all available points and combine the vertices of the triangle to which p belongs. According to [31], the triangulation can be build in $\mathcal{O}(n \log n)$, while the simplest solution using a k-NN search is $\mathcal{O}(n)$. We did not implement this option to avoid increasing the algorithm runtime for now, but we shall investigate it further in the near future.

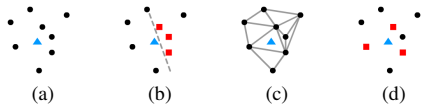


Fig. 3. In (a), the blue triangle is the target pixel p and each black dot is its nearest pixel q_k from an LR image I_k. In (b), 3-NN chooses the red pixels to be combined, but they may not be a good choice because they do not form a region to which p belongs. In (c), a triangulation is calculated among all black pixels. In (d), p belongs to the triangle formed by the red pixels.

In GSR_4, p is the average among all q_k (the nearest pixel from p in each LR image I_k). If we have 30 input images, for example, q_k is the nearest pixel from p in each of the 30 LR images and p is the average among all of them.

Finally, GSR_5 sets a maximum radius distance with respect to p, which is calculated as the weighted average among all q_k within this circular region (as in GSR_3, the weights are inversely proportional to their distances). Figure 4 illustrates how each variation chooses among all available q_k.

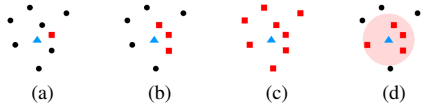


Fig. 4. Variations of the investigated algorithm: (a) The nearest q_k in GSR_1; (b) 3-NN in both GSR_2 and GSR_3; (c) all available q_k in GSR_4; and (d) all q_k inside a circular region in GSR_5.

It is important to mention that we do not apply any transformation to the LR images, because it would cause loss of information. Instead, we use the matrices M_k, calculated in the registration step, to know exactly the xy position where each q_k is with respect to the HR grid. Moreover, each input image is individually processed, so there will not be more than one LR image in the memory at the same time. The memory consumption of all five variations is $\mathcal{O}(w \times h)$ for an HR image of $w \times h$ pixels. Basically, the algorithms need to store, in memory, the following information during each step: (1) the $w \times h$ pixels p in the HR grid and the distances to their nearest pixels q_k (GSR_4 and GSR_5 do not store such distances); (2) the $\frac{w}{r} \times \frac{h}{r}$ pixels of the current LR image, for a resizing factor of r; and (3) the positions of the $\frac{w}{r} \times \frac{h}{r}$ LR pixels with respect to the HR grid.

IV. Experimental Methodology

For a quantitative validation of a SR algorithm, it is a standard practice in the literature to choose an HR target image and generate a series of LR versions of it. Such original image can be used as a target for the HR result, created using the pool of LR images as input. The most similar, according to a given similarity measure, the HR image is to the original one, the most accurate the super-resolution (a similarity metric is

described in Section IV-A). This is a fictitious situation, but allows us to quantitatively compare algorithms.

For a qualitative evaluation, as there is no target image in most applications, the results from one algorithm often are visually compared to images generated by other techniques. As most works in the literature, we use both qualitative and quantitative validation approaches to evaluate the discussed super-resolution techniques. In a first group of experiments, we create a dataset to quantitatively validate the five algorithms. We introduce the dataset in Section IV-B. In a second group of experiments, we perform qualitative evaluation of results using real photos taken by a mobile phone in burst mode. We present the experiments in Section V. All experiments were performed on an Intel i5 2.53GHz processor with 4GB of RAM.

A. Validation metrics

There are three standard metrics to evaluate the similarity between two images: Mean Squared Error (MSE), Peak signal-to-noise ratio (PSNR) and Structural Similarity Index (SSIM) [32]. We used these three metrics, but we only show the SSIM results herein due to space constraints. The structural similarity is designed to improve on traditional methods like PSNR and MSE, and its results are more similar to human perception [33]. It attempts to measure the change in luminance (modelled as pixel intensity), contrast (variance between the reference and distorted image), and structure (cross-correlation between the two images) in an image:

$$SSIM(x,y) = \frac{(2\mu_x \mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \quad (2)$$

where μ_x and μ_y are, respectively, the local sample means of x and y, σ_x and σ_y are, respectively, the local sample standard deviations of x and y, and σ_{xy} is the sample cross-correlation of x and y after removing their means. The items C_1 and C_2 are small positive constants that stabilize each term, so that near-zero sample means, variances, or correlations do not lead to numerical instability. The SSIM index is computed locally within a sliding window that moves pixel-by-pixel across the image, resulting in an SSIM map. The SSIM score of the entire image is then computed by simply averaging the SSIM values across the generated map. The maximum value $SSIM = 1$ happens if and only if the two images are equal. The higher the SSIM value, the more similar one image to the other.

B. Dataset

In the first group of experiments, we have a set of target HR images and we need to simulate sequences of LR images to be input for the super-resolution algorithms. We collected 30 free[1] high-resolution images with resolutions varying from approximately $2,400 \times 1,600$ pixels to $6,000 \times 4,000$ pixels to be target images. Every HR image generated 200 random LR images (with size 20% of the original image) in a preprocessing step, which we shall discuss next.

[1]All pictures were collected from *http://www.gratisography.com/*. The images are free of Copyright Restrictions, under "Creative Commons Zero" license, and dedicated to public domain.

We start creating n random rigid transformations that will be applied to an HR image I_{HR} in order to generate n low-resolution versions of the image. Each transformation is composed by a scaling matrix S, a rotation matrix R around the center of the image, and a translation matrix T:

$$I_{LR} = (TRS)I_{HR}. \quad (3)$$

Additionally, we are able to calculate what should be the "correct alignment" among the LR generated images. Let T_1, R_1, and S_1 be the translation, rotation, and scaling matrices mapping an HR image I_{HR} onto the first LR image I_1. We can write I_{HR} with respect to I_1, as:

$$I_{HR} = (T_1 R_1 S_1)^{-1} I_1. \quad (4)$$

Now let T_k, R_k and S_k be the translation, rotation, and scaling matrices mapping the HR image I_{HR} to an arbitrary LR image I_k ($k \in [2, n]$):

$$I_k = (T_k R_k S_k) I_{HR},$$
$$I_k = (T_k R_k S_k)(T_1 R_1 S_1)^{-1} I_1. \quad \text{(from Equation 4)}$$

Finally, let N_k be the matrix $(T_k R_k S_k)(T_1 R_1 S_1)^{-1}$:

$$I_k = N_k I_1. \quad (5)$$

From Equation 5, we have transformation matrices N_k to describe any LR image I_k ($k \in [2, n]$) with respect to I_1. The matrices N_k are exactly the transformations M_k that should be found in the registration step of the super-resolution algorithms. The more similar a matrix M_k is with respect to N_k, the more accurate will be the super-resolved image.

It is important to mention that such "correct alignment" may not be available in real applications, but it is an important aid for evaluating the registration results. When this step is executed, the target is to achieve an HR image as similar as possible to the original image that generated the input sequence. In real cases, we may estimate perspective transformations instead of rigid transformations in the registration step (sequences of real photos may differ not only by rotations and translations).

V. Experiments and results

We divide the experiments into two groups: (1) *Experiment #1* uses the dataset described in Section IV-B and the target images allow us to perform a quantitative evaluation of the algorithms; and (2) *Experiment #2* uses real input images taken by mobile phones for a qualitative evaluation.

A. Experiment #1 (quantitative evaluation)

Here, we perform the following evaluations:

- *Round #1*: Comparison among the feature detectors SIFT, SURF, and ORB. **Objective:** Point out out the most appropriate registration algorithm for our purposes;
- *Round #2*: Comparison among the reconstruction algorithms GSR_1, GSR_2, GSR_3, GSR_4, and GSR_5. We also compare them to simple interpolation algorithms. **Objective:** To find the most efficient reconstruction algorithm with the best SSIM values.

- *Round #3*: Comparison among our best SR algorithm of *Round #2* and other reconstruction techniques available in the literature. **Objective:** To put the designed solution in perspective with respect to existing counterparts in terms of efficiency and effectiveness.

The reconstructed images are $5\times$ higher than the LR images in all rounds, so the HR size is the same as the target image. The simulations are repeated at least four times for each target image, because the LR images are randomly chosen among the 200 LR images in the dataset for each target case. The five reconstruction algorithms are always performed using from 2 to 100 LR images. The results in charts are always showed as a function of the number of images in the input sequence.

Experiment #1 – Round #1: For a fair comparison among the three feature detectors (SIFT, SURF, and ORB), we choose only one algorithm for the reconstruction step (GSR_1). Hence, different SSIM values in the results are due to the registration step. First, we show a visual comparison among SIFT, SURF, ORB, and the correct registration in Figure 5, using 25 LR images. For a better visualization of the reconstructed details, Figure 5 shows only a piece of the HR images (all images are available on supplementary material).

(a) NN int., SSIM=0.35 (b) SIFT, SSIM=0.89 (c) SURF, SSIM=0.86

(d) ORB, SSIM=0.89 (e) Cor. reg., SSIM=0.9 (f) Target image

Fig. 5. Piece of an image (a) zoomed using nearest neighbor interpolation. The same piece reconstructed with 25 LR images using GSR_1 and (b) SIFT, (c) SURF, (d) ORB and (e) the correct registration. (f) The target image.

The HR images generated by GSR_1 using SIFT, SURF, ORB, and the correct alignment in the registration step are visually similar to the original image in Figure 5, and their SSIM values are much higher than that of simple interpolation. In Figure 1 (enticing Figure of the paper, page 1), we can also visually compare a GSR_1 result using SIFT (with 35 input images) to a nearest neighbor interpolation.

Figure 6 summarizes a quantitative evaluation of the registration results for all target images in the dataset. Each curve in Figure 6 is the mean out of the results of four simulations for each of the 30 HR images from the dataset. We can see that SIFT resulted in better SSIM values, but ORB results are almost equal to SIFT results. The chart in Figure 7 shows that ORB's runtime is significantly lower than SURF's and SIFT's. Therefore, we can conclude from this experiment that ORB is the appropriate registration method for our purposes, generating good SSIM values in a small runtime.

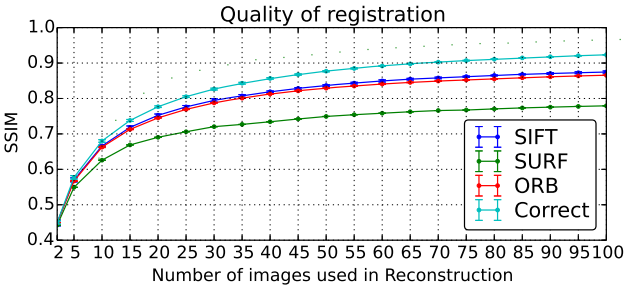


Fig. 6. SSIM values using different algorithms in the registration step.

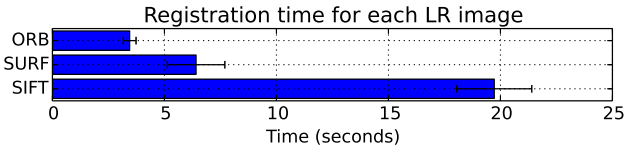


Fig. 7. Registration runtime for each LR image.

Experiment #1 – Round #2: Now, we evaluate the five variations of the algorithm, using only the correct alignment. The correct registration has been chosen here because we can calculate the best possible results for each reconstruction algorithm, and bad results are not resultant of the feature detector. Figure 8 depicts super-resolved images using each algorithm with 35 LR images in the input sequence. Only in GSR_4, we can easily visualize a worst result. Indeed, the other four variations have similar SSIM values, and no algorithm had better results than GSR_1, for this particular example.

Figure 9 summarizes the comparison among the algorithms (each curve is the average of four simulations to each input sequence from all target images in the dataset). The chart also contains the SSIM for simple resizing using Nearest Neighbor, Bilinear, Bicubic, and Lanczos interpolations (their values along the X-axis do not change). GSR_1 had the better results, and all algorithms outperformed the simple resizing, even using only two LR images. GSR_3 is the algorithm which results are most similar, but not better, to GSR_1.

Finally, the chart in Figure 10 shows that the reconstruction step runs in linear time as we increase the number of input images. GSR_1 and GSR_4 are the fastest algorithms. As GSR_1 also produces the best SSIM values (see Figure 9), we can conclude, from this experiment, that such algorithm is an appropriate reconstruction method for our purposes, generating good SSIM values in small runtime. Combining ORB (the fastest feature detector in our experiments) and GSR_1, we spend 63% of the super-resolution time during the registration step and 37% during the reconstruction step.

(a) NN int., SSIM=0.20 (b) GSR_1, SSIM=0.82 (c) GSR_2, SSIM=0.78

(d) GSR_3, SSIM=0.80 (e) GSR_4, SSIM=0.38 (f) GSR_5, SSIM=0.78

Fig. 8. (a) Piece of an LR image zoomed using nearest neighbor interpolation, and reconstruction of 35 LR images with correct registration calculated by (b) GSR_1, (c) GSR_2, (d) GSR_3, (e) GSR_4, and (f) GSR_5.

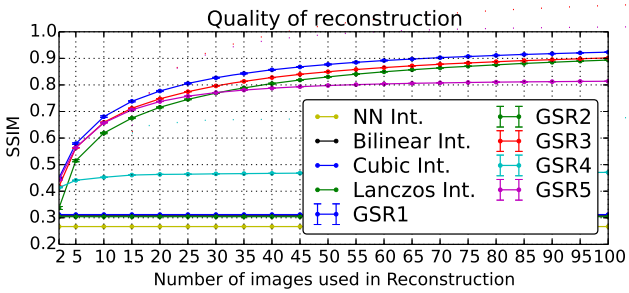


Fig. 9. Quality of different reconstruction algorithms as function of the number of LR input images.

Experiment #1 – Round #3: Finally, we compare GSR_1 to other reconstruction algorithms available in the literature, using the implementations in MATLAB® from [34]: An algorithm inspired in the works of Papoulis [7] and Gerchberg [8] (PG); IBP [4], [5]; Robust Super-Resolution (RS) [6]; $POCS$ [10]–[12]; and Structure-Adaptive Normalized Convolution (NC) [18] (see Section II for details about them). Here, we use only up to 35 input images and six target images from

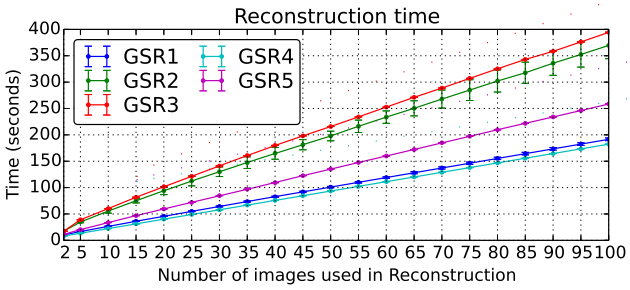


Fig. 10. Reconstruction runtime.

the dataset, due to the long runtime spent by classic algorithms.

For a fair comparison, we executed all algorithms using the correct alignment in the registration step. All classic algorithms, except NC, failed to super-resolve more than two input images. The chart, in Figure 11, summarizes the comparison among GSR_1, PG, IBP, RS, $POCS$, and NC.

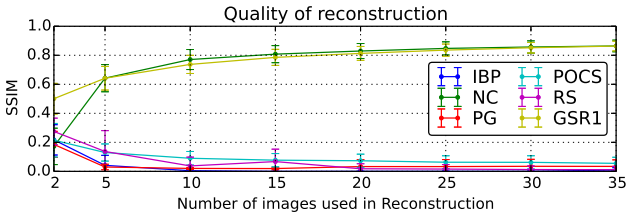


Fig. 11. Quality of different reconstruction algorithms as a function of the number of LR input images.

Figure 12 shows a comparison between the runtime of the best investigated reconstruction algorithm (GSR_1) and the best classic one in the experiments (NC). Although GSR_1 and NC present similar SSIM values as we increase the number of input images (Figure 11), we can conclude, from Figure 12, that GSR_1 is more advantageous due to its low execution time (two orders of magnitude faster).

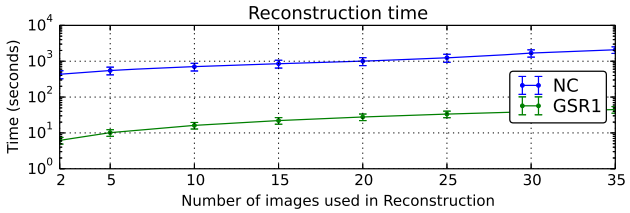


Fig. 12. Reconstruction runtime (logarithmic scale for viewing purposes).

B. Experiment #2 (qualitative evaluation)

We now evaluate GSR_1 with a series of real input images, taken by an iPhone 5 in burst mode. Each input sequence was gathered in a few seconds, with very small relative motion, and there were only static objects in the scenes. In the *Experiment #1*, we generated sequences of LR images applying rotations and translations to the scaled HR images. However, photos taken by a mobile phone may differ from each other by perspective transformations. In *Experiment #2*, we first investigate if GSR_1 can handle perspective transformations. In a first round, we use 35 images with the maximum resolution available in the device to generate 35 LR versions of these images (one per image), with size 20% of the original one.

Unlike *Experiment #1*, now each LR image comes from a different HR image. Hence, any of the 35 original images can be a target for the SR. We then super-resolve the LR images and create a $5\times$ higher image. Figure 13 shows small pieces of the result. GSR_1 takes 80 seconds to super-resolve the 35 LR images, while NC spends $4,860$ seconds. In these

cases, the registration time was not considered in this reported time. One possible reason for the bad results of NC (the best counterpart evaluated in Section V-A) is that it considers only rigid transformations. However, GSR_1 works correctly with the sequences of real photos (see Figure 13).

(a) Nearest neighbor interpolation

(b) SIFT in registration step and NC in reconstruction step

(c) SIFT in registration step and GSR_1 in reconstruction step

(d) Target image

Fig. 13. LR versions of 35 real images taken by a mobile phone generate a $5\times$ higher image.

Finally, we super-resolve the 35 HR original images, in a second round, to generate an even higher image ($12,240 \times 16,320$ pixels). We refer to such final image as a *super high-resolution (SHR) image*. Figure 14 depicts small pieces of the result. The input sequence has been aligned with ORB and SIFT. Here, we do not have a quantitative comparison among the algorithms because there is no HR original image to be the reconstruction target and to calculate a correct registration.

VI. Suggestion of an Ideal Number of LR Input Images

After developing and evaluating different algorithms, we now discuss what could be an ideal number of input images to be used by a multi-frame super-resolution reconstruction with quality and efficiency constraints. Figure 15 illustrates an example of pixels in an LR image with respect to the grid of a $3\times$ higher image (each red pixel may contribute to a region of 3×3 pixels in the HR grid). It means that each region

(a) NN interpol. (b) SIFT + GSR_1 (c) ORB + GSR_1

Fig. 14. 35 HR images ($2,448 \times 3,264$ pixels) taken by a mobile phone generate an SHR image $5\times$ higher ($12,240 \times 16,320$ pixels).

needs 3^2 informations from the input images to generate the HR one. Therefore, we may intuitively suggest that we need at least m^2 LR frames for a super-resolved image by m.

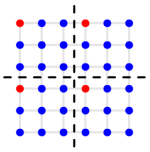


Fig. 15. LR pixels (red dots) distributed throughout an HR grid.

In *Experiment #1*, we evaluated five variations of a geometric k-nearest neighbor SR algorithm using up to 100 LR images in order to generate a $5\times$ higher resolution image. Figures 6 and 9 show that for more than 5^2 LR images, we have a very small improvement in the quality of generated HR images. Their curves grow quickly until 25 LR images, and then they become stable as we enlarge the number of images. Therefore, increasing the sequence input length after m^2 may not compensate the consequential increase in execution time. For those reasons, m^2 may be a good number when choosing the length of an input sequence: any number close to that may generate a good solution in a small response time. But if a large dataset is available, with more than m^2 input images, the reconstruction can still be calculated in an acceptable time.

VII. Conclusion

Using geometric k-nearest neighbors to combine pixels from several images into a higher resolution image is an interesting topic: our experiments show that it is feasible and produces good results in qualitative and quantitative terms when compared to other methods. In addition, these algorithms are fast and present low memory footprint, so they can be effectively applied to always-on low-power environments (such as the ones involving mobile computing), forensics, and biometrics.

GSR_1 may have achieved the best results because the nearest neighbor q_k from a pixel p in the HR grid seems to have enough information about p, due to the number of input images considered. The more information we add from other pixels to p in the other variations of the algorithm, the more we decrease the quality of results. The small number of HR pixels that do not have a good candidate q_k might not compensate finding other neighbors in the remaining image in the pool.

Registration plays an important role in a multi-frame SR. Our experiments showed that classic feature detectors, such as

SIFT and ORB, can perform a good alignment. Although SIFT outperformed ORB and SURF, ORB also resulted in good SSIM values, but within an acceptable time for our purposes. Therefore, ORB seems to be more appropriate than SIFT for an always-on low-power environment.

As a suggestion, any number near m^2 may be a good amount of input images to reconstruct an $m\times$ higher resolution image with an acceptable cost and good quality.

Moreover, we highlight that the discussed algorithms can become even faster: for now, they are implemented in Python (we may achieve faster runtime if using C) and they still have no parallelism. For example, they can take advantage of multiple cores in recent mobile phones.

For future work, we may study other nearest neighbors approaches in order to combine the LR pixels, compare the algorithms to the latest methods in the literature (putting all of them in this context of limited memory and battery), parallelize GSR_1 operations, and adapt it to run on more recent mobile phones. We may also investigate if we can use the accelerometer and gyroscope from such devices to calculate (or help to calculate) the motion between two consecutive images. It could aid the registration step.

ACKNOWLEDGMENT

The research for this paper was financially supported by the Coordination for the Improvement of Higher Education Personnel (CAPES) through PhD scholarship and the DeepEyes project; the National Council for Scientific and Technological Development (CNPq) through Grants #454082/2014-2, #308882/2013-0, #304352/2012-8 and #477662/2013-7; and the Microsoft Research.

Finally, it is worth mentioning that a part of the created dataset and associated transformation matrices are freely available on http://dx.doi.org/10.6084/m9.figshare.1453209 .

REFERENCES

[1] R. Y. Tsai and T. S. Huang, "Multiframe image restoration and registration," *Advances in Computer Vision and Image Processing : Image Reconstruction from Incomplete Observations*, 1984.

[2] K. Nasrollahi and T. B. Moeslund, "Super-resolution: a comprehensive survey," vol. 25, pp. 1423–1468, 2014.

[3] J. Tian and K.-K. Ma, "A survey on super-resolution imaging," *Signal, Image and Video Processing*, vol. 5, no. 3, pp. 329–342, 2011.

[4] M. Irani and S. Peleg, "Super resolution from image sequences," *Intl. Conf. on Pattern Recognition*, vol. C, no. 90, pp. 115–120, 1990.

[5] ——, "Improving resolution by image registration," *CVGIP: Graph. Models Image Process.*, vol. 53, no. 3, pp. 231–239, apr 1991.

[6] A. Zomet, A. Rav-Acha, and S. Peleg, "Robust super-resolution," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2001, pp. 645–650.

[7] A. Papoulis, "A new algorithm in spectral analysis and band-limited extrapolation," *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 22, pp. 735–742, 1975.

[8] R. W. Gerchberg, "Super-resolution through Error Energy Reduction," *Optica Acta*, vol. 21, pp. 709–720, Sep. 1974.

[9] P. Vandewalle, S. Ssstrunk, and M. Vetterli, "Superresolution images reconstructed from aliased images," in *SPIE/IS&T Visual Communications and Image Processing Conference*, vol. 5150, 2003, pp. 1398–1405.

[10] H. Stark and P. Oskoui, "High-resolution image recovery from image-plane arrays, using convex projections," *J. Opt. Soc. Am. A*, vol. 6, no. 11, pp. 1715+, Nov. 1989.

[11] M. Elad and A. Feuer, "Restoration of a single superresolution image from several blurred, noisy, and undersampled measured images," *Trans. Img. Proc.*, vol. 6, no. 12, pp. 1646–1658, Dec. 1997.

[12] M. Zibetti, *Super-resolução simultânea para sequência de imagens*. Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia Elétrica., 2007.

[13] M.-C. Chiang and T. E. Boult, "Efficient image warping and super-resolution," in *IEEE Workshop on Applications of Computer Vision*, ser. WACV '96. Washington, DC, USA: IEEE, 1996, pp. 56–.

[14] ——, "Local blur estimation and super-resolution," in *Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE, 1997, pp. 821–.

[15] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multiframe super resolution," *Trans. Img. Proc.*, vol. 13, no. 10, pp. 1327–1344, Oct. 2004.

[16] K. Simonyan, S. Grishin, D. Vatolin, and D. Popov, "Fast video super-resolution via classification." in *IEEE Intl. Conf. on Image Processing*. IEEE, 2008, pp. 349–352.

[17] H. Nasir, V. Stanković, and S. Marshall, "Singular value decomposition based fusion for super-resolution image reconstruction," *Image Commun.*, vol. 27, no. 2, pp. 180–191, feb 2012.

[18] T. Q. Pham, L. J. van Vliet, and K. Schutte, "Robust fusion of irregularly sampled data using adaptive normalized convolution," *EURASIP Journal on Appl. Signal Proc.*, vol. 2006, pp. 236–236, Jan. 2006.

[19] M. Protter and M. Elad, "Super resolutionwith probabilistic motion estimation," *Trans. Img. Proc.*, vol. 18, no. 8, Aug. 2009.

[20] E. Kanumuri, O. G. Guleryuz, and M. R. Civanlar, "Fast super-resolution reconstructions of mobile video using warped transforms and adaptive thresholding," in *Proc. SPIE Conf. on Applications of Digital Image Processing XXX*, 2007.

[21] M. Shen and P. Xue, "Low-power video acquisition with super-resolution reconstruction for mobile devices," *IEEE Transaction on Consumer Electronics*, vol. 56, no. 4, pp. 2520–2528, November 2010.

[22] C.-H. Chu, "Super-resolution image reconstruction for mobile devices," *Multimedia Systems*, vol. 19, no. 4, pp. 315–337, 2013.

[23] A. Amanatiadis, L. Bampis, and A. Gasteratos, "Accelerating image super-resolution regression by a hybrid implementation in mobile devices," in *Intl. Conf. on Consumer Electronics*, Jan 2014, pp. 335–336.

[24] D. Lowe, "Object recognition from local scale-invariant features," in *IEEE Intl. Conf. on Computer Vision, 1999.*, vol. 2, 1999, pp. 1150–1157 vol.2.

[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[26] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[27] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *IEEE Intl. Conf. on Computer Vision*, Nov 2011, pp. 2564–2571.

[28] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration." in *VISAPP*, A. Ranchordas and H. Arajo, Eds., 2009, pp. 331–340.

[29] K. Hajebi, Y. Abbasi-Yadkori, H. Shahbazi, and H. Zhang, "Fast approximate nearest-neighbor search with k-nearest neighbor graph," in *Intl. Joint Conf. on Artificial Intelligence - Volume Volume Two*, 2011, pp. 1312–1317.

[30] S. Lertrattanapanich and N. Bose, "High resolution image formation from low resolution frames using delaunay triangulation," *IEEE Trans. on Image Processing*, vol. 11, no. 12, pp. 1427–1441, Dec 2002.

[31] L. P. Chew, "Constrained delaunay triangulations," in *Third Annual Symposium on Computational Geometry*. New York, NY, USA: ACM, 1987, pp. 215–222.

[32] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[33] Z. Wang and A. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, Jan 2009.

[34] P. Vandewalle, S. Süsstrunk, and M. Vetterli, "A frequency domain approach to registration of aliased images with application to super-resolution," *EURASIP Journal on Appl. Signal Proc.*, vol. 2006, pp. 1–14, March 2006.