Super-resolution of faces using the epipolar constraint

Richard J.M. den Hollander, Dirk-Jan J. de Lange and Klamer Schutte

TNO Defence, Security and Safety
Oude Waalsdorperweg 63
P.O. Box 96864, 2509 JG Den Haag, The Netherlands
richard.denhollander@tno.nl

Abstract

In this paper we present a super-resolution scheme specifically designed for faces. First, a face detector is used to find faces in a video frame, after which an optical flow algorithm is applied to track feature points on the faces. Given the set of flow vectors corresponding to a single face, we propose to use the epipolar geometry for rejecting outlying flow vectors. This will improve the registration of the face over multiple frames, and thus lead to an improved super-resolution image. An iterative backprojection method is used for acquiring the super-resolution images.

1 Introduction

Nowadays many public areas are secured by surveillance cameras. The quality of the recorded material is sometimes low, making reliable recognition of individuals difficult. The use of super-resolution (SR) techniques may be promising for improving the quality of such video material.

A few approaches to SR on faces have been reported in literature. In [4] a SR frame is computed using information from past and future low-resolution (LR) frames. It calculates optical flow between interpolated versions of these frames and the initial SR frame. The updated SR frame is the average of the current SR frame and warped versions of neighboring interpolated LR frames. The algorithm considers only 5 neighboring frames, in which not much movement of the face is to be expected. In [7] the motion between frames is also estimated using optical flow. A probabilistic scheme is employed which determines for each input frame whether the pixels are visible in the SR frame or not. Only the visible pixels are used to update the SR frame. A somewhat different type of approach is the use of prior face knowledge to build SR images. It works, however, with the requirement that the observed face has the same pose as the model, e.g. a frontal view [3]. An extension to multiple poses is possible, but the results are not yet convincing [11].

Thus far, SR has mainly been applied for rigid planar objects or scenes, due to the fact that alignment is relatively easy in this case. When the object or scene is not planar, i.e. for a full 3D object, the transformation will not be a simple one-to-one mapping of pixel positions. For instance, parts of the object may be occluded after it is rotated. Not every object point will therefore be visible in all the frames which show the object.

In general, the relation between different views of a rigid 3D object is governed by the epipolar geometry [9]. Faces, however, are non-rigid 3D objects since they may change expression. Though, in practice, facial expressions will not change constantly, so that the epipolar geometry can be assumed to hold approximately.

In this paper, we demonstrate a SR method for faces based on optical flow, which makes use of the epipolar geometry. Once the epipolar geometry is known, outliers among the flow vectors can be removed. With the resulting improved motion estimates, the SR face images will be improved as well.

In Section 2 the SR scheme for faces is introduced. Experimental results for the SR scheme are given in Section 3. Finally, in Section 4, we conclude the paper.

2 A super-resolution scheme for faces

The approach we follow for SR on faces is in line of [4, 7], where optical flow is used to determine the motion. However, optical flow is only computed for feature points inside the face regions of the frame, which are found by the use of a face detector. This makes it possible to compute for each face the epipolar geometry, which describes the change in face pose w.r.t. the camera. The reason for the restriction to faces, is that the applied optical flow algorithm uses a hierarchical search method which becomes too complex for the processing of all image pixels. For every face a number of feature points is selected which will be tracked by the optical flow algorithm. The flow vectors are then used to compute the epipolar geometry describing the change in face pose w.r.t. the camera. Outliers among the flow vectors are then removed by retaining only those vectors which support the epipolar geometry. The computed optical flow is not dense, and therefore the per-pixel flow vectors are found by averaging sets of neighboring feature flow vectors. The applied SR algorithm is based on backprojection of the difference between the current frame and the simulated LR frame. An overview of the scheme is shown in Fig. 1.

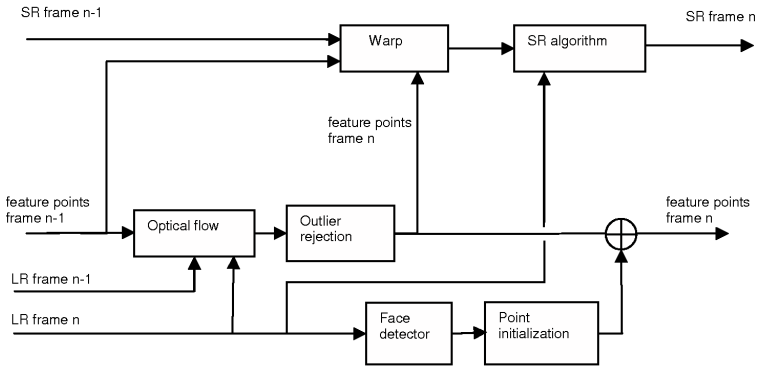
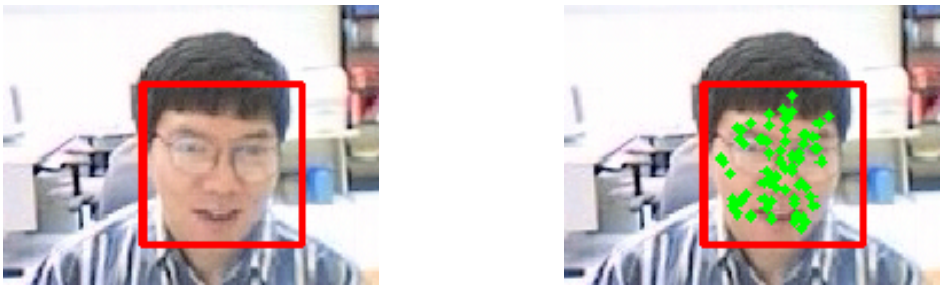


Figure 1: A schematic overview of the SR method for faces.

Next, we describe the separate elements in the SR scheme in more detail. For every element it is indicated whether it works on the whole frame (i.e. no subdivision per face) or on a per face basis.

2.1 Face detection - per frame

The face detector is taken from the OpenCV library [2], and is based on [12]. It is a fast detector which comes equipped with a set of trained classifiers. We have selected the default frontal face detector for our purpose, since recognition is best possible from frontal face views, and let it detect faces of size 20x20 pixels or larger. The classifier uses a collection of Haar features, which describe the differences in pixel sums between connecting regions. The features are examined in a cascaded classifier, which passes candidate face regions in a sequential fashion. When a region is not a face, it will be immediately rejected so that valuable processing time is saved. The output of the detector is a set of rectangles enclosing the detected faces, see Fig. 2(a). Note that although the images are displayed here in color, all processing is done on gray value images.



(a) (b)

Figure 2: A detected face in a video frame (a), and initialized feature points inside the rectangle (b).

2.2 Point initialization - per face

To estimate the motion of the face, a set of feature points (80 in total) is initialized inside the rectangle of the face detector. This is done because the hierarchical flow algorithm is too complex to apply for a dense pixel grid. The point set is connected to a single face and is processed separately in the **outlier rejection** stage. The points are equally spread inside a circle contained in the rectangle, which gives the highest probability of having all points on the face since the rectangle will typically cover parts of the background (see Fig. 2(b)). The circle radius is chosen such that it completely fits inside the rectangle.

Another option would be to find certain interest points in the image, like corners, and use these to initialize the points. These points may be more stable, but their positions in the face may not be equally spread. Since the displacement of pixels will depend on neighboring feature points (see **Warp** section), it is important to have sufficient features in each pixel's neighborhood.

2.3 Optical flow - per frame

The feature points of all faces are being tracked across the frames using a variant of the Lucas-Kanade optical flow algorithm from the OpenCV library [2]. This algorithm computes the flow using a pyramidal representation of the images [5]. It is set to run at

maximum 20 iterations or stop when the update is smaller than 0.03 pixel. In case the flow for a particular feature point can not be found, the point will be removed. Starting at the coarsest pyramid level and descending to the original size, the flow is computed at each level by using the flow of the previous level as initialization. The pyramid levels are created iteratively by low-pass filtering and factor 2 subsampling of the original image. Points falling outside the image borders will be marked as lost. When the displacement of the point is smaller than a threshold (0.2 pixel) in both x and y-direction, we assume that the point is locked on the static background and remove it. This is motivated by the fact that faces are expected to be continuously moving in the scene, that is, at least over one or more pixels per frame. In case there are no points left after the flow computation, the face will not be processed any further (except when it is detected again).

2.4 Outlier rejection - per face

As stated earlier, the displacements of correctly tracked points should obey the epipolar geometry [9], given that the facial expression does not change significantly. The mathematical representation of the epipolar geometry, the fundamental matrix F, is given by

$$\mathbf{x}'^{\top} F \mathbf{x} = 0 \tag{1}$$

where $\mathbf{x} = (x, y, 1) \leftrightarrow \mathbf{x}' = (x', y', 1)$ is a corresponding point pair across two frames, given in homogeneous coordinates.

The fundamental matrix can be computed from 7 corresponding point pairs. Since there are outliers among the correspondences, we use the robust RANSAC algorithm [6]. By taking random samples of 7 pairs and evaluating the support for the corresponding fundamental matrices, the fundamental matrix with most support among the feature pairs is found. Point pairs which do not support this fundamental matrix are labeled as outliers. They can be removed from the point set since their displacements do not conform to the movement of the face. In Fig. 3, examples of such outlying features are given. When there are too few features left for fundamental matrix estimation (say < 25), they are removed and the face is not processed anymore.

(a) (b)

Figure 3: Two examples of outliers among the feature points. The cyan lines indicate the features' positions in the previous frame. The red dots are outliers and the green dots inliers w.r.t. the computed fundamental matrix. The outliers will be removed from the point set.

The number of iterations J needed to find an all-inlier sample can be determined from the required probability of success, i.e. the probability that at least one all-inlier sample is found in J iterations [6]. Let ε denote the outlier ratio in the data. If p is the required probability of success, e.g. 0.99, then we have the relation

$$p = 1 - (1 - (1 - \varepsilon)^7)^J \tag{2}$$

In case the value of ε is known, the number of iterations to be executed would be known beforehand. Usually, however, the outlier ratio is unknown and we must use an alternative way for determining the number of iterations. An efficient manner is to begin RANSAC with a conservative estimate of ε, for example 0.99, and adjust it during following iterations [9]. When a certain hypothesized model has the largest number of support points seen so far, we know that there are at least that many inliers in the data. This imposes a lower bound on $1 - \varepsilon$ for the data set, and we can adjust the remaining number of iterations J according to (2). The RANSAC algorithm using this strategy is given in Fig. 4. Here S_j is used to indicate the set of support points in iteration j, which are the points within a small distance T from the model. The largest support set found during the algorithm is denoted by S_{max}. The ratio $\frac{|S_{max}|}{n}$ with n the number of correspondences, is therefore a lower-bound for the inlier ratio $1 - \varepsilon$.

The re-estimation step at the end of the algorithm is performed to improve the accuracy of the final model. Since S_{max} will contain only points on a model, an ordinary least squares estimate will be sufficient.

- $j = 1, \quad J = \infty, \quad S_{max} = \emptyset$
- **while** $j < J$ **do**
  - Randomly select a sample of 7 points from the data and compute the corresponding fundamental matrix.
  - Determine the set of support points S_j for the fundamental matrix by verifying which points are within threshold distance T.
  - **if** $|S_j| > |S_{max}|$ **then**
    - $J = \log(1 - p) \cdot \log^{-1}\left(1 - \left(\frac{|S_j|}{n}\right)^7\right)$
    - $S_{max} = S_j$
  - **end if**
  - $j = j + 1$
- **end while**
- Re-estimate the fundamental matrix based on the largest support set S_{max}.

Figure 4: The RANSAC algorithm for fundamental matrix estimation.

2.5 Warp - per frame

The previous SR frame will be warped into the coordinate frame of the current frame. This ensures that the SR frame always follows the current LR frame. For the warp only positions around feature positions are considered, i.e. detected faces containing feature points are being warped while the rest of the image is left unaffected. This is justified since the primary interest is the increased resolution of faces. The displacements for the pixel positions are obtained by simply averaging the displacements for the neighboring feature points. Let us denote the total set of feature correspondences between frame $n-1$ and n by $\mathbf{x}_{n-1} \leftrightarrow \mathbf{x}_n$ where $\mathbf{x} = (x,y)$. We then create a reversed dense flow map $f(\mathbf{x})$ and accompanying feature counter $n(\mathbf{x})$ by the accumulation process

$$\begin{aligned} & f(\mathbf{x}) = 0 \quad \forall\, \mathbf{x} \\ & \textbf{for each } \mathbf{x}_{n-1} \leftrightarrow \mathbf{x}_n \textbf{ do} \\ & \quad f([\mathbf{x}_n]) \leftarrow f([\mathbf{x}_n]) + (\mathbf{x}_{n-1} - \mathbf{x}_n) \\ & \quad n([\mathbf{x}_n]) \leftarrow n([\mathbf{x}_n]) + 1 \\ & \textbf{end for} \end{aligned} \tag{3}$$

where $[\mathbf{x}]$ denotes the rounded coordinate values. Subsequently, both arrays are convolved with kernel k

$$f(\mathbf{x}) \leftarrow k * f(\mathbf{x}) \qquad n(\mathbf{x}) \leftarrow k * n(\mathbf{x}) \tag{4}$$

where k is chosen as a square-shaped 21x21 kernel with unity elements, though other choices for k are also possible. The array $n(\mathbf{x})$ will eventually contain the number of feature points which fall within the support region of k centered at $\mathbf{x}$. The final flow map is created by

$$f(\mathbf{x}) \leftarrow \frac{f(\mathbf{x})}{n(\mathbf{x})} \quad \forall\, \mathbf{x} \tag{5}$$

After $f(\mathbf{x})$ for the LR frame is constructed in this way, it is interpolated with SR factor z and its values multiplied by z to yield the flow map in SR dimensions. The warped SR frame is then obtained by bicubic interpolation of SR frame $n-1$ at the positions indicated by $f(\mathbf{x})$. An additional step that is taken during the warp, is that at positions where $f(\mathbf{x}) = 0$ the SR pixels are directly copied from the bilinear interpolated LR frame. Since the warping from previous frames can introduce persistent artefacts in the images, a reset of the background is periodically required. It is unlikely that a face is affected in this way, since faces are assumed to be constantly moving.

2.6 SR algorithm - per frame

The SR method we use is from [10], which computes the difference between the observed LR frame I_{LR} and the simulated LR frame from the (warped) SR frame I_{SR}. The difference is then backprojected onto the SR frame. In particular, the SR frame is obtained by iterative application of

$$I_{SR} \leftarrow I_{SR} + \lambda g * ((I_{LR} - (g * I_{SR}) \downarrow z) \uparrow z) \tag{6}$$

where g, z and λ are the camera's point spread function (in SR dimensions), the SR factor and a weighting factor, respectively. The downward arrow represents subsampling of the

image, and the upward arrow upsampling with the insertion of zeros. The value of g has been chosen heuristically, and we have taken $\lambda = 0.2$. We update the SR frame according to (6) over 10 iterations. At the start of the video sequence, the initial SR frame is the bilinear interpolated LR frame 1. At frame n it is the warped SR frame $n-1$.

3 Experimental results

The SR scheme has first been applied to a video sequence from the NRC-IIT Facial Video Database [8], which contains compressed video sequences of size 160x120 pixels at 20 fps recorded by a webcam. The point spread function g was chosen as a Gaussian with size 7x7 and standard deviation 1.5. Only a single face and accompanying point set was allowed to be tracked in the sequence, since otherwise the point density would be much higher due to multiple detections. The threshold for including a point pair in the support set of a fundamental matrix was 0.5 pixel. In Fig. 5 the original, bilinearly interpolated and SR frames are shown with $z = 2$ for different frames in a sequence. An improvement of the SR images w.r.t. the original and interpolated images can be observed.

The effect of outlier rejection is shown in Fig. 6, where the SR result of a frame is also shown with omission of the steps in Section 2.4. Outlying feature points then remain in the point set and continue to produce misalignments, until they can not be tracked anymore or the total point set becomes too small.

In order to perform a quantitative measurement of the performance, we have down-sampled the sequence from Fig. 5 to 80x60 pixels and subsequently run the algorithm with $z = 2$. In this way the mean squared error (MSE) of the results with the original frames can be computed. The MSE values are only calculated for the center image square of 20x20 pixels, which is most of the time enclosed by the face. The results for the first 150 frames are shown in Fig. 7. Indicated are the MSE values for the bilinearly interpolated frames w.r.t. the original frames, and for the proposed SR method w.r.t. the original frames. There are four periods when the face is covered with sufficient feature points to allow SR; here the MSE is smaller than for the interpolated frames. Due to face movement the feature points are lost over time, and only re-initialized when the face is detected again. The peaks where the MSE for the SR result exceeds that for the interpolated frames, are due to too large detection rectangles so that feature points are initialized outside of the face.

Although the effect of SR in the sequences of Fig. 5 is noticeable, it is not necessary for recognition of individuals; the original frame is large enough to perform recognition. A scene where recognition of individuals is considerably more difficult is shown in Fig. 8. This scene is taken from [1] which contains recordings made in a public area. The data is recorded at 25 fps with frame size 384 x 288 pixels, and is made available in JPEG format. In Fig. 8, the largest face is cropped from the frame, and shown separately together with the interpolated and SR faces. Since the face is 17x21 pixels in size, it is about the smallest face that can be found with the detector. We have used for g a Gaussian with standard deviation 1.2 in this example. It is true that reliable recognition from the SR image is still difficult, but the SR image does give more pronounced face features than the original or interpolated frames.

(a) original frames

(b) interpolated frames

(c) SR frames

Figure 5: The original, interpolated and super resolved faces for different frames of a webcam sequence.

4 Discussion

A SR scheme has been described for improving face images. The scheme relies on a face detector, so that optical flow vectors can be computed per face in the scene. It was proposed to use the epipolar geometry for rejecting outliers among the flow vectors. With the removal of such motion errors, the resulting SR image was improved.

In case persons move their head very fast or change their expression, the motion estimates may still be erroneous and the SR result deteriorates. Another issue is rotation of the head which may occlude some face parts or the whole frontal face area.

Currently, the SR scheme does not add feature points to already detected faces, so that the point set will eventually loose all its points while the face may still be visible. However, the addition of new points is not trivial, especially when the point set covers only a part of the face.

(a) No outlier rejection (b) Outlier rejection

Figure 6: A SR frame without (a) and with (b) outlier rejection.

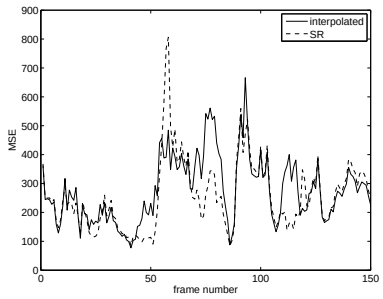


Figure 7: Mean squared errors for the center square of the interpolated and SR frames.

In addition to using the epipolar geometry merely for outlier rejection, it may also be used for inlier correction. The inlying points can be repositioned to fit the calculated geometry even better. Furthermore, a dense flow field may be obtained by exhaustive searching along the epipolar lines.

References

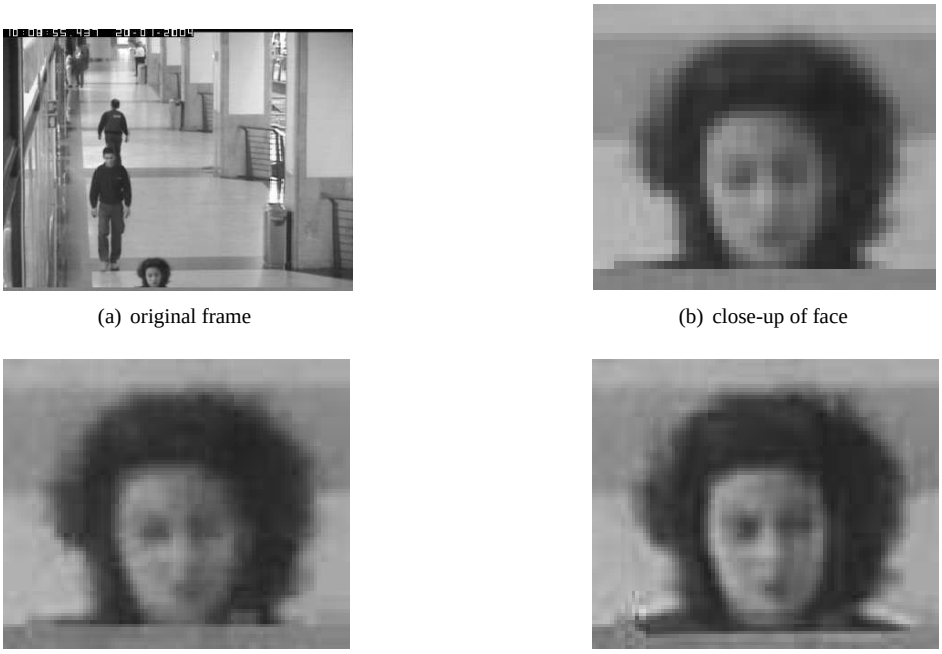
[1] EC funded caviar project / IST 2001 37540. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/.

[2] Open computer vision library, Windows version beta 5 (21-7-2005). available from http://opencvlibrary.sourceforge.net/.

[3] S. Baker and T. Kanade. Hallucinating faces. Technical Report CMU-RI-TR-99-32, Robotics Institute, Carnegie Mellon University, 1999.

[4] S. Baker and T. Kanade. Super-resolution optical flow. Technical Report CMU-RI-TR-99-36, Robotics Institute, Carnegie Mellon University, 1999.

[5] J.-Y. Bouguet. Pyramidal implementation of the lucas kanade feature tracker. Included in OpenCV library.



(a) original frame (b) close-up of face

(c) interpolated image (d) SR image

Figure 8: SR performed on a face seen by a surveillance camera.

[6] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[7] R. Fransens, C. Strecha, and L. Van Gool. A probabilistic approach to optical flow based super-resolution. In *The workshop on Generative-Model Based Vision*, 2004.

[8] D.O. Gorodnichy. Video-based framework for face recognition in video. In *Proc. of the Second Canadian Conference on Computer and Robot Vision*, 2005.

[9] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2003.

[10] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP: Graphical Models Image Processing*, 53(3):231–239, 1991.

[11] K. Jia and S. Gong. Multi-modal tensor face for simultaneous super-resolution and recognition. In *IEEE International Conference on Computer Vision*, 2005.

[12] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. CVPR*, 2001.