Correcting Rolling-Shutter Distortion of CMOS Sensors using Facial Feature Detection

Brian Heflin, Walter Scheirer and Terrence E. Boult
Securics Inc. & University of Colorado at Colorado Springs
Colorado Springs, CO 80918

Abstract— **This paper proposes a fully automated post image processing scheme based on facial feature detection to correct the horizontal temporal shear or rolling shutter distortion. This distortion occurs when obtaining images or video sequences from a CMOS camera with a rolling shutter whenever there is relative horizontal movement between the sensor and the object being imaged during the integration time of the image frame. Unlike CCD sensors, such as the interline CCD, which provides an electronic shutter mechanism called a global shutter in which the light collection starts and ends at exactly the same time for all pixels, CMOS sensors can not hold and store all the pixels at the same time. Each scanline is exposed, sampled, and stored in sequence, resulting in the rolling shutter effect or temporal distortion of the image that will cause inaccurate facial recognition results.**

Facial feature detection is performed using correlation based methods with low computational complexity. The location of key facial feature points is then used to calculate the temporal horizontal shear or the distortion of the image. This information can then be used to remove the temporal horizontal shear distortion from the detected face or the entire image. We present experimental results on controlled data sets and real scenes to show that the proposed method yields excellent results in reversing the temporal horizontal shear caused by the CMOS rolling shutter sensor and significantly improves the accuracy of our facial recognition algorithm.

I. Introduction

Many different face recognition algorithms have been developed over the last few years [7]. However, performing facial recognition in unconstrained environments has not been entirely solved because of its fundamental difficulties concerning various factors in the real world such as resolution, pose (in-plane and out-plane rotation), scale, illumination changes, occlusion, improper alignment, facial expressions, atmospheric and motion blur. Another issue in unconstrained, long-range face recognition is the trade off between field of view (FOV) and the resolution of a detected face. For example, the FOV using a 1 MPixel sensor at 100 meters is only about 1 meter which is unsuitable for unconstrained applications such as monitoring a choke point for border security. This necessitates the use of a higher resolution camera since it will give us a larger field of view (FOV) while maintaining an acceptable resolution to perform facial identification at ranges $\geq$ 50 meters. Unfortunately, many lower cost, high resolution cameras such as the Canon 5D camera which has a resolution of 12.8 MP or the Canon 7D camera which has a resolution of 18.0 MP are being equipped with low-cost, low-power, complementary-metal-oxide-semiconductor (CMOS) sensors that can geometrically distort the image if there is movement of the camera or the object being imaged during the integration time of the image.

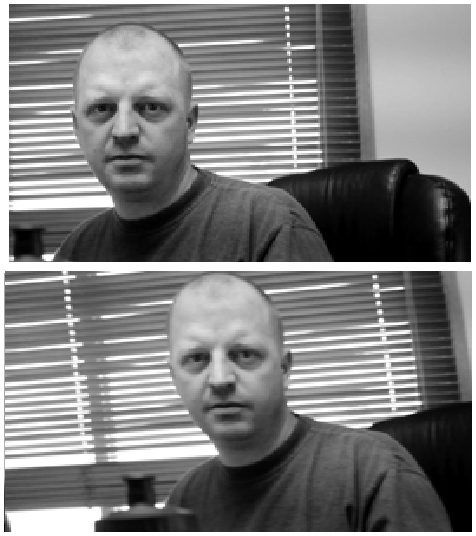

Fig. 1. Example of Rolling Shutter Distortion. Top: Non-Distorted Image. Bottom: Image With Real Rolling Shutter Distortion

An image sensor is a device that converts photons into electrons or converts an optical image into an electrical signal. Typically image sensors are either a charge-coupled device (CCD) or a complementary-metal-oxide-semiconductor (CMOS). A CMOS sensor, unlike the CCD sensor, does not expose the entire array at the same time since it cannot store and hold all of the individual pixel charges for the entire array. Instead it uses a rolling shutter where each row or scanline is exposed at different times, read out sequentially, and then merged together to form a single image. The output image does not include any geometric distortion as long as the camera and the object being imaged are stationary with each other. However, if there is relative movement horizontally between the sensor and the object being imaged the output image will be distorted or temporally sheared as shown in Figure 1. This is referred to as the rolling shutter effect.

The resulting image is unaesthetically pleasing and cannot be used directly for many computer vision applications such as facial recognition, as shown in Section V of this paper. This paper proposes a fully automated post image processing scheme based on facial feature detection to correct the

This work was supported by ONR MURI (Award Number N00014-08-1-0638)

horizontal temporal shear or rolling shutter distortion from a single image or individual frames from a video sequence. Our algorithm, while originally designed to improve the results of our facial recognition algorithm, has a dual use in recreational photography since many consumer grade camcorders, cameras, PDAs, and cellular phones are being equipped with CMOS sensors and have integrated face detection software or the ability to perform face detection by downloading 3rd party face detection software. It is first assumed that a frontal face image has been detected in the scene. Subsequent facial feature detection of the eyes and nose is performed using correlation based methods with low computational complexity. The location of the feature points is then used to calculate temporal horizontal shear or the distortion of the image. This information can then be used to reverse the horizontal shear distortion from the detected face or the entire image if we assume global motion model.

We organize the rest of the paper as follows: Previous works on rolling shutter distortion and compensation are presented in Section II. Our rolling shutter sensor model and analysis of the temporal horizontal shear distortion caused by the CMOS rolling shutter sensor architecture is presented in Section III. Section IV contains the details and implementation scheme for the purposed algorithm. Experimental results are given in Section V. Section VI presents our conclusions and direction for future work.

II. Previous Works

Rolling shutter cameras are not commonly used in computer vision applications due to the distortions that can occur in the output image. Over the past few years, analysis of the rolling shutter sensor and a few applications to remove the distortion has been described in literature.

A. Rolling Shutter Camera Models and Distortion Compensation

Boult et al. [4] provided a short analysis of the rolling shutter model and its effects on facial recognition. Baker et al. [2] purposed an algorithm to remove wobble artifacts from a video captured with a rolling shutter camera. Their algorithm used a form of temporal super-resolution to infer the high frequency motion of the camera from low-frequency optical flow. They also extended their algorithm to use an affine motion model and to model low frequency independent motion. Forssen et al. [9] proposed an algorithm for rectifying video sequences from rolling shutter cameras without the use of camera specific parameters or calibration. [9] modeled the rolling shutter distortions as being caused by the 3D motion of the camera. Gu et al.[10] proposed a new readout architecture for CMOS image sensors, called coded rolling shutter. By controlling the readout timing and exposure per row, [10] demonstrated several coding schemes that could be applied within one frame and their applications. [10] claims the required controls can be readily implemented in standard CMOS image sensors. The coded rolling shutter achieves benefits such as less skew or higher temporal resolution or higher dynamic range, at the cost of reduced vertical resolution. Ait-Aider et al.[1] proposed an algorithm to recover pose and velocity from a single view of a moving rigid object based on feature point correspondences. They used the fact that with the combined effect of rolling shutter and object motion, straight lines are distorted to curves as they get imaged with a rolling shutter camera. Wilburn et al. [23] proposed a technique to capture multi-thousand frames-per-second (fps) video sequences using a dense array of CMOS rolling shutter sensors. Compensation for rolling shutter distortion is obtained by taking scanlines that were captured at the same time from different camera views and then stacking them into one distortion free image. Liang et al. [8] purposed a method to compensate for rolling shutter distortion based on global and local motion vector detection. "The block matching technique is similar to that of MPEG-4 where we cannot but sacrifice the accuracy of motion vector to decrease the computation time. Excessive computational load due to the smoothing operation to compensate for the inaccuracy of motion vector is not acceptable in mobile devices" [6]. [5] proposed 2 methods for removing the spatio-temporal distortions in video sequences caused by rolling shutter cameras. The first method involved the use of active illumination which limits the applicability of the approach. The second method involved computationally intensive optical flow based warping. [6] also proposed post image processing scheme also based multi-frame optical flow methods. Meingast et al. [14] presented a rolling shutter camera projection model and show how it is affected by different types of camera motion and present a technique to recover the distorted image. However, correcting the distorted image using this method requires accurate measurement of the 3-D position and movement of the object and camera, which is not a simple task in practice [8]. Nicklin et al. [15] present a rolling shutter compensation algorithm for a robotic application where they use the sensor values from the robot along with knowledge of camera timing to calculate the effect of the robot's movements on the image. This information is then used to remove the rolling shutter distortion from the image.

The Foundry [1] has developed a plug-in that will remove rolling shutter distortion from video footage. Their approach also uses computationally expensive local motion estimation to correct parts of the image that are moving relative to the camera, even if they are moving at different speeds or directions. In summary, many of the previous works on rolling shutter distortion correction have been related to computationally expensive global and local motion estimation using multiple input image frames to model and reverse the geometric distortion caused by the rolling shutter sensor. However the processing time of these algorithms makes them not suitable for a real time face recognition system.

This paper proposes a fully automated post image processing scheme based on facial feature detection to correct only the horizontal temporal shear or rolling shutter distortion from a single image or individual frames from a

[1] www.thefoundry.co.uk

video sequence with low computational complexity. While our approach does not explicitly account for any vertical movement in the scene we feel that our approach is still valid for situations such as a PTZ unit panning across a scene while tracking a target or a security camera scanning across a parking lot, and is able to run in real time with minimal additional computational complexity added to our facial recognition pipeline.

III. Rolling Shutter Sensor Distortion Model

In this section we will describe the rolling shutter sensor model used for our analysis of the temporal shear distortion caused by the rolling shutter architecture.

CMOS sensors can not hold and store all the pixels at the same time. Each scanline is exposed, sampled, and stored in sequence and then immediately sent over the data bus. We define our sensor array as a $(rows * columns)$ pixel array with the origin located at the top left corner of the sensor. Chun et al. [6] presents a rolling shutter sensor model where the skew angle θ can be computed using knowledge of relative velocity of the object and the camera in the $x-direction$. [6] defines a motion vector defined by $(V_x, 0)$. It will then take $(image\ rows{*}\tau)$ to read out the entire pixel array, where τ is the time to read out a single row. Since the $y-coordinate$ of the nth now is $(y+n-1)$ then the time it takes the sensor to reach the nth row is given by $(y+n-1){*}\tau$. Furthermore, since the row is also moving with relative velocity (V_x) the displacement of the nth row in the $x-direction$ can be calculated by the following equation:

$$D(x,k) = V_x\tau(y+n-1) \tag{1}$$

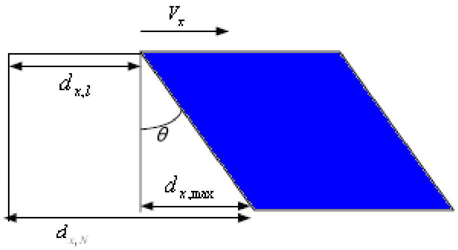

Fig. 2. Distortion due to horizontal motion; a rectangular object is distorted into a parallelogram due to rolling shutter distortion

As shown in Figure 2 a rectangular area is distorted into a parallelogram, where the skew angle θ is formed by the side of the parallelogram and the $y-axis$. [6] then defines maximum horizontal skew, $D_{x,max}$, as the difference between the displacement of the first row and the displacement of the last row as shown below in equations (2) and (3).

$$\begin{aligned} d_{x,max} &= (d_{x,N} - d_{x,1}) \\ &= v_x\tau\,(y_1 + N - 1) - v_x\tau y_1 \\ &= v_x\tau\,(N-1) \end{aligned} \tag{2}$$

$$\theta = \tan^{-1}\frac{d_{x,max}}{N-1} = \tan^{-1} v_x\tau \tag{3}$$

However, under our assumption of rectilinear and global motion the skew distortion due to the rolling shutter can be modeled by an affine or a shear transformation from non-skewed space $(x, y, 1)$ to skewed space $(x', y', 1)$. For a shear parallel to the x axis is defined as $x' = x + ky$ and $y' = y$; the shear matrix, applied to column vectors, is:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{4}$$

where (k) is the shearing parameter. To recover the image, since we are assuming a constant velocity vector V_x, we will calculate the shearing parameter (k) by determining the $x-displacement$ between 2 points in the image at different $y-locations$ that should be aligned or have the same $x-location$ in a non-distorted image.

$$k = x_{displacement(pt1\&pt2)}/y_{difference(pt1\&pt2)} \tag{5}$$

Once the shearing parameter k has been determined the image can be recovered by applying the following matrix to the column vectors of the image:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & -k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{6}$$

IV. Rolling Shutter Distortion Compensation

As discussed in the previous section we are attempting to determine the shearing parameter (k), and then apply an affine transformation to the image to remove the image distortion and recover the image. Our approach is based on using the coordinates of two specific facial features and then determining the horizontal geometric distortion based on the average offset between these points on both sides of the face. The two feature points that we have chosen are the x, y coordinates of the center of the eye slightly offset to in-between the Plica semilunaris and the inner Canthus of the eye. The second feature point is slightly offset from the Alar-side wall located on the side of the nose. As shown in Figure 3 these two points should have approximately the same x-coordinates in a non-distorted image.

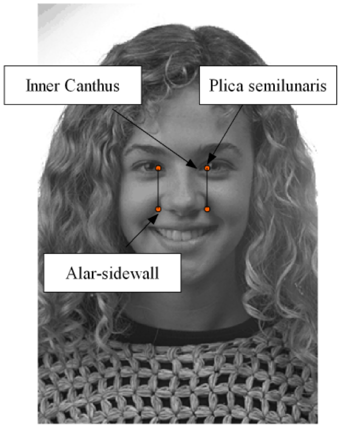

Fig. 3. Feature points used to calculate rolling shutter distortion

Due to the symmetry of the face even when there is some facial roll face the average offset between the two points in

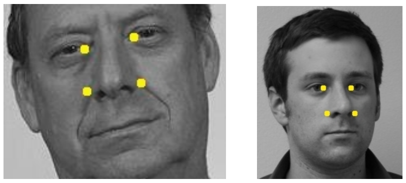

Fig. 4. Example facial images with roll and rotation. The average offset between the feature points on both sides of face is negligible; less than 1 pixel.

an image with no shear distortion is minimal. Pose variation is another issue for our restoration algorithm. Obviously we cannot recover a face if the 4 defined feature points are not visible. However, our algorithm can handle some pose variation up to 20 degrees with minimal initial error. Both of the images shown in Figure 4 have an initial error of less than 1 pixel.

A. *Facial Feature Detection using Image Correlation*

To determine the shear parameter (k) we need to first determine the x, y coordinates of the eyes and nostril. A frequently used approach for feature detection is to use correlation based technique with a model template. There are many different types of correlation filters, and the differences lie in how the filters are formed. Examples of correlation filters include the Minimum Average Correlation Energy (MACE) filters [13], the Optimal Tradeoff Filters (OTF) [18], the Unconstrained Minimum Average Correlation Energy (UMACE) filter [20], the Average of Synthetic Exact Filters (ASEF) [3], the Synthetic Discriminant Functions (SFD) [11] , the Minimum Variance Synthetic Discriminant Functions (MVSDF) [12], and the Adaptive Average Correlation Energy (AACE) [21]. To perform this task we have opted to use an UMACE filter. The UMACE filter was selected since it is less susceptible to over-fitting the training data, as compared to other methods, and it performs well in a wide variety of environmental and lighting conditions [21]. The UMACE filter will produce a single correlation filter for a set of training images. For feature detection, this technique will produce a sharp correlation peak after filtering in the positive case, from which the correct coordinates for the feature can be derived.

B. *UMACE Filter for Eye Detection*

Synthesis of the Unconstrained Minimum Average Correlation Energy (UMACE) filter began with cropping out 64x64 regions of the training data with the eye centered at coordinates (32,32). For our experiments the UMACE filter was synthesized with 1500 eye images. One advantage of the UMACE filter over the MACE filter is that over-fitting of the training data is avoided by averaging the training images. After the eyes were cropped each cropped eye region is transformed to the frequency domain using a 2D Fourier transform. Next, the average training images and the average of the power spectrum is calculated. The UMACE filter is synthesized using the following formula:

$$h = D^{-1}m \tag{7}$$

where D is the average power spectrum of the N training images, and m is the 2D Fourier transform of the average training image.

Separate filters were designed for both the left and right eyes. The UMACE filter is stored in its frequency domain representation to eliminate another 2D Fourier transform before the correlation operation is performed. Since we are performing the correlation operation in the frequency domain the UMACE filter had to be preprocessed by a Hamming window to help reduce the edge effects in the spectrum. Our experiments showed that windowing both the filter an input image decreased the accuracy of the UMACE eye detector. Since the UMACE filter is trained off line it was chosen as the input that was preprocessed by the Hamming window. Finally, after the correlation operation is performed the global maximum or peak location is chosen as the detected eye location in the original image with the appropriate offsets. Figure 5 shows an example correlation output with the detected eye centered at coordinates (18,35).

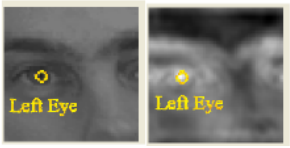

Fig. 5. Left eye localization and correlation output

C. *UMACE Filter for Nose Detection*

As presented in [3] one issue with using the UMACE and other correlation filters as an eye detector is that if the image was not cropped properly the wrong eye, nostrils, mouth, or center of the forehead was frequently detected as the eye. We decided to make use to that phenomenon and use our UMACE eye filter as our nostril detector through proper cropping of the input image. We are currently using the left eye filter for our nostril detector. We use the previously detected eye coordinates as input to our cropping algorithm, and then perform the same procedure presented in the previous section to obtain the coordinates of both the left and right nostrils. Once we have determined the x, y location of the nostrils we offset them to the Alar-side wall. Figure 6 shows an example correlation output with the detected nostrils at coordinates (32,38), and (48,38).

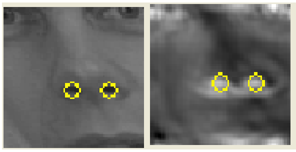

Fig. 6. Left and right nostril localization and correlation output

D. *Image Reconstruction using Facial Feature Coordinates*

One difficulty with performing image reconstruction using facial features is the issue of scale changes. In an unconstrained environment detected faces can range from only 30 pixels to the entire image. To overcome this difficulty we are scaling the detected face to have a width of 150 pixels while preserving the aspect ratio when computing the corresponding image height. This allows us to have a common face size in which we can derive the appropriate offsets from the location of the eye coordinates and the nose coordinates to compute our shearing parameter (k) while only having to train 2 UMACE filters for the left and right eye. Once we have determined the shearing parameter (k), we then scale the shearing parameter (k) by the factor that we used to scale the image. We can then recover the detected face image or the entire image, if we assume there was only global motion in the scene, using the computed shearing parameter (k), equation (6), and our custom interpolation algorithm that performs a subpixel shift on each row.

To review, the outline of the purposed algorithm is given as follows:

1) Scale the image to a width of 150 x N rows to preserve the aspect ratio.
2) Crop and Determine x, y location of the left and right eye.
3) Crop and Determine the x, y location left and right nostrils.
4) Determine the shearing parameter (k) using equation (5) and image scale factor.
5) Recover the face image or the original image using the determined shearing parameter (k) and equation (6).

V. Experiments and Results

A. Synthesized Data

We demonstrate results for our rolling-shutter distortion correction algorithm on a well-known public dataset. The FERET (Face Recognition Technology) [16] set was chosen since the since the baseline recognition for the set using our G96 recognition algorithm, based on the work of [17], without any distortion is 94.09%. This allows us to demonstrate the crippling effects rolling shutter distortion on our face recognition system while also providing a significant margin within which to demonstrate improvement as a result of our methodology. The G96 recognition algorithm creates a feature vector for each input image composed of 96 Gabor responses fed into a multiclass Support Vector Machine (SVM) to form a classification model. However, due to the constraints of our SVM-based classification method, we had use three gallery images per subject. As a result, it was necessary to deviate from the published protocol for FERET. The FERET subset chosen (dubbed, FERET240) was determined by choosing the subjects for whom the full set contained four or more images. Of these, the first three, determined by an alphabetic sort, were utilized as gallery; the fourth in the listing was used as probe. This subset contained 240 subjects and 960 face chips. Subsequently, the probe sub-set of the FERET240 set was processed using MATLAB's interp2 function to simulate horizontal shear using the shearing parameters shown in Table I.

TABLE I

Rank 1 recognition results for baseline and rolling shutter corrected. FERET240 Baseline Accuracy 94.09%.

k	Raw Images	Corrected Images	Median Estimate (k)
0.14	87.9%	90.7%	0.15
0.20	80.7%	90.8%	0.19
0.33	52.3%	83.8%	0.30
0.40	25.7%	83.6%	0.37

Faces were subsequently found using the OpenCV face detector based on a well known face detection algorithm [22]. We then applied our rolling shutter distortion correction algorithm on the detected face images. Once the face was recovered, we performed eye detection again and then used the CSU Face Identification Evaluation System [6] to perform geometric normalization based on detected eye coordinates, followed by Self Quotient Image (SQI) lighting normalization as described in [19] for both the recovered images and the distorted images.

We then ran a series of recognition tests on the synthetic data using our G96 recognition algorithm. The recognition results for the raw and the rolling shutter distortion corrected images are shown in Table I.

The G96 recognition core achieves rank-1 recognition improvement of up to 57.9% whenever there was severe horizontal shear distortion, e.g., k=0.40, successfully demonstrating the effectiveness of our rolling shutter distortion correction algorithm and showing that it is a worthy candidate for improvements specific to long-distance face recognition problems. To reduce the effects of extreme outliers due to wrong parameter estimation, the median estimate for each data set was used to evaluate the shearing parameter estimation accuracy also shown in Table I. The majority of errors in estimation of the shearing parameter was related to misclassification of one of the eyes or nostrils due to stronger responses from features of the face such as dark rimmed glasses. It should also be noted that the face detection rate began to fall off dramatically to $\leq$ 40% whenever the input image contained severe horizontal shear distortion.

B. Real Data

In order to further validate our approach, we processed several videos of subjects skewed by rolling shutter camera mounted onto a tripod. In total, 4 different videos were processed, each containing about 100 frames. The gallery consisted of a small watchlist of 50 subjects (typical of small-scale surveillance applications), including the subjects in each video and 46 random subjects from the FERET240 data set. Each frame was processed with exactly the same procedure as the above experiments. Only rolling shutter distortion correction was applied to each frame. Examples of the original source imagery and our recovered imagery are given in Figure 7.

The rank 1 recognition results are given in Table II. From this table, we see a performance improvement on real world data. The measured shearing parameter (k) ranged from 0.20 to 0.30. Our previous experiments on the FERET240 data set using synthetic horizontal shear show that an improvement of 10% to 30% on data that exhibited the same amount of distortion without any motion blur. Thus based on our comparable results, we conclude that our proposed approach is a worthy candidate for improvements specific to long-distance

TABLE II

RANK 1 RECOGNITION RESULTS FOR FRAMES FROM VIDEOS FOR FOUR DIFFERENT SUBJECTS. THE GALLERY WAS A SIMULATED SURVEILLANCE WATCHLIST OF 50 SUBJECTS.

Subject	Range (k)	Raw Images	Corrected Images
1	0.35-0.54	56%	75%
2	0.19-0.35	59%	69%
3	0.21-0.29	25%	32%
4	0.25-0.32	48%	52%

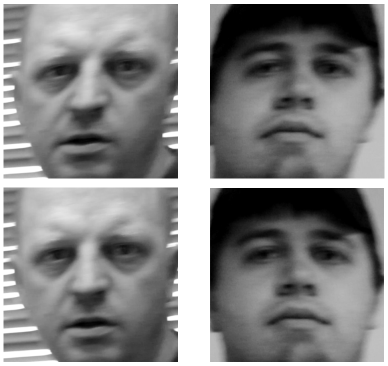

Fig. 7. Examples of original imagery from video (top row) and the resulting corrected imagery for comparison (bottom row)

face recognition problems. Due to the large amount motion blur present in the images, which also had a significant impact on our recognition results, we feel that a complete solution consists of performing both rolling shutter distortion correction and deblurring before the image is sent to the recognition core.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have presented an algorithm for a fully automated post image processing scheme based on facial feature detection to correct the horizontal temporal shear or rolling shutter distortion when obtaining images or video sequences from a CMOS camera with a rolling shutter whenever there is relative horizontal movement between the sensor and the object being imaged during the integration time of the image frame. While our approach does not explicitly account for any vertical movement in the scene we feel that our approach is still valid for situations such as a PTZ unit panning across a scene while tracking a target or a security camera scanning across a parking lot, and is able to run in real time with minimal additional computational complexity added to our facial recognition pipeline.

We have presented results on a controlled data set (FERET240) to show that the rolling shutter distortion can severely cripple our recognition system that performs with baseline accuracy of 94.09%. We also processed a series of videos containing real rolling shutter distortion in a live setting. We have demonstrated a significant increase in recognition rates as a direct result of our rolling shutter correction techniques over the recognition results on the raw source images.

A. Future Works

Future work includes incorporating compensation for vertical movement during the image capture and using our motion blur estimation and deblurring techniques as a preprocessing step to the rolling shutter correction, since the real video also exhibited a large amount of motion blur in addition to rolling shutter distortion.

VII. ACKNOWLEDGMENTS

Portions of the research in this paper use the FERET database of facial images collected under the FERET program, sponsored by the DOD Counterdrug Technology Development Program Office.

REFERENCES

[1] Ait-Aider and O. et al. Exploiting rolling shutter distortions for simultaneous object pose and velocity computation using a single view. In *ICVS*, page 35, 2006.

[2] S. Baker, E. Bennett, S. B. Kang, and R. Szeliski. Removing rolling shutter wobble. In *IEEE CVPR*, 2010.

[3] D. Bolme, R. Beveridge, and B. Draper. Average of synthetic exact filters. In *IEEE CVPR*, 2009.

[4] T. Boult and W. Scheirer. Long range facial image acquisition and quality. In M. Tistarelli, S. Li, and R. Chellappa, editors, *Handbook of Remote Biometrics*. Springer, 2009.

[5] D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich. Synchronization and rolling shutter compensation for consumer video camera arrays. In *IEEE CVPR Workshops*, pages 1–8, 2009.

[6] J. Chun, H. Jung, and C. Kyung. Suppressing rolling-shutter distortion of cmos image sensors by motion vector detection. *IEEE Transactions on Consumer Electronics*, 54(4):14791487, 2008.

[7] J. R. del Solar, R. Verschae, and M. Correa. Recognition of faces in unconstrained environments: A comparative study. *EURASIP Journal on Advances in Signal Processing*, pages 1–20, 2009.

[8] C. L. et al. Rolling shutter distortion correction. *SPIE*, 5960:1315–1322, 2005.

[9] P. Forssen and E. Ringaby. Rectifying rolling shutter video from handheld devices. In *IEEE CVPR*, 2010.

[10] J. Gu, Y. Hitomi, T. Mitsunaga, and S. Nayar. Coded rolling shutter photography: Flexible space-time sampling. In *ICCP*, 2010.

[11] C. F. Hester and D. Casasent. Multivariant technique for multiclass pattern recognition. *Appl, Opt.,*, 19(11):1758–1761, 1980.

[12] B. V. K. V. Kumar. Minimum-variance synthetic discriminant functions. *J. Opt. Soc. Am. A*, 3(10):1579–1584, 1986.

[13] A. Mahalanobis, B. V. K. V. Kumar, and D. Casasent. Minimum average correlation energy filters. *Appl, Opt.,*, 26(17):3633, 1987.

[14] M. Meingast, C. Geyer, and S. Sastry. Geometric models of rolling-shutter cameras. *Proc. of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras*, 2005.

[15] S. Nicklin, R. Fisher, and R. Middleton. Rolling shutter image compensation. In *RoboCup 2006*, pages 402–407, 2007.

[16] P. J. Phillips, H. Moon, P. J. Rauss, and S. Rizvi. The FERET evaluation methodology for face recognition algorithms. *IEEE TPAMI*, 22(10), 2000.

[17] N. Pinto, J. J. DiCarlo, and D. D. Cox. How far can you get with a modern face recognition test set using only simple features? In *IEEE CVPR*, 2009.

[18] P. Refregier. Optimal trade-off filters for noise robustness, sharpness of the correlation peak, and horner efficiency. *Optics Letters*, 16:829–832, 1991.

[19] M. Savvides and B. Kumar. Efficient design of advanced correlation filters for robust distortion-tolerant face recognition. In *AVSS*, pages 45–52, 2003.

[20] M. Savvides and B. V. K. V. Kumar. Efficient design of advanced correlation filters for robust distortion-tolerant face recognition. In *AVSS*, pages 45–52, 2003.

[21] W. Scheirer, A. Rocha, B. Heflin, and T. Boult. Difficult detection: A comparison of two different approaches to eye detection for unconstrained environments. In *BTAS*, 2009.

[22] P. Viola and M. Jones. Robust real-time face detection. *Int. Journal of Computer Vision*, 57:137–154, 2004.

[23] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz. High-speed videography using a dense camera array. In *IEEE CVPR*, 2004.