# Detection and Tracking of Faces in 3D Using a Stereo Camera Arrangements

Faleh AlQahtani, Jasmine Banks, Vinod Chandran, and Jinglan Zhang

***Abstract*—3D facial tracking has become vital to the continued integration of computers, technology, and human society. In recent decades, the integration of technology has increased, and the use of surveillance, conference calls, gaming components, and other similar applications has spurred demand for the ability to recognize the distinctive features of humans. However, in order for these new technologies to function effectively and reach their fullest potential, a great deal of work is still needed. The field of facial mapping and tracking is still in its early developmental stages, necessitating additional research into the best methods of tracking and monitoring specific human faces. To this end, an algorithm has been created that would allow for improvements in this area; however, a video was first required that could be used effectively for the algorithm. Two web cameras running on Raspberry Pi were used to gather the footage necessary for detecting and tracking specific facial features. While certain limitations were identified throughout the process, the algorithm still achieved significant successful tracking results. In spite of this success, further efforts are still needed to effectively explore the proposed algorithm and improve upon these initial results.**

***Index Terms*—3D face tracking, face detection, realtime tracking, facial landmark tracking, deformable face model, camera calibration.**

## I. Introduction

Research at the end of the twentieth century and the beginning of the twenty-first century has focused heavily on improving human-computer interfaces due to the massive impact of technology on all aspects of life. The use of computer visualizations has been rapidly increasing in the world of computing and has recently been adopted in various fields, including gaming, teleconferencing, biometrics, marketing, emotion analysis, facial recognition, surveillance, indexing, and image searching. Advancements in computer visualization technology have contributed significantly to improvements in fraud avoidance, security, activity recognition, and the detection of other technological crimes that are associated with human recognition mechanisms. With the increased use of video surveillance in institutions, such as schools, hospitals, shopping malls, banks, offices, and factories, video analysis techniques and algorithms have been developed to find mechanisms by which faces can be identified with ease, regardless of the person on camera. In light of these advancements, this paper built on the work of Alqahtani *et al*. [1], whose methods are being developed for (a) the detection of human faces, (b) the tracking of landmark face points over time, and (c) the estimation of depth information using a stereo camera arrangement. For experimental purposes, two web cameras on Raspberry Pi platforms were used to capture video data. All major processing was performed using an open-source library for computer vision (OpenCV), which was originally developed by Intel. Furthermore, an algorithm to track the significant landmark points on faces in the captured videos has been developed and implemented. The remainder of the paper as organized as follows: Section II discusses related work, Section III describes the camera calibration scheme, Section IV presents the experimental protocol and results, and Section V contains the conclusions.

Manuscript received February 7, 2018; revised June 6, 2018.

The authors are with Queensland University of Technology, Australia (e-mail: dr.faleh@outlook.com, j.banks@qut.edu.au, v.chandran@qut.edu.au, jinglan.zhang@qut.edu.au;).

## II. Related Work

3D facial recognition has been developed to enhance facial identification for individuals by using specific features that are unique to those individuals [2]-[9]. Face tracking relies on the extraction of random uniformly distributed points on the face of a human being, which are first generated in a 2D form and later used to generate a 3Dface model based on head movements, changes in pose, and the introduction of new feature points [10]-[13].The process of tracking faces with the goal of improving human-computer interactions is prone to errors due to the extreme number of computations required to capture real-time information from an individual. Errors from cluttered backgrounds, occlusion, human skin color, and illumination can all cause problems when attempting to generate accurate and reliable information that can be used for tracking specificfaces [14]-[17]. Many computational algorithms in this field have been proposed with the goal of curbing these universal problems, which have prevented the advancement of face tracking technology.

The real-time 3D tracking and modeling framework created by Choi *et al*. (2010) automatically reinitializes and reacquires images in order to reduce errors due to occlusion and instability. They use an automatic pose correction mechanism with a RANSAC-PnP process that dynamically records random uniformly distributed points on the human face with the goal of increasing the range of poses [10], [18]. This mechanism measures the errors that accumulated during video capture and allows the system to perform automatic corrections, which will help in reducing the drift that results from occlusion, facial variability, a wide field of view, and instability due to fast movement [19]. However, their algorithm uses biased tracking points and weak reacquisition mechanisms that could be replaced with a more advanced RANSAC-PnP algorithm. This would introduce an automatic

doi: 10.18178/ijmlc.2019.9.1.762

correction mechanism and allow for higher error-tolerance. The Active Appearance Model (AAM), which is currently being researched by Huitema *et al*. (2014), is another attempt to overcome the shortcomings of visual technology by constricting the fit of the human face by using a linear 3D morphable model [20], [21]. This results in the generation of facial landmarks that aid in feature-based tracking using the AAM algorithm and appearance-based generative face models, which can be tracked using a 3D morphable mechanism [21].

The CamSHIFT and KLT algorithms are additional face-tracking mechanisms that have been developed in order to describe faces in a frame-by-frame manner [22]-[25]. CamSHIFT controls color distribution probability by maintaining proper sizes, and uses the meanSHIFT mechanism to locate the center of human faces [23]. On the other hand, the KLT mechanism tracks the movement of a human and is responsible for calculating brightness changes by assuming that every feature in the human face is constant. Lukas and Kanades optical flow algorithms reduce image instability by using a 2-mode tracking system that incorporates both short-range and long-range features [26] [27]. The lack of computing resources with which to analyze the dynamics of visual technology has resulted in both generative and discriminative models [28]-[30]. A generative model identifies similar regions in the target image area, while a discriminative mechanism uses background differentiation and assigns binary numbers to the target object, which are later analyzed using a local orientation histogram and pixel colors [31]. A tracking-modeling-detection (TMD) mechanism is used for tracking the objects that train the classifiers [28], [32]. Cascade Regressors-based face recognition mechanisms create semi-dense 3D objects that record texture information in an attempt to build a holistic 3D representation from video frames [33]. This algorithm is known to be very effective at capturing facial expressions without the need for person-specific training, especially in wild videos. This eliminates the need for manual subject training, which is known to be very time-consuming and expensive. Stereo information can also be used for calculating active facial processes. Two stereo cameras are installed such that they have different fields of view. One is used for wide face tracking and the other is used for narrow face identification. Skin color segmentation has been introduced to replace feature- and Eigen-based mechanisms with RGB color combinations and morphological operations [22]. Multi-face detection and tracking systems using a TLD algorithm in combination with Viola Jones object detectors and AdaBoost are also important for recording incremental facial alignments with the aid of a generic model for regressor cascades [34], [35].

## III. Camera Calibration

This section deals with the preprocessing of the video streams, specifically the correction of distortion due to the non-ideal lens of the camera and the geometric correction required for both cameras to use a common frame of reference.

### A. Camera Calibration and Lens Distortion Correction

One model that describes the procedure of capturing an image using a camera is the so-called pinhole camera model.

According to this model, a point located at the coordinates $[X\ Y\ Z]^T$ in three-dimensional space is projected onto a pixel $[u\ v]^T$ in the image captured by the camera based on the following formula:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

where:

- $[X\ Y\ Z]^T$ is a point in the 3D scene that has to be recorded and $[u\ v]^T$ are the corresponding pixel coordinates.
- The matrix

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

is called the camera matrix, or the matrix of intrinsic parameters. The parameters are the point $[c_x\ c_y]^T$, which is typically at the center of the image, and the horizontal and vertical focal lengths of the camera, denoted $f_x$ and $f_y$, respectively.

Finally, there is a matrix that performs the rotation and translation of the 3D point. This is decomposed into a $3\times3$ matrix $R$ and a $3\times1$ vector $t$.

$$[R \mid t] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

A more general model for the process of capturing a 3D scene using a camera also considers the distortion caused by a non-ideal lens. This extended model is described by the following equations [12]:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t$$

$$\begin{aligned} x' &= x/z \\ y' &= y/z \\ r^2 &= x'^2 + y'^2 \\ x'' &= x' \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2_{p1} x' y' + p_2(r^2 + 2x'^2) \\ y'' &= y' \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2_{p1} x' y' + p_2(r^2 + 2x'^2) \\ u &= f_x x'' + c_x \\ v &= f_y y'' + c_y \end{aligned}$$

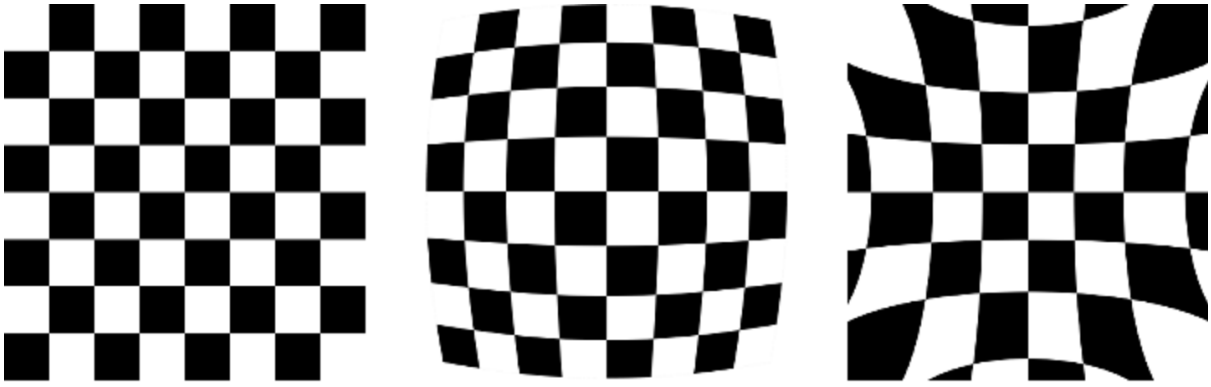


Fig. 1. Example of the distortion created by the lens of a camera.

In the above equations, $k_1, k_2, k_3, k_4, k_5$, and $k_6$ are radial distortion coefficients, while $p_1$ and $p_2$ are called tangential distortion coefficients. All these coefficients must be estimated during a calibration phase in which the camera captures images with a known geometry (e.g., a checkerboard) and the estimated coefficients are used to correct the captured images. The estimation of camera-intrinsic and camera-extrinsic parameters by using images of several views of a calibration pattern has been attempted here. A typical calibration pattern is that of a "checkerboard image", such as the images presented in Fig. 2. Using a set of such images, the computation of the cameraintrinsic and camera-extrinsic parameters for the two cameras used in the experiments was performed, which was found to be equal to

$$\boldsymbol{A_1} = \begin{bmatrix} 1400.45 & 0 & 600 \\ 0 & 1400.45 & 250 \\ 0 & 0 & 1 \end{bmatrix}$$

for the first camera and

$$\boldsymbol{A_2} = \begin{bmatrix} 1399.41 & 0 & 550 \\ 0 & 1399.41 & 274 \\ 0 & 0 & 1 \end{bmatrix}$$

Fig. 1: Example of the distortion created by the lens of a camera. Instead of capturing an image with no distortion (left image), a camera typically captures an image that looks like an object on the outer surface of a sphere (middle image) or the inner surface of a sphere (right image). The estimation of the parameters $k_1, k_2, k_3, k_4, k_5$, and $k_6$, as well as $p_1$ and $p_2$, can model the exact form of the non-linear distortion of the lens. Thus, by finding the values of these parameters, distortion due to the lens can be corrected.

For the second camera. Similarly, for the lens distortion parameters, these values were found:

$$[k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3] = [-0.5 \quad 0.21703 \quad 0 \quad 0 \quad 0]$$

for the first camera and:

$$[k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3] = [-0.5 \quad 0.22908 \quad 0 \quad 0 \quad 0]$$

for the second camera. These parameters were then transformed into rotation and translation vectors. These vectors describe the transformation that must be applied to coordinates in the model coordinate space to find the related coordinates in the world coordinate space. The values for the translation and rotation vectors were:

$$\boldsymbol{r} = \begin{bmatrix} 0.007848 \\ 0.006589 \\ 0.000576 \end{bmatrix}$$

and:

$$\mathbf{T} = \begin{bmatrix} -120 \\ 0 \\ 0 \end{bmatrix}$$

It should be noted that the rotation vector can be transformed into a rotation matrix. The equation 1 is used to perform this operation.

### A. Stereo Rectification

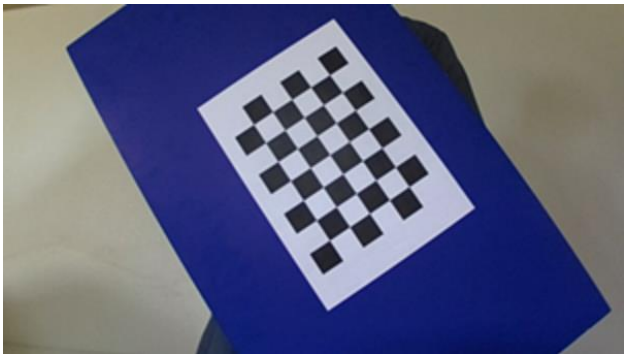
Next, the camera matrices for the two cameras $A_1$ and $A_2$ were computed, the lens distortion parameters for both cameras, and the translation vector $T$ and rotation matrix $R$, as described above. Then, the rectification transformations $R_1$ and $R_2$ for the two cameras (rotation matrices) and two projection matrices $P_1$ and $P_2$, in the new (rectified) coordinate system for the two cameras were computed. In the experiment in this study, a horizontal stereo setup in which the first and second camera views are shifted relative to one another mainly along the x-axis, was used. In this case, the projection matrices computed by the aforementioned functions have the forms:

$$\boldsymbol{P_1} = \begin{bmatrix} f & 0 & cx_1 & 0 \\ 0 & f & cy & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

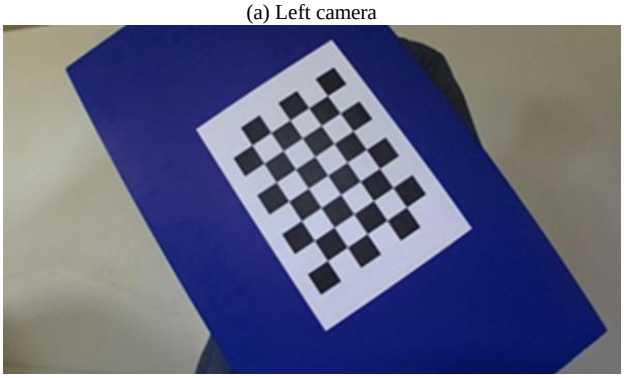
$$\theta = norm\ (r) r = \frac{r}{\theta} R = cos(\theta) I + (1 - cos(\theta)) r r^T + sin(\theta) \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \quad (1)$$

and:

$$\boldsymbol{P_2} = \begin{bmatrix} f & 0 & cx_2 & T_x * f \\ 0 & f & cy & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



(a) Left camera



(b) Right camera

Fig. 2. Example checkerboard used to compute the parameters of the left and right cameras.

$$x \leftarrow (u - c'_x)/f'_x$$

$$y \leftarrow (v - c'_y)/f'_y$$

$$[X \quad Y \quad Z]^T \leftarrow \mathbf{R}^{-1} * [x \quad y \quad 1]^T$$

$$x' \leftarrow X/W$$

$$y' \leftarrow Y/W$$

$$x'' \leftarrow x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x'y' + p_2(r^2 + 2x'^2)$$

$$y'' \leftarrow y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y'^2) + 2p_2 x'y'$$

$$map_x(u, v) \leftarrow x'' f_x + c_x$$

$$map_y(u, v) \leftarrow y'' f_y + c_y$$

where $T_x$ is the horizontal shift between the cameras. The first three columns of the matrices $P_1$ and $P_2$ will effectively become the new rectified camera matrices. These matrices, together with the matrices $R_1$ and $R_2$, are passed to the initUndistortRectifyMap [36] function to initialize the rectification maps. This is performed once for each camera.

initUndistortRectifyMap [36] computes the transformation required for joint undistortion (due to the non-ideal cameralens) and rectification. This transformation is computed in the form of a pair of maps, as required by the remap [36] function, which actually performs the transformation. The output images, after the application of the joint transformations, will look similar to the original images, but without the distortion due to the lens. The output images will appear to have been captured by a camera with a camera matrix equal to $P_1$ for the first camera frame and $P_2$ for the second camera frame. In practice, the initUndistortRectifyMap [36] function computes an inverse mapping. For each pixel ($u$ ; $v$) in the destination image (the corrected and rectified image), this function also computes the corresponding coordinates in the source image (the original image acquired by the camera). This mapping is computed using the following equations:

In III-B, some sample results after the application of the undistortion and rectification transformations have been presented. The original frames in Fig. III-B(a) and III-B(b) are transformed into the corrected frames that appear in Fig. III-B(c) and III-B(d), respectively. It is evident that the distortion due to the lens has been corrected, because the lines in the corrected images appear to be straight, while they were clearly bent in the original images. Additionally, although this is less apparent, the images have been registered with one another.

### *B. Map and Depth Estimation*

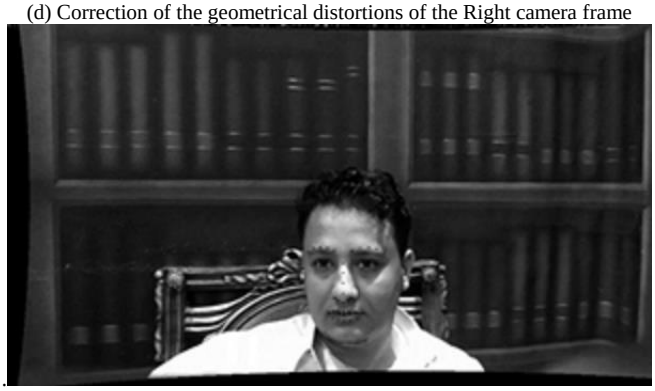
Using a set of two properly aligned cameras, it is possible to estimate the depth of various objects in a scene. This procedure mimics the method that humans use to compute depth information from the images captured by the eyes. First, a disparity map has been computed, and then the disparities are transformed into depths. In more detail, the disparity is the displacement of a particular object (or pixel) between the left and right images. For example, objects very far away in the view should appear at the same locations in the left and right images, meaning the disparity in such a case is equal to zero. Because depth is inversely proportional to disparity, the depth is infinite in this case. More generally, if an object has a disparity greater than zero, meaning it appears with some displacement between the left and right images, then it has some finite depth, which can be estimated.

(a) Frame 15 of the original video from the Left camera

(b) Corresponding image from the Right camera

(c) Correction of the geomerical distortions of the Left camera frame

(d) Correction of the geometrical distortions of the Right camera frame



(e) Application of the landmark detection algorithm to the (corrected) Left frame.

(f) Application of the landmark detection algorithm to the (corrected) Right frame.

Fig. 3. Results after the application of the undistortion and rectification transformations.

Thus, in order to perform depth estimation, pairs of corresponding pixels between the left and right images must be determined. That is why the cameras must be calibrated to have a common frame of reference. Furthermore, when given two carefully aligned images, all pixels in the first image (along with their neighborhoods) are considered and the aligning of these pixels to the second image by considering a set of potential left-right displacements is attempted. The best displacement (e.g., the one that minimizes squared error) is the estimated disparity for that pixel. If the left-right image frames are properly aligned, then a disparity map containing depth information can be computed. A semi-global block-matching algorithm is used.

An example disparity map that corresponds to the corrected images in Fig. 3(c) and 3(d) is presented in Fig. 4(a). From this image, brighter pixels are observed to correspond to objects that are closer to the camera, while darker pixels correspond to objects that are further away. Another interesting observation is that one can see certain areas in Fig. 4(a) that appear as black ”shadows”. These areas correspond to pixels in the first image (left) that cannot be aligned to the second image (right) with a small degree of error, because they do not exist in the second image. These portions of the left image do not exist in the second image because an object was placed in front of them. This phenomenon is known as occlusion.

## IV. Tracking Landmark Points on the Face

A discriminative deformable facial model that was trained by a cascade of regressors was used for the video in this study. This particular method was developed as a computationally efficient method for updating a deformable model, allowing the tracking of several specific landmark points on the human face from one frame to the next. In Fig. 3(e) and 3(f), several examples are provided that show the detection of 49 landmark points on the face for the left and right images using the video sample discussed previously. As a result of the requirement for these points to have the same frame of reference for the disparity map, certain facial points on the undistorted (or corrected) images were tracked. Thus, using Fig. 3(e) and 3(f),it became possible to detect various parts of the human face with a high level of accuracy. The following table I contains the various points detected and tracked by the algorithm.

### A. Depth Estimation Using the Disparity Map

In Fig. 4(a), the disparity map that corresponds to the frames presented in Fig. 3(c) and 3(d) can be seen. One can see that although the disparity map appears quite noisy, it has provided a smaller (darker) value for the background and a larger (brighter) value for the face and the body, as expected. This indicates that the calibration was performed correctly. Additionally, Fig. 4(b) presents a disparity map on which the key landmark points have been drawn, as computed and applied to the calibrated left frame. One can see that these points are placed correctly on the disparity map.

More specifically, let us denote two images that correspond to the left and right camera frames at time $t$ as $L((t))(u;v)$ and $R((t))(u;v)$. Assume that these images have been properly rectified. Then, the computation of the disparity will generate a new image, $D((t))(u;v)$ which is called the disparity map. This map is the solution to the following optimization problem:

$$D^{(t)}(u,v) = \arg\min_{j}\left(M\left(L^{(t)}(u,v), R^{(t)}(u,v+j)\right)\right)$$

That is to say, it will use some distance metric

$M(A,B)$ to determine where the image information for pixel $(u\,;v)$ in the left image (and its neighborhood) will appear on the right image, at some displacement $j$ and return the displacement that minimizes the metric. As an example, the distance metric could be the squared error for a neighborhood of pixels around pixel $(u\,;v)$, that is illustrate in equation 2.

where $w$ defines the size of the neighborhood of pixels. The optimization problem defined by the previous equations is solved by the StereoSGBM method [36]. The disparity information can be transformed into depth information by using the equation:

$$\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = Q * \begin{bmatrix} u \\ v \\ D(u,v) \\ 1 \end{bmatrix}$$

where the time notation has been dropped because this equation holds true for all instants in time. Matrix $Q$ is a $4\times4$ matrix computed by the stereoRectify [36] function.

Assuming that the points computed by the discriminative deformable facial model, when applied to the left camera image, are denoted $P_L^{(t)}(n)$, where $n$ ranges from 1 to 49, the disparity of the each of the points is given by the following equation:

Based on experimental results, the computed disparity to contain noticeable noise was determined. Thus, a decision to use these points was taken for estimating the "average disparit" for various points on the human face, which can in turn be used to compute the "average depth" of those points. For example, the average disparity corresponding to the left eye (points 26 through 31, as shown in the previous table) can be computed. In order to make computations more accurate, for each point, values in a $5\times5$ window centered on that point were used. This procedure was repeated for all the required points. The average disparities computed for the entire video are plotted in Fig. 5.

### B. Depth Estimation from Tracked Points

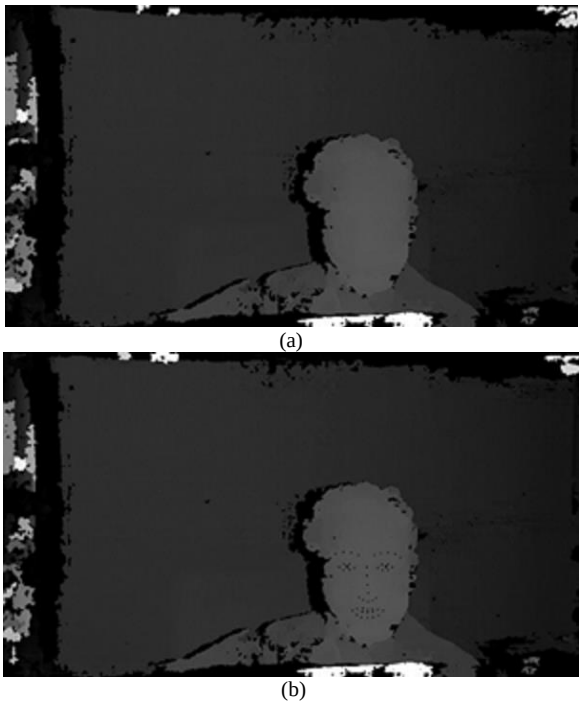
As mentioned in Section IV-A, depth estimation for landmark points on the face can be performed using a method that does not rely on a disparity map. Specifically, if points on the left and the right frames after being properly rectified are tracked, two sets of points are obtained that each correspond to the locations of the key landmark points on the face in the left and right frames. Thus, by using for the fact that disparity is defined as the displacement/distance between corresponding pixels in the left and right images, the distance between the tracked points in the left and right frames can be computed to estimate the disparity.

Let the points computed by the execution of the deformable facial model on the images from the left and right (both rectified) camera frames be denoted $P_L^{(t)}(n)$ and $P_R^{(t)}(n)$, respectively. The disparity/displacement for a point n can be computed by simply calculating the Euclidean distance between its location in left and right frames as follows:

$$B(n) = \left\| P_L^{(t)}(n) - P_R^{(t)}(n) \right\|_2$$

where $P_L^{(t)}(n)$ and $P_R^{(t)}(n)$ are assumed to be vectors containing the locations of the corresponding landmark points. Because this distance was computed in the "pixel domain", a decision was taken to normalize it by dividing it by a constant value so that the results can be compared to the results from the previous section. A constant value of 200 was chosen. The results are presented in Fig. 6. Specifically, Fig. 6 presents the normalized results for B(1) , B(10),B(14), B(25),and B(31). The following conclusions were obtained:

1) In both Fig. 5 and Fig. 6, the relative disparities between various points are the same. If the curves from Frame 62 are considered, in which the maximums are obtained, the nose has the maximum disparity (minimum depth), followed by the left eye, then the right eye, then the corner of the left eye, and finally, the corner of the right eye. In this respect, the results seem to be consistent.



(a)

(b)

Fig. 4. (a) An example of the disparity map computed using data from frame 15 of the given video. Darker pixel intensities correspond to objects that are further away from the camera array. (b) Drawing of the landmark face points onto the disparity map.

TABLE I: POINTS DETECTED AND TRACKED BY THE ALGORITHM

| POINTS | DESCRIPTION |
|---|---|
| 1 to 5 | Five points along the right eyebrow. |
| 6 to 10 | Five points along the left eyebrow. |
| 11 to 14 | Four points along the vertical part of the nose, in the center of |
| 15 to 19 | Five points along the horizontal part of the nose, above the |
| 20 to 25 | Six points around the right eye. |
| 26 to 31 | Six points around the left eye. |
| 32 to 38 | Seven points along the upper, outer lips. |
| 39 to 43 | Five points along the lower, outer lips. |
| 44 to 46 | Three points along the upper, inner lips. |
| 47 to 49 | Three points along the lower, inner lips. |

$$M\big(A(u,v),B(x,y)\big)=\sum_{i=-w}^{i=w}\sum_{k=-w}^{k=w}\big(A(u+i,v+k)-B(x+i,y+k)\big)^2 \quad (2)$$

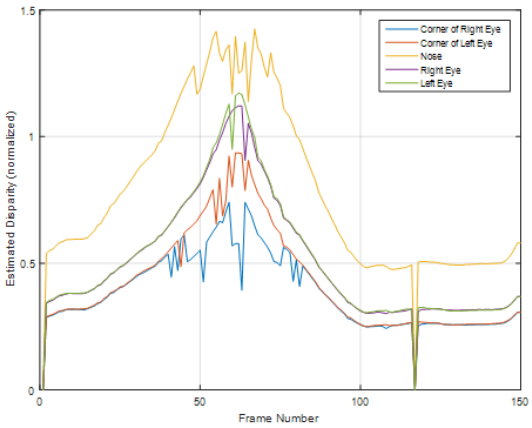



Fig. 5. Disparity (inversely proportional to depth) for major landmark areas on the face in the analyzed video. The results are based on the computed disparity map.

2) It should be noted that the disparities presented in Fig. 6 are more accurate than the disparities presented in Fig. 5. This is largely due to the noisy disparity map computed by the sgbm [36] function, which could be caused by non-optimal parameter values for the function or errors in the camera calibration parameters. Additionally, as described above, a method was used to reduce the effect of noise on the disparity map by considering average disparity values. However, the direct computation of the disparities presented in Fig. 6 is more accurate. This can be deduced from the fact that the respective disparity curves are smooth and maintain their relative positions for the entire duration of the video sequence.

As a final note, it seems that using a stereo camera and an algorithm that can detect or track certain points of interest in both the left and right frames produced by the camera is a very accurate method of depth estimation. However, such a method is only able to estimate the depth of particular points, not the depths of entire objects in the scene. Furthermore, another drawback of such an approach appears if the algorithm detects or tracks points in the left and right frames when these points do not correspond to the same physical object. For example, this situation could arise if the left and right cameras detect different faces in a scene containing multiple people.

### C. Tracking Performance

In this section, an experiment is performed to investigate the facial point tracking performance. The dataset in this study was used to perform and evaluate 3D face tracking. In all experiments, the following values have been used for the parameters of the algorithm:

- Failure Checker Interval = 1: This parameter determines how often the test to determine whether or not tracking was successful is executed. It has been set to one frame in order to force the test to be executed after every frame. This is slow, but will provide clear results.
- Failure Checker Score Threshold = -0.25: This is a numeric value that is used as a threshold to determine if tracking was successful. This value was left equal to the original value, as set by the creators of the Chehra software.
- Number of Consecutive Frames Failed before Reinitialization = 1: This parameter determines the number of failed tracking frames that must occur before the algorithm stops tracking and re-computes the landmark points from scratch. This value was set to one.

Having set the tracking parameters for the algorithm as described above, the two items are examined:

1) The number of frames for which the algorithm fails to track the points and a reinitialization is required.
2) If successful tracking was performed for the previous and current frames, the maximum Euclidean distance (in pixel space) is computed among the 49 tracked points. The maximum over all successful frames indicates the maximum displacement of a landmark point on the face that the algorithm is able to track. Specifically, if landmark points are denoted as $P_L^{(t)}(n)$ and $P_R^{(t)}(n)$ for the left and right frames, respectively, at time $t$, for $n=1$ to $n=49$, tracking performance can be assessed by computing the following:

$$p_L=\max_{t,n}\left\|P_L^{(t)}(n)-P_L^{(t+1)}(n)\right\|_2$$

and:

$$p_R=\max_{t,n}\left\|P_R^{(t)}(n)-P_R^{(t+1)}(n)\right\|_2$$

For the left video stream, a reinitialization was performed on frames 1, 109, and 116, while for the right video stream; a reinitialization was required for frames 109 and 117. Note that the video streams in this study consist of a total of 150 frames. Thus, for the left video, the algorithm was 98% successful, and for the right video, it was 98:67% successful.

Regarding the maximum distance between pairs of points that the algorithm was able to track correctly, for the left video stream, the maximum value was $p_L$=279.415, while for the right video stream, the maximum value was $p_R$=491.753. It should be noted that frames with a resolution of $960\times540$ were used.

## V. Conclusion

In recent years, increasing efforts have been made to improve human-computer interfaces, largely due to the massive impact technology has had on all aspects of life. The use of computer visualizations has increased dramatically and has been adapted in fields including gaming, teleconferencing, biometrics, marketing, emotion analysis, facial recognition, surveillance, indexing, and image

searching. These advancements have come with a host of benefits and detriments, but, regardless of their application, their usefulness is undeniable. As a result of these integrations, video analysis techniques and algorithms have been developed to quickly identify human faces, regardless of the person being recorded and the camera settings used. This study is intended at developing new methods for (a) the detection of human faces, (b) the tracking of landmark face points over time, and (c) the estimation of depth information using a stereo camera arrangement. For the purposes of this study, data was captured using two web cameras running on Raspberry Pi platforms. All major processing was performed using an open-source library for computer vision called OpenCV, which was originally developed by Intel. The collected data allowed for the development and implementation of an algorithm that can potentially be used to track significant landmark points on faces in videos. While the initial results are promising, further efforts will be required to advance this technology, focusing on reduction of both the identified limitations and the potential for error.

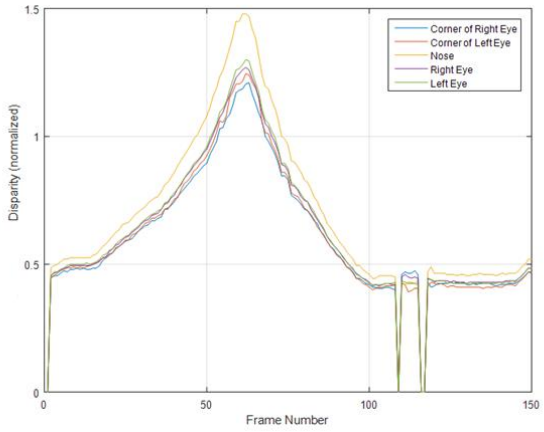



Fig. 6. Disparity (inversely proportional to depth) for major landmark areas on the face in the analyzed video. The results rely upon tracked points in the left and right frames.

## REFERENCES


[1] F. Alqahtani, V. Chandran, and J. Banks, "3d face tracking using stereo camera," in *Proc. International Conference on Computer Vision and Image Analysis ( ICCVIA )*, pp. 119–127, 2017.

[2] V. Q. Nhat, S.-H. Kim, H. J. Yang, and G. Lee, "Real-time face tracking with instability using a feature-based adaptive model," *International Journal of Control, Automation, and Systems*, vol. 13, no. 3, p. 725, 2015.

[3] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic, "Incremental face alignment in the wild," in *Proc. 2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1859–1866.

[4] H. Drira, B. B. Amor, A. Srivastava, M. Daoudi, and R. Slama, "3d face recognition under expressions, occlusions, and pose variations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2270–2283, 2013.

[5] Y. Lei, M. Bennamoun, M. Hayat, and Y. Guo, "An efficient 3d face recognition approach using local geometrical signatures," *Pattern Recognition*, vol. 47, no. 2, pp. 509–524, 2014.

[6] H. Li, D. Huang, J.-M. Morvan, Y. Wang, and L. Chen, "Towards 3d face recognition in the real: A registration-free approach using fine-grained matching of 3d keypoint descriptors," *International Journal of Computer Vision*, vol. 113, no. 2, pp. 128–142, 2015.

[7] D. Smeets, J. Keustermans, D. Vandermeulen, and P. Suetens, "meshsift: Local surface features for 3d face recognition under expression variations and partial data," *Computer Vision and Image Understanding*, vol. 117, no. 2, pp. 158–169, 2013.

[8] H. Tang, B. Yin, Y. Sun, and Y. Hu, "3d face recognition using local binary patterns," *Signal Processing*, vol. 93, no. 8, pp. 2190–2198, 2013.

[9] O. M. Parkhi, A. Vedaldi, A. Zisserman *et al.*, "Deep face recognition," *BMVC*, vol. 1, no. 3, 2015, p. 6.

[10] J. Choi, A. Tran, Y. Dumortier, and G. Medioni, "Real-time 3-d face tracking and modeling framework for mid-res cam," in *Proc. IEEE Winter Conference on Applications of Computer Vision*, March 2014, pp. 660–667.

[11] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2387–2395.

[12] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Gool, "Random forests for real time 3d face analysis," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 437–458, 2013.

[13] J. Chatrath, P. Gupta, P. Ahuja, A. Goel, and S. M. Arora, "Real time human face detection and tracking," *Proc. 2014 International Conference on Signal Processing and Integrated Networks (SPIN)*, IEEE, 2014, pp. 705–710.

[14] J. C. Terrillon, A. Pilpre, Y. Niwa, and K. Yamamoto, "Druide: A real-time system for robust multiple face detection, tracking and hand posture recognition in color video sequences," in *Proc. the 17th International Conference on Pattern Recognition, ICPR 2004*, vol. 3, Aug. 2004, pp. 302–305.

[15] H. Li, Z. Lin, J. Brandt, X. Shen, and G. Hua, "Efficient boosted exemplar-based face detection," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1843–1850.

[16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779– 788.

[17] K. Zhang and H. Song, "Real-time visual tracking via online weighted multiple instance learning," *Pattern Recognition*, vol. 46, no. 1, pp. 397– 411, 2013.

[18] M. A. Haj, J. Orozco, J. Gonzalez, and J. J. Villanueva, "Automatic face and facial features initialization for robust and accurate tracking," in *Proc. 2008 19th International Conference on Pattern Recognition*, Dec. 2008, pp. 1–4.

[19] H. E. Tasli, A. Gudi, and M. den Uyl, "Remote ppg based vital sign measurement using adaptive facial regions," *Proc. 2014 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2014, pp. 1410–1414.

[20] T. Zhang and H. M. Gomes, "Technology survey on video face tracking," *IS&T/SPIE Electronic Imaging*, International Society for Optics and Photonics, 2014, pp. 90 270F–90 270F.

[21] N. Smolyanskiy, C. Huitema, L. Liang, and S. E. Anderson, "Real-time 3d face tracking based on active appearance model constrained by depth data," *Image and Vision Computing*, vol. 32, no. 11, pp. 860–869, 2014.

[22] B. Dahal, A. Alsadoon, P. W. C. Prasad, and A. Elchouemi, "Incorporating skin color for improved face detection and tracking system," in *Proc. 2016 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, March 2016, pp. 173–176.

[23] D. Chatterjee and S. Chandran, "Comparative study of camshift and klt algorithms for real time face detection and tracking applications," in *Proc. 2016 Second International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, Sept 2016, pp. 62–65.

[24] M. M. Kassir and M. Palhang, "A region based camshift tracking with a moving camera," in *Proc. 2014 Second RSI/ISM International Conference on Robotics and Mechatronics (ICRoM)*, IEEE, 2014, pp. 451–455.

[25] C. Chen, M. Zhang, K. Qiu, and Z. Pan, "Real-time robust hand tracking based on camshift and motion velocity," in *Proc. 2014 5th International Conference on Digital Home (ICDH)*, IEEE, 2014, pp. 20–24.

[26] D. Sibbing and L. Kobbelt, "Building a large database of facial movements for deformation model-based 3d face tracking," *Computer Graphics Forum.*, Wiley Online Library, 2017.

[27] A. Plyer, G. L. Besnerais, and F. Champagnat, "Massively parallel lucas kanade optical flow for real-time video processing applications," *Journal of Real-Time Image Processing*, vol. 11, no. 4, pp. 713–730, 2016.

[28] M. Lidegaard, R. F. Larsen, D. Kraft, J. B. Jessen, R. Beck, T. R. Savarimuthu, C. Gramkow, O. K. Neckelmann, J. Haustad, and N. Krger, "Enhanced 3d face processing using an active vision system," in *Proc.*

*2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 3, Jan 2014, pp. 466–473.
[29] Y. Wu and Q. Ji, "Discriminative deep face shape model for facial point detection," *International Journal of Computer Vision*, vol. 113, no. 1, pp. 37–53, 2015.
[30] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proc. European Conference on Computer Vision*, Springer, 2016, pp. 499–515.
[31] S. Borah, S. Konwar, T. Tuithung, and R. Rathi, "A human face detection method based on connected component analysis," in *Proc. 2014 International Conference on Communication and Signal Processing*, April 2014, pp. 1205–1208.
[32] S.-K. Kang, K.-Y. Chung, and J.-H. Lee, "Development of head detection and tracking systems for visual surveillance," *Personal and Ubiquitous Computing*, vol. 18, no. 3, pp. 515–522, 2014.
[33] "Face authentication using fusion of 3d shape and texture," in *Proc. 2014 International Conference on Advances in Engineering Technology Research (ICAETR 2014)*, Aug 2014, pp. 1–6.
[34] W. Cao, G. Xu, B. Lei, P. Yin, and F. Dong, "A multiple face detection and tracking system based on tld," in *Proc. the Fifth International Conference on Internet Multimedia Computing and Service*, ACM, 2013, pp. 386–389.
[35] P. Xu, Y. Long, D. Zheng, and R. Liu, "The face-tracking of sichuan golden monkeys via s-tld," in *Proc. Chinese Conference on Image and Graphics Technologies*, Springer, 2016, pp. 85–91.
[36] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV library*, O'Reilly Media, Inc.", 2008.

**Faleh AlQahtani** received the B.Eng. and M.Eng. degrees from RMIT University in 2011 and 2013, respectively, where he is currently pursuing the Ph.D. degree with the School of Electrical Engineering and Computer Science. His current research interests include artificial intelligence, object detection, face tracking , image processing, and computer vision.

**Jasmine Banks** received the B.Eng. degree in electronics and information technology, and the Ph.D. degree from the Queensland University of Technology, in 1993 and 2000, respectively. She is currently a Lecturer with the School of Engineering Systems, Queensland University of Technology. Her research interests include artificial intelligence and image processing, computer hardware, and electrical and electronic engineering.

**Vinod Chandran** received a Ph.D. in electrical and computer engineering from Washington State University in 1990. He holds a bachelors degree in electrical engineering from the Indian Institute of Technology, Madras, MS in electrical engineering from Texas Tech University and MS in computer science from Washington State University. His research contributions span signal processing, image processing and pattern recognition with applications to biometrics and biomedical systems.

He has supervised 14 PhD students as the principal supervisor to completion and has authored or co-authored more than 170 journal and conference papers. He is currently an adjunct Professor at Queensland University of Technology, Australia.

**Jinglan Zhang** is a senior lecturer in Queensland University of Technology. She received her PhD in information technology in 2003 from Queensland University of Technology. Dr. Jinglan Zhangs broad research area falls in artificial intelligence and information systems. In particular, her research interests include visual and acoustic information (graphics, images, and sound) processing and retrieval, big data analysis and visualization, computer human interaction, escience, software engineerig, and mobile and web applications.