Efficient Multi-task based Facial Landmark and Gesture Detection in Monocular Images

Jon Goenetxea[1], Luis Unzueta[1], Unai Elordi[1], Oihana Otaegui[1] and Fadi Dornaika[2,3]

[1]*Vicomtech, Parque Científico y Tecnológico de Gipuzkoa, Donostia, San Sebastian, Spain*

[2]*Computer Engineering Faculty, University of the Basque Country EHU/UPV, Manuel de Lardizabal, 1, 20018 Donostia, Spain*

[3]*Ikerbasque, Basque Foundation for Science, Alameda Urquijo, 36-5, Plaza Bizkaia, 48011 Bilbao, Spain*

Keywords: Facial Feature Point Detection, Gesture Recognition, Multi-task Learning.

Abstract: The communication between persons includes several channels to exchange information between individuals. The non-verbal communication contains valuable information about the context of the conversation and it is a key element to understand the entire interaction. The facial expressions are a representative example of this kind of non-verbal communication and a valuable element to improve human-machine interaction interfaces. Using images captured by a monocular camera, automatic facial analysis systems can extract facial expressions to improve human-machine interactions. However, there are several technical factors to consider, including possible computational limitations (e.g. autonomous robots), or data throughput (e.g. centralized computation server). Considering the possible limitations, this work presents an efficient method to detect a set of 68 facial feature points and a set of key facial gestures at the same time. The output of this method includes valuable information to understand the context of communication and improve the response of automatic human-machine interaction systems.

1 INTRODUCTION

Human communication includes several information channels, including verbal and non-verbal messages. A complete analysis of human communication needs to evaluate the person's gestures and expressions. As a representative example, facial gestures are powerful communication elements used by any single person instinctively. Using low-cost monocular cameras, an automatic detection system can extract those expressions, being a key element for most human-machine interaction interfaces.

The final implementation of a facial gesture analysis tool needs to consider several technical factors for the final deployment. Considering computationally constrained edge devices like smartphones and onboard systems, computational efficiency is a key element that determines the final performance. Apart from the optimization techniques used for the final implementation, the adopted analysis method is one of the main factors influencing efficiency.

Moreover, the facial analysis method may need to identify different kinds of facial attributes depending on the final application scope. The list of possible attributes includes such diverse elements as the facial feature points, the orientation of the head, the eye-gaze and the local gestures, among others. There are synergies between some attributes that could be harnessed to improve the performance of the final estimation. For instance, facial feature points and the estimation of the facial gesture are strictly related attributes, and it is possible to combine them taking advantage of this relationship to improve the accuracy of both estimations.

This article proposes an efficient method based on multi-task, multi-modal and multi-level approach (named *M3*) to improve the landmark and gesture estimations, as well as other attributes like eye gaze and head orientation. It divides the analysis into local facial regions, where the estimated attributes have a strong correlation (e.g. mouth landmarks and mouth gestures). This division improves local estimations in expressiveness and accuracy, maintaining a high level of computational efficiency.

We compared computational performance and accuracy of our method with similar approaches using the same hardware, as reference for their use in platform with computational limitations.

Goenetxea, J., Unzueta, L., Elordi, U., Otaegui, O. and Dornaika, F.
Efficient Multi-task based Facial Landmark and Gesture Detection in Monocular Images.
DOI: 10.5220/0010373006800687
In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021) - Volume 5: VISAPP*, pages 680-687
ISBN: 978-989-758-488-6

2 RELATED WORK

The analysis of these gestures and emotions using monocular images can estimate the state of inattention, level of liking or other interesting indicators for multiple applications. However, the great diversity of facial shapes and possible deformations make it difficult to define those facial gestures.

In the last decades, various notations have been proposed to define facial gestures. Summarising, we can point to three main notations: a) gestures based on micro-expressions (commonly called Action Units or AUs), b) gestures based on cardinal expressions (e.g., happiness, sadness, surprise, fear, anger and disgust), and c) gestures based on a combination of local expressions (e.g., smile, kiss, blink and brow rise).

The methods based on micro-expressions (Li et al., 2017; Shao et al., 2018; Rathee and Ganotra, 2018; Sanchez-Lozano et al., 2018) combine elemental facial movements to generate more complex gestures (e.g., movements of several muscles of the mouth to define the smile gesture). The methods based on emotion or cardinal expressions (Aneja et al., 2017; Pons and Masip, 2018; Jain et al., 2018) use a set of six complex emotional gestures - happiness, surprise, sadness, fear, disgust and anger- proposed by (Ekman et al., 2002) to define a wide range of facial gestures. The methods based on face region expressions can be seen as a middle ground between the other two. Such methods estimate local zone-specific gestures and combine them to generate the global face estimation as a complete expression (Cao et al., 2016; Zhang et al., 2014; Ranjan et al., 2018).

The facial expression estimation methods use facial feature point estimations, facial texture analysis or a combination of both elements.

The method proposed in (Baltrušaitis et al., 2015) analyses the face texture using a set of previously detected facial feature points and a Histograms of Oriented Gradients (HoG) analysis to extract the gesture features. With both type of features as input, the authors propose training a specific AU regressor to estimate the gestures. Experiments show real-time performance on a desktop PC, but there are no measurements on computationally constrained devices.

In addition, the method in (Rathee and Ganotra, 2018) proposes first to detect the facial feature landmarks of the users' face and then estimate the facial expression using the face texture with the landmark location as reference. In this second stage, a set of Gabor filters analyse the area around the feature points. This method uses information that feature point detection is not using (like skin wrinkles, for example).

Within the scope of automatic detection of facial feature points, there are many and diverse methods to detect characteristic points in monocular images (Jin and Tan, 2017; Wu and Ji, 2018).

A widely adopted method was presented in (Saragih et al., 2009) which was later adopted by works like (Baltrušaitis et al., 2018). The method proposed iteratively improving an initial rough estimation of the face, analysing the region around each landmark to improve the previous location with each repetition. On each iteration, a two-dimensional structural model constrains the new locations considering all the landmark set to avoid unnatural or impossible point distributions.

Later, studies like (Ren et al., 2014; Kazemi and Sullivan, 2014) suggest using specialised regressors to detect the face landmarks, dramatically improving the performance of the process. They both assume that the regression method includes the global facial shape constraints for the entire facial structure, avoiding the use of a shape verification process. However, in both cases, accuracy suffers in the presence of out-of-plane head rotations.

As a real-time Deep Learning (DL) method, the study (Zhang et al., 2014) outlines the extraction of facial features and a set of facial gestures using the same computation adopting a multi-task strategy. Based on a direct relationship between the landmark locations and the face gesture, the proposed method uses the facial gestures to improve the landmark estimation, and the landmarks to estimate the face gestures. This method achieves real-time performance on a desktop PC with a limited set of facial gestures.

3 MULTI-LEVEL APPROACH

The proposed approach divides the task in two levels of analysis (see Figure 1): a) holistic face attribute detection and b) facial zone-specific attribute estimation.

The first level extracts general facial attributes. It takes the entire face in to consideration in order to estimate an initial set of landmarks and the head pose (i.e., the face yaw orientation).

The second level uses the output of the first level as input - it extracts the attributes of each facial zone (i.e., landmarks, gestures or gaze). First, it generates a set of normalised facial zone patches, one for each face region (i.e., mouth, eyebrows and each eye). Then, it estimates the zone-specific landmarks and gestures using the zone patches and the orientation of the head.

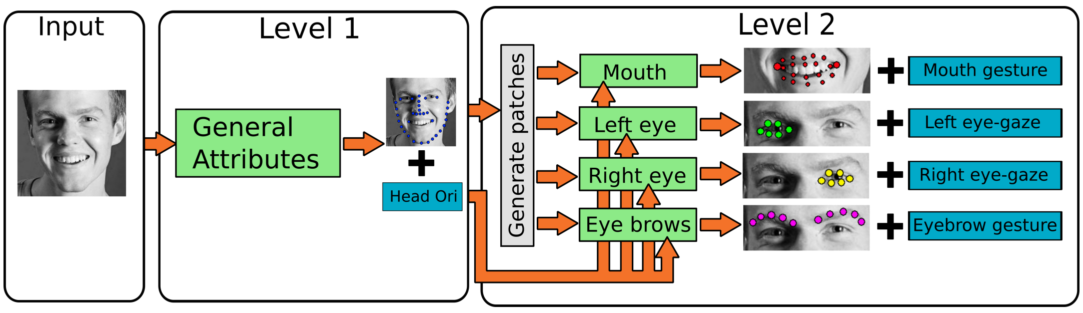

Figure 1: Flowchart of the proposed multi-level analysis pipeline.

The final output merges all the landmarks from both levels into a single set of 68 landmarks. Figure 2 shows all the landmarks detected on each level and the final set of landmarks for a face image.

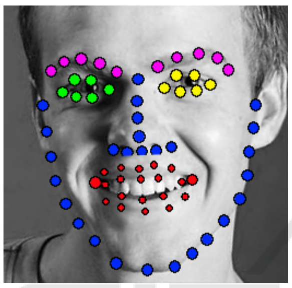

Figure 2: A combined representation of all the detected features. Each colour represents the landmarks for each face zone.

The zone patch generation process uses two *reference landmarks* from the landmark set extracted in the first step. Then, it applies a warping procedure to align the *reference landmarks* with two *target locations* in a previously defined zone patch template. The warping procedure normalises the orientation and size of the zone image region. Note that this process could add some unwanted distortion in patch regions occluded by a large out of plane head rotation.

The reference feature points used to generate the patches are those that maintain their relative location during gestures. Considering the high deformability of the mouth region, we chose feature points out of the gesture influence, like the bottom of the nose and the chin. We selected those landmarks because the influence of the evaluated mouth gestures in those locations is small enough to assume that they are stable or relatively static. Likewise, the reference points for each eye region are the eye corner landmarks, and the reference points for the eyebrow region are the left eye left corner and the right eye right corner landmarks, including both eyebrows in the same image patch. Figure 3 shows some examples of normalised face patches, including the location of the reference landmarks.

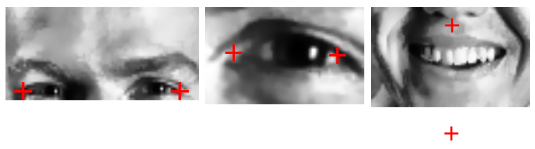

Figure 3: Examples of (from left to right) an eyebrow patch, a right eye patch and a mouth patch using the image shown in Figure 2. The red cross represents the *target location* of the *reference landmarks* for each patch. Note that the reference point of the mouth patch is out of the patch region.

To enhance the zone patches for the inference step, we apply a bilateral filtering process to reduce the possible noise in the warped image. At the same time, it preserves the information of edges and general shape. In the case of the eye regions, the warping process smooths the edge information because the size of this region tends to be small compared with the entire face region. An image intensity histogram equalisation compensates the sharpness reduction increasing the contrast and normalising the brightness for the eye patches.

Finally, each zone-specific neural network extracts the landmarks and attributes of each zone patch, also considering the estimated head orientation. The inclusion of the head orientation in the estimation process helps to ignore or reduce distortions caused by large head rotations during the warping procedure, making the estimation more reliable.

4 FACIAL ATTRIBUTE DEFINITION

The final estimation includes different facial attributes: a) the orientation of the head, b) the gesture of the mouth, c) the gesture of the eyebrows and d) the eye gaze of each eye.

The head pose (i.e., face yaw orientation) is defined as a classification between 5 possible classes *-60* (for angles <-60), *-30* (for angles from -60 to -30), *0* (for angles from -30 to 30), *30* (for angles from 30 to 60) and *60* (for angles $>$60).

The included zone-specific gestures are: smile, mouth frown and kiss gestures for the mouth region and inner eyebrow raise and inner eyebrow frown for the eyebrow region. The gestures in each region are mutually exclusive, so only one can be active in a frame. Other gestures (e.g., mouth open) can be directly extracted from the landmark positions, measuring the distance between inner mouth landmarks, for example.

Gesture estimation is a classification process which estimates the probability distribution of the group of gestures for each face element. Moreover, we include an additional neutral class for each region to represent the absence of all gestures. The final probability estimation of the gestures in a face region is normalised using a softmax function.

Eye gaze estimation defines a gaze vector for each eye as a normalised vector of three values.

5 NETWORK STRUCTURE

Each network relies on a Multi-Task Learning (MTL) approach for the estimation process. It defines a CNN model to generate a list of outputs of different types and meanings using the same computation, similar to the method proposed by (Zhang et al., 2014). Conventional MTL approaches try to optimise all tasks at once using a combined loss function. In our case, the loss function combines the loss of each task, assigning different priorities to each one. This priority designation defines the landmark detection as the primary task, setting the gesture estimation as an auxiliary task. Even if the landmarks have higher priority than the gestures, the gesture information improves the accuracy of the landmark detection due to the direct correlation between the position of the landmarks and the definition of each gesture.

The structure of each network has two parts, a) the trunk which performs the general feature extraction, and b) the leaves in charge of the final task estimations. The trunk of the network estimates all the features needed by the leaves to generate their estimations. The trunk shares its output with all the leaves, which use this information to estimate the landmarks and gestures. During training, the process updates the weights of the trunk and leaves combining all the outputs of the network. Hence, all the estimation tasks influence the training of the feature detection, enhancing their results through the synergies between them. The first row of Figure 4 shows a schematic example of a multi-task network, divided in the trunk (in green) and leaves (in blue).

The trunk is a concatenation of four convolutional layers, including a pooling step after each convolution, and a final fully connected layer. Each convolution layer includes a *Rectified Linear Unit* (ReLU) activation function. This network configuration is similar for all the cases, with small variation in layer sizes. Figure 4 shows the detailed configuration of each layer.

The input of the network in the first layer is only the cropped image of the face. However, the networks in the second level have two inputs: the normalised face zone patch and the orientation of the head estimated in layer one. To include the orientation of the head in the estimation process, the networks in the second level include an extra fully connected layer. This layer includes the head orientation values as concatenated elements (see Figure 4). Then, the final fully connected layer integrates all the elements into a single output vector.

Figure 4 includes the details and configuration of the proposed networks, and how the output of level 1 interacts with the estimation process of level 2.

6 LEARNING PROCESS

Due to the multilayer structure of the method, the learning process has two phases. In the first phase, we train the neural network defined for level 1. Then, in a second phase, we use the model trained for level 1 to generate the input data to train the models in level 2.

In all the cases, we define the landmarks $\mathbf{y}_i$ as a list of x and y coordinate values in the image $\mathbf{I}_i$, ordered as a unique list $\mathbf{y}_i = \{x_i^0, y_i^0, x_i^1, y_i^1, ..., x_i^n, y_i^n\}$. The error function for the landmarks is set as a least square problem,

$$L(\mathbf{y}, \mathbf{I}, \mathbf{W}) = \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{y}_i - f(\mathbf{I}_i; \mathbf{W})\|^2 \tag{1}$$

where f is the model function, N is the number of images in the batch and $\mathbf{W}$ is a list of weights that encapsulates the trained parameters of the neural network.

As mentioned previously, the model in level 1 classifies the head pose values (i.e., head yaw orientation) in five classes. Each head orientation class takes a probability value in the range $[0,1]$. For the ground truth, a class has the value 1 if the head pose matches the class or 0 if it does not. As an orientation can only belong to one of the classes, they are mutually exclusive.

To compute the head's pose probability, the method adopts a softmax function P, which mod-

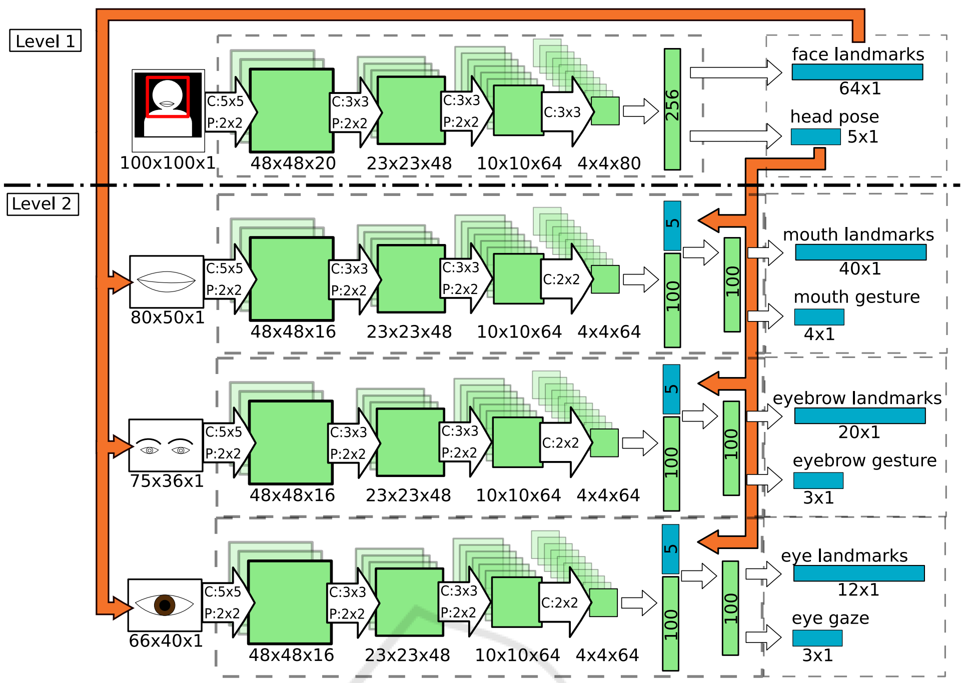

Figure 4: The neural network configuration for the attribute estimation process. The first row shows the network configuration for the first estimation level, including the input and the output elements. The rows below show the network configuration for each estimation model in the second level. A single eye model estimates the eye attributes for both eyes, using symmetry to estimate the attributes of the second eye. For each model, 'C' represents the kernel size of the convolutional layer and the 'P' represents the configuration of the pooling layer.

els the class posterior probability. P combines all orientation classes in a list of probabilities $\mathbf{h} = \{h_0, h_1, \ldots, h_4\}$. Equation (2) shows the cross-entropy loss function used for the head orientation error during the training process for a list of A classes. In this equation, $h_{i,a}$ represents the ground truth value of the orientation class a for the image i.

$$H(\mathbf{h}, \mathbf{I}, \mathbf{W}) = -\sum_{i=1}^{N}\sum_{a=1}^{A} h_{i,a} \log(P(h_{i,a}|\mathbf{I}_i; \mathbf{W})) \quad (2)$$

For the final error function, we combine equations (1) and (2) in a single expression to produce a combined output. The final loss function (3) includes a regularisation term to penalise high layer weight values, similar to the function proposed in (Zhang et al., 2014). To maintain the landmark estimation as the primary task, the final loss function includes a priority factor λ to manage the influence of the orientation class estimation error in the learning process. We define the λ value experimentally to correctly balance the landmark and gesture estimations.

$$\arg\min_{\mathbf{W}} \left\{ L(\mathbf{y}, \mathbf{I}, \mathbf{W}) + \lambda H(\mathbf{h}, \mathbf{I}, \mathbf{W}) + \|\mathbf{W}\|_2^2 \right\} \quad (3)$$

For models in level 2, the learning process follows a similar approach. However, the data used as ground truth for the training of the models in level 2 needs to be generated using the model trained for level 1. Thus, we processed the training dataset using the model trained for level 1, storing the output data for the posterior level 2 training phase.

For the mouth and eyebrow regions, each gesture g gets a value in the range $[0,1]$, 1 being if the gesture is present and 0 if the gesture is not present. For the training process, it uses a loss function similar to (3) to determine the gesture probability model.

The softmax function P models the class posterior probability. To simplify the output and assuming that the gestures in the training dataset are mutually exclusive, it combines all gestures for a facial zone in a list of gesture probabilities $\mathbf{g} = \{g_0, g_1, \ldots, g_n\}$. For example, for the gestures in the mouth region (smile, mouth frown and kiss) the gesture probability distribution would be $\mathbf{g} = \left\{g_{smile}, g_{frown}, g_{kiss}, g_{neutral}\right\}$. The last value ($g_{neutral}$) defines the absence of any gesture, having the value of 1 when none of the defined gestures is present. Equation (4) shows the cross-entropy loss function used for the gesture error during the training process for a list of B gestures (including the neutral gesture). In this equation, $g_{i,b}$ represents the ground-truth value of the gesture b for the image i.

$$G(\mathbf{g}, \mathbf{I}, \mathbf{W}) = -\sum_{i=1}^{N}\sum_{b=1}^{B} g_{i,b} \log(P(g_{i,b}|\mathbf{I}_i; \mathbf{W})) \quad (4)$$

For the landmarks in the mouth and eyebrow regions, the learning process uses the same loss function presented in Equation (1). Both loss functions (L and G) are combined in a single expression (equation (5)) to produce a combined output. The equation also includes the regularisation term and the balance modifier λ.

$$\underset{\mathbf{W}}{\arg\min}\left\{L(\mathbf{y},\mathbf{I},\mathbf{W})+\lambda G(\mathbf{g},\mathbf{I},\mathbf{W})+\|\mathbf{W}\|_2^2\right\} \quad (5)$$

For the attributes in the eye region, the learning process replaces the classification loss function with a regression function similar to that presented in Equation (1). In Equation (6), g is the model function, the vector v_i represents the ground truth values for the eye gaze vector on image i, and $\mathbf{v}$ lists the eye gaze vector values for all the images.

$$V(\mathbf{v},\mathbf{I},\mathbf{W})=\frac{1}{2}\sum_{i=1}^{N}\|v_i-g(I_i;\mathbf{W})\|^2 \quad (6)$$

Similar to previous examples, the final loss function for the eye region (Equation (7)) combines the error of the landmarks and the eye gaze estimation, with the additional λ factor to manage the influence of the secondary task.

$$\underset{\mathbf{W}}{\arg\min}\left\{L(\mathbf{y},\mathbf{I},\mathbf{W})+\lambda V(\mathbf{v},\mathbf{I},\mathbf{W})+\|\mathbf{W}\|_2^2\right\} \quad (7)$$

7 EXPERIMENTAL RESULTS

To evaluate the general performance of M3, this section compares the computation time of its implementation with two real-time methods proposed in the literature: (Kazemi and Sullivan, 2014) and (Baltrušaitis et al., 2018). Although the method by (Baltrušaitis et al., 2018) is also capable of estimating certain gestures, the experimental comparison focuses only on the result of the main task, which is the extraction of facial feature points.

The authors in (Kazemi and Sullivan, 2014) propose a fast landmark detection method, able to estimate 68 face landmarks in a single millisecond. It relies on linear regressors, using binary pixel comparisons as a feature.

The work presented in (Baltrušaitis et al., 2018) combines the AU detector presented in (Baltrušaitis et al., 2015) with efficient methods to extract facial feature landmarks (Zadeh et al., 2017), head orientation estimation, and facial texture generation in a single analysis tool.

7.1 Model Training

For the training phase, we used a subset of the images from the Multi-Task Facial Landmark (MTFL) dataset presented in (Zhang et al., 2014). However, we manually re-annotated the entire dataset with 68 facial landmarks, the gestures of the mouth (smile, mouth frown and kiss) and the gestures of the eyebrows (inner eyebrow raise and inner eyebrow frown).

First, we generated a mirrored copy of each image, doubling the number of samples. Then, we used an automatic landmark detection based on (Kazemi and Sullivan, 2014) to get a first estimation of the facial landmarks. Finally, we manually checked the results to discard the images with significant landmark estimation errors. The final dataset had 18720 images in total.

To increase the number of samples in the training dataset and increase the robustness of the estimation in different image capture conditions, we augmented the number of samples using different image perturbations (e.g., random noise, affine transformations and occlusions).

In addition, we balanced the dataset for the classification. Thus, the final training dataset for each classification process (i.e., the models for level 1, eyebrows and mouth) had the same number of samples for each of the classes to classify. For example, for the model in level 1, we generated a different number of augmentation images so that the number of final samples was the same for each orientation. We followed the same approach to create the training data for mouth and eyebrow models in level 2.

For the eye regions, we used the samples created using a synthetic eye image generator based on the approach proposed in (Wood et al., 2016). It creates a set of photo-realistic 3D eye renders, simulating eyes from different persons by variations in shape, skin colour and skin texture, among others. It also varies the eye-gaze and the orientation of the head with respect to the camera randomly.

7.2 Performance Comparison

For the comparison, we used a desktop PC to measure the processing time of each method. The PC included an Intel i5-9400F (@ 2.90GHz) as the main CPU, and 16 gigabytes of RAM, using an Ubuntu 18.04 as the operating system. No hardware acceleration was used in any of the cases.

The database used to make the comparison is the one presented by (Sagonas et al., 2016). This database includes 600 images of different resolutions annotated manually with 68 facial feature points that include

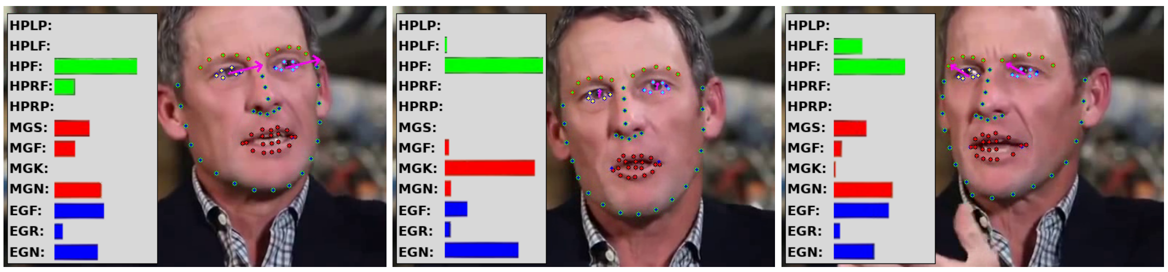

Figure 5: Each image shows the estimated gesture probabilities as bars (left), the tracked facial feature points (dots) and the estimated eye gaze vector (purple arrows). The gesture definitions are, from top to bottom, head pose left profile (HPLP), head pose left-front (HPLF), head pose front (HPF), head pose right-front (HPRF), head pose right profile (HPRP), mouth gesture smile (MGS), mouth gesture frown (MGF), mouth gesture kiss (MGK), mouth gesture neutral (MGN), eyebrow gesture frown (EGF), eyebrow gesture raise (EGR) and eyebrow gesture neutral (EGN).

points for the eyes, eyebrows, nose, mouth and the contour of the face. Since the three compared methods needed a facial region to initialise the estimation process, the bounding boxes of the landmarks defined in the database were used as input for each evaluated method.

The point detection error is the mean square error (MSE) using the 68 points on the face and averaging across all images in the dataset. We normalised the estimation of each image using the interocular distance. This normalisation avoided biases caused by different image sizes in the dataset.

The measurement of computing time in each case included only the time necessary for the calculation of the feature points, excluding any image handling process (like loading or resizing processes) or the initialisation of the models.

7.3 Qualitative Results

During tracking, the method extracted the landmark positions and gesture probabilities from the users face. Figure 5 shows some examples of attributes extracted from a monocular video sequence.

Moreover, in the experiments using embedded devices, the implementation reached real-time performance (> 25 fps) using the embedded camera as image source. The hardware used for the experiments was the Iphone SE, which has an A9 (ARMv8-A dual-core @ 1.85 GHz) central processor.

7.4 Quantitative Results

Taking all this into account, Table 1 presents the data extracted from the comparison. The table includes the landmark estimation error and the average time for their calculation.

The method proposed in (Baltrušaitis et al., 2018) is more accurate than the other methods, but the computation time is also significantly higher. In the case of (Kazemi and Sullivan, 2014), it estimates the feature points very efficiently, but at the cost of significantly higher error.

Table 1: Results of the landmark estimation time measurement.

Method	MSE	Seconds
(Kazemi and Sullivan, 2014)	0.0872	0.0018
(Baltrušaitis et al., 2018)	0.0184	0.0389
M3 (Ours)	0.0227	0.0074

To see the relationship between the MSE and the computation time, Figure 6 shows the values in a visual representation. The image shows how the proposed method improves the CPU time and error compared to the other two.

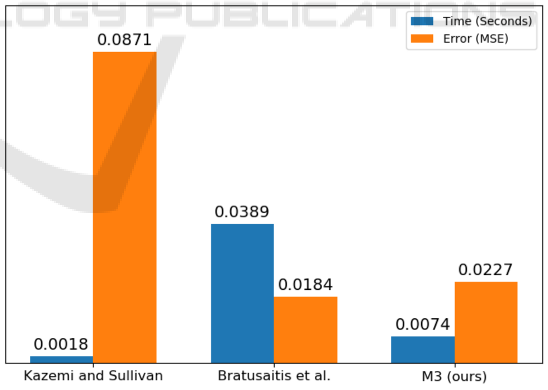

Figure 6: Visual representation of the MSE and time values for (Kazemi and Sullivan, 2014), (Baltrušaitis et al., 2018) and the proposed method (M3).

M3 presents a better balance between computational load and precision, extracting data from the face gesture in the process.

8 CONCLUSIONS

This study presents a computationally efficient method to estimate the facial feature points as well as several additional attributes (i.e., head orientation, facial gestures and eye gaze orientation) in a single computation.

The experimental section shows that even if methods like (Baltrušaitis et al., 2018) have more accurate landmark estimation and real-time performance, our method uses significantly fewer resources ($\sim$81% faster) with a less significant accuracy loss ($\sim$23%).

Considering hardware with computational constraints, and the results of the experiments as a reference, the proposed method maintains real-time performance even on smartphones with ARM architecture like the IphoneSE. Thus, it allows the integration of real-time face tracking and analysis systems into several types of constrained devices.

REFERENCES

Aneja, D., Colburn, A., Faigin, G., Shapiro, L., and Mones, B. (2017). Modeling Stylized Character Expressions via Deep Learning. In Lai, S.-H., Lepetit, V., Nishino, K., and Sato, Y., editors, *Computer Vision ACCV 2016*, volume 1, pages 136—-153. Springer International Publishing.

Baltrušaitis, T., Mahmoud, M., and Robinson, P. (2015). Cross-dataset learning and person-specific normalisation for automatic Action Unit detection. In *International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, pages 1–6. IEEE.

Baltrušaitis, T., Zadeh, A., Lim, Y. C., and Morency, L. P. (2018). OpenFace 2.0: Facial behavior analysis toolkit. In *Proceedings - 13th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2018*. IEEE.

Cao, C., Wu, H., Weng, Y., Shao, T., and Zhou, K. (2016). Real-time facial animation with image-based dynamic avatars. *ACM Transactions on Graphics*, 35(4):1–12.

Ekman, P., Friesen, W. V., and Hager, H. (2002). *Facial action coding system. Investigator's Guide*. Salt Lake City.

Jain, N., Kumar, S., Kumar, A., Shamsolmoali, P., and Zareapoor, M. (2018). Hybrid deep neural networks for face emotion recognition. *Pattern Recognition Letters*, 115:101 – 106. Multimodal Fusion for Pattern Recognition.

Jin, X. and Tan, X. (2017). Face alignment in-the-wild: A Survey. *Computer Vision and Image Understanding*, 162:1–22.

Kazemi, V. and Sullivan, J. (2014). One Millisecond Face Alignment with an Ensemble of Regression Trees. *Computer Vision and Pattern Recognition (CVPR)*.

Li, W., Abtahi, F., and Zhu, Z. (2017). Action unit detection with region adaptation, multi-labeling learning and optimal temporal fusing. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:6766–6775.

Pons, G. and Masip, D. (2018). Multi-task, multi-label and multi-domain learning with residual convolutional networks for emotion recognition. *CoRR*.

Ranjan, A., Bolkart, T., Sanyal, S., and Black, M. J. (2018). Generating 3d faces using convolutional mesh autoencoders. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, *European Conference on Computer Vision (ECCV)*, pages 725–741, Cham. Springer International Publishing.

Rathee, N. and Ganotra, D. (2018). An efficient approach for facial action unit intensity detection using distance metric learning based on cosine similarity. *Signal, Image and Video Processing*, 12(6):1141–1148.

Ren, S., Cao, X., Wei, Y., and Sun, J. (2014). Face alignment at 3000 FPS via regressing local binary features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1(1):1685–1692.

Sagonas, C., Antonakos, E., Tzimiropoulos, G., Zafeiriou, S., and Pantic, M. (2016). 300 faces In-the-wild challenge: Database and results. *Image and Vision Computing*, 47:3–18.

Sanchez-Lozano, E., Tzimiropoulos, G., and Valstar, M. (2018). Joint Action Unit localisation and intensity estimation through heatmap regression. In *BMVC*, pages 1–12.

Saragih, J. M., Lucey, S., and Cohn, J. F. (2009). Face alignment through subspace constrained mean-shifts. In *IEEE 12th International Conference on Computer Vision*, pages 1034–1041.

Shao, Z., Liu, Z., Cai, J., and Ma, L. (2018). Deep Adaptive Attention for Joint Facial Action Unit Detection and Face Alignment. *ArXiv*.

Wood, E., Baltrušaitis, T., Morency, L.-P., Robinson, P., and Bulling, A. (2016). Learning an appearance-based gaze estimator from one million synthesised images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pages 131–138.

Wu, Y. and Ji, Q. (2018). Facial Landmark Detection: a Literature Survey. *International Journal of Computer Vision*.

Zadeh, A., Baltrušaitis, T., and Morency, L. P. (2017). Convolutional Experts Constrained Local Model for Facial Landmark Detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, volume 2017-July, pages 2051–2059. IEEE.

Zhang, Z., Luo, P., Loy, C. C., and Tang, X. (2014). Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision (ECCV)*, pages 94–108. Springer International Publishing.