COSMETICS APPLICATION USING COMPUTER VISION

[1]Akash Pal, [2]Anmol Kumar, [3]Dr. R. C. Jaiswal
[1]Student, [2]Student, [3]Professor
[1]Department of Electronics and Telecommunication,
[1]Pune Institute of Computer Technology, Pune, India.

Abstract : Computer vision is a new emerging field of artificial intelligence using which machines are trained to make sense out of digital images and videos. An end solution which is directly going to benefit the producer and consumer in the cosmetics industry is planned to be developed with the help of state of art technologies like Computer Vision and Flask Framework. The solution planned is based on Image processing and comes under the banner of augmented reality. It is going to augment the cosmetics decided by the user and will apply on the face of the end user. It is beneficial for the user because it bypasses the need of physically applying the cosmetics on the face and instead can do it virtually with this solution. On the other hand it is also going to benefit the Producer because the producer doesn't have to waste its cosmetic's material for testing and instead can lend the consumer this product for trial of the cosmetics. In order to develop this solution the likes of Computer Vision in Python and Flask Framework for Web development have to be used**.**

I. INTRODUCTION

Even though the cosmetics industry has been digitizing by providing applications for buying cosmetics products at the comfort of home but there is no such confidence if the required product is going to suffice the need of the consumer. The work in the field of Computer Vision has been extensively done for detecting faces[1] and extracting features from it. Facial recognition systems were developed for security and surveillance systems. Now various applications of Computer vision lies in Automatic Inspection, Controlling processes, Navigation apart from Security and Surveillance.The work has been done previously just for the sake of Security and Surveillance, whereas the technology of Computer Vision and Augmented Reality can be used in the Cosmetics Industry which has a very great potential for future endeavors. Some applications were developed which can tell the approximate age and current mood of the user. To bridge this technical gap the idea was provoked to create an end solution which would help the user to try different cosmetics at the comfort of the user

II. LITERATURE SURVEY

The main research was done for detection of face and possibly determining the contour region inside the face. The work done by the open source community in Computer Vision has been referred to. OpenCV[2] is a python and C++ library developed by the Open source community which has pre trained classifiers which help in processing the frames for determining the face and related contours.

Documentation of OpenCV[2] has been done for various kinds of blurs, filters and Image segmentation techniques. DLIB is an open source library containing machine learning algorithms and tools. Specific algorithms like YOLO (which uses the likes of pretrained models and frameworks such as Tensorflow, Torch, Darkent and Caffe) are designed to detect objects. Face landmarks detection has been carried out by Ensemble Regression trees using algorithms devised by Vahid Kazemi and Josephine Sullivan[1].

III. BLOCK DIAGRAM

The basic building blocks of the whole program can be shown as follows.

The important elements are

- Input Video: Input Video will be taken by the laptop's webcam. The OpenCV[2] will be used to call the camera and use those images for further image processing.
- Face Detection Model: The trained model will be used to detect the face and further features will also be extracted like eyes and lips.
- Feature Detection model: The model is trained on the facial landmarks which will extract the required landmarks from the given video frames.
- Adding the layers of colors which are selected by the user. These will be thus sent to display as final images.
- Output Video: The processed output image frames are displayed on a user UI.

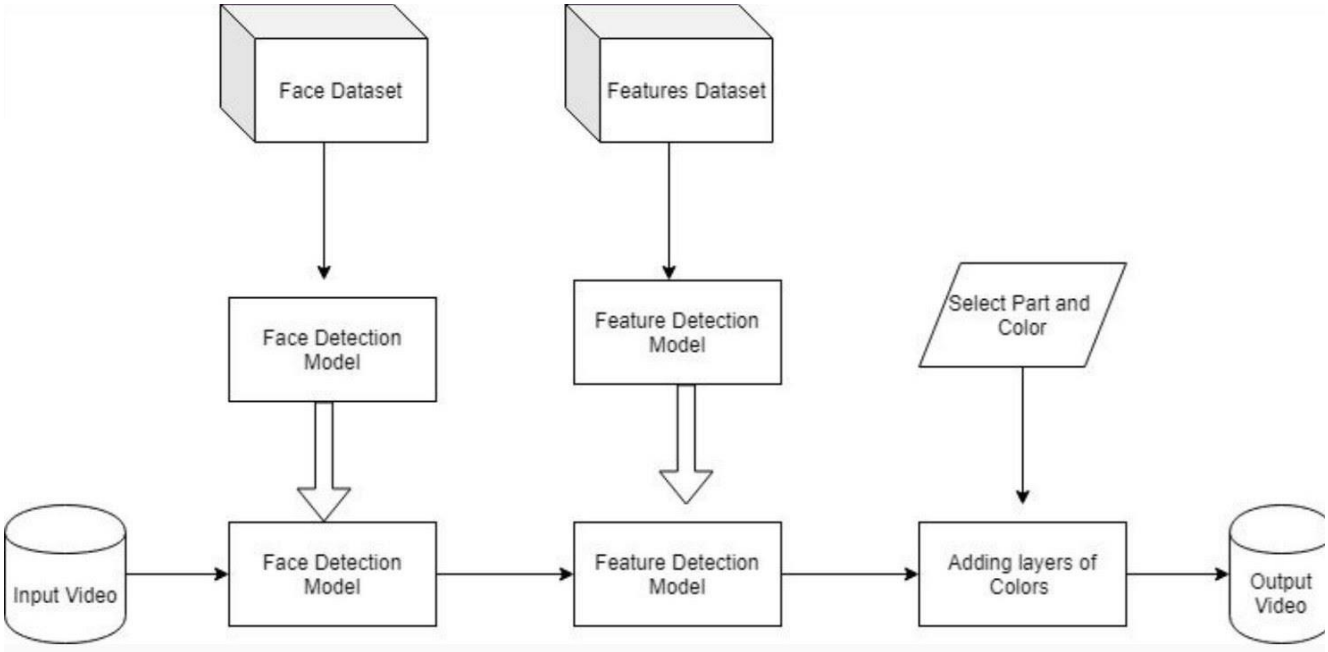

Figure 3.1: System Architecture

IV. FLOW CHART

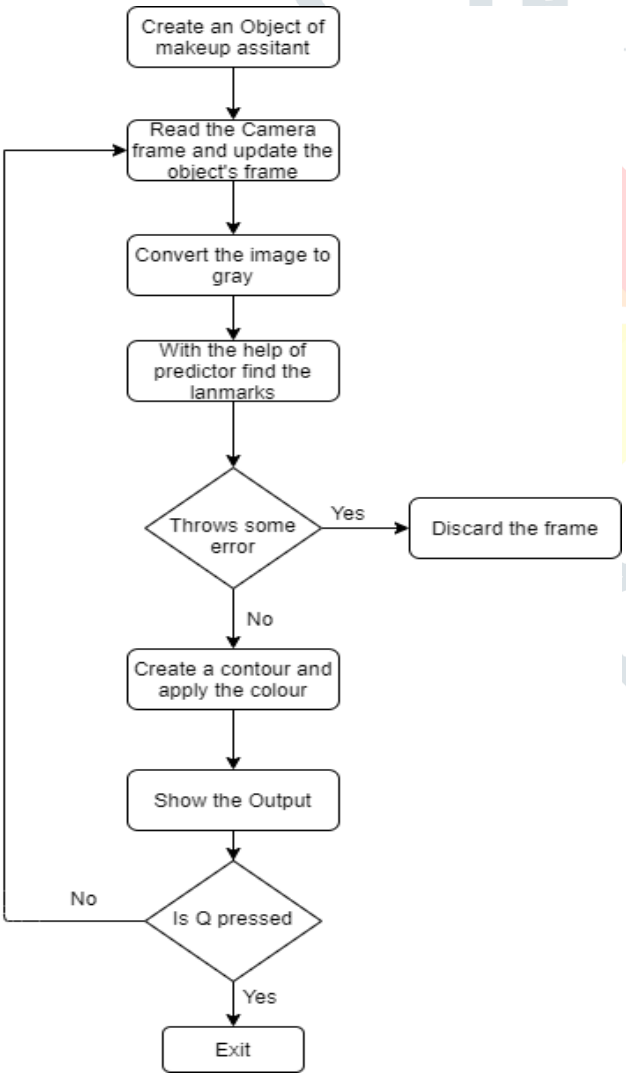

Figure 4.1: Flow chart

The basic building blocks of the whole program can be shown.

- The images will be read by camera and an object for the assistant would be initiated in the program
- The frame would be continuously read and updated as the object's frame.
- The predictor will be used on the converted gray frame to detect the landmarks inside the Face ROI.
- If it throws some error then the frame will be discarded else a contour would be made out of the landmarks.
- Finally the colour would be augmented and the output would be shown on a UI.

V. IMPLEMENTATION, TESTING AND DEBUGGING

The paper[1] is presented with novel an algorithm which boasts millisecond face alignment and the accuracy is achieved greater or comparable to the state-of-the-art methods.The increased speed is due to the recognition of the basic parts of previous face arrangements and then organising it in a streamlined definition over a course of high relapse capacities which is learnt via inclination boosting. In other papers[3, 4], that face arrangement can be illuminated with a course of relapse capacities. For our

situation every relapse work in the course proficiently appraises the shape from an underlying assessment and the powers of a scanty arrangement of pixels listed comparative with this underlying evaluation. Our work expands on the enormous measure of research throughout the most recent decade that has brought about critical advancement for face arrangement [5, 6]. In particular,we join into our educated relapse capacities two key components that are available in a few of the effective calculations referred to.

The significant commitments of this paper[1] are:

● An epic strategy for arrangement dependent on the outfit of relapse trees that

performs shape invariant element choice while limiting a similar misfortune work

during preparing time as we need to limit at test time.

● Presentation of a characteristic expansion of our technique that handles absent or

unsure marks.

● Quantitative and subjective outcomes are introduced that affirm that our technique creates excellent expectations while being considerably more productive than the best past technique (Fig 5.1)

● The impact of the amount of preparing information, utilization of halfway marked information and orchestrated information on the nature of forecasts are broken Down.

Fig 5.1: Examples of past techniques

Let S be a vector such that

S= $(x_1^T, x_2^T, x_3^T, \dots, x_p^T)^T \in R^{2p}$

The S will denote all the facial landmark's coordinates of the image.

The basic purpose of the course is that the regressor rt makes its forecasts dependent on highlights, for example, pixel force esteems, registered from I and listed comparative with the present shape gauge Sˆ(t). This presents some type of geometric invariance into the procedure and as the course continues one can be increasingly sure that an exact semantic area on the face is being listed. Later we portray how this ordering is performed.

Note that the scope of yields extended by the troupe is guaranteed to lie in a direct subspace of preparing information if the underlying evaluation Sˆ(0) has a place with this space. We thus don't have to implement extra imperatives on the forecasts which incredibly improves our technique. The underlying shape can basically be picked as the mean state of the preparation information focused and scaled by the bouncing box yield of a conventional face finder.

To prepare each rt we utilize the slope tree boosting calculation with an aggregate of square mistake misfortune as portrayed in [7].

The model was developed as a Dat file with the help of DLIB's documentation based on paper[1] trained and tested on ibug dataset[8] which makes use of papers[9]. The model was called in a python program and the camera images were read using opencv's python module[2]. The read images were converted to gray scale to consume less resources while doing image processing. After obtaining the gray level images, these images were passed to the trained facial landmark detector to obtain the landmarks.

With the help of scipy's interpolation function in 1D[10] the landmarks were interpolated to obtain all the points in between and thus helpful to make the contours.

A different class with the name of assistant was created to streamline the flow of the program and passing of the images. Finally in the detected contours the colours were augmented and the final output was displayed keeping some transparency of the contour background such that it doesn't look bold(Fig 5.2).

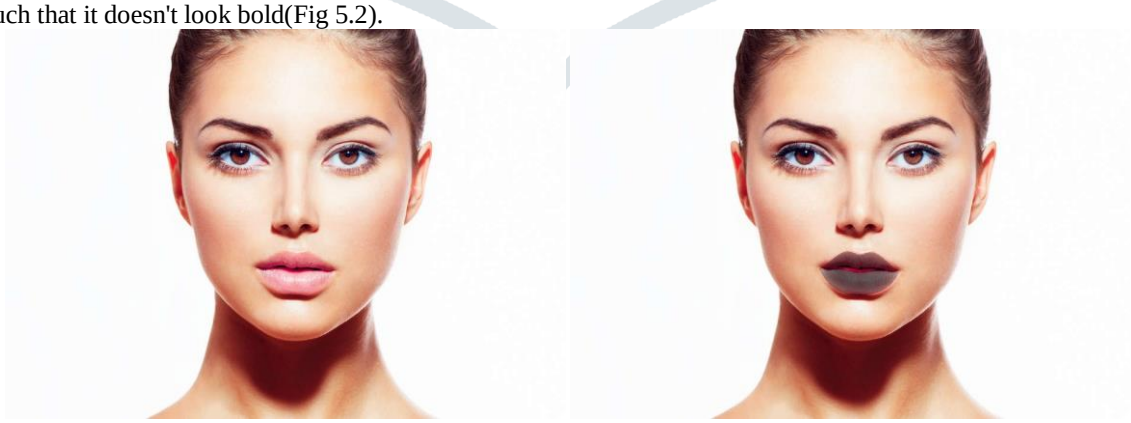

Figure 5.2 Input and Output Images

Different ways of detection of faces were tested during the initial implementation of the project. HaarCascade is one such classifier which successfully detects facial regions. Object Detection is an innovation identified with computer vision, picture preparing and profound discovery that manages distinguishing occurrences of items in pictures and recordings. Haar Cascade classifiers are a powerful path for object identification. This strategy was suggested in a paper by Michael Jones and Paul Viola 'Rapid Object Detection utilizing a Boosted Cascade of Simple Features'[11] .Haar Cascade is an AI based methodology where a ton of positive and negative pictures are utilized to prepare the classifier. Initially detection of faces were done using the same said Haarcascade classifier.

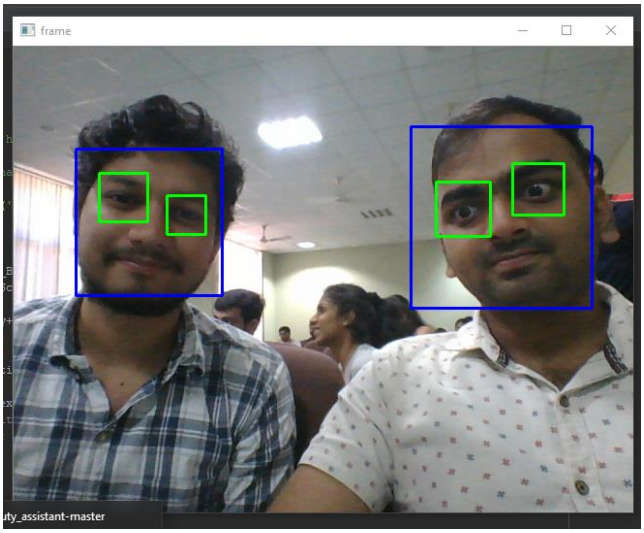

Figure 5.3: HaarCascade Classifier Output

The eyes and faces were successfully detected during initial development as shown in the Fig 5.3.

With improved knowledge of computer vision, DLIB's dependency was used to develop a trained model on a dataset[8]. This model was trained on 1000 images and was tested on approximately 2000 images of the same dataset. Due to the machine's limited capability the model was trained on 1000 images.

The dataset contains images which are annotated as 68 landmarks coordinates (Fig 5.4). Due to the project's requirement the model was trained on the selected landmarks starting from 48 to 67. Thus a trained model was generated with a size of approximately 29 mb.

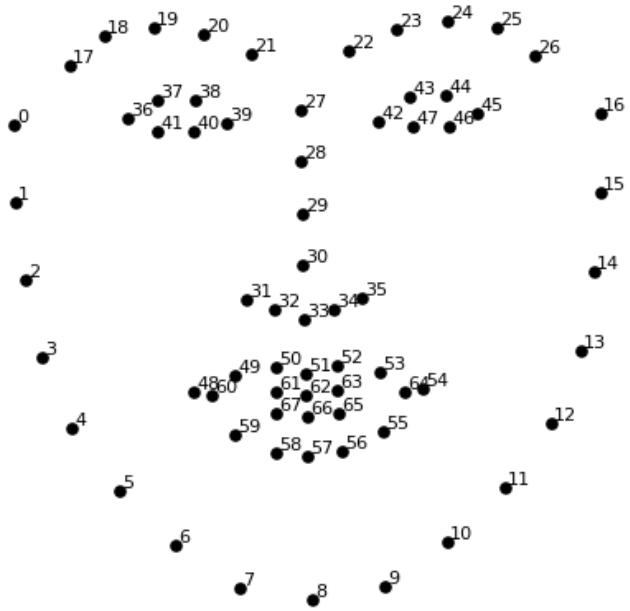

Figure 5.4: All 68 Facial landmark's coordinates

VI. RESULTS

The model is trained on 1000 images of the dataset[8]. Thus it is found to be less accurate than the model trained on all of the images. The comparison of the model's accuracy with respect to the complete trained model is shown in table 6.1 The error which are observed are the Mean absolute error (MAE) which tell the absolute difference of the calculated landmarks points with respect to the original coordinates.

	Completely Trained Model	Limited Trained custom model
No of images	6666	1000
MAE	3.631152	17.783082

Table 6.1 Comparison of Mode

Also the model was tested on the training and testing data in size of 500 images per testing. The observations for error recorded are shown in the following tables.

Data (Training data)	Mean absolute Error (MAE)
1st 500 Training Images	17.783082279172792
Next 500 Training Images	16.349184602602747
Next 500 Training Images	17.55948941753158
Average Training Error	17.2305821331

Table 6.2 Mean absolute Error on Training Data-1

Thus the average error obtained on the training data is 17.23%.

Data (Testing data)	Mean absolute Error (MAE)
1st 500 Testing Images	17.08640841541962
Next 500 Testing Images	18.638506187661584
Average Testing Images	17.8624573015

Table 6.3 Mean absolute Error on Testing Data-2

Thus the average error obtained on the testing data is 17.86%.

Also the web UI is found at 127.0.0.1 art port 5000 (Fig 6.6). The port address can be changed manually as well.

```
G:\cloud_deployement\final>python main.py
 * Serving Flask app "main" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [09/Apr/2020 04:42:20] "?[37mGET / HTTP/1.1?[0m" 200 -
127.0.0.1 - - [09/Apr/2020 04:42:20] "?[37mGET /video_feed HTTP/1.1?[0m" 200 -

G:\cloud_deployement\final>
```

Figure 6.1: Terminal's Output

VII. Conclusion

A program has been developed for reading the camera frames from a webcam. The webcam will read the images in a matrix and will store it as a variable. Then the image will be passed through a detector. This detector will detect the face if present inside the image frame and successively detect the region of lips as a contour. A program has been developed for detecting the same in python using OpenCV. Also the web's UI has been developed using HTML, CSS and Javascript. Flask is used as a web framework to connect python's backend with the HTML's frontend.

In order to achieve the above said results we have learnt the concepts of image processing and web development. Topics like filters, blurs, color model conversions were studied and the same were implemented in a program. Web development in HTML was studied. Machine learning using DLIB was studied and implemented. Various algorithms like Ensemble Regression Trees were studied and implemented in order to get a trained model.

Successful face detection and eyes detection has been done which has laid the groundwork for next steps in the project. Finally the colours have been augmented on the detected contours and the processed output is shown on the web's UI.

The system can be used in various Cosmetics Stores, Malls and Online Merchants where this can be a boom for digitizing the cosmetics industry. It is going to provide ease for the consumers as well as the producers and sellers.

VIII. Acknowledgement

We would express our gratitude towards our mentor, Dr. R. C. Jaiswal for being of great support and guiding us through the research. He gave this paper the insight and the expertise it needed for making it a presentable one. His advice, professional acumen, and encouragement proved to be a valuable guidance.

References

[1] Kazemi, Vahid & Sullivan, Josephine. (2014). One Millisecond Face Alignment with an Ensemble of Regression Trees. 10.13140/2.1.1212.2243

[2] Opencv.org. 2020. Opencv. [online] Available at: <http://opencv.org/> [Accessed 9 April 2020]

[3] P. Dollar, P. Welinder, and P. Perona. Cascaded pose regres- ´ sion. In CVPR, pages 1078–1085, 2010

[4] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. In CVPR, pages 2887–2894, 2012
[5] G. J. Edwards, T. F. Cootes, and C. J. Taylor. Advances in active appearance models. In ICCV, pages 137–142, 1999
[6] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models-their training and application. Computer Vision and Image Understanding, 61(1):38–59, 1995
[7] T. Hastie, R. Tibshirani, and J. H. Friedman. The elements of statistical learning: data mining, inference, and prediction. New York: Springer-Verlag, 2001
[8] 30 Ibug.doc.ic.ac.uk. 2020. I·Bug - Resources - 300 Faces In-The-Wild Challenge (300-W), ICCV 2013. [online] Available at: <https://ibug.doc.ic.ac.uk/resources/300-W/> [Accessed 9 April 2020]
[9] C. Sagonas, E. Antonakos, G, Tzimiropoulos, S. Zafeiriou, M. Pantic. 300 faces In-the-wild challenge: Database and results. Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation "In-The-Wild". 2016
[10] Docs.scipy.org. 2020. Interpolation (Scipy.Interpolate) — Scipy V1.4.1 Reference Guide. [online] Available at: <https://docs.scipy.org/doc/scipy/reference/interpolate.html> [Accessed 8 April 2020]
[11] Viola, Paul & Jones, Michael. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE Conf Comput Vis Pattern Recognit. 1. I-511.10.1109/CVPR.2001.990517