SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES
VOL. 3, NO. 3, DECEMBER 2020
DOI: 10.35377/saucis.03.03.730505

Adding Virtual Objects to Realtime Face Images; A Case Study in Augmented Reality

Çağla Ediz[1]
[1]Sakarya University, Department of Management Information Systems, Turkey; cediz@sakarya.edu.tr; +90 505 493 79 92

Received 1 May 2020; Revised 7 August 2020; Accepted 7 October 2020; Published online 30 December 2020

Abstract

Augmented reality applications related with faces such as make-up, hair design, wearing glasses are mostly prepared for entertainment purposes. Facilitating the preparation of augmented reality applications and more accurate analysis of real-world data in applications will enable these applications to be used more widely in different sectors such as R&D, education and marketing. In generally, the steps in image-based augmented reality applications can be listed as follows; detection of the targeted object, finding two reference points for each targeted object in 2D images, determining the boundaries of virtual object in its image and inserting the virtual object in real time. In this study, the problems that may be encountered in preparations of these augmented reality applications expected to be used more in the future are examined through a case study. Firstly, haar cascade classifiers, used to find different face areas, are compared and as a result of the comparison, it is decided to use eye haar cascade. Afterwards, rule-based approaches have been used to eliminate the wrong ones among the found eyes and to match the eyes of the same face. Then the position, size and angle of the virtual object to be added are calculated and it is added to the face using affine transformations. The problems encountered in augmented reality and algorithms used for problem solving are explained through the virtual hat application, but these simply prepared algorithms, can be used for different objects such as hair and glasses by changing the target points.

Keywords: augmented reality, image processing, affine transformation, opencv, virtual object, haar cascade.

Gerçek Zamanlı Yüz Görüntülerine Sanal Nesneler Eklenmesi; Artırılmış Gerçeklik Üzerine Bir Örnek Çalışma

Öz

Makyaj, saç tasarımı, gözlük takma gibi yüzlerle ilgili artırılmış gerçeklik uygulamaları çoğunlukla eğlence amaçlı hazırlanmaktadır. Artırılmış gerçeklik uygulamalarının hazırlanmasının kolaylaşması ve uygulamalardaki gerçek dünyaya ait verilerin daha doğru analiz edilebilmesi, bu uygulamaların Ar-Ge, eğitim ve pazarlama gibi farklı sektörlerde daha yaygın olarak kullanılmasını sağlayacaktır. Genel olarak görüntü tabanlı artırılmış gerçeklik uygulamalarındaki adımlar; görüntülerde bulunması hedeflenen nesnelerin tespiti- edilmesi, 2D görüntülerde hedeflenen her nesne için iki referans noktasının bulunması, eklenmesi istenen nesneye ait görüntüdeki sanal nesnenin sınırlarının belirlenmesi ve sanal nesnenin gerçek zamanlı olarak yerleştirilmesi şeklinde sıralanabilir. Bu makalede, gelecekte daha fazla kullanılması beklenen bu artırılmış gerçeklik uygulamalarının hazırlanmasında karşılaşılabilecek problemler bir örnek olay üzerinden incelenmiştir. İlk olarak, farklı yüz alanlarını bulmak için kullanılan haar cascade sınıflandırıcıları karşılaştırılmış ve karşılaştırma sonucunda göz bölgesi haar cascade sınıflandırıcısı kullanılmasına karar verilmiştir. Bulunan gözler arasından yanlış olanların elenmesi ve aynı yüze ait gözlerin eşleştirilmesi için kural tabanlı yaklaşımlardan faydalanılmıştır. Daha sonra eklenecek sanal nesnenin pozisyonu, boyutu ve açısı hesaplanmakta ve afin dönüşüm yöntemleri kullanılarak yüzde istenen bölge görüntüsüne eklenmektedir. Arttırılmış gerçeklikte karşılaşılan problemler ve problem çözümü için kullanılan algoritmalar sanal şapka uygulaması üzerinden anlatılmıştır, ancak yalın olarak hazırlanan bu algoritmalar hedef noktalardeğiştirilerek saç, gözlük gibi farklı objeler için de kullanılabilir.

Anahtar Kelimeler: artırılmış gerçeklik, görüntü işleme, afin dönüşüm, opencv, sanal nesne, haar cascade

1. Introduction

In augmented reality studies, objects in the physical world such as image and sound are processed and presented in a different way in real time. In these applications, pointers are often used to introduce desired objects. With the increase in the processing level of computers, there is a transition from fixed image pointers to applications where targets are found by using trained object features [1]. The algorithms in these applications recognize targeted objects such as face, hand, 2D barcode image on camera images. Then the target points in these images are found and combined with virtualimages [2]. If the targeted points are in the facial area, different detection methods can be used. Wang et al. (2018) divided these methods into two groups basically; parametric shape basic model and nonparametric shape model [3]. The haar cascade classifier used in this study is in the non-parametric shape model based method group. Haar cascade algorithms are most commonly used algorithms for detecting areas such as face, eyes and ears in the photographs. The haar cascade algorithms used in object detection were developed by Viola and Jones and were presented for the first time at a conference in 2001 [4]. Viola and Jones used rectangular frames, painted in black and white, to detect facial areas in the image and found that different parts of the face are similar to the black and white frames created with different combinations. For this reason, they moved the black and white frames in different combinations and scales on the whole image and thus, determined the places similar to the black and white frames sought in the image. They also developed "integral display algorithm" to ensure that this process can be done quickly. Thus, they enabled the use of haar cascade algorithms on real-time images. Boosted cascade detectors developed using haar features represent the most advanced method of face detection [5]. Athough several boosted approaches are developed to find face points, most of them are slow because of their computationally costness [6]. In this study, a virtual reality study that can be used in real time with simple algorithms has been prepared and the problems encountered during the preparation stages have been examined.

The haar cascade algorithms in this study are run by using the Emgu CV.4.1.0 library in Visual Studio environment developed from open source OpenCV. Here, haar cascade algorithms detect the eyes. However, due to high error rates in the haar cascade detectors and the method presented in this study is prepared for two-dimensional frontal images, it is necessary to eliminate some of the eyes found with the algorithm and to identify the faces suitable for adding hats. After determining the faces, the findings obtained by averaging different human faces [7] are used to determine the position of the hat to be added to the image. While determining the hat position, the center points of the two eyes are combined with an imaginary line and thus the degree of rotation is calculated. Affine algorithms have been used for rotating and scaling processes. Afterwards, the difference between the hat color and the background color in the hat image has been used in order not to add the background parts in the hat image.

Augmented reality studies which are related with face have been used in different areas. Some of them are related with fashion like makeup [8, 9], wearing glasses [10, 11] or hair design [11, 12]. Some of them are studied for security needs. For example, driving warnings are done according to eye opennings [13] or the head poses monitored by using the vehicle cameras [14]. Also, augmented reality including facial detections, is used for entertainment in some studies. For instance, Peng (2015) proposed a development framework by using face detection in OpenCV and put on different funny face masks on frontal faces [15]. Another study for entertainment belongs to Mahmood et. al. (2017). They prepared an application showing the statistical data of the athlete on the screen by detecting and recognizing his/her face [16]. Different from augmented reality studies with face detections, this article is focused on the challengings in the development processes of augmented reality with facial detections. Also, it is not used any equipment in the study except computer and camera. It identifies targeted points using only the eye haar cascade algorithm and facial average shapes. In addition, this study proposes a simple method that can be easily applied for different virtual face area objects.

2. Limitations of the Study

The constraints in the study can be listed as follows:

- For proper working haar cascade algorithms, the eyes should not be closed and the number of pixels of the eye image should not be too small. Therefore, when the eyes were open and the person was not far from the camera, the eyes could be identified and the prepared application could work.
- In very dark and bright environments, performance of haar cascade algorithms decreases [17]. Therefore, lighting should be at a sufficient level in application environments.
- In the application, only two dimensional front view object images can be used.
- Since the front view object image is two-dimensional, the application will not work on the faces turning around the neck (yawing) and bent down or up (pitching).
- Since the front view object borders are created by using the color difference between itself and the background in the algorithm, the background should be plain and should not have the same colours with the object.
- It is assumed that the front view object image is adjusted to pass through the borders of itself.

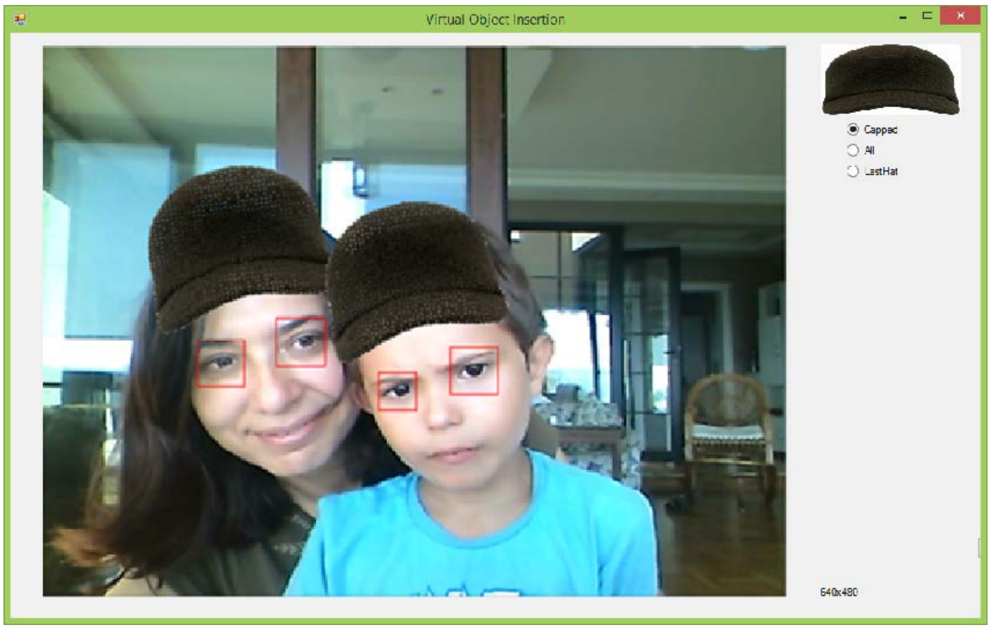

Figure 1 A Screenshot taken from the program

3. Encountered Problems for Hat Adding Example

The operations performed in the study can be grouped under three parts. These are;

- Finding the most suitable faces for wearing the loaded hat image among the people,
- Determining the place of the hat according to the position of the eyes on the selected faces and the angle of the axis passing through the eyes centers,
- Positioning the hat on the planned location by scaling and rotating it.
- These processes were attempted to be made in real time. The encountered problems and solutions developed in the study are eximined under the following headings.

3.1 Decision of Which Face Area to Use in Hat Positioning

In order to find the facial areas, the most known haar cascade patterns used to find the human facial areas. Eye, nose, mouth and frontial face profiles were found in the test photographs using haar cascade patterns and each facial area found was shown in different colors.

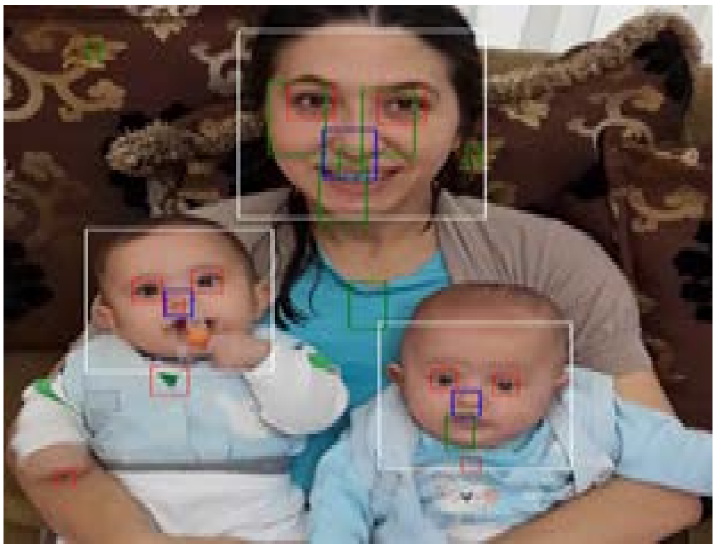

Figure 2 Eye (red), nose (blue), mouth (green) and front (white) frames obtained in an image by haar cascade detectors

The following determinations wereobtained with the haar cascade images obtained from the 30 test photos used:

- Although the front face haar cascade algorithms have high accurate, it has been observed that front frames are located in different places than front faces. In this case, front face haar cascade algorithms was not preferred to use.
- The eyes were not detected when the displayed faces became smaller. Since a notebook camera from near distance was used in this study, the eye area size was not considered to be a problem.
- While cascades other than mouth draw a frame to cover the area they are looking for, the area drawn by the haar cascade searching the mouth mostly covers the mouth area partially.

When more than one frame is found for the same face area, one of the frames is considered correct.

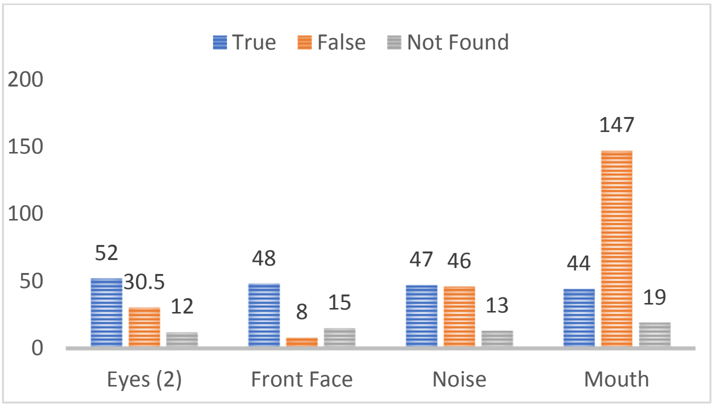

Figure 3 Correct detection, wrong detection and undetectable numbers of facial area

On 30 test photos, haar cascade algorithms for eyes, nose, mouth and frontial face were run and the eye haar cascade algorithm was observed with the highest accuracy rate (Fig. 3). For this reason, it was decided to work with eye haar cascade. Since the haar cascade algorithms scan every area of the images with patterns of different sizes, scanning time increases in proportion to the number of pixels in the image. The working time of haar cascade classifiers of the eye, nose, mouth and front face is four times greater than the working time of the eye haar cascade alone. For this reason, a second algorithm other than eye detection algorithm was not used to take a smoother screenshot from the camera. In this case, it is necessary to eliminate the eyes found wrong in the eye haar cascade application and to find the faces to be applied with various algorithms.

3.2 Eliminating Unsuitable Eyes

In haar cascade algorithms, incorrect detections can also be made besides correctly detected face areas. Haar cascade algorithms can correctly detect all faces in the photos if background is plain, and correct face detection rate reduced if the background is mixed [18, 19]. On the other hand, as mentioned in the previous section, haar cascade algorithms searching for eyes can also capture images which are not actually eyes. In this case, false eye detections should be eliminated with the algorithms prepared. In addition, the image of the hat used in the study is two-dimensional. For this reason, eyes that are not from frontial view should also be eliminated. After these eliminations, the eyes that are thought to belong to the same face are matched with the prepared algorithm.

In the study, with the eye haar cascade algorithms, the starting point, rectangle width and rectangle height of the rectangle drawn for each eye were found on the horizontal and vertical axis. By using these inputs, the corner points of the rectangle drawn for each eye were kept in memory and some analyses were performed by matching these rectangles in pairs. The processes for matching suitable eyes and eliminating unsuitable eyes have some rule-based algorithms described under following subheadings.

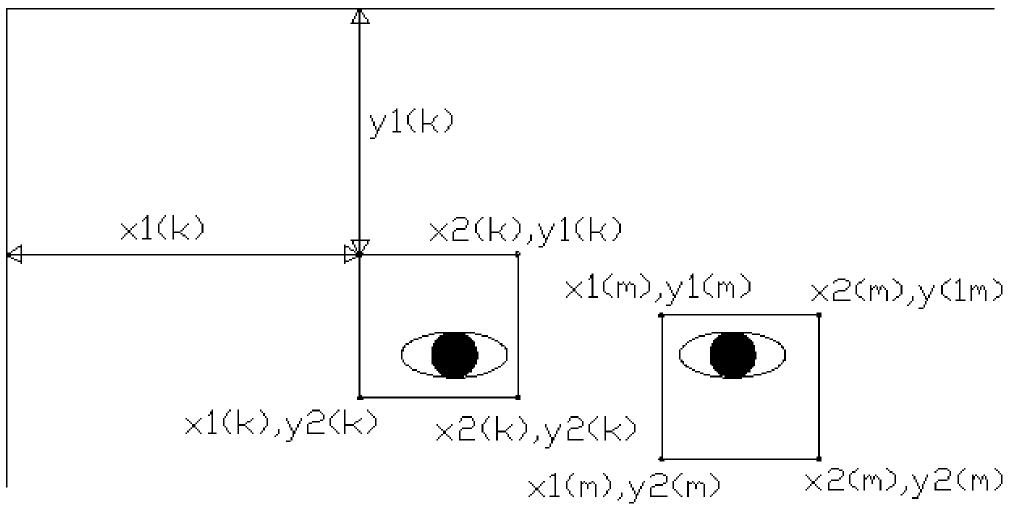

Figure 4 Coordinations of eye's rectangle corners

3.3.1 Examining the position heights of the eyes in the image

When looking at a face image from the front, both eyes are expected to be close to each other on the vertical axis. Therefore, if there is more than twice the height of an eye between the y values of the paired eyes, those eyes are not matched.

$$|(y1[k] - y1[m])| < |(y2[k] - y1[k])|.2 \quad (1)$$

3.3.2 Evaluation of the distance of two matching eyes from each other

Although two eyes are located close to each other on the vertical axis, these two eyes may not actually belong to the same face. If the ratio of the distance between two eyes to the width of one of these eyes is too high, these two eyes cannot belong to the same face. To measure this, the ratio of the distance of the eye regions to each other to one eye width is calculated and it is desired that this ratio is not more than 3 times.

$$|((x1[k] - x1[m]) \,/\, (x2[k] - x1[k]))| < 3 \quad (2)$$

3.3.3 Preventing eyes matching between multiple drawn rectangles for the same eye

Another common error with eye haar cascade is to be drawn two rectangles for the same eye. In order to prevent this error, it is necessary to eliminate the eye rectangles that intersect or cover each other. To fulfill this rule, if one of the paired eyes on the horizontal axis starts before the other ends, these two eyes are not matched.

$$\begin{gathered}(x1[k] < x1[m] \; and \; x2[k] < x1[m] \; and \; x2[k] < x2[m]) \\ or \\ (x1[m] < x1[k] \; and \; x2[m] < x1[k] \; and \; x2[m] < x2[k])\end{gathered} \tag{3}$$

3.3.4 Elimination of faces seen from the side view

Since two-dimensional front-facing hats were used in the application, it was requested to eliminate the faces rotating around the neck (yawing) in the application. On the face that has been turned around the neck, the area of the eye close to the camera is larger, while the other eye away from the camera is smaller. Depending on this, it was observed that haar cascade eye algorithms draw different sizes of rectangles for the two eyes of the face rotating around the neck. Two eye width ratios were evaluated to avoid processing on these faces. If the ratio between the two eyes selected is less than 0.5 or greater than 2 (there are two different possibilities as the wider rectangule can be in the numerator or denominator), these eyes are not matched.

$$\begin{gathered}(x2[m] - x1[m])/(x2[k] - x1[k]) < 2 \\ and \\ (x2[m] - x1[m])/(x2[k] - x1[k]) > (5/10)\end{gathered} \tag{4}$$

3.4 Perception of the Borders of the Hat

In images using the RGB model, all color values are obtained with a mixture of red (R), green (G) and blue (B) color. This model is based on the Cartesian coordinate system. Each point in the coordinate system is expressed in pixels and the color value of a pixel is expressed as C (R, G, B). When 8 bits are allocated for each color, all three colors take values between 0 and 255 [20]. Each of these three colors appears as black when it gets 0 and white when it gets 255. Pixels with equal R, G, and B values, will get a gray color according to their value. Pixels are converted to C (x, x, x) to convert the picture to black and white tones. This x value is calculated as follows [21].

$$X = (R + G + B)/3 \tag{5}$$

While adding a hat to the image in real time, the pixel values of the hat image are kept in memory as long as the application is running in order to reduce the time problem as much as possible. The colors covered by the hat are chosen differently from the hat background color, and by taking advantage of this color difference, the hat image is distinguished from the background image. The first grayscale pixel value of the top left corner of the hat was accepted as the background color of the hat and the grayscale value of each pixel in the hat image was calculated and compared with this background value. Considering that the background color may change slightly due to different reasons such as light fluctuations, the colors from the first pixel value to 20 units difference are accepted as the background color. A boolean array evaluating this has been created, and the pixels within the boundary of the hat are assigned the value of “true” and the areas outside are assigned the value of “false”.

3.5 Calculating the Hat Position

In 2011, the Face Research Lab of Aberdeen University in Scotland developed an interactive online web application to combine different human faces with an average image. When looking at the average face image created by this software, it is seen that the Golden Ratios, an irrational mathematical constant that fascinated mathematicians 2,500 years ago, were found in the average face dimensions [7]. The golden ratio is calculated from (1 + √5) / 2 to approximately 1.618.

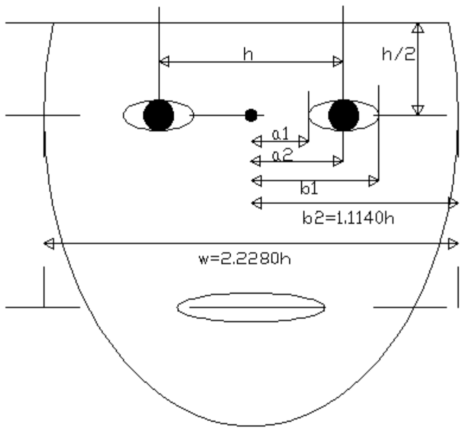

Figure 5 Golden ratios used for hat position in the study [7]

Using the golden ratios between a1-a2 and b1-b2 in Figure 5 and assuming the facial areas are symmetrical, the hat position can be approximately determined. In determining the position of the hat, the coordinates of the two eyes are taken as input and as a result, two referance points for each hat are reached.

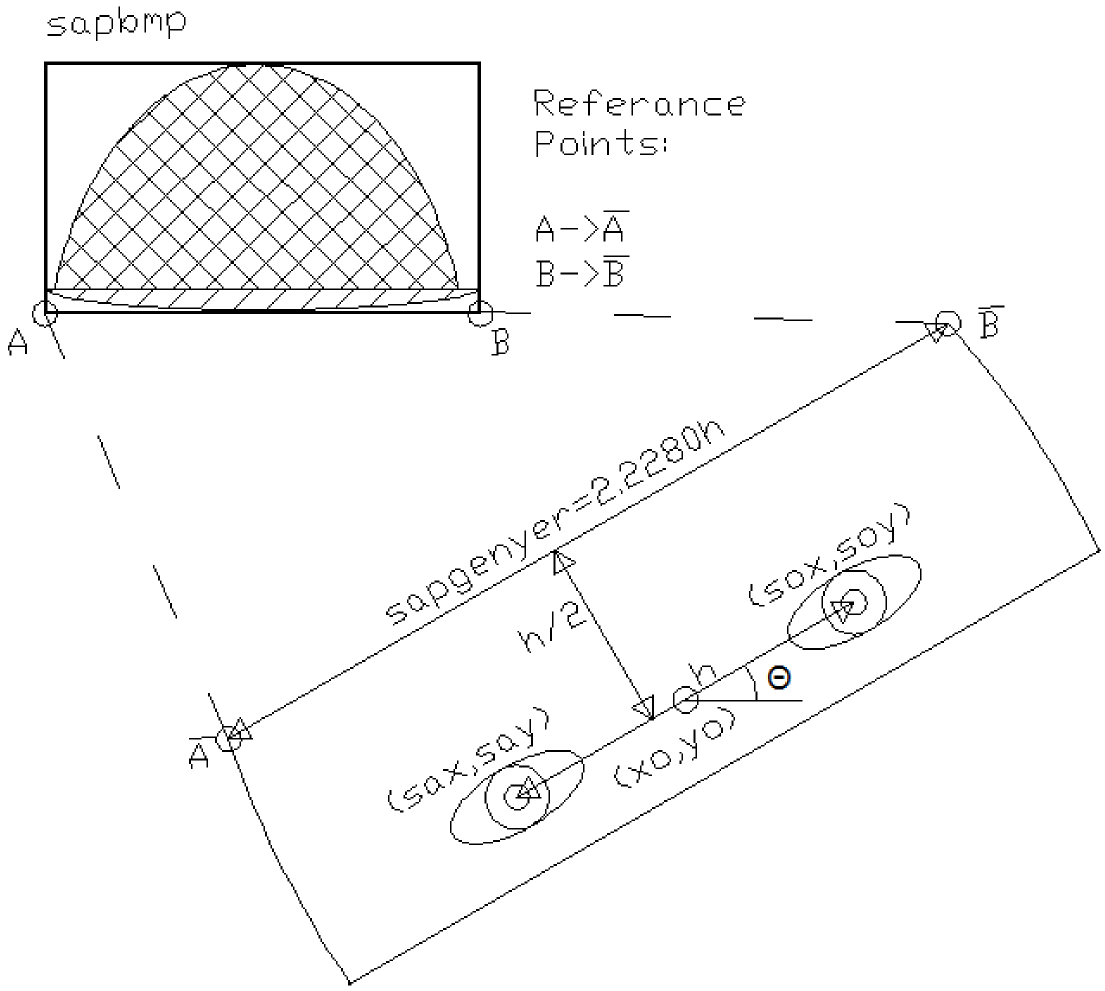

Figure 6 Variable images used in object placement commands

3.5.1 Scaling the size of the hat according to the head area

In the study, as it is aimed to reach uninterrupted images in real time, "affine transform" having less calculation is used in image scaling instead of pixel interpolations. Thus, the pixel color values were not calculated separately, the unit pixel values in the picture were used only by changing quantities. The bottom width of the hats iare assumed to be equal to the top width of the head. So for hat example explained in this case study, it is not suitable for wide-brimmed hats such as fedora.

The affine matrix (Tsc) for scaling is as follows [22; 23]:

$$[x\ y\ 1] = [v\ w\ 1]\,Tsc = [v\ w\ 1]\begin{vmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{vmatrix} \tag{6}$$

Using this equation, the image size changes by assigning the pixel values at the v and w positions to the x and y coordinates with coefficients c_x and c_y.

3.5.2 Adjusting the angle of the hat to the angle of the face

On the other hand, the affine transform used to rotate the image at a certain angle is as follows:

$$[x\ y\ 1] = [v\ w\ 1]\,Trot = [v\ w\ 1]\begin{vmatrix} Cos\,\alpha & Sin\,\alpha & 0 \\ -Sin\,\alpha & Cos\,\alpha & 0 \\ 0 & 0 & 1 \end{vmatrix} \tag{7}$$

3.5.3 Hat borders sticking out of the image to which it was added

After determining the faces that are suitable for wearing a hat, the hat is sized according to where it will be located and it is added to the image by making the necessary rotating operation. While doing this, some parts of the hat may have gone beyond the bounds of the image. In this case, it is necessary to carry out border checks so that the software prepared does not fail. So, if the position values that need to be added to the hat take a negative value, or if the position values of the hat are larger than the width or height of the image, these images are not added to the image. To show this situation, some famous faces are selected and hats are added in the Kinship Verification Database (Figure 7).

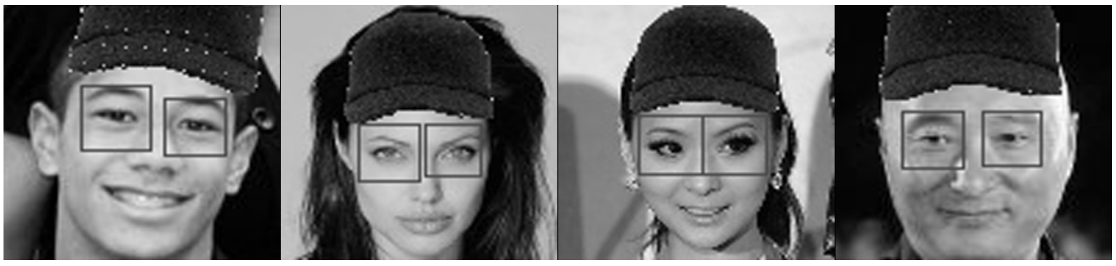
Figure 7 Images with virtual hats from Kinship Verification Database [24]

3.6 Object Placement Codes

The processes for Object Placement are (Partial C# codes):

- Finding the width and height of the object to be added (sapgenyer and sy);
- Calculation of cos and sin values of object rotation angle (cosQ and sinQ)
- Determination of the location coordinates to be added (xbr, ybr)
- Examining whether every pixel in the visual object exceeds the image boundaries,
- Finding the hat image coordinates corresponding to the place to be added (ssx, ssy)
- Are hat image coordinates in hat boundaries (sapbool (sx, sy) == true)
- Adding.

Partial C# Codes 1: (8)

```
sapgenyer = 2.3 * h;
orangen = Convert.ToDouble(sapbmp.Width) / sapgenyer;
oranyuk = Convert.ToDouble(sapbmp.Height) / Convert.ToDouble(sapbmp.Width);
sy = oranyuk * sapgenyer;
cosQ = (sax - sox) / h;
sinQ = (say - soy) / h;
    for (int K = 1; K < sapgenyer-1; K++)
    {
    for (int M = 1; M < sy - 1; M++)
         {
             ssx = Convert.ToInt32(K * orangen);
            ssy = Convert.ToInt32(M * orangen);
            if (sapbool[ssx, ssy] == true)
            {
               dy = (h / 2 + sy - M);
               dx = (K - sapgenyer / 2);
               xbr = xo + Convert.ToInt32(Math.Ceiling(dy * sinQ - cosQ *dx));
               ybr = yo - Convert.ToInt32(dx * sinQ + dy * cosQ);
               if (xbr < imageframe.Bitmap.Width && ybr < imageframe.Bitmap.Height
                                      && xbr > 0 && ybr > 0)
               { imageframe.Bitmap.SetPixel(xbr, ybr, sapbmp.GetPixel(ssx, ssy)); }
              }
            }
       }
```

3.7 Problems with Real Time Rendering Images

In the prepared application, hats are added to the real-time images obtained from the camera and the user is shown these processed images. The frame per second (fps) taken by camera value is 30. Hovewer, the seen fps value is not 30 fps. Because, the frame is shown after hat adding process. This process is taking times aproximately 0.3 sec by the notebook (*Intel® Core™ i5-2450M CPU @ 2.50GHz*) This value isn't enough, but it is better than the previous tests made in this study. To reach this value, only one haar cascade used. In addition, if none of the faces are in a suitable position to wear the hat photographed from the front image, the last processed image is displayed on the screen and only after these people come into suitable position, the image changes. So, in this case fps will be more lower than the avarage value.

4. Conclusion

Augmented reality studies are carried out by adding virtual images on images obtained from cameras. In this study, an application is developed by preparing simple algorithms in order to determine target objects correctly, to set correct virtual object position and to add different virtual objects to the image. In the developed application, it is aimed to display these images in real time. Step by step, the methods prepared by making comparisons are explained and the points to be improved are argued. Thus, a simple method prepared for augmented reality studies, allowing adding different objects to the images, is presented by determining the sides needing improvement.

In this study, virtual hat images were added to people in real-time camera images. After comparing the cascade algorithms detecting different face areas, the eye haar cascade was selected. Then the eyes of the same faces were matched to each other and their suitability was evaluated. In addition, using the

Table 1 Aims, steps, problems and some solutions of these problems

AIM	STEPS	PROBLEMS	SOLUTION
Detection of eyes	Using Haar cascade and EmguCV library to capture video images and to detect eyes	Wrong detections	Mismatched eyes are eliminated while eyes that are thought to belong to the same face are matched
		Eyes not detected	-
		Re-detection of the same eye	Evaluation of the intersection of the eye frames
Detection of faces	Determining the eyes belonging to the same face.	Incorrect matching of the eyes	Evaluation of horizontal eye distances relative to eye size
			Evaluation of vertical eye distances relative to eye size
Finding the face in the suitable position	Evaluation of the suitability of the face position	Face looking rear	Comparing two eye sizes
		Face looking down or up	-
Finding the hat position	Determining two base points on which the hat will be positioned	Personal and style differences in face	-
		Rolling of the face	Finding the axis angle through the matching two eye centers
Finding hat image	Separation of the hat image and the background image by using color differences	The color nearness between hat and background	Choosing different hat backcolour from hat colours
Adding hat image	Scaling the hat according to the head width	Inappropriateness of the size of the hat	Scaling size of the hat according to the size of placement location
		The disappearance of hat patterns from the smallness of the place to add a hat	-
	Positioning the hat according to the angle of rolling rotation	Unable to assign pixels to some coordinates in the added hat	-
	Add the hat to the image	Head's wearing position cannot be determined for different types of hats	-
	Adding hat's pixel values to the image	The coordinates of hat pixel can be out of image	Evaluation assigned coordinates of hat pixels
Real-time rendering of the image	Repetition of steps	Image continuity not being achieved	One haar cascasde classifier is used, Rule based algorithms are used to eliminate wrong eyes and to match the eyes of the same face, Color difference is used to separate hat and backcolour

golden ratios of the average face and eye positions, a virtual object with the required angle and size was added to the images. Even though the problems encountered in adding a virtual hat are presented as a case study here, the prepared algorithms can be also used for different objects by changing the position and size.

Although performing well when adding virtual objects to real-time images, some pointst need improvement. First, when rotating the head to the side, each pixel value in the cartesian coordinate system of the hat image is assigned to a new position based on the angle of rotation. As a result of this, it is possible to skip some points ensuring continuity in new location coordinates. These deficiencies caused various patterns according to the angle of rotation on the hats. In the future, to prevent transparency on the virtual hat, an algorithm controlling continuity and assigning value to the pixels can be added to this application.

Another issue that needs to be improved is the need for a program to take action by evaluating the result of personal differences. In the study, the position of the hat is determined on the average face size. However, the width of eyes, face and forehead varies from person to person. In addition, many parameters such as the bulk of the hair, style of hair, shortness of hair, length, curly or straight hair can affect the hat position. Additional algorithms that will position the hat by evaluating these parameters will improve the study. Moreover, the scope of this study can be expanded by algorithms determining the angles of pitching and yawing of the head by using three-dimensional hat in future applications.

References

[1] P. Chen, Z. Peng,., D. Li, and L.Yang, , "An improved augmented reality system based on AndAR", *Journal of Visual Communication and Image Representation*, 37, 63-69, 2016.

[2] T. İçten and B. A. L Güngör, "Artırılmış gerçeklik üzerine son gelişmelerin ve uygulamaların incelenmesi", *Gazi Üniversitesi Fen Bilimleri Dergisi Part C: Tasarım ve Teknoloji*, 5(2), 111-136, 2017.

[3] N. Wang, , X. Gao, D. Tao, H.Yang and X. Li, "Facial feature point detection: A comprehensive survey", *Neurocomputing*, 275, 50-65, 2018.

[4] P Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", In *Proceedings Of The 2001 IEEE Computer Society Conference On Computer Vision And Pattern Recognitio*n, CVPR 2001 (Vol. 1, pp. I-I). IEEE, December, 2001.

[5] E. Skodras and N. Fakotakis, "Precise localization of eye centers in low resolution color images", *Image and Vision Computing*, 36, 51-60, 2015.

[6] D. Chen, S. Ren, Y.Wei, X. Cao, and J. Sun, "Joint cascade face detection and alignment", In *European Conference On Computer Vision*, pp. 109-122, Springer, Cham, September, 2014.

[7] V. Schwind, "The golden ratio in 3D human face modeling", *Stuttgart Media University*, Stuttgart, 2011.

[8] A. M. Rahman, T. T Tran, S.A. Hossain and A. El Saddik, "Augmented rendering of makeup features in a smart interactive mirror system for decision support in cosmetic products selection", In *2010 IEEE/ACM 14th International Symposium on Distributed Simulation and Real Time Application*s, pp. 203-206, IEEE, October, 2010.

[9] A. Javornik, Y. Rogers, A. M. Moutinho and R.Freeman, "Revealing the shopper experience of using a" magic mirror" augmented reality make-up application", In *Conference on designing interactive systems*, Vol. 2016, pp. 871-882, Association for Computing Machinery (ACM), 2016.

[10] Z Feng, F. Jiang and R.Shen, "Virtual Glasses Try-on Based on Large Pose Estimation", *Procedia Computer Science*, 131, 226-233, 2018.

[11] J. J. Lv, X. H. Shao, J. S. Huang, X. D. Zhou and X. Zhou, "Data augmentation for face recognition", *Neurocomputing*, 230, 184-196, 2017.

[12] M. Kim and K. Cheeyong, "Augmented reality fashion apparel simulation using a magic mirror", *International Journal Of Smart Home*, 9(2), 169-178, 2015.

[13] K. Diaz-Chito, A. Hernández-Sabaté and A. M. López, "A reduced feature set for driver head pose estimation", *Applied Soft Computing*, 45, 98-107, 2016.

[14] E. Murphy-Chutorian and M. M. Trivedi, "Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness", *IEEE Transactions on intelligent transportation systems*, *11*(2), 300-311, 2010.

[15] H.Peng, "Application research on face detection technology based on OpenCV in mobile augmented reality", *International Journal of Signal Processing, Image Processing and Pattern Recognition*, *8*(2), 249-256, 2015.

[16] Z. Mahmood, T. Ali, N. Muhammad, N. Bibi, I. Shahzad and S. Azmat, "EAR: Enhanced Augmented Reality System for Sports Entertainment Applications", *KSII Transactions on Internet & Information Systems*, *11*(12), 2017.

[17] F.Pereira, C. Silva and M. Alves, "Virtual fitting room augmented reality techniques for e-commerce", *In International Conference On Enterprise Information Systems*, pp. 62-71, Springer, Berlin, Heidelberg, October, 2011.

[18] V Singh, V Shokeen and B Singh, "Face detection by haar cascade classifier with simple and complex backgrounds images using opencv implementation", *International Journal of Advanced Technology in Engineering and Science*, 1(12), 33-38. 2013.

[19] S. Soo, "Object detection using Haar-cascade Classifier", *Institute of Computer Science*, University of Tartu, 1-12, 2014.

[20] T.Kumar and K. Verma, "A Theory Based on Conversion of RGB image to Gray image", *International Journal of Computer Applications*, 7(2), 7-10, 2010.

[21] F.Patin, "An introduction to digital image processing", [Online]. Available: http://www. programmersheaven. com/articles/patin/ImageProc. pdf, 2003. [Accessed: 20-Jan-2020]

[22] G. Wolberg, " Digital image warping", Vol. 10662, pp. 90720-1264, Los Alamitos, *CA: IEEE Computer Society Press*, 1990.

[23] R. C. Gonzalez and R. E. Woods, *Digital image processing*, Pearson International Edition, Thid Edition, USA., 2008.

[24] R. Fang, K. D. Tang, N. Snavely and T. Chen, "Towards Computational Models of Kinship Verification", *IEEE International Conference on Image Processing*, Hong Kong, September 2010 (ICIP '10)