International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.9, No.1 (2016), pp.63-84
http://dx.doi.org/10.14257/ijsip.2016.9.1.07

# Augmented World: Real Time Gesture Based Image Processing Tool with Intel RealSense™ Technology

Rupam Das and Dr. K. B. Shivakumar

*Research Scholar, VTU, Belgavi,*
*Professor, Dept. Of Telecommunication*
*rupam.iics@gmail.com, kbssit@gmail.com*

## *Abstract*

*Intel RealSense™ is an exciting new technology that offers innovative multimodal human computer interaction with hand tracking, face tracking, emotion detection, speech synthesis and voice recognition. The technology supports real time background separation by combining the depth and RGB streams of RealSense™ camera. Several past works have provided robust techniques for background separation. Background separated frame is used to augment segmented user photo in an artificial scene. In this work we propose a novel real time technique for augmenting the user in an artificial scene and apply various image filters on the constructed scene. We also propose a "Split Grid" based novel hand gesture driven UI control system that enables the user an intuitive way of working with the system with on-air hand movement and gesture. The system can be used in a wide range of applications like puppetry, photo retouching, and video recording on augmented scene .Usability tests performed by both amateur as well as professional digital photo artists proves ease and efficiency of the tool in generating quick wonderful photo effect in multimodal environment.*

***Keywords**: Augmented Reality, Real Time Image Filters, Hand Gesture, Intel RealSense™.*

## 1. Introduction

Augmented Reality [1] has been one of the most fascinating areas research in recent times. Augmented reality enables virtual objects to render in a live scene which can then be moved and controlled by the user.

Gael Gordon[2] has shown how stereo and depth data can be effectively used for more realistic augmented reality experience.

When a composite scene is to be constructed with real and virtual content, handling the occlusion becomes very important. In the case of a film, such an occlusion can be handled offline in a studio and the objects can be rendered carefully in an offline environment. However, in the case of augmented reality, virtual as well as real objects are to be handled online [3].

In an augmented reality that renders users into an artificially constructed scene and controls certain object based on user movement and gestures, segmenting user from the background is an essential step. Jonathan Ventura [3] and Wang [4] emphasize on background separation with depth data. Wang's method uses a Time of Flight camera which has an IR sensor and RGB sensors acquiring simultaneous streams. These streams can then be projected to segment the user from the background. Ryan Crabb [5] also use Time of Flight (ToF) sensor based depth information to successfully achieve background separation. Enrique J [6] adopted the depth based information to propose a new technique for background separation with Microsoft Kinect technology.

ISSN: 2005-4254 IJSIP

Kyungnam Kim [7] suggested a technique for modeling background and then subtracting the same from the video frame to segment the foreground. Though this technique is simpler and does not require depth information, results show that the method cannot fully segment out complex background and that many residual objects are left behind. The efficiency of the method depends upon the uniformity of illumination as well as foreground and background color difference. Therefore RGB based segmentation is not ideal in an augmented reality. Even though Michael Van den Bergh [8] suggested a technique for modulating background information in order to obtain a better segmentation, the result also has noisy edges.

Therefore, it is obvious to use depth data for segmenting the user from the background for an effective augmented reality experience.

Nathan Silberman [9] has shown that depth information can be used not only for background separation, but also effective segmentation of background objects by utilizing point cloud effectively. Michael Van den Bergh [10] has shown that background separation with depth data is not only useful in augmented reality, but at the same time this can be used for effective hand gesture recognition. Shahram Izadi [11] has proposed a method called KinetFusion which allows a 3D reconstruction through depth data.

Therefore, depth based processing allows the user to be separated from the background, use hand gesture and reconstruct the artificial scene combining segmented data with virtual objects. Objects can also be controlled by hand gestures in such an environment.

Intel RealSense™ is a fascinating new technology from Intel that leverages a special camera with depth and RGB sensors to achieve a very high frame rate, fast and efficient background segmentation technique [12]. RealSense™ also offers hand tracking, face tracking and voice recognition system. This opens up infinite possibilities in modern and next generation UI design. Features of the SDK provide an opportunity to combine augmented reality with a modern multimodal UI system.

Andrew D. has demonstrated how computer vision can be effectively used to build interactive gesture based UI [13, 14]. However, his techniques use RGB camera which is not efficient in building a robust gesture based system. If the depth map based gesture control [10] is combined with the UI control mechanism proposed by [14], it can offer a much more natural and intuitive user experience. Sharon Oviatt [15] has proposed a multimodal UI design principle by combining pen and speech recognition. Margrit Betke [16] has proposed a face tracking based HCI system. The author has shown how the system can be helpful for users with severe disability. Cristina Manresa-Yee [17] has suggested methods for enhancing user experience in a vision based control system and demonstrated the claim with a face tracking based UI control system. Christine L. Lisetti developed MAUI [18], a multimodal user interface that combines speech and facial expressions.

Therefore, it is clear that we can build interactive systems with face tracking, gesture tracking, touch and voice recognition. But it is always a challenge to resolve the input modalities in a complex UI workflow. We need a standardized mechanism for different modalities to work seamlessly. Douglas B. Moran [19] has proposed an Agent based system for combining different modalities into a seamless user experience. The agent based technique allows input modalities from being separated from UI elements. The work describes how suitable triggers can be used by the modalities to raise events to control UI.

Therefore, it becomes clear that by adopting an Intel RealSense™ SDK, we can effectively combine background segmentation, gesture tracking, face tracking, emotion detection, speech recognition in a natural UI system for Augmented reality.

Adobe Photoshop has redefined the image processing through its photo retouching ability [20]. There are various apps available in different app stores which provide the user to apply different filters to images to modify or retouch the images to make them look better. However, most of the commercially available retouching software is limited to still images. We realized that there is a vacant area of research between image processing and augmented reality. We found no suitable literature and past works that have effectively used augmented user as virtual objects and allowed image filters on augmented users. Ammar Anuar [21] has suggested a very simple method of using OpenCV for applying some simple filters on live video frames. This has motivated us to develop next generation Augmented Reality system called Augmented World which is discussed in detail in following section.

## 2. Proposed Work

### 2.1. Problem Analysis

What if there was a system that allowed the users to be augmented in a synthetic scene through background separation and then allow the user to use photo retouching techniques on the segmented foreground object or the background scene? Then we can literally create an augmented reality experience where the user himself is the “virtual object”.

Stylizing Augmented reality [22] is a fascinating new domain in Augmented reality. In this technique virtual objects are retouched with different image filters. Results of [23] show that retouching or changing the objects with suitable filters can enhance the user experience to a great deal. Jan Fischer [23] shows that using virtual objects creates aliasing effect in the rendered scene which can be reduced significantly through stylizing. Chien-Hsu Chen [24] has provided some basic filters that can be applied in an AR context. Thus, it is quite clear that real time image processing [21] can be applied over augmented scene [24] to get more realistic results. Kevin Dale et al.[25] has shown a way of replacing faces in real time video. This is a very interesting direction which suggests that if such “actor” based methods can be adopted in AR then we can have a more realistic AR scenario where even the user can be “Augmented” and stylized. We didn’t find any such system in past literature.

Even though Augmented reality leverages certain features of background separation and gesture tracking, they are not used to develop a multimodal system. Therefore Augmented reality, multimodal UI, object stylization, photo retouching and background segmentation remain an isolated area of research. The proposed work uses the Intel RealSense™ SDK to offer an innovative solution to bridge these research areas into a single solution with the future direction of research in this area. Our contribution is presented in detail in contribution subsection.

### 2.2. Contribution

In this work we propose a novel Augmented Reality system where users can be augmented in a scene as Virtual object and common photo retouching filters like sharpening, smoothing, pencil sketch, gray conversion, sepia can be applied to the augmented user object. Our method uses APIs provided by Intel RealSense™ SDK [26] for live background segmentation. The method uses AForge. Net [27] and OpenCV [28] libraries to apply image processing techniques on live segmented user augmented over an artificial image. The work provides options through which user may choose to apply image transforms on either segmented frame or the live frame or on a static image. Transforms are provided with slider based parameter controls that allows user to change the effect of transformations. Our work also provides a multimodal next generation UI that can be used with touch, hand gesture and multi touch. The system allows users to take still images of the background or foreground transformed scene and also video recording of the scene. Video recording extends the applicability of the work from just an image processing application to puppetry and movie making tool which are discussed in detail in the result section.

The work contributes significantly towards developing the next generation UI that can be effectively controlled by hand movement through an innovative hand movement to the UI translation system called “Split Grid”. The design of each component of the system, their challenges and the detailed technique are discussed in the Methodology section.

## 3. Methodology

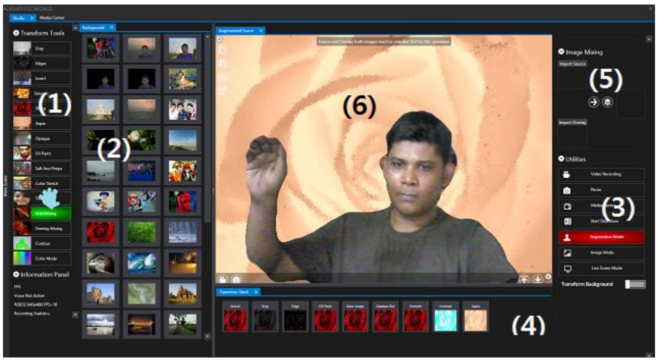
Augmented world supports several utilities to create and generate amazing photo effects as well as videos with different image transform effects by incorporating various algorithms at different levels. We divide the work into following sections:

- User Interface Design
- Producing Photo Effects
- Scene Management

### 3.1. User Interface Design

*UI Specification*

The UI uses resizable three column layout common to most professional software and IDE which is shown in Figure 1.



(1) Transform Tools (2) Gallery (3) Utilities (4) Operation Stack (5) Image Mixing Panel (6) Rendering Panel

**Figure 1. UI Of the Proposed Work**

Now a day’s computer hardware comes with different display resolutions. Therefore the system must be able to adopt to display resolution.  Rendering Panel is one where scenes are rendered. This is adjusted with left, right and bottom grid splitter.

Transform tools contains image transform tools that can be applied to segmented image, live stream or just the background image. Transform tool also contains options for applying transforms locally in an image.

The gallery contains the captured still images and retouched images.

Operation stack is a list box that stores the thumbnail of all the images which are produced through various transforms that are applied to the image. A transform is applied to the result of the previous transform if the background is already retouched.  User can select any intermediate result by selecting the thumbnail. When transform is applied to live stream or segmented user stream, only current transform is applied on the scene.

Utilities contains menu option for selecting Modes: Live, Segmented, Image. The modes are described in detail in Proto Effect Section. The image mixing panel contains options that help mixing two different images.

*UI Control*

Figure 2 shows the process of controlling UI elements by using hand movement.

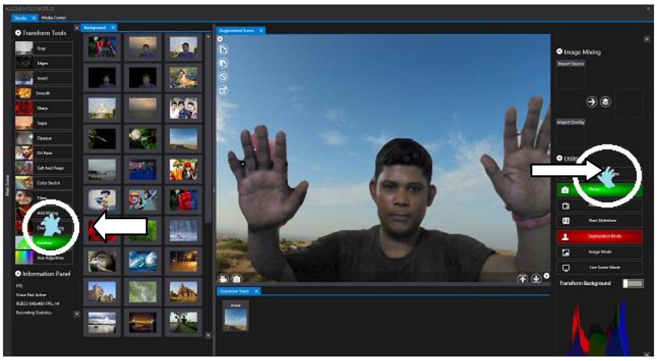


**Figure 2. Controlling UI with On-Air Hand movement**

User can move his both hands in front of the camera to move to hand thumbnails, one on each side of the screen. RealSense™ SDK has a hand tracking mechanism. The SDK provides 3D coordinate data of both hands along with hand openness percentages. Depth stream has a resolution of 640x480 which is the basis for calculating hand coordinates. Thus the coordinate data is bound in a rectangle of 640x480. The challenge is to map this data over the entire screen.

Cristina Manresa [29] has proposed RGB based gesture detection. Primary gesture like hand close, move from left to right can be used as events and trigger UI action. Manresa [30] has also extended the work as Finger Mouse and shown how finger movement can be converted into mouse movement.

As Intel RealSense™ technology uses depth camera based hand tracking which enhances the tracking over the RGB image as has been already discussed in the Introduction section, analysis of some gesture driven systems are important. Zhou Ren [31] has used Microsoft Kinect to detect ten primary gestures.

Past works on Hand gesture based UI control [31, 32] rely on mapping the 2D hand coordinates to mouse position and synthesizing the "click" operation through certain gestures like a fist. The major problem with such a system is that users can use only one hand to control the cursor. When 640x480 coordinate of hand position is mapped over high resolution screen like 1900x1600, the movements become extremely jerky. In a single hand controlling mechanism, using a hand on the other side of UI is extremely difficult. For example, it is very difficult to position a cursor over the Menu panel through our right hand, but it is quite easy to do so for Assets panel. Similarly, it is very difficult to access the elements in the Assets panel through the left hand, but it is easy to use left hand to move a cursor over menu panel or preview panel. Hence it is a natural instinct of the user to try to use respective hands. We therefore introduce a novel mechanism called “Split Grid” for allowing efficient and easy UI control. As we allow the user to use both his hands for controlling UI and animation elements, we use two “Hand Markers” as a visual clue for the user to know where his hands are pointing at the screen.

### *Hand Markers*

At the beginning, we segment and resize hand part from 3D data and store corresponding binary thumbnail images from both the hands at different openness percentage.

We keep the images into two thumbnail arrays called leftMarker and rightMarker. Six images par hand taken at openness separated by 20% i.e. ( 0,20%, 40%, 60%, 80% and 100%) are stored in an array. For every frame from the camera we process, we first find number of hands being detected and then loop through every hand. For each hand, we check the distance (normalized z) of hand from the camera. If the distance is less than 0.45 meters, then we render the marker depending upon the openness of the specific hand. Two hand markers can be seen on either side of the UI in figure 2 shown by arrow.

*Split Grid*

The fundamental objective this method is to conceive the space between the user and the camera into a virtual cube of equally space grid and then map the hand position in different grids to 2D screen with different resolution. The Split Grid for the proposed work is shown in Figure 3.

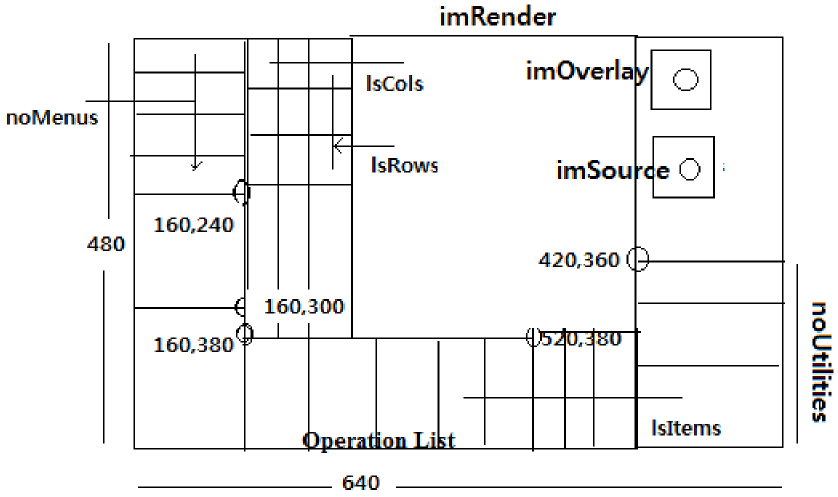


**Figure 3. 2D Split Grid for the Proposed Work**

The Grid is 2D. We consider normalized Z=0. 45 as the distance from the camera at which the Grid will be active. *noMenus* defines number of menu items. Each menu button is of height *h*. While selecting the menus, left hand needs no x-axis movement. Thus, if the hand position is less than (160, 240), left hand marker is set at position in the screen over the menu object given by

*menuNumber=int(240/noMenus)*

If the left hand position in x axis is less than 160 and the y axis is between 240 to 300, the left hand marker is set over the Preview panel.

Right to menu panel is another panel that holds the gallery list box. The width of this panel is adjustable. Therefore the number of columns of the list box differ. User needs to select a particular list item. Thus, rather than resolving the movement into positional data, we find out *row=int(480/lsRows)* and *col=int(x-160)/lsCols*. Now if the left hand x-position is between 160 and 240, the absolute position from 80 pixels resolves to column number and the y position is resolved into row number. The left hand marker is moved to specific list element image at calculated {row, col}.

The central rectangle specified by diagonal coordinates ((240, 0), (420, 380)) is the location of the scene where the augmented scene is rendered. Users can use both hands to manipulate menu items. Therefore, we permit both left hand as well as the right hand locations to position the respective markers.

Marker positioning is as given below.

Lx= WIDTH*$(lx\text{-}240)/(520\text{-}240)$

Ly=HEIGHT*(ly)/(380)

Rx=WIDTH*(rx)/520

Ry=HEIGHT*ry/380

Where WIDTH, HEIGHT is actual screen width and height of the augmented reality image object, (Rx, Ry) and (Lx, Ly) are right and left hand marker positions on the screen and (rx, ry),(lx, ly) are the left and right hand position returned by the SDK.

Bottom operation list box has a number of image thumbnails which are generated through consecutive operations. Number of items in the list box increase with the number of operations. As the list box is located almost near the center of the grid, the user may use either hand. Note as the list box has a single row, user need no y movement in this area. Thus if either of the hand is below 380, then their respective x axis position is used to calculate the item index on which the respective hand marker must be set.

Therefore the Split Grid concept can be used for any GUI depending upon the UI layout and different areas of the UI can be accessed and used independently depending upon their property.

*Click Event*

User can move the hand markers by keeping the hand open. He can select any list item or a menu by synthesizing the click even through hand closing. It is observed that when a user wants to close the hand, the position is invariably displaced. Therefore the proposed work translates hand position to hand marker position when the hand is opened more than 80% and synthesizes the close event when the openness falls below 20%. Therefore the displacement at the time of closing the fist is translated into the marker position which allows the user to select the item in a more controlled way.

It is very difficult for the user using hand movement to control the UI to control the marker without any visual clue. Due to the fact that the marker position is obtained through integer translation of the actual position over only 640x480 plane. Thus the motion is less noisy. Hence we offer visual clue to the user. When the hand is over a menu, the menu background changes, when the hand marker is over the preview panel. When a user opts for the click event, selected menu's background is changed.

Figure 4 demonstrates the hand postures for moving over the menu and selecting the item.

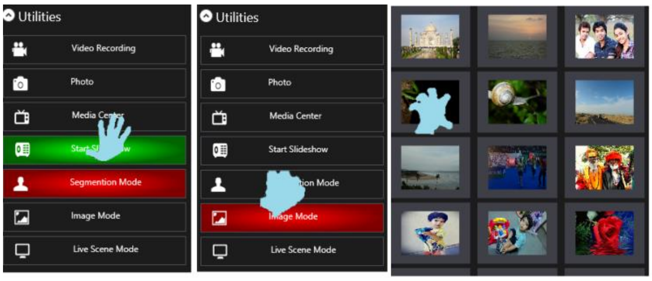


**Figure 4. Navigation and Selection of UI Elements**

```
Struct {
x,y,z:float
openness:int
hand:LEFT or RIGHT
} Hand[2] //x,y,z and openness is obtained through
sdk
            //for each frame
handMarker[2][6] //array of hand marker thumbnails
```

```
LEFT=0
RIGHT=1
Foreach(NEW_FRAME)
 N=NUM_HANDS;//obtained through SDK
For(i=0:N)
  If(Hand[i].z]>.45)
      Display(
handMarker[Hand[i].hand][Hand[i].openness/100]);
UIAction(Hand[i].hand,Hand[i].x,Hand[i].y);
end
end
//UIAction() calculates respective hand marker
position and event
```

**Table 1. Algorithm of Hand Tracking System**

Complete hand tracking and mapping 3D hand coordinates to hand marker is presented in algorithm is presented in Table 1.

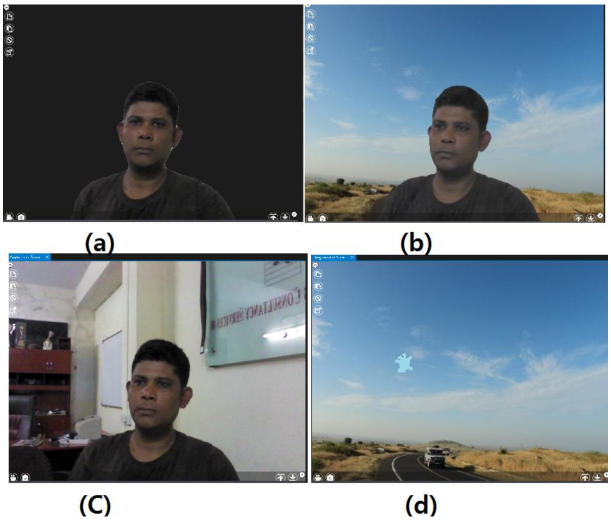
### 3.2. Rendering Modes

The work supports three rendering modes: Live, Segmented and Image. In Live mode the RGB stream of the camera is rendered.

In the Segmented mode, background segmented user stream is rendered against a static image background.

In Image mode, only the background image is displayed. Therefore, users can perform pure image processing operations in the image mode.

In all the modes hand markers are active which allows users to control all UI elements in all the modes. Image mode is helpful when the system is used for photo retouching or image mixing. Segmentation mode is used for augmenting user against a background. This can be used for capturing user’s photo against a desirable background or for video recording. The user may also use different transforms into both foreground and background to change the visual of the scene. These transforms are discussed in detail in the next subsection. Rendering under different user modes as shown in Figure 5.



a) Segmented user stream with no background b) Segmented stream with image background c) Live Scene d) Only Background Image

**Figure 5: Rendering Modes**

In the next subsection we discuss about various transforms and effects that can be applied on these different rendered scenes. These modes can be selected by selecting respective menus from the utility panel as can be seen in Figure 4.

The rendering process is depicted by Figure 6.

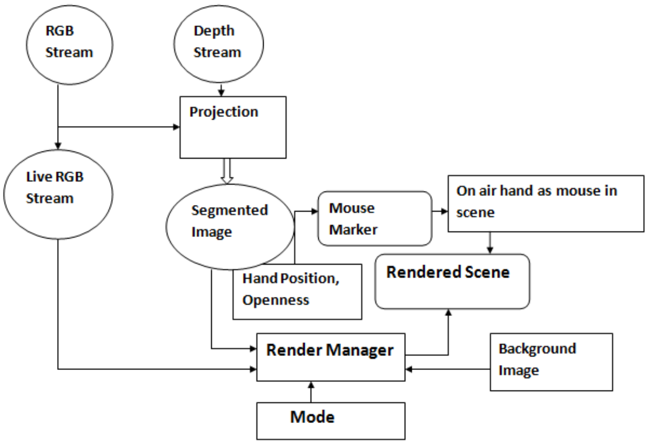


**Figure 6. Block Diagram of Rendering Process**

It is clear that Live scene frames are obtained from the RGB sensor and segmented frames are obtained by combining RGB stream with depth strum through projection APIs provided by the system. Rendering manager combines this scene with a background image (if provided) in the segmentation mode. Mouse marker images are rendered over the augmented streams by positioning the hand marker into a position returned by hand tracking system of the SDK.

### 3.3. Image Transform

Augmented World is a bridge between Stylized Augmented Reality and Image processing. Rather than rendering artificial objects on the screen and controlling them through gesture, we propose technique to apply stylizing directly in the segmented user stream. The methods are extended to both background as well as the live stream so as to extend the application of the proposed work to image retouching and video recording with live effects.

Following transforms are supported by the work.

- Gray conversion
- Edge Detection
- Color Invert
- Sepia
- Image Sharpening
- Image Blurring
- Smoothing
- Oil Paint effect
- Sketch Pen(Mean Shift Segmentation)
- Pixilated effect

- Opaque Glass Effect (Jitter transform)
- Hue filter
- Salt and Peeper Noise

We have used Aforge.Net library for implementing these options. Complete details of the methods, their explanation and API reference can be found in [27].

The objective of the work is to adopt the existing filter to create an immersive and intuitive user experience. Hence we shall focus on adaptation details of the filter by the system rather than elaborating the details of the transform tools as they are quite common and elementary image processing context.

Block diagram of applying transform tool over different streams as well as on the background is shown in Figure 7.

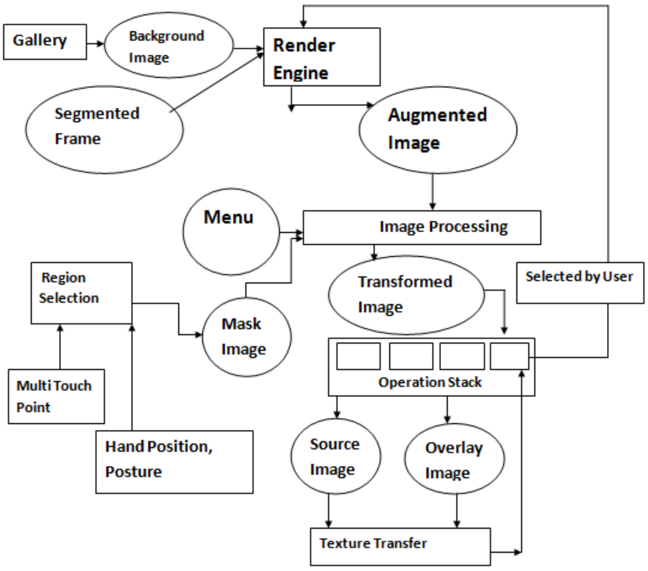


**Figure 7. Image Transform Block Diagram**

There are two ways on which transforms are applied: Foreground and background that can be selected from option switch in selection panel.

In live mode, transforms are always applied on the live scene. In Segmentation mode, transform can be applied to background image or to segmented stream by selecting foreground transform.

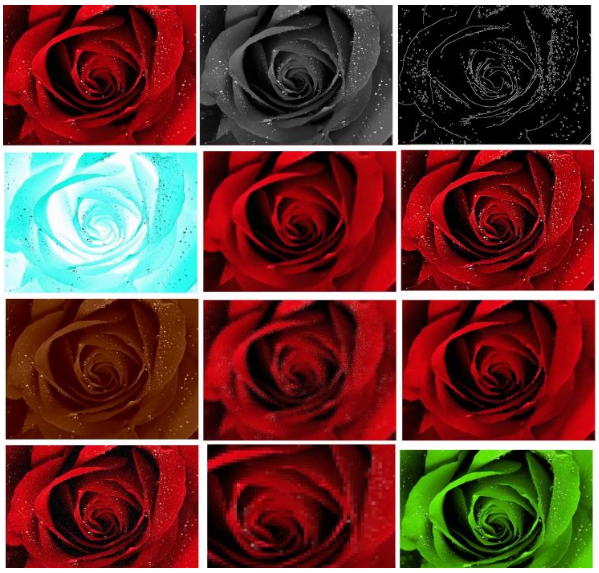
Most of the Stylized Augmented reality works focuses on artificial object being stylized. We realized that by extending the option to even background the application of the work could be extended to image processing app. Therefore this two individual ways of applying transform is provided.

When the transform is applied to background the result is saved temporarily and it's thumbnail is put in Operation Stack list box. User can at any instance select a previous image from this list. When a new image is selected from the gallery, this list is deleted and a new list is created with just selected image as the first image in the stack.

*Global Transform*

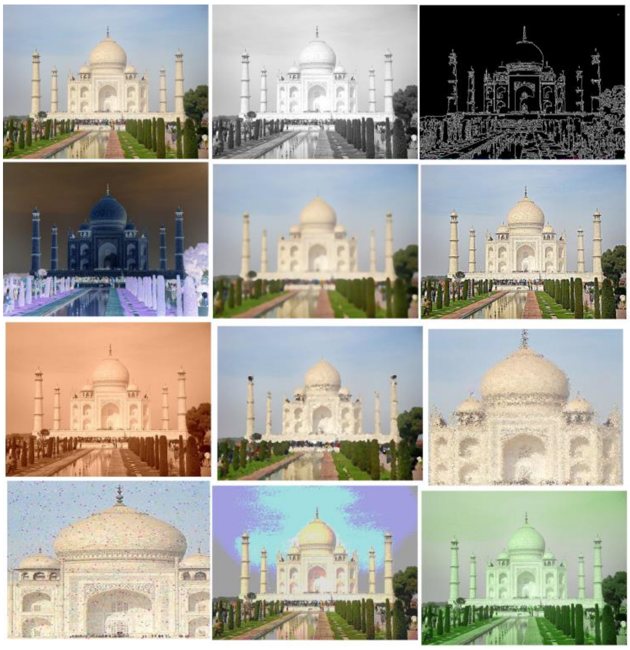
In background processing two modes of operations are supported: Global and Local. *i.e.*, The user can apply a transform globally over entire background image or part of the image which is defined by a mask. User can select a homogeneous area on the image as mask just by double tapping on an area of an image or by hand close gesture on the area of the image. The transform is applied only over the region of the image defined by the mask.

We also present a method of image mixing. Two images, namely source and overlay can be selected by the user and the images can be mixed by either texture transfer of overlay to the source or color channel addition of the images. Figure 8 depicts the results of various transforms.



| | | |
|---|---|---|
| Original | Gray | Edge |
| Invert | Smooth | Sharp |
| Sepia | Opaque Glass | Oil Paint |
| Salt and Pepper | Pixilated | Hue Filter |

**Figure 8. Image Transform on Rose Image**



| | | |
|---|---|---|
| Original | Gray | Edge |
| Invert | Smooth | Sharp |
| Sepia | Opaque Glass | Oil Paint |
| Salt and Pepper | Sketch Pen | Hue Filter |

**Figure 9. Transform Applied to Tajmahal Image**

Figure 8 and Figure 9 gives us a very good idea about the behavior of the filter. Having understood how the filters work, let us see what effects these transforms bring when they are applied on the foreground and background.

Figure 10 helps us understand how the transforms are applied to foreground in Segmentation and Live mode and on background in Segmentation mode.

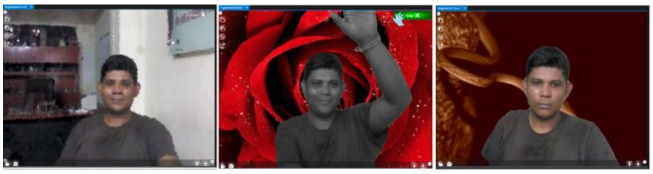


**Figure 10 (a, b, c): Effect of Transform: Oil paint applied to live stream b) gray scale applied to segmented stream c) sepia applied to background.**

It can be seen from Figure 10 that by stylizing the foreground and background with different filters, wonderful effects can be generated which can be used to record effective short movies for spreading social messages or simple advertisement campaign so on.

We have already discussed how the proposed work extends the image processing capabilities of the system by providing a method for applying the transforms locally. Following subsection describes the concept of local transform of the image.

*Local Transform*

One of the most important aspect of the system is its ability to support local operation. The filter can be applied either on entire image or over certain areas of the image. A subset region over which a transform or filter can be applied is called a mask area.

A mask can be selected by quick region selection tool. Quick region selection is one of the most innovative technique for quickly applying transform over an area. User can quickly select an area of the image by double tapping on that area or by synthesizing click even through hand close on a particular region.

Double tapping click event will return the pixel {x, y} where the event was triggered. Fast flood fill algorithm [33] finds out all the pixels in it's neighborhood whose pixel values are similar and generates a mask containing all these pixels set to 1 and every other pixel on the image set to zero.

Mask image is a binary image with 1 in the pixels in the selected area and 0 otherwise. The Algorithm is presented in Table 2.

*Let M be the mask, IR is the image produced from I using any generic transform T such that :*

*IR= T(I)*
*Then*
*IRM= I* !M +IR *M*

*IRM is the result of image processing operation T applied locally to image I.*

**Table 2. Algorithm of Local Transform**

Just like the global application of filter, locally applied filters also can be stacked and the operation results can be cascaded.

Once a user chooses to delete the mask whose thumbnail is displayed at the time of the local operation, operation mode becomes global.

Figure 11 demonstrates the process of selecting mask area over an image and then apply the transform to the region bounded by the mask.

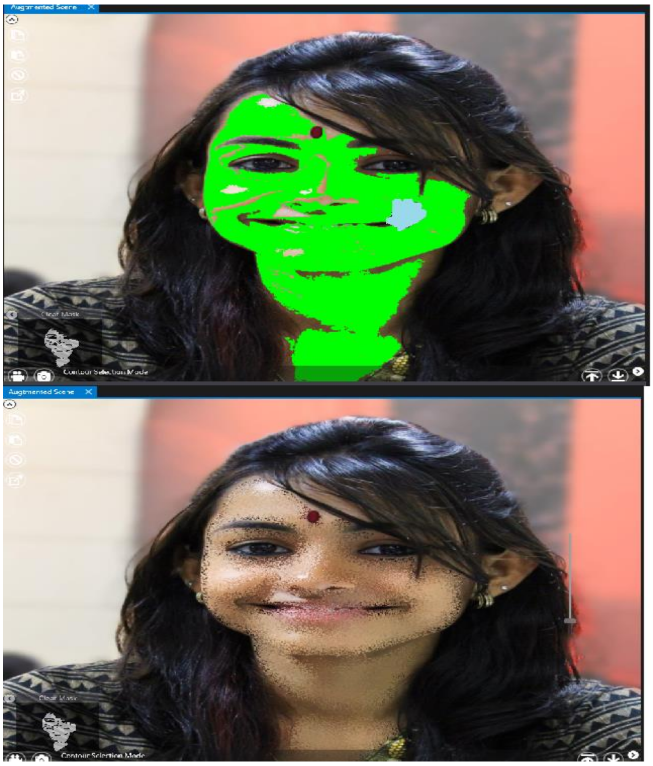


**Figure 11. Locally Applying Transforms**

Top: Selecting mask with hand, Bottom: opaque glass effect applied locally.

*Image Mixing*

One of the most interesting features of the proposed system is its ease to produce images by using texture transfer. It is observed that by applying the texture of grass, rose, water etc to any image enhances an image's aesthetic beauty to multifold. Texture transfer is performed by simple addition and XOR operation on two images namely source and overlay. An image mixing algorithm is presented in Table 3.

```
IA=I1+I2%255
IO=I1 XOR I2
IA and IO are  added and overlaid images generated from two image I1 and I2.
```

**Table 3. Image Mixing Algorithm**

Proposed system supports all transforms to be used in conjunction. *i.e.*, Local filter can be applied to IA and then that can be used for texture transfer image I1.

Texture transfer can be applied to the result of a transform or the result of texture transfer can be transformed. Figure 12 shows image mixing process.

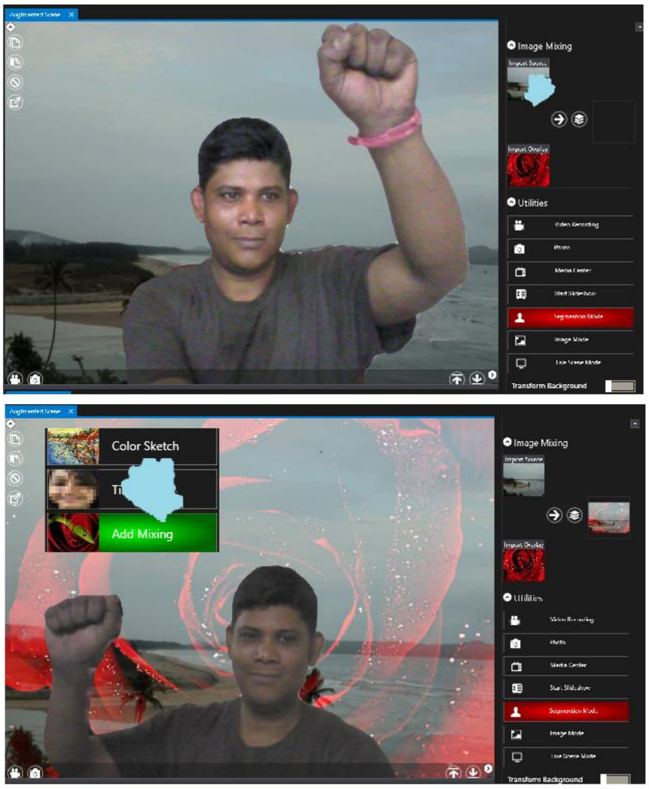


**Figure 12. Image Mixing (Top: Selecting Source Image Bottom: Selecting Mixing Operation from Menu)**

Mixing operation is allowed only over a pair of background images. Background transforms are allowed in both Segmentation as well as in Image mode. This extends the capabilities of the system and enables the users to use the transforms even while recording a movie.

## 4. Results and Discussion

We wanted to test the Augmented Reality as well as photo editing features. We divide the evaluation in two distinct tests: Augmented Reality use case, Photo Editing use case. Following are the category of the users volunteered for the test.

*Participants*

Once the system was developed we wanted to conduct a usability test across different user groups. We also study the usage by users with formal training in photo editing and users without any such formal training.

Ten second year students from Government Fine Art college were invited under “Artist” category of users. These artists were more trained in physical brush photos and had not had any formal training on Photoshop.

Ten final year Computer Science diploma students from HKE Polytechnic college were invited to test the system under “Armature” category.

We also invited five professional Photoshop artists from Ants school of animation. Their test were studied under “Photoshop Artist” category.

Ten school students from Kendriya Vidyalaya studying in class IX volunteered to test the system under “Kids” category.

Artists, Armatures, Photoshop Artists and Kids took independent tests. Their usability study is presented below.

Details of the apparatus and the nature of test is described in detail in subsequent subsections.

*Apparatus*

One, Lenovo All in One devices with Intel 4th generation i5Core with 8GB RAM running Windows 8.1 operating system was used for the tests. Intel RealSense™ SDK was installed in the devices. RealSense™ camera was provided with test machine. We had setup the system in  HKE college computer science laboratory, Government Fine Arts College and Kendriya Vidyalaya for two days each with appropriate permission from the authorities to allow the student's to take the test on weekends. Photoshop Artists took the test in their own system.

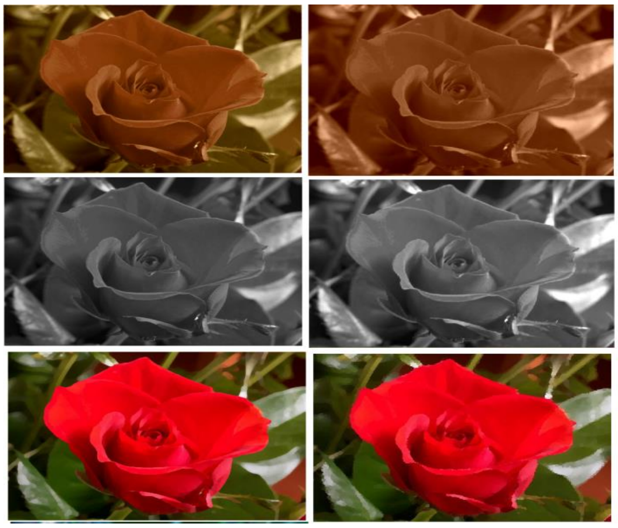
*Procedure*

We asked all the groups other than Photoshop artists to create image effects using proposed system with a rose image. We also asked Photoshop artists to create different effects with Photoshop tool. Participants in proposed system were given a formal training jointly for two hours in Government Arts college. Then each participants were allowed to have a hands on with the system for one hours. Our aim was to evaluate how many effects can participants generate in a stipulated time. As the proposed work is meant for beginner level photo effects, we had not invited any professionals to evaluate the photo quality.

Further we asked the participants to use live capture feature and come up with different photo effects. Professional Photoshop artists were allowed to use any photo capturing app they wished to capture live photos.

Figure 13 shows the comparative images generated by professional artists in Photoshop and generated through our work.

It can be seen from the figure that effects generated by the proposed work were at par with the similar effects being generated through Photoshop. Photoshop has several times more option than the proposed work, but even the kids found the proposed work easy to work with. In Photoshop effects, central exposure to objects was more where as in the proposed system the exposure is distributed over the entire image.



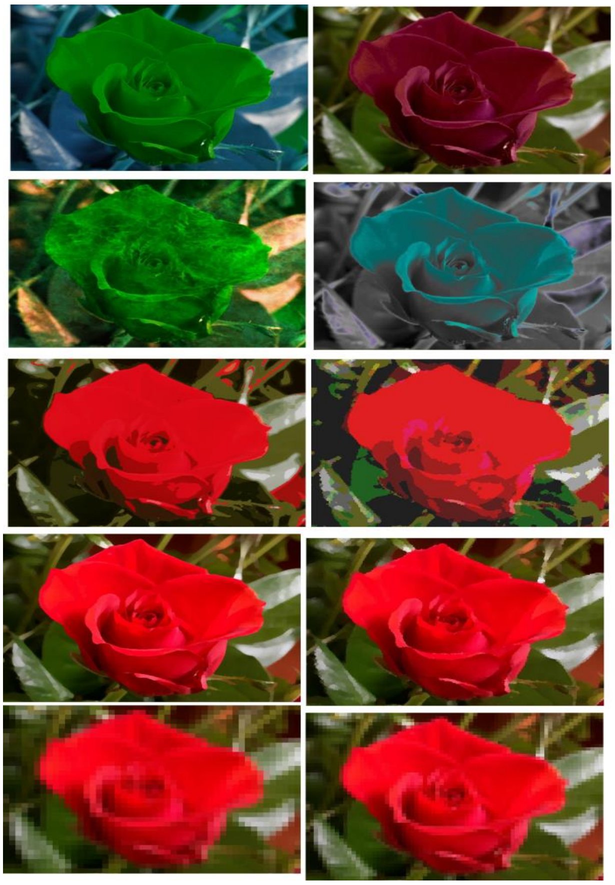


**Figure 13. Comparative results from Photoshop (left)& proposed work(Right) for Sepia, Gray Scale, Sharpening, Hue adjustment, Texture Transfer, Sketch Pen, Oil Paint, Pixilated effect ( From top to bottom order)**

We had allocated 30 minutes of time to each participants to generate as many photo effects as they could from proposed work. Photoshop artists were also allowed to use Photoshop for same interval to generate as many effects as they could.

### 4.1. Usability Tests

*Test Results*

Table 4 presents the comparative aggregated results of for Photoshop and proposed system for 30 minutes session.

| *Category* | *Total Filters* | *Images Produced* |
|---|---|---|
| *Amateur* | *35* | *23* |
| *Kids* | *39* | *31* |
| *Artists* | *23* | *13* |
| *Photoshop Artists* | *37(Photoshop operations)* | *7* |

**Table 4. Image Processing Results with Rose Image**

We observed that Photoshop needed more time in producing images in comparison to proposed work. One of the reasons for this is that the proposed work has a much simpler workflow than Photoshop. Images produced by Photoshop were much more professional than the proposed work. However the fact that even kids could love and produce so many images gives proposed work a distinct edge over Photoshop. It also proved the usability.

Artists were found to be more thoughtful while creating the photos. They were found to use more filters and blending for generating each photo.

In our next test we asked participants to work with their photo and decorate. It was observed that Photoshop took immense effort for carefully removing the background using lasso tool, whereas users of the proposed system could generate many more images with cool effects. The result is shown in Table 5.

| *Category* | *Photos Used* | *Retouched Images* |
|---|---|---|
| *Amateur* | *22* | *8* |
| *Kids* | *17* | *12* |
| *Artists* | *9* | *4* |
| *Photoshop Artists* | *2* | *3* |

**Table 5. User's Photo Retouching with Different Backgrounds**

The clear advantage of the proposed system in real time scenario over Photoshop can be seen in Table 7. It was observed that due to automatic segmentation users for proposed system could use many different photos and frames. Offline production of segmented photos needed more efforts even by professional Artists.

Figure 14, Figure 15 Shows the performance of the system for more than one user.

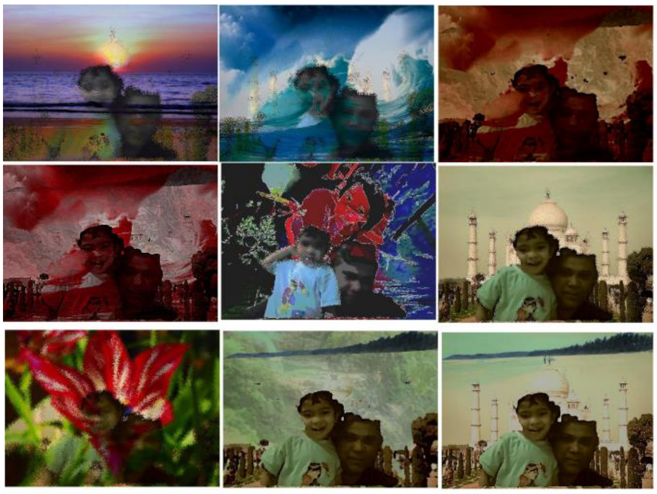


**Figure 14. Image Effect on Two Users of Different Age Group**

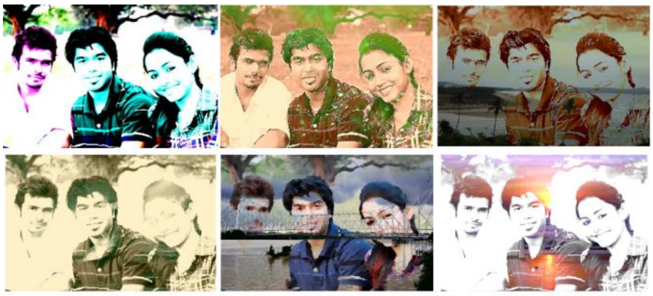


**Figure 15. Image Effects on Three Users**

It can be clearly seen in Figure 15 that the segmentation of the system also performs well for multiple users. Segmented images can be further retouched with different blending and cascaded filters. However Figure 14 shows that some part of the upper head was not segmented properly. A Kid and an adult in a frame created a different depth profile which resulted in bad segmentation.

### 4.2. Performance Analysis

We divided each testing category group into group of two. One group was provided with multi-modal version of the app and the other with plain mouse driven UI. Table 1 presents the access log data.

*User Engagement*

**Table 6. Engagement Data of Different User Group**

| *User* | *Total time before closing App(min)* | *Average Number of Operation Performed* | *% of Voice Command Used* | *Number of New image par user Saved in Gallery* | *% of Touch* | *% Hand Gesture* |
|---|---|---|---|---|---|---|
| *Pro-Multimodal* | *12.2* | *17* | *51* | *3* | *29* | *12* |
| *Pro-Conventional* | *3.43* | *11* | *-* | *1* | *-* | *-* |
| *Artist-Multimodal* | *21.6* | *39* | *59* | *9* | *41* | *44* |
| *Artist Conventional* | *15.2* | *22* | *-* | *3* | *-* | *-* |
| *Armature-Multimodal* | *23* | *19* | *64* | *12* | *52* | *43* |
| *Amature Conventional* | *14* | *14* | *-* | *4* | *-* | *-* |

Table 6 revealed some surprising facts that professional artists who work with photoshop still prefers mouse and keyboard. They used touch or hand gesture very less even when the option was available.

We observed that artists and armatures were far more attracted to produce new images and changing the texture of the images. They enjoyed saving images and viewing it again more than Photoshop user.

In all the cases multimodal design was found to be more engaging. Users, who were given the multimodal version of the software with hand, touch, mouse, voice input option were found to be working with the software for far longer period of time in comparison to users who used mouse driven interface.

*System Resource Utilization*

Different user group’s data was aggregated for Resource Utilization test. Test included logging CPU utilization, memory utilization and Frame rate for different criteria which is presented in Table 7.

**Table 7. Resource Utilization by Various Modules**

| *Operation* | *CPU %* | *RAM%* | *Frame Rate* | *Battery Consumption % par Hour* |
|---|---|---|---|---|
| *RGB Stream* | *8* | *6* | *34* | *30* |
| *Background Segmented Stream over Synthetic Background* | *61* | *72* | *17* | *93* |
| *Mouse Marker* | *63* | *66* | *24* | *83* |
| *Voice Recognition with RGB Stream* | *14* | *9* | *30* | *37* |
| *Voice Recognition, Hand Gesture, Multi touch with Mouse Marker* | *66* | *68* | *20* | *85* |

Table 7 provides a very important input about the performance of the system. It is clear from the table that Intel Perceptual techniques are not tuned to minimize resource utilization. Background segmentation process was observed to be very resource intensive. Another fact that was observed was that prolonged rendering of background segmented data generated immense heat which also lead to quick battery drainage.

*Effect of Image Size on the Performance*

Large images requires more time for operations to be completed. Bigger the size, longer it takes for any masked operation to be completed. In an environment which also supports video recording and rendering of frames in the screen, a reasonable frame rate and audio visual synchronization was one of the most essential part of the design.

It was important to design techniques that could support parallelism and that could be completed in less amount of time.

In order to meet the time and speed constraint as mentioned above, we developed our own fast and parallel implementation of every technique. In this approach we replaced every inner *for* loop in generic two *for* loop processing blocks with *Parallel.For*. Thus methods attained highest degree of parallelism.

**Table 8. Effect of Image Size on the Performance**

| *Image Size* | *Avg time /Instruction in ms for Global Opn* | *Avg time /Instruction in ms for Masked Opn* | *Frame Rate ( With Mouse Marker)* | *CPU Utilization* | *Memory Utilization* |
|---|---|---|---|---|---|
| *640x480* | *32* | *51* | *24* | *11.3* | *4* |
| *720x560* | *39* | *87* | *23* | *12* | *4.2* |
| *960x800* | *48* | *124* | *23* | *12.7* | *4.8* |
| *1024x944* | *61* | *210* | *22* | *13.22* | *5.1* |
| *1536x1326* | *112* | *389* | *20* | *14.13* | *9.0* |
| *4000x3000* | *500* | *1017* | *20* | *18.92* | *12.4* |

It can be seen from the table that actual time taken by image processing algorithms to alter images of digital cameras takes much higher time than smaller images. But frame rate provides interesting insight about the performance of the proposed system. We can see that the frame rate does not change significantly depending upon image size. This is because all the techniques are applied in the background and only after image is ready, it is rendered in the front panel. Also CPU utilization is almost consistent due to use of parallel for loop. Parallel processing in .Net does not block CPU resources. Hence performance of algorithms is supreme if they are implemented with parallel programming.

It was also observed that size of image or number of image in operation stack did not affect the performance by great extend. Operation Stack is a panel where only thumbnail of a result is displayed. Original result is saved as metadata with parameter info. The System stores only this metadata. Whenever any image from the stack is selected, the operation represented by the thumbnail is performed and image is reproduced. If there are many operation before the current one then the operations are performed one after the other on the result of previous operation.

## 5. Conclusion

Image processing and Image editing tools and software are popular applications in today's consumer centric App environment. Even though several photo effect apps are present in different app stores, new applications are often developed to overcome shortcomings of the existing software. Professional software like Photoshop and paint remains popular photo editing options.

However, with the introduction of Intel Ultrabook Platform, touch PCs have become new and fashionable “must have” devices. Such multi touch Ultrabooks have different form factor and can be veru intuitive if the applications are written that suits the platform.

Intel's Perceptual Computing SDK has revolutionized human computer interaction. Though Microsoft's Kinect is a popular platform for gaming based on gesture interaction, close range gesture supported systems were required for the technology to be used in personal computing.

One of the biggest challenge for application developers is the difficulty to incorporate various modalities in a natural way. Developing software that supports touch, sensors, hand and head gestures, voice recognition provides a great platform but it also throws immense challenges before the developer.

Augmented World is developed as a proof of concept and case study for multi modal image application. We tailor-made several techniques and gestures for the user to be leaving the system. We carried out extensive study among different user groups to come to the conclusion that the multimodal platform with modern metro style design that supports gesture and voice has a great potential to engage new users and to get then glued to the system.

The proposed method is not only path breaking in its design and integration of multi modality into a complex processing system, it also presents a workflow for great image editing experience. Stack based processing, touch and gesture based quick region selection, combining local and global image operations, augmentation of background removed image with synthetic background image, ability to record screen operations with audio, face close up are some of the novel and unique operations for an Image editing platform.

Various, mixed and texture transferred images produced by proposed system can compete with most popular photo editing apps.

One of the drawback observed during the evaluation of the system is SDK's high resource consumption for few operations. But with the newer and better depth camera and improvement in firmware and software, the proposed system can be made more realistic and non resource intensive, which would make it possible for the application to run on Ultrabooks in battery mode.

## References

[1] E. Kruijff, J. E. Swan II and S. Feiner, "Perceptual issues in augmented reality revisited", *ISMAR*, vol. 9, **(2010)**.

[2] G. Gordon, "The use of dense stereo range data in augmented reality", *Proceedings of the 1st International Symposium on Mixed and Augmented Reality,* IEEE Computer Society, **(2002)**.

[3] J. Ventura and T. Höllerer, "Depth compositing for augmented reality", *SIGGRAPH posters*, **(2008)**.

[4] L. Wang, "Tofcut: Towards robust real-time foreground extraction using a time-of-flight camera", *Proc. of 3DPVT*, **(2010)**.

[5] R. Crabb, "Real-time foreground segmentation via range and color imaging", *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08, IEEE Computer Society Conference on,* IEEE, **(2008)**.

[6] E. J. Fernandez-Sanchez, J. Diaz and E. Ros, "Background subtraction based on color and depth using active sensors", *Sensors*, vol. 13, no. 7, **(2013)**, pp. 8895-8915.

[7] K. Kim, "Real-time foreground–background segmentation using codebook model", *Real-time imaging,* vol. 11, no. 3, **(2005)**, pp. 172-185.

[8] M. Harville, G. Gordon and J. Woodfill, "Foreground segmentation using adaptive mixture models in color and depth", *Detection and Recognition of Events in Video, 2001. Proceedings, IEEE Workshop on*. IEEE, **(2001)**.

[9] N. Silberman, "Indoor segmentation and support inference from RGBD images", *Computer Vision–ECCV 2012,* Springer Berlin Heidelberg, **(2012)**, pp. 746-760.

[10] M. Van den Bergh and L. Van Gool, "Combining RGB and ToF cameras for real-time 3D hand gesture interaction", *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*. IEEE, **(2011)**.

[11] S. Izadi, "KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera", *Proceedings of the 24th annual ACM symposium on User interface software and technology*, ACM, **(2011)**.

[12] Hariharan, Harini Priyadarshini. "INFRARED-AIDED SUPERPIXEL SEGMENTATION." (2015).

[13] Wilson, Andrew D. "Depth-sensing video cameras for 3d tangible tabletop interaction." *Horizontal Interactive Human-Computer Systems, 2007. TABLETOP'07. Second Annual IEEE International Workshop on*. IEEE, 2007.

[14] Wilson, Andrew D. "Robust computer vision-based detection of pinching for one and two-handed gesture input." *Proceedings of the 19th annual ACM symposium on User interface software and technology*. ACM, 2006.

[15] Oviatt, Sharon, et al. "Designing the user interface for multimodal speech and pen-based gesture applications: state-of-the-art systems and future research directions." *Human-computer interaction* 15.4 (2000): 263-322.

[16] Betke, Margrit, James Gips, and Peter Fleming. "The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities." *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* 10.1 (2002): 1-10.

[17] Manresa-Yee, Cristina, et al. "User experience to improve the usability of a vision-based interface." *Interacting with Computers* 22.6 (2010): 594-605.

[18] Lisetti, Christine L., and Fatma Nasoz. "MAUI: a multimodal affective user interface." *Proceedings of the tenth ACM international conference on Multimedia*. ACM, 2002.

[19] Moran, Douglas B., et al. "Multimodal user interfaces in the Open Agent Architecture." *Proceedings of the 2nd international conference on Intelligent user interfaces*. ACM, 1997.

[20] Margulis, Dan. *Photoshop LAB color: The canyon conundrum and other adventures in the most powerful colorspace*. Peachpit Press, 2005.

[21] Anuar, Ammar, et al. "OpenCV based real-time video processing using android smartphone." *Intl. Journal of Computer Tech. and Electronics Engineering (IJCTEE)* 1 (2011): 58-63.

[22] Fischer, Jan, Dirk Bartz, and Wolfgang Straber. "Stylized augmented reality for improved immersion." *Virtual Reality, 2005. Proceedings. VR 2005. IEEE*. IEEE, 2005.

[23] Bartz, Jan Fischer1 Douglas Cunningham Dirk, and Christian Wallraven Heinrich Biilthoff Wolfgang StraBer. "Measuring the discernability of virtual objects in conventional and stylized augmented reality." *12th Eurographics Symposium on Virtual Environments, Lisbon, Portugal, May 8th-10th, 2006*. Transaction Publishers, 2006.

[24] Chien Hsu Chen Real-time coherent stylization for augmented reality, TORONTO, HILTON. "ICME 2006." (2006).

[25] Dale, K., Sunkavalli, K., Johnson, M. K., Vlasic, D., Matusik, W., & Pfister, H. (2011). Video face replacement. *ACM Transactions on Graphics (TOG), 30*(6), 130. [26] Intel RealSense, https://software.intel.com/en-us/realsense/home

[26] Kirillov, Andrew. "AForge .NET framework." *2010-03-02)[2010-12-20]. http://www. aforgenet. com* (2013).

[27] Bradski, Gary. "The opencv library." *Doctor Dobbs Journal* 25.11 (2000): 120-126.

[28] Varona, Javier, Cristina Manresa-Yee, and Francisco J. Perales. "Hands-free vision-based interface for computer accessibility." *Journal of Network and Computer Applications* 31.4 (2008): 357-374.

[29] Manresa-Yee, Cristina, et al. "Experiences using a hands-free interface."*Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*. ACM, 2008.

[30] Ren, Zhou, Junsong Yuan, and Zhengyou Zhang. "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera." *Proceedings of the 19th ACM international conference on Multimedia*. ACM, 2011.

[31] Choondal, Jishmi Jos, and C. Sharavanabhavan. "Design and Implementation of a Natural User Interface Using Hand Gesture Recognition Method."*International Journal of Innovative Technology and Exploring Engineering* 2.4 (2013).

[32] Lee, Jayong, and Hoon Kang. "Flood fill mean shift: A robust segmentation algorithm." *International Journal of Control, Automation and Systems* 8.6 (2010): 1313-1319.

## Authors

**Rupam Das**, He received his BE degree in Electronics andCommunication from VTU, Belgaum in 2002 and M. Tech degree in Computer Science and Engineering from VTU, Belgaum in 2012. He is founder and CEO of Integrated Ideas, a R&D firm in Gulbarga, Karnataka. He is also heading the R&D department of the company. He has created over 30 applications in RealSense technology and is awarded Pioneer and Trailblazer in Intel Perceptual and Intel RealSense technologies respectively. He is currently pursuing his PhD in Computer Science from VTU, Belgaum. His research area includes Image Processing, Pattern Recognition, Internet of Things, RealSense Technology, Augmented reality and Gesture driven techniques.

**K. B. Shivakumar**, He received the BE degree in Electronics & Communication Engineering, ME degree in Electronics, MBA from Bangalore University, Bangalore and M Phil from Dravidian University, Kuppam. He obtained his Ph.D. in Information and Communication Technology from Fakir Mohan University, Balasore, Orissa. He has got 50 publications in International journals and conferences. He is currently working as Professor, Dept. of Telecommunication Engineering, Sri Siddhartha Institute of Technology, Tumkur. His research interests include Signal processing, Multi rate systems and filter banks and Steganography.