Image Processing Based Indoor Localization System for Assisting Visually Impaired People

Tasnia Ashrafi Heya[1], Sayed Erfan Arefin[1], Amitabha Chakrabarty[1], Md. Mazidul Alam[2]
Department of CSE, BRAC University[1]
Department of EEE, American International University Bangladesh[2],
{tasnia.heya, erfanjordison, mazidul.alam14}@gmail.com, amitabha@bracu.ac.bd

***Abstract*— Indoor localization or indoor positioning system is a known as a process of detecting position of any object or people inside a building or room by different sensory data collected from different devices using different techniques such as radio waves, magnetic fields, acoustic signals or other procedures. However, lacking of a standard localization system is still a very big concern. Solution of this issue can be very beneficial for people in many cases but it can be especially very beneficial for the visually impaired people. In this paper, an image processing based indoor localization system has been developed using OpenCV and Python by following color detection technique to detect position of the user with maximum accuracy and then location of user is determined by analyzing that location matrix. Location accuracy depends on the size of the matrix and successful identification of target color. Firebase real time database was added to the system which made real time operations between server and the user end device easier. To justify the proposed model, successful experiments were conducted in indoor environments as well and correct result was achieved each time by detecting accurate locations. This will be very advantageous to observe the fully or partially sightless people and guide them towards their destination and also to inspect them for their security purpose.**

Keywords—Image Processing, Indoor localization, Indoor positioning system, Wireless communication, Color Segmentation, Connected object detection

I. INTRODUCTION

Functional activities of visually impaired people are unavoidable concerns all over the world, for instance, almost 12.05 million people undergoes from this circumstance in Southeast Asian region [1] where most of the buildings are not properly equipped for easy movement. Moreover, though developed countries are very supportive towards the sightless people, but in the developing countries like Bangladesh, where almost 675,000 visually impaired adults exist according to a survey of 2000 [2], the scenario is opposite which describes the deprivation of technical supports.

White cane, trained dog or assistance from other people are some typical helpful approach for the sightless people to provide a reliable navigation. In addition, there are multitudinous devices and apps which are very efficient in guiding with outdoor directions using GPS nowadays with additional features such as voice command, smart walking sticks equipped with ultrasonic sensors and puddle detectors. However, these technologies are only supportive in case of outdoor guidance but not for indoor navigations as indoor positioning system is still under conjecture of researchers.

Multifarious ideologies and speculations have been established relating the Indoor Positioning System (IPS), for example, using Infrared, RFID, Wi-Fi, Ultrasonic sound, Audible sound, image processing, etc. After studying table 1 [3,4], it can be summarized that, though ultrasonic sound performs with more accuracy than others, but considering overall scenario with adaptability with smartphones and environment limitations, camera is more suitable for indoor localization using image processing than other approaches.

Table 1 Comparison of existing technologies in use

Technology	Positioning Accuracy	Adaptability with Smart phones	Environment Limitation
Wi-Fi	2-100m	Available with Smart phones	Requires open space
Ultrasonic sound	1-10cm	Not available with Smart phones	n/a
Infrared light	5-10m	Available with Smart phones	n/a
RFID	5cm-5m	Available with some Smart phones	n/a
Sound	5-10m	Available with Smart phones	Requires Silent Environment
Camera	1cm-1m	Available with Smart phones	n/a

Some experimental researches related to camera based indoor navigation systems like Aoki's Real-time personal positioning system [5] or Tilch's CLIPS [6] etc have produced good results but there are some limitations with these systems which need to be mentioned. Real-time personal positioning system developed by Aoki features big user burden and large processing time which makes it infeasible for real time use. Furthermore, using laser in the system, CLIPS has accuracy in cm range which can prove problematic for blind people having a risk of occurring skin and eye problems.

Therefore, to assist the unsighted or partially sighted people, a specially designed accurate indoor localization system is obligatory to help them become self-sufficient in navigating inside buildings. The main cornerstones of this system must include:

978-1-5386-3755-5/18/$31.00 ©2018 IEEE

1. Accuracy
2. Minimum user burden
3. Real time operation with fast processing time
4. Easy to control

Accumulating all of the focal points, in this paper, a system using camera and based on image processing with maximum accuracy has been proposed. Nonetheless, line of sight can be a tricky limitation while working with cameras, but there are possible solutions to tackle this issue as well.

II. Related works

Aoki in [5], initiated the proposal of image-based positioning system where a wearable camera and a laptop was utilized to capture images and hue histogram for image features. Afterwards, image matching system constructed upon SURF and SIFT [7, 8] was developed aiming more accuracy in case of location estimation. Nevertheless, these methods took a huge amount of computation time with several hundred seconds for each image and were inadequate for real-time applications.

Liang, Corso, Turner and Zakhor have contrived three stages of pipeline [9] which are,

1. Preparing a 2.5D locally referenced image database using a movable backpack equipped with laser scanners, cameras, and an OS controlled by human for mapping the intramural structure of a building for producing a locally cited 2.5D image database completed with SIFT features.
2. Image retrieval applying a kd-tree.
3. Pose recovery from the retrieved database image implying the profundity of SIFT which results in 6 degree of freedom pose.

Besides that, for evaluating the confidence values for both image retrieval and pose estimation, a method was introduced in their proposed image-based positioning system [9]. A twofold methodology for IPS has been introduced in [4], firstly, rapid database establishment with omnidirectional camera, image matching, and visualization interface and secondly, PCA-SIFT along with Locality Sensitive Hashing (LSH) [10] has fasten the image matching with addition of a "confidence" parameter.

In [11], after capturing images with almost 70% overlapping to create a query image database, a comparison between SURF and Affine-SIFT (ASIFT) has been done for a structured image database. Next, the Histogram of Oriented Gradients (HoG) has been applied for the detection of the pictures containing homogeneous objects or scenarios. A number of image sequences construct a topological map and learning vector quantization (LVQ) is used for clearness in the topology-based theory [12]. In order to discover regional pattern among the query images, nearest neighbor method has been applied in the searching phase. Sometimes this method results in wrong classification because of hypothesis about the distinctiveness of the navigation path in the topology-based map.

In brief, almost all the previous works done in the past related to indoor positioning with camera have used SIFT and SURF for feature extraction facing different errors in image classification. Moreover, in case of a bigger picture for instance, shopping mall or hospitals, these procedures can face more failures or errors detecting individuals from huge amount of indoor positioning of thousands of people and can misguide people in navigation. This is a huge disadvantage and matter of stress for the visually impaired people.

III. Proposed System

To overcome the erroneous situation mentioned in the previous section, in this paper a new model has been proposed for indoor localization using color extraction by image processing using camera where each individual will be assigned with different colors on their phones and cameras will be notified of that color by the system. If, two same colors are found in the same location by one camera, the system will change the color to let the camera detect each individual.

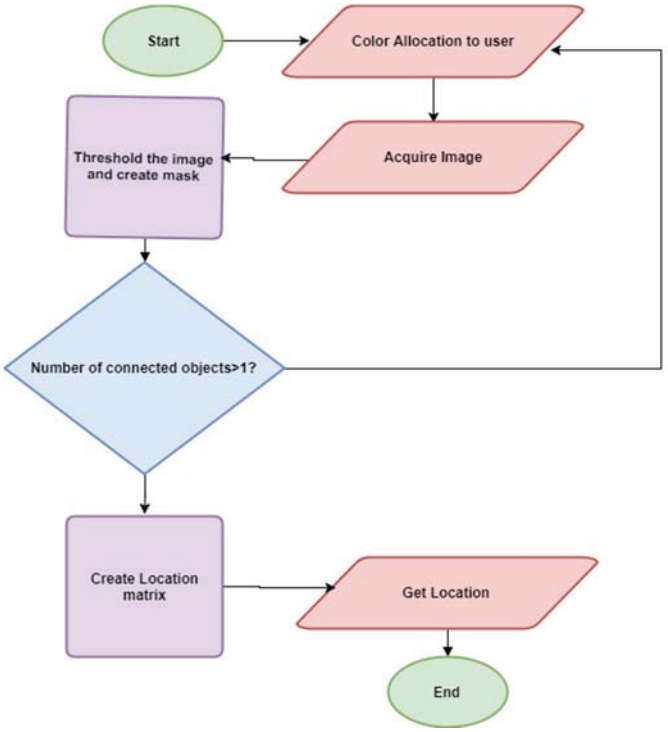

Figure 1 Flowchart of the proposed system

A. System Overview

Every camera is marked and tracked by the system by a unique camera id assigned by the system. Camera's coverage can be overlapping but the common parts are adjusted by the system and will increase accuracy. Every user device is marked with a unique user id and combination of user id and camera id leads to an accurate positioning.

To keep the user burden as low as possible, a server computer is used to complete all the heavy processing works. A Alongside the server computer there is a smartphone app to connect the user device to our server. Camera quality and the required coverage area dictates the number of cameras needed.

The images captured from the cameras are fed to a server computer which processes the images for required information. Server computer processes the captured images for required information. OpenCV and python are used for software development and image processing. Our system flow chart can be seen in Fig. 1.

B. *Reference grid*

After allocating all the cameras according to the resolution and covering range, a reference grid is assigned [13]. Picture taken from each camera by the system are called "Source Images" which are cleaved into a grid of square cells and a matrix of same size as the square cells is constructed.

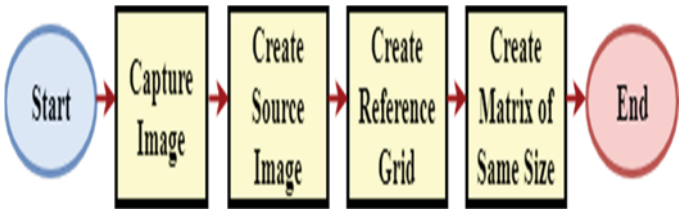

Figure 2Flowchart of Grid System

C. *Color allocation*

Soon after opening the application, the device gets an allocated color from the server. Because of external interference like sunlight, room light or energy bulbs, allocated color may seem different or undetectable. In these circumstances the allocated color is immediately changed to a new color.

A unique color is assigned to every device so that two different devices are not miss-identified as one. If the system runs out of colors, it uses the combination system to allocate color. The procedure that server follows for color allocation is given below:

1. When a new color request is received, the system checks the database for assigned colors and determines a unique color which is not assigned yet.

2. Then the color is send as reply to the device by generating a json response.

3. The color is saved in the database in "assigned colors" table so that it can be used later to get unique colors for other devices.

D. *Image Acquiring process*

After capturing images from the preinstalled cameras, which are taken with an interval of 1 second, they are converted into desired resolution for processing purpose. Lower resolution images also enable the use of cheap cameras which is essential for maintaining a low-cost system.

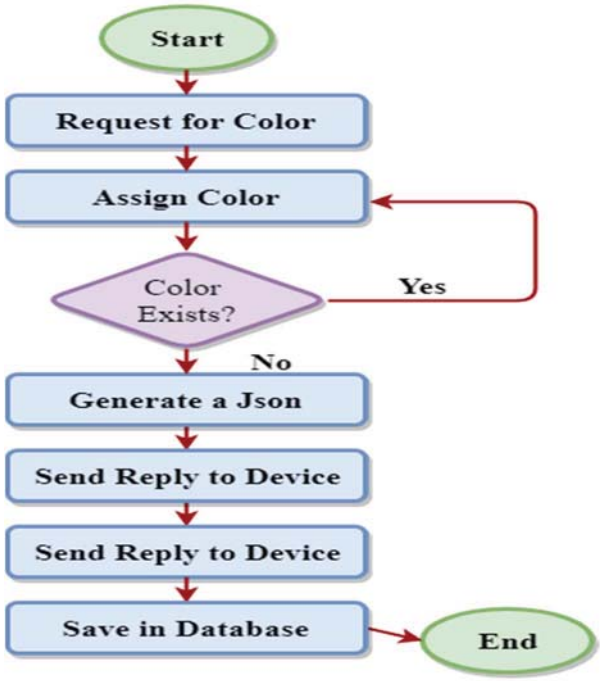

Figure 3 Flowchart of Color Allocation

E. *Color Processing*

At the very beginning of the process, BGR values of every pixel are stored in a variable for further processing. Then, the image is converted to HSV color space which allows easier processing of colors than RGB. After determining HSV values for required color, they are used to create a threshold for that specific color. Afterwards, a color mask is created using the threshold which is used on the image to keep only the desired color on the image. If the mask applied image returns zero value, then our user is either not detected or not in the coverage area. In this situation our system automatically starts working with the next image.

The processed image is then converted to grayscale to check for the number of connected objects. If the system detects more than one connected objects then this information is sent to the server immediately which prompts the server to allocate a new color to the user instantly. As the whole process is completed in split seconds when a new picture is taken, only one change can be observed and that would be the user end device with the changed color.

After processing the new image for the newly assigned color, the mask created for the previous image is used on the new processed image to locate the user.

F. *Location matrix*

The final processed image is then converted to BGR color space again to calculate pixel values. Mean BGR values of 100x100 pixels are stored inside cells. These values are analyzed to pinpoint the user. Cells with non-zero values indicate presence of the user. Cell sizes are determined in such

a way that one cell is just enough to allocate one user's position in entirety. So, if a user happens to stand on the intersection of two or four cells there can be two or four non-zero cells. Pixel values of those cells will then determine the exact position of the user. Higher the pixel value the more area is covered of that cell by the user.

G. Distance measurement

The navigated pixels (x,y) are calculated by using bfs and are mapped to real distance by multiplying the values using the following equation:

$$distance = d \times \frac{pixel\ length\ of\ the\ cell}{actual\ length\ of\ the\ cell} \quad (2)$$

Actual length of the cell has to be calculated manually using Equation (1)

H. Smartphone application

Users will login using the app. The android app will authenticate or register the user using firebase authentication. After successful login the android app will have a "Firebase User" which has a unique id. This unique id is used throughout the system as the "user id". In the firebase database. Under the "users" node a new node is opened using this unique id as "Firebase Push ID" (Firebase push ID is used in the firebase system). The data model used for users is also very simple. Two string type variables are taken for user name and user ID. There is another node similar to the "users" node called "cameras". Here string type variables are taken for camera device ID, camera type which can be of IP or usb and camera_id, which will be the unique push ID for the camera in firebase. Two other variables are taken to store X-Y co-ordinates of the camera coverage.

Every user is identified by a unique color which is assigned by the system when a user opens up the app and is added under the "assigned_colors" node with a unique push id, which will be marked as "assigned_color_id". This color is unique in the system and is checked before adding to the "assigned_colors" node. After successful add to the "assigned_colors" the user will have an "assigned_colors" object with string variables like color_hex, assigned_color_id and assigned_user_id. Color_hex will contain the hex code of the color. Other two variables will store the unique push ID of the assigned color and the user ID assigned to the color respectively.

The android program will assign the color in the screen, which will turn the screen to that color. The python program running in the server checks for changes in the "assigned_colors" node of the real time database using the python sdk of firebase. When there is a new entry in the "assigned_colors" node, the new child of the node is available and from that the python program will have the color which needs to be searched for. Looping through all the camera from the "cameras" node and running detection code will determine the camera which is observing the device or the user. This is how we will get the camera_id of the camera which is observing the user. Using the camera_id we will be able to calculate the exact location of the user using the "camera_x" and "camera_y" properties of the camera object.

IV. PSEUDOCODE

Table 2 Pseudocode of the proposed model

SERVER BACK END

```
Sub listenForNewUser()
    listen for changes in the child node named
   "detectionQueue" of firebaseRootNode
   If there is a change in the child node Then
       save it in Variable queue
       Call detectPosition(Parameter: Color from queue
       variable)
       If there is a current Position detected Then
          save it in variable currentPos
            Call notifyPath(Parameter: currentPos, targetPos
          from queue variable)
.       End If
    End if
End Sub
```

```
Sub notifyPath(Parameters: currentPos, targetPos)
    Call bfs(Parameters: currentPos, targetPos)
   If Success Then
       save it variable turns[ ]
       Call scaleToActualLength( Parameters: turns[ ] )
       If Success Then
          For i = 1 to Size of turns
             Add turns[i] to  child of "queue.getUserId"
             Which is a Child of "navigate"
              Which is a child of firebaseRootNode
          Next i
        End if
    End if
End Sub
```

MOBILE BACK END

```
Sub mainMethod()
            "someone@somedomain.someextension" ← userId
            Call assignUniqueColor()
            If Success Then
             save it variable color
             Display Parameter: color
             If Success Then
                        Call notifyServer (Parameter: color, userId )
                        If Success Then
                                    Call vibrate()
                        End If
            End if
End Sub
```

```
Sub assignUniqueColor()
            Call getRandomColor()
            If Success Then
              save it variable color
                If isColorAssigned(Parameter: color) == TRUE Then
                 Call getRandomColor()
                    If Success Then
                       save it variable color
                    End If
                End if
            End if
```

```
        Return color
End Sub
```

```
Sub isColorAssigned(Parameters: color)

    If value of color
        Which is a Child of "assignedColors"
            Which is a child of firebaseRootNode EXISTS
                Then
                 Return TRUE
     End if
     Else
         Return TRUE
       End Else
End Sub
```

```
Sub notifyServer(Parameters: color, userId)
        Variable queue ← new queue (Parameters: color,userId)
        Add queue to node "detectionQueue"
                Which is a child of  firebaseRootNode
End Sub
```

```
Sub vibrate()
        Listen For Changes in Node "userId"
                Which is a child of "navigate"
                        Which is a child of firebaseRootNode
        If Success Then
                save it variable dir
                If value of dir == "left" Then
                        Vibrate Device (Parameters: Number of
                Vibrates: 1)
                End If
                Else If value of dir == "right" Then
                        Vibrate Device (Parameters: Number of
                Vibrates: 2)
                End Else If
                Else If value of dir == "forward" Then
                        Vibrate Device (Parameters: Number of
                Vibrates: 3)
                End Else If
                Else If value of dir == "backward" Then
                        Vibrate Device (Parameters: Number of
                Vibrates: 4)
                End Else If
        End If

End Sub

Class Queue
        Variable color;
         Variable userId;
         Variable targetPos;
End Class
```

From Table. 2 we can see that, upon user opening and connecting to the network, the app will notify the server and call assignUniqueColor method. The method will call assign a color to the mobile phone display as based on a table which maintains all the assigned color. Upon user exist from the system, this color will get removed from the system. As the server gets notified by the app, a new entry is made in the queue table, with the assigned color and user id. This user id is taken from the user, upon sign up as the user email address which is unique in the world by its functional definition. The server creates an instance of the camera service that searches for the assigned color by taking snaps at a small interval (1 sec to 1 min, depending on the server load) by processing the image. In such a scenario, rectangular box filled with the assigned color is searched for. The result gives one or more pixel of the image which indicates the user position. Than the pixel is mapped to the original distance by calculating the centimeter per pixel for the image and corresponding image height and width. This process locates the user. For navigation the turns are calculated using bfs and turn are send out to the user via live database (firebase in our implementation). When notification is received in the mobile app, it is represented to the user via vibration. For right, left, forward & backward consequently 1,2,3,4 vibrations are made.

V. Experimental Result

The system was implemented inside a room with a single camera. A normal webcam was used as the capturing device and t+3sze54he area we covered was$1.26 \times 1.67m$. The images were converted to 500×300 resolution for the ease of calculation. 0.20 sec time was taken on average to run the code.

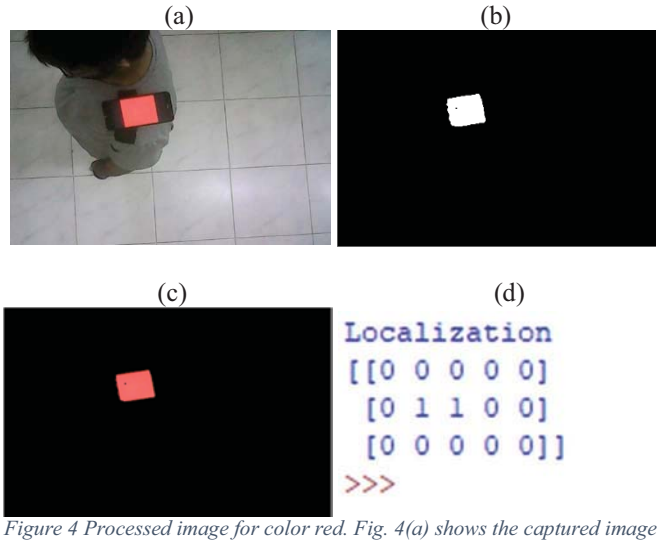

Figure 4 Processed image for color red. Fig. 4(a) shows the captured image and Fig. 4(b) shows the created mask. Fig. 4(c) is the final processed image while Fig. 4(d) shows the location matrix.

Figure. 4, 5 and 6 are for single connected objects in HSV color space. The process was repeated many times to ensure all of the scenarios were covered. Thus, the pictures were taken in day time under sunlight coming from the window as well as in night time with tube lights on. User's location was also changed in all possible ways to see if the line of sight from the camera to our user creates any problem or not.

In Figure 7, firstly, Fig. 7(a) shows the scenario of an image with multiple person got assigned with the same color or existence of multiple objects with same color. Secondly, Figure. 7(b) shows the first mask as a result of the processed image of 7(a). Next, Figure. 7(c) shows the masked image with color detection, afterwards Figure. 7(d) exhibits the new taken image with new allotted individual unique color. Finally, in Figure. 7(e), the new mask generated from the unique color of 7(d), while, Figure. 7(f) implies the up to date masked image of the

current color which is the ultimate processed image. At last, Figure. 7(g) has the location matrix generated from the reference grid and shows the processing time of 0.073395 seconds for the whole procedure.

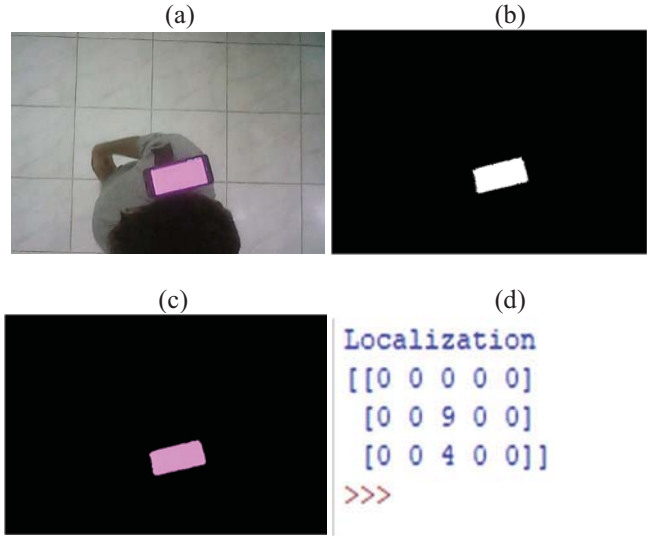

Figure 5 Processed image for color purple. Fig. 5(a) shows the captured image and Fig. 5(b) shows the created mask. Fig.5(c) is the final processed image while Fig. 5(d) shows the location matrix.

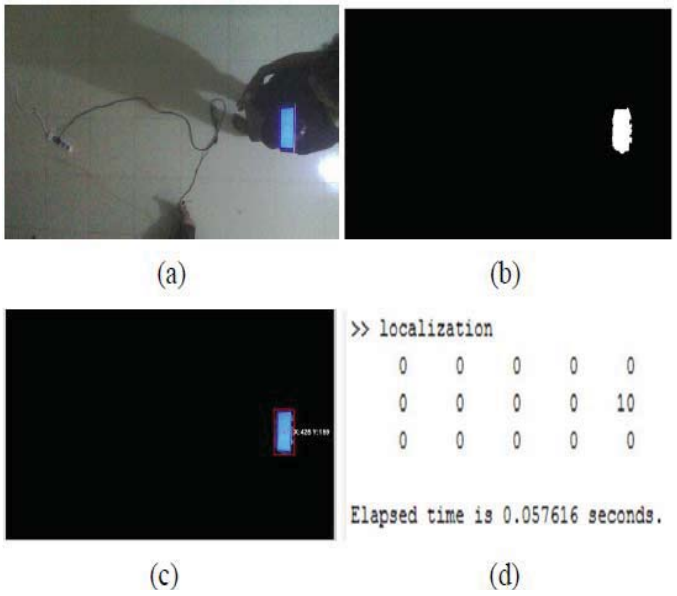

Figure 6 Processed image for color blue. Fig. 6(a) is the captured image and Fig. 6(b) shows the created mask. Fig. 6(c) is the final processed image while Fig. 6(d) shows the location matrix and elapsed time for the whole process.

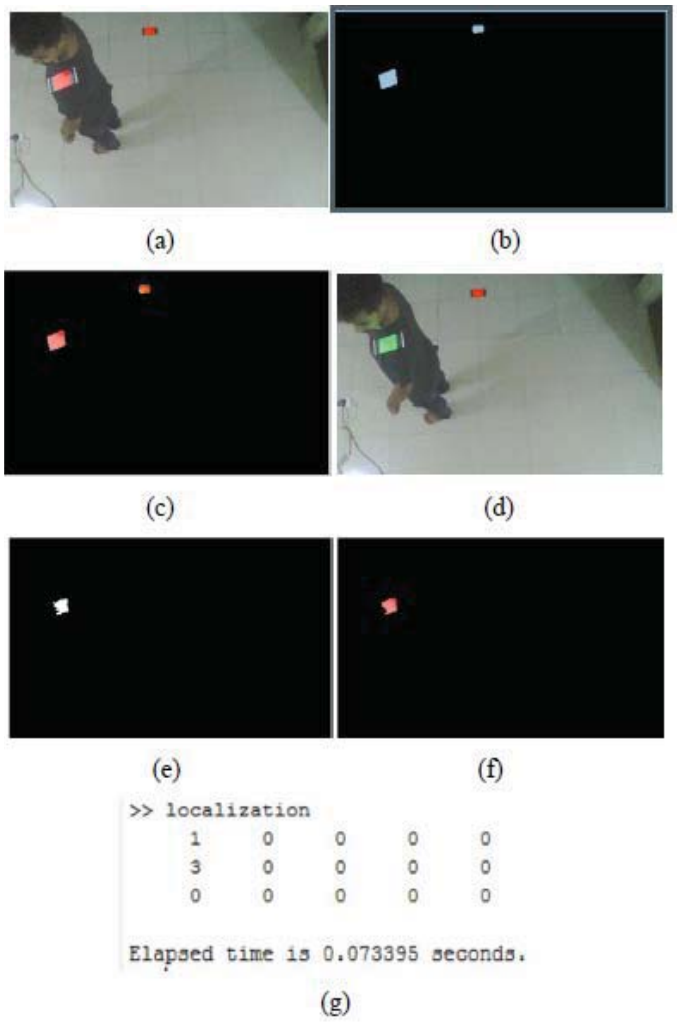

Figure 7 Processed image for multiple connected objects.

All of the experiments yielded good results and to get a firm error measurement of the system, distance between two objects were measured both manually and using the system. Results of those measurements are enlisted in Table 3. It can be seen from table 2 that the maximum error recorded was 1cm. Out of twenty distance measurement tests, seventeen distance accuracies were up to the decimeter mark.

Table 3 Distance accuracy

Case No.	Actual distance	Distance measured by our system	Error	Error Percentage	Processing Time
01	.685m	.6852m	0.0002m	0.029%	0.21s
02	0.93m	0.931m	0.001m	0.101%	0.352s
03	1.05m	1.059m	0.004m	0.85%	0.18s
04	0.41m	0.42m	0.01m	2.44%	0.11s
05	.72m	0.721m	0.001m	0.14%	0.53s

VI. Conclusion

An indoor localization system was developed earlier using MATLAB. But the problem with MATLAB was the resources it took to run was huge and cost of the software was also significant. Also working only in RGB color space limited the number of detectable colors which meant no multi user support. These issues were addressed by using OpenCV and python. Now the range of color has been increased which enables multi user support. The use of firebase real time database also made the system more versatile. One major drawback from the MATLAB based system is the increased processing time but the usage of open source software ensures simple and cheap implementation. Outputs were overwhelming as most of the times we got correct location with maximum error of 2cm and an average processing time of 0.08s. So, this paper provides solution to reliable and fast localization system for real time application.

The outputs were satisfying with 1cm maximum error which shows clear improvement of the system. Processing time can be reduced significantly by using simpler codes and faster server. With multiple user support and open source platform future work will be done to add more cameras and cover a large indoor area to develop a complete indoor localization system.

References

[1] WHO, "Global data on visual impairment," in *World Health Organization*, World Health Organization, 2012. [Online]. Available: http://www.who.int/blindness/publications/globaldata/en/. Accessed: Aug. 27, 2017.

[2] "National eye care,". [Online]. Available: http://nec.gov.bd/nio.php. Accessed: Aug. 27, 2017.

[3] K. Muthukrishnan, M. Lijding, and P. Havinga, "Towards smart surroundings: enabling techniques and technologies for localization", First International Workhop on Location- and Context- Awareness, LNCS3479, pp. 350-362 (2005)

[4] Kawaji, Hisato, Koki Hatada, Toshihiko Yamasaki, and Kiyoharu Aizawa. "Image-based indoor positioning system: fast image matching using omnidirectional panoramic images." In Proceedings of the 1st ACM international workshop on Multimodal pervasive video analysis, pp. 1-4. ACM, 2010.

[5] H. Aoki, B. Schiele and A. Pentland, "Realtime personal positioning system for a wearable computer," Wearable Computers, 1999. Digest of Papers. The Third International Symposium on, San Francisco, CA, USA, 1999, pp. 37-43. doi: 10.1109/ISWC.1999.806642

[6] S. Tilch and R. Mautz, "CLIPS proceedings," *2011 International Conference on Indoor Positioning and Indoor Navigation*, Sep. 2011.

[7] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: speed-up robust features", 9th European Conference on Computer Vision, pp. 404-417 (2006)

[8] R. Boris, K. Effrosyni, and D. Marcin, "Mobile museum guide based on fast SIFT recognition", 6th International Workshop on Adaptive Multimedia Retrieval, pp. 26-27 (2008)

[9] Liang J.Z., Corso N., Turner E., Zakhor A. (2015) Image-Based Positioning of Mobile Devices in Indoor Environments. In: Choi J., Friedland G. (eds) Multimodal Location Estimation of Videos and Images. Springer, Cham

[10] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni "Locality-sensitive hashing scheme based on p-stable distributions", in Proceedings of the 20th Annual Symosium on Computational Geometry, pp. 253-262 (2004)

[11] Huang Y., Wang H., Zhan K., Zhao J., Gui P., Feng T., "Image-Based Localization for Indoor Environment Using Mobile Phone", International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-4/W5, 2015, pp.211-215 (2015)

[12] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer vision–ECCV 2006* (pp. 404-417). Springer Berlin Heidelberg.

[13] M. M. Alam, S. E. Arefin, M. A. Alim, S. I. Adib and M. A. Rahman, "Indoor localization system for assisting visually impaired people," 2017 International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox's Bazar, 2017, pp. 333-338. doi: 10.1109/ECACE.2017.7912927