SBA: Smart Board Application Electronic Board Software System Development for Future Learning Spaces

Theodore Benito C. Coteok II
De La Salle University
Manila, Philippines
theodore_coteok@dlsu.edu.ph

Joseph V. Portugal
De La Salle University
Manila, Philippines
joseph_portugal@dlsu.edu.ph

Isaiah Jassen L. Tupal
De La Salle University
Manila, Philippines
isaiah_tupal@dlsu.edu.ph

Reginald Geoffrey L. Bayeta IV
De La Salle University
Manila, Philippines
reginald_geoffrey_bayeta_iv@dlsu.edu.ph

Abstract— **Human-computer interaction (HCI) is a field of study focusing on the interaction between humans and computers and it exists ubiquitously in our daily lives. Since HCI is achieved by using a physical controller such as a mouse or a keyboard, it hinders National User Interface (NUI) since there is a strong barrier between the user and the computer. The authors of this paper presents the first version of Smart Board Application (SBA), a machine vision based smart Electronic Board System (EBS) for future learning spaces, which aims to improve HCI and NUI by being able to track objects which will act as the pen and functioning as a reusable writing surface for creating texts, diagrams, drawings, and such as well as removing or erasing using the user's real-time pen object by utilizing Python and Open Computer Vision Library (OpenCV). The program works by using the computer's camera and separates and recognizes an object which will serve as the pen for creating drawings and diagrams. The electronic board or the display board where the user can draw on and the pen or marker will be the user's hand. Furthermore, the object and its motion will be tracked using color separation and paint will be displayed.**

Keywords—OpenCV, Python, color separation, electronic board, human computer interaction, feature space

I. INTRODUCTION

This paper presents the design and development of the first version of Smart Board Application (SBA), a machine vision based smart electronic board software system development for future learning spaces. Furthermore, this paper covers the specifications, interface, features, and overall functionality of the program. Moreover, this is the first version of SBA and is open for constant improvement and addition of features.

Vision-based technology of hand gesture recognitions an important part of human-computer interaction (HCI). HCI is a multidisciplinary field of study focusing on the design of computer technology and, in particular, the interaction between humans and computers. Since then, HCI has grown to be broader, larger, and much more diverse than computer itself. It no longer makes sense to regard HCI as a specialty of computer science and HCI expanded from its initial focus on individual and generic user behavior to include social and organizational computing, accessibility for the elderly, the cognitively and physically impaired, and for all people, and for the widest possible spectrum of human experiences and activities. It expanded from desktop office applications to include games, learning and education, commerce, health and medical applications, emergency planning and response, and systems to support collaboration and community. It expanded from early graphical user interfaces to include myriad interaction techniques and devices, multi-modal interactions, tool support for model-based user interface specification, and a host of emerging ubiquitous, handheld and context-aware interactions [1].

In this paper, the researchers present SBA, a machine vision based smart electronic board system for future learning spaces utilizing Python and OpenCV for minimal hardware requirement. The aim of this program is to provide basic functionalities of an electronic board system and promote HCI and NUI within the user and the computer.

II. REVIEW OF RELATED WORK

There are various techniques used for hand detection, examples include background subtraction, color detection, image segmentation, finger-tip detection, and so on. One hand detection technique known as the Viola and Jones detector and scale invariant feature transform based hand detection algorithm which provides result with high accuracy but with sensitivity to the background [2]. On the other hand, image segmentation is another technique used for hand detection, which uses HSV color space model rather than RGB color space for determining the color of the human skin which provides better background separation and region boundary. However, this technique will encounter problems when the skin color is of the same color as the background [3]. Another common technique is known as learning-based recognition in Adaptive Boosting algorithm that can integrate the information of same category of objects. It trains the network by combining all weak classifiers into one strong classifier. The Adaptive Boosting learning algorithm selects the best weak classifier from a set of positive and negative image samples. This algorithm provides result with better accuracy and fast speed but sometimes training period is more to train the network [4]. The mean-shift algorithm and the cam-shift algorithm (which is considered an improvement over mean-shift) is another object tracking technique designed to find color densities in a feature space (color space specifically). In this algorithm, the specific color (the chroma) matters in specific object tracking. The object is treated as one if the color of the object that is desired to be tracked falls under its shade of color, so color intensity doesn't matter too much [5]. Another object-tracking algorithm, the homography algorithm, is utilizing a "generated matrix" which is a mapping of a given set of points in a provided image - to the corresponding set of points in another image (in this case corresponding it with the real-time "video"). In the algorithm's practical case in this context, key features of the sample image of the object are to be automatically set to as those points; and corresponds them with the real-time video, tracking desired object [6].

.

Numerous existing algorithms are available for hand detection, another technique is finding convex hulls and there are multiple existing algorithms for finding convex hulls and most of the time, more than one is algorithm is used for finding convex hulls since tracking hands and its motion can be complicated and some gesture and its motion can be quite complicated [7]. The reason for these algorithms since recent trends of computer vision techniques are easy, natural and less costly since most of the laptops have an integrated webcam along with it so it is an easily available device [8]. This is an important feature as gestures can contain more information than simple words and it reveals more information that sometimes cannot be found in speech. Hand gesture recognition has been widely used in human–computer interaction, like in virtual reality, robotics, computer gaming and so on [9].

A. *Planning Stage*

In designing SBA, the following has been considered during the planning stages, namely:

a) The program being able to track and object which will serve as the pen for the electronic board.
b) The program being able to draw by tracking the object real time.
c) The program being able to utilize an erase mode function to delete a portion or all the writings made by the user.
d) The program is limited to separating the color of the object from the background.
e) The program can export files to an image file or do a screen recording mode and export to a video file..

Furthermore, a user-case story was developed to help the designers in developing the software from the end user perspective and help meet the requirements needed.

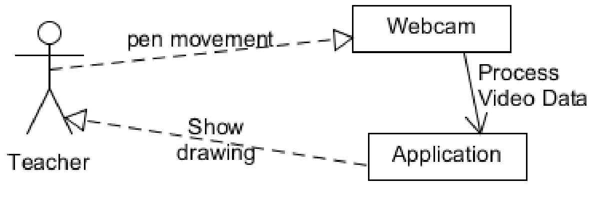


Figure 1.1 Art Teacher Case Story
As a creative preschool teacher, I want to be able to write on the screen when I move my pen.

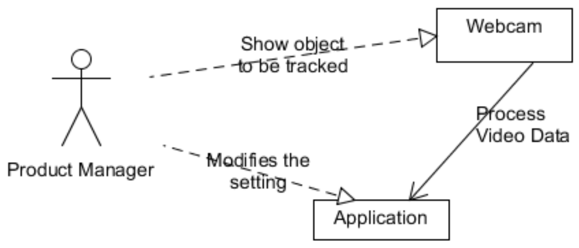


Figure 1.2 Manager Case Story
I have a bunch of markers in my house, so I want the app to fit my specific marker color. I also want it to track whatever I have at the moment so settings for which to track is what I need.

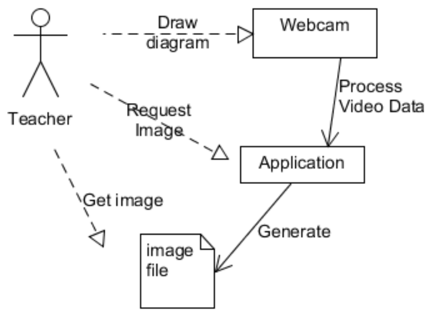


Figure 1.3 Science Teacher Case Story
I want my diagrams to be saved to an image file so I can share my diagrams to my students when I conduct an online lecture.

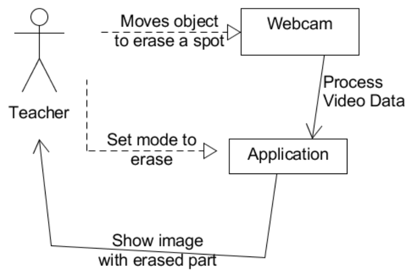


Figure 1.4 Teacher Case Story
I make a lot of mistakes, so I want it to have a feature that gives me the ability to erase sections of my diagrams.

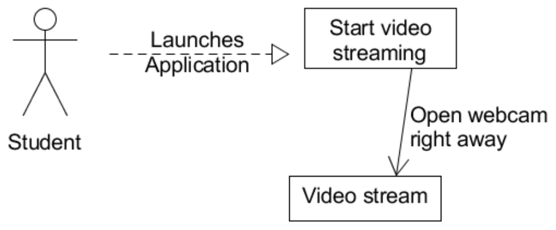


Figure 1.5 Student Case Story
I am always busy, so I want the program to go straight into the board when launched.

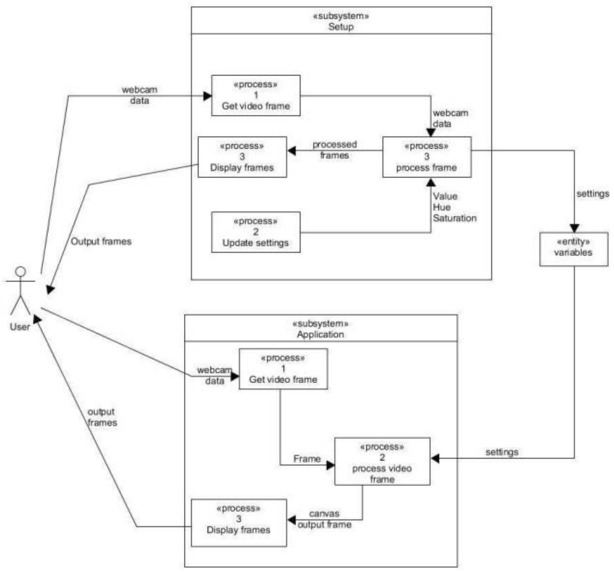


Fig. 2. Use-Case Diagram

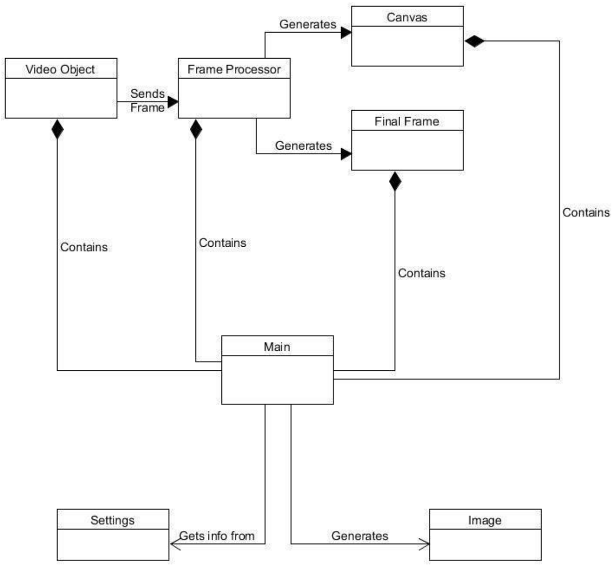


Fig. 3. Object Diagram

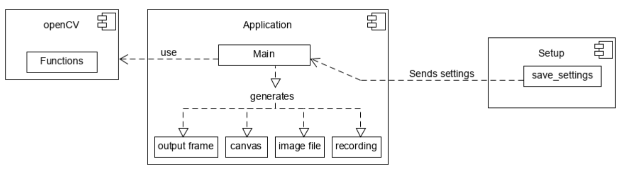


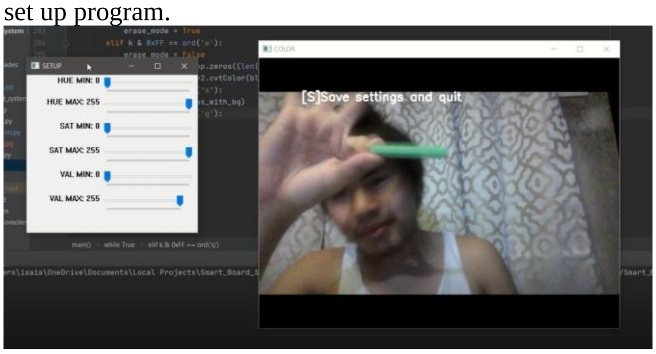
Fig. 4. Prototype Architecture

III. RESULTS AND DISCUSSION

A. *Color Isolation*

Isolating the color was done using the inRange function of the openCV library. The package deals with images through the HSV color space and inRange function removes the pixels (by setting it to zero value) that do not fit within a certain HSV parameters. One of the issues with this is that since the objective is to let the user decide which to object to track, the HSV parameters should be variable by the user. This is done by making the user go through a set up phase.

The idea behind the set up phase is to let the user isolate the color of the object on their own in a simple way. The key here is to provide instructions which value to change and make it simple to use. Fig 5 shows the sample UI of the set up program.

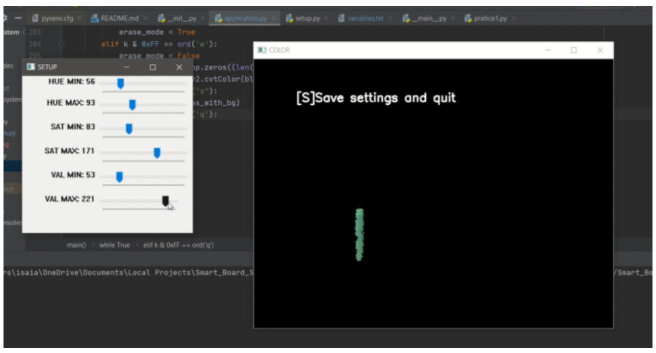


HUE MIN: 0
HUE MAX: 255
SAT MIN: 0
SAT MAX: 255
VAL MIN: 0
VAL MAX: 255

[S]Save settings and quit

Fig 5. HSV settings of the setup program

The inRange function takes in 6 parameters which are the minimum and the maximum of the following: Hue, Saturation, and Value. When the set up is launched, the maximum of each parameter is set to the absolute max (which is 255) while the minimum is set to the absolute min (which is 0). The process of isolating the color is simple, the minimum is moved closer to the other side until the color of the object disappears. This is also what should be done to the maximum just in a different direction. Fig 6. shows a completely color isolated object. This frame in which the color is isolated is referred into this paper as the mask or mask frame for brevity.



HUE MIN: 56
HUE MAX: 93
SAT MIN: 83
SAT MAX: 171
VAL MIN: 53
VAL MAX: 221

[S]Save settings and quit

Fig 6. Completely Isolated color

B. *Writing*

After setting the values for the program, the next step is tracking the object and making it write on the screen or canvas. This entire process is broken into two: tracking and drawing. The idea behind the first one is discussed in the previous chapter with additional concepts.

After the color of the object is isolated, the edges of that object are then detected using the getContours method from the opencv library. The contour method returns a contour object wherein its location, perimeter, area and rectangle that bounds it can be computed via a simple function. Algorithm 1 presents this process.

Algorithm 1 get pen point

```
Input mask_frame, previous_point
Output point
1.  contours = cv2.findContours( mask_frame )
2.  try
3.      if cv.contourArea(contour) > 10 do
4.              x,y,w,h = cv2.boundingRect(contour)
5.              return ( x+w/2 , y+h/2 )
6.      else
7.              return point
8.   except  noContour
9.      return ( -1. -1)
```

After getting the point, the program now decides whether or not the program will draw a line or a point on the screen. If the get pen point function returns a negative value, nothing will be written. If the previous point is the negative value then the only thing that would be written would be a point. A line will be drawn into the canvas if the previous point is not a negative value and the get contour function returns a valid point. This process is presented in algorithm 2.

Algorithm 2 draw on canvas

```
Input canvas, erase_canvas, point, previous_point,
erase_mode
Output canvas
1.  if point == (-1, -1)
2.      pass
3.  else if prevous_point == (-1, -1)
4.      return draw_point_canvas_point(canvas,
    erase_canvas, erase_mode)
5.   else
6.      if not erase_mode
7.              return cv2.line(canvas,poin1,point2)
8.      else
9.              return
    cv2.line(erase_canvas,point1,point2)
```

Erase mode is a boolean value and erase_canvas is the canvas wherein if erase mode is true, the point or line will be written there instead. The idea behind this is that when erase mode is enabled by the user, the drawings will be written there instead of the actual canvas. The point of having two canvasses is for erasing.

C. Erasing and Displaying

When erase mode is enabled, the drawing will be written on erase_canvas. Using openCV functions, erasure can be done by using the erase_canvas as a subtraction to the actual canvas. This is done for every loop which means that regardless if erase mode is true or false, the erase_canvas is always subtracting the canvas producing the final_canvas This is presented in algorithm 3.

Algorithm 3 get final canvas

```
Input  canvas, erase_canvas
Output final_canvas
1.  erase_canvas = cv2.bitwise_and(canvas, erase_canvas)
2.  final_canvas = cv2.bitwise_xor(canvas,erase_canvas)
```

The first bitwise operation means that if a certain pixel has the same color between the two frames, it stays. If that pixel, however, has no color in one of the two canvas, that pixel is converted to a zero value. The idea behind here is that everything that is not overlapping is removed. The second line on the other hand is a xor function which entails that the pixel will only remain if only one of the two has color. So the overlapping part of the erase_canvas is now subtracted from the final_canvas. This is the very principle of erasure in this program. Additionally, there is also an option of one-time erasure or clearing the whole canvas, which can be used when moving on to the next illustration.

D. Recording

To enter record mode, keyboard button ‘R’ must be pressed to start recording everything the webcam and the electronic board sees and waits for the user to press the keyboard button ‘T’ to stop the recording and export the file to a video file. This is useful for creating videos showing how a certain drawing is made using the smart board application.

E. Usability

The most important part of the project is its overall usability. The developers tested this product and are pleased with the results as it tracks the object accurately when the background is clear. Fig 7. is the sample usable scenario

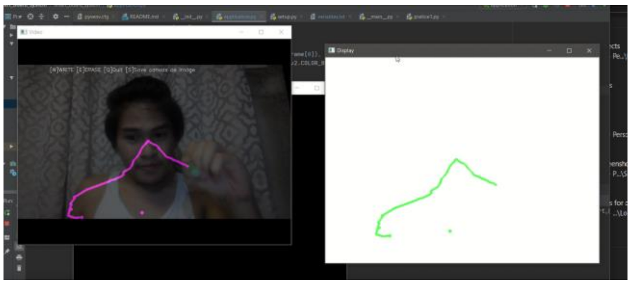


Fig 7. Usable state sample.

The issue arises there are objects in the background with the same color as the object being tracked. This falsely detects random images as the object which can ruin the drawing.

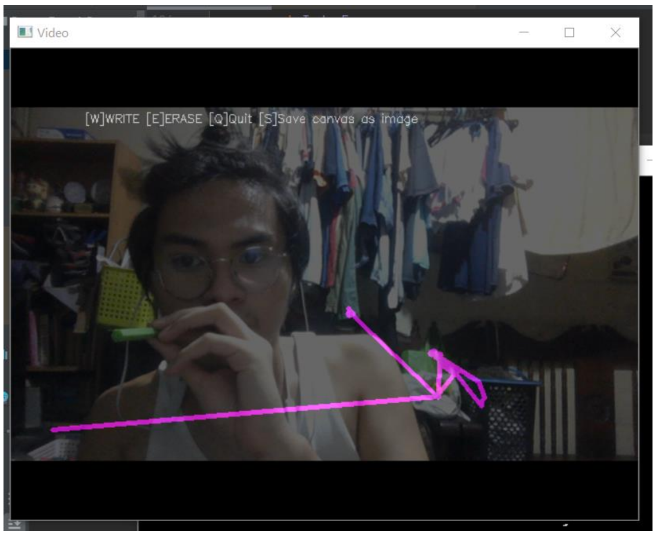
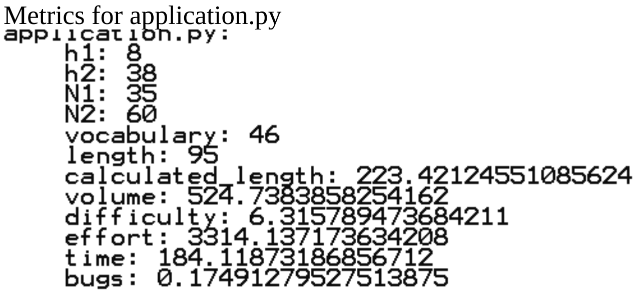


Fig 8. Unusable state sample

F. Software Metrics

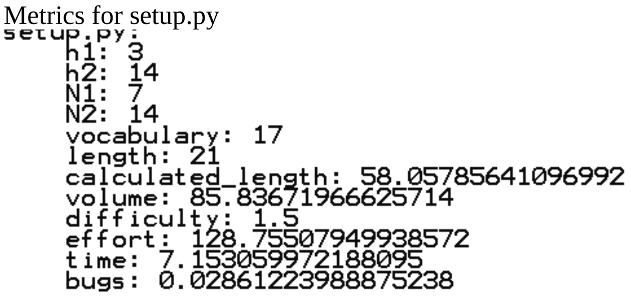
Measure of software characteristics which are measurable or countable are measured in this program which is useful for measuring software performance, productivity, and more. This program utilized Halstead’s Metrics, counting tokens and classifying them into operators and operands and the metrics were generated using Radon, a Python tool that computes various metrics from the source code.

Metrics for application.py



```
application.py:
    h1: 8
    h2: 38
    N1: 35
    N2: 60
    vocabulary: 46
    length: 95
    calculated_length: 223.42124551085624
    volume: 524.7383858254162
    difficulty: 6.315789473684211
    effort: 3314.137173634208
    time: 184.11873186856712
    bugs: 0.17491279527513875
```

Metrics for setup.py



```
setup.py:
    h1: 3
    h2: 14
    N1: 7
    N2: 14
    vocabulary: 17
    length: 21
    calculated_length: 58.05785641096992
    volume: 85.83671966625714
    difficulty: 1.5
    effort: 128.75507949938572
    time: 7.153059972188095
    bugs: 0.02861223988875238
```

IV. CONCLUSION

In this paper, SBA, a smart board application electronic board software system development for future learning spaces was designed and developed. The program was able to achieve the minimal functionalities of a smart board application and met the requirements set during the planning phase: have the program detect an object that will serve as the pen or marker, allow that marker to make drawings and diagrams based on the movements made by the user, have an eraser function that will allow the user to erase or correct his or her drawing, and have a function where the work made on the canvas be exported to an image file. Nonetheless, SBA’s first version is a functional smart board application which can be used by many. The program can still be improved in certain aspects such as better algorithms and better tracking. Moreover, better cameras will also lead to better performance for the software since the image and video capture is clearer. The program does what it does, promote HCI with minimal hardware requirements. It is definitely a subject for improvements.

V. FUTURE WORK

The software can be used as a baseline for future software developments, feature-wise. Since given its current state, the features to track an object, draw, erase, clear and capture an image or video are already present, thus these features can be further expanded upon to improve the overall functionality of this software. For example, the feature to draw can possess “pen” color changing capabilities, depending on the user’s desire. The feature to capture multimedia (image or video capture), can also possess the ability of media playback as well. Most notably, the ability of the program to track objects may include not only object recognition and tracking, but gesture recognition as well - to perform program functions; given further future developments of the SBA.

REFERENCES

[1] J.M. Caroll, *Human-computer interaction in the new millenium*, 2002, New York: ACM Press.

[2] Kenji Oka and Yoichi Sato "Real-Time Fingertip Tracking and Gesture Recognition" IEEE proceeding on Computer Graphics and Applications Nov/Dec 2002

[3] Qing Chen Nicolas, D. Georganas, and Emil M. Petriu "Hand Gesture Recognition Using Haar-Like Features And A Stochastic Context-Free Grammar" IEEE ,Vol. 57, No. 8, August 2008

[4] Soowoong Kim, Jae-Young Sim, And Seungjoon Yang "Vision-Based Cleaning Area Control For Cleaning Robots", IEEE Vol. 58, No. 2, May 2012

[5] Demirovic, D. (2019). An Implementation of the Mean Shift Algorithm. *Image Processing On Line*, pp. 251–268. doi: 10.5201

[6] DeTone, D., Malisiewicz, T., & Rabinovich, A. (2016). Deep Image Homography Estimation. doi: 1606.03798

[7] Kenji Oka and Yoichi Sato "Real-Time Fingertip Tracking and Gesture Recognition" IEEE proceeding on Computer Graphics and Applications Nov/Dec 2002

[8] G. R. S. Murthy & R. S. Jadon "A Review Of Vision Based Hand Gestures Recognition" International Journal of Information Technology and Knowledge Management, July-December 2009, Volume 2, No. 2, pp. 405-410

[9] M. Novack and S. Goldin-Meadow, “Learning from Gesture: How Our Hands Change Our Minds,” Educational Psychology Review, vol. 27, no. 3, pp. 405–412, 2015.