Development of Virtual Pen Program with Object Detection and Tracking

Author: 19-124 Jeuk Hwang positiveperiod.dev@gmail.com

Author: 19-039 Sunbin Park loganpark5617@gmail.com

Director: Omitted

Mathematics and Computer Science, Korea Science Academy of KAIST

Abstract

It's not easy to show others what you think when you can't use a note, pen, especially having a virtual meeting without a touchable monitor. Since COVID-19 gets worse, these situation keeps people make annoying when they have critical moments as sharing the formulas or hand drawings during online classes. The program mainly based on Python and OpenCV can be used with any circular daily-life object having uniform color. Using various image processing methods enables the user to detect and track the color of the object and display what you write in the air as using no additional equipment but a laptop with a webcam. Although the detection quality often depends on the light intensity of the environment, it shows somewhat functional performance. New algorithms using machine learning and convenient GUI can potentially develop the quality of it.

Keyword: object detection, object tracking, image processing, virtual pen, COVID-19

Execution video: https://www.youtube.com/watch?v=7ez4rfBx4nU

Code: https://github.com/PositivePeriod/Touchable

The Ministry of Science and ICT funded this work.

Introduction

Online classes and meetings have become more popular because of the effect of COVID-2019. Although online courses have several advantages, they also have some inconveniences. One of the major problems is the regulation of freedom of expression. Although telecommunication technology enables people to discuss while breaking down the barriers of physical distances, it is hard to send the information expressed as the drawing by hand, which has contents that is hard to convert into the text. To be specific, many students struggle to convey the equation and figures in solution during online math classes.

Furthermore, though most students have a mouse, it is quite hard to write or draw well with the mouse. However, students who have a laptop with a touchable screen don't experience much uncomfortable. Thus, to solve the problem, the attempt to make the software that can supplement the hardware defect is started. Although the early concept started from eyeball tracking and gesture detection, as considering the difficulty and practicality, the final program uses daily life items such as pens to write what the user wants by detecting and tracking the object.

Background & Attempts

The program focused on executing even in low-performance hardware, the laptop's webcam, without any additional specialized device.

There are many ways to draw and write on a virtual screen. One of the methods is hand gestures, but using gestures is quite hard because the program should be accurate. However, hand gestures can't be precise with an ordinary webcam with low resolution. Furthermore, the detection of hands' motion is hard and usually needs some technology about machine learning. Using eyeball tracking leads to a similar situation comparing to the use of the hand gesture.

Additionally, although eyeball detection might be possible when the person is close to the webcam, detection of the motion of eyeball is hard since eyeball has the too small angular distance to detect with low-resolution webcam. It can be easily calculated as assuming the maximum amplitude of eyeball and the average distance between the person and the laptop's webcam. People cannot control their eyeball motion since eyes usually move very fast without realizing, which will cause serious confusion for using the program. Moreover, most of the successful project about eyeball detection looks like to rely on the products form Tobii company.

Therefore, the team changed the focus and thought about tracking a specific object that is not part of the human body. Since the object should be easy enough to obtain and common and detectable, its color should be different from the background, and its shape should be a circle, which is one of the most

comfortable shapes to detect. Its perfect symmetry leads to the high accuracy of tracking and comfortable use to users when they use it to write or draw. From these conditions, the team concludes that users should use objects with circular shapes with uniform colors such as a pen or some cosmetics. Since the user can choose an object which they will use, the user can use different colors for different situations, which enables expecting a more optimized solution for users.

There are several approaches to detection and tracking, including color detection, shape detection. Before explaining the color detection, the definition of the color is essential to know. There are many ways to classify the color, and one of the common ways to express the arbitrary color is the RGB model, which represents the color as a mixture of red, green, and blue. However, since RGB is not easy to use when the color is changed, for example, if the color is changed from blue to red, it will change both the red and blue channels.

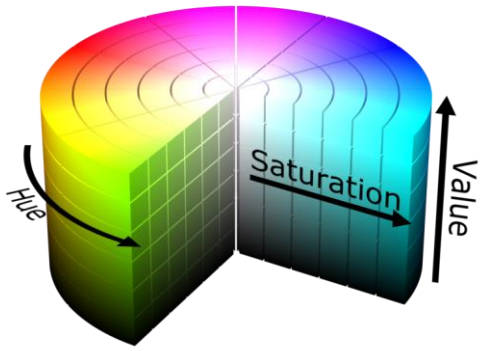


Figure 1. The color space of HSV in the shape of solid cylinder from Wikimedia

However, HSV model using hue, saturation, and value, can usually do color classification just by using the hue channel, since the light intensity is only concerned with the saturation and value, not hue. (Figure 1. and Figure 2.) Therefore, the transition from blue to red can be detected by the change of the hue channel. Thus, the HSV model and its alternative representations like the HSL model and the HSB model is usually used for image processing and computer graphics.

To detect the object by color difference between background and other objects, one of the most critical processes is deciding proper color well expressing the object's feature. For a comfortable and straightforward way, the average color of the object can be used. However, since the area in the image that is not the object can occur a big difference between the real average color and calculation results. For instance, for the circular blue object on the red background, the image's background might lead to some difference, and the average color might be purple rather than blue. Therefore, to use the average method, most raw images should contain the color of the object. The early program that the team makes also uses the average method with the square inscribed in pen. (Figure 3.)

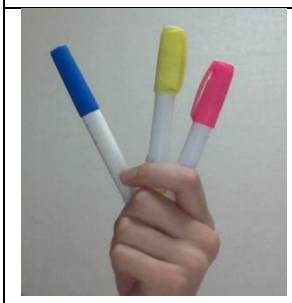
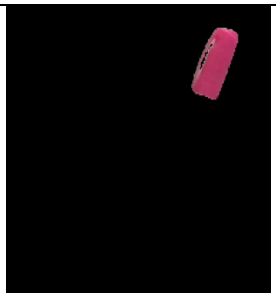
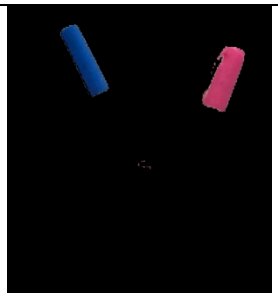
<table>
<tr><td colspan="4">Filter by HSV</td></tr>
<tr><td rowspan="2">None (Original)</td><td>range of H: 138~179</td><td>range of H: 81~179</td><td>range of H: 25~179</td></tr>
<tr><td colspan="3">range of S: 80 ~ 255
range of V: 135 ~ 255</td></tr>
<tr><td>  </td><td>  </td><td>  </td><td>  </td></tr>
</table>

Figure 2. Example of object detection by using color difference in HSV model

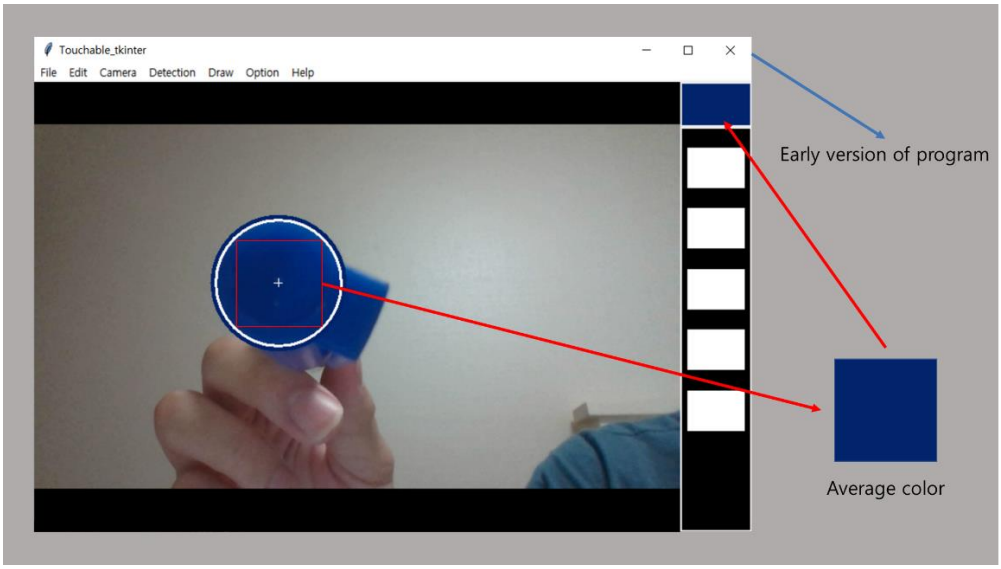


Figure 3. Determining of the color of the pen by using average color of specific area

Not only the average method, there are but also other ways to get the representative color of objects such as dominant colors. The dominant colors of the image can be determined by using color space which is the space that each point represent the different color as the axis means the variable in each color model. For example, figure 1. represents the color space in HSV. After plotting each pixel's color in the color space, the k number of dominant colors can be calculated by using clustering algorithms. However, it consumes too much time to apply for the real-time program, especially when the k is more significant than 3. (Figure 4.)

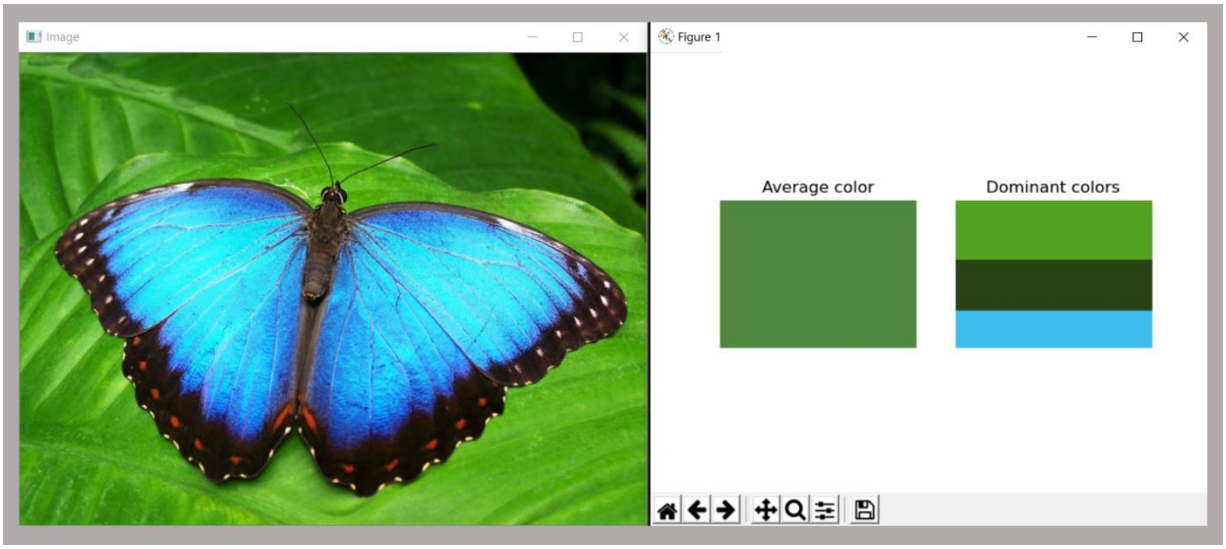


Figure 4. Example of determining dominant colors and average color of image

Moreover, since the light intensity changes depending on the environment and the hardware itself also control the image's intensity by image processing and control the aperture, the color of the object can not be assumed as constant. Therefore, the positive feedback by using the calculated color for each frame can be considered to calibrate. Although it can optimize the object's color ideally in reality, the small error can make a big difference and even stop the detection because of too different colors between real objects and incorrectly calibrated color. Thus, some standards for the object color is also needed for the initialization after incorrectly calibration occurs. Even though the calibration by positive feedback leads to a good result average color is enough to most cases if calibration gets the worse result.

The shape of the object can also be used for detecting and tracking it. This project is relatively easy because the program ideally only needs to find the circle, the shape of perfect symmetry. However, the program also needs algorithms for detecting ellipse since the object can have a form of ellipse depending on the angle of view in reality. The tracking circle is to implement since a Hough transform can be used to identify any type of shape represented as the mathematical equation like circle and line. Additionally, it is obvious to detect the ellipse is harder than the circle since arbitrary ellipse needs five variables: the center, minor axis, major axis, and the angle, on the other hand, circle only needs the three variables, the position of the center and radius. Some algorithms exist to recognize the ellipse, but most of them are very hard to implement from scratch. Although the algorithm mentioned in one paper is easy to implement, it even takes lots of time to detect the big definite ellipse, which is not acceptable for real-time programs. (Figure 5.)

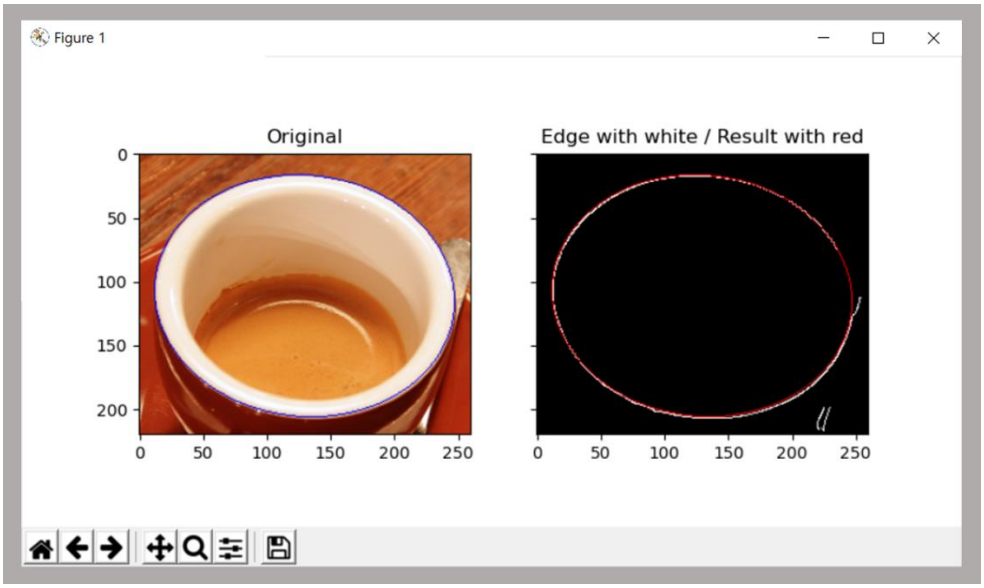


Figure 5. Xie's algorithm mentioned in consumes more than 5 seconds to find the ellipse.

The processing can also focus more on the object's size, not the shape by using contour. To understand it, blur and canny is also essential. Blurring is the applying filter like a gaussian filter and bilateral filter to reduce the noise. The Gaussian filter is good at reducing the Gaussian noise with no specific features but similar to the typical noise. The median filter is good at reducing the salt and pepper noise, which occurs as some random sharp white and black pixels. The bilateral filter is good at maintaining the edge of the image. Like these filters, each filter has different advantages and disadvantages as applying different matrix for each pixel. After blurring, the image should be grayscale to use the canny algorithm. Contour is a special kind of line that connects points having similar features. It can well detect the shape of the object but also has lots of mistakes, but mistakes can be reduced as filter by a minimum of area and eccentricity. Similarly, the Canny algorithm is another famous edge detection algorithm that applies Sobel operators for each axis to calculate the change rate of color and detect the edge by filtering the change rate.

Background subtraction is also used for reducing the error by enabling to ignoring pixels that look like background and have a similar color for a long time. There are various algorithms with different characteristics. For instance, some of them need initialization by beginning frames, and others make too much after image. The team experienced multiple subtractors, including MOG, MOG2, and GNG, but finally considered to use KNN that is supported by extension of OpenCV, which does not require initialization and much afterimage since it looks acceptable for real-time program.

As realizing making new tracker is very hard, the team also researches tracker algorithms like TLD, KCF, and GOTURN supported by OpenCV. A similar concept is little used for the final program from

the inner structure of the tracker and detector that exchanges the effect. Some disadvantages of using subtractors are slow and might remove the proper parts if the parts do not move for a while. Thus, there was an attempt to erase the object's last frame before applying subtractor, but it gets some problems. However, implementation was quite successful as making a mask with the shape of quadrilateral and circle. (Figure 6.)

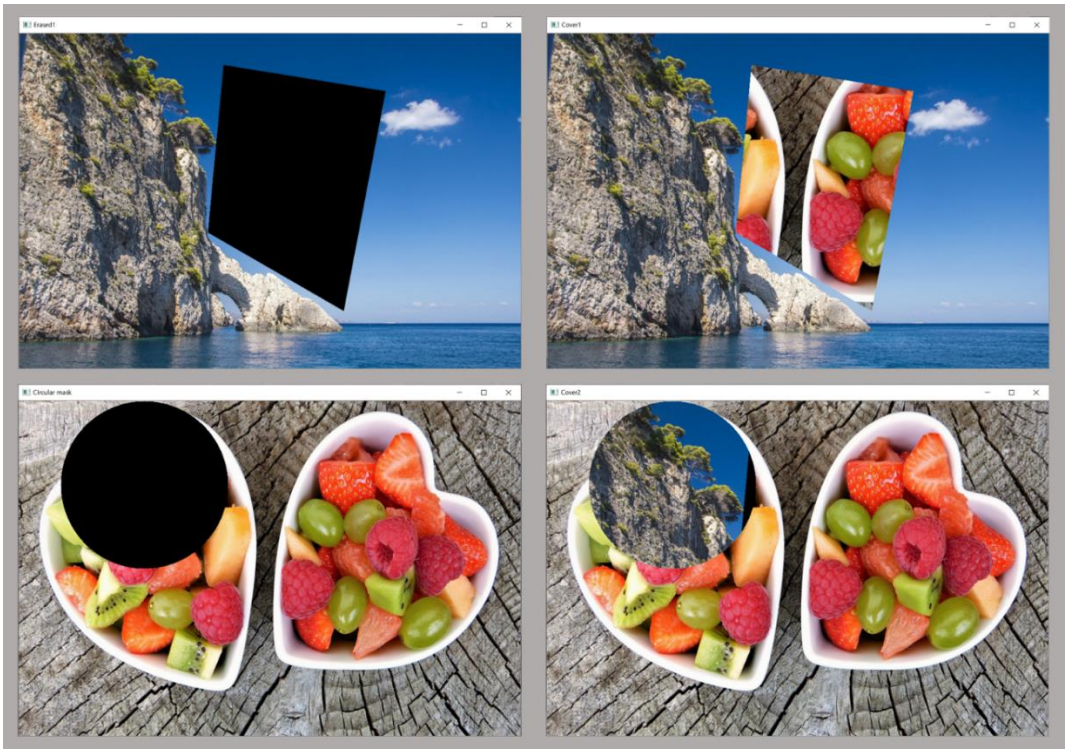


Figure 6. several shape of masks that can be used for morphing

Result

The program recommends on the laptop with a webcam that can support 1280×720. Then, Python 3 and the following python modules are a requirement on the device; Python3 tested on 3.8.2, OpenCV-contrib tested on 4.2.0, NumPy tested on 1.18.3, PIL tested on 7.1.2, SciPy tested on 1.4.1.

The program can be divided mainly into two parts; detecting and tracking pen, showing the user's drawing with the graphical user interface. Since the user can use various pens with different colors, the pen's information is saved as the Pen object. To initialize and make the Pen object, the user should fill the red square with the pen to enable the program to recognize the pen's color. If the user has a Pen object, the program uses some image process methods like background subtraction by KNN subtractor, blurring, and HSV filter. After that, the program tries to find the biggest contour to find a pen. Separately from its algorithm, MOSSE tracker and SimpleBlobDetector also try to find a pen if the previous fails.

Three methods try to cover other algorithms' disadvantages to make better overall results. Additionally, tracking pen also considers the change of the object's position and color in each frame to reduce the error since the location and color of the object usually does not change suddenly. After detection, the users can draw various things by using the program; for example, users can draw points, straight lines, circles, polygons, and smooth curves using cubic spline interpolation. Because of the limitation of hardware, the delay is inevitable, which occurs discontinuous input. Therefore, the team uses interpolation to let the user draw continuous and natural curves. Also, at the beginning of the development, the program has too much delay with getting the image from the webcam; thus, the team solves the problem and enables to maintain the higher frame rates with making additional Video object which always updates frame from the camera using threading. Since the program is quite complicated to explain the whole, only the flowchart about the detection part will be shown. (Figure 7.) (Figure 8.)

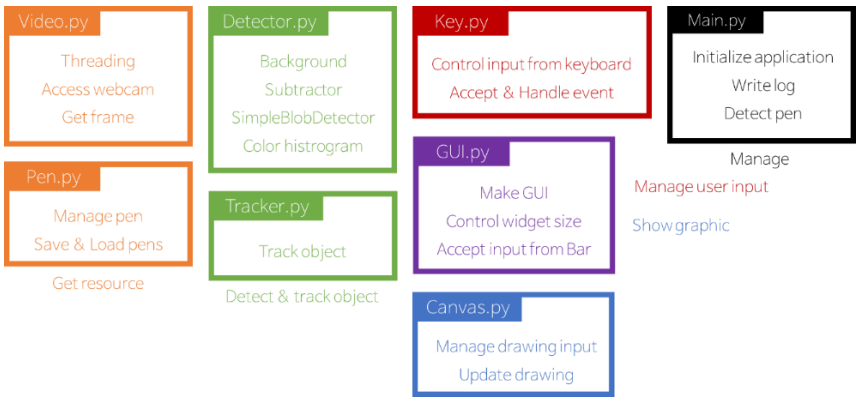


Figure 7. Brief structure for major files

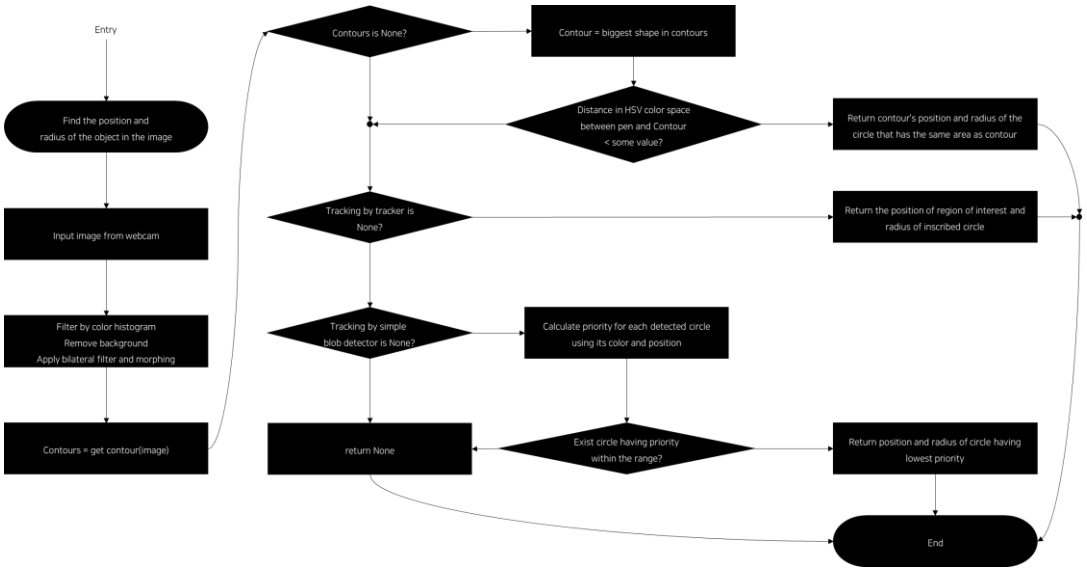


Figure 8. Brief flow chart for detection objects

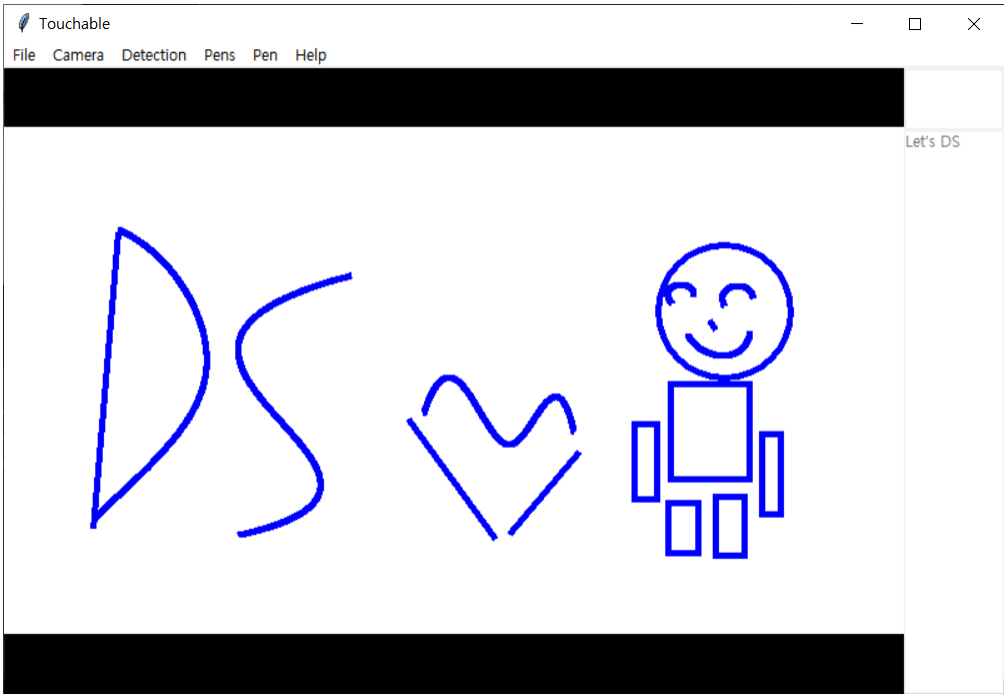


Figure 9. Example with things that can be done with the program

Figure 9. well shows one possible example that can be done by the program and the student's passion for the data structure class.

Conclusion

As the program uses various methods such as background subtractor, contour, and SimpleBlobDetector, it can do its intended job well, as shown in Figure 9. Not only making a program, but the team can also learn many additional things from the project; image processing, the way to use external modules without having detailed documentation, and how to cooperate with the team member. Although the central part of the program about detection shows worse result than expected, the team sets blazing rather than sad since realizing object detection is one of the most enthusiastic and hardest areas in computer science, especially in these days when highlights the computer vision. The team will make an effort more to become better while wishing that someone can get some help from the project.

Discussion

Although the current program shows impressive results under some environment, more accurate methods like machine learning need for generality, its recognition is not valid when the light intensity keeps changing or the object is glowing because of the reflected light.

While doing the poster presentation, the team got some feedback; one was connecting it with other existing programs like Word, PDF, and PowerPoint. It might be edited and uploaded to use as a portable web version, which gives much better user accessibility for several devices.

Additionally, since the current graphical user interface is based on Tkinter, which has lower expandability, the program can be overall remade as using PyQt, which enables developers to make a much user-friendly user interface as supporting drag-drop, and dynamic design. It also allows a more straightforward implement when making complicated designs and features.

As getting sweet but imperfect results, the team both enjoy and become more zealous. To get more information on the Internet, the team wants to learn C++ and MATLAB while learning machine learning to improve the project by detecting hand gestures.

References

The following addresses are accessed in May and June 2020.

Python official documentation, https://docs.python.org/3/

OpenCV documentation, https://opencv-python.readthedocs.io/

OpenCV official documentation, https://docs.opencv.org/master/index.html

Tk documentation, http://tcl.tk/man/tcl8.5/contents.htm

The online version of the book, Do it! Jump to Python, https://wikidocs.net/book/1

Articles about Tkinter, https://076923.github.io/posts/

Articles about OpenCV, https://blog.naver.com/PostList.nhn?blogId=samsjang

Articles about trackers, https://darkpgmr.tistory.com/

Python function explanation, https://kite.com/python/docs/

Python module explanation, http://pythonstudy.xyz/

Ellipse detection algorithm, Xie, Yonghong, and Qiang Ji. “A new efficient ellipse detection method.” Pattern Recognition, 2002. Proceedings. 16th International Conference on. Vol. 2. IEEE, 2002