Self-Aware Standing Desk: Height Adjustment through Gesture Control

Matthew Knox
Student, *Author*
University of Canterbury
mrk45@uclive.ac.nz

Richard Green
Supervisor, *Associate Professor*
Department of Computer Science
University of Canterbury
richard.green@canterbury.ac.nz

Abstract – This paper proposes a method to control the height of a standing desk by using hand gestures. The method proposed uses a combination of background learning and subtraction, skin tone filtering by changing colour space, erode/dilate morphology and edge detection. This will allow for locating, counting and determining the pose of fingers on a hand. The aim is that this method will be compatible with low quality cameras that many end users have attached to their computing devices.

Using default values for each operation, the success rate was 51%. By researching the potential value for each operation it was determined that the ideal kernel size was 20x20 for morphology operations and ideal learning rate was 0.01 for learning the background. By converting the image to HSV colour space, the sensitivity of the algorithm to fingers not being separated increased. This conversion also slightly improved the accuracy of classification.

There was an overall 81% success rate for determining the number of fingers shown to the camera and a 93% success rate for determining hand pose.

I. Introduction

With the rise of the knowledge based society, working while sitting down has been becoming increasingly commonplace. However, research suggests that being seated for extended periods is bad for a person's health. Long periods of sitting has been shown to increase the risk of diabetes, heart disease, and premature y death [1]. The standing desk is one method that has been developed to help address this problem. Standing desks have the ability to adjust height, so that they can used while sitting or standing, with the potential for better health outcomes for the user.

An electronic control mechanism is usually provided to change the height of these desks. This mechanism has a series of switches or buttons. There is a limited number of user interactions that can be made with these controls, however the interactions are not natural. Rather than learning how to use the control system, a better alternative may be to use natural, familiar hand gestures in order to control the height of the desk. This would create an intuitive and easy desk control system for the end user.

The aim of this research paper is to control the height of a standing desk through gesture controls. This includes the use of up, down and stop gestures. The research has potential to expand to include gestures that shift the desk to specific pre-set heights. This would depend on a high success rate at this first stage.

II. Background

A. Dedicated Cameras

There have been various attempts made to implement hand gesture control within the camera sensor. The Leap Motion Controller and the Microsoft Kinect camera have provided significant advances in the field of hand gesture recognition. Gesture recognition is provided by the Software Development Kit and is done entirely in hardware [2]. In one paper it was shown that the Kinect sensor provides a mean accuracy of gesture recognition at a rate as high as 90.6% [3]. However, Kinect depends on the specialised hardware provided by the camera. It would not be possible to utilise the same methods provided by Kinect for gesture recognition in a comparable off the shelf camera. Furthermore, the accuracy statistic only measured a short distance. As the resolution of the Kinect sensor is only 640x480, it would struggle to recognise gestures at a distance without this hardware accurately.

These sensors have other challenges. The Leap Motion controller, according to one piece of research loses accuracy as a hand rotates to be perpendicular to the controller [4]. As the detection is done entirely by hardware, it is not possible to overcome this problem in software. The sensors are also significantly more expensive than webcams, with the Kinect camera as of March 2016 costing NZ$184.95 [5]. This can be contrasted with a low end webcam like the Logitech C170 which has the same resolution as the Kinect camera but only costs NZ$25.30 [6].

B. Detection and Filtering Techniques

There are numerous of detection and filtering techniques used in hand gesture recognition. These are a small selection of techniques that could be integrated into the end method.

1) Background Subtraction

Fig. 1 A hand found using background subtraction (black and white difference image)

Background subtraction involves the subtraction of a frame from a 'background' frame or frames, leaving a difference image behind.

$$\text{Difference Image} = \text{Current Frame} - \text{Background Frame}$$

This image can be processed to determine what has changed in the scene. There are a number of different approaches to performing this task from simple to highly complex, including but not limited to:

- Frame differencing (simple subtraction)
- Running Gaussian average
- Temporal median filters
- Eigen backgrounds [7]

Despite being significantly less complex than some other approaches, methods such as a running Gaussian average "offer acceptable accuracy" at detecting movement in a scene [8]. Detailed testing of the performance of all methods is not possible due to the challenge in implementation. The accuracy is also heavily dependent on what is classified and how the tests are run. The accuracy mentioned earlier for example did not take into account anything other than twenty images taken in a lab scenario. Real world performance would require further testing.

2) Skin Colour Filtering

Skin colour filtering is another method of background removal/noise reduction used in the process of recognizing hand gestures. This involves selecting a colour range in a colour space within which human skin colour lies. All other colours are filtered from the image leaving the skin tones behind.

Skin sensitive colour spaces such as HSV (Hue, Saturation, Value) are often used to help separate skin tones from other colours in scenes [9]. One paper which researched the possibility of detecting a face using skin tones estimates that the accuracy of detecting skin within in the HSV colour space is between 80% and 82% [10].

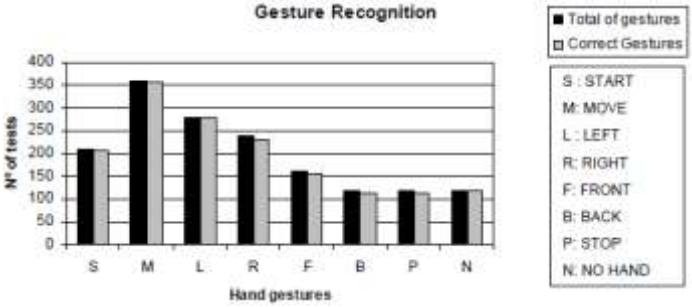

Fig. 2 Gesture classification accuracy after skin tone filtering

As can be seen in Fig. 2, using filtering of this type to remove the background can lead to high accuracy in recognition [11], as only areas of skin remain in the image. There are however limitations in this approach. The research did not take into account a variety of skin colours and had challenges detecting only one. The research also did not do any testing outside of a laboratory environment.

3) Hand Detection

A variety of methods for classifying an object in a scene as a human hand have been developed. The most common techniques include:

- Hand Shape
- Texture/Tone (related to skin colour filtering)
- Depth Information
- Markers

The shape of the human hand can be classified using its edges. Once edges are detected, a convex hull can be constructed which indicates all of the extremities of a hand [12]. Points on the convex hull can be combined to generate an artificial "centre of gravity", which can in turn be used to determine the pose of the hand [13]. This is the approach taken within the Intel Realsense SDK [14]. The accuracy level of this "centre of gravity" is not stated. Average accuracy was around 85% for the images that were tested with no skew hand gestures in the relatively constant lighting conditions.

Depth information can be combined with both hand shape and texture/tone detection to significantly improve the accuracy of the marker-less detection provided both individually [12]. This is only theoretical. It was not directly tested in the cited paper.

Markers are used commonly to suggest locations on objects and their directions. They are designed to be easy to filter in a received image [15]. Limitations of using markers are that the person needs to place the markers in appropriate positions before using the system. Additionally, depending on the type of marker, special hardware may be required to pick up the information they convey.

III. Proposed Method

The method proposed is to detect the fingertips of a hand, in order to count how many fingers are being shown to the camera and what pose they are in. Using this method should be possible without the need for any reference artefacts (such as markers or

a glove) as is needed in some prior research. This research utilises a low end commercial camera. This is different to the specialised and high end cameras such as Microsoft Kinect and Leap Motion. A low end camera has been chosen so that there is a solution provided that does not depend on a type of specialised or expensive camera. Consequently, the solution will be relatively affordable to the end user and inexpensive for a manufacturer to incorporate into a standing desk. There is also the advantage that new hardware will not be needed by the majority of potential users, as most computers have an existing web camera.

Rather than using individual techniques to achieve each gesture recognition – the common approach in most of the background research – this paper will combine a range of techniques. The advantages of each approach will be used to combat the weaknesses in others.

To detect the fingertips and control the desk, the following elements will be combined into a processing pipeline:

A. Background Removal and Noise Reduction

Background removal will be used to attempt to reduce the number of background objects in the scene. This will help isolate the hand from any other item in the scene and make hand gesture detection more achievable. This will be primarily done through converting to a colour space where a hand is significantly more visible (skin colour filtering, HSV colour space) and learning the background for subtraction. To learn the background a gaussian average/mixture function will be used:

$$\mu_t = M\mu_{t-1} + (1 - M)(I_t\rho + (1 - \rho)\mu_{t-1})$$

Where μ_t describes the mean of each pixel at time t in terms of I_t, the pixels intensity, M wheather the pixel is foreground or background and ρ the rate which the change will be learned.

This will maintain a model of the background and update it over time as it changes based on a learning rate. As a result, the average background over a period of time will be subtracted, rather than the background at a particular moment in time (which may not reflect the true background due to artefacts in the frame that do not usually exist within the image). This will also give it the ability to adjust changes in lighting and movement of the camera.

Noise in the image will be reduced through gaussian blurring. The following erosion morphology function will be used to overcome gaps in edge detection due to grainy very low resolution images.

$$dst(x,y) = \min_{(x',y'):\, element\ (x',y') \neq 0} src(x + x', y + y')$$

Where dst is the destination matrix and src is the source matrix.

Additionally, dilation will be performed with the same goal using the function:

$$dst(x,y) = \max_{(x',y'):\, element\ (x',y') \neq 0} src(x + x', y + y')$$

B. Edge/Contour Detection

Using an edge detector, the outline edges of the hand in the scene will be found and marked. To do this a Canny edge detection algorithm will be used. The Canny edge detector will first run a Gaussian filter over the image to reduce noise. Thresholding of gradients will then be used to find and mark edges.

From the edges/contours detected the largest enclosed area will be taken and assumed to be the hand. If the hand does not meet an experimental threshold area, then it will be discarded as noise.

C. Convex Hull Hand Detection

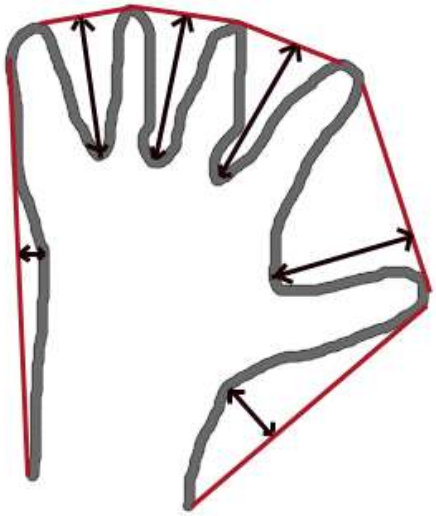

Fig. 3 Convexity Defects of a Hand Contour [16]

Convexity defects will be used to attempt to detect the number of fingers present on a hand. As can be seen in Fig. 3, when the convex hull of a hand is created. This is done using Sklansky's convex hull algorithm [17] which finds a convex hull by finding extreme vertices using monotone chains. The convex hull will have a series of defects indicated by the arrows. By locating these defects, it will be possible to determine the locations of each finger. By allocating an upper and lower threshold for the depth of the defects, the number of convexity defects will correspond to the number of fingers present in the image. Using clustering of points on the hull and the angle between them and an axis (relative to the centre of the convex hull) the direction the hand is facing will be determined.

D. Hand Pose Detection

Using the points from the convex hull, the centre of gravity method from the background research will be used to calculate hand pose. Once the centre of gravity has been calculated, an average angle between the horizontal and the fingers, using the centre of gravity as the origin will be used to estimate pose.

E. Desk Control

The number of fingers that are raised on the hand will be counted and from there an appropriate action will be determined. Changes to the desk height will be done through calling a RESTful web service that will be created for the standing desk.

IV. RESULTS

The results were achieved using the following hardware (not all hardware was used at all stages):

Table 1 Hardware used during research. Although a large number of computers were used, this is only because of numerous hardware failures that occurred during research – only one computer is needed to run the control code.

Item	Specifications
Camera	Logitech C170 (640x480, 30FPS)
Devices	Raspberry Pi (v2 B) • OS: Raspbian Jessie • Processor: ARM Cortex-A7 • Speed: 900MHz • GPIO: SainSmart 8-Channel Relay Apple MacBook Air Mid 2012 • OS: OSX El Capitan • Processor: Intel Core i5 2467M • Speed: 2.6GHz Dell XPS 15 9550 • OS: Windows 10 • Processor: Intel Core i7 6700HQ • Speed: 3.5GHz Dell XPS 15 L502x • OS: Windows 10 • Processor: Intel Core i7 2630QM • Speed: 3.0GHz Custom Build Desktop • OS: Windows 10 • Processor: Intel Core i7 4790K • Speed: 4.4GHz
IDE	Microsoft Visual Studio 2015 (Update 2)
Languages	C++15, Javascript (Node.js)
Libraries	OpenCV 3.1, Casablanca HTTP SDK, rpi-gpio, express.js 3.0
Desk	Cubit Highrise 1800 x 800 Standing Desk

A. Desk Control

The desk was modified for computer control [18]. This was done using a combination of a Raspberry Pi and a GPIO controlled relay switch which was wired into the internal manual switches of the desk. Desk control functionality was exposed using a Node.js based HTTP server with RESTful API's. The majority of standing desks do not expose these controls using an API, so even without the gesture control component added the possibilities opened up by providing this capability are large.

Trial and error determined that it takes approximately 20 seconds for the desk to reach its highest point from its lowest point. Conversely, it takes 14 seconds to reach its lowest point from its highest. The desk itself does not contain hardware to prevent the motor from going beyond the maximum/minimum limits, however the RESTful service has been designed to limit moving the desk up and down so the motor cannot be burnt out.

B. Gesture Recognition

To test the success rate of the classifier, first the camera was setup pointing at a largely blank surface. The algorithm was given five minutes to learn the background. After this, the classifier was started and for each gesture the had was placed at a distance of one meter away from the camera. Over the course of 30 seconds the hand was moved to a distance of 20cm away from the camera. Each classification the classifier made during this time was recorded and checked for validity.

Table 2 Initial results from Hand Classification Testing. In these tests, all values are default and there is no skin tone filtering.

Test	Successful Tests	Failed Tests
No hand in frame	300	1
No fingers raised	52	236
1 Finger	79	195
2 Fingers	72	256
3 Fingers	239	17
4 Fingers	169	101
5 Fingers	102	165

The initial results, using default values and no skin tone filtering proved to be mixed in terms of success (see table above). On average there is a 51% success rate when classifying the number of fingers raised on a hand.

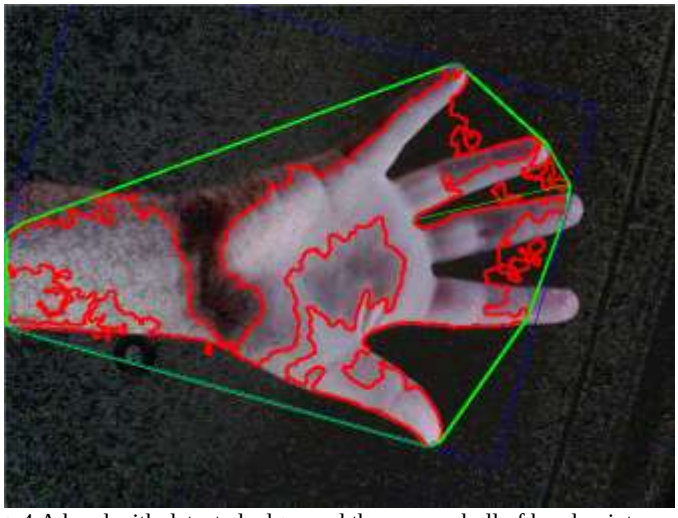

Fig. 4 A hand with detected edges and the convex hull of hand points shown.

As the image above shows, this appears to be because the edges of the image are not being correctly detected. The low light conditions that the tests were occurring in and the non-contrasting background were having a significant effect on the Canny edge detector.

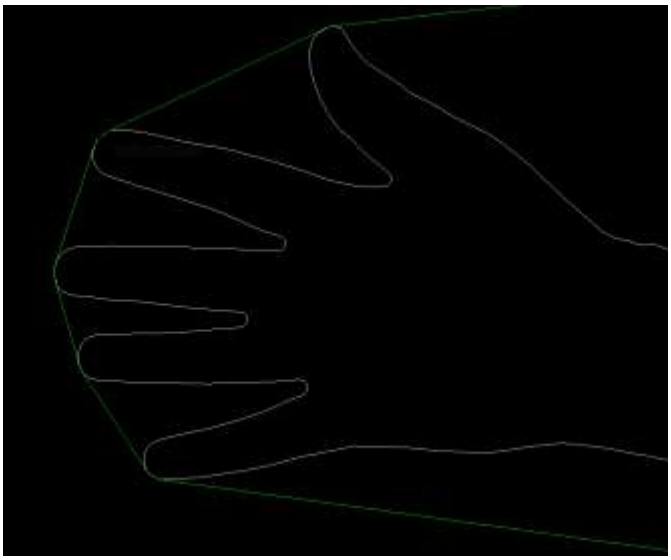

Fig. 5 Correctly detected contours and hand

Repeating the same test in better conditions resulted in the image above which is what was originally intended. However, results in this research are from the original conditions to amplify any potential flaws in the design of the algorithm.

Other factors also appeared to be affecting classification. Some contours such as that of the end of the index finger in Fig. 4 were being detected but were not being joined to the main contour of the hand. A brief experiment with a laptop webcamera in the same conditions revealed that this seemed to be caused by the graininess of the image. Altering the kernel size used with the morphology operators (erode and dilate) had different effects on this problem.

Using the 51% success rate from the initial test as a base line, tweaking the kernel size of erode and dilate gave the following results:

Table 3 Morphology Kernel Size Optimisation

Kernel Size (px)	Success Rate (%)	Improvement (%)
3x3 (Original)	51	0
5x5	49	-2
7x7	54	3
10x10	60	9
15x15	69	19
25x25	74	24
40x40	70	20

From this information, it can be seen that using a kernel size of 25x25 can improve overall classification by up to 24% on the initial 51%.

Fig. 6 A picture of a hand in HSV colour space. The hand is significantly brighter than its surroundings and appears to have a greenish glow surrounding it.

Background research suggested that the doing background subtractions in a skin tone sensitive colour space can also provide improvements to the algorithm. By first converting the background into HSV colour space, skin tone will be highlighted relative to the dark testing background.

Table 4 Success rate after converting to HSV colour space relative to the first, initial test

Test	Success Rate in HSV (%)	Improvement over original (%)
No hand in frame	99	-1
No fingers raised	22	2
1 Finger	30	2
2 Fingers	34	12
3 Fingers	86	-7
4 Fingers	75	13
5 Fingers	61	23

As can be seen from the table above the results from the research backed up. On average when identifying each number of fingers, using a skin tone sensitive colour space made an overall improvement. It is possible that this could be improved further if filtering actually took place rather than just increasing the sensitivity of the algorithm to the skin tone colours. As the success rate of classifying three fingers is already high, a decrease in accuracy by 7% relative to the increase in accuracy for almost all other finger combinations is a net positive gain.

Using this colour space also gave the camera enough sensitivity to still separate between fingers when they were not held apart.

Another factor that can be improved in the algorithm is the learning rate of the background subtractor. The learning rate as discussed in the proposed method, is how fast the learned background will adapt to changes in the scene.

Table 5 Success rate of the algorithm after changing the learning rate of the background subtractor over the original value

Learning Rate	Success Rate (%)	Improvement over original (%)
0.010	56	5
0.025	54	3
0.050	54	3
0.075	53	2
0.100 (Original)	51	0
0.250	46	-7
0.500	40	-13

Table 5 shows that overall lower learning rates are better for the accuracy of the classifier. This is probably because higher learning rates have a higher likelihood of producing ghosting – where the hand itself is partially learned to be part of the background image. By lowering the learning rate only changes that are constant (or frequent occurrences) will be learned. However, this means that changes in lighting conditions or movement of the camera will not be compensated for very quickly

Table 6 Classification success rate after implementing all optimal values for algorithms

Test	Successful Tests	Failed Tests
No hand in frame	250	0
No fingers raised	93	14
1 Finger	79	80
2 Fingers	72	68
3 Fingers	239	42
4 Fingers	199	47
5 Fingers	206	21

With the improvements stated implemented, Table 6 shows that overall the success rate has improved to 81%. As can be seen, lower numbers of fingers have a lower success rate, particularly one and two fingers. This is probably because generating a useful convex hull on this number of fingers is difficult – the fingers do not have as larger effect on its shape as they usually do in larger numbers.

C. Hand Pose

Hand pose was implemented using the centre of gravity method from the background research. The algorithm differs in that it only determines if the hand is in an ‘up’ or ‘down’ orientation as this is all that is needed to control the desk.

Testing this approach was done using a series of twenty images passed through the algorithm. Each of these images were tested in six different orientations. Success was determined by a correct up/down pose classification.

Table 7 Success rate of determining hand pose using the "centre of gravity" method. Angles are measured relative to the horizontal.

Test Angle (approximate)	Successful Tests	Failed Tests
45°	18	2
90° (up)	20	0
135°	19	1
225°	19	1
270° (down)	20	0
315°	16	4

Table 7 shows the outcome from these tests. Overall, pose estimation was 93% successful with angles closer to the horizontal having lower success rates. The angles of 0° and 180° were not tested, as these cannot be correctly classified without extending the possible outcomes to include ‘left’ and ‘right’.

V. Limitations

The tests performed were done in controlled conditions, with a controlled background and lighting. If the camera was actually mounted to the desk none of these factors would be controlled so it is not possible to determine the actual success rate in a real world scenario. Additionally, the camera was stationary during these tests. If the camera was mounted on the standing desk it would move when the desk does. This would cause problems for the current implementation of the background subtraction algorithm which assumes a stationary background.

Tests were only done using one camera, other cameras may present their own unique problems and scenarios. The camera used was quite low resolution so processing time was not a factor. Higher resolution cameras would greatly increase the processing time.

The distances specified for testing will not be exact. Certain distances will be more accurate than others, so the results gathered will only be indicative of the average over the range rather than the optimal distance away.

VI. Conclusion

Overall, the proposed method was successful. 81% of the time, it would generate a correct classification for the number of fingers and 93% of the time the hand pose was correct. This would cause the desk height to be adjusted accordingly. However, the research dataset has a large number of limitations that mean that the performance of the method in the real world remains to be seen. The success rate is fairly close to that seen on the Kinect sensor which has dedicated hardware for solving the problem.

Compared to similar research, the hardware used was low quality comparatively very cheap. This puts this method of gesture within the reach of the average person as almost all computers/phones have a camera of greater or higher quality than the one used. It is also non-invasive, as specialised

markers do not need to be used. By combining techniques, the method was effective even in non-optimal conditions.

Utilising hand gestures with standing desks will give users a more natural and familiar interaction with their working environment without needing to learn control interfaces. This paper proves that such a control system is achievable with the hardware users already have at hand or can easily acquire without adding large cost to an already expensive desk.

VII. Future Research

Future research should focus on removing inaccuracies in classifying lower numbers of fingers. This could be done through implementing full skin tone filtering rather than just using a different colour space and performing research into the optimal thresholds to use while doing gaussian edge detection.

VIII. References

[1] D. W. Dunstan, B. Howard, G. N. Healy and N. Owen, "Too much sitting – A health hazard," American Journal of Preventive Medicine, vol. 97, no. 3, pp. 368-376, 2012.

[2] F. Weichert, D. Bachmann, B. Rudak and D. Fisseler, "Analysis of the Accuracy and Robustness of the Leap Motion Controller," Sensors, vol. 13, no. 5, 2013.

[3] Z. Ren, J. Meng, J. Yuan and Z. Zhang, "Robust Hand Gesture Recognition with Kinect Sensor," Proceedings of the 19th ACM international conference on Multimedia, pp. 759-760, 2011.

[4] L. E. Potter, J. Araullo and L. Carter, "The Leap Motion controller: a view on sign language," Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration, pp. 175-178, 2013.

[5] Microsoft, "Kinect for Xbox One," 24 March 2016. [Online]. Available: http://www.microsoftstore.com/store/msnz/en_NZ/pdp/Kinect-for-Xbox-One/productID.307395100. [Accessed 24 March 2016].

[6] PriceSpy, "Logitech Webcam C170," PriceSpy, 1 April 2016. [Online]. Available: http://pricespy.co.nz/product.php?p=896583. [Accessed 1 April 2016].

[7] Z. Xu, P. Shi and I. Y.-H. Gu, "An Eigenbackground Subtraction Method Using Recursive Error Compensation," Advances in Multimedia Information Processing, vol. 4261, pp. 779-787, 2006.

[8] M. Piccardi, "Background subtraction techniques: a review," in IEEE International Conference on Systems, Man and Cybernetics, 2004.

[9] V. Oliveria, "Skin detection in HSV colour space," 2009. [Online]. Available: http://www.matmidia.mat.puc-rio.br/sibgrapi2009/media/posters/59928.pdf. [Accessed 29 April 2016].

[10] V. Bhat and J. Pujari, "Face detection system using HSV color model and morphing operations," September 2013. [Online]. Available: http://inpressco.com/wp-content/uploads/2013/09/Paper39200-204.pdf. [Accessed 29 April 2016].

[11] C. Manresa, J. Varona, R. Mas and F. J. Perales, "Hand Tracking and Gesture Recognition for Human-Computer Interaction," Electronic Letters on Computer Vision and Image Analysis, 2005.

[12] S. Qin, X. Zhu, Y. Yang and Y. Jiang, "Real-time Hand Gesture Recognition from Depth Images Using Convex Shape Decomposition Method," Journal of Signal Processing Systems, vol. 74, no. 1, pp. 47-58, 2013.

[13] T.-H. Tsai, C.-C. Huang and K.-L. Zhang, "Embedded virtual mouse system by using hand gesture recognition," Consumer Electronics - Taiwan (ICCE-TW), pp. 352-353, 6 August 2015.

[14] Intel, "Hand Tracking Tutorial from IDF 2015 Intel RealSense Lab," 21 8 2015. [Online]. Available: https://software.intel.com/sites/default/files/managed/7b/89/SFTL006_100.docx. [Accessed 1 4 2016].

[15] E. Murphy-Chutorian and M. M. Trivedi, "Head Pose Estimation in Computer Vision: A Survey," Pattern Analysis and Machine Intelligence, pp. 607-626, 2009.

[16] OpenCV Project, "Structural Analysis and Shape Descriptors," OpenCV 2.4.13.0 documentation, 01 April 2016. [Online]. Available: http://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html#convexitydefects. [Accessed 29 April 2016].

[17] G. Aloupis, "Sklansky 1982," [Online]. Available: http://cgm.cs.mcgill.ca/~athens/cs601/. [Accessed 29 April 2016].

[18] M. Knox, "Dions Desk," Dions Desk, 1 April 2016. [Online]. Available: http://desk.makereti.co.nz. [Accessed 7 April 2016].