Gesture Recognition by Machine Learning Combined with Geometric Calculation

Hiroshi Yokoyama[1], Paul Schmalenberg[2], Muhamed Farooq[2], and Ercan M. Dede[2]

Abstract— **Human Machine Interface (HMI) in the vehicle environment is becoming more complex due to an increased number of desired user functions. These added functions increase the driver's distraction since the driver must visually identify a particular physical interface that they want to control. A gesture recognition system is one possible method for controlling HMI functionalities without visual cues. Some technical approaches to gesture recognition that exploit deep learning have been proposed. However, these methods require increased computational cost in relation to processing time. Ideally, a gesture recognition system should enable fast and accurate driver action to avoid a distracted driver and maximize safety. In this study, we propose an algorithm that combines geometric calculation and machine learning techniques to estimate gestures with low computational overhead. The algorithm is explained in detail, and results are reported including a gesture recognition accuracy of 85% across three gestures for a moving hand from eight people.**

I. INTRODUCTION

Modern vehicles have numerous human machine interface (HMI) systems to control functionality including navigation, information display, climate control, audio, and advanced driver assistance systems. While these systems are expected to make drivers more comfortable, they might cause a driver to be distracted and increase cognitive workload since a driver must identify a particular physical interface to control. In this case, these systems might not be able to maximize their safety performance [1], [2]. A touch-less gesture recognition system is one possible method to reduce the driver's workload when controlling these systems [3]–[7].

Hand gestures can be estimated by using deep learning, especially a Convolutional Neural Network (CNN) architecture with images captured by a red green blue (RGB) or an RGB-depth camera [8]–[11]. With this method, large amounts of image data is collected to learn the types of gestures and to classify the gestures accurately using a large CNN. Thus, this deep learning-based method requires computational resources for learning datasets and classifying a gesture in real-time. Furthermore, it is difficult to extract the hand from other parts of an image using an RGB camera when the skin color is similar to the background scene. Since a gesture recognition system should work fast and accurately for vehicle applications, we therefore investigate other methods. Geometric calculation-based gesture recognition using a depth camera is another method to estimate the hand state with low computational cost [12]–[14]. Additionally, using the image depth data, the hand can be easily extracted from the background regardless of the color of background scene [15]–[19]. To estimate a gesture, machine learning is eventually employed to understand the type of gesture after pre-processing [20]–[22]. An estimated hand is extracted by thresholding the depth [23]. In other methods, a fixed-size square window localizes the hand area [21], [24]. For in-vehicle gesture recognition, the driver's hand is not always located in the central area of the sensor view because precise positioning of the hand represents an unacceptable increase of workload for the driver. However, in previous studies, accuracy within the whole field-of-view of a sensor has not been discussed.

Here, hand gestures captured by a depth sensor are classified with geometric calculation and machine learning. First, the palm center and fingertips are estimated from the image depth data by geometric calculation, Sec. II. Then, some features of the gesture are obtained from relationships between the fingertip positions and the palm center. A gesture is finally classified by machine learning. Considering eventual vehicle use, gesture data of hands in motion is further collected and analyzed. Experiments and data analysis are explained in Sec. III. The results are summarized in Sec. IV. Discussion and conclusions are provided, respectively, in Secs. V and VI.

II. GEOMETRIC CALCULATION

The image processing procedure is shown in Fig. 1. In this study, the depth image is captured using a Time-of-Flight (ToF) camera (EVK75024, Melexis [25]), and the image, Fig. 2 (a), is used to estimate a gesture. This sensor utilizes different time-delays of the illumination pulse over several pulses and calculates the distance by the ratio of light arriving in a time-gated pixel of fixed length over the different illumination pulses. After the depth image is captured, image processing

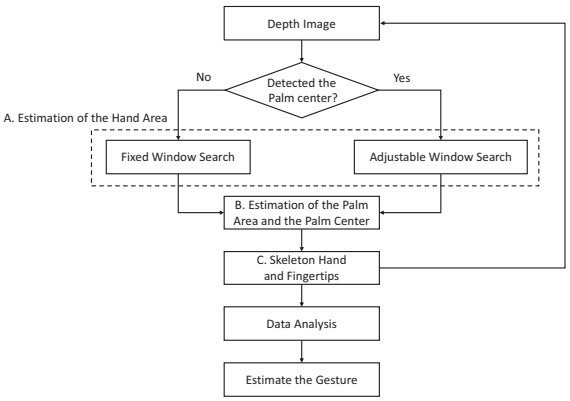
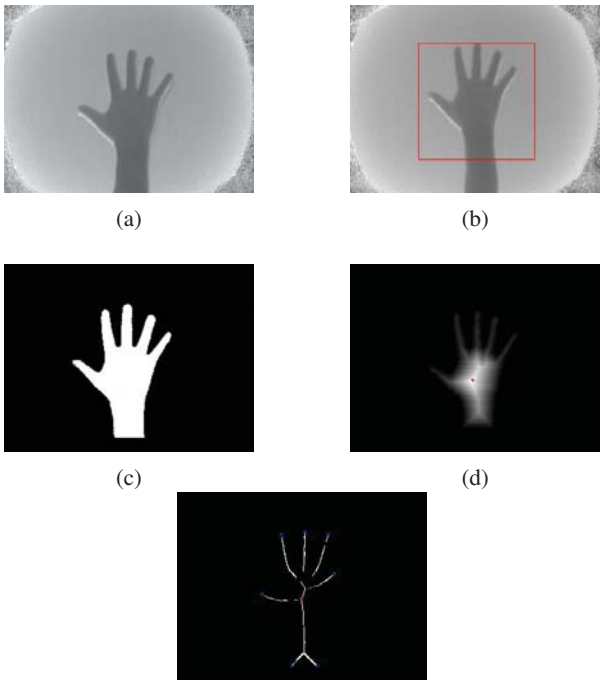


Fig. 1: Image processing work flow

[1]Electronics Control System Development Division, Toyota Motor Corporation, Toyota City, Aichi, Japan, Email: hiroshi.yokoyama@toyota.com

[2]Electronics Research Department, Toyota Research Institute of North America, Ann Arbor, MI, USA, Email: paul.schmalenberg, muhamed.farooq, eric.dede@toyota.com



(a) (b) (c) (d) (e)

Fig. 2: Images corresponding to the image processing work flow: (a) raw camera image; (b) image with calculated window; (c) 0-1 binary image of the hand; (d) palm center identification; (e) hand skeleton estimation

to obtain features for the estimation of a gesture is conducted by geometric calculation to approximate the palm center and eventually the fingertip positions. An explanation of the method follows.

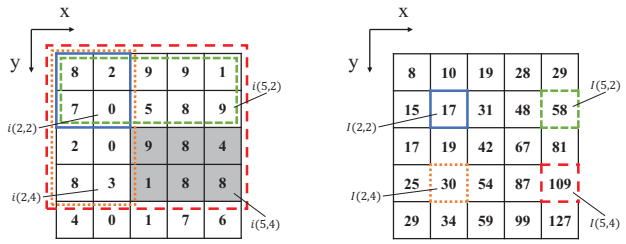
A. Estimation of the Hand Area

The average depth, D, in a specific area of the image is calculated by using a search window within the entire depth image. Here, the integral image method from OpenCV [26] is implemented to quickly calculate the average value within a search window. The sum, I, of all pixels from $(0,0)$ to (x,y) is calculated, and then this value is defined as $I(x,y)$ in (1) , where $i(x,y)$ is a value of the pixel at (x,y) of the depth image. The average depth within a search window of size, w, is calculated from the integral image based on I as shown in (2). An example of an integral image is provided in Fig. 3.

$$I(x,y) = \sum_{x'=0}^{x} \sum_{y'=0}^{y} i(x',y') \tag{1}$$

$$\begin{aligned} D(x,y) =& (I(x,y) - I(x-w,y) - I(x,y-w) \\ &+ I(x-w,y-w))/w^2 \end{aligned} \tag{2}$$

The window has a fixed size and searches within the integral image by shifting pixel by pixel according to (2). Then, the window which has the largest value is defined as the hand area, as shown in Fig. 2(b). Previous methods search within the whole image to find the hand area for every frame. Here, after



(a) (b)

Fig. 3: An example of an integral image with 5×5 pixels to calculate the sum from $(3,3)$ to $(5,4)$: (a) is a raw image; (b) is an integral image. The result of $I(5,4) - I(2,4) - I(5,2) + I(2,2)$ shows the sum from $(3,3)$ to $(5,4)$.

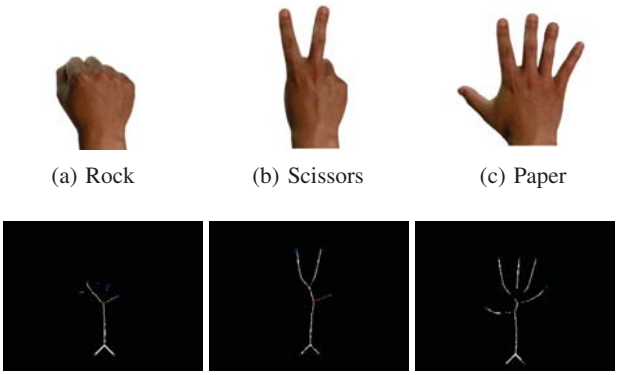
hand area estimation in the first frame with hand detection, the size of the search window is changed in the next frame based on the estimated palm size which is calculated based on depth in Sec. II-B. This new window size will be used at the same location as the previous frame to estimate the palm center in the current frame. The window size corresponding to the hand size is defined as two times the length of the palm size based on NASA-STD-3000 [27].

B. Estimation of the Palm Area and the Palm Center

The palm area is defined based on all pixels closer to the camera than the average depth, D. The palm area and background scene are separated with this threshold value. The pixel values are replaced with binary values, with 1 representing palm presence and 0 representing empty space; see Fig. 2(c). After obtaining the binary image for the hand, a distance transform is performed with OpenCV to estimate the palm center location [28]. Each pixel that was identified previously as the palm area now has a value, P, associated with it. Here, P is larger in magnitude the closer the pixel is to the estimated palm center. The image that is obtained by this process is shown in Fig. 2(d). The palm center is then defined as the point which has the largest value of P on the distance transform image. In this study, once the palm center is detected, the center of the search window is updated frame by frame to be coincident with the palm center.

C. Estimation of the Hand Skeleton and Fingertips

A window of 3×3 pixel size is used to search for local peaks on the distance transform image to obtain the hand skeleton image. If no more than three pixels are larger than the central pixel in this window, and the central pixel value is larger than a tenth of the largest P value in the whole image, the central pixel is defined as a part of the hand skeleton. After searching over the whole distance transform image, the hand skeleton image is obtained, as shown in Fig. 2(e). Finally, the smallest convex envelope that contains all of the hand skeleton points is calculated to obtain the fingertips. Vertices of this convex set are defined as the fingertips, and the image naturally ends up including a portion of the arm at the edge of the image. The convex hull OpenCV library is exploited to calculate the convex set and fingertip positions [29].



(a) Rock (b) Scissors (c) Paper

(d) (e) (f)

Fig. 4: Sample images for three gestures: from (a) to (c), "rock", "scissors," and "paper," respectively, and corresponding skeleton images of the three gestures: from (d) to (f), respectively,"rock," "scissors," and "paper"

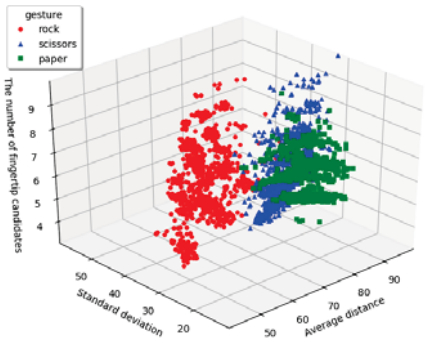


Fig. 5: A 3-D feature space comprising the average distances, standard deviation of distances, and number of estimated fingertip candidates

III. EXPERIMENT AND DATA ANALYSIS

To confirm that our method efficiently works for gesture recognition, gesture data collection was conducted as part of the experiments. Three type of gestures including "rock," "scissors," and "paper" are classified. Figures 4(a)-(c) show examples of the three hand gestures, and Figs. 4(d)-(f) show the corresponding estimated images of the palm center, hand skeleton, and fingertips obtained though geometric calculation. The result of the geometric calculation process is analyzed to retrieve pertinent features of each hand gesture. There are six features including the x, y, and depth positions of the palm center, the average of the distances between the palm center and fingertips, the standard deviation of these distances, and the number of estimated fingertips. The entire geometric calculation, data analysis, and classification procedure is coded with Python 3.6 and executed on a computer containing an Intel Core i9-9880H. A description of the data collection and analysis process follows.

A. Data Collection

Stationary gesture data from one participant with both hands was collected, and gesture data with a moving hand was obtained from eight participants. For the moving hand gestures, data was obtained from each participant using both left and right hands in separate experiments. Note that although there are a lot of open source datasets for hand gestures, it is difficult to find available gesture data with a moving hand. The ToF camera has a 110 $\times$ 90 degree field-of-view and captures depth images each with 320 $\times$ 240 pixels. For each gesture, 500 frames were captured, which means the total number of frames for the stationary hand data was 3,000 versus 24,000 for the moving hand data (3 gesture $\times$ 2 hands $\times$ 500 frames $\times$ 8 participants). The pictures in Figs. 4(a)-(c) were shown to the study participants to illustrate each gesture. In the moving hand experiments, participants can make their gestures wherever within the field of view of the ToF camera, and they can move their hands in x, y, and depth directions while keeping their gesture fixed. Fig. 5 demonstrates three representative features in a three-dimensional (3-D) feature space using the average distance, standard deviation, and number of fingertip candidate features. From Fig. 5 we observe that utilizing the three aforementioned features is insufficient to provide optimal class separation. How all six features are created and how they are utilized for gesture classification is described below. After feature extraction, Support Vector Machine (SVM) is employed with all six features to identify the three different gestures classes.

B. The Position and the Depth of the Palm Center

In a real driving environment, it is difficult to guide the driver's hand into a specific position to perform a gesture. Thus, the system should detect the hands organically wherever they happen to be located. Therefore, in this study, participants are allowed to move their hands freely. The position and the depth of the palm center are used as features to estimate a gesture regardless of the hand location.

C. Average of Distances

The distances between the palm center and the fingertips can be different depending on the type of gesture since some fingers may be bent. Therefore, these distances can be features used to classify small and large gestures such as "rock" and "paper," respectively. Accordingly, the average of these distances calculated for each frame is used since the number of estimated fingertips is different among frames.

D. Standard Deviation of the Distances

The distances between the palm center and each fingertip can be similar when, for example, a participant makes a "rock" or "paper" gesture. In this case, the variation of these distances for these gestures might be similar. In contrast, the variation of distances for a "scissors" gesture might be larger in comparison to the other gestures. Thus, the standard deviation of the distances between the palm center and fingertips distances is calculated to assist in classifying various gestures.

E. The Number of Fingertip Candidates

The number of candidates for estimated fingertip points found through geometric calculation may be different depending on the gesture. In the case of the "paper" gesture, the number of estimated fingertips usually agrees with the number

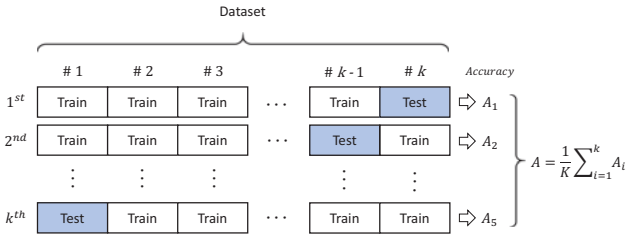
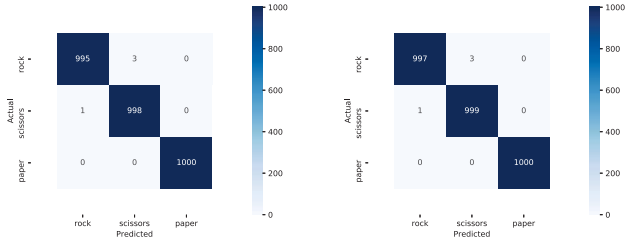


Fig. 6: K-fold Cross Validation



(a) (b)

Fig. 7: Confusion matrix of the results from stationary gestures: (a) results obtained using a fixed window method; (b) results obtained using our proposed method

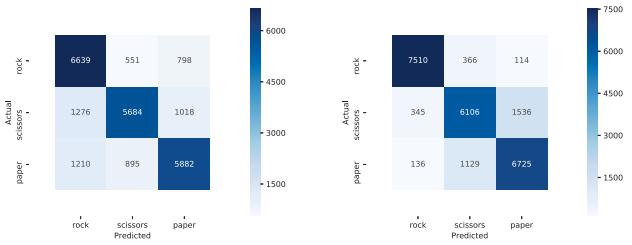
of actual fingertips. However, it is difficult to estimate the number of fingertips for gestures such as "rock" and "scissors" since their convex set can have more vertices than reality depending on the convex set condition. Appropriately, the number of fingertip candidates may also be exploited as a feature to classify the different gestures.

F. Estimation of Gestures

After collecting the features to classify the gestures through feature extraction, SVM is employed to estimate the gesture from these features. We use Scikit – Learn as SVM library [30]. The K-fold cross-validation shown in Fig. 6 is used to evaluate the accuracy of SVM estimation and avoid overfitting. In the K-fold cross-validation, the dataset is randomly divided into K subsets. Then, $K-1$ subsets are used for model training, and the remaining subset is retained for model testing. This process is repeated K iterations until each subset is utilized once for training and once for testing. The accuracy is calculated by taking the average of all iterations. In this study, 5-fold cross-validation is conducted $(K=5)$. The SVM parameters were set to kernel = "rbf," $C=1$, and $\gamma=1$.The C parameter represents the tradeoff between the correct classification of training samples and the maximization of the decision function's margin (i.e. large C values signify smaller margin and vice versa). On the other hand the gamma parameter represents the span of the training samples' influence with large values signifying close influence reach and low values representing far influence reach.

IV. RESULTS

First, the stationary gestures were estimated with a fixed window method and our proposed method. We confirmed that both methods can classify with 99% accuracy, as shown in

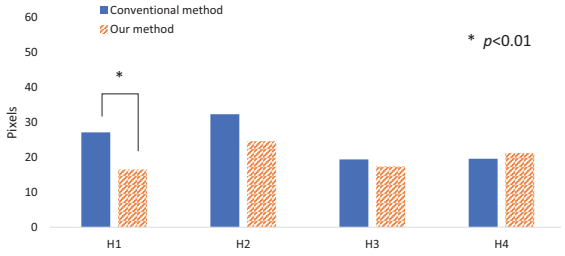


(a) (b)

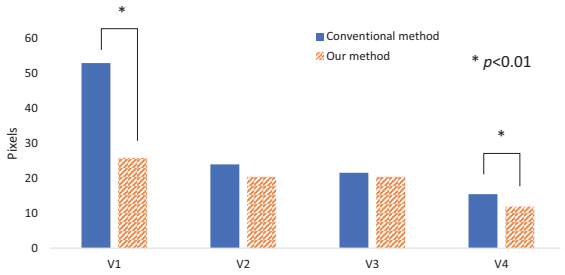
Fig. 8: Confusion matrix of the results from the gestures with a moving hand: (a) results obtained using a fixed window method; (b) results obtained using our proposed method

Fig. 7. The size of the collected data was 3,000 frames, with 1,000 frames collected for each gesture. Three frames were not properly classified using the fixed window method (i.e. two frames with "rock" and one frame with "scissors"). For the stationary gesture, there is almost no difference in the results obtained using the two methods. However, for the gestures with moving hands, the accuracy with the fixed window method was 76%, while the accuracy with our method was 85%, as shown in Fig. 8. In this case, 47 frames were not properly classified with the fixed window method (i.e. 12 frames with "rock," 22 frames with "scissors," and 13 frames with "paper"), and 33 frames were not classified with our method (10 frames with "rock," 13 frames with "scissors," and 10 frames with "paper"). With the fixed window method, the accuracy for "rock", "scissors," and "paper" was, respectively, 83%, 71%, and 74%. In our method, the accuracy was 94% for "rock" , 76% for "scissors, and 84% for "paper." These results indicate that our algorithm improves the estimation for all three gestures considering a moving hand. The processing speed with our method, 8.4 fps, is also faster than the speed for the fixed window method, 6.5fps.

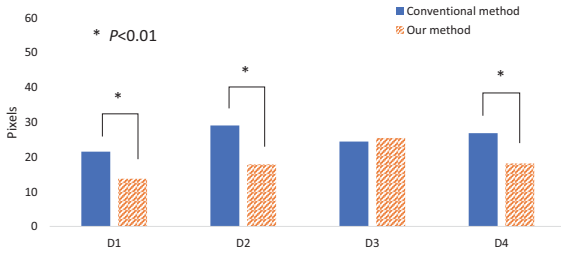
A follow up experiment was performed to further confirm the difference between the fixed window method and our method. Specifically, a "paper" gesture made with the right hand was collected to confirm how accurate both methods are in estimating the palm center. The actual palm center position was annotated manually for 500 frames to compare with the palm center estimated using both methods. Error values were calculated as the number of pixels between the annotated value and the calculated value in four areas for horizontal (H1-H4), vertical (V1-V4) and depth (D1-D4) directions, as shown as Fig. 9. The error values shown were calculated as the average value in each area. For the horizontal direction, there is a significant difference in the error value between the fixed window method and our method in the H1 area. While the fixed window method estimates part of arm as the palm center in this area, our method estimates the hand area and the palm center correctly; refer to Fig. 10. In the V1 area, the fixed window method also estimates a part of arm as the palm center, as shown in Fig. 11. The source of the error in the D1 and D2 areas is also the same. In the V4 and D4 areas, the window size for the fixed window method is too small to



(a)



(b)



(c)

Fig. 9: Errors between the actual palm center position and the estimated position over four areas for each direction: (a) horizontal, (b) vertical; (c) depth

estimate the hand area, and as a result, the method can not estimate the palm center correctly; see Fig. 12.

V. DISCUSSION

Three gestures were estimated from six features using SVM. While the accuracy of the two studied methods is almost the same for stationary gestures, the accuracy for moving gestures is higher using our proposed method. The new method uses an adjustable window corresponding to the estimated palm center position and its size. An additional experiment was conducted to confirm the accuracy of the estimation of the palm center. There are significant differences in the error values of the palm center obtained using the fixed window method and our method. Specifically, part of arm can affect the results obtained using the fixed window method. When the arm occupies a large area in the image frame, this part of the body can have the largest value when using a fixed window. Another reason for these differences is that the window size for the fixed window method is too small related to the palm size when the hand is located far from the ToF camera. In this case, the fixed window method can again estimate a part of arm as the palm.

(a)

(b) (c)

Fig. 10: Hand estimation in the H1 area: (a) raw depth image; (b) estimated image with the fixed window method; (c) estimated image with our proposed method

(a)

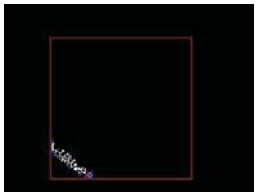
(b) (c)

Fig. 11: Hand estimation in the V1 area: (a) raw depth image; (b) estimated image with the fixed window method; (c) estimated image with our proposed method

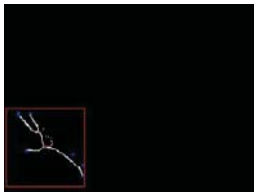
The processing speed using our method is also improved. There is no need to search within the whole image in each frame, so the computational cost is reduced.

Finally, while the "rock" gesture is estimated with over 90% accuracy, the "scissors" and "paper" gestures are estimated with less than 90% accuracy. For the "scissors" gesture, the thumb, ring and little fingers are sometimes not detected. Hence, the average of the distances and the standard deviation is similar to "paper" for similar depth data. Also, when the index and middle fingers are close, the number of detected fingers may be one. Here, the average of the distances and the standard deviation are similar to "rock." Thus, other features for classifying gestures with high accuracy should be established.

(a)



(b)



(c)

Fig. 12: Hand estimation in the V4 and D4 area: (a) raw depth image; (b) estimated image with the fixed window method; (c) estimated image with our proposed method

VI. CONCLUSIONS

In this study, depth images of three hand gestures obtained using a ToF camera were analyzed using geometric calculation and machine learning. Six features were extracted and hand gestures were recognized using an SVM classifier. The estimation accuracy with our method is 85%, which is higher than the 76% accuracy for the fixed window method. An adjustable window corresponding to the estimated palm center position and size is used to accurately search the hand area, and this also contributes to a reduced processing time. While the accuracy for estimating the "rock" gesture is over 90%, the "scissor" and "paper" gestures are estimated with less accuracy. Future work will focus on improving the recognition accuracy and extending the functionality of the proposed method to identify dynamic gestures, such as swiping up and down.

ACKNOWLEDGMENTS

The authors thank Hiroomi Eguchi for his support.

REFERENCES

[1] S. Cafiso, and G. L. Cava, "Driving performance, alignment consistency, and road safety real-world experiment," Transportation Research Record: Journal of the Transportation Research Board, vol. 2102, no. 1, pp. 1–8, 2009.

[2] C.A. Pickering, "Driver distraction trends and issues," Computing and Control Engineering, vol. 16, no. 1, pp. 28-30, 2005.

[3] V.I. Pavlovic, R. Sharma, and T.S. Huang, "Visual interpretation of hand gestures for human-computer interaction: a review," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp.677-695, 1997.

[4] H.-K. Lee, and J.H. Kim, "An HMM-based threshold model approach for gesture recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 10, pp. 961-973, 1999.

[5] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," IEEE Transactions on Multimedia, vol. 15, no. 5, pp. 1110-1120, 2013.

[6] S. Gupta, P. Molchanov, X. Yang, K. Kim, S. Tyree, and J. Kautz, "Towards selecting robust hand gestures for automotive interfaces," IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, pp. 1350-1357, 2016.

[7] F. Parada-Loira, E. González-Agulla, and J. L. Alba-Castro, "Hand gestures to control infotainment equipment in cars," IEEE Intelligent Vehicles Symposium, Dearborn, MI, pp. 1-6, 2014.

[8] T. Yamashita, and T. Watasue, "Hand posture recognition based on bottom-up structured deep convolutional neural network with curriculum learning," IEEE International Conference on Image Processing (ICIP), Paris, France, pp. 853-857, 2014.

[9] X. Yingxin, L. Jinghua, W. Lichun, and K. Dehui, "A robust hand gesture recognition method via convolutional neural network," 6th International Conference on Digital Home (ICDH), Guangzhou, China, pp. 64-67, 2016.

[10] M. Madadi, S. Escalera, X. Baró and J. González, "End-to-end Global to Local CNN Learning for Hand Pose Recovery in Depth data," ArXiv, 1705.09606, 2017.

[11] G. Li, H. Tang, Y. Sun, J. Kong, G. Jiang, D. Jiang, B. Tao, S. Xu, and H. Liu, "Hand gesture recognition based on convolution neural network," Cluster Computing, vol. 22, supp. 2, pp. 2719–2729, 2019.

[12] Z.-H. Chen, J.-T. Kim, J. Liang, J. Zhang, and Y.-B. Yuan, "Real-time hand gesture recognition using finger segmentation," The Scientific World Journal, vol. 2014, Article ID 267872, 9 pages, 2014.

[13] A. Malima, E. Ozgur, and M. Cetin, "A fast algorithm for vision-based hand gesture recognition for robot control," IEEE 14th Signal Processing and Communications Applications, Antalya, Turkey, pp. 1-4, 2006.

[14] S. Chaman, J. Jani, H. Fernandes, R. Dhuka, and D. Mehta, "Real time gesture to automotive control," 2018 International Conference on Current Trends towards Converging Technologies (ICCTCT), Coimbatore, India, pp. 1-6, 2018.

[15] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," IEEE International Conference on Computer Vision Workshops (ICCV Workshops), Barcelona, Spain, pp. 1228-1234, 2011.

[16] C. Wang, Z. Liu, and S. Chan, "Superpixel-based hand gesture recognition With kinect depth camera," IEEE Transactions on Multimedia, vol. 17, no. 1, pp. 29-39, 2015.

[17] Q. De Smedt, H. Wannous, and J. Vandeborre, "Skeleton-based dynamic hand gesture recognition," IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Las Vegas, NV, pp. 1206-1214, 2016.

[18] M. R. L. Varshini, and C. M. Vidhyapathi, "Dynamic finger gesture recognition using kinect," International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Ramanathapuram, India, pp. 212-216, 2016.

[19] W. Chen, C. Wu, and C. H. Lin, "Depth-based hand gesture recognition using hand movements and defects," International Symposium on Next-Generation Electronics (ISNE), Taipei, Taiwan, pp. 1-4, 2015.

[20] T. Q. Vinh, and N. T. Tri, "Hand gesture recognition based on depth image using kinect sensor," 2nd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS), Ho Chi Minh City, Vietnam, pp. 34-39, 2015.

[21] Y.-K. Ahn, and Y.-C. Park, "The hand gesture recognition system using depth camera," The Tenth International Conference on Advances in Computer-Human Interactions, Nice, Fance pp. 234-238, 2017.

[22] T. Nguyen, D. Vo, H. Huynh, and J. Meunier, "Geometry-based static hand gesture recognition using support vector machine," 13th International Conference on Control Automation Robotics & Vision (ICARCV), Singapore, pp. 769-774, 2014.

[23] L. Jaemin, H. Takimoto, H. Yamauchi, A. Kanazawa and Y. Mitsukura, "A robust gesture recognition based on depth data," The 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision, Incheon, pp. 127-132, 2013.

[24] M. Oberweger, P. Wohlhart and V. Lepetit, "Hands Deep in Deep Learning for Hand Pose Estimation," ArXiv, 1502.06807, 2015.

[25] Melexis NV, https://www.melexis.com/

[26] Integral Image by OpenCV, https://docs.opencv.org/4.1.1/d7/d1b/group__imgproc__misc.html#gadeaf38d7701d7ad371278d663c50c77d

[27] NASA, Man-Systems Integration Standards, https://msis.jsc.nasa.gov/

[28] Distance Transform by OpenCV, https://docs.opencv.org/4.1.1/d7/d1b/group__imgproc__misc.html#ga8a0b7fdfcb7a13dde018988ba3a43042

[29] Convex Hull by OpenCV, https://docs.opencv.org/4.1.1/d3/dc0/group__imgproc__shape.html#ga014b28e56cb8854c0de4a211cb2be656

[30] Support Vector Machine, Support Vector Classification, http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html