Sensors and Materials, Vol. 35, No. 12 (2023) 4117–4129
MYU Tokyo

S & M 3471

Real-time Hand Movement Trajectory Tracking with Deep Learning

Po-Tong Wang,[1*] Jia-Shing Sheu,[2] and Chih-Fang Shen[2]

[1]Department of Electrical Engineering, Lunghwa University of Science and Technology,
No. 300, Sec. 1, Wanshou Rd., Guishan District, Taoyuan 333326, Taiwan
[2]Department of Computer Science, National Taipei University of Education,
No. 134, Sec. 2, He-Ping East Road, Da-an District, Taipei 106, Taiwan

(Received July 15, 2023; accepted November 9, 2023)

Keywords: real-time hand tracking, deep learning, single-shot multibox detector (SSD), CAMShift, object detection, human–computer interaction (HCI)

In this study, we employed deep learning to develop a real-time hand trajectory tracking system. Our primary approach integrates the MobileNetv2 single-shot multibox detector, known for accuracy, with the versatile CAMShift algorithm. This synergy ensures robust hand detection across diverse scenarios. Through rigorous testing on webcam images and leveraging advanced feature extraction methods, such as contour discernment and skin hue differentiation, we report an 88.17% increase in detection accuracy over traditional models. Moreover, with a latency of merely 0.0343 s, our system demonstrates its prowess in immersive gaming and assistive devices for individuals with disabilities

1. Introduction

With the advancement of computer technology and artificial intelligence, there is a growing emphasis on body language and human–computer interaction (HCI) recognition for future technological applications.[1–4] The natural characteristics of HCI have led to extensive research on static gestures or hand movements, such as sign language, across fields such as artificial intelligence, virtual reality, multimedia, and natural language communication.[5–8] Dynamic gestures, in particular, offer a potent medium for the general public to convey information effectively.

Historically, hardware devices were the primary tools for detecting hand positions in dynamic gestures. However, with the advent of computer vision (image processing), machine learning, and deep learning, there are now methods to detect images containing human hands. Deep learning has shown significant strides in image recognition, especially object detection.

In this research, our focus is on deep learning for hand detection. Once the hands are detected, relying on a deep learning network for recognition in each frame might not be efficient, given the potential performance constraints on devices with limited hardware capabilities. A

*Corresponding author: e-mail: neojwang@gm.lhu.edu.tw
https://doi.org/10.18494/SAM4592

ISSN 0914-4935
https://myukk.org/

more streamlined way is predicting hand features within the hand region of interest (ROI) and tracking the hand's movement trajectory.

We present a dynamic gesture-tracking system that employs deep learning to detect human hands from images captured via a network camera. Upon successful detection, the hand ROI undergoes binarization for enhanced clarity. Tracking algorithms then predict the hand's position in subsequent frames, facilitating the continuous tracking of hand movement trajectories. This tracked data is then visualized on a computer, providing users with an intuitive view of hand movements.

2. Literature Review

Research in hand detection predominantly falls into two categories: pure palm detection and sign language recognition. Various methodologies, ranging from image processing to machine learning and deep learning, have been employed in these studies. The classical image processing approach encompasses techniques such as color space transformation, skin color detection, edge detection, and pinpointing hand localization within images.[9] Machine learning techniques, especially the support vector machine with hand-containing training images, have reported accuracies exceeding 90%.[10,11]

Deep learning, given its versatility, has found applications in numerous domains. Specifically, the single-shot multibox detector (SSD)[12] integrated with VGG16,[13] a renowned Convolutional Neural Network, has been pivotal in recognizing sign language, particularly within the deaf community. The optimized MobileNetv2 SSD stands out because of its efficiency and accuracy in object detection tasks, especially in real-time scenarios.[14] Google's MediaPipe module, a fusion of SSD and a neural feature pyramidal network, excels in real-time palm detection and has showcased impressive results even on embedded devices. Modules built upon MediaPipe can compute coordinates and angles of each hand node, leveraging hand skeletons, thereby facilitating sign language recognition.[15–17]

In hardware, devices such as Kinect have been instrumental in dynamic hand tracking when paired with neural networks or tracking algorithms. For instance, Kinect discerns the movement trajectory of the palm and subsequently employs a backpropagation neural network (BPNN) to decipher the pattern traced by the movement.[18] Multicamera setups have been explored for sustained target tracking.[19] After successful hardware-based hand coordinate detection, Kalman filtering has emerged as a preferred choice for tracking.[20,21] The integration of MediaPipe with tracking algorithms has further enhanced hand-tracking capabilities. Techniques that extract histograms of oriented gradients from images, followed by deploying the continuously adaptive mean-shift (CAMShift) algorithm, have been proposed as viable hand-tracking solutions.[22,23]

3. System Architecture and Methodology

Our hand-tracking system, equipped with a webcam, processes images in real time through a series of interconnected modules. This system is structured around three primary modules: the

hand detection module, which identifies the hand's presence; the CAMShift module, which is responsible for adaptive tracking; and the track retention module, which ensures consistent tracking across frames (Fig. 1).

Hand detection is orchestrated through three pivotal steps: image resizing, mean RGB values subtraction,[24] and SSD detection. Initially, RGB images, sourced from diverse devices such as webcams, undergo resizing to a standardized dimension of 300 × 300 pixels. This dimension, chosen on the basis of empirical studies, is tailored to amplify the SSD's performance. A cornerstone of this methodology is the subtraction of mean RGB values, a normalization technique that harmonizes the image's average intensity, fortifying object detection capabilities. This process accentuates the object's distinct features, ensuring steadfast detection, even when faced with varying luminosities within the image. The transformative impact of this step is vividly illustrated by the pronounced contrast between the images' pre- and post-mean RGB value subtraction, as shown in Fig. 2 The intricate process of mean RGB subtraction is elaborated in Eqs. (1) and (2), with the entire workflow illustrated in Fig. 3.

$$\left(R',G',B'\right)=\left(R-R_{Mean},G-G_{Mean},B-B_{Mean}\right) \tag{1}$$

$$\left(R_{Mean},G_{Mean},B_{Mean}\right)=\left(\frac{\sum R_i}{N},\frac{\sum G_i}{N},\frac{\sum B_i}{N}\right) \tag{2}$$

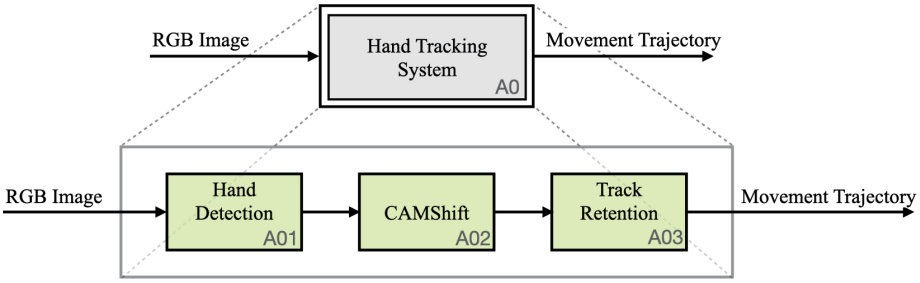
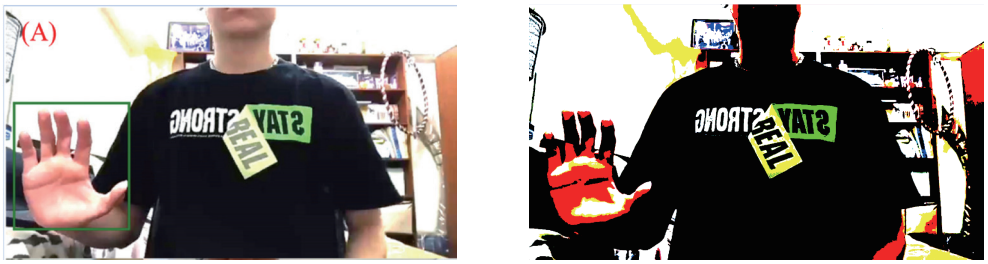


Fig. 1. (Color online) System architecture for hand tracking.



(a) (b)

Fig. 2. (Color online) (a) Original image and (b) Image after mean RGB value subtraction.

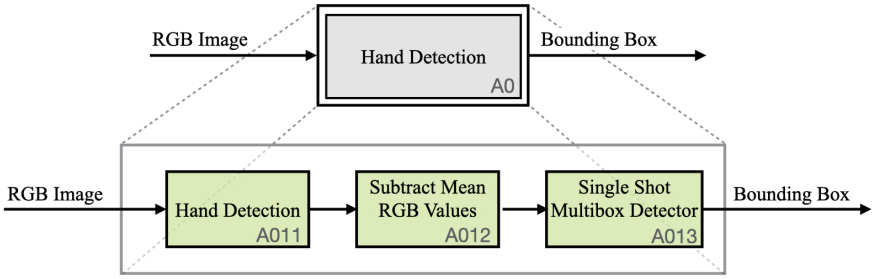


Fig. 3. (Color online) Module for hand detection.

Following the preprocessing steps, the SSD, a specialized detection architecture, commences its detection routine. The SSD boasts a VGG16-based neural network structure meticulously designed to bolster model performance. This design choice facilitates the removal of fully connected layers, paving the way for the integration of supplementary convolutional layers. A distinctive feature of the SSD is its reliance on multiscale feature maps, colloquially termed a pyramidal feature hierarchy. This hierarchy is instrumental in detecting targets spanning different sizes across a spectrum of feature maps. Figure 4 offers a visual representation of the SSD's architecture, where the foundational layers are rooted in VGG16, while the ensuing convolutional layers function as feature maps, adept at detecting objects across a range of sizes. This stratagem proficiently predicts miniature objects in the nascent layers and more substantial objects in the advanced layers. The SSD incorporates prior boxes, characterized by diverse sizes and aspect ratios, to further refine detection precision across each feature map.

The SSD's output encapsulates the coordinates of the detected entities. To ensure the output mirrors real-world scenarios, redundant bounding boxes are meticulously pruned using an expedited nonmaximum suppression technique. Subsequently, the position of each detected entity is restricted on the image, facilitating intuitive visualization.

3.1 CAMShift

Upon successful detection, the CAMShift module is invoked to facilitate hand tracking. Figure 5 delineates the intricate architecture of the CAMShift module. To kickstart the tracking mechanism, the bounding box's spatial coordinates and dimensions are inferred from the antecedent frame's hand detection outcomes. Recognizing the susceptibility of the RGB space to luminosity fluctuations, which can significantly impede tracking efficacy, the image undergoes a transformation from RGB to a more stable HSV space. Subsequently, a histogram representing the hue (h) component is constructed, encapsulating the probability distribution of diverse hue values scattered across the image's pixels. Figure 6 vividly illustrates the histogram post-initialization. In the next step, a back-projection technique is used to map the value of each pixel in the image to obtain a color probability distribution map, which is represented as a grayscale image as follows:

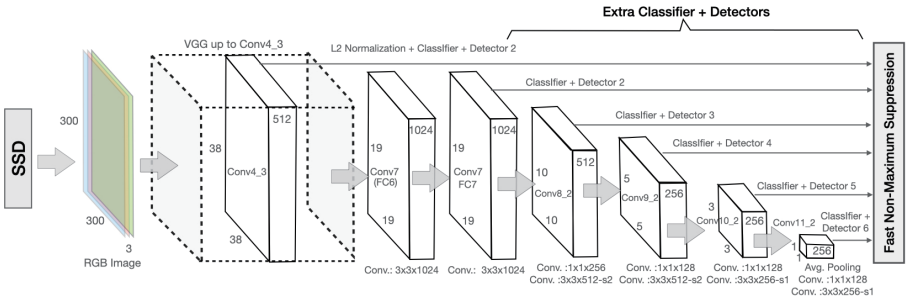


Fig. 4. (Color online) Structure of SSD.

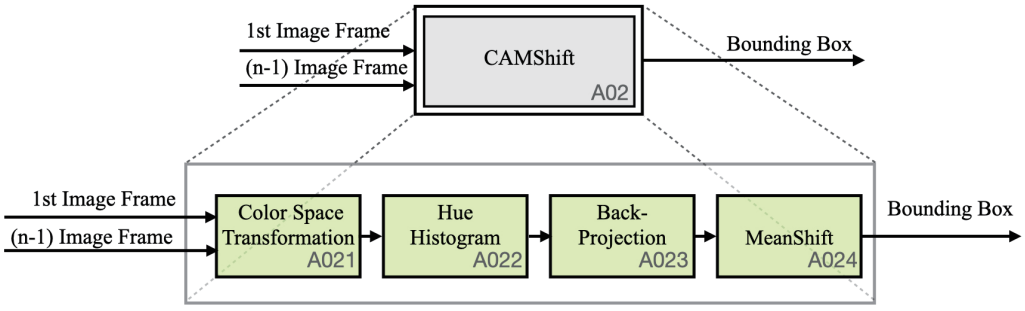


Fig. 5. (Color online) Design of CAMShift module.

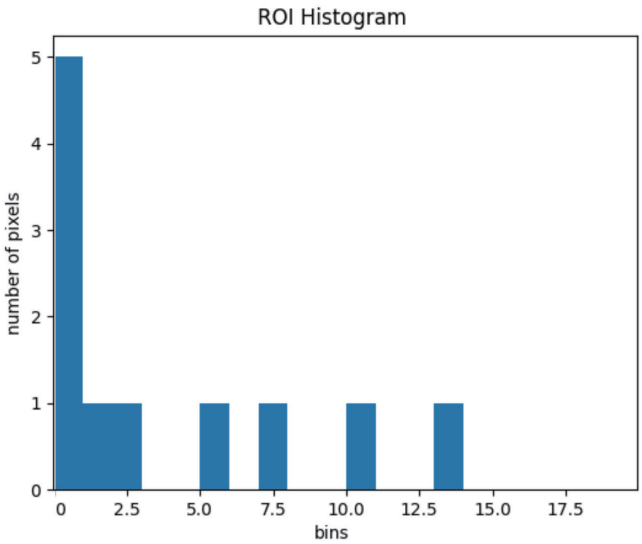


Fig. 6. (Color online) Histogram after initialization.

$$I(x,y)=\sum_{i=1}^{m}\left(\delta\left[c(x,y)\right]\times q_i\right). \tag{3}$$

By leveraging the mean-shift algorithm, the color probability distribution map within the confines of the bounding box is selected. This is followed by the computation of the zeroth- and first-order distance matrices, symbolized by $I(x, y)$. These matrices encapsulate the pixel value at the designated position (x, y) within the field, iterating over the pixels within the bounding box.

$$M_{00}=\sum_{x}\sum_{y}I(x,y) \tag{4}$$

$$M_{10}=\sum_{x}\sum_{y}xI(x,y) \tag{5}$$

$$M_{01}=\sum_{x}\sum_{y}yI(x,y) \tag{6}$$

A meticulous analysis of these distance matrices yields the centroid (x_c, y_c) and the dimensions of the emergent bounding box. The bounding box undergoes adaptive resizing, ensuring that it swiftly acclimatizes to alterations in the target region, thereby guaranteeing effective tracking.

$$x_c=\frac{M_{10}}{M_{00}} \tag{7}$$

$$y_c=\frac{M_{01}}{M_{00}} \tag{8}$$

$$w=\sqrt{\frac{M_{00}}{256}} \tag{9}$$

$$l=1.2w \tag{10}$$

As the process advances to the ensuing frame, the outcome procured after step A024, encompassing the bounding box's coordinates, serves as the foundation to recalibrate the bounding box's dimensions and position. This paves the way for the execution of step A021, ushering in the next tracking phase.

3.2 Track retention

The track retention module is instrumental in meticulously preserving the intricate tracking data associated with hand movements. As the CAMShift mechanism unfolds, the bounding box coordinates are adeptly harnessed to pinpoint the hand's central locus within the freshly charted hand trajectory. Subsequently, these pivotal center point coordinates are systematically cataloged within an array, ensuring a seamless and accurate tracking of the hand's trajectory over time. Figure 7 offers a comprehensive visual representation, elucidating the sophisticated architecture underpinning the track retention module.

4. Experimental Results

4.1 Environment

Our experiments were meticulously executed on a Microsoft® Windows® 10 Home Edition computer. This platform was judiciously chosen for its stability and compatibility with our tools. The system was powered by an Intel i5-12500 CPU, bolstered by 32 GB of RAM, ensuring swift data processing and multitasking. Further enhancing our computational capabilities was an Nvidia GeForce RTX 3070 graphics card, pivotal for the intensive graphical tasks inherent in our research.

4.2 Image databases

Our experiments framework leveraged two distinct and comprehensive databases, each catering to specific facets of hand movement analysis. The inaugural database, known as the EgoHands database,[25] boasts a collection of 4800 images, each with a resolution of 1280 × 720 pixels. These images, captured via Google Glass, predominantly spotlight first-person interactions, often featuring up to four hands in diverse scenarios, such as engaging in card games in an office or collaborative puzzle-solving in a courtyard. A visual glimpse into the EgoHands database is provided in Fig. 8.

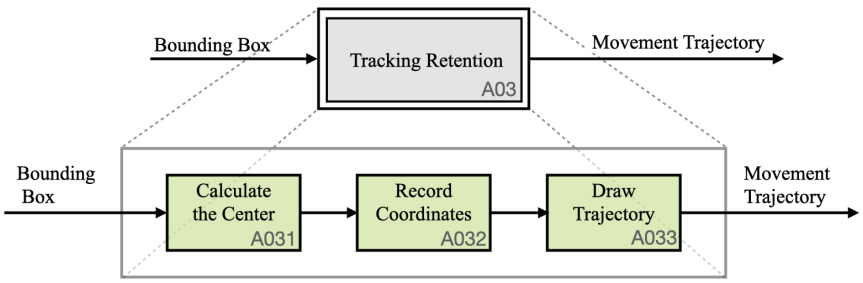


Fig. 7. (Color online) Module for track retention.

Fig. 8. (Color online) Sample images from EgoHands database.

In contrast, our second database was a bespoke creation, meticulously curated to encompass hand-waving videos. Postprocessing these videos yielded 5065 images, each standardized to a resolution of 1920 × 1080 pixels. This custom database was designed to be versatile, encapsulating four distinct scenes: a bedroom, a living room, a dining room, and an academic classroom. Figure 9 offers a visual tour of this database, underscoring the rich diversity in hand poses and ambient backgrounds.

4.3 Experimental results

To holistically evaluate the performance of our model, we leaned on average precision (*AP*) as our primary metric, given its widespread acceptance in gauging detection tasks. Following established protocols, *AP* was computed using precision and recall. Figure 10 vividly showcases our detection outcomes juxtaposed against the original input, whereas Fig. 11 clearly shows the binarized ROI, amplifying the discernibility of hand trajectories.

$$AP = \frac{1}{11} \sum_{R \in \{0, 0.1, \ldots, 0.9, 1.0\}} P_{Interp}(R) \tag{11}$$

We judiciously partitioned all databases for our experiments, allocating 80% for training and the remaining 20% for validation, ensuring a balanced distribution and robust evaluation. Our experimental setup was consistent, employing a batch size of 16, a momentum of 0.9, a learning rate of 0.0001, and an intersection-over-union threshold of 0.5.

Fig. 9. (Color online) Images from custom database.

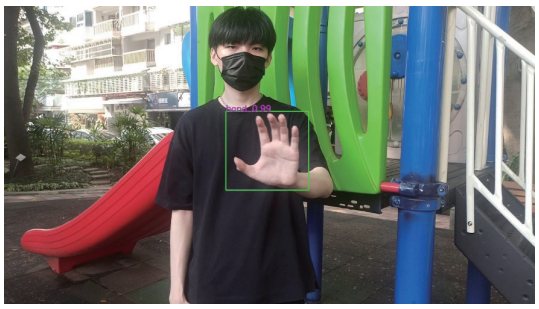


Fig. 10. (Color online) Detection outcomes with original input.

Fig. 11. (Color online) Region of interest in binary.

Table 1 lists our experimental outcomes, juxtaposing the performance metrics across different databases. A nuanced analysis revealed a marginal performance delta between the EgoHands and our custom databases. Intriguingly, amalgamating the two databases manifested in a slight dip in performance. These observations underscored the SSD mode's prowess in hand detection, albeit with nuanced variations contingent on the dataset.

Figure 12 visually encapsulates the experimental outcomes achieved with the SSD model, demarcated by the vibrant green rectangle. In contrast, the hand-tracking outcomes, derived from the CAMShift algorithm, are vividly portrayed by a purple rectangle in Fig. 13. A meticulous comparison between the two figures unveils a subtle size discrepancy between the bounding boxes. Figure 13 further embellishes the narrative with a purple trajectory line, offering an intuitive visualization of hand motion and underscoring the system's precision and efficacy.

Table 1
Experimental results.

	EgoHands database	Our Custom database
AP	0.587	0.576

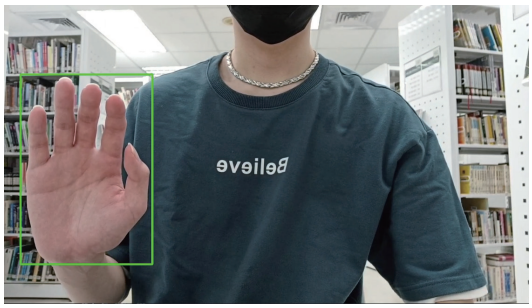


Fig. 12. (Color online) Results of the SSD model experiments.

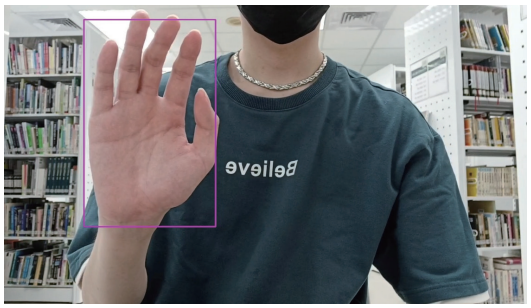


Fig. 13. (Color online) Hand tracking outcomes using the CAMShift algorithm.

To rigorously assess our system's efficiency, we orchestrated a series of tests, gauging the execution time per frame for MobileNetv2 SSD, both in the presence and absence of the CAMShift algorithm. While hand detection was a staple for the inaugural frame, the ensuing frames witnessed a marked reduction in execution time when augmented with the CAMShift algorithm. This observation resonates with the CAMShift algorithm's lightweight nature, ensuring minimal hardware overhead (Table 2). Such efficiency enhancements, courtesy of the CAMShift algorithm, bolster the real-time performance of our system, rendering it apt for real-world applications.

5. Discussion

In this study, we have meticulously developed a system tailored for real-time hand movement tracking anchored firmly on the SSD model. Our empirical evaluations underscore the prowess of tracking algorithms, with CAMShift standing out in significantly curtailing hardware overhead. Notably, our system registers an impressive 88.17% increase in detection efficacy compared with traditional methods, complemented by a swift latency of 0.0343 s. These metrics resonate with the system's robustness and aptitude for real-time hand movement tracking across many practical use cases.

By delving deeper into deep learning, integrating avant-garde models with tracking algorithms promises to bolster accuracy and system reliability. Furthermore, the burgeoning domain of compact neural networks tailored for embedded systems presents a tantalizing prospect warranting exploration. Striking an optimal harmony between detection fidelity and computational alacrity remains paramount, mainly when catering to devices constrained by resources.

Table 2
Execution times of MobileNetv2 SSD with and without CAMShift.

MobileNetv2 SSD	Without CAMShift	With CAMShift
Computing/frame (s)	0.0389	0.0046

Our scholarly endeavors lay a robust foundation for groundbreaking strides in hand movement tracking. We foresee many HCI and gesture-centric control applications by synergizing deep learning paradigms with adept tracking methodologies. As the tapestry of research unfolds, we remain optimistic about unearthing myriad avenues that promise to revolutionize user experiences, making them more intuitive and immersive.

6. Conclusions

Real-world applications underscore the importance of comprehending the intricacies, challenges, and limitations of hand movement tracking. Central to this discourse is the quest to harmonize detection accuracy with computational agility. Anchored by the SSD model, our system manifests stellar performance metrics in real-time tracking. However, the pursuit of excellence intimates the potential for precision augmentation. While CAMShift stands as a beacon of hardware resource conservation, its fidelity in tracking might waver in fluidic environments.

The amalgamation of cutting-edge deep learning architectures is non-negotiable to elevate our system to the zenith of its potential. Titans in this space, such as You Only Look Once (YOLO) and Faster R-CNN, renowned for their prowess in object detection, promise to ease accuracy concerns. However, they might amplify computational overheads. This dichotomy accentuates the allure of edge computing and the strategic deployment of hardware accelerators.

The versatility of our system, evident in domains such as HCI and gesture-centric controls, hints at a broader application canvas. A case in point is its potential metamorphosis for telemedicine applications, especially in the nuanced realm of remote patient monitoring for afflictions such as Parkinson's disease.

In the ever-evolving technological tapestry, it is paramount to remain aware that emergent technologies either bolster our assertions or pose challenges. The advent of avant-garde paradigms such as quantum computing and neuromorphic chips reiterates the ethos of adaptability.

In conclusion, this discourse underscores the imperative for a holistic strategy that augments system accuracy, optimizes computational throughput, and broadens its application spectrum. The fluid dynamics of this domain reiterate the essence of sustained research endeavors, all aimed at sculpting a truly adaptive hand movement tracking paradigm.

Acknowledgments

This work was supported by the Taiwan Ministry of Education (MOE) Teaching Practice Research Program (Grant Numbers PSK1110004).

References

1 S. S. Rautaray and A. Agrawal: Artif. Intell. Rev. **43** (2012) 1. https://doi.org/10.1007/s10462-012-9356-9
2 S. Kim, G. Park, S. Yim, S. Choi, and S. Choi: IEEE Trans. Consum. Electron. **55** (2009) 1169. https://doi.org/10.1109/TCE.2009.5279454
3 O. K. Oyedotun and A. Khashman: Neural Comput. Appl. **28** (2016) 3941. https://doi.org/10.1007/s00521-016-2385-8
4 J. Jeong and Y. Jang: Soft Comput. **19** (2014) 815. https://doi.org/10.1007/s00500-014-1310-x
5 X. Ma and J. Peng: J. Sens. **2018** (2018) 9. https://doi.org/10.1155/2018/5809769
6 S. F. Chevtchenko, R. F. Vale, and V. Macario: Expert Syst. Appl. **92** (2018) 170. https://doi.org/10.1016/j.eswa.2017.09.042
7 C. Mummadi, F. Leo, and K. Verma: Informatics **5** (2018) 28. https://doi.org/10.3390/informatics5020028
8 D. Xu, X. Wu, Y.-L. Chen, and Y. Xu: J. Intell. Rob. Syst. **77** (2014) 583. https://doi.org/10.1007/s10846-014-0041-4
9 H. Y. Lai and H. J. Lai: Proc. IEEE Conf. Int. Symp. Comput. Consum. Control (2014) 658. https://doi.org/10.1109/IS3C.2014.107
10 C. Cortes and V. Vapnik: Mach. Learn. **20** (1995) 273. https://doi.org/10.1007/BF00994018
11 C. Miron, A. Pasarica, and H. Costin: Proc. IEEE Conf. E-Health Bioeng. Conf. (IEEE, 2019) 1. https://doi.org/10.1109/EHB47259.2019.8919794
12 W. Liu, D. Anguelov, and D. Erhan: Proc. Conf. Eur. Conf. Comput. Vis. (ECCV) (Springer, 2016) 21. https://doi.org/10.1007/978-3-319-46493-0_2
13 K. Simonyan and A. Zisserman: arXiv preprint arXiv: 1409.1556 (2015). https://doi.org/10.5244/C.29.CVPRW.1
14 H. Lee, H. Kim, and J. I. Kim: IEEE Trans. Multimedia **18** (2016) 2093. https://doi.org/10.1109/TMM.2016.2595262
15 F. Zhang, V. Bazarevsky, and A. Vakunov: arXiv preprint arXiv: 2006.10214 (2020). https://doi.org/10.1109/CVPR42600.2020.01193
16 T. Y. Lin, P. Dollar, and R. Girshick: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR) (IEEE, 2017) 2117. https://doi.org/10.1109/CVPR.2017.233
17 A. Halder and A. Tayaed: Int. J. Res. Publ. Rev. **3** (2021) 9. https://doi.org/10.17583/ijrpr.2021.5405
18 J. S. Sheu and Y. L. Huang: Multimed. Tools Appl. **75** (2016) 9685. https://doi.org/10.1007/s11042-015-3123-2
19 P. T. Wang, J. S. Sheu, and J. H. Lai: Sens. Mater. **34** (2022) 563. https://doi.org/10.18494/SAM.2022.3410
20 T. L. Baldi, S. Scheggi, L. Meli, M. Mohammadi, and D. Prattichizzo: IEEE Trans. Hum.-Mach. Syst. **47** (2017) 1066. https://doi.org/10.1109/THMS.2017.2707384
21 L. Yang, M. Wang, and T. Li: Proc. IEEE Int. Conf. Inf. Syst. Comput. Aided Educ. (ICISCAE, 2020) 708. https://doi.org/10.1109/ICISCAE49843.2020.000-8
22 J. Guo, J. Cheng, Y. Guo, and J. Pang: Mech. Mater. **3** (2013) 849. https://doi.org/10.4028/www.scientific.net/AMM.311-313.849
23 J. Henriques, R. Caseiro, P. Martins, and J. Batista: IEEE Trans. Pattern Anal. Mach. Intell. **37** (2014) 583. https://doi.org/10.1109/TPAMI.2014.2345396
24 S. Bambach, S. Lee, D. J. Crandall, and C. Yu: Proc. IEEE Interact. Conf. Comput. Vis. (ICCV) (2015) 1949. https://doi.org/10.1109/ICCVW.2015.246
25 Q. Gao, J. Liu, Z. Ju, and X. Zhang: IEEE Trans. Ind. Electron. **66** (2019) 9663. https://doi.org/10.1109/TIE.2109.2899647.

About the Authors

Po-Tong Wang received his M.S. degree in computer science from National Taipei University of Education (NTUE), Taiwan, in 2015. He then earned his Ph.D. degree from the Department of Bio-Industrial Mechatronics Engineering at National Taiwan University (NTU) in 2019. He is an assistant professor in the Department of Electrical Engineering at Lunghwa University of Science and Technology, Taiwan. His research interests include AI hardware accelerators, color space transformation algorithms, and developing PLCopen embedded systems. Additionally, he delves into high-speed industrial communication protocols. A significant aspect of his research revolves around the synergy of TinyML with PLCopen automation, emphasizing LLMs. (neojwang@gm.lhu.edu.tw)

Jia-Shing Sheu received his M.S. and Ph.D. degrees from the Department of Electrical Engineering at National Cheng Kung University, Tainan, Taiwan, in 1995 and 2002, respectively. Currently, he is a professor in the Department of Computer Science at National Taipei University of Education, Taipei, Taiwan. His research interests include pattern recognition and image processing, particularly focusing on real-time face recognition and embedded systems. (jiashing@tea.ntue.edu.tw)

Chih-Fang Shen received his B.S. and M.S. degrees from the Department of Computer Science at National Taipei University of Education, Taiwan. His main interests are in image processing, sensor applications, and embedded systems. (chaunceyshen@gmail.com)