Hand Action Recognition

Nour Atamni
Computer Science
Ben-Gurion University of the Negev
Beer-Sheva, Israel
atamni@post.bgu.ac.il

Said Naamneh
Computer Science
Ben-Gurion University of the Negev
Beer-Sheva, Israel
saeednaa@post.bgu.ac.il

Jihad El-Sana
Computer Science
Ben-Gurion University of the Negev
Beer-Sheva, Israel
el-sana@cs.bgu.ac.il

***Abstract*—This paper presents a new dataset for hand action detection for manipulating (assembling and dismantling) mechanical devices and an action detection model based on Transformers. An entry in this dataset is a first-person-view video segment that shows hands performing an action. These hands may utilize a tool and act on an object of the device. These actions were categorized into 12 classes for simple representation. The deep learning model extracts features from each frame in a video, adds position embedding, and feeds the obtained feature vectors to a Transformer Encoder. The output vector goes through a fully connected network to obtain the final class. We have implemented our model and trained it using the presented dataset. We experimentally evaluate the learning and obtain encouraging results.**

***Index Terms*—Hand recognition, action recognition, action recognition dataset**

I. Introduction

Modern life relies heavily on mechanical and electrical devices, encompassing everything from household appliances, automobiles, and aircraft to machinery, industrial setups, and power plants. These essential components of our daily lives are subject to failure, demanding consistent maintenance that usually requires professional workers. Nevertheless, the shortage of skilled labor increases maintenance and repair expenses, resulting in extended service delays. Furthermore, this scarcity often opens the door for less qualified and experienced technicians to take on repair tasks, potentially leading to higher costs and prolonged repair duration.

Several conventional guidance methods employ a combination of virtual, mixed, or Augmented Reality (AR) interfaces to assist in the operation of machinery. These approaches eliminate the need for users to carry physical manuals while performing maintenance or repair tasks, instead displaying instructions within the real-world work environment. However, these techniques are primarily utilized in high-end industries, such as automotive and aerospace, as they cater to expert users and rely on clearly defined environments and predefined workflows. Creating these predefined workflows can be costly, involving a deep understanding of the procedure, skilled engineers, and the manual creation of guiding illustrations and animations by artists and engineers. Consequently, AR has remained inaccessible and prohibitively expensive for low-end enterprises like small businesses, garages, and repair workshops. One approach to overcome this limitation is automatically creating assembly and disassembly workflows for mechanical devices from video segments. These workflows are then employed to assist inexperienced users in executing these workflows through an augmented reality interface. Detecting and analyzing hand action is an essential first step for automatically creating such workflows.

Human Action Recognition (HAR) automatically identifies and classifies human actions or activities from visual data, such as videos or image sequences. This paper deals with human actions that involve only the hands (of a human body) and may include additional objects. In some sense, this is similar to hand gestures.

This paper presents the VML-Working-Hands, a novel dataset designed for recognizing hand actions, and a deep learning model trained on this dataset to identify these actions. The dataset focuses on capturing the process of assembling and dismantling mechanical devices. Each video segment in the dataset portrays a single action and encompasses multiple frames before and after the action.

The action recognition model is based on the Transformers architecture, similar to ViT [1], as it only uses the encoder part. We create a *saliency map* encompassing the working hands, the applied tool/s, and the manipulated device part for each frame. An input frame is multiplied by its corresponding mask to guide a Convolutional Neural Network (CNN) to extract representative features from the region of interest, see Figure 2. The sequence of CNN features, computed from the series of frames that define the action, is fed to a Transformer model. The model's output goes through a fully connected network that determines the action.

The rest of this paper is organized as follows. The following section overviews related work, Section II. Next, we present our new dataset, VML-Working-Hands, and discuss our hand action recognition model in the Sections IV and III, respectively. Section V reviews our implementation details and experimental results. Finally, we draw some conclusions and directions for future work, Section VI.

II. Related Work

Wearable cameras, mounted on the head or chest, enable the examination of hands from a standpoint that offers a firsthand outlook on the surroundings. This realm of study within computer vision is recognized as *egocentric* or *first-person vision* (FPV). Egocentric vision offers advantages over third-person vision, capturing the user's perspective, minimizing

obstructions, and aligning with actions. However, FPV's challenge is the camera's mobility, causing quality and distinction issues due to rapid movements and lighting changes.

Understanding daily activities in an egocentric context emphasizes the importance of object-hand interactions for action recognition. Next, we briefly overview closely related work on hand localization, hand pose estimation, and action and interaction.

A. Hand Localization

Hand localization algorithms aim to estimate the accurate position of the hand within the image [2]. While numerous hand-detection, pose-estimation, and segmentation algorithms have been developed for third-person vision [3], [4], the egocentric point of view poses unique challenges that hinder a straightforward adaptation of these methods. Betancourt et al. [5] introduced a method utilizing HOG features and an SVM classifier for frame-level hand presence prediction, effectively reducing false positives. Zhao et al. [6] detect hands in each frame by leveraging the typical hand interaction cycle, which includes a preparatory phase, interaction, and hands exiting the frame. Based on this cycle, they introduced an ego-saliency metric to estimate the likelihood of hands being present in a frame. Bambach et al. [7] proposed a probabilistic approach combining spatial biases and appearance models to generate region proposals. Using classification, they generated 2,500 regions per frame and applied GrabCut [8] to obtain hand segmentation masks within bounding boxes for comprehensive coverage. zhu2016two

Zhu et al. [9] applied structured random forest to create hand probability maps at the pixel level. These maps were then fed into a multitask CNN to locate the hand's bounding box, shape within the box, and wrist/palm positions. Cartas et al. [10] used skin region segmentation to propose regions and determine if they correspond to one or two arms. Jian [11] employed a Hand Localization Network (HALNet) based on ResNet50, trained on synthetic data, to predict the hand's center position. Then, they cropped a Region of Interest (ROI) around this point, adjusting for its distance from the camera. General object detection approaches such as YOLO (You Only Look Once) [12] were applied to localize hands in FPV [13].

B. Hand Pose Estimation

Hand pose estimation identifies hand components represented as 2D joints or semantic sub-regions. It usually focuses on regions of interest (ROIs) previously detected through either a hand detection or segmentation algorithm.

Liang et al. [14] employed a conditional regression forest (CRF) to estimate hand pose from binary hand masks, considering different camera distances. They adopted a segmentation step to improve joint localization by dividing the binary silhouette into twelve semantic hand regions using a random forest and binary context descriptors. Zhu et al. [15] et al. [40] employed a structured forest for hand segmentation into thumb, fingers, palm, and forearm regions, adapting the existing structured regression forest framework for this multiclass segmentation challenge.

Some approaches adapted CNN architectures for human pose estimation, such as OpenPose [16], to handle hand pose estimation, including localizing hand joints [16]. Tekin et al. [17] employed an FCN to simultaneously estimate the 3D poses of both the hand and an object. The FCN generated a 3D grid for each frame, and the 3D hand joint positions were determined by combining the predicted locations within this grid.

C. Action and Interaction

Betancourt et al. [18] explore the popular processing steps for developing hand-based applications and suggest a hierarchical structure that optimally switches between each level to reduce the computational cost and improve its performance.

Actions and interaction approaches could be classified into two main classes: those relying solely on hands as the prediction cue and those utilizing a combination of object and hand cues for prediction.

Singh et al. [19] proposed a CNN-based method for recognizing actions. They feed hand segmentation, head motion information, and a saliency map to 2-stream architecture, combining 2D and 3D CNNs for feature extraction. They apply SVM to predict action from these features. Urabe et al. [20] identifies cooking actions by analyzing the hand region, using 2D and 3D CNNs to create appearance and motion maps. Combining these outputs through class-score fusion yielded better results than using each stream alone. Tang et al. [21] enhanced action recognition using a multi-stream deep neural network (MDNN) using optical flow and depth maps. They included a hand stream consisting of a CNN with the hand mask as input. They improved the recognition rate by combining the hand stream features with the MDNN through weighted fusion.

Ma et al. [22] used a multi-stream approach, with one stream for object recognition and another for action prediction, combining object labels and action verbs for interaction recognition. Zhou et al. [23] employed hand segmentation, object features, and optical flow to localize and recognize active objects, then used non-linear SVMs to recognize interactions. Both approaches highlight the value of combining object and hand cues for improved recognition.

III. Dataset [1]

We capture first-person perspective videos showing hands engaged in various activities, with a particular emphasis on tasks related to assembling and dismantling mechanical devices; see sample examples in Figure 1. Our dataset categorizes these activities into 12 distinct hand-action categories, as outlined in Table I. Each category encompasses a compilation of specific actions, the potential tools utilized for these actions, and the number of hands typically involved in executing these tasks.

[1]The dataset is intended to be public, and we will include the link in the final version of the paper once it is accepted.

Fig. 1. Sample images illustrating the elements of the collected dataset.

We presented the dataset concisely in Table I according to categories to avoid a long list of actions. A typical action appears in various permutations, depending on the number of hands and the applied tools. For example, the pull action in the Tug category was observed with one hand and two hands. It is performed with bare hand/s (no tool) or pliers.

Category	Action	Hand	Tool	Video
Screw	in, out	1,2	Screwdriver, spanner, wrench, Allen wrench, E.Screwdriver	569
Hammering	Hammering	1	hammer, mallet	250
Tug	Push, Pull	1,2	∅, pliers	220
Cut	Cut Wire/Sheet	1,2	∅, pliers, saw	200
Plug	Plug, Unplug	1,2	∅, pliers	76
OpenClose	Open, Close	1	∅, pliers	111
Click	Click	1	∅	36
Measure	Measure	1,2	Roller, Tap. Caliper	24
Cover	Cover, Uncover	1,2	∅	151
Attach	Attach, Detach	1,2	∅, hammer, pliers	273
Lift	Lift	1	∅, pliers	95
Piping	Open, Close	1,2	wrench, pliers	265

TABLE I

DATASET SUMMARY: THE HAND COLUMN INDICATES THE POSSIBLE NUMBER OF HANDS USED TO PERFORM THE ACTIONS IN THIS CATEGORY, THE TOOL COLUMN INCLUDES THE LIST OF POSSIBLE TOOLS, AND THE VIDEO COLUMN SHOWS THE NUMBER OF VIDEO SAMPLES IN THIS CATEGORY.

We assign a label to each video segment and generate a *saliency mask* for every frame within these segments, highlighting the regions of interest. Our initial step involves segmenting the hands, the applied tools, and the components of the device, which are part of this action. These components are usually in close proximity to the working hands. Furthermore, we aim to incorporate the nearby background in our analysis. To achieve this, we calculate the bounding ellipse, denoted as $\mathcal{E}$, encompassing the detected hands, tools, and components relevant to the action definition. This ellipse, $\mathcal{E}$, serves as the foundation for constructing a *saliency mask*, with values within $\mathcal{E}$ receiving full attention (1.0), while those outside exhibit Gaussian fading. This configuration enables our learning model to prioritize the hand pose and the constituent elements influencing the action.

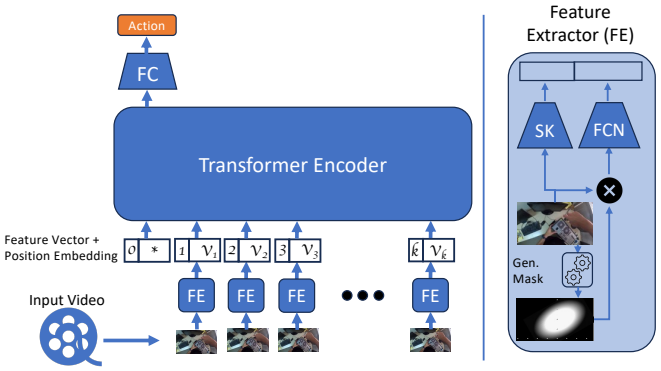

Fig. 2. The Action Recognition Architecture: The video frames are passed through a feature extraction model (FE) on the left. Then, a position order is added to each vector and passed to a Transformer model. The output goes through a fully connected network (FC) to determine the action label.

IV. HAND ACTION RECOGNITION

Action detection identifies when a specific action or activity begins and ends within a video segment, action classification assigns a label or category to the recognized action, such as screwing, moving, or pushing, and an action localization determines the position of the hands performing the action in the image space.

Following the taxonomy of Tekin et al. [17], an action is characterized as a single verb, e.g., push, whereas an interaction is described as a verb-noun pairing, e.g., pushing a button.

Detecting hand or manual action involves localizing the working hands, the applied tools, and the manipulated part of the device. Let us refer to these elements as the *action components*. The *action components* allow for narrowing down the analysis to specific regions of interest (ROIs), excluding irrelevant background data. The pose of the hands, which includes the relative location of the joints, is crucial to interpreting hand actions.

To analyze a video segment, i.e., a sequence of frames, of hands performing an action, such as *screw a bolt*, we start by localizing the hands, the applied tool/s, and the manipulated part of the device, i.e., the action components, which usually nearby. The nearby background usually includes features that may play a role in determining the action. Therefore, we define the *region of the action*, $\mathcal{R}$, as the bounding ellipse of the components of the action. The region $\mathcal{R}$ defines the *saliency mask*, which has one within the $\mathcal{R}$ and Gaussian fading values outside. This setting allows the learning model to focus on the hand pose and the components that determine the action. The model is based on the Transformers architecture, similar to ViT [1], as it only uses the encoder part.

We extract features from each frame using our Feature Extraction (FE) module, which combines Convolutional Neural Network (CNN) features from the frame and skeletal features

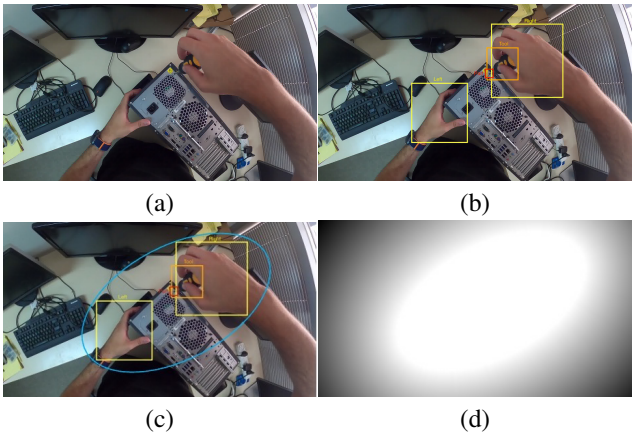

Fig. 3. Action Components:(a) The input frame, (b) The detected action components; the yellow color indicates the left and right hand, the orange and red bound the used tool and manipulated part, (c) The action region marked with a blue ellipse, and (d) The saliency mask.

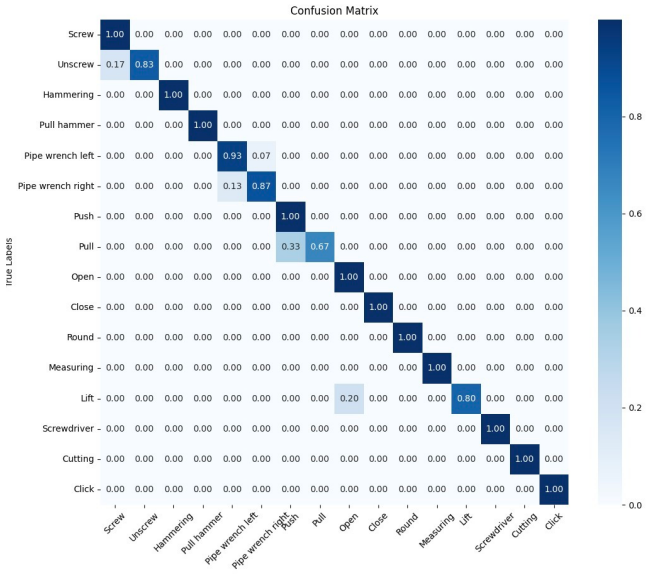

Fig. 4. The confusion matrix for sample action recognition, as seen, the performance of our model is good.

from the working hands, as shown in Figure 2(right). To compute the skeletal features, the FE module localizes the hands, identifies them (left and right), and determines the skeleton of each hand, including the relative joint position. The skeletons of the two hands are concatenated and encoded in a skeletal (SK) vector. Absent hand represented by zero in skeletal vector. Note that we assume one person performs the action; thus, at most, two hands appear in the frame. In addition, the FE module detects the tool and the manipulated part. It is not always possible to detect the manipulated part, as it can be a small screw or nut. Therefore, we assume it is near the head of the applied tool, as shown in Figure 3(b). Upon detecting the three action components, we compute the region of the action and the *saliency mask*, as illustrated in Figure 3(d). The mask is multiplied by the frame to reduce the influence of irrelevant background on the learning process. We passed the product of the mask and the frame to a CNN model for feature extraction, as illustrated in Figure 2(right).

The learning model accepts a sequence of video frames and utilizes the FE module to capture features from each one. The resulting series of feature vectors is enclosed by start and end symbols. A position embedding is added to each vector, which is fed to a standard Transformer Encoder. The output from the Transformer model passed through a Fully Connected (FC) network to discern the action, as illustrated in Figure 2(left). We adopt the standard learnable 1D position embedding, as we have not detected notable improvements in performance by employing more sophisticated position embedding.

In our current setting, the hand localization and skeleton (joints) detection models are pre-trained and frozen during the model training. We start with pre-trained CNN models of the FE, but unlike the previous models, they are not kept static (frozen). Instead, we allow them to be updated during training. We train our model end-to-end using the dataset presented in Section III.

V. Results

We have implemented our model in Python, leveraging the PyTorch library [24]. Our feature extractor module applies YOLO [12] to detect and localize the working hands. It utilizes MediaPipe [25] to detect hand landmarks (skeleton) and computes CNN features using VGG19 [26]. It is challenging to determine the object to which the action is applied. To overcome this, we use the region the tool acts on as the manipulated object's hint.

The VGG19 model to extract feature was pre-trained but was not frozen during the training of the entire model, end-to-end. We subdivide our dataset into 70% for training, 10% for testing, and 10% for validation.

We have applied video augmentation that includes changing illumination and color, blurring at various levels, rotation by small angles, and horizontal sheer by small angles. We managed to reach 300 videos for each label, where each video is represented by 100-500 frames.

We train the model end-to-end for 300 epochs, continuously improving its representations and honing its ability to distinguish between different hand actions. We evaluated our model's performance using accuracy, precision, and recall metrics. The outcomes were encouraging. The model exhibited good accuracy in recognizing hand actions, with precision and recall scores indicating its adeptness in accurately categorizing positive instances, see Figure 4. These results demonstrate the efficacy of our approach and emphasize its potential for practical applications where precise detection of manual operations holds great significance.

The confusion matrix of the detection results provides another view of the performance of our model. It is a valuable tool for visualizing and systematically evaluating the model's

predictions compared to the actual hand action labels. In addition, it provides an overview of the model's strengths and areas where it encountered difficulties. This in-depth analysis enabled us to pinpoint specific categories that were frequently misclassified, providing deeper insights into the model's behavior. Figure 4 provides the confusion matrix for a subset of the evaluated hand-actions; the detection accuracy is promising.

VI. Conclusion

We have introduced a novel dataset designed to detect hand actions involved in manipulating mechanical devices, encompassing tasks such as assembling and dismantling. In addition, we describe a novel action detection model that leverages Transformer-based architecture. Within this dataset, each entry corresponds to a first-person-view video segment capturing hands engaged in specific actions, which may involve using tools and manipulating device components. To simplify representation, these actions have been categorized into 12 distinct classes.

The deep learning model employed in this study extracts features from individual frames within the video segments, incorporating position embedding, and subsequently inputs these feature vectors into a Transformer Encoder. The resulting output vector undergoes further processing through a fully connected network to produce the final classification. Our model has been implemented and trained using the provided dataset, and we have conducted experimental evaluations, yielding promising results.

The scope of future work includes extending the dataset and including information concerning the applied tool into the input of the model.

References

[1] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[2] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1949–1957.

[3] R. Li, Z. Liu, and J. Tan, "A survey on 3d hand pose estimation: Cameras, methods, and datasets," *Pattern Recognition*, vol. 93, pp. 251–272, 2019.

[4] B. Doosti, "Hand pose estimation: A survey," *arXiv preprint arXiv:1903.01013*, 2019.

[5] A. Betancourt, M. M. López, C. S. Regazzoni, and M. Rauterberg, "A sequential classifier for hand detection in the framework of egocentric vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 586–591.

[6] Y. Zhao, Z. Luo, and C. Quan, "Coarse-to-fine online learning for hand segmentation in egocentric video," *EURASIP Journal on Image and Video Processing*, vol. 2018, pp. 1–12, 2018.

[7] S. Bambach, D. J. Crandall, and C. Yu, "Viewpoint integration for hand-based recognition of social interactions from a first-person view," in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, 2015, pp. 351–354.

[8] C. Rother, V. Kolmogorov, and A. Blake, "" grabcut" interactive foreground extraction using iterated graph cuts," *ACM transactions on graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004.

[9] X. Zhu, W. Liu, X. Jia, and K.-Y. K. Wong, "A two-stage detector for hand detection in ego-centric videos," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2016, pp. 1–8.

[10] A. Cartas, M. Dimiccoli, and P. Radeva, "Detecting hands in egocentric videos: Towards action recognition," in *Computer Aided Systems Theory–EUROCAST 2017: 16th International Conference, Las Palmas de Gran Canaria, Spain, February 19-24, 2017, Revised Selected Papers, Part II 16*. Springer, 2018, pp. 330–338.

[11] S. Jian, H. Kaiming, R. Shaoqing, and Z. Xiangyu, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision & Pattern Recognition*, 2016, pp. 770–778.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[13] R. J. Visee, J. Likitlersuang, and J. Zariffa, "An effective and efficient method for detecting hands in egocentric videos for rehabilitation applications," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, no. 3, pp. 748–755, 2020.

[14] H. Liang, J. Yuan, and D. Thalman, "Egocentric hand pose estimation and distance recovery in a single rgb image," in *2015 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2015, pp. 1–6.

[15] X. Zhu, X. Jia, and K.-Y. K. Wong, "Structured forests for pixel-level hand detection and hand part labelling," *Computer Vision and Image Understanding*, vol. 141, pp. 95–107, 2015.

[16] H. Song, W. Feng, N. Guan, X. Huang, and Z. Luo, "Towards robust ego-centric hand gesture analysis for robot control," in *2016 IEEE International Conference on Signal and Image Processing (ICSIP)*. IEEE, 2016, pp. 661–666.

[17] B. Tekin, F. Bogo, and M. Pollefeys, "H+ o: Unified egocentric recognition of 3d hand-object poses and interactions," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4511–4520.

[18] A. Betancourt, P. Morerio, L. Marcenaro, E. Barakova, M. Rauterberg, and C. Regazzoni, "Towards a unified framework for hand-based methods in first person vision," in *2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE, 2015, pp. 1–6.

[19] S. Singh, C. Arora, and C. Jawahar, "First person action recognition using deep learned descriptors," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2620–2628.

[20] S. Urabe, K. Inoue, and M. Yoshioka, "Cooking activities recognition in egocentric videos using combining 2dcnn and 3dcnn," in *Proceedings of the Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management*, 2018, pp. 1–8.

[21] Y. Tang, Z. Wang, J. Lu, J. Feng, and J. Zhou, "Multi-stream deep neural networks for rgb-d egocentric action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 10, pp. 3001–3015, 2018.

[22] X. Zhu, X. Jia, and K.-Y. K. Wong, "Pixel-level hand detection with shape-aware structured forests," in *Computer Vision–ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part IV 12*. Springer, 2015, pp. 64–78.

[23] Y. Zhou, B. Ni, R. Hong, X. Yang, and Q. Tian, "Cascaded interactional targeting network for egocentric video analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1904–1913.

[24] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[25] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. Yong, J. Lee *et al.*, "Mediapipe: A framework for perceiving and processing reality," in *Third workshop on computer vision for AR/VR at IEEE computer vision and pattern recognition (CVPR)*, 2019.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.