Understanding Human Hands in Visual Data

Ph.D. Thesis Proposal presented

by

Supreeth Narasimhaswamy

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

October 2022

Abstract of the Thesis Proposal

Understanding Human Hands in Visual Data

by

Supreeth Narasimhaswamy

Doctor of Philosophy

in

Computer Science

Stony Brook University

2022

Hands are the central means by which humans interact with their surroundings. Understanding human hands help human behavior analysis and facilitate other visual analysis tasks such as action and gesture recognition. Recently, there has been a surge of interest in understanding first-person visual data [29], and hands are the dominant interaction entities in such activities. Also, there is an explosion of interest in developing computer vision methods for augmented and virtual reality. To deliver an authentic augmented and virtual reality experience, we need to enable humans to interact with the virtual world and allow virtual avatars to communicate and interact with each other. Since hands are the dominant interaction entities in such cases, a thorough understanding of human hands is essential in developing computer vision methods for augmented and virtual reality.

The first step toward the visual understanding of human hands is to detect hands in images or videos. Localizing hands in the wild are challenging since numerous hands can be present, and there can be tremendous occlusions between hands, objects, and people. To address these issues, we propose a contextual attention method [60] to detect hands.

While it is essential to detect hands, this is not sufficient for a more fine-grained semantic understanding of hands. When humans interact

with their surroundings, their hands contact objects and other humans around them. Therefore we need to understand their contact information to have a more meaningful understanding of hands. To address this, we propose to study hand contact recognition [61].

To understand how human hands interact with the world, we need to know how hands move across *time*. In other words, we need to *track* hands in videos. While there are hand-tracking methods, they are mostly developed for tracking only one or two hands. Tracking more than two hands is challenging in unconstrained conditions. To address this, we study the problem of tracking more than two hands [38].

While it is important to detect, track and obtain contact states of hands for detailed activity understanding, this is not sufficient. For a scene containing multiple people, we need to know what object is manipulated by whom and which person is performing what activity. In other words, we must detect hands and localize the corresponding person simultaneously. To tackle this, we study the hand-body association in images [62].

Contents

List of Figures

List of Tables

Publications

The methods being presented in this thesis proposal have appeared in the following publications:

1. Roy Shilkrot, Supreeth Narasimhaswamy, Saif Vazir, Minh Hoai, WorkingHands: A Hand-Tool Assembly Dataset for Image Segmentation and Activity Mining, Proceedings of the British Machine Vision Conference (BMVC), 2019

2. Supreeth Narasimhaswamy*, Zhengwei Wei*, Yang Wang, Justin Zhang, Minh Hoai, Contextual Attention for Hand Detection in the Wild, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

3. Supreeth Narasimhaswamy, Trung Nguyen, Minh Hoai, Detecting Hands and Recognizing Physical Contact in the Wild, Advances in Neural Information Processing Systems (NeurIPS), 2020.

4. Mingzhen Huang, Supreeth Narasimhaswamy, Saif Vazir, Haibin Ling, Minh Hoai, Forward Propagation, Backward Regression and Pose Association for Hand Tracking in the Wild, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

5. Supreeth Narasimhaswamy, Thanh Nguyen, Mingzhen Huang, Minh Hoai, Whose Hands Are These? Hand Detection and Hand-Body Association in the Wild, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

Chapter 1

Introduction

We use our hands for most of our day-to-day interactions with our surroundings. Therefore understanding human hands helps human behavior analysis and facilitates other visual analysis tasks such as action and gesture recognition. For example, if we can detect hands and their interaction object, the guitar, we can guess that the person in Fig. 1.1a is *playing guitar*. Similarly, by detecting hands, the knife, and the vegetable in Fig. 1.1b we can recognize that the person is *chopping vegetables*.

(a) A person playing guitar (b) A person chopping vegetables

Figure 1.1: Detecting hands and their interaction objects can shed light on the ongoing activity.

Recently, there has been an explosion of interest in developing computer vision methods for augmented and virtual reality. To deliver an authentic augmented and virtual reality experience, we need to enable hu-

mans to interact with the virtual world and allow virtual avatars to communicate and interact with each other. Since hands are the dominant interaction entities in such cases, a thorough understanding of human hands is essential in developing computer vision methods for augmented and virtual reality. Fig. 1.2 shows two augmented reality scenarios. In the first image, a doctor is demonstrating surgery using holograms. In the second image, a person interacts with a virtual cube using their hand. In both cases, tracking hands and their interaction with objects is essential.

Figure 1.2: Hand tracking and hand-object interaction are essential for Augmented and Virtual Reality.

A long-standing goal of robotics is to build robots that can perform activities similar to humans. Grasping and manipulating objects is one of the critical functions required by robots. Fig. 1.3 shows a robot arm trying to grasp an object and the human counterpart grasping the min. By learning the hand pose, hand and ball contact areas, and relative poses between the human hand and the tennis ball, we can design a robotic arm to grasp the tennis ball.

Figure 1.3: Hand-object grasping helps develop robotic object grasping.

In this thesis proposal, we study human hands in the context of computer vision. Specifically, we address four critical problems: hand detection, hand contact recognition, hand tracking, and hand-body association. We will describe these problems in the following. We will conclude the thesis proposal by identifying some future directions.

1.1 Hand Detection

The first step toward the visual understanding of human hands is to detect hands in images or videos. The hand detection can be a simple rectangular bounding box enclosing the hand region or a more precise pixel-wise binary segmentation hand mask.

The computer vision community has well-studied hand detection. Early works mostly used skin color to detect hands [17, 99, 114], or boosted classifiers based on shape features [44, 64]. Later on, context information from human pictorial structures was also used for hand detection [12, 42, 45]. Mittal et al. [54] proposed to combine shape, skin, and context cues to build a multi-stage detector. Saliency maps have also been used for hand detection [67]. However, the performance of these methods on unconstrained images is poor, possibly due to the lack of access to deep learning and powerful feature representations.

Recent works are based on Convolutional Neural Networks (CNNs). Hoang Ngan Le et al. [35] proposed a multi-scale FasterRCNN method to avoid missing small hands. Roy et al. [75] proposed to combine FasterRCNN and skin segmentation. Duan et al. [22] proposed a framework based on pictorial structure models to detect and localize hand joints from depth images. Deng et al. [21] proposed a CNN-based method to detect hands and estimate the orientations jointly. While deep learning based have better hand detection performance than traditional machine learning methods, its performance is still quite poor; they mistake other skin areas for hands and fail to detect the low resolution and occluded hands. One possible reason is the lack of a large-scale dataset to train hand detectors and the lack of a mechanism to differentiate between hand and non-hand skin areas.

In Chapter 2, we will address hand detection in unconstrained images. We propose a contextual attention method aggregating useful hand detection features from contextually important image regions.

1.2 Hand Contact Recognition

While it is essential to detect hands, this is not sufficient for a more fine-grained semantic understanding of hands. When humans interact with their surroundings, their hands contact objects and other humans around them. Therefore we need to understand their contact information to have a more meaningful understanding of hand interactions. While understanding contact is more meaningful using three-dimensional distances between hands, human or object meshes, obtaining 3D meshes for images in the wild is a tricky problem. We, therefore, propose to address this problem in two dimensions using images, the classification of hand contact states into (1) No-Contact, (2) Self-Contact, (3) Person-Contact, and (4) Object-Contact. Hand contact recognition in two dimensions is still a useful semantic understanding problem and can serve as a weak signal for understanding hand contact in three dimensions.

There are prior works for hand detection [12, 21, 42, 44, 45, 54, 64, 75, 79, 99, 114] and hand pose estimation [30, 74, 81, 116, 117]. However, they do not study hand contact state recognition. One way to estimate the contact state of a hand is to reason based on its estimated pose. However, obtaining a reasonably good hand pose in unconstrained conditions is challenging. For example, obtaining the hand pose in low-resolution visual data such as surveillance images in a supermarket or an elevator is highly difficult. Therefore recognizing the physical contact states of hands using their pose is not a robust approach.

Some works consider hand-object interactions. Hasson et al. [30] propose reconstructing hand and object meshes from monocular images. Bambach et al. [6] aims to locate hands interacting with an object, focusing on first-person videos containing only two people. Moreover, most of the activities focus on hands playing cards or puzzles. Tekin et al. [86] propose to model hand-object interactions by jointly estimating the 3-D poses for hands and objects. However, none of these works estimate the physical contact state of the hand.

In Chapter 3, we address a novel problem of recognizing hand contact states. We develop a convolutional network architecture that can jointly detect hands and recognize their physical contact states.

1.3 Hand Tracking

To understand how human hands interact with the world, we need to know how hands move across *time*. For example, when a person is trying to wash dishes, their hands first approach the washing soap, open the soap bottle, pour the soap onto a cleaning sponge, and finally approach the utensils to clean them. In other words, this task requires us to understand where our hands move and what objects they interact with over time. Essentially, we need to *track* hands across different video frames.

Sridhar et al. [82] proposed a method to track hands captured using a depth camera. Zhang et al. [108] proposed a hand tracking solution that predicted a hand skeleton of a human from a single RGB camera for AR/VR applications. Wang and Popović [90] used a single camera to track a gloved hand with an imprinted pattern. Sharp et al. [78] provided a hand tracking and pose estimation system based on a single depth camera. Mueller et al. [57] developed a 3D hand tracking approach for monocular RGB videos using a kinematic 3D hand model. Sridhar et al. [82] proposed a method to track hands manipulating objects in RGB-D videos. However, none of these methods was developed for videos in the wild; they required unique markers, depth information, ego-centric perspectives, or scenes with plain backgrounds.

While there has been a significant effort in studying multiple object tracking problems, such as pedestrian tracking, such methods do not work well for tracking hands in the wild. Hand tracking is difficult because hands are highly deformable objects, move fast, disappear, get occluded, or even move past other hand instances. A hand instance's shape, size, and appearance change drastically over time. Simultaneously, two different hand instances can look alike, so distinguishing them would be difficult even for a sophisticated re-identification module explicitly trained for hands.

In Chapter 4, we propose an online hand tracking method that can simultaneously detect and track hands in unconstrained videos.

1.4 Hand-Body Association

While it is essential to detect, track and obtain contact states of hands for detailed activity understanding, this is not sufficient. For a scene con-

taining multiple people, in addition to localizing hands, we also need to know what object is manipulated by whom and which person is performing what activity. In other words, we must detect hands and localize the corresponding person simultaneously; that is, we need to perform hand-body association. There has been a surge in interest [2, 84] in constructing full-body three-dimensional human poses from partial observations, and hand-body association from images can provide good starting features to tackle this problem in three dimensions.

Associating hands with bodies is challenging since numerous people can be in the scene with varying degrees of overlap and occlusion. One possible approach to associate hands with their bodies is to detect hands and bodies separately and then perform an association between them using heuristics such as distances or overlaps between them. However, these methods do not work well since there can be tremendous variation in relative scales between hands and bodies. Furthermore, one person's hand can completely overlap with another person's body. Another possible approach is to use human pose estimation methods to detect body joints and hand key points and subsequently link hands and bodies. However, pose detection is unreliable for a scene containing multiple interacting people for several reasons. First, pose methods cannot detect hand key points for every visible hand. Second, pose-based methods might not detect body joints in occluded people. Third, even if the joints are detected, the human skeleton of one person can be mixed with another person.

In Chapter 5, we study a novel problem associating hands and bodies in unconstrained images and propose a convolutional neural network architecture to detect hand locations and their corresponding person locations jointly.

1.5 Organization of this Thesis

The rest of this thesis is organized as follows. The next chapter provides details about our hand detection method. Chapter 3 describes our studies on the novel problem of hand contact recognition. Chapter 4 details our work on tracking multiple hands in videos. Chapter 5 presents our investigation of the novel problem of hand-body association. Chapter 6 discusses some directions for future studies.

Chapter 2

Hand Detection

In this chapter, we address the hand detection problem in unconstrained images. We introduce Hand-CNN, a novel convolutional network architecture for detecting hand masks and predicting hand orientations. Hand-CNN extends MaskRCNN with a novel attention mechanism to incorporate contextual cues in the detection process. This attention mechanism can be implemented as an efficient network module that captures non-local dependencies between features. This network module can be inserted at different stages of an object detection network, and the entire detector can be trained end-to-end.

We also introduce large-scale annotated hand datasets containing hands in unconstrained images for training and evaluation. We show that Hand-CNN outperforms existing methods on the newly collected datasets and the publicly available PASCAL VOC human layout dataset [23].

2.1 Introduction

People use their hands to interact with each other and the environment, and most human actions and gestures can be determined by the location and motion of their hands. As such, detecting hands reliably in images and videos will facilitate many visual analysis tasks, including gesture and action recognition. Unfortunately, detecting hands in unconstrained conditions is challenging due to the tremendous variety of hands in images. Hands are highly articulated, appearing in various orientations, shapes, and sizes. Occlusion and motion blur further increase variations in the

Figure 2.1: **Hand detection in the wild**. We propose Hand-CNN, a novel network for detecting hand masks and estimating hand orientations in unconstrained conditions.

appearance of hands.

Hands can be considered a generic object class, and an appearance-based object detection framework such as DPM [25] and MaskRCNN [32] can be used to train a hand detector. However, an appearance-based detector would have difficulty detecting hands with occlusion and motion blur. Another approach for detecting hands is to consider them as a part of a human body and determine the locations of the hands based on the detected human pose. Pose detection, however, does not provide a reliable solution, especially when several human body parts are not visible (e.g., in TV shows, the lower body is frequently not contained in the image frame).

To address hand detection, we propose Hand-CNN, a novel CNN architecture to detect hand masks and predict hand orientations. Hand-CNN is founded on MaskRCNN [32], with a novel attention module to incorpo-

rate contextual cues during the detection process. The proposed attention module is designed for non-local contextual pooling: one based on feature similarity and the other on the spatial relationships between semantically related entities. Intuitively, a region is more likely to be a hand if there are other regions with similar skin tones, and the location of a hand can be inferred by the presence of other semantically related body parts such as the wrist and elbow. The contextual attention module encapsulates these two types of non-local contextual pooling operations. These operations can be performed efficiently with a few matrix multiplications and additions, and the parameters of the attention module can be learned together with other parameters of the detector end-to-end. The attention module as a whole can be inserted in already existing detection networks. This illustrates the generality and flexibility of the proposed attention module.

Finally, we address the lack of training data by collecting and annotating two large-scale hand datasets. Since annotating many images is a laborious process, we develop a method to semi-automatically annotate most of the data, and we only manually annotate a portion of the data. Altogether, the newly collected data contains more than 35K images with around 54K annotated hands. This data can be used for developing and evaluating hand detectors.

2.2 Related Work

There exist several algorithms for hand detection. Early works mostly used skin color to detect hands [17, 99, 114], or boosted classifiers based on shape features [44, 64]. Later on, context information from human pictorial structures was also used for hand detection [12, 42, 45]. Mittal et al. [54] proposed to combine shape, skin, and context cues to build a multi-stage detector. Saliency maps have also been used for hand detection [67]. However, the performance of these methods on unconstrained images is poor, possibly due to the lack of access to deep learning and powerful feature representation.

Recent works are based on Convolutional Neural Networks (CNNs). Hoang Ngan Le et al. [35] proposed a multi-scale FasterRCNN method to avoid missing small hands. Roy et al. [75] proposed to combine FasterRCNN and skin segmentation. Duan et al. [22] proposed a framework based on pictorial structure models to detect and localize hand joints from

depth images. Deng et al. [21] proposed a CNN-based method to detect hands and estimate the orientations jointly. While deep learning based have better hand detection performance than traditional machine learning methods, its performance is still quite poor; they mistake other skin areas for hands and fail to detect the low resolution and occluded hands. One possible reason is the lack of a large-scale dataset to train hand detectors and the lack of a mechanism to differentiate between hand and non-hand skin areas.

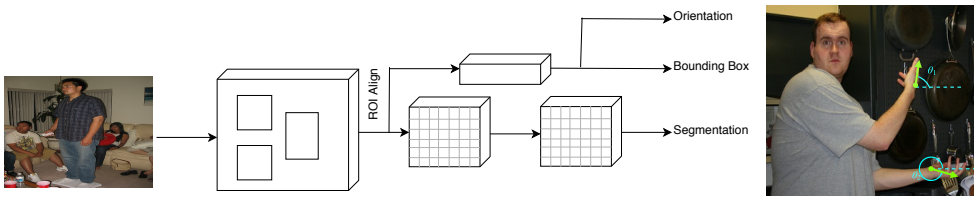
The contextual attention module for hand detection developed in this paper shares some similarities with some recently proposed attention mechanisms, such as Non-local Neural Networks [91], Double Attention Networks [16], and Squeeze-and-Excitation Networks [37]. These attention mechanisms, however, are designed for image and video classification instead of object detection. They do not consider spatial locality, but locality is essential for object detection. Furthermore, most of them are defined based on similarity instead of semantics, ignoring the contextual cues obtained by reasoning about the spatial relationship between semantically related entities.

2.3 Hand-CNN

Hand-CNN is developed from MaskRCNN [32], with an extension to predict the hand orientation, as depicted in Fig. 2.2a. Hand-CNN also incorporates a novel attention mechanism to capture the non-local contextual dependencies between hands and other body parts.

2.3.1 Hand Mask and Orientation Prediction

Our detection network is founded on MaskRCNN [32]. MaskRCNN is a robust state-of-the-art object detection framework with multiple stages and branches. It has a Region Proposal Network (RPN) branch to identify the candidate object bounding boxes, a Box Regression Network (BRN) branch to pull features inside each proposal region for classification and bounding box regression, and a branch for predicting the binary segmentation of the detected object. The binary mask is better than the bounding box at delineating the object's boundary, but neither the mask nor the bounding box encodes the object's orientation.



(a) (b)

Figure 2.2: **Processing pipeline of Hand-CNN, and Hand Orientation illustration.** (a): An input image is fed into a network for bounding box detection, segmentation, and orientation estimation. The Hand-CNN extends the MaskRCNN to predict the hand's orientation by an additional branch. The Hand-CNN also has a novel attention mechanism. This attention mechanism is implemented as a modular block and is inserted before the RoIAlign layer. (b): The green arrows denote vectors connecting the wrist and the center of the hand. The cyan dotted lines are parallel to the x-axis, θ_1 and θ_2 denote orientation angles for the right hand and left hand of the person, respectively.

We extend MaskRCNN to include an additional network branch to predict hand orientation. Here, we define the hand's orientation as the angle between the horizontal axis and the vector connecting the wrist and the center of the hand mask (see Fig. 2.2b). The orientation branch shares weights with the other branches, so it does not incur significant computational expenses. Moreover, the shared weights slightly improved the performance in our experiments.

The entire hand detection network with mask detection and orientation prediction can be jointly optimized by minimizing the combined loss function $L = L_{RPN} + L_{BRN} + L_{mask} + \lambda L_{ori}$. Here, $L_{RPN}, L_{BRN}, L_{mask}$ are the loss functions for the region proposal network, the bounding box regression network, and the mask prediction network, as described in [32, 72]. In our experiments, we use the default weights for these loss terms, as specified in [32]. L_{ori} is the loss for the orientation branch, defined as:

$$L_{ori}(\theta, \theta^*) = |arctan2(sin(\theta - \theta^*), cos(\theta - \theta^*))|. \tag{2.1}$$

In the above, θ and θ^* are the predicted and ground truth hand orientations (the angle between the x-axis and the vector connecting the wrist and the center of the hand, see Fig. 2.2b). We use the above loss function

instead of the simple absolute difference between θ and θ^* to avoid the modular arithmetic problem of the angle space (i.e., 359° is close to 1° in the angle space, but the absolute difference is significant). Weight λ is a tunable parameter for the orientation loss, which was set to 0.1 in our experiments.

2.3.2 Contextual Attention Module

Hand-CNN has a novel attention mechanism to incorporate contextual cues for detection. Consider a three dimensional feature map $\mathbf{X} \in \mathbb{R}^{h \times w \times m}$, where h, w, m are the height, width, and the number of channels. For a spatial location i of the feature map $\mathbf{X}$, we will use $\mathbf{x}_i$ to denote the m dimensional feature vector at that location. Our attention module computes a contextual feature map $\mathbf{Y} \in \mathbb{R}^{h \times w \times m}$ of the same size as $\mathbf{X}$. The contextual feature vector $\mathbf{y}_i$ for location i is computed as:

$$\mathbf{y}_i = \sum_{j=1}^{hw} \left[\frac{f(\mathbf{x}_i, \mathbf{x}_j)}{C(\mathbf{x}_i)} + \sum_{k=1}^{K} \alpha_k \, p_k(\mathbf{x}_j) \, h_k(d_{ij}) \right] g(\mathbf{x}_j).$$

This contextual vector is the sum of contextual information from all locations j's of the feature map. The contextual contribution from location j toward location i is determined by several factors as explained below.

Similarity Context. One type of contextual pooling is based on non-local similarity. In the above formula, $f(\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{R}$ is a measure for the similarity between feature vectors $\mathbf{x}_j$ and $\mathbf{x}_i$. $C(\mathbf{x}_i)$ is the normalizing factor: $C(\mathbf{x}_i) = \sum_j f(\mathbf{x}_i, \mathbf{x}_j)$. Thus $\mathbf{x}_j$ provides more contextual support to $\mathbf{x}_i$ if $\mathbf{x}_j$ is more similar to $\mathbf{x}_i$. Intuitively, a region is more likely to be a hand if other regions have similar skin tones, and a region is less likely to be a hand if there are non-hand areas with a similar texture. Similarity pooling can provide contextual information to increase or decrease the probability that a region is a hand.

Semantics Context. Similarity pooling, however, does not take into account semantics and spatial relationship between semantically related entities [34]. The second type of contextual pooling is based on the intuition that the location of a hand can be inferred by the presence and locations of other body parts, such as the wrist and elbow. We consider having K

(body) part detectors, and $p_k(\mathbf{x}_j)$ denotes the probability that $\mathbf{x}_j$ belongs to part category k (for $1 \leq k \leq K$). The variable d_{ij} denotes the L_2 distance between positions i and j, and $h_k(d_{ij})$ encodes the probability that the distance between a hand and a body part of category k is d_{ij}. We model this probability using a Gaussian distribution with mean μ_k and variance σ_k^2. Specifically, we set: $h_k(d_{ij}) = \exp\left(-\frac{(d_{ij}-\mu_k)^2}{\sigma_k^2}\right)$. Some part categories provide more informative contextual cues for hand detections than other categories, so we use the scalar variable α_k ($0 \leq \alpha_k \leq 1/K$) to indicate the contextual importance of category k. The variables α_k's, μ_k's, and σ_k's are automatically learned.

The functions f, g, and p_k's are also learnable. We parameterize them as follows:

$$f(\mathbf{x}_i, \mathbf{x}_j) = \exp\left((\mathbf{W}_\theta \mathbf{x}_i)^T \left(\mathbf{W}_\phi \mathbf{x}_j\right)\right), \tag{2.2}$$

$$g(\mathbf{x}_j) = \mathbf{W}_g \mathbf{x}_j, \quad p(\mathbf{x}_j) = \text{softmax}(\mathbf{W}_p \mathbf{x}_j), \tag{2.3}$$

where $\mathbf{W}_\theta, \mathbf{W}_\phi, \mathbf{W}_g \in \mathbb{R}^{m \times m}$ and $\mathbf{W}_p \in \mathbb{R}^{K \times m}$. We set $p_k(\mathbf{x}_j)$ as k^{th} element of $p(\mathbf{x}_j)$. The above matrix operations involving $\mathbf{W}_\theta$, $\mathbf{W}_\phi$, $\mathbf{W}_g$, and $\mathbf{W}_p$ can be implemented efficiently using 1×1 convolutions. Together with μ_k's, σ_k's, and α_k's, these matrices are the learnable parameters of our attention module.

2.4 Hand Detection Datasets

We aim to train a hand detector that can detect all occurrences of hands in images, regardless of their shapes, sizes, orientations, and skin tones. Unfortunately, there was no existing training dataset that was large and diverse enough for this purpose, so we collected and annotated some data ourselves. The data consists of two parts. Part I contains image frames extracted from video clips of the ActionThread dataset [33]. Part II is a subset of the Microsoft COCO dataset [49]. Images from Part I were manually annotated by us, while the annotations for Part II were automatically derived based on a hand pose detection algorithm and the existing wrist annotations of the COCO dataset. We refer to Part I as the TV-Hand dataset and Part II as the COCO-Hand dataset.

2.4.1 TV-Hand Dataset

Data source. The TV-Hand dataset contains 9498 image frames extracted from the ActionThread dataset [33]. The ActionThread dataset consists of video clips from various TV series for human actions. We chose ActionThread as the data source because of several reasons. Firstly, we want images with multiple hand occurrences, as is likely with video frames from human action samples. Secondly, TV series are filmed from multiple camera perspectives, allowing for hands in various orientations, shapes, sizes, and relative scales (i.e., hand size compared to other body parts such as the face and arm). Thirdly, we are interested in detecting hands with motion blur, and video frames contain better training examples than static photographs in this regard. Fourthly, hands are not usually the main focus of attention in TV series, so they appear naturally with various levels of occlusion and truncation (compared to other types of videos such as sign language or egocentric videos). Lastly, a video-frame hand dataset will complement COCO and other datasets compiled from static photographs.

Video frame extraction. Video frames were extracted from videos of the ActionThread dataset [33]. This dataset contains a total of 4757 videos. Of these videos, 1521 and 1514 are training and test data for the action recognition task; the remaining videos are ignored. For the TV-Hand dataset, we extracted frames from all videos. Given a video from the ActionThread dataset, we first divided it into multiple shots using a shot boundary detector. Among the video shots that were longer than one second, we randomly sampled one or two shots. For each selected shot, the middle frame of the shot was extracted and subsequently included in the TV-Hand dataset. Thus, the TV-Hand dataset includes one to two frames from each video.

We divided the TV-Hand dataset into the train, validation, and test subsets. To minimize the dependency between the data subsets, we ensured that images from a given video belonged to the same subset. The training data contains images from 2433 videos, the validation data from 810 videos, and the test set from 1514 videos. All test images are extracted from the test videos of the ActionThread dataset. This ensures that the train and test data come from disjoint TV series, furthering the independence between these two subsets. Altogether, the TV-Hand dataset contains 9498 images. Of these images, 4853 are used as training data, 1618

Figure 2.3: **Some sample images with annotated and unannotated hands from the TV-Hand dataset.** Annotators were asked to draw a quadrilateral for any visible hand region larger than 100 pixels, regardless of the amount of truncation and occlusion. Annotators also identified the side of the quadrilateral that connects to the arm (yellow sides in this figure). This is a challenging dataset where hands appear at multiple locations, having different shapes, sizes, and orientations. Severely occluded and blurry hands are also present. The blue boxes are some instances that were not annotated.

as validation data, and 3027 as test data.

Notably, all videos from the ActionThread dataset are normalized to have a height of 360 pixels and a frame rate of 25fps. As a result, the images in the TV-Hand dataset all have a height of 360 pixels. The widths of the images vary to keep their original aspect ratios.

Annotation collection. This dataset was annotated by three annotators. Two were asked to label two different parts of the dataset, and the third annotator was asked to verify and correct any annotation mistakes. The annotators were instructed to localize every hand that occupies more than 100 pixels. We used the threshold of 100 pixels so that the dataset would be consistent with the Oxford Hand dataset [54]. Because it is difficult to visually determine if a hand region is larger than 100 pixels in practice, this served as an approximate guideline: our dataset contains several hands that are smaller than 100 pixels. Truncation, occlusion, and self-occlusion were not considered; the annotators were asked to identify truncated and occluded hands as long as the visible hand areas were more than 100 pixels. To identify the hands, the annotators were asked to draw a quadrilateral box for each hand, aiming for a tight box that contained as many hand pixels as possible. This was not a precise instruction and led to subjective decisions in many cases. However, there was no better alternative. One option is providing a pixel-level mask, which would require

enormous human effort. Another option is to annotate the axis-parallel bounding box for the hand area. But this annotation type provides poor hand localization due to its highly articulate nature. Given the annotation effort, we found that a quadrilateral box had the highest annotation quality. In addition to the hand bounding box, we also asked the annotators to identify the side of the quadrilateral that corresponds to the direction of the wrist/arm. In the TV-Hand dataset, Fig. 2.3 shows some examples of annotated and unannotated hands.

The total number of annotated hands in the dataset is 8646. The number of hands in train, validation and test sets are 4085, 1362, and 3199, respectively. Half of the data contains no hands, and a large proportion contains one or two hands. The most significant number of hands in one image is 9. Roughly fifty percent of the hands occupy an area of 1000 square pixels or fewer. 1000 pixels corresponds to a 33×33 square, and it is relatively small compared to the image size (recall that all images have a height of 360 pixels).

2.4.2 COCO-Hand Dataset

In addition to TV-Hand, we propose to use images from the Microsoft's COCO dataset [49]. COCO is a dataset containing common objects with various annotations, including segmentations and keypoints. The many images containing people and annotated joint locations are most useful for us. However, the COCO dataset does not contain bounding boxes or segmentation annotations for hands, so we propose an automatic method to infer them for a subset of the images where we can confidently do so.

Our objective here is to automatically generate non-axis aligned rectangles for hands in the COCO dataset so that they can subsequently be used as annotated examples to train a hand detection network. This process requires running a hand *keypoint* detection algorithm (to detect wrist and finger joints) and uses a conservative heuristic to determine if the detection is reliable. Specifically, we used the hand keypoint detection algorithm of [80], which was trained on a multiview dataset of hands and annotated finger joints. This algorithm worked well for many cases but produced many wrong detections. We used the following heuristics to determine the validity of detection as follows (see also Fig. 2.4).

1. Identify the predicted wrist location, called $\mathbf{w}_{pred}$

2. Calculate the average of the predicted hand keypoints, called $\mathbf{h}_{avg}$.

3. Considering $\mathbf{h}_{avg} - \mathbf{w}_{pred}$ as the direction of the hand, determine the minimum bounding rectangle aligned with this direction and contains the predicted wrist and all hand keypoints.

4. Calculate length L of the rectangle side parallel to the hand direction.

5. Compute the error between the predicted wrist location $\mathbf{w}_{pred}$ and the *closest* annotated wrist location $\mathbf{w}_{gt}$, $E = \|\mathbf{w}_{pred} - \mathbf{w}_{gt}\|_2$.

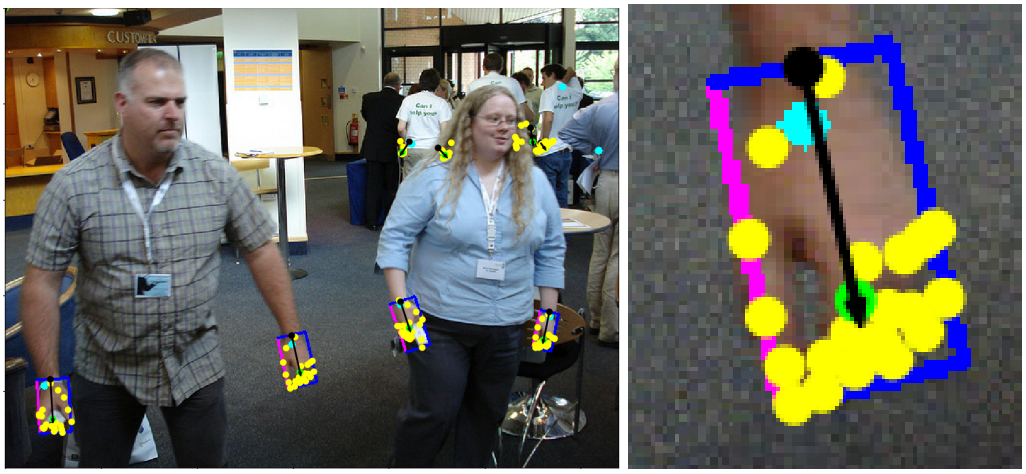
6. Discard a detected hand if the error (relative to the size of the hand) is greater than 0.2 (chosen empirically), i.e., discard a detection if $E/L > 0.2$.

The COCO dataset also has annotations for the visibility of hands; we used them to discard occluded hands. We ran the detection algorithm on 82,783 COCO images and detected 161,815 hands. The average area of the bounding rectangles is 977 pixels. Of these detections, our conservative heuristics determined 113,727 detections unreliable. A total of 48,008 detections survived to the next step.

The above heuristics can reject false positives but cannot retrieve missed detections (false negatives). Unfortunately, using images with missed detections can hurt the training of the hand detector because a hand area might be deemed a negative training example. Meanwhile, hand annotation is precious, so an image with at least one true positive detection should not be discarded. We, therefore, propose to keep images with true positives but mask out the undetected hands using the following heuristics (see also Fig. 2.5).

1. For each undetected hand, we add a circular mask of radius $r = \|\mathbf{w}_{gt} - \mathbf{e}_{gt}\|_2$ centered at $\mathbf{w}_{gt}$, where $\mathbf{w}_{gt}$ and $\mathbf{e}_{gt}$ denote the wrist and elbow keypoint locations, respectively, as provided by the COCO dataset. We set the pixel intensities inside the masks to 0.

2. Discard an image if there is any overlap between any mask and any correctly detected hands (true positives).



(a) (b)

Figure 2.4: **Heuristics for discarding bad detection on COCO.** (a): the hand keypoint algorithm is run to detect hands. The left hand of the man on the left is shown in (b). (b): black dot: predicted wrist $\mathbf{w}_{pred}$; cyan dot: closest annotated wrist $\mathbf{w}_{gt}$; yellow dots: predicted keypoints; green dot: center of the predicted keypoints $\mathbf{h}_{avg}$; blue-magenta box: smallest bounding rectangle for the hand keypoints; magenta side is the side of the rectangle that is parallel to the predicted hand direction, its length is L. We consider a detection unreliable if the distance between the predicted wrist and the closest annotated wrist is more than 20% of L.

(a) (b)

Figure 2.5: **Heuristics for masking missed detections on COCO.** (a): the hand keypoint algorithm failed to detect the left hand of the man. (b): A black circular mask centered at the wrist is added. The radius is determined based on the distance between the wrist and the elbow keypoints.

Applying the above procedures and heuristics, we obtained the COCO-Hand dataset with 26,499 images totaling 45,671 hands. Additionally, we perform a final verification step to identify images with good and complete annotations. This subset has 4534 images with 10,845 hands, and we refer to it as COCO-Hand-S. The more extensive COCO dataset is referred to as COCO-Hand.

2.4.3 Comparison with other datasets

There exist several hand datasets, but most existing datasets were collected in the lab environments, captured by a specific type of camera, or developed for specific scenarios, as shown in Table 2.1. We are, however, interested in developing a hand detection algorithm for unconstrained images and environments. To this end, only the Oxford Hand dataset is similar to ours. However, this dataset is much smaller than the datasets we collected.

Name	Scope	# images	Label
EgoHands [6]	Google glasses	4,800	Manual
Handseg [53]	Color gloves	210,000	Auto
NYUHands [87]	Three subjects	6,736	Auto
WorkingHands [79]	Three subjects	7,905	Man.+Syn.
ColorHandPose [116]	Specific poses	43,986	Synthetic
HandNet [95]	Ten subjects	212,928	Auto
GTEA [48]	Four subjects	663	Manual
Oxford-Hand [54]	Unconstrained	2686	Manual
TV-Hand	Unconstrained	9498	Manual
COCO-Hand-S	Unconstrained	4534	Semiauto
COCO-Hand	Unconstrained	26499	Semiauto

Table 2.1: **Comparison with other hand datasets.**

2.5 Experiments

In this section, we describe experiments on hand detection and orientation prediction. We evaluate the performance of Hand-CNN on test sets of the TV-Hand dataset and the Oxford Hand dataset. We do not evaluate the performance of the COCO-Hand dataset due to the absence of manual annotations. For a better cross-dataset evaluation, we do not train or fine-tune our detectors on the train data of the Oxford-Hand dataset. We only use the test data for evaluation. The Oxford-Hand test data contains 821 images with a total of 2031 hands.

2.5.1 Details about the training procedure

We trained Hand-CNN and MaskRCNN starting from the GitHub code of Abdulla [1]. To train a MaskRCNN detector, we initialized it with a publicly available ResNet101-based MaskRCNN model trained on Microsoft COCO data. This was also the initialization method for the MaskRCNN component of Hand-CNN. The contextual attention module was inserted right before the last residual block in stage 4 of ResNet101, and the weights were initialized with the Xavier-normal initializer.

Method	AP
DPM [28]	36.8%
ST-CNN [40]	40.6%
RCNN [27]	42.3%
Context + Skin [54]	48.2%
RCNN + Skin [75]	49.5%
FasterRCNN [72]	55.7%
Rotation Network [21]	58.1%
Hand Keypoint [80]	68.6%
Hand-CNN (proposed)	**78.8**%

Table 2.2: **Comparison of the state-of-the-art** hand detection algorithms on the Oxford-Hand dataset.

2.5.2 Hand Detection Performance

We used the TV-Hand dataset and COCO-Hand to train a Hand-CNN. Table 2.2 compares the performance of Hand-CNN with the previous state-of-the-art methods on the publicly available Oxford-Hand data test set. We measure performance using Average Precision (AP), an accepted standard for object detection [23]. To be compatible with the previously published results, we use the exact evaluation protocol and evaluate the performance based on the intersection over the union of the axis-aligned predicted and annotated bounding boxes. As can be seen, Hand-CNN outperforms the best previous method by a wide margin of 10% on the absolute scale. This impressive result can be attributed to 1) the novel contextual attention mechanism and 2) the use of a large-scale training dataset. Next, we will perform ablation studies to analyze the benefits of these two factors.

Comparison to a heuristic based on 2D body pose. Given the success of 2D body pose keypoint estimation methods, one might wonder if we can detect hands by simply extending the direction from elbow to wrist and guessing the extended vector from the wrist as the hand part. To compare with this heuristic baseline, we used [80] to obtain keypoints for elbows and wrists and extend the vector from the elbow to the wrist to find the center of the hand. Suppose the elbow and the wrist distance is R; we set

the extended distance to αR, with α being a controllable parameter. The spatial extension of the hand is heuristically defined as a circular region with radius αR. Table 2.3 reports the APs of this method on Oxford data for various values of α, which are much lower than the AP of the Hand-CNN.

α	0.05	0.1	0.2	0.4	0.8	1.2	1.6
AP	28.27%	30.41%	33.56%	33.91%	24.22%	14.18%	9.29%

Table 2.3: **AP of the heuristic baseline**. The table reports the results on Oxford data as a function of the parameter α.

Benefits of contextual attention. Table 2.4 compares the performance of Hand-CNN with its variants. All models were trained using the train set of the TV-Hand data and the COCO-Hand-S data. We did not use the full COCO-Hand dataset for training here because we wanted to rule out the possible interference of the black circular masks in our analysis of the benefits of non-local contextual pooling.

On the Oxford-Hand test set, Hand-CNN significantly outperforms MaskRCNN, and this indicates the benefits of the contextual attention module. MaskRCNN is essentially Hand-CNN without a contextual attention module. We also train a Hand-CNN detector without the semantics context component and another detector without the similarity context component. As seen from Table 2.4, both types of contextual cues are helpful for hand detection.

The benefit of the contextual module is not as evident in the TV-Hand dataset. This is possibly due to images from TV series containing only the closeup upper bodies of the characters, and hands can appear out of proportion with the other body parts. Thus contextual information is less meaningful in this dataset. For reference, the Hand Keypoint method [80] also performs poorly on this dataset (38.9% AP); this method relies heavily on context information.

Benefits of additional training data. One contribution of our paper is the collection of a large-scale hand dataset. Undoubtedly, the availability of this large-scale dataset is one reason for the impressive performance of our hand detector. Table 2.5 further analyzes the benefits of using more and

Method	Oxford-Hand	TV-Hand
MaskRCNN	69.9%	59.9%
Hand-CNN	73.0%	60.3%
Hand-CNN w/o semantic context	71.4%	59.4%
Hand-CNN w/o similarity context	70.8%	59.6%

Table 2.4: **The benefits of context for hand detection.** The performance metric is AP. All models were trained using the train set of the TV-Hand and COCO-Hand-S. MaskRCNN is essentially Hand-CNN without using any type of context. It performs worse than Hand-CNN and other variants.

	Test Data	
Train Data	Oxford-Hand	TV-Hand
TV-Hand	62.5%	55.4%
TV-Hand + COCO-Hand-S	69.9%	59.9%
TV-Hand + COCO-Hand	76.7%	63.5%

Table 2.5: **Benefits of data.** This shows the performance of MaskRCNN trained with different training data.

more data. We train MaskRCNN using three datasets: TV Hand, COCO-Hand-S, and COCO-Hand. The TV-Hand dataset has 4853 training images, the COCO-Hand-S has 4534 images, and COCO-Hand has 26,499 images.

A detector trained with the training set of TV-Hand data already performs well, including on the cross-data: Oxford-Hand dataset. This proves the generalization ability of our hand detector and the usefulness of the collected data. Table 2.5 also suggests the importance of having extra training data from Microsoft COCO. We see that using COCO-Hand data instead of COCO-Hand-S improves AP by 6.8% on the Oxford-Hand and 3.6% on the challenging TV-Hand data. As explained in Section 2.4.2, COCO-Hand-S data was obtained from the COCO-Hand data by discarding images with even one unannotated hand without caring about the

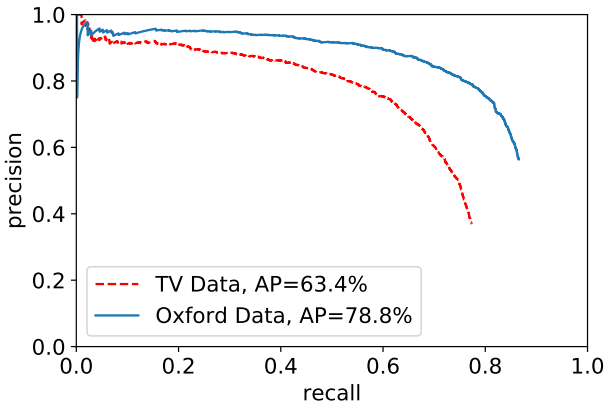


Figure 2.6: **Precision-recall curves of Hand-CNN**, trained on TV-Hand + COCO-Hand, tested on test sets of the Oxford-Hand and the TV-Hand data.

good hand annotations the image possibly contains. In COCO-Hand, we preserved images with good annotations by masking unannotated hands. The results of the experiments clearly show the benefits of doing so.

Precision-Recall curves. Fig. 2.6 plots precision-recall curves of the Hand-CNN on the test sets of the Oxford-Hand and TV-Hand datasets. The Hand-CNN was trained on the train sets of the TV-Hand and COCO-Hand datasets. The Hand-CNN has high precision values. For example, at 0.75 recall, the precision of Hand-CNN is 0.81.

2.5.3 Orientation Performance of the Hand-CNN

Table 2.6 shows the accuracy values of the predicted hand orientations of the Hand-CNN. We measure the angle difference between the predicted and annotated orientations. We consider three different error thresholds of 10, 20, and 30 degrees and calculate the percentage of predictions within the error thresholds. As can be seen, the prediction accuracy is over ~75% for the error threshold of 30°. Note that we only consider the performance of the orientation prediction for correctly detected hands.

Figure 2.7: **Some detection results of Hand-CNN**. Hands with various shapes, sizes, and orientations are detected.

	Prediction error in angle		
Test Data	$\leq 10°$	$\leq 20°$	$\leq 30°$
Oxford-Hand	41.26%	64.49%	75.97%
TV-Hand	37.65%	60.09%	73.50%

Table 2.6: **Accuracy of hand orientation prediction** of the Hand-CNN on test sets of the Oxford-Hand and TV-Hand data. This table shows the percentage of correct orientation predictions for the three error thresholds of 10, 20, and 30°. The error is calculated as the angle difference between the predicted and annotated orientations. We only consider the performance of the orientation prediction for hands with an intersection over the union greater than 0.5 with the corresponding ground truth.

2.5.4 Qualitative Results and Failure Cases

Fig. 2.7 shows some detection results of the Hand-CNN trained on both TV-Hand and COCO-Hand data, Fig. 2.8 compares MaskRCNN and Hand-CNN. MaskRCNN mistakes skin areas as hands in many cases. Hand-CNN uses contextual cues provided by contextual attention for disambiguation to reduce such mistakes. Hand-CNN also predicts hand orientations, while MaskRCNN does not. Fig. 2.9 shows some failure cases of Hand-CNN. False detections are often due to other skin areas. Contextual cues help to reduce this type of mistake, but errors still occur due to skin area at plausible locations.

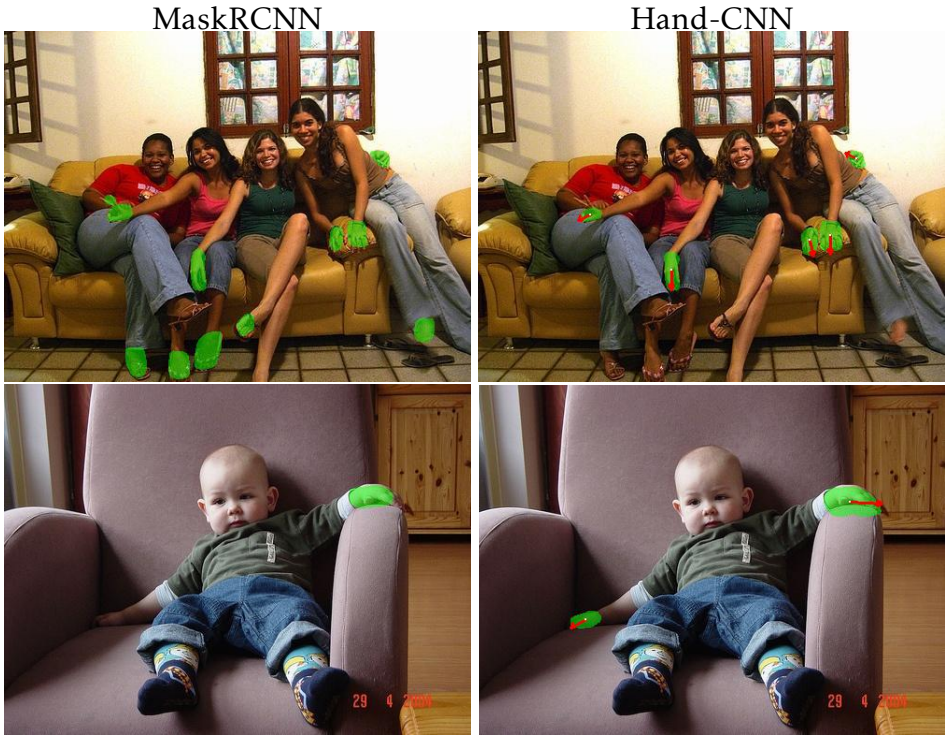


Figure 2.8: **Comparing the results of MaskRCNN (left) and Hand-CNN (right).** MaskRCNN mistakes skin areas as hands in many cases. Hand-CNN avoids such mistakes using contextual attention. Hand-CNN also predicts hand orientations, while Mask RCNN does not.

Figure 2.9: **Some failure cases of Hand-CNN.**

2.6 Conclusions

In this chapter, we described Hand-CNN, a novel convolutional architecture for detecting hand masks and predicting hand orientations in unconstrained images. Our network is founded on MaskRCNN but has a novel contextual attention module to incorporate contextual cues in the detection process. The contextual attention module can be implemented as a modular layer and is inserted at different stages of the object detection network. We have also collected and annotated a large-scale dataset of hands. This dataset can be used for training and evaluating the hand detectors. Hand-CNN outperforms MaskRCNN and other hand detection algorithms by a wide margin on two datasets. For hand orientation prediction, more than 75% of the predictions are within 30 degrees of the corresponding ground truth orientations.

Chapter 3

Hand Contact Recognition

In this chapter, we investigate a new problem of detecting hands and recognizing their physical contact state in unconstrained conditions. This is a challenging inference task, given the need to reason beyond the local appearance of hands. The lack of training annotations indicating which object or parts of an object the hand is in contact with further complicates the task. We propose a novel convolutional network based on Mask-RCNN that can jointly learn to localize hands and predict their physical contact to address this problem. The network uses outputs from another object detector to obtain the locations of objects in the scene. It uses these outputs and hand locations to recognize the hand's contact state using two attention mechanisms. The first attention mechanism is based on the hand and a region's affinity, enclosing the hand and the object, and densely pools features from this region to the hand region. The second attention module adaptively selects salient features from this plausible contact region. To develop and evaluate our method's performance, we introduce a large-scale dataset called ContactHands, containing unconstrained images annotated with hand locations and contact states. The proposed network, including the parameters of attention modules, is end-to-end trainable.

3.1 Introduction

The work in this chapter aims to detect hands in images and recognize their physical contact state. By physical contact state, we mean to recognize the following four conditions for each hand instance, namely (1)

No-Contact: the hand is not in contact with any object in the scene; (2) Self-Contact: the hand is in contact with another body part of the same person; (3) Other-Person-Contact: the hand is in contact with another person; and (4) Object-Contact: the hand is holding or touching an object other than people. These conditions are not mutually exclusive, and a hand can be in multiple states; for example, a hand can contact another person and simultaneously hold an object. Detecting hands and recognizing their physical contact is an essential problem with many potential applications, including harassment detection, contamination prevention, and activity recognition.

However, recognizing the contact state of a hand in unconstrained conditions is challenging because the hand's appearance alone is insufficient to estimate its contact state. This task also requires us to consider the relationships between the hand and other objects in the scene. This can be a complex inference problem for many real-world situations, especially where numerous people and objects surround the hand. Furthermore, even for a pair of hands and objects with corresponding segmentation masks, it is not easy to recognize whether the hand is in contact with the object due to the lack of depth information. A heuristic-based method using occlusion or overlapping criteria would not work well because the hand can hover in front of the object without touching it.

In this work, we propose a Contact-Estimation neural network module for recognizing the physical contact state of hands. This module can be integrated into an object detection framework to jointly detect hands and recognize their contact states. With the hand detector, we can train the Contact-Estimation module end-to-end using training images where hands are localized and annotated with corresponding contact states. Our method does not require annotation for the contact object or areas. One technical contribution of our paper is learning to recognize contact states using weak annotations.

Specifically, we implement our method based on Mask-RCNN [32], a state-of-the-art object detection framework. Mask-RCNN has a Region Proposal Network (RPN) that first generates a candidate hand proposal box. A box regression head and a mask head then obtain the bounding box and a binary segmentation map of the hand. Additionally, we obtain the locations of other objects in the scene using a generic object detector pretrained on the COCO [49] dataset. We then use the Contact-Estimation branch to recognize the contact state for detected hands. The inputs to

this new branch are: (1) the feature maps for the hand and (2) a set of K feature maps, one for each hand-object union box, where K is the number of detected objects.

Given the above inputs, we use the Contact-Estimation network module to compute scores for each of the K hand-object pairs. We first combine the hand feature map with the hand-object union feature map at particular spatial locations. Intuitively, if the location A of the hand is in contact with the location B of the object, it would be helpful to combine hand features at A with the object features at B. We formalize this notion using a cross-feature affinity-based attentional pooling module that can combine hand and hand-object union features from various locations based on the affinities between them. Second, the hand-object union feature map encodes the regions between the hand and the object and can contain possible contact regions. We propose a spatial attention method to learn to focus on salient regions. Finally, we obtain contact state scores for each of the K hand-object pairs independently using the cross-feature affinity-based and spatial attention modules. The proposed attention modules are trained end-to-end together with the Contact-Estimation branch.

We also collect a large-scale dataset for development and evaluation. Our dataset comprises around 21K images containing bounding box annotations for 58K hands and their physical contact states. The dataset contains many challenging images in the wild, where it is not trivial to determine the physical contact states of hands. This dataset can be used to develop real-world applications that require contact states of hands, such as contamination prevention and harassment detection.

3.2 Related Work

We build upon two-stage object detection frameworks such as [27, 28, 32, 72]. The current object detection frameworks recognize an object's presence or absence in a particular region of interest by classifying the pooled feature inside this region. However, such a framework is insufficient for our problem. In our case, we need to detect hands and recognize their physical contact state by reasoning about other surrounding objects.

There are prior works for hand detection [12, 21, 42, 44, 45, 54, 64, 75, 79, 99, 114] and hand pose estimation [30, 74, 81, 116, 117]. However, they do not study hand contact state recognition. One way to estimate

the contact state of a hand is to reason based on its estimated pose. However, obtaining a reasonably good hand pose in unconstrained conditions is challenging. For example, obtaining the hand pose in low-resolution visual data such as surveillance images in a supermarket or an elevator is highly difficult. Therefore recognizing the physical contact states of hands using their pose is not a robust approach.

Some works consider hand-object interactions. Hasson et al. [30] propose reconstructing hand and object meshes from monocular images. Bambach et al. [6] aims to locate hands interacting with an object, focusing on first-person videos containing only two people. Moreover, most of the activities focus on hands playing cards or puzzles. Tekin et al. [86] propose to model hand-object interactions by jointly estimating the 3-D poses for hands and objects. However, none of these works estimate the physical contact state of the hand.

Closely related to our problem is the work from Shan et al. [77]. They propose a video-frame dataset of everyday interactions from Youtube and annotate them with hand locations, hand side, contact state, and contact object location. In our work, we aim to recognize physical contact in the wild, and the images in our dataset are unconstrained. Shan et al. [77] also developed a method using Faster-RCNN to detect hands and predict contact based on the hand's appearance. Our method instead predicts hand contact by considering the hand and other surrounding objects. Another notable difference is that our method does not assume that a hand can only be in one contact state. [77] treats contact recognition as a multi-class classification problem. Instead, we treat it as a multi-label classification problem and train our method using four independent binary cross-entropy losses, one for each of the four possible contact states.

Our method consists of two attention mechanisms. The spatial attention method shares similarities with several visual attention methods that have gained much interest over the past years [5, 16, 26, 36, 55, 60, 73, 89, 91, 109]. The cross-feature affinity-based attentional pooling is inspired by [91]. We design it to pool hand-object union features to hand features by considering their affinities at every spatial location.

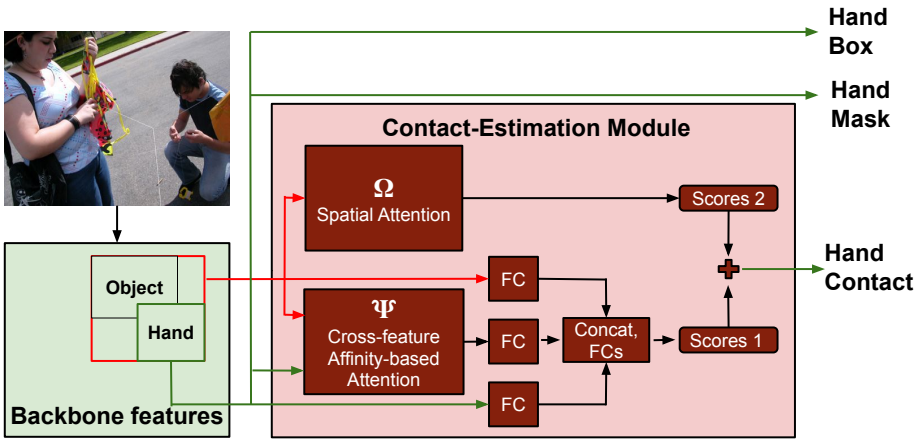


Figure 3.1: **Processing pipeline for joint hand detection and contact state recognition.** The bounding box regression head and mask head use the hand feature map to generate the hand's bounding box and mask. The Contact-Estimation module takes the hand feature map and hand-object union feature map as inputs. The cross-feature affinity-based attentional pooling pools hand-object union features to the hand features. The spatial attention method focuses on selective regions in the hand-object union feature map.

3.3 Approach

In this section, we will describe our framework's overall architecture and provide details of the Contact-Estimation module used to recognize the physical contact state of a hand.

3.3.1 Model Overview

The proposed architecture is illustrated in Fig. 3.1. A Region Proposal Network (RPN) obtains rectangular hand proposals. For each proposal, we extract ResNet backbone features of dimensions $h \times w \times d$ using the RoI Align operation and perform the bounding box regression and the binary mask segmentation. Additionally, we use a pre-trained object detector to detect other objects in the image. We use such detected objects to obtain hand-object union regions. Suppose the number of detected objects is K. We then extract K ResNet features of dimensions $h \times w \times d$, one

for each K hand-object union regions. These features, together with the hand features, are then passed to the Contact-Estimation module. The Contact-Estimation module then computes 4-dimensional score vectors $\mathbf{s}_k \in \mathbb{R}^4, 1 \leq k \leq K$, one for each K hand-object pairs. The K score vectors are then combined to a single vector $\mathbf{s} \in \mathbb{R}^4$ to obtain the contact state class scores for the hand. We now provide more details about computing the contact state class scores $\mathbf{s} \in \mathbb{R}^4$ from the hand features $\mathbf{H} \in \mathbb{R}^{h \times w \times d}$ and K hand-object union features $\mathbf{U}_1, \cdots, \mathbf{U}_K \in \mathbb{R}^{h \times w \times d}$.

3.3.2 Recognizing Physical Contact using Multiple Objects

We now present the forward logic for the Contact-Estimation module. It takes two sets of inputs, the hand feature map $\mathbf{H} \in \mathbb{R}^{h \times w \times d}$ and K hand-object union feature maps $\mathbf{U}_1, \cdots, \mathbf{U}_K \in \mathbb{R}^{h \times w \times d}$, one for each K detected objects. The output of this module is a vector of contact state scores $\mathbf{s} \in \mathbb{R}^4$.

For each hand-object pair k, $1 \leq k \leq K$, we first obtain a vector of scores $\mathbf{s}_k \in \mathbb{R}^4$ as follows:

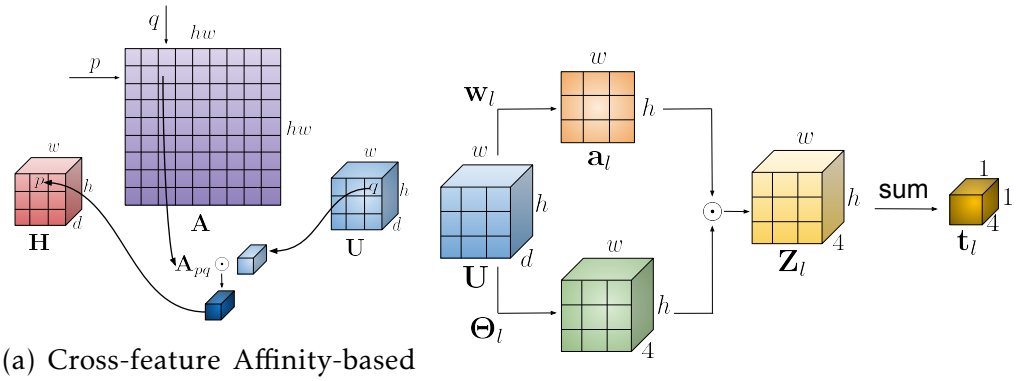
1. We obtain features $\Psi(\mathbf{H}, \mathbf{U}_k) \in \mathbb{R}^{h \times w \times d}$ by combining hand-object union features $\mathbf{U}_k$ with the hand features $\mathbf{H}$ using the cross-feature affinity-based attentional pooling module Ψ.

2. We pass features $\Psi(\mathbf{H}, \mathbf{U}_k), \mathbf{H}, \mathbf{U}_k$ through fully-connected layers, concatenate them, and finally project using fully-connected layers to obtain a first set of scores $\mathbf{s}_k^{(1)} \in \mathbb{R}^4$.

3. We obtain a second set of scores $\mathbf{s}_k^{(2)} := \Omega(\mathbf{U}_k) \in \mathbb{R}^4$ by passing the hand-object union feature map $\mathbf{U}_k$ through the spatial attention module Ω.

4. We compute the class scores $\mathbf{s}_k \in \mathbb{R}^4$ for the k^{th} hand-object pair as $\mathbf{s}_k := \mathbf{s}_k^{(1)} + \mathbf{s}_k^{(2)}$.

Once we obtain scores $\mathbf{s}_k \in \mathbb{R}^4$ for each K hand-object pairs, we compute the contact state scores $\mathbf{s} \in \mathbb{R}^4$ for the hand feature $\mathbf{H}$ by taking the element wise maximum of K scores: $\mathbf{s} := \max_{1 \leq k \leq K} \mathbf{s}_k$.



(a) Cross-feature Affinity-based Attention

(b) Spatial Attention

Figure 3.2: **(a) Cross-feature Affinity-based Attentional Pooling.** We pool the hand-object union feature from $\mathbf{U}$'s q^{th} location to the hand feature $\mathbf{H}$'s p^{th} location, weighted by the affinity $\mathbf{A}_{pq}$ between them. We do this densely for all spatial locations p and q. **(b) Spatial Attention.** The attention map $\mathbf{a}_l$ encodes salient regions of the hand-object union region. We use $\mathbf{a}_l$ to select scores from such locations to obtain $\mathbf{Z}_l$. We finally obtain the scores $\mathbf{t}_l$ by summing scores from all spatial locations of $\mathbf{Z}_l$.

3.3.3 Cross-feature Affinity-based Attentional Pooling to Combine Features

We now describe a method to combine hand features with object features. Based on the intuition that different regions of the object have different affinities to contact the hand, we propose an attention method that combines features at each spatial location of the hand with features from all possible spatial locations of the hand-object union region, weighted by affinities. We parameterize these affinities in the attention module's weights and learn them end-to-end during training. Fig. 3.2a illustrates this attention method.

The attention module takes as input the hand features $\mathbf{H} \in \mathbb{R}^{n \times d}$ and the hand-object union features $\mathbf{U} \in \mathbb{R}^{n \times d}$. Here, $n := hw$ denotes the number of spatial locations, and d denotes each feature's dimensions. The attention module outputs combined features $\Psi(\mathbf{H}, \mathbf{U}) \in \mathbb{R}^{n \times d}$ as:

$$\Psi(\mathbf{H}, \mathbf{U}) := \mathbf{H} + \text{softmax}(\mathbf{A})\mathbf{U}. \tag{3.1}$$

Here, $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a matrix such that $\mathbf{A}_{p,q}$ encodes the affinity between

the p^{th} hand features and the q^{th} hand-object union features and softmax($\mathbf{A}$) denotes the softmax taken along the last dimension of $\mathbf{A}$. We parameterize $\mathbf{A}$ using weights $\mathbf{W}_\alpha \in \mathbb{R}^{d\times d}$ and $\mathbf{W}_\beta \in \mathbb{R}^{d\times d}$ as follows:

$$\mathbf{A} := (\mathbf{H}\mathbf{W}_\alpha)(\mathbf{U}\mathbf{W}_\beta)^T. \tag{3.2}$$

We implement the weights $\mathbf{W}_\alpha$ and $\mathbf{W}_\beta$ using 1×1 convolutions and learn them during training. Notably, we can implement this attention module's entire forward logic as a few matrix multiplications and additions in less than five lines of PyTorch code.

3.3.4 Spatial Attention to Learn Salient Regions

The hand-object union feature map encodes both the hand and the object's appearance and can contain crucial contextual regions that determine the hand's physical contact state. However, it is not trivial to select features from such regions. This subsection proposes an attention method to learn salient regions adaptively and predict contact state scores based on such regions' features. The attention module takes as input the hand-object union features $\mathbf{U} \in \mathbb{R}^{n\times d}$, where $n := hw$ denotes the number of spatial locations and d denotes feature's dimensions. The output of the attention module is a vector of contact state class scores $\mathbf{s}^{(2)} = \Omega(\mathbf{U}) \in \mathbb{R}^4$.

To recognize the physical contact state of the hand, we first localize the areas of the hand-object union region that are relevant for the recognition decision. Specifically, we learn L spatial attention maps $\mathbf{a}_1, \cdots, \mathbf{a}_L \in \mathbb{R}^n$ that focus on selective regions of the hand-object union features. Corresponding to each such attention map $\mathbf{a}_l$, $1 \le l \le L$, we obtain score vector $\mathbf{t}_l \in \mathbb{R}^4$. We do this by predicting score vectors at each spatial location of the hand-object union region and averaging them, weighted by the attention map. We finally obtain the contact state scores $\mathbf{s}^{(2)}$ by averaging score vectors $\mathbf{t}_1 \cdots, \mathbf{t}_L$ corresponding to all L attention maps. We illustrate the proposed attention method in Fig. 3.2b.

Formally, we first define the attention maps $\mathbf{a}_1, \cdots, \mathbf{a}_L \in \mathbb{R}^n$ by $\mathbf{a}_l :=$ softmax($\mathbf{U}\mathbf{w}_l$), where $\mathbf{w}_l \in \mathbb{R}^d$ is a learnable weight vector for the l^{th} attention map $\mathbf{a}_l$.

Next, for each attention map $\mathbf{a}_l$, we define $\mathbf{Z}_l \in \mathbb{R}^{n\times 4}$ as $\mathbf{Z}_l := \mathbf{a}_l \odot (\mathbf{U}\mathbf{\Theta}_l)$. Here, $\mathbf{\Theta}_l \in \mathbb{R}^{d\times 4}$ are learnable weights and $\odot$ denotes the element-wise multiplication by broadcasting elements of $\mathbf{a}_l$. Intuitively, $\mathbf{Z}_l$ encodes

scores at all n spatial locations weighted by the l^{th} attention map $\mathbf{a}_l$. We then compute the score vector $\mathbf{t}_l \in \mathbb{R}^4$ corresponding to the l^{th} attention map $\mathbf{a}_l$ by summing scores at all spatial locations of $\mathbf{Z}_l$.

Finally, we compute the contact state class scores $\mathbf{s}^{(2)} \in \mathbb{R}^4$ by averaging scores $\mathbf{t}_1, \cdots, \mathbf{t}_L$ corresponding to all L attention maps $\mathbf{a}_1, \cdots, \mathbf{a}_L$: $\mathbf{s}^{(2)} := (\sum_{l=1}^{L} \mathbf{t}_l)/L$.

We implement the weights $\mathbf{w}_l$ and $\mathbf{\Theta}_l$ in the above equations as 1×1 convolutions and learn them end-to-end during training. The entire arithmetic operations can be implemented as vectorized operations within ten lines of PyTorch code.

3.3.5 Loss Function for the Proposed Architecture

We train the entire network containing the bounding box regression, mask generation, and contact-estimation branches end-to-end by jointly optimizing the following multi-task loss:

$$\mathcal{L} := \mathcal{L}_{cls} + \mathcal{L}_{box} + \mathcal{L}_{mask} + \lambda \mathcal{L}_{contact}. \tag{3.3}$$

Here, $\mathcal{L}_{cls}$, $\mathcal{L}_{box}$, $\mathcal{L}_{mask}$ denote the classification, the bounding box regression, and the segmentation mask losses, respectively. These are the standard loss terms of the Mask-RCNN object detection framework [32]. The term $\mathcal{L}_{contact}$ denotes the loss for the physical contact state of the hand, and λ is a tunable hyperparameter denoting the weight of the contact loss. The contact loss $\mathcal{L}_{contact}$ is the sum of four independent binary cross-entropy losses corresponding to four possible contact conditions, i.e., $\mathcal{L}_{contact} := L_1 + L_2 + L_3 + L_4$. We define the contact loss based on independent binary cross-entropy losses instead of a single softmax cross-entropy loss since a hand can have multiple contact conditions. Thus, it is better to treat contact recognition as a multi-label classification problem rather than a multi-class classification.

3.4 ContactHands Dataset

We collect a large-scale dataset of unconstrained images to develop and evaluate our model. We aim to collect images containing diverse hands with various shapes, sizes, orientations, and skin tones. We name the

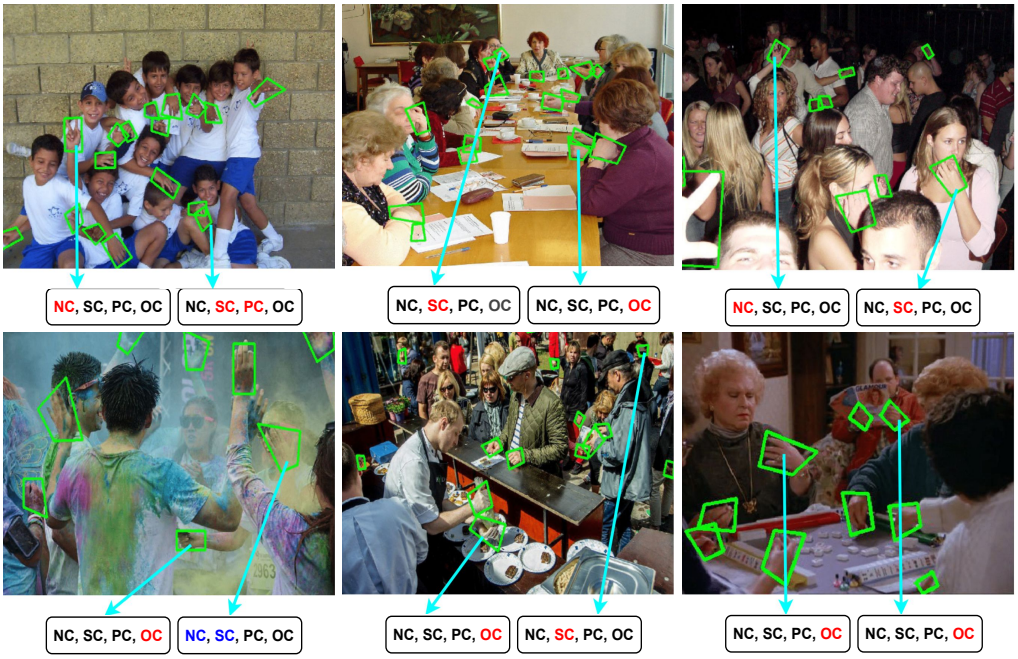
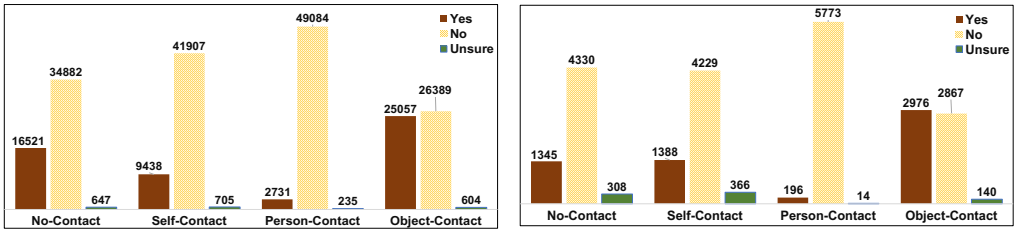


Figure 3.3: **Sample data from ContactHands.** We show the bounding box annotations in green color. We display contact states for only two hand instances per image to avoid clutter. The notations NC, SC, PC, and OC denote No-Contact, Self-Contact, Other-Person-Contact, and Object-Contact. We highlight the contact state for a hand by red color. If a contact state is unsure, we highlight it in blue.



(a) Train Data (b) Test Data

Figure 3.4: **ContactHands dataset statistics.** There are 52,050 and 5,893 annotated hand instances in the training and the test set. For each hand instance, we provide contact state annotations by choosing *Yes*, *No*, or *Unsure*.

proposed dataset ContactHands, which has numerous hand instances for which it is challenging to recognize the contact state.

Dataset Source. We collect two types of data, still photos and video frames. For still images, we collect images from multiple sources. First, we select images that contain people from popular datasets such as MS COCO [49] and PASCAL VOC [23] datasets. The COCO dataset images have everyday objects containing various annotations, including bounding boxes, keypoints, and segmentation masks for persons. Similarly, the PASCAL VOC is a benchmark in visual object category recognition and object detection. However, these datasets do not have annotations for hand bounding boxes and their physical contact states. As a second source for still photos, we scrape some photos from Flickr using keywords that are likely to return pictures containing people. For example, we use keywords such as *people*, *cafeteria*, *parks*, *party*, *shopping*, *library*, *students*, *camping*, *vacation*, *outdoors*, *meeting*, *hanging-out*, *tourists*, and *festivals*. We only collect those pictures which have appropriate copyright permissions. Also, we manually inspect each image to keep only those that contain at least one person. Together with the MS COCO and PASCAL VOC dataset images, these images form our still images group. To complement the still photos, we also collect frames sampled from videos. For this purpose, we use the training and validation split of the Oxford Hand dataset [54] and TV-Hand [60] dataset. Altogether, our dataset has 21,637 images.

Annotation and Quality Control, Dataset Statistics. We annotate the

data using multiple annotators and subsequently verify them. We ask annotators to localize hand instances by drawing a tight quadrilateral bounding box that contains as many hand pixels as possible. We instruct them to localize all hands for which the minimum side of the resulting axis-parallel box has a length greater than $\min(H, W)/30$, where H and W are the height and width of the image. We ask annotators to localize truncated and occluded hands as long as the visible hand areas' size is greater than the previously mentioned threshold. We choose quadrilateral boxes instead of axis-parallel bounding boxes since hands are incredibly articulate, and the axis-parallel bounding box provides poor hand localization. In addition to localizing hands, we ask annotators to identify the physical contact state for each hand instance. Since hands can be in multiple contact states, we instruct the annotators to consider the four possible contact states independently; we ask them to answer *Yes*, *No*, and *Unsure* for each of the four possible contact states separately.

We collect annotations in batches. We ask an additional annotator to verify each batch's annotations for quadrilateral boxes and contact states. We further verify each batch's physical contact states' annotations by randomly sampling a fraction of images and independently annotating contact states for every hand instance. We then quantitatively measure the error in annotations to verify that the error is within 2% for all annotations batches. Figure Fig. 3.3 shows some sample images and annotations from our dataset ContactHands.

The total number of annotated hand instances is 58,165. We randomly sampled 18,877 images from these annotated images to be our training set and 1,629 images to be our test set. The train and test sets have 52,050 and 5,983 hand instances, respectively. Fig. 3.4 displays some statistics about contact states annotations.

3.5 Experiments

In this section, we will provide details about model implementation and hyperparameters. We will also explain the evaluation metric and present experimental results.

Model Implementation and Hyperparameters. We implement the proposed architecture using Detectron2 [100]. We add a Group Normalization layer before the residual connection in the cross-feature affinity-based

	Axis-Parallel		Quadrilateral	
Contact State	Exact	Extended	Exact	Extended
No-Contact	24.49 %	45.82 %	40.90 %	29.27 %
Self-Contact	24.76 %	38.57 %	34.63 %	30.16 %
Other-Person-Contact	3.49 %	3.99 %	4.11 %	3.92 %
Object-Contact	51.63 %	65.88 %	62.19 %	58.53 %
mAP	26.10 %	38.56 %	35.46 %	30.47 %

Table 3.1: **Hand contact recognition APs** of ResNet-101 classifiers. The performance is evaluated on the test set of the ContactHands dataset.

attentional pooling to stabilize the training. We set the number of attention maps L for the spatial attention module to be 32. The weight λ for the contact state loss $\mathcal{L}_{contact}$ in the Eq.(3.3) is set to 1. The binary cross-entropy losses for all four contact states have equal weights; i.e., we do not scale the losses. The fully-connected layers in the Contact-Estimation branch have dimensions 1024. Note that tuning the loss weights for four states, parameter L, and dimensions for fully-connected layers can likely give better results. We train the network using SGD with an initial learning rate of 0.001 and a batch size of 1. We reduced the learning rate by 10 when the performance plateaued. We do not penalize contact state predictions for *Unsure* contact states hand instances during training.

Evaluation Metric. We measure the performance of joint hand detection and contact recognition using the average VOC precision metric. We consider a detected hand instance to be a true positive if: (1) the Intersection over Union (IoU) between the axis parallel bounding box of the detected hand and ground truth bounding box is larger than 0.5; and (2) the predicted contact state matches the ground truth. More precisely, for each contact state, we only consider hand boxes annotated *Yes* for that contact state to be ground truth boxes. We then measure the joint hand detection and contact state recognition AP by multiplying the hand detection score with the predicted contact score. We do not measure the performance for detections that overlap with *Unsure* contact state hand instances.

Baselines. Given the hand's location, one might think of learning a clas-

	No-Contact	Self-Contact	Other-Person-Contact	Object-Contact	mAP
AP	35.50 %	38.29 %	4.48 %	61.30 %	34.89 %

Table 3.2: **Hand contact recognition APs** of a method based on human pose estimation. The performance is evaluated on the test set of the ContactHands dataset.

Method	M-RCNN	Proposed	M-RCNN	Proposed	Proposed	Proposed
Train data	100DOH	100DOH	C-Hands	C-Hands	100DOH	C-Hands
Test data	100DOH	100DOH	C-Hands	C-Hands	C-Hands	100DOH
No-Contact	67.30 %	68.23 %	60.52 %	62.48 %	44.45%	47.13 %
Self-Contact	54.94 %	58.52 %	51.62 %	54.31 %	32.03 %	38.85 %
Other-Person	6.56 %	12.94 %	33.79 %	39.51 %	7.32 %	6.77 %
Object-Contact	90.34 %	92.70 %	67.43 %	73.34%	49.68 %	74.27 %
mAP	54.78 %	58.10 %	53.31 %	57.41 %	33.37 %	41.76 %

Table 3.3: **Joint hand detection and contact recognition APs** using different methods and datasets. M-RCNN denotes Mask-RCNN. 100DOH denotes video frames dataset [77] and C-Hands denotes our dataset ContactHands.

sifier on such hand crops to obtain its contact state. To see how such a method performs, we train ResNet-101-based classifiers on hand crops from the training set of the ContactHands dataset. We consider two types of hand crops, one corresponding to the hand's axis-parallel bounding box and another corresponding to the quadrilateral bounding box. To construct a rectangular image from a quadrilateral box, we first obtain a rotated rectangular bounding box and then build an axis-parallel image crop. We further consider two variants for each type of hand crop, the exact bounding box, and the extended bounding box, to provide surrounding context information. To obtain this extended bounding box, we increase each side of the hand crop's length by 50% so that the expanded bounding box has an area of 2.25 times the original bounding box area. Altogether, there are four variants, and we train four different ResNet-101

Train data	100DOH + C-Hands	
Test data	100DOH	C-Hands
No-Contact	70.16 %	63.90 %
Self-Contact	58.66 %	59.30 %
Other-Person	21.15 %	42.01 %
Object-Contact	88.21%	70.49 %
mAP	59.54 %	58.93 %

Table 3.4: **Cross dataset evaluation performance.** a model trained on the ContactHands dataset has better cross-dataset generalization performance than the 100DOH dataset model

classifiers and evaluate contact recognition performance on the test set of ContactHands. We summarize the results in Table 3.1. These results show that learning a classifier directly on hand crops is inadequate for recognizing their contact states.

Given the success of 2D human pose estimation methods, we want to know if we can use the relationship between a hand and joint locations of humans to reason about the hand's contact state. For this purpose, we build a feature vector $\mathbf{h} \in \mathbb{R}^{52}$ for each hand instance using the following heuristic. We use [80] to detect keypoints corresponding to 25 human joints. Additionally, we use an object detector to detect all possible objects in the scene. Then for each hand instance, we build three types of features. First, we construct a vector $\mathbf{h_s} \in \mathbb{R}^{24}$ of distances from the wrist joint to the other 24 joints of the same person. Second, we obtain a vector $\mathbf{h_p} \in \mathbb{R}^{25}$ of average distances from the wrist joint to 25 joint locations of other people in the scene. Here, the average is with respect to other people. Finally, we obtain a vector $\mathbf{h_o} \in \mathbb{R}^3$ encoding the relationship between the hand and the detected objects. Precisely, the first component of $\mathbf{h_o}$ is the mean distance of the hand from the detected objects. The second and third components of $\mathbf{h_o}$ are the mean overlap and the mean IoU of the hand with the detected objects. We obtain the final feature vector $\mathbf{h} \in \mathbb{R}^{52}$ for the hand by concatenating $\mathbf{h_s}$, $\mathbf{h_p}$ and $\mathbf{h_o}$. We use such hand feature vectors $\mathbf{h}$ to train a classifier on the training set of ContactHands. Table 3.2 summarizes the classifier's performance on the test set of ContactHands. The results show that human pose heuristic methods are insufficient to

estimate hands' contact states in unconstrained conditions.

Main Results. We present the proposed method's results and compare them to Faster-RCNN and Mask-RCNN to detect and recognize hand contact states. For this purpose, we use the training and test splits from the ContactHands dataset and the 100DOH [77] dataset. The 100DOH is a video-frame dataset with 79,920 training images and 9,983 test images. We conduct experiments by training the proposed architecture and comparing it to a modified Mask-RCNN that can detect hands and recognize contact states. We summarize the results of these experiments in Table Table 3.3.

The proposed method performs better than the Mask-RCNN since it considers surrounding objects when making a contact decision. We also experimented with Faster-RCNN instead of Mask-RCNN, which performs similarly to Mask-RCNN. Specifically, we found that Faster-RCN has 54.04 % mAP when trained and tested on the 100DOH dataset and 53.23 % mAP when trained and tested on the ContactHands dataset.

Table 3.4 show the cross dataset evaluation performance. We can see that a model trained on the ContactHands dataset has better cross-dataset generalization performance than the 100DOH dataset model. These results show the benefit of our data. The last two columns show that a model trained on a combination of the 100DOH dataset and the ContactHands data performs better than models trained on individual datasets separately.

Ablation Studies. We conduct experiments to study the effect of different components of the Contact-Estimation Branch. Specifically, we train the proposed network on the training set of ContactHands by removing the cross-feature affinity-based attention module, the spatial attention module, and both. We evaluate these methods on the test set of ContactHands, and they achieve 56.08%, 55.91%, and 55.12% mAP on the joint task of detecting hands and recognizing their contact. Comparing these results with the entire architecture with 57.41% mAP shows that both the attention methods help estimate the hand's contact state.

Qualitative Results and Failure Cases. Fig. 3.5 shows some qualitative results from the proposed model, trained on the ContactHands dataset. The first three rows show promising results, and the last row shows failure cases. The failure cases are mainly from two sources, false hand detec-

Figure 3.5: **Qualitative results and failure cases.** The first three rows show some good qualitative results, and the last row shows some failure cases from our method. We visualize detected hand instances by their predicted contact state color. We add additional contact state labels if a hand is in more than one contact state.

tions and bad contact state predictions. First, sometimes other skin areas are mistaken for hands, and thus hand detections are not perfect. Second, even if a hand is detected correctly, its predicted contact state can be incorrect; when a hand is surrounded or occluded by other objects, the lack of depth information can make the contact decision challenging.

3.6 Conclusions

In this chapter, we investigated a new problem of hand contact recognition. We introduced a novel Contact-Estimation neural network module that can be trained end-to-end with any two-stage object detector to detect hands and simultaneously recognize their physical contact states. We also collected a challenging large-scale dataset of unconstrained images annotated with hand locations and contact states. Hand contact recognition is a less-explored problem with essential applications. It is also a challenging problem, especially in unconstrained environments, with massive room for improvement. We hope our work will further spark the community's interest in addressing this critical problem.

Chapter 4

Hand Tracking

In this chapter, we propose HandLer, a novel convolutional architecture that can jointly detect and track hands online in unconstrained videos. HandLer is based on Cascade-RCNN with additional three novel stages. The first stage is Forward Propagation, where the features from frame t-1 are propagated to frame t based on previously detected hands and their estimated motion. The second stage is the Detection and Backward Regression, which uses outputs from the forward propagation to detect hands for frame t and their relative offset in frame t-1. The third stage uses an off-the-shelf human pose method to link any fragmented hand tracklets. We train the forward propagation, backward regression, and detection stages end-to-end together with the other Cascade-RCNN components.

To train and evaluate HandLer, we also contribute YouTube-Hand, the first challenging large-scale dataset of unconstrained videos annotated with hand locations and trajectories. Experiments on this dataset and other benchmarks show that HandLer outperforms the existing state-of-the-art tracking algorithms by a large margin.

4.1 Introduction

Hand tracking is an essential problem in various application scenarios, from gesture and activity recognition to contact tracing and skill evaluation. One approach for tracking hands is to consider them parts of a human body and then perform hand tracking based on the tracked hu-

man pose. But pose detection and tracking can be unreliable, especially for people partially occluded or outside the field of view of the camera. Another approach for hand tracking is to use off-the-shelf tracking methods. Unfortunately, single-object trackers are not appropriate for tracking multiple hands, while existing multiple-object trackers do not work well for hands even though they have shown impressive performance for tracking pedestrians and vehicles [7, 11, 52, 92, 96, 97, 112]. Hand tracking is difficult because hands are not ordinary objects, given the extreme articulation of hands and the frequent interaction of hands with other objects. In a short period of a few frames, a hand's size, shape, location, and visibility can change dramatically and frequently. Many existing multiple-object trackers use the detection and association paradigm. However, hand detection would fail in motion blur and occlusion, while hand linking across time is difficult as the size, location, pose, and appearance of a hand can change drastically. Simultaneously, two different hand instances might look alike, so distinguishing them would be difficult even for a sophisticated re-identification module explicitly trained for hands.

In this work, we develop a novel convolutional architecture that can detect and track hands in unconstrained videos. We name the proposed architecture HandLer, which stands for *Hand Linker*. HandLer takes as input two consecutive video frames at times $t-1$ and t, and output the detected hands in frame t as well as their corresponding locations in frame $t-1$. The processing pipeline consists of three stages. The first stage is Forward Propagation, which propagates features from frame $t-1$ to frame t based on the locations of previously detected hands and their estimated movements. The second stage is the Detection and Backward Regression that uses outputs from the Forward Propagation to obtain the hand locations for frame t as well as their counterparts in frame $t-1$, and estimate their confidence conditioned on both the objectness scores at frames t and $t-1$. This allows us to link hand detections between two frames. Third, we establish correspondence between hand tracklets via pose association. This is to leverage the fact that hands are undetachable parts of a human body, so we can use pose to recover prematurely terminated hand tracklets.

Each stage of the proposed processing pipeline has its benefits. The propagation and conditional confidence estimation steps help detect blurry and occluded hands. The detection step is necessary to account for new hands in a video and avoid the potential drifting problem common in

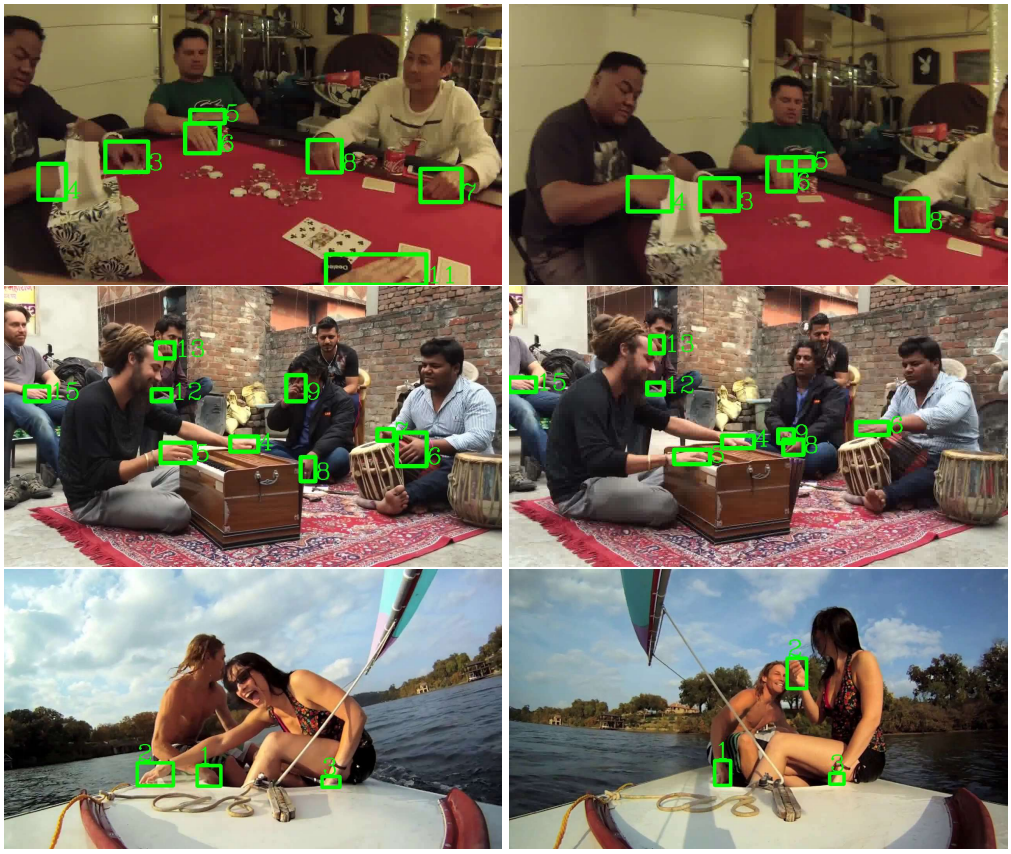


Figure 4.1: Representative image sequences from our dataset. The size, shape, location, appearance, and visibility of a hand can change drastically and frequently.

methods solely based on propagation. The regression step brings detections at two different times into a common reference frame for a more reliable linking. The high-level pose association step avoids frequent ID switches due to motion blur and occlusion.

We also introduce a new YouTube-Hand dataset for developing and evaluating hand tracking algorithms. YouTube-Hand contains 240 video sequences from diverse scene categories, including kitchens, mechanical workshops, and gyms. This dataset has 19,728 annotated frames with 864 unique hand instances.

4.2 Related Work

Many prior works only focused on hand detection in static images [12, 21, 42, 44, 45, 54, 60, 64, 67, 75, 99, 114], and works on hand tracking were developed for constrained settings such as laboratory environments and ego-centric perspectives. Sridhar et al. [82] proposed a method to track hands captured using a depth camera. Zhang et al. [108] proposed a hand tracking solution that predicted a hand skeleton of a human from a single RGB camera for AR/VR applications. Wang and Popović [90] used a single camera to track a gloved hand with an imprinted pattern. Sharp et al. [78] provided a hand tracking and pose estimation system based on a single depth camera. Mueller et al. [57] developed a 3D hand tracking approach for monocular RGB videos using a kinematic 3D hand model. Sridhar et al. [83] proposed a method to track hands manipulating objects in RGB-D videos. However, none of these methods was developed for videos in the wild; they required special markers, depth information, ego-centric perspectives, or scenes with a plain background.

Hands are objects, and we can consider Multiple-Object Tracking (MOT) methods. A popular MOT approach is tracking-by-detection, where an object detector first localizes objects, and then an association method constructs trajectories. Depending on the association method, we can categorize MOT methods as offline tracking or online tracking. Given a current frame t, offline methods [70, 102, 104] can use future frames and pose the association as a global optimization method. Meanwhile, most online methods [68, 101, 105, 113] are constrained to use frames up to frame t only. A typical way to associate detections over different frames is the Hungarian algorithm [59] with the affinity costs defined based on the overlapping criterion. Bewley et al. [9] proposed to predict bounding box movement with Kalman Filter and use a Hungarian algorithm for linking those boxes into tracks. However, this approach does not work well for unconstrained videos since hands often move fast, interact, and cross each other. Moreover, the two-step approach of detecting hands and associating hands can lead to suboptimal results since the two steps are not jointly optimized end-to-end.

There are existing methods to alleviate the disadvantages of the two-step tracking-by-detection paradigm. Bergmann et al. [7] developed a framework that uses object locations in the current frame to directly regress their corresponding locations in the following frames. However, this method

only uses current frame object locations as region proposals for the next frame. This method does not work well for tracking hands since hand locations change drastically over frames. Zhou et al. [112] proposed a point-based framework for joint detection and tracking, representing each object by a single point and tracking such points. This method outputs an offset vector from the current object center to its center in the previous frame for tracking. However, only using a point representation does not work well for hands, which are highly deformable.

Some methods process multiple frames at the same time. Feichtenhofer et al. [24] introduced correlation features that represented object co-occurrences across time to generate two-frame tracklets. However, this method does not work well when an object undergoes heavy occlusions, which is often the case for hands. Peng et al. [65] extended [24] by adding an appearance-based identity attention and proposed an online method to link two-frame tracklets. Wu et al. [98] proposed to generate a re-ID embedding in each pixel and estimate objects movement offset from this embedding. This offset can be used to propagate features and associate objects. However, those algorithms are appearance-based that do not work well for hands since the appearance of a hand can change drastically over time, and different hand instances can have a similar appearance. Instead of using correlation features or appearance-based approaches, our method directly estimates the relative offsets of hands in the previous frame, given the hand locations in the current frame. Our experiments show that this makes our hand tracking system more robust to occlusions or motion blur and reduces identity switches with other hand instances.

4.3 Proposed Method

In this section, we describe our novel method for online tracking of multiple hands. We illustrate the proposed architecture in Fig. 4.2. Our method's core is a convolutional network that operates on a pair of two consecutive frames at a time. At time t, the input to the network is a pair of video frames at time $t-1$ and t, and the output of the network are locations and confidence scores of the detected hands in frame t as well as their corresponding locations and confidence scores in frame $t-1$. We use the estimated locations of hands in time $t-1$ to establish the association with the existing hand tracks, assuming that we have tracked hands in the

video until time t-1.

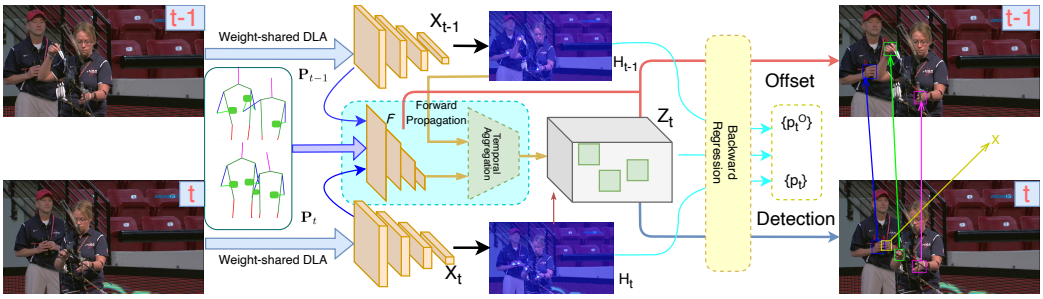


Figure 4.2: **Processing pipeline of HandLer.** Given input video frames at time t-1 and time t, we first extract their DLA features $\mathbf{X}_{t-1}$ and $\mathbf{X}_t$. We estimate the flow map O from frame t-1 to frame t, and also obtain a heatmap $\mathbf{H}_t$ in frame t from CenterNet [111]. Along with heatmap $\mathbf{H}_{t-1}$, we aggregate feature as described in Eq. (4.1) to obtain a feature map $\mathbf{Z}_t$. We then extract RoI features from $\mathbf{Z}_t$ and detect hands in frame t and also estimate their corresponding offset and probability in the frame t-1 with the backward regression.

Specifically, given two frames $\mathbf{I}_{t-1}$ and $\mathbf{I}_t$ at time t-1 and t, we use a backbone network to obtain their features $\mathbf{X}_{t-1}, \mathbf{X}_t \in \mathbb{R}^{h \times w \times d}$. Here, $h \times w$ denotes the spatial size and d denotes the number of channels. We also use an off-the-shelf pose tracker [63] to obtain pose heatmaps $\mathbf{P}_{t-1}, \mathbf{P}_t \in \mathbb{R}^{h \times w \times 15}$ corresponding to 15 human joints. Let $\mathbf{H}_{t-1} \in \mathbb{R}^{h \times w}$ denote heatmap for hands that were detected in frame $\mathbf{I}_{t-1}$. We pass features $\mathbf{X}_{t-1}$ and $\mathbf{X}_t$, pose heatmaps $\mathbf{P}_{t-1}$ and $\mathbf{P}_t$, and hand heatmap $\mathbf{H}_{t-1}$ to the forward propagation stage.

4.3.1 Forward Propagation

Given features $\mathbf{X}_{t-1}$ and $\mathbf{X}_t$, pose heatmaps $\mathbf{P}_{t-1}$ and $\mathbf{P}_t$, and hand heatmap $\mathbf{H}_{t-1}$, the forward propagation stage estimates a flow map $\mathcal{F}_t \in \mathbb{R}^{h \times w \times 2}$ and uses this flow map to obtain temporally aggregated features $\mathbf{Z}_t \in \mathbb{R}^{h \times w \times d}$.

Flow Estimation. To estimate the flow map $\mathcal{F}^t$, we propose to use the Flow Estimation Network [39]. The inputs to this network are multi-scale features $\mathbf{X}_{t-1}$ and $\mathbf{X}_t$, and the output is the 2-channeled flow map $\mathcal{F}_t \in \mathbb{R}^{h \times w \times 2}$ denoting the motion between frames $\mathbf{I}_{t-1}$ and $\mathbf{I}_t$. The two channels in $\mathcal{F}_t$ correspond to flows in the horizontal and vertical directions.

We train this Flow Estimation Network end-to-end along with other components of the network as follows. Given a pair of hands that have the same ID in frames t-1 and t, we obtain two binary masks $\mathbf{M}_{t-1}, \mathbf{M}_t \in \mathbb{R}^{h \times w}$ corresponding to two frames. These masks are the ground-truth binary segmentation maps for the hand in frames t-1 and t, respectively. We then use a bilinear warping function $\mathcal{W}$ proposed by [115] to estimate a binary segmentation map for the hand at time t: $\mathbf{M}'_t = \mathcal{W}(\mathbf{M}_{t-1}, \mathcal{F}_t)$. We then define a loss for hand motion as the MSE loss between estimated $\mathbf{M}'_t$ and the groundtruth $\mathbf{M}_t$: $L_{hmo} := \text{MSE}(\mathbf{M}'_t, \mathbf{M}_t)$.

Similarly, we also use the pose heatmap pair ($\mathbf{P}^{t-1}$, $\mathbf{P}^t$) to define a loss for pose motion. We first obtain an estimated pose at time t: $\mathbf{P}'_t = \mathcal{W}(\mathbf{P}_{t-1}, \mathcal{F}_t)$. We then define a loss for poss motion as the MSE loss between estimated $\mathbf{P}'_t$ and the groundtruth $\mathbf{P}_t$: $L_{pmo} := \text{MSE}(\mathbf{P}'_t, \mathbf{P}_t)$.

Temporal Feature Aggregation. The output $\mathcal{F}_t$ from the Flow Estimation Network is used to aggregate features from time t-1 to features from time t. Specifically, we propagate features $\mathbf{X}_{t-1}$ to features $\mathbf{X}_t$ to obtain $\mathbf{Z}_t$:

$$\mathbf{Z}_t = [1 + \mathcal{W}(\mathbf{H}_{t-1}, \mathcal{F}_t)] \odot \mathbf{X}_t + \mathcal{W}(\mathbf{H}_{t-1} \odot \mathbf{X}_{t-1}, \mathcal{F}_t) \tag{4.1}$$

In the above equation, $\odot$ is the Hadamard product, $\mathcal{F}_t$ is the estimated flow map from frame t-1 to frame t, and $\mathcal{W}$ is the bilinear warping function.

4.3.2 Hand detection and backward regression

The second important component of our architecture is the hand detection and backward regression module. The input to this module is the propagated feature map $\mathbf{Z}_t$ along with the estimated flow map $\mathcal{F}^t$. First, a CenterNet [111] will be used to obtain a dense set of hand proposals at every pixel. Second, for each proposal, we compute: (1) the bounding box of the hand at frame t, (2) the probability of this bounding box being a hand, (3) the relative offset bounding box of this hand at frame t-1, and (4) the confidence of detected box and offset box belong to same hand identity.

Tracking-based detection. We observe that our model would yield relatively lower confidence scores for some blurry and occluded hands even though those hands are visible in previous frames. Detections with low confidence scores might be dropped, leading to false negatives. To address this problem, we formulate the detection probability at frame t to

be conditioned on both the objectness scores at frames t and t-1. Consider a proposal C_k^t at time t and position k and its corresponding detection $\mathcal{D}_k$ for the anchor box at location k, we use $P(\mathcal{D}_k) = P(\mathcal{D}_k{=}hand)$ to denote the probability that $\mathcal{D}_k$ is a hand, and $P(C_k^t) = P(C_k^t = object)$ to denote the objectness probability for the proposal C_k^t. The detection likelihood is formulated as:

$$P(\mathcal{D}_k) = P(\mathcal{D}_k|C_k^t)P(C_k^t) \tag{4.2}$$

$$= P(\mathcal{D}_k|C_k^t)\sum_j^N P(C_k^t|C_j^{t-1})P(C_j^{t-1}). \tag{4.3}$$

We further assume that $P(C_k^t|C_j^{t-1}) = 0$ if there is no motion from j to k, and $P(C_k^t|C_j^{t-1}) = P(C_k^t)$ otherwise. Thus the detection likelihood becomes:

$$P(\mathcal{D}_k) = P(\mathcal{D}_k|C_k^t)\sum_{j\in\mathcal{F}_k^t} P(C_k^t)P(C_j^{t-1}), \tag{4.4}$$

where $\mathcal{F}_k^t$ denotes the set of pixel locations with motion vectors pointing to k in the optical flow map $\mathcal{F}^t$.

4.3.3 Hand-track continuation and initialization

We now describe how a newly detected hand is linked with an existing hand track or used to initialize a new hand track. Consider a particular hand $\mathcal{D}$ obtained by running the detection module with the input being the two frames at t-1 and t. Frame t detection $\mathcal{D}_t$ is represented by a quadruple: $\mathcal{D}_t = (B_t, p_t, B_t^O, p_t^O)$, where B_t is the hand location in frames t, B_t^O is its corresponding offset location in frame t-1, p_t is the corresponding detection confidence and p_t^O is the confidence of B_t and B_t^O belong to same hand identity. Note that we only keep a detection where p_t is greater than a detection threshold θ_{det}.

We then use the Hungarian algorithm [59] to match a detection $\mathcal{D}_t^i$ and also other detections with the set of existing hand tracks. This is a joint optimization process, where the best set of one-to-one correspondences is determined. If $\mathcal{D}_t^i$ is matched with an existing hand track, we will use it to continue the track. Otherwise, we will either initialize a new hand

track for B_t^i if the detection score p is higher than a threshold θ_{new}, or discard this detection. Note that θ_{new} should be higher than θ_{det} to avoid propagating false positives. In our experiments, we set $\theta_{det} = 0.6$, and $\theta_{new} = 0.9$.

The Hungarian matching process is done as follows. The inputs to this process are: (1) a set of detected hands, represented by the set of bounding boxes $\{\mathcal{D}_t\}$ in frame t, and (2) a set of active hand tracks, represented by a set of last bounding boxes of the tracks $\{\mathcal{T}_{t-1}\}$. Note that the last bounding box of $\mathcal{T}_{t-1}$ might not be at frame $t-1$. Following previous MOT methods, we only remove a hand track from the set of active hand tracks if this hand track is not matched to any new detection for more than σ frames. Given the two set of bounding boxes $\{B^O\}$ and $\{\mathcal{T}_{t-1}\}$, we obtain an affinity matrix $\mathcal{M}$ as $\mathcal{M}^{ij} = (\alpha + p_t^O) IoU(B_t^{Oi}, \mathcal{T}_{t-1}^j)$, and use the Hungarian algorithm to find the best set of one-to-one correspondences to maximize the total sum of the affinity. In our experiments, we set $\alpha = 0.1$ and $\sigma = 50$.

4.3.4 Pose association

Since hands are undetachable body parts of a human, we propose to use tracking results to guide our model for hand motion estimation and tracking. Specifically, we consider the state-of-the-art open source pose tracking algorithm LightTrack [63] and observe that it has a lower recall than our hand tracker, but most detected poses are generally accurate. We, therefore, propose to use LightTrack [63] to help estimate motion flows (described in Sec. 4.3.1) and link a newly detected hand to an existing hand track.

Also, recall that a newly detected hand is represented by a quadruple $\mathcal{D} = (B, p, B^O, p^O)$. In most cases, this detection will be used to continue a hand track as described in Sec. 4.3.3. In some cases, we will discard $\mathcal{D}$ if p is low, and we will create a new track if either p^O is low or there is no matching hand track for B^O. But these actions can lead to a false negative or a false identity switch, so we propose to address these problems with pose tracking as follows. First, given a set of detected hands and a set of wrist locations of detected poses, we run the Hungarian algorithm to find the optimal matching, where the matching cost for a hand and a wrist is based on their distance. Second, we discard detections with low p values and no matching wrists. Third, we use the procedure described

in Sec. 4.3.3 to link some detected hands with existing hand tracks. For a detected hand $\mathcal{D}$ that has not been linked to any hand track, we will link it with a hand track $\mathcal{T}$ if: (1) $\mathcal{D}$ is linked with the right/left wrist of a pose $\mathcal{P}_t$ in frame t; (2) $\mathcal{T}$ is linked with the right/left wrist of a pose $\mathcal{P}_{t-1}$ in frame t-1; and either (3a) $\mathcal{P}_t$ and $\mathcal{P}_{t-1}$ are linked via pose tracking, or (3b) the left/right wrist of $\mathcal{P}_t$ is linked with another hand $\mathcal{D}'$ that is linked with the hand track $\mathcal{T}'$, which in turn is linked with the left/right wrist of $\mathcal{P}_{t-1}$.

4.3.5 Loss function

To train this hand detection and regression module, we optimize the combined loss function: $\mathcal{L} = \mathcal{L}_{hmo} + \mathcal{L}_{pmo} + \mathcal{L}_{RPN} + \mathcal{L}_{class} + \mathcal{L}_{reg} + \mathcal{L}^{O}_{class} + \mathcal{L}^{O}_{reg}$. Here, $\mathcal{L}_{RPN}$ is the loss for the region proposal network, $\mathcal{L}_{hmo}$ and $\mathcal{L}_{pmo}$ are flow map losses, and the other terms are for the classification of the bounding box or offset regression.

4.4 YouTube-Hand Dataset

Figure 4.3: Existing hand datasets are very different from ours. This shows some representative images from: VIVA [71] (top left), EpicKitchen [18] (top right), BSL [66] (bottom left) and SynthHands [56] (bottom right).

We aim to develop a tracker that can track hands in unconstrained scenes, which may contain many people interacting with each other and the other surrounding objects. We needed a dataset of diverse conditions for training and evaluation, but such a dataset did not exist. We, therefore, compiled a new dataset containing unconstrained videos and annotated them with hand locations and trajectories.

Dataset source. We name our dataset YouTube-Hand because most of the videos (200 out of 240) were collected from YouTube. Specifically, we scraped 200 videos from 10 scenarios: casinos, concerts, cooking, dancing, driving, gyms, kids playing, mechanical workshops, sanitizing, and sports. We collected different videos from different YouTube uploaders to have a diverse dataset. We manually verified the collected videos to ensure that they were unconstrained and diverse in terms of lighting conditions, camera perspectives, skin tones, and ages. We did not collect videos that have copyright marks. Altogether, we downloaded 200 videos from YouTube, with 20 videos for each scenario. Additionally, we selected 40 videos from the PoseTracks dataset and annotated them. The videos have spatial resolutions from 640×480 to 1920×1080 and frame rates from 24 to 30 fps.

		Data source split		Train/test split	
	Total	YouTube	PoseTrack	Train	Test
#Videos	240	200	40	150	90
#Frames	232K	227K	5K	166K	65K
#Anno. hands	60K	41K	19K	30K	30K
#Trajectories	864	666	198	519	345

Table 4.1: **Statistics of the proposed YouTube-Hand dataset.**

Annotation. For each collected video, we extracted frames using the original frame rate of the video and annotated every fifteenth frame. We annotated only those hand instances whose visible areas' axis-parallel bounding box had more than 100 pixels and whose trajectory appeared for more than 50 frames. Our dataset was annotated by three annotators and subsequently verified by two people.

Train/test split. We split our data into disjoint training and testing sets. The training set contains 150 videos, randomly selected from the 200

Dataset	Scene/camera Constraints	Has Video	#Hand Trajs.	Maximum #trajs/video
EgoHands [6]	Google glasses		0	n/a
Handseg [53]	Color gloves		0	n/a
NYUHands [87]	Hands keypoints		0	n/a
ColorHandPose [114]	3D hands keypoints		0	n/a
HandNet [95]	Fingertips		0	n/a
GANeratedHands [57]	Synthetic		0	n/a
Oxford-Hand [54]	Unconstrained		0	n/a
TV-Hand [60]	Unconstrained		0	n/a
COCO-Hand [60]	Unconstrained		0	n/a
Contact-Hand [61]	Unconstrained		0	n/a
100DOH [77]	Unconstrained	✓	0	n/a
GTEA [48]	Ego-centric	✓	0	n/a
WorkingHands [79]	Down-facing cam.	✓	0	n/a
BSL [66]	TV show, segmentation	✓	2	2
SynthHands [56]	Ego-centric	✓	1	1
ICP-PSO [69]	Hand keypoints	✓	6	1
EpicKitchen [18]	Ego-centric, auto-label	✓	1400	2
VIVA [71]	Vehicle-mounted	✓	45	4
YouTube-Hand	Unconstrained	✓	864	15

Table 4.2: **Comparing YouTube-Hand with other hand datasets.**

YouTube videos. The remaining 90 videos are used for testing.

Statistics and comparison with other hand datasets. Table 4.1 shows the statistics of our dataset. Table 4.2 compares our dataset with other existing hand datasets; most of them are for hand detection only, either having no video data or hand trajectories. Some datasets contain hand trajectories, but they only have videos for constrained camera settings, such as ego-centric or in-vehicle mounted cameras. Fig. 4.3 shows some images from these datasets, which are much more constrained than our dataset as shown in Fig. 4.1

4.5 Experiments

This section compares our method with various generic object-tracking methods and hand-tracking algorithms. We also perform ablation studies, report qualitative results, and discuss failure cases.

Methods	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDs↓	MOTP↑	MOTA↑	LTR↑	HOTA↑
LightTrack [63] (Pose)	53.4	101	70	6240	12816	1955	74.5	30.8	<u>48.4</u>	48.5
FairMot [107]	41.4	96	57	**2065**	12753	3448	76.8	39.9	31.3	39.0
MPNTrack [11] (Offline)	49.0	156	66	5918	11263	<u>1039</u>	77.0	40.0	44.3	40.7
CenterTrack[112]	37.2	113	62	<u>2279</u>	12379	3362	76.5	40.7	27.3	39.0
CenterTrack*[112]	<u>57.8</u>	137	<u>43</u>	3208	10317	1647	<u>79.0</u>	50.0	37.5	<u>49.1</u>
SORT [9]	48.3	101	72	2295	12960	1475	76.7	44.9	47.6	46.1
TraDeS [98]	53.6	<u>168</u>	<u>43</u>	3271	<u>9102</u>	1982	76.4	<u>52.7</u>	44.4	46.4
HandLer (proposed)	**70.9**	**218**	**23**	2412	**5986**	**712**	**79.9**	**70.0**	**64.3**	**59.4**

Table 4.3: **Hand tracking performance on the test set of YouTube-Hand.** In terms of MOTA, the most indicative MOT metric, HandLer outperforms other methods by a large margin. In each column, the best result is highlighted in **bold**, and the second best result is <u>underlined</u>.

4.5.1 Implementation details and evaluation metrics

Architecture Details. We implemented HandLer using Detectron2 [100]. Specifically, we built upon a Cascade-RCNN with a DLA-34 [31] backbone with a Bi-directional Feature Pyramid Network (Bi-FPN) [85]. The network can be trained end-to-end, and the inference speed is 5Hz.

Training Details. The core of HandLer is a network that takes as input two frames and outputs the linked detections across these frames. The input to the network is not necessarily a pair of consecutive frames at neither training nor testing time. To handle a wide range of video frame rates and hand movements, including low frame rate videos and fast-moving hands, we actually sampled training video frames (t', t), with varying distance between t and t'. Specifically for each t, we used $t' = t - 15k$, for $1 \leq k \leq 5$, because the training videos are annotated every fifteenth frame.

We pre-trained HandLer using static images from the TV-Hand [60] and COCO-Hand [60] datasets by using the same static image as both frames $t-1$ and t. This was to utilize the larger datasets of annotated

hands. Subsequently, we fine-tuned the network on the proposed YouTube-Hand dataset. For fine-tuning, we optimized the training loss for 12K iterations using SGD, with an initial learning rate of 0.0005 and a batch size of 48. We reduced the learning rate by a factor of 10 after 8K iterations.

Evaluation metrics. We used the standard multiple-object-tracking evaluation metrics [8, 51]: the identification F1 score (IDF1), the percentage of mostly tracked trajectories (MT), mostly lost trajectories (ML), false positives (FP), false negatives (FN), identity switches (IDs), multiple object tracking precision (MOTP), multiple object tracking accuracy (**MOTA**) and higher order tracking accuracy (**HOTA**). MOTA is considered the most crucial metric to quantify the overall detection and tracking performance among these evaluation metrics. MOTA is defined as:

$$\text{MOTA} := 1 - \frac{\sum_t (\text{FN}_t + \text{FP}_t + \text{IDs}_t)}{\sum_t \text{GT}_t},$$

where FN_t, FP_t, IDs_t, and GT_t are the number of false negatives, false positives, identity switches, and number of true hands, respectively, for the frame t.

We found that none of the commonly used MOT metrics measures the recovery ability; they do not quantify how well a tracker can right the wrong ID switch by reconnecting a new hand tracklet with a prematurely terminated one. In particular, while the ID switches (IDs) metric measures the fragmentation of a ground truth trajectory, it would apply the same penalty to any identity switch, whether the tracker switches to a new-and-wrong ID or an old-but-correct ID. For example, the sequences of trajectory IDs ($a \rightarrow b \rightarrow c$) and ($a \rightarrow b \rightarrow a$) would have the same performance in current metrics, but the later is more desirable. Thus, we introduce a new metric called Longest-Tracklet-Ratio (**LTR**). For a particular ground truth trajectory matched to multiple predicted tracklets with different IDs, LTR is defined as the ratio between the longest predicted tracklet and the length of the entire trajectory. We will use the average LTR on all trajectories of a test set as the new performance metric.

4.5.2 Main Results

Table 4.3 compares the performance of our hand trackers with other state-of-the-art MOT tracking methods. **TraDes**, **CenterTrack** and **FairMOT**

were end-to-end trainable MOT methods, which were trained to detect and track hands jointly, but they performed relatively poor on hands, perhaps because they were geared towards less deformable and articulated classes such as pedestrians and vehicles.

We also implemented several tracking-by-detection methods, where the detection results were provided by HandCNN [60], which is the state-of-the-art hand detection method. **LightTrack** used pose tracklets to link hands. We first used LightTrack to detect and track human body joints then associated HandCNN detected hand to a person based on the distances between the predicted wrist keypoint and the center of the detected hand bounding box. **CenterTrack*** was a method where HandCNN replaced the detection component of CenterTrack. **MPNTrack** was an offline tracking method using a Message Passing Network (MPN) for HandCNN detection association. For all methods, we first pre-trained using static images from TV-Hand and COCO-Hand datasets to improve the hand detection performance and then fine-tuned them using the training set of YouTube-Hand.

Based on those metrics, HandLer outperforms the others by a wide margin. Fig. 4.4 shows some representative results and failure cases by HandLer.

4.5.3 Ablation Studies

We now present our experiments to study the effectiveness of different components of the proposed architecture.

Effectiveness of HandLer. To study the importance of the proposed forward propagation for hand tracking, we trained a model where there was no forward propagation. Similarly, we trained a model with no backward regression to frame $t-1$. In this case, we linked hand detections using the Hungarian algorithm with the hand bounding boxes in frame t. Finally, we trained and tested the model without pose. The results are shown in Table 4.4. We use **HandLer** to refer to our whole model, and **HandLer-NP** is HandLer without the pose. As can be seen, all those three components are essential components of HandLer.

	FP↓	FN↓	IDs↓	MOTA↑	LTR↑
HandLer	2412	5986	712	70.0	64.3
HandLer w/o forward	3107	6432	761	66.1	62.1
HandLer w/o backward	2838	6195	1488	65.4	58.4
HandLer-NP	2875	6169	1256	66.1	59.0
HandLer-NP w/o forward	3076	6821	1203	63.4	56.4
HandLer-NP w/o backward	2301	6965	1536	64.4	52.2

Table 4.4: **Effectiveness of each component of HandLer.**

Robustness to low frame rates. We studied how the tracking performance changed as the frame rate of a video dropped. For this purpose, we ran HandLer on every K-th frame for various values of K. Specifically, we used $K = 1, 3, 5, 15$, which corresponded to 30, 10, 6, and 2 frames per second (fps). The results are shown in Table 4.5. As can be seen, the MOTA of HandLer did not decrease much when the fps were reduced from 30 to 6.

Compared to SORT [9] (with HandLer detection), another tracking method described in Sec. 4.5.2, this level of MOTA reduction was relatively small. This demonstrates the robustness of our linking algorithm across different time gaps.

Tracking	SORT				HandLer			
Stride	FP↓	FN↓	IDs↓	MOTA↑	FP↓	FN↓	IDs↓	MOTA ↑
$K = 1$	2446	36297	1902	64.9	2412	5986	712	70.0
$K = 3$	1177	2977	2903	59.2	1099	3525	1284	65.8
$K = 5$	915	2297	3301	55.6	906	2861	1468	64.3
$K = 15$	664	1636	3569	51.2	651	2077	1759	62.7

Table 4.5: **Performance of tracking algorithms as the frame rate of videos decreases**. K is the stride of the tracking algorithm.

4.5.4 Hand detection

HandLer can also be used for hand detection if the input is a video. To study the effectiveness HandLer for detecting hands especially blurry and

occluded hands, we sample a subset of YouTube-Hand that only contains blurry and occluded hands to test the effectiveness of HandLer for detecting such hands. Here we use the hand keypoints estimation method proposed in [80] to detect hand keypoints within every ground truth hand box. We claim that the hand is blurry or occludes if [80] cannot detect all hand keypoints. Along with YouTube-Hand and VIVA datasets, we evaluated the performance of various hand detection methods on those three datasets using the VOC average precision metric. Since hand keypoints estimation [80, 108] cannot detect hands well for in-the-wild videos, we compared with HandCNN [60], the state-of-the-art hand detection method and summarize the results of these experiments in Table 4.6. Moreover, using HandLer as a detector (without linker) would also boost the tracking performance of other tracking methods, as can be seen in Table 4.7 for the YouTube-Hand and VIVA datasets. Note that here we only report the performances of the methods that support tracking with external detections.

	Dataset		
Method	YouTube-Hand	Blur&Occ Split	VIVA [71]
HandCNN [60]	72.4	62.8(13.1% ↓)	89.2
HandLer	**84.1**	**76.7**(8.8% ↓)	**95.3**

Table 4.6: **Hand detection performance**. The colored number is the percentage of performance dropped on blurry and occluded hand split compared to the full set of the YouTube-Hand dataset. Compared with HandCNN, which runs with around 2fps, our method achieves both efficiency and effectiveness.

	IDF1↑	IDs↓	MOTA↑	LTR↑
SORT[9]	60.6(+12.3)	1902(+427)	64.9(+20.0)	53.6 (+15.1)
MPNTrack[11]	61.1(+12.1)	1288(+249)	65.2(+25.2)	57.3(+13.0)
CenterTrack[112]	61.3(+24.1)	2167(-1195)	62.7(+22.0)	51.1(+23.8)
LightTrack[63]	71.0(+17.6)	1635(-320)	61.7(+30.9)	65.7(+17.3)

Table 4.7: **Using HandLer as a detector with other MOT methods on YouTube-Hand dataset.** The colored number indicts performance improvement or descent comparing with Table 4.3.

4.5.5 Other datasets & tasks

We also evaluate the tracking and detection performance of HandLer on other datasets: VIVA, and BSL. Note that all methods below use HandLer detection and associate detected hands with their own linker.

The VIVA dataset [71] contains frames sampled from 20 videos captured by ego-centric cameras. It was collected to develop an algorithm to detect the hands of a driver and a passenger. We used 11 videos for training and the remaining 9 for evaluation. The results are shown in Table 4.8.

	IDF1↑	FP↓	FN↓	IDs↓	MOTA↑
CenterTrack[112]	45.6	341	1287	79	68.7
SORT[9]	44.1	517	884	93	72.6
MPNTrack[11]	46.2	793	545	**46**	74.7
HandLer	**62.0**	**272**	**367**	58	**87.2**

Table 4.8: **Comparing different methods on VIVA dataset**

The British Sign Language (BSL) dataset [66] contains 6000 frames from BBC TV shows, 296 of them have been annotated with hand segmentation. All methods reported in Table 4.9 were trained on the YouTube-Hand training set and then tested on the BSL dataset.

	IDF1↑	FP↓	FN↓	IDs↓	MOTA↑
CenterTrack[112]	**22.2**	65	128	177	45.9
MPNTrack[11]	11.6	144	**76**	64	58.8
SORT[9]	13.6	92	82	71	63.2
HandLer	20.5	**60**	89	**39**	**72.5**

Table 4.9: **Tracking performance on the BSL dataset.**

Pose tracking. Since hands are attached to the wrists, one might wonder if we can track the human pose and the wrists instead. We hypothesize that pose tracking is a difficult problem; its performance is not better than hand tracking. We perform experiments on the PoseTrack Split of Youtube-Hands to validate this hypothesis. Pose tracking tracks the wrist points, but comparing point tracking results with bounding box tracking results is not trivial because MOTA computations are done differently. For a fair comparison, we consider two transformations: (1) Box2Point: represent a bounding box by its center; (2) Point2Box: match a wrist point to a detected hand by HandLer as explained in Sec. 4.5.2. Table 4.10 compares the performance of HandLer and LightTrack after making these transformations.

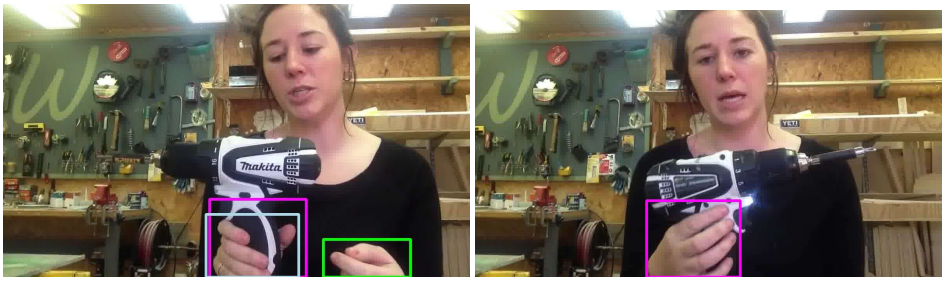
	Box2Point	Point2Box
LightTrack [63]	60.7	49.2
HandLer	**69.6**	**61.2**

Table 4.10: **Comparing with pose tracking algorithm (LightTrack) on the PoseTrack split**. The evaluation metric is MOTA. Pose tracking is a difficult problem and does not perform as well as HandLer.

4.6 Conclusions

In this chapter, we introduced HandLer, a novel convolutional architecture to detect and track hands in unconstrained videos. We also collected and annotated a large-scale challenging hand tracking dataset, YouTube-Hand. This dataset contains videos of hands in unconstrained environments and can be used to develop and evaluate hand tracking systems.

(a) Tracking results by HandLer. This visualizes hand tracking results across two frames. Hands that belong to the same trajectory are visualized with the same color.



(b) Hand detection and backward regression results. The left and right images correspond to frames `t-1` and `t`, respectively. The detected hand in frame `t` and its corresponding location obtained using backward regression in frame `t-1` are shown in magenta color. The detected hands in frame `t-1` are visualized in blue and green.

(c) Comparing HandCNN and HandLer. HandCNN fails to detect blurry and occluded hands. Benefiting from our temporal feature aggregation and tracking-based detection, HandLer can detect those hands.

(d) Failure cases from HandLer. The left image shows a case where a hand is not detected due to heavy occlusions, and the second image shows a case where other skin areas are mistaken for hands.

Figure 4.4: **Qualitative results on YouTube-Hand dataset.**

Chapter 5

Hand-Body Association

In this chapter, we study a new problem of detecting hands and finding the location of the corresponding person for each detected hand. This task is helpful for many downstream tasks such as hand tracking and hand contact estimation. Associating hands with people is challenging in unconstrained conditions since multiple people can be present in the scene with varying overlaps and occlusions.

We propose a novel end-to-end trainable convolutional network that can jointly detect hands and the body location for the corresponding person. Our method first detects a set of hands and bodies and uses a novel Hand-Body Association Network to predict association scores between them. We use these association scores to find the body location for each detected hand. We also introduce a new challenging dataset called BodyHands, containing unconstrained images with hands and their corresponding body location annotations. We conduct extensive experiments on BodyHands and another public dataset to show the effectiveness of our method. Finally, we demonstrate the benefits of hand-body association in two critical applications: hand tracking and hand contact estimation. Our experiments show that hand tracking and hand contact estimation methods can significantly improve by reasoning about the hand-body association.

5.1 Introduction

Hand analysis is an important problem in Computer Vision with applications in human understanding, action, gesture, and sign-language recog-

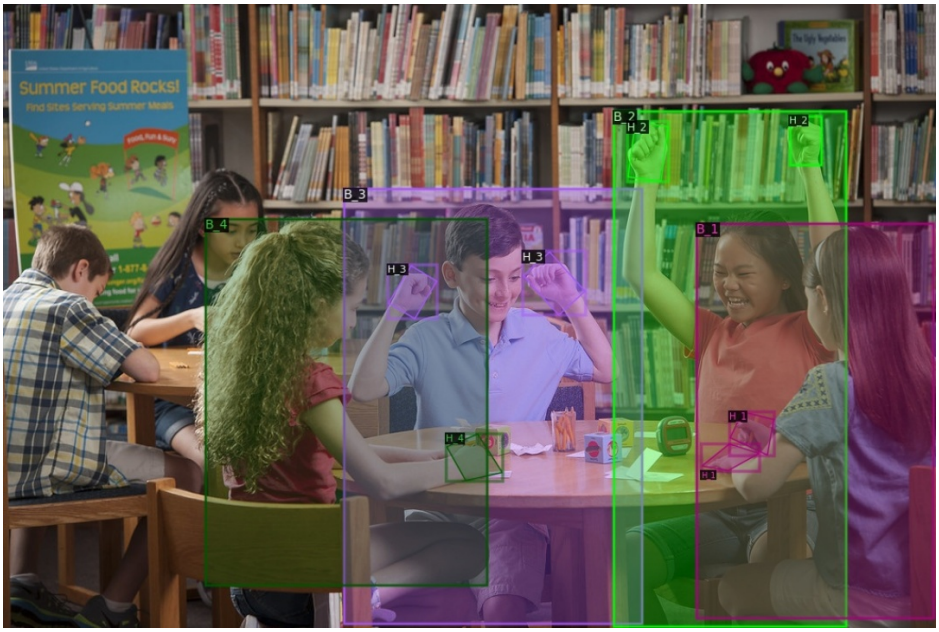


Figure 5.1: **Hand Detection & Hand-Body Association.** We develop a method to detect hands and their corresponding body locations. Hands and bodies belonging to the same person have bounding boxes in the same color and identification numbers.

nition. The visual analysis of human hands is also vital for Augmented & Virtual Reality applications. Although the Computer Vision community has studied problems such as hand detection [12, 44, 45, 54, 64, 67, 99, 114], hand pose estimation [30, 74, 116, 117], hand tracking [78, 82, 83, 90, 108], and hand contact estimation [10, 56, 61, 77], there has been no significant effort in studying hand-body association.

In this work, we study the problem of detecting hands in an image and finding the location of the corresponding person for each detected hand. This task is useful for action recognition and scene understanding, especially for multiple-person images and videos. For example, it is helpful to identify people when understanding hand gestures in human-human communication. Another example is to assess the motor and social skills of children with mental disorders by tracking their hands and how hands interact with objects and other people in a tabletop game. Hand-body association helps develop safety applications and assists people working

with hand-held tools in manufacturing settings.

Detecting hands and associating them with suitable bodies is challenging in unconstrained conditions. As shown in Fig. 5.1, an image may contain multiple people with substantial overlaps and occlusions between hands and bodies. One approach is to detect hands and people separately and use heuristics based on their sizes, distances, or overlapping areas to establish correspondences between hands and bodies. However, such methods do not perform well due to the extreme articulation of hands and bodies, leading to tremendous variations in the relative locations and sizes between a hand and the corresponding human body. An alternative method is to use a human pose detector to find humans' skeletons and the hands of each detected pose in the image. However, pose detection by itself is unreliable. For a scene of congregated or interacting people, the hand and arm of one person might be entangled with the skeleton of another person. Furthermore, the pose detector might not detect poses for everyone in the image, especially for people partially occluded or partly outside the camera's field of view. Thus we cannot solely rely on pose detection to associate hands with people. Our experiments empirically show that pose-based approaches are unreliable for associating hands and bodies.

This work proposes a novel convolutional architecture that can jointly detect hands and bodies and associate them. Specifically, we build upon MaskRCNN [32], a state-of-the-art object detector, and extend it by adding a novel Hand-Body Association Network module. We first use a Region Proposal Network to generate candidate hand and body proposal boxes. We then use the bounding box regression and mask generation heads to obtain the bounding box and segmentation maps for hands and bodies. The detected hands and bodies are then passed to the Hand-Body Association Network to obtain an association between them.

The Hand-Body Association Network has two novel modules. **The first module** is the Overlap Estimation Module that uses the visual features of hands and bodies to estimate if they can overlap. Intuitively, if a hand and a body have no overlap, they cannot belong to the same person. The converse, however, does not hold; a hand and a body can overlap even though they belong to different people. For instance, in the proposed BodyHands dataset, more than 33% of the people have their hands overlapping with other people. The overlap is a piece of mutual geometric information between two regions. Learning mutual geometric information between

hands and bodies using their appearance features allows learning-rich discriminative representations useful for associating hands and bodies. **The second module** is the Positional Density Module that uses hand features to estimate a density over possible body locations for each detected hand. Intuitively, the appearance and location of a hand provide some cues for estimating its body location. However, directly locating the body from the hand can be difficult due to the tremendous variation in relative scales between hands and bodies and mutual occlusions between people. We thus first estimate a density over possible locations and use these density values to find compatible matching for all hand-body pairs using the Hungarian Algorithm.

We also contribute a large-scale dataset of unconstrained images containing annotations for hand locations and their corresponding body locations. The dataset has around 20K images with bounding box annotations for more than 57K hand and 63K body instances. This dataset has numerous images containing multiple people with varying degrees of occlusions and overlap, where it is challenging to detect and associate hands and bodies.

Finally, we demonstrate the benefits of the hand-body association in two crucial downstream tasks: hand tracking and hand physical contact estimation. We show that hand tracking and hand contact estimation methods can be improved by reasoning about the hand-body association.

5.2 Related Work

Hand Analysis. Hands have been extensively studied by the Computer Vision community and there are methods for hand detection [12, 44, 45, 54, 64, 67, 99, 114], hand pose estimation [14, 15, 30, 43, 46, 50, 74, 76, 103, 116, 117], hand tracking [56, 78, 82, 83, 90, 108], and hand contact estimation [10, 58, 61, 77]. However, previous works do not consider the problem of hand-body association. Existing works mostly focus on constrained scenarios such as ego-centric perspectives with a single subject in a video. In such cases, the full body is not always visible, and finding them may not be essential. Some works analyze hands in unconstrained conditions [60, 61, 77] but they do not address this problem either. Zhou et al. [110] address the problem of hand-raiser recognition in classroom scenarios. However, their work is developed for indoor classroom environments

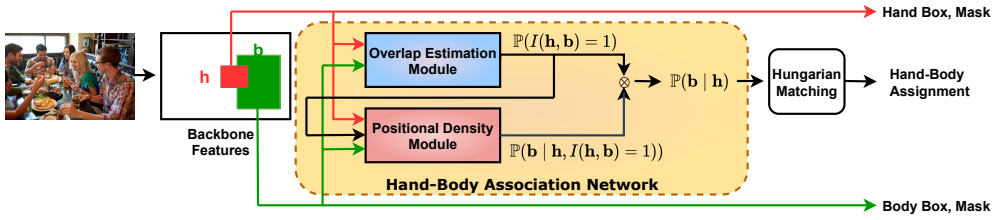


Figure 5.2: **Proposed Architecture.** A ResNet network extracts the backbone features of the input image. We use the feature maps of hand and body proposal boxes to obtain their bounding boxes and binary segmentation masks. The Overlap Estimation Module uses the feature maps of the hand and body to predict if they can overlap, i.e., $\mathbb{P}(I(\mathbf{h},\mathbf{b}) = 1)$. The Positional Density Module uses the hand features and the output from the Overlap Estimation Module to estimate the conditional likelihood $\mathbb{P}(\mathbf{b} \mid \mathbf{h}, I(\mathbf{h},\mathbf{b}) = 1)$. The outputs from these two modules are combined to obtain the likelihood that the body $\mathbf{b}$ belongs to the hand $\mathbf{h}$, i.e., $\mathbb{P}(\mathbf{b}|\mathbf{h})$. We use the estimated conditional likelihood to find compatible matching for all hand-body pairs using the Hungarian Algorithm (used only during inference).

and is unsuitable for unconstrained outside environments. Lee et al. [47] and Tsutsui et al. [88] study the problem of hand disambiguation in egocentric videos. However, they identify only the person's identity, not their body location. On the other hand, we try to address the hand-body association problem, and our work focuses mainly on third-person views.

Hand datasets. Although there are several datasets with annotated hand locations, such as [60, 61], they do not have annotations for the corresponding body locations. Another option would be to use human pose datasets [3, 4, 49], which have human body joint locations. However, such datasets do not have bounding box annotations for hands. Zhou et al. [110] propose a dataset containing hand and body locations, but they develop the dataset in indoor environments. Moreover, their dataset is not publicly available. Bambach et al. [6] propose a dataset containing 48 videos of first-person interactions between two people. However, they provide annotations for only hand locations but not body locations. Jin et al. [41] develop the COCO-WholeBody dataset by annotating hand key points for images from the COCO dataset. Compared to this dataset,

the proposed BodyHands dataset has a higher number of crowded images with significant overlaps and occlusions between people; 34% of people in BodyHands have their hands overlapping with different people compared to 19% from COCO-WholeBody.

5.3 Problem Definition and Proposed Method

This section describes the proposed architecture that can jointly detect hands and bodies and provide an association score between them. We also provide details on the training objective that allows training of the proposed architecture end-to-end. In the following subsection, we will first formally define the problem.

5.3.1 Problem Definition

Given an image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, our goal is to:

1. Detect the bounding box locations $\mathbf{H} = \{\mathbf{h}_i \in \mathbb{R}^4 : 1 \leq i \leq m\}$ and $\mathbf{B} = \{\mathbf{b}_j \in \mathbb{R}^4 : 1 \leq j \leq n\}$ for hands and bodies, respectively. Here, m and n denote the number of hands and bodies in the image $\mathbf{I}$. Each bounding box is represented by a 4-dimensional vector for its left, top, right, bottom locations.

2. For each detected hand $\mathbf{h} \in \mathbf{H}$, we need to associate a body $\mathbf{b} \in \mathbf{B}$ such that the following two constraints are satisfied: (1) each hand $\mathbf{h} \in \mathbf{H}$ is associated with exactly one body $\mathbf{b} \in \mathbf{B}$; (2) each body $\mathbf{b} \in \mathbf{B}$ can be associated to at most two hands in $\mathbf{H}$. Note that we consider any visible regions of the human as the body. Therefore when the detector fails to detect any humans, i.e., $\mathbf{B} = \emptyset$, we treat a hand bounding box as its corresponding person bounding box.

5.3.2 Architecture Overview

We illustrate the proposed architecture in Fig. 2.2. We build upon a two-stage object detector [27, 28, 32, 72] such as MaskRCNN. Given an input image, we use a ResNet to obtain backbone features and a Region Proposal Network to obtain proposals corresponding to two object classes: hands and bodies. We then use the RoIAlign operation to extract features

corresponding to these proposals and perform bounding box regression and mask generation.

For each detected hand $\mathbf{h} \in \mathbf{H}$, we use a novel Hand-Body Association Network to estimate the conditional-likelihood $\mathbb{P}(\mathbf{b}|\mathbf{h})$ over all the detected bodies $\mathbf{b} \in \mathbf{B}$. The conditional $\mathbb{P}(\mathbf{b}|\mathbf{h})$ denotes the probability that the body $\mathbf{b}$ is associated with the hand $\mathbf{h}$. We use $\mathbb{P}(\mathbf{b}|\mathbf{h})$ as weights of a bipartite graph between hands and bodies and pose the hand-body association problem as finding a maximum-weighted assignment satisfying the constraints described by the problem definition in Sec 5.3.1. We finally use the Hungarian algorithm [59] to obtain a solution for this matching problem. We implement the Hand-Body Association Network as a new branch of MaskRCNN and train this module end-to-end together with other MaskRCNN components.

5.3.3 Hand-Body Association Network

The inputs to the Hand-Body Association Network are the set of detected hands $\mathbf{H}$ and bodies $\mathbf{B}$. For each detected hand instance $\mathbf{h} \in \mathbf{H}$, it outputs the conditional-likelihood $\mathbb{P}(\mathbf{b}|\mathbf{h})$ over all bodies $\mathbf{b} \in \mathbf{B}$. The probability $\mathbb{P}(\mathbf{b}|\mathbf{h})$ is high whenever the body $\mathbf{b}$ belongs to the hand $\mathbf{h}$, otherwise it is low. We show that under some independence assumptions, the term $\mathbb{P}(\mathbf{b}|\mathbf{h})$ can be factorized as a product of two terms involving overlap between $\mathbf{h}$ and $\mathbf{b}$ and positional density over $\mathbf{b}$:

$$\mathbb{P}(\mathbf{b}|\mathbf{h}) = \underbrace{\mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 1)}_{\text{overlap between h \& b}} \cdot \underbrace{\mathbb{P}(\mathbf{b}|\mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 1)}_{\text{density over b}}. \tag{5.1}$$

To see this, we first note an important relationship between the hands and body that belong to the same person. Since hands are a part of the human body, a hand bounding box and a body bounding box that belong to the same person must have a positive overlap. In other words, if a hand and a body have no overlap, they cannot belong to the same person. The converse, however, does not hold. Hands and bodies can overlap even though they belong to different people. For instance, in the proposed BodyHands dataset, more than 33% of people have their hands significantly overlapping with other people. Formally, if we let $I_{\mathbf{h},\mathbf{b}}$ be an indicator random variable to denote whether $\mathbf{h}$ and $\mathbf{b}$ have any overlap, we have

$$\mathbb{P}(\mathbf{b}|\mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 0) = 0. \tag{5.2}$$

We can use the law of total probability and condition over possible values of $I_{\mathbf{h},\mathbf{b}} \in \{0, 1\}$ to write:

$$\begin{aligned}\mathbb{P}(\mathbf{b} \mid \mathbf{h}) = \mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 0 \mid \mathbf{h}) \cdot \mathbb{P}(\mathbf{b} \mid \mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 0) \\ + \mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 1 \mid \mathbf{h}) \cdot \mathbb{P}(\mathbf{b} \mid \mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 1),\end{aligned} \tag{5.3}$$

Combining Eq. (5.2) and Eq. (5.3), we get:

$$\mathbb{P}(\mathbf{b} \mid \mathbf{h}) = \mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 1 \mid \mathbf{h}) \cdot \mathbb{P}(\mathbf{b} \mid \mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 1). \tag{5.4}$$

The independence assumption $\mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 1|\mathbf{h}) = \mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 1)$ reduces Eq. (5.4) to Eq. (5.1). We learn the probabilities $\mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 1)$ using the Overlap Estimation Module and $\mathbb{P}(\mathbf{b} \mid \mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 1)$ using the Positional Density Module.

Overlap Estimation Module. This module takes as input the visual features corresponding to the hand bounding box $\mathbf{h}$ and the body bounding box $\mathbf{b}$ and estimates the probability of them overlapping each other. Specifically, we use a neural network $f_{overlap}$ to model $\mathbb{P}(I_{\mathbf{h},\mathbf{b}} = 1) := f_{overlap}(\mathbf{h}, \mathbf{b})$.

We implement $f_{overlap}$ as an additional branch of MaskRCNN using convolutional and fully-connected layers. This network module is computationally light and we learn their parameters together with MaskRCNN during training using the following binary cross-entropy loss:

$$\begin{aligned}\mathcal{L}_{overlap} := &- Y^{(gt)}_{\mathbf{h},\mathbf{b}} \log f_{overlap}(\mathbf{h}, \mathbf{b}) \\ &- \left(1 - Y^{(gt)}_{\mathbf{h},\mathbf{b}}\right) \log\left(1 - f_{overlap}(\mathbf{h}, \mathbf{b})\right).\end{aligned} \tag{5.5}$$

In the above, $Y^{(gt)}_{\mathbf{h},\mathbf{b}}$ denotes the groundtruth and is equal to 1 if $\mathbf{h}$ and $\mathbf{b}$ overlap and 0 otherwise.

Note that we predict $f_{overlap}(\mathbf{h}, \mathbf{b})$ using the appearance features of the hand and the body rather than computing the overlap between bounding boxes $\mathbf{h}$ and $\mathbf{b}$ directly. This is because the overlap is a piece of mutual geometric information between two regions. Learning mutual geometric information between hands and bodies using their appearance features allows learning-rich discriminative representations useful for associating hands and bodies. We show this empirically in our experiments.

Positional Density Module. We use this module to model the term $\mathbb{P}(\mathbf{b} \mid \mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 1)$ in Eq. (5.1). Specifically, given any hand $\mathbf{h}$, for any possible body location $\mathbf{b}$ with $I_{\mathbf{h},\mathbf{b}} = 1$, we model this probability using the following distribution:

$$f_{density}(\mathbf{b}|\mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 1) \propto \exp\left(-\frac{\|\mathbf{b}^{\mathbf{h}} - \mu^{\mathbf{h}}_{body}\|}{2\sigma^2}\right). \tag{5.6}$$

In the above equation, $\mu^{\mathbf{h}}_{body} \in \mathbb{R}^4$ is the mean body location relative to the hand $\mathbf{h}$, $\mathbf{b}^{\mathbf{h}}$ is an encoding of the body box coordinates $\mathbf{b}$ relative to the hand $\mathbf{h}$, and σ is a tunable hyperparameter. More specifically, inspired by the bounding box regression formulation in FasterRCNN [72], we use

$$\mathbf{b}^{\mathbf{h}} = \left(\frac{\mathbf{b}_x - \mathbf{h}_x}{\mathbf{h}_w}, \frac{\mathbf{b}_y - \mathbf{h}_y}{\mathbf{h}_h}, \log\frac{\mathbf{b}_w}{\mathbf{h}_w}, \log\frac{\mathbf{b}_h}{\mathbf{h}_h}\right). \tag{5.7}$$

In the above, $(\mathbf{h}_x, \mathbf{h}_y)$ denotes the (x, y) coordinates of the center of $\mathbf{h}$, $\mathbf{h}_w$ and $\mathbf{h}_h$ denotes the width and height of $\mathbf{h}$. Similarly, $(\mathbf{b}_x, \mathbf{b}_y)$ denotes the (x, y) coordinates of the center of $\mathbf{b}$, $\mathbf{b}_w$ and $\mathbf{b}_h$ denotes the width and height of $\mathbf{b}$. We predict $\mu^{\mathbf{h}}_{body}$ in Eq. (5.6) using the appearance features and bounding box location of the hand $\mathbf{h}$.

Intuitively, the appearance features and location of the hand provide some cues on estimating its body location. However, directly locating the body from hand features can be difficult due to the tremendous variation in relative scales between hands and bodies and mutual occlusions between people. We, therefore, first estimate a density over possible locations and use these density values to find compatible matching for all hand-body pairs using the Hungarian Algorithm. If the body $\mathbf{b}$ is far from the estimated mean body location $\mu^{\mathbf{h}}_{body}$, then $\mathbb{P}(\mathbf{b}|\mathbf{h}, I_{\mathbf{h},\mathbf{b}} = 1)$ is small, and therefore according to Eq. (5.1), $\mathbb{P}(\mathbf{b}|\mathbf{h})$ is also small.

We can efficiently implement the network $f_{density}$ as an additional branch of MaskRCNN using convolutional and fully-connected layers. We train $f_{density}$ together with MaskRCNN end-to-end by minimizing the smooth-$L1$ loss between the predicted $\mu^{\mathbf{h}}_{body}$ and the groundtruth body $\mathbf{b}^{\mathbf{h}}_{(gt)}$ associated with the hand $\mathbf{h}$:

$$\mathcal{L}_{density} := \sum_{i=1}^{4} \text{Smooth-}L_1\left(\mu^{\mathbf{h}}_{body}[i] - \mathbf{b}^{\mathbf{h}}_{(gt)}[i]\right), \tag{5.8}$$

In the above equation, $\mu^{\mathbf{h}}_{body}[i]$ and $\mathbf{b}^{\mathbf{h}}_{(gt)}[i]$ denote the i^{th} components of four dimensional vectors $\mu^{\mathbf{h}}_{body}$ and $\mathbf{b}^{\mathbf{h}}_{(gt)}$.

5.3.4 Training Objective

We train the proposed Hand-Body Association network together with the MaskRCNN end-to-end by optimizing the following multi-task loss:

$$\mathcal{L}_{total} = \mathcal{L}_{cls} + \mathcal{L}_{box} + \mathcal{L}_{mask} + \mathcal{L}_{association}. \tag{5.9}$$

Here, $\mathcal{L}_{cls}$, $\mathcal{L}_{box}$, $\mathcal{L}_{mask}$ denote the classification, the bounding box regression, and the segmentation mask losses for the detection. These are the standard losses used in MaskRCNN [32]. The term $\mathcal{L}_{association}$ denotes the hand-body association loss and is defined as:

$$\mathcal{L}_{association} := \lambda_1 \mathcal{L}_{overlap} + \lambda_2 \mathcal{L}_{density}, \tag{5.10}$$

In the above, $\mathcal{L}_{overlap}$ denotes the loss for the Overlap Estimation Module, and $\mathcal{L}_{density}$ is the loss for the Positional Density Module. These losses are defined in Eq. (5.5) and Eq. (5.8). The scaling factors λ_1 and λ_2 are tunable hyperparameters denoting the relative importance between the overlap estimation and positional density estimation.

5.3.5 Hungarian Hand-Body Assignment

Given a set of detected hands $\mathbf{H} = \{\mathbf{h}_i : 1 \leq i \leq m\}$, bodies $\mathbf{B} = \{\mathbf{b}_j : 1 \leq j \leq n\}$, and the conditional distribution $\mathbb{P}(\mathbf{b}|\mathbf{h})$ estimated from the Hand-Body Association network, we need an assignment strategy to match hands and bodies subject to the constraints described in Sec. 5.3.1.

We follow the bipartite matching strategy and use $\mathbb{P}(\mathbf{b}|\mathbf{h})$ as the weight between hand $\mathbf{h}$ and body $\mathbf{b}$ in the bipartite graph. We obtain a maximum-weighted assignment between the detected hands $\mathbf{H}$ and bodies $\mathbf{B}$ using the Hungarian Algorithm [59].

Note that the Hungarian algorithm matches each hand with exactly one body, but also it produces an undesirable result: each body can match to at most one hand. However, we need the flexibility to match a body to two hands. We provide a simple solution to this by duplicating $\mathbf{B}$ to ensure that each body is present exactly twice before running the Hungarian algorithm. This ensures that a body can have two hands associated with them.

5.4 BodyHands Dataset

This section describes BodyHands, the new dataset collected to develop and evaluate hand-body association methods. BodyHands is a large-scale dataset containing unconstrained images with annotations for hand and body locations and correspondences.

Dataset Source. We built the BodyHands dataset starting from the images from the ContactHands dataset [61]. ContactHands is a large-scale dataset containing unconstrained images annotated with hand polygon locations and their contact states. It has images from popular datasets such as MS COCO [49], PASCAL VOC [23], Oxford-Hand [54], TV-Hand [60], and COCO-Hand [60]. We chose the ContactHands dataset for several reasons. First, we wanted to develop a dataset that we can use to train methods that robustly detect and associate hands and bodies regardless of shape, size, skin tone, and motion blur. Second, we want to detect and associate hands and bodies in challenging cases where people have mutual occlusions. Third, we wanted to use hand-body association in existing applications such as hand contact estimation and thus require contact state annotations. The ContactHands dataset has numerous images which satisfy these requirements.

Annotation and Quality Control. We hired several annotation workers to annotate our dataset. For each person with an annotated hand instance in the ContactHands dataset, we asked an annotator to draw a rectangular bounding box around the person and enter an identification number for the hand and the body. The hands and body which belong to the same person have the same identification number and therefore serve as an association between hands and bodies. We asked the annotators to draw the human bounding box to include all visible parts of the person. If an image contains N people who did not have hand location annotations, we asked annotators to annotate body bounding boxes for all N people if $N \leq 5$ and for at least five people otherwise. This can help us use such human bounding boxes as negative pairs with other hand instances. We also instructed the annotators to ensure that each body has at most two hands associated with it, and also each hand is associated with precisely one body. Thus, every hand instance in our dataset has a body associated with it. When hands are the only visible regions of the person, we use the hand bounding box as the human bounding box. There are some human bounding

Figure 5.3: Representative images from the BodyHands dataset. Hands and bodies belonging to the same person have bounding boxes in the same color and identification numbers.

boxes with no associated hands; this is when hands are occluded or not visible. We collected annotations in batches and manually verified the annotation results ourselves.

Statistics. The BodyHands dataset has 20,490 images with 57,898 annotated polygons for hands and 63,095 axis-parallel rectangular bounding boxes for people. There are 19,810 people with one annotated hand, 19,044 people with two annotated hands, and 24,241 annotated people with no annotated hands (because their hands were either occluded or too small). We use the same training and test splits as the ContactHands dataset to be backward compatible. Fig. 5.3 shows some representative images.

5.5 Experiments

In this section, we describe two sets of experiments. The first experiments analyze the proposed method's hand-body association performance and benchmark it against several other baseline methods. The second set of experiments demonstrates the benefits of hand-body association for hand tracking and hand contact estimation.

5.5.1 Hand-Body Association Experiments

This subsection describes the evaluation metrics used and experimental results for the hand-body association task.

Dataset	BodyHands		
Method	Hand AP	Cond. Accuracy	Joint AP
DOPE	9.09	32.51	2.27
OpenPose	39.69	74.03	27.81
Keypoint Communities	33.62	71.48	20.71
MaskRCNN + Feature Distance	84.82	41.38	23.16
MaskRCNN + Feature Similarity	84.82	39.12	23.30
MaskRCNN + Location Distance	84.82	72.83	50.42
MaskRCNN + IoU	84.82	74.52	51.74
Proposed	84.82	83.44	63.48
Proposed (with hand self-association option)	84.82	84.12	**63.87**

Table 5.1: **Hand detection and hand-body association performance** of several methods evaluated on BodyHands.

Dataset	COCO-WholeBody [41]		
Method	Hand AP	Cond. Accuracy	Joint AP
DOPE	15.02	47.56	9.09
OpenPose	30.22	82.97	18.65
Keypoint Communities	44.39	87.91	40.89
MaskRCNN + Feature Distance	75.92	59.97	38.44
MaskRCNN + Feature Similarity	75.92	53.60	33.72
MaskRCNN + Location Distance	75.92	78.47	50.92
MaskRCNN + IoU	75.92	79.53	53.08
Proposed	75.92	88.05	62.87
Proposed (with hand self-association option)	75.92	88.69	**62.92**

Table 5.2: **Hand detection and hand-body association performance** of several methods evaluated on COCO-WholeBody.

Evaluation Metrics

We measure the hand detection performance using the standard VOC Average Precision (AP) metric. To measure the hand-body association per-

formance, we consider two metrics: **(1) Conditional Accuracy** for body association. We define this as the percentage of correctly associated bodies among the correctly detected hand instances. Here we define that a body is correctly associated with the hand if the Intersection over Union (IoU) between the associated body box and the corresponding ground truth body box is greater than 0.5. We call this conditional accuracy since we only consider associated bodies corresponding to the correctly detected hand instances. Note that hand detection is correct if the IoU between the detected hand bounding box and a ground truth bounding box is greater than 0.5. **(2) Joint AP** for hand detection and body association. In this metric, a detected hand is considered a true positive if: (a) the Intersection over Union (IoU) between the bounding box of the detected hand and a ground truth hand bounding box is greater than 0.5, and (b) the Intersection over Union (IoU) between the body bounding box associated with the detected hand instance and the ground truth body bounding box is greater than 0.5.

Competing Methods and Comparison Results

We conducted several experiments to measure the hand-body association performance and compare them to the proposed method. Note that in the proposed method variant with an option to match the detected hand to itself, we allow the hand box to be its corresponding body box when running the Hungarian Algorithm. We summarize the results in Table 5.1 and Table 5.2. The proposed method outperforms other methods by a significant margin. We describe the methods in the comparison below.

2D Human Pose. We run different 2D pose estimation methods such as OpenPose [13, 80, 93], Keypoint Communities [106] and DOPE [94] to obtain hand keypoints and body joints. We obtain the hand bounding boxes and corresponding body bounding boxes using these keypoints and joints. We use a less-stricter evaluation protocol since the detected hand keypoints can be very noisy: we consider a hand to be a true positive if its bounding box has positive IoU with a ground-truth bounding box. These methods do not perform well since obtaining accurate hand and body pose in unconstrained conditions is challenging.

MaskRCNN + X. We train MaskRCNN using a ResNet101 backbone to detect hands and bodies. We then use the Hungarian matching algorithm to

match hands to bodies using several cost functions: (1) **Feature Distance** first extracts MaskRCNN's box regression 1024-dimensional feature vectors for hands and bodies and then uses the L_2 distance between these feature vectors; (2) **Feature Similarity** first extracts MaskRCNN's box regression 1024-dimensional feature vectors for hands and bodies, and then uses the inner product between these feature vectors; (3) **Location Distance** uses the L_2 distance between the center of the detected hand and body bounding boxes; (4) **IoU** uses the Intersection over Union (IoU) overlap between the detected hand and body bounding boxes.

Ablation Studies. We conduct ablation studies to study the effects of different components of the proposed method. Specifically, we train three different models using the training set of BodyHands: (1) the proposed method without the Overlap Estimation Module; (2) the proposed method without the Positional Density Module; and (3) the proposed method using overlap computed from hand and bounding boxes instead of Overlap Estimation Module. The Joint AP on the BodyHands test set of these methods are 59.03%, 50.29%, and 60.34 %, respectively. These results show that both the overlap estimation module and the positional density module are helpful for the hand-body association.

Qualitative Results. Fig. 5.4 shows some qualitative results and failure cases from our method. Failure cases are mainly due to incorrect hand detections and false body association, especially in crowded images.

5.5.2 Benefits of Hand-Body Association

The ability to associate each detected hand with a human body is beneficial for many downstream tasks. This subsection demonstrates the benefits of this ability for two such tasks: hand tracking and hand contact estimation.

Hand-Body Association for Hand Tracking

Hand tracking is essential with many applications, including gesture recognition and skill evaluation. We hypothesize that the ability to associate hands with human bodies can improve tracking results. Intuitively, by associating hands with human bodies and linking human bodies across frames, we can establish correspondence between detected instances of

Figure 5.4: **Qualitative results and failure cases.** We visualize hands and bodies that belong to the same person using the same color and identification numbers.

the same hand across different frames, reducing identity switches in tracking.

Proposed hand tracking method and other baselines. Tracking hands is a multi-object tracking (MOT) problem, and a popular approach to address this problem is tracking by detection. This approach consists of two main steps: (1) detecting hands in individual video frames and (2) linking the detected hands between frames to form hand tracks. We adopt this tracking by detection approach in this work. For detection, we use our network trained for hand detection and hand-body association. For linking, we use the Hungarian algorithm [59] to optimize for the best set of one-to-at-most-one correspondence between a set of detected hands in

frame t and a set of previously established hand tracks up until frame $t-1$. The matching outcome by the Hungarian algorithm depends on the affinity/cost matrix that defines the compatibility/cost for matching a hand to a hand tracklet. A popular approach is to define the affinity based on the Intersection over the Union (IoU) value between two detected objects (i.e., hands in this case). We will refer to this as the **Hand-IoU** baseline. However, hands are fast-moving objects, and the location and size of a hand can change drastically from one frame to the next. Thus, linking hands using Hand-IoU leads to incorrect identity switches in many cases. We consider a simple approach for linking based on hand-body association that treats a hand-and-body pair as a single identity. We define their affinity for two hand-body pairs detected at two different frames based on the weighted sum of the hand IoU and the body IoU. We refer to this method as **Hand-&-Body-IoU**. We also consider several other linking methods as follows. In **Re-ID**, the matching cost between two detected hands is defined based on the distance between the corresponding embedding vectors. In **Pose-based**, we use LightTrack [63] to detect and track skeleton keypoints and associate each detected hand instance to a skeleton based on the distances between the predicted wrist keypoint and the center of the detected hand bounding box. **Flow-based** is the method that uses optical flows to link detections. Here, we use the average optical flow for pixels inside the detected object to link it with detection in the previous frame.

Evaluation dataset. There were no publicly available datasets for tracking hands in unconstrained environments. Most of the existing datasets [18, 56, 66, 71] for hand tracking were captured in constrained environments such as ego-centric perspectives and contained only one or two hands. To evaluate hand tracking methods in unconstrained conditions, we collected 20 videos from YouTube and manually annotated hand bounding boxes and their trajectories. Specifically, we annotated every 15 frames, and altogether the dataset has 3299 annotated frames, 8893 hand instances, and 131 hand trajectories. We call this dataset YoutubeHands-20, and this dataset has many videos that contain multiple people interacting in the scene, so tracking hands in such cases is challenging. YoutubeHands-20 has now been expanded to a larger dataset YoutubeHands containing 200 videos [38].

Evaluation metric. To evaluate hand tracking performance, we use the standard multi-object-tracking evaluation metrics [8]: False Positives (FP),

	FP↓	FN↓	IDs↓	MOTA↑
FairMOT [107]	412	3859	114	8.6
CenterTrack [112]	376	3909	2	10.7
MPNTrack [11] (offline)	1192	1074	545	41.4
CenterTrack (our detection)	458	1553	750	42.5
Re-ID	681	1284	817	42.0
Hand-IoU	681	1284	624	46.1
Flow-based	681	1284	882	40.7
Pose-based	681	1284	591	46.8
Hand-&-Body-IoU (proposed)	681	1284	436	**50.0**

Table 5.3: **Hand tracking results.**

False Negatives (FN), Identity Switches (IDs), and Multiple Object Tracking Accuracy (MOTA). MOTA is the combined metric, and it is considered the most crucial metric to quantify the overall detection and tracking performance.

Tracking results. Table 5.3 compares the tracking results of all methods. CenterTrack [112] and FairMot [107] are end-to-end methods in which object detection and association are performed together. To the best of our knowledge, there is no publicly available large-scale hand tracking datasets to train these methods. We do our best to train these two methods: first, we use static images from TVHand [60] and COCOHand [60] datasets to pre-train these methods. We then use the VIVAHandTracking [71] dataset to finetune them to perform hand tracking. We also conducted experiments by replacing the detection component of CenterTrack with the proposed hand detector. MPNTrack [11] is an offline tracking method. We pre-train MPNTrack on the VIVA [71] dataset and then use hands detected from our method as inputs to the tracker. These methods do not work well on hands, perhaps because they are geared towards less deformable classes such as pedestrians and vehicles.

The methods Re-ID, Hand-IoU, Flow-based, Pose-based, and Hand-&-Body-IoU use the same hand detector, so they have the same FP and FN. The main differences are how we link the detected hands into tracks. As seen, using both hands and bodies for linking yields the highest MOTA.

Hand-Body Association for Hand Contact Analysis

We now demonstrate the benefits of hand-body association for recognizing the physical contact state of a hand, which could be: (1) No-Contact, (2) Self-Contact, (3) Person-Contact, and (4) Object-Contact. These conditions are not mutually exclusive, and a hand can be in more than one state. Recognizing the physical contact states of hands has many applications in human understanding, augmented reality, and virtual reality.

Contact state recognition is a complex problem in general, and the most challenging category to recognize is Person-Contact, with the current state-of-the-art result being 39.51% Average Precision (AP) [61]. This is due to the difficulty of distinguishing between Person-Contact and Self-Contact. The visual appearance of a hand and its surrounding local context can determine if the hand is touching a body part. However, it is not easy to know if this body part is part of the same person (Self-Contact) or a different person (Person-Contact). Next, we will describe two approaches to improve the performance of Person-Contact recognition by reasoning about the hand-body association.

Heuristic method. We consider a simple post-processing heuristic to improve the performance of an off-the-shelf contact estimation network [61] as follows. Given a detected hand H and its corresponding person-contact score s obtained by running the pre-trained hand-contact network of [61], our simple heuristic method will adjust s while leaving the scores of other contact states unchanged. First, we use the hand-body association network developed in this paper to detect hands and obtain the associated human bodies for each detected hand; let $\{(A_i, B_i)\}$ denote the set of hand-body pairs obtained. If H does not overlap with any A_j, we will terminate this process and leave the person-contact score s unchanged. Otherwise, we will associate H with A_j which has the highest IoU with H, and subsequently associate H to the body B_j. Second, we use a pre-trained MaskRCNN [32] to detect all people in the image; let $\mathcal{P}$ denote this set. We then associate the body B_j with the person $P_k \in \mathcal{P}$ with the highest IoU with B_j. Third, we consider all detected people in $\mathcal{P}$ different from P_k and determine the overlapping region between them and the hand H. If none of the overlapping regions is larger than 15% of the hand area, we heuristically determine that this hand has a low probability of contact with another person. We then decrease the person-contact score using the formula: $s^{new} = \max(s - 0.5, 0)$. This heuristic improves the average preci-

	NC	SC	OC	PC	mAP
Previous SoTA [61]	62.48	54.31	73.34	39.51	57.41
Leveraging hand-body association					
Heuristic	62.48	54.31	73.34	40.89	57.56
End-to-end	64.74	56.12	74.32	47.09	60.56

Table 5.4: **Hand contact estimation results.** The states NC, SC, PC, and OC denotes No-Contact, Self-Contact, Person-Contact, and Object-Contact, respectively. We can advance the state-of-the-art by leveraging the ability to associate detected hands with bodies.

sion for detecting another person's contact from 39.51% to 40.89%. This heuristic is simple, but it is only possible because we have a network that tells us who is the self person among the set of detected people. Note that this heuristic only adjusts the person-contact score.

End-to-end method. We build a new architecture that extends the proposed method in Sec. 5.3 with an additional branch to estimate the contact state of a detected hand. The inputs to this new branch are the RoI feature maps of the detected hand and the corresponding body. We concatenate the RoI features and use fully-connected layers to obtain the contact state scores for the hand. We train this new architecture end-to-end using the following multi-task loss:$\mathcal{L} := \mathcal{L}_{cls} + \mathcal{L}_{box} + \mathcal{L}_{mask} + \mathcal{L}_{association} + \mathcal{L}_{contact}$. The losses $\mathcal{L}_{cls}, \mathcal{L}_{box}, \mathcal{L}_{mask}, \mathcal{L}_{association}$ are the same as described in Eq. (5.9). The term $\mathcal{L}_{contact}$ is the loss of contact state of the hand. Following [61], we define $\mathcal{L}_{contact}$ to be the sum of four independent binary cross-entropy losses corresponding to four possible contact states. We train this architecture on the training set of ContactHands [61] and evaluate its performance on the test set of ContactHands. This method improves the AP for Person-contact from 39.51% to 47.09%. We summarize the results in Table 5.4.

5.6 Conclusions

In this chapter, we investigated an important problem of detecting hands and bodies and associating them with their corresponding person. We introduced a novel architecture based on MaskRCNN, and we also collected a large-scale dataset of images annotated with hand locations and corre-

sponding body locations. Finally, we demonstrated the benefits of this new problem in two tasks, hand sutracking and hand contact estimation.

Chapter 6

Proposal: Hand Active Object Detection and Tracking

In this chapter, we provide some future directions we propose to explore in this thesis.

When humans perform activities, they interact with several objects. For example, consider a person preparing coffee. Preparing coffee is a complex process and involves many smaller actions, and the person has to interact with several objects. For example, they have to use objects such as *dripper*, *filter*, *water*, *kettle*, *coffee beans*, *mugs*, and *coffee grinder*. The person has to perform actions ranging from 'folding the filter paper,' 'putting the dripper on the coffee mug,' 'boiling the water,' 'grounding the coffee beans, 'pouring the ground coffee to the dripper,' and 'pour the water to the dripper.' For a computer vision system trying to recognize actions, it is beneficial to detect and recognize the **active** objects.

Active objects are those that are important and are used for ongoing tasks. For example, consider a person grinding the coffee. They have to scoop the coffee beans using a spoon and grind them. The active objects here are *spoon*, *coffee beans can*, and *coffee grinder*. Note that the active object might not be in contact with the hand. Given an egocentric video clip of a person performing activities, we propose to detect active objects at each time frame. In addition, we also propose to track each detected active object throughout the video.

Recognizing active objects provides cues to recognize ongoing actions and is crucial for other computer vision tasks such as the episodic memory proposed in [29]. The episodic memory addresses the task of locat-

ing where the object was seen previously. This helps the user locate the objects they were interacting with in the past. Recognizing active objects also helps address a recently proposed task, **Where Did This Come From? (WDTCF)** [20]. For example, consider a person using coffee beans, then the WDTCF task aims to trace the object *coffee beans* back in time to identify where they came from. One possible answer for this is *the coffee beans were taken out from the cupboard*.

Problem Definition. Given a video sequence $I_1, I_2, \cdots I_n$ consisting of n frames, we propose to detect hands and the active objects for each frame I_j for $1 \leq j \leq n$. Furthermore, for each detected active object in frame I_j, we propose to obtain their corresponding locations in frames I_k for $j+1 \leq k \leq n$.

Datasets. To address the active object detection and tracking, we propose to use recently published datasets such as Ego4D [29] and Epic-Kitchens-VISOR [20]. Ego4D is a large-scale dataset spanning 3670 hours of daily-life activities spanning hunderds of scenarios. They provide annotations for hands and objects using bounding boxes. VISOR introduces pixel-wise annotations for hands and active objects for videos collected from the Epic-Kitchens dataset [19]. We propose to use these two datasets to address active object tracking. While VISOR has good quality pixel-wise annotations, the videos are limited to kitchen activities and do not include other daily scenarios. Ego4D has videos for several scenarios encountered in our daily activities. However, the annotations for hands and objects are sparse. Specifically, they only have annotations for three frames per action and therefore not sufficient for developing methods to train robust active object tracking methods. To overcome this, we propose to annotate part of the Ego4D more densely. In addition, we also plan to capture videos for some scenarios not present in Ego4D.

Bibliography

[1] W. Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. https://github.com/matterport/Mask_RCNN, 2017. 20

[2] S. Aliakbarian, P. Cameron, F. Bogo, A. Fitzgibbon, and T. J. Cashman. Flag: Flow-based 3d avatar generation from sparse observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 6

[3] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 71

[4] M. Andriluka, U. Iqbal, E. Insafutdinov, L. Pishchulin, A. Milan, J. Gall, and B. Schiele. Posetrack: A benchmark for human pose estimation and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 71

[5] J. Ba, V. Mnih, and K. Kavukcuoglu. Multiple object recognition with visual attention. In *Internation Conference on Learning Representations*, 2015. 32

[6] S. Bambach, S. Lee, D. Crandall, and C. Yu. Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 4, 20, 32, 58, 71

[7] P. Bergmann, T. Meinhardt, and L. Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 48, 50

[8] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008. 60, 83

[9] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *IEEE International Conference on Image Processing (ICIP)*, 2016. 50, 59, 62, 64, 65

[10] S. Brahmbhatt, C. Tang, C. D. Twigg, C. C. Kemp, and J. Hays. ContactPose: A dataset of grasps with object contact and hand pose. In *The European Conference on Computer Vision (ECCV)*, 2020. 68, 70

[11] G. Brasó and L. Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 48, 59, 64, 65, 84

[12] P. Buehler, M. Everingham, D. P. Huttenlocher, and A. Zisserman. Long term arm and hand tracking for continuous sign language tv broadcasts. In *Proceedings of the British Machine Vision Conference*, 2008. 3, 4, 9, 31, 50, 68, 70

[13] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 80

[14] Z. Cao, I. Radosavovic, A. Kanazawa, and J. Malik. Reconstructing hand-object interactions in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 70

[15] Y.-W. Chao, W. Yang, Y. Xiang, P. Molchanov, A. Handa, J. Tremblay, Y. S. Narang, K. Van Wyk, U. Iqbal, S. Birchfield, J. Kautz, and D. Fox. Dexycb: A benchmark for capturing hand grasping of objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 70

[16] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng. Aˆ2-nets: Double attention networks. In *Advances in Neural Information Processing Systems*, 2018. 10, 32

[17] H. Cooper and R. Bowden. Large lexicon detection of sign language. In

International Workshop on Human-Computer Interaction, 2007. 3, 9

[18] D. Damen, H. Doughty, G. M. Farinella, A. Furnari, J. Ma, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Rescaling egocentric vision. *CoRR*, abs/2006.13256, 2020. x, 56, 58, 83

[19] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2021. 89

[20] A. Darkhalil, D. Shan, B. Zhu, J. Ma, A. Kar, R. Higgins, S. Fidler, D. Fouhey, and D. Damen. Epic-kitchens visor benchmark: Video segmentations and object relations. In *Proceedings of the Neural Information Processing Systems (NeurIPS) Track on Datasets and Benchmarks*, 2022. 89

[21] X. Deng, Y. Zhang, S. Yang, P. Tan, L. Chang, Y. Yuan, and H. Wang. Joint hand detection and rotation estimation using cnn. *Transactions on Image Processing*, 2018. 3, 4, 10, 21, 31, 50

[22] L. Duan, M. Shen, S. Cui, Z. Guo, and O. Deussen. Estimating 2d multi-hand poses from single depth images. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018. 3, 9

[23] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. In *International Journal on Computer Vision*, 2015. 7, 21, 39, 77

[24] C. Feichtenhofer, A. Pinz, and A. Zisserman. Detect to track and track to detect. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 51

[25] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. In *Transactions on Pattern Analysis and Machine Intelligence*, 2010. 8

[26] R. Girdhar and D. Ramanan. Attentional pooling for action recognition. In *Advances in Neural Information Processing Systems*, 2017. 32

[27] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierar-

chies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 21, 31, 72

[28] R. Girshick, F. Iandola, T. Darrell, and J. Malik. Deformable part models are convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 21, 31, 72

[29] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, M. Martin, T. Nagarajan, I. Radosavovic, S. K. Ramakrishnan, F. Ryan, J. Sharma, M. Wray, M. Xu, E. Z. Xu, C. Zhao, S. Bansal, D. Batra, V. Cartillier, S. Crane, T. Do, M. Doulaty, A. Erapalli, C. Feichtenhofer, A. Fragomeni, Q. Fu, A. Gebreselasie, C. González, J. Hillis, X. Huang, Y. Huang, W. Jia, W. Khoo, J. Kolár, S. Kottur, A. Kumar, F. Landini, C. Li, Y. Li, Z. Li, K. Mangalam, R. Modhugu, J. Munro, T. Murrell, T. Nishiyasu, W. Price, P. Ruiz, M. Ramazanova, L. Sari, K. Somasundaram, A. Southerland, Y. Sugano, R. Tao, M. Vo, Y. Wang, X. Wu, T. Yagi, Z. Zhao, Y. Zhu, P. Arbeláez, D. Crandall, D. Damen, G. M. Farinella, C. Fuegen, B. Ghanem, V. K. Ithapu, C. V. Jawahar, H. Joo, K. Kitani, H. Li, R. Newcombe, A. Oliva, H. S. Park, J. M. Rehg, Y. Sato, J. Shi, M. Z. Shou, A. Torralba, L. Torresani, M. Yan, and J. Malik. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18995–19012, June 2022. ii, 88, 89

[30] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid. Learning joint reconstruction of hands and manipulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4, 31, 32, 68, 70

[31] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 59

[32] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 8, 10, 11, 30, 31, 37, 69, 72, 76, 85

[33] M. Hoai and A. Zisserman. Thread-safe: Towards recognizing human ac-

tions across shot boundaries. In *Proceedings of the Asian Conference on Computer Vision*, 2014. 13, 14

[34] M. Hoai and A. Zisserman. Talking heads: Detecting humans and recognizing their interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 12

[35] T. Hoang Ngan Le, Y. Zheng, C. Zhu, K. Luu, and M. Savvides. Multiple scale faster-rcnn approach to driver's cell-phone usage and hands on steering wheel detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2016. 3, 9

[36] X. Hou and d Liqing Zhang. Dynamic visual attention: Searching for coding length increments. In *Advances in Neural Information Processing Systems*, 2008. 32

[37] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 10

[38] M. Huang, S. Narasimhaswamy, S. Vazir, H. Ling, and M. Hoai. Forward propagation, backward regression and pose association for hand tracking in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. iii, 83

[39] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 52

[40] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, 2015. 21

[41] S. Jin, L. Xu, J. Xu, C. Wang, W. Liu, C. Qian, W. Ouyang, and P. Luo. Whole-body human pose estimation in the wild. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 71, 79

[42] L. Karlinsky, M. Dinerstein, D. Harari, and S. Ullman. The chains model for detecting parts by their context. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 3, 4, 9, 31, 50

[43] D. U. Kim, K. I. Kim, and S. Baek. End-to-end detection and pose estimation of two interacting hands. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 70

[44] M. Kölsch and M. Turk. Robust hand detection. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, 2004. 3, 4, 9, 31, 50, 68, 70

[45] M. P. Kumar, A. Zisserman, and P. H. Torr. Efficient discriminative learning of parts-based models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2009. 3, 4, 9, 31, 50, 68, 70

[46] T. Kwon, B. Tekin, J. Stühmer, F. Bogo, and M. Pollefeys. H2o: Two hands manipulating objects for first person interaction recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 70

[47] S. Lee, S. Bambach, D. J. Crandall, J. M. Franchak, and C. Yu. This hand is my hand: A probabilistic approach to hand disambiguation in egocentric video. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014. 71

[48] Y. Li, Z. Ye, and J. M. Rehg. Delving into egocentric actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 20, 58

[49] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft COCO: Common objects in context. In *The European Conference on Computer Vision (ECCV)*, 2014. 13, 16, 30, 39, 71, 77

[50] S. Liu, H. Jiang, J. Xu, S. Liu, and X. Wang. Semi-supervised 3d hand-object poses estimation with interactions in time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 70

[51] J. Luiten, A. Osep, P. Dendorfer, P. Torr, A. Geiger, L. Leal-Taixé, and B. Leibe. Hota: A higher order metric for evaluating multi-object tracking. In *International Journal on Computer Vision*, 2021. 60

[52] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, X. Zhao, and T. Kim. Mul-

tiple object tracking: A literature review. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 48

[53] S. R. Malireddi, F. Mueller, M. Oberweger, A. K. Bojja, V. Lepetit, C. Theobalt, and A. Tagliasacchi. Handseg: A dataset for hand segmentation from depth images. In *ArXiv: abs/1711.05944*, 2017. 20, 58

[54] A. Mittal, A. Zisserman, and P. H. S. Torr. Hand detection using multiple proposals. In *Proceedings of the British Machine Vision Conference*, 2011. 3, 4, 9, 15, 20, 21, 31, 39, 50, 58, 68, 70, 77

[55] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, 2014. 32

[56] F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt. Real-time hand tracking under occlusion from an egocentric rgb-d sensor. In *Proceedings of the IEEE/ CVF International Conference on Computer Vision Workshops (ICCVW)*, 2017. x, 56, 58, 68, 70, 83

[57] F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt. Ganerated hands for real-time 3d hand tracking from monocular rgb. In *Proceedings of the IEEE/ CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5, 50, 58

[58] L. Müller, A. A. A. Osman, S. Tang, C.-H. P. Huang, and M. J. Black. On self-contact and human pose. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 70

[59] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5(1):32–38, March 1957. 50, 54, 73, 76, 82

[60] S. Narasimhaswamy, Z. Wei, Y. Wang, J. Zhang, and M. Hoai. Contextual attention for hand detection in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. ii, 32, 39, 50, 58, 59, 61, 63, 70, 71, 77, 84

[61] S. Narasimhaswamy, T. Nguyen, and M. Hoai. Detecting hands and recognizing physical contact in the wild. In *Advances in Neural Information*

Processing Systems, 2020. iii, 58, 68, 70, 71, 77, 85, 86

[62] S. Narasimhaswamy, T. Nguyen, M. Huang, and M. Hoai. Whose hands are these? hand detection and hand-body association in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. iii

[63] G. Ning and H. Huang. Lighttrack: A generic framework for online top-down human pose tracking. In *Proceedings of CVPR Workshop on Towards Human-Centric Image/Video Synthesis and the 4th Look Into Person Challenge*, 2020. 52, 55, 59, 64, 65, 83

[64] E.-J. Ong and R. Bowden. A boosted classifier tree for hand shape detection. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 2004. 3, 4, 9, 31, 50, 68, 70

[65] J. Peng, C. Wang, F. Wan, Y. Wu, Y. Wang, Y. Tai, C. Wang, J. Li, F. Huang, and Y. Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In *The European Conference on Computer Vision (ECCV)*, 2020. 51

[66] T. Pfister, J. Charles, M. Everingham, and A. Zisserman. Automatic and efficient long term arm and hand tracking for continuous sign language tv broadcasts. In *Proceedings of British Machine Vision Conference (BMVC)*, 2012. x, 56, 58, 64, 83

[67] P. K. Pisharady, P. Vadakkepat, and A. P. Loh. Attention based detection and recognition of hand postures against complex backgrounds. *Internation Journal on Computer Vision*, 2013. 3, 9, 50, 68, 70

[68] H. Possegger, T. Mauthner, P. M. Roth, and H. Bischof. Occlusion geodesics for online multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 50

[69] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 58

[70] Z. Qin and C. R. Shelton. Improving multi-target tracking via social grouping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pat-*

tern Recognition (CVPR), 2012. 50

[71] A. Rangesh, E. Ohn-Bar, M. M. Trivedi, et al. Driver hand localization and grasp analysis: A vision-based real-time approach. In *International Conference on Intelligent Transportation Systems*, 2016. x, 56, 58, 63, 64, 83, 84

[72] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015. 11, 21, 31, 72, 75

[73] P. Rodríguez, J. M. Gonfaus, G. Cucurull, F. X. Roca, and J. Gonzàlez. Attend and rectify: a gated attention mechanism for fine-grained recovery. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2018. 32

[74] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 2017. 4, 31, 68, 70

[75] K. Roy, A. Mohanty, and R. R. Sahay. Deep learning based hand detection in cluttered environment using skin segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, 2017. 3, 4, 9, 21, 31, 50

[76] V. Rudnev, V. Golyanik, J. Wang, H.-P. Seidel, F. Mueller, M. Elgharib, and C. Theobalt. Eventhands: Real-time neural 3d hand pose estimation from an event stream. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 70

[77] D. Shan, J. Geng, M. Shu, and D. F. Fouhey. Understanding human hands in contact at internet scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. xiii, 32, 42, 44, 58, 68, 70

[78] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, E. Krupka, A. Fitzgibbon, S. Izadi, and P. Kohli. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015. 5, 50, 68, 70

[79] R. Shilkrot, S. Narasimhaswamy, S. Vazir, and M. Hoai. WorkingHands: A hand-tool assembly dataset for image segmentation and activity mining. In *Proceedings of British Machine Vision Conference*, 2019. 4, 20, 31, 58

[80] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 16, 21, 22, 43, 63, 80

[81] A. Spurr, J. Song, S. Park, and O. Hilliges. Cross-modal deep variational hand pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4, 31

[82] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt. Fast and robust hand tracking using detection-guided optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5, 50, 68, 70

[83] S. Sridhar, F. Mueller, M. Zollhoefer, D. Casas, A. Oulasvirta, and C. Theobalt. Real-time joint tracking of a hand manipulating an object from rgb-d input. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 50, 68, 70

[84] O. Taheri, V. Choutas, M. J. Black, and D. Tzionas. Goal: Generating 4d whole-body motion for hand-object grasping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 6

[85] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, 2019. 59

[86] B. Tekin, F. Bogo, and M. Pollefeys. H+O: unified egocentric recognition of 3d hand-object pose and interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4, 32

[87] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. In *ACM Transactions on Graphics*, 2014. 20, 58

[88] S. Tsutsui, Y. Fu, and D. J. Crandall. Whose hand is this? person identifica-

tion from egocentric hand gestures. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021. 71

[89] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. 32

[90] R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. *ACM Transactions on Graphics*, 2009. 5, 50, 68, 70

[91] X. Wang, R. Girshick, A. Gupta, and K. He. Non local neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 10, 32

[92] Z. Wang, L. Zheng, Y. Liu, and S. Wang. Towards real-time multi-object tracking. In *arXiv preprint arXiv:1909.12605*, 2019. 48

[93] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 80

[94] P. Weinzaepfel, R. Brégier, H. Combaluzier, and G. Leroy, Vincentand Rogez. Dope: Distillation of part experts for whole-body 3d pose estimation in the wild. In *Proceedings of European Conference on Computer Vision*, 2020. 80

[95] A. Wetzler, R. Slossberg, and R. Kimmel. Rule of thumb: Deep derotation for improved fingertip detection. In *Proceedings of British Machine Vision Conference*, 2015. 20, 58

[96] N. Wojke and A. Bewley. Deep cosine metric learning for person re-identification. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2018. 48

[97] N. Wojke, A. Bewley, and D. Paulus. Simple online and realtime tracking with a deep association metric. In *IEEE International Conference on Image Processing (ICIP)*, 2017. 48

[98] J. Wu, J. Cao, L. Song, Y. Wang, M. Yang, and J. Yuan. Track to detect and segment: An online multi-object tracker. In *Proceedings of the IEEE/CVF*

International Conference on Computer Vision (ICCV), 2021. 51, 59

[99] Y. Wu, Q. Liu, and T. S. Huang. An adaptive self-organizing color segmentation algorithm with application to robust real-time human hand localization. In *Proceedings of the Asian Conference on Computer Vision*, 2000. 3, 4, 9, 31, 50, 68, 70

[100] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. 40, 59

[101] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 50

[102] B. Yang, C. Huang, and R. Nevatia. Learning affinities and dependencies for multi-target tracking using a crf model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 50

[103] L. Yang, S. Chen, and A. Yao. Semihand: Semi-supervised hand pose estimation with consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 70

[104] Yichen Wei, Jian Sun, Xiaoou Tang, and Heung-Yeung Shum. Interactive offline tracking for color objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2007. 50

[105] J. H. Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon. Online multi-object tracking via structural constraint event aggregation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 50

[106] D. Zauss, S. Kreiss, and A. Alahi. Keypoint communities. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 80

[107] Y. Zhan, C. Wang, X. Wang, W. Zeng, and W. Liu. A simple baseline for multi-object tracking. In *arXiv preprint arXiv:2004.01888*, 2020. 59, 84

[108] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann. Mediapipe hands: On-device real-time hand track-

ing. *arXiv preprint arXiv:2006.10214*, 2020. 5, 50, 63, 68, 70

[109] J. Zhang, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 32

[110] H. Zhou, F. Jiang, and R. Shen. Who are raising their hands? hand-raiser seeking based on object detection and pose estimation. In *Proceedings of the Asian Conference on Machine Learning (ACML)*, 2018. 70, 71

[111] X. Zhou, D. Wang, and P. Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019. x, 52, 53

[112] X. Zhou, V. Koltun, and P. Krähenbühl. Tracking objects as points. In *The European Conference on Computer Vision (ECCV)*, 2020. 48, 51, 59, 64, 65, 84

[113] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M.-H. Yang. Online multi-object tracking with dual matching attention networks. In *The European Conference on Computer Vision (ECCV)*, 2018. 50

[114] X. Zhu, J. Yang, and A. Waibel. Segmenting hands of arbitrary color. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 2000. 3, 4, 9, 31, 50, 58, 68, 70

[115] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei. Flow-guided feature aggregation for video object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 53

[116] C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single rgb images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 4, 20, 31, 68, 70

[117] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 4, 31, 68, 70