Hand Keypoint Detection in Single Images using Multiview Bootstrapping.

by

Mohammad Montasim -Al- Mamun Shuvo : 160201059
Abid Hasan : 160201060

Bachelor of Science in Computer Science & Engineering

Department of Computer Science and Engineering
Faculty of Computer and Electrical Engineering
Bangladesh Army University of Science and Technology

AUGUST 2020

Hand Keypoint Detection in Single Images using Multiview Bootstrapping

A project
submitted for partial fulfillment of the requirement for the degree of
Bachelor of Science in Computer Science & Engineering (Honor's Degree)
to the Department of Computer Science & Engineering of
Bangladesh Army University of Science & Technology (BAUST).

By
Mohammad Montasim -Al- Mamun Shuvo (160201059)
Abid Hasan (160201060)

Supervised by
Professor Dr. Md. Mamunur Rashid, Dean, Faculty of ECE & Professor
Abu Saleh Musa Miah, Lecturer
&
Mohammad Hasan, Lecturer
Department of Computer Science & Engineering (CSE)
Bangladesh Army University of Science & Technology (BAUST)

Department of Computer Science & Engineering

Bangladesh Army University of Science & Technology

Saidpur Cantonment, Nilphamari, Bangladesh.

AUGUST 2020

The project titled **"Hand Keypoint Detection in Single Images using Multiview Bootstrapping"** submitted by Mohammad Montasim -Al- Mamun Shuvo, Roll No. 160201059, Session 2020-2021, Abid Hasan, Roll No. 160201060, Session 2020-2021 has been accepted as satisfactory in fulfillment of the requirement for the degree of Bachelor of Science in Computer Science & Engineering (CSE) as B.Sc. Engineering to be awarded by the Bangladesh Army University of Science & Technology (BAUST) on 22-02-2021.

BOARD OF EXAMINERS

1. _____ Chairman

Name of the Supervisor: Professor Dr. Md. Mamunur Rashid
Designation: Dean, Faculty of ECE & Professor
Department of Computer Science & Engineering (CSE)
Bangladesh Army University of Science & Technology (BAUST)

2. _____ Supervisor

Name of the Supervisor: Abu Saleh Musa Miah
Designation: Lecturer
Department of Computer Science & Engineering (CSE)
Bangladesh Army University of Science & Technology (BAUST)

3. _____

Name of the Supervisor: Mohammad Hasan Supervisor
Designation: Lecturer
Department of Computer Science & Engineering (CSE)
Bangladesh Army University of Science & Technology (BAUST)

4. _____

Name of the Member : Dr. Mohammed Sowket Ali — Member

Designation : Assistant Professor

Department of Computer Science & Engineering (CSE)

Bangladesh Army University of Science & Technology (BAUST)

5. _____ Member

Name of the Member: Ananna Hoque Shathi

Designation : Lecturer

Department of Computer Science & Engineering (CSE)

Bangladesh Army University of Science & Technology (BAUST)

6. _____ Member

Name of the Member: Afsana Afrin

Designation : Lecturer

Department of Computer Science & Engineering (CSE)

Bangladesh Army University of Science & Technology (BAUST)

CANDIDATE'S DECLARATION

It is hereby declared that the contents of this project are original and any part of it has not been submitted elsewhere for the award of any degree or diploma.

…… ……

Mohammad Montasim -Al- Mamun Shuvo

160201059

Abid Hasan

160201060

DEDICATION

We dedicate this thesis project to our beloved parents, who have been our source of inspiration and gave us strength when we thought of giving up, who continually provided their moral, spiritual, emotional, and financial support. To our brothers, sisters, relatives, mentor, friends, and classmates who shared their words of advice and encouragement to finish this study. And lastly, we dedicate this book to Almighty Allah, thank you for the guidance, strength, power of the mind, protection, skill, and for giving us a healthy life. All of these, we offer to you.

ACKNOWLEDGMENT

First of all, we are grateful to almighty Allah who enabled us to complete this thesis successfully. Thereafter our sincerest thanks and gratitude to our honorable project supervisor, Abu Saleh Musa Miah, Lecturer, Department of Computer Science & Engineering (CSE) of Bangladesh Army University of Science & Technology for his valuable suggestions, positive advice, encouragement, sincere guidance, and above all for having improved our critical thinking throughout the thesis work. His knowledge, guidance, and patience during our ups and down made this research possible. His suggestions, teachings, and consistent support have been so helpful in the completion of this amazing research field.

Then we wish to express our deepest gratitude for all the people that helped us during this thesis work. We would like to thank Professor Dr. Md. Mamunur Rashid, Dean, Faculty of ECE & Professor, Department of Computer Science & Engineering (CSE) of Bangladesh Army University of Science & Technology for the chance he gave us in doing our thesis work.

We also convey our special thanks and gratitude to all the respected teachers of the Department of Computer Science & Engineering (CSE) of Bangladesh Army University of Science & Technology, BAUST, for teaching us so many things during these years. We would like to thank all the staff of the department for their valuable suggestions and assistance.

Finally, yet importantly, we would like to thank all the people who morally supported us during the thesis. Furthermore, we would like to show our gratitude to our beloved classmates and friends for sharing the best memories.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS OF TECHNICAL SYMBOLS AND TERMS

Terms	Meanings
CNN	Convolutional Neural Network
SGN	Density Network with a Single Gaussian Model.
MDN	Mixture Density Network
HMDN	Hierarchical Mixture Density Network
DoF	Degree of Freedom
PSO	Particle Swarm Optimization
HSO	Hierarchical Sampling Optimization
HSFON	Hierarchical Sampling Optimization based on random forests with brutal force optimization
HSFOP	Hierarchical Sampling Optimization based on random forests with PSO
HSCON	Hierarchical Sampling Optimization based on CNN with brutal-force optimization
HSCOP	Hierarchical Sampling Optimization based on CNN with PSO
HolisticCNN	Regressing the full pose by a single CNN
HierCNN	Regressing the full pose following the hierarchy by CNNs without sampling and optimization
HolisticForest	Regressing the full pose by a single random forest
HierForest	Regressing the full pose following the hierarchy by random forests without- out sampling and optimization

ABSTRACT

Hand key point detection plays a fundamental role in natural human-computer interactions. The problem is challenging due to complicated variations caused by complex articulations, multiple viewpoints, self-similar parts, severe self-occlusions, different shapes, and sizes. To handle these challenges, the thesis makes the following contributions.

We present an approach that uses a multi-camera system to train fine-grained detectors for key points that are prone to occlusion, such as the joints of a hand. We call this procedure multiview bootstrapping. First, an initial Keypoint detector is used to produce noisy labels in multiple views of the hand. The noisy detections are then triangulated in 3D using Multiview geometry or marked as outliers. Finally, the reprojected triangulations are used as new labeled training data to improve the detector.

We repeat this process, generating more labeled data in each iteration. We derive a result analytically relating the minimum number of views to achieve target true and false-positive rates for a given detector.

The method is used to train a hand Keypoint detector for single images. The resulting Keypoint detector runs in real-time on RGB images and has accuracy comparable to methods that use depth sensors. The single view detector, triangulated over multiple views, enables 3D marker less hand motion capture with complex object interactions.

CHAPTER 1

Introduction

1.1 Overview:

Hand pose estimation is an important computer vision topic due to its wide range of applications in human-computer interaction, augmented/virtual reality, and gaming. Such applications often require hand segmentation, articulated hand pose estimation, and tracking.

Recent methods in body pose estimation [6, 11] can be used to detect and segment hands using human body hand joint features. Articulated hand pose estimation from monocular RGB images is still a largely unsolved problem in several aspects. This is because human hand configurations are very diverse, with the pose of a human hand having over 20 Degrees of Freedom (DoF). Hands are smaller than the body, and thus they occupy a small part of the image when the full body is visible. Besides, hand keypoints are oftentimes occluded by other parts of the same hand, the other hand, or the rest of the body.

Deep learning-based methods currently achieve the state of the art performance for human body pose estimation. Estimating body pose is an articulated pose estimation problem similar to hand pose estimation. However, body pose estimation is easier, due to the body orientation being upright most time, and also due to occlusions being a less frequent and less severe problem for full-body images compared to hand images.

We investigate deep learning-based methods for hand pose estimation that perform holistic articulated pose estimation. Pixel-wise pose estimation approaches could be slow for real-time applications and do not take advantage of important holistic hand features due to per pixel constraints.

In this work, we focus on RGB-based articulated hand pose estimation. We prefer this modality due to the availability and ease of deployment of regular color cameras compared to depth cameras. Our contribution aims towards partial hand pose estimation problem on single RGB image frames. The hand key points that we estimate are the wrist, fingertips for the thumb, index finger, middle finger, ring finger, little finger.

We provide a novel RGB benchmark dataset for hand keypoints estimation and perform evaluations to provide quantitative evaluation for current state-of-the-art methods for this task. This dataset includes hand gestures and keypoint annotations for gestures about rhythmic hand movements. Our motivation is that tasks involving such movements can be used for cognitive assessments, in conjunction with tasks involving whole-body motion [3]. There is a need for computational methods that would help with the automatic computation of various physical performance metrics, to improve the accuracy and efficiency of human-made assessments. Articulated hand pose recognition is an important step in recognizing and assessing physical exercises that contain hand gestures.

We discuss the selected hand gestures in Section 4, where we describe the physical exercise tasks and the importance of articulated hand pose estimation in assessing performance in those tasks.

Recognition of rhythmic movements for rapid sequential hand gestures poses the additional challenges that the motion is fast and complicated. Moreover, the hand can be in any orientation and the dexterity of the hand makes it hard to estimate and track finger positions.

The paper is further organized as follows: In Section 1, we discuss Introduction; in Section 2, we discuss Literature Review; Section 3 describes Methodology; in Section 4 we describe Implementation; in Section 5, we discuss our experimental Result & Analysis.

1.2 Necessity of Hand Recognition

I think Hand Recognition (HR from now on) is an interesting problem since it has a good trade-off between the complexity of the problem and the satisfaction it gives you to see your algorithm detecting your hand live on the webcam. The detection uses Convolutional Neural Networks, which are now the standard when it comes to image recognition, and if you understand what do you want your algorithm to do, it's easy to implement using frameworks such as TensorFlow.

According to some market research studies I've found; the hand gesture recognition market is expected to reach USD 30 billion by 2025. If you think about how much of a change did the introduction of touch screens to our software UI interfaces and how we interact with our devices, touchless screens are for sure the next big step. While it's not so easy to imagine how this will impact mobile phones and laptops, there are no doubts that hand recognition will make it easier to use the car's computer or to turn up and down the television's volume without having to figure out where the remote control is.

1.3 Background and Present State

Early work in hand pose estimation originally considered RGB data, with Rehg and Kanade [15] exploring vision-based Human-Computer Interaction (HCI) applications. Most methods were brittle, based on fitting complex 3D models with strong priors, including e.g., physics or dynamics [10], multiple hypotheses [21], or analysis-by synthesis [5]. Cues such as silhouettes, edges, skin color, and shading were demonstrated in controlled environments with restricted poses and simple motions. The method of Wang and Popovic [´ 28] lifted some of these restrictions but required a specialized colored glove. Multiview RGB methods are often similarly based on fitting sophisticated mesh models (e.g., [3, 20]) and show excellent accuracy, but again under highly controlled conditions.
With the introduction of commodity depth sensors, single-view depth-based hand pose estimation became the major focus of research, resulting in a large number of depth-based methods. These are broadly classifiable into generative methods [14], discriminative methods [24, 25, 9, 30, 22, 27], or hybrid methods [20, 16, 18, 26, 31]. Recently, the hybrid method of Sharp et al. [16] demonstrated practical performance across a large range, but there are still difficult cases such as hand-hand interactions and hand object interactions. Discriminative and hybrid approaches to

depth-based hand pose estimation rely heavily on synthetic data [23]. Oberwerger et al. [13] use feedback loops to generate synthetic training data for hand pose estimation, motivated by the same principles as our work, but focus on generating depth images. The semi-automatic data annotation scheme presented in [12] is also similar in motivation, however, our approach uses multiview geometry and keypoint detection to provide automated supervision.

Discriminative methods, especially approaches that rely on deep architectures, require large annotated training sets. These datasets are relatively easy to synthesize for depth maps, but present significant challenges for RGB as rendering is far more complicated, requiring photorealistic appearance and lighting. Multiview bootstrapping is an approach that allows the generation of large annotated datasets using a weak initial detector. This, in turn, enables the creation of the first real-time hand keypoint detector for RGB images "in-the-wild".

1.4 Scopes and limitations

When the pose parameters are joint locations, the pose estimation task can be treated as detecting the key points from input images and therefore shares some similarities with other vision problems such as facial landmarking, 6D Object Detection, human body pose estimation. Among these problems, hand pose estimation has the most common challenges with human body estimation which has enjoyed a great

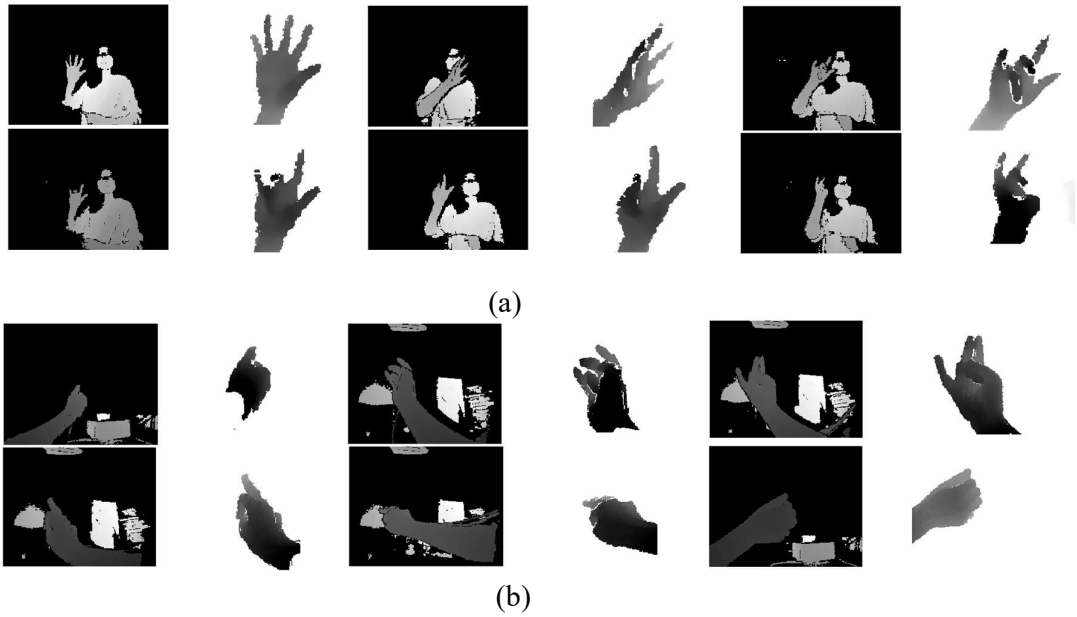

Figure 1.1: Depth images captured by Intel RealSense SR300 [Intel] and cropped hand areas. (a) images captured in third-person viewpoints; (b) images captured in first-person (ego-centric) viewpoints.

progress [Shotton et al., 2011; Jiu et al., 2014; Girshick et al., 2011; Ouyang et al., 2014; Papandreou et al., 2017; Cao et al., 2016] in the last decade. Both of them model the articulated objects with many degrees of freedom and self-occlusions while hand pose estimation has some unique difficulties due to complicated variations caused by high DoF articulations, multiple viewpoints, self-similar parts, severe self-occlusions, different shapes, and sizes.

1.5 Objectives

We present an approach that uses a multi-camera system to train fine-grained detectors for key points that are prone to occlusion, such as the joints of a hand. We call this procedure multiview bootstrapping. First, an initial keypoint detector is used to produce noisy labels in multiple views of the hand. The noisy detections are then triangulated in 3D using multiview geometry or marked as outliers. Finally, the reprojected triangulations are used as new labeled training data to improve the detector.

We repeat this process, generating more labeled data in each iteration. We derive a result analytically relating the minimum number of views to achieve target true and false-positive rates for a given detector.

The method is used to train a hand keypoint detector for single images. The resulting keypoint detector runs in realtime on RGB images and has accuracy comparable to methods that use depth sensors. The single view detector, triangulated over multiple views, enables 3D markerless hand motion capture with complex object interactions.

Chapter 2

Literature Review

2.1 Overview

Most vision-based articulated hand pose estimation methods can be categorized based on the modality of the data that they use, their application towards first-person view (ego-centric) or third-person view, and whether the methods are discriminative or generative. Below we briefly provide a review of articulated hand pose estimation based on depth, colour, or a combination of both depth and colour modalities.

2.2 Convolutional Pose Machines

The CPM is a convolutional neural network for human pose estimation on single 2D human pose estimation datasets, such as MPII, LSP, and Frames Labelled in Cinema (FLIC). The model uses CNN [19–21] for human pose estimation. Its main contribution lies in the use of sequential convolution architecture to express spatial information and texture information [10]. The sequential convolution architecture can be divided into several stages in the network. Each stage has a part of the supervised training [17,22], which avoids the problem of gradient disappearance in the deep network [23–26]. In the first stage, the original image is used as input. In the later stages, the feature map of the first stage is used as input. The main purpose is to fuse spatial information, texture information, and central constraints. In addition, the use of multiple scales to process the input feature map and response map for the same convolution architecture not only ensures accuracy, but also considers the distance relationship between the key points of each human skeleton. The overall structure of the CPMs is shown in Figure 2. In Figure 2, the “C”, “MC1, MC2,” means different convolution layers, and the “P” means different pooling layers. The “Centre map” is the centre point of the human body picture, and it is used to aggregate the response maps to the image centres. The “Loss” is the minimum output cost function, and it is the same as the “Loss” of the subsequent figures.

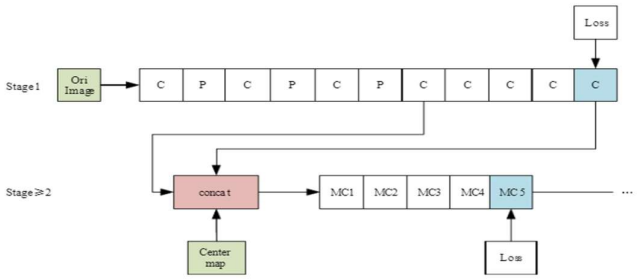

Figure 2.1: The overall structure of the convolutional pose machines.

The first stage of the CPMs is a basic convolutional neural network (white convs) that directly generates the response map of each human skeleton's key points from images. The whole model has the response maps of 14 human skeleton key points and a background response map, with a total of15 layers of response maps. The network structure with the stage ≥ 2 is completely consistent. A feature image with a depth of 128, which is from stage 1, is taken as the input and three types of data (texture features, spatial features, and centre constraints (the centre point of the human body picture is used to aggregate the response maps to the image centres)) are fused by the concept layer.

2.3 The Steps Towards Hand Recognition

Before the invention of Deep Learning, old methods had to perform a lot of sophisticated calculations and heuristics to infer which seen shapes were hands. Strong priors and physics restrictions had to be imposed, and the results weren't that good when they were evaluated in the wild, outside the highly controlled conditions of the laboratory. Furthermore, the detection was usually made using depth sensors, which provided more useful information than normal cameras. The arrival of Neural Networks and Deep Learning changed it all on Computer Vision. Not only the algorithms used now are somehow more friendly, but the trend to use depth sensors to detect body parts is over. While it's true that RGB cameras are just another type of sensor, they are cheaper than depth sensors and are present almost anywhere.

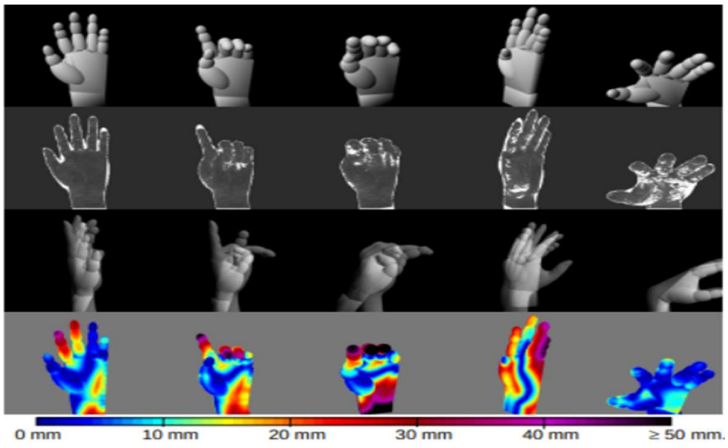

Figure 2.2: A 2011 paper estimating surfaces on the hand

Chapter 3

Methodology

3.1 Overview

Overview: This method is basically a training or learning method which learns by Multiview system. In our daily life, we see an object with multiple viewing sight. Every object has a different shape from different angles. Designing computer vision algorithms for camera networks requires an understanding of how images of the same scene from different viewpoints are related. An approach that uses a multi-camera system to train fine-grained detectors for keypoints that are prone to occlusion, such as the joints of a hand. We call this procedure Multiview Bootstrapping: first, an initial keypoint detector is used to produce noisy labels in multiple views of the hand. The noisy detections are then triangulated in 3D using Multiview geometry or marked as outliers. Finally, the reprojected triangulations are used as new labeled training data to improve the detector. We repeat this process, generating more labeled data in each iteration.

3.2 Algorithm Idea

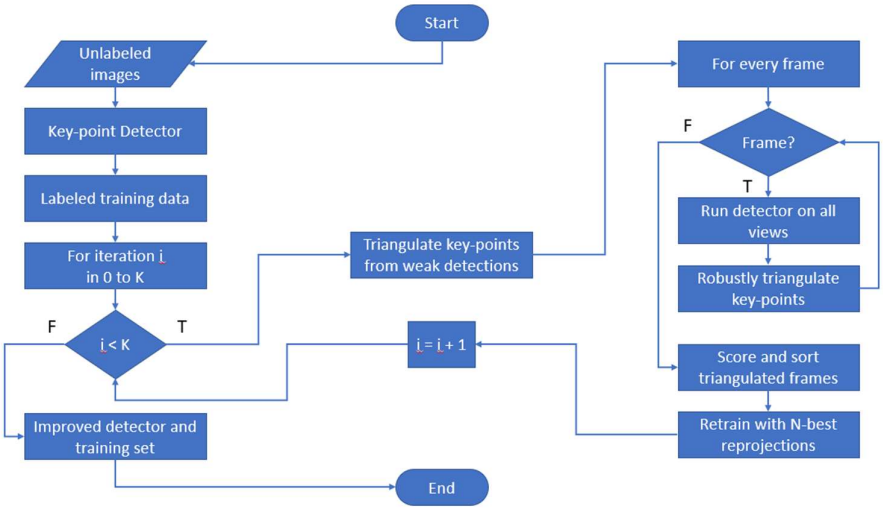

Figure 3.1: Flowchart of methodology design

The algorithm model used by the author in this article is CMU Perceptual Computing Lab's open-source library Open Pose, which integrates human, face, and hand Keypoint detection. The Hand Keypoint detector algorithm comes from the CVPR2017 paper "Hand Keypoint Detection in Single Images using Multiview Bootstrapping. Due to different perspectives and flexible fine movements of human hands in 3D space, it is difficult to obtain accurately labelled data sets. In this paper, the author proposes an iterative improvement algorithm for hand key point detection called Multiview Bootstrapping, which realizes a detection algorithm with higher accuracy.

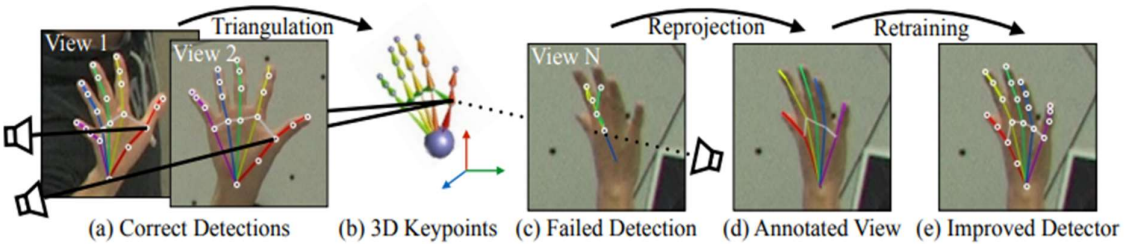

Figure 3.2: Multiview Bootstrapping. (a) A multiview system provides views of the hand where keypoint detection is easy, which are used to triangulate (b) the 3D position of the keypoints. Difficult views with (c) failed detections can be (d) annotated using the reprojected 3D keypoints, and used to retrain (e) an improved detector that now works on difficult views.

As shown in the figure above, the author proposes to first train the Convolutional Pose Machines neural network with a small number of labelled data sets containing key points of human hands, use 31 high-definition cameras with different viewing angles to shoot human hands, and use the above detection model to initially detect key points and combine these key points Construct a triangle (triangulation) according to the pose of the camera to obtain the 3D position of the key point, and then reproject the calculated 3D point position to each 2D image with a different perspective, and then use these 2D images and key point annotations to train the detection model After several iterations of the network, a more accurate hand key point detection model can be obtained. The model proposed in the original paper can generate 22 key points, of which 21 points are human hands, and the 22nd point represents the background. The figure below shows the 21 key points of the human hand.

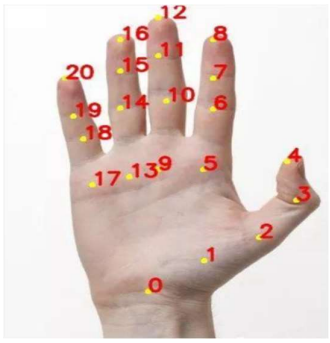

Figure 3.3: 22 key points

3.3 Dataset

We introduce the Hand Keypoint Dataset (HKD), which comprises annotated RGB images captured from participants while they perform rhythmic finger movements. Our dataset comprises 782 color image frames captured from four different participants. The rapid

sequential rhythmic finger movements are performed as described in Section 3. This is a novel benchmark dataset for hand Keypoint detection and/or tracking from RGB images. We provide original frames with annotations as well as annotated cropped frames that

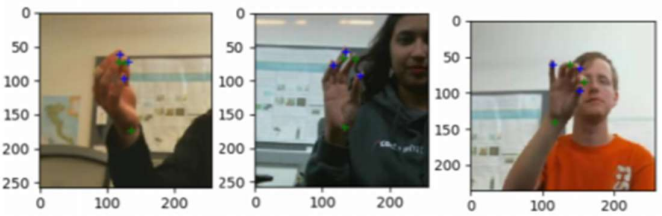

Figure 3.4: Example annotations of cropped images from HKD dataset

are loosely cropped around the centroid location of the hand in the frame. We provide annotations for 6 hand keypoints namely - W, TT, IT, TM, T R, T L, for wrist and the tips of the thumb, index, middle, ring, and little finger respectively. We also provide the hand centroid location in the original RGB frames. For data collection, participants were told to perform the rapid sequential finger gestures specified in Section 3 and each participant performed these movements thrice with different hand orientations from the camera. The dataset consists of hand movements from four participants (2M, 2F), and the annotations are done manually by two annotators who used the annotation toolkit that we developed to perform standardized annotations. Additional details of our dataset,

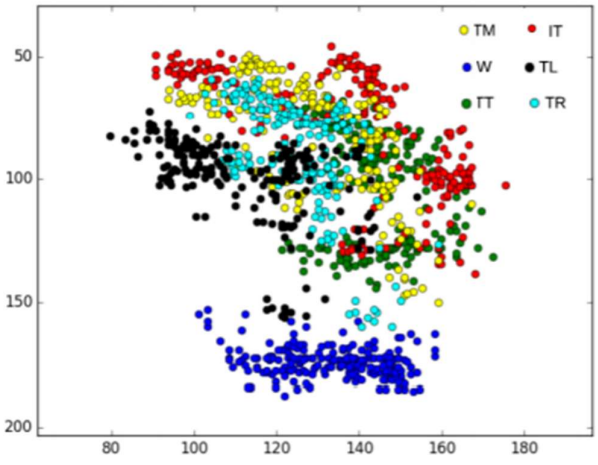

Figure 3.5: Variance of hand key points of Subject1 in HKD

3.4 Proposed Methodology

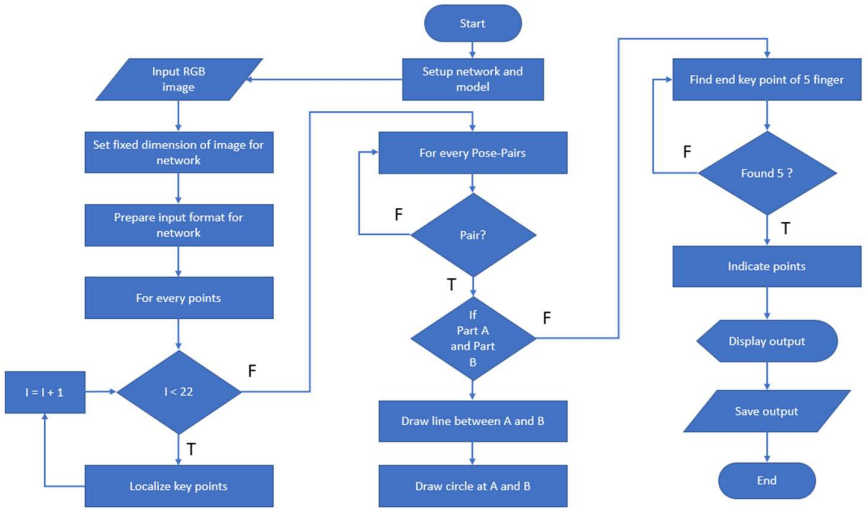

Figure 3.6: Proposed Flowchart

We setup our network module and dataset with the help of OpenCV, and prototext and caffemodel file. Then we take an RGB image which is containing hand. Detecting keypoints is more like finding interesting points of hand as like as joint of human hand. We setup 22 points for localizing hand keypoints and setup pairs of keypoints to join the specific point as like human skeleton. When this method localized all the keypoints then we draw the skeleton by drawing line between the pairs and draw the keypoints as circle. Finally, we display the output with detected keypoints and skeleton.

3.5 Unlabeled image

Unlabeled data consists of data which is either taken from nature or created by human to explore the scientific patterns behind it. Some examples of unlabeled data might include photos, audio recordings, videos, news articles, tweets, x-rays, etc. The main concept is there is no explanation, label, tag, class or name for the features in data.

We take unlabeled images for every views and frames as an input.

$$\boldsymbol{I}_v^f \text{ for } v \in \text{views}, f \in \text{frames}$$

3.6 Key-point detector

Keypoint detection involves simultaneously detecting hand and localizing their keypoints. Keypoints are the same thing as interest points. They are spatial locations, or points in the image that define what is interesting or what stand out in the image.

We have 22 keypoints to localize a hand in a picture. An initial detector first detects a keypoint and the detector is used to produce noisy labels in in multiple views. A keypoint detector d (.) maps a cropped input image patch $I \in R^{w \times h \times 3}$ to P keypoint locations $x_p \in R^2$, each with an associated detection confidence c_p:

$$d_0(I) = \{(x_p, c_p) \text{ for } p \in \text{points}\}$$

Each point *p* corresponds to a different landmark (e.g., the tip of the thumb, the tip of the index finger), and we assume that only a single instance of the object is visible in *I*.

3.7 Labeled Training data

Labeled data is a designation for pieces of data that have been tagged with one or more labels identifying certain properties or characteristics, or classifications or contained objects. Labels make that data specifically useful in certain types of machine learning known as supervised machine learning setups. In supervised machine learning, labeled data acts as the orientation for data training and testing exercises. The supervised machine learning program may start out with a set of entirely labeled data, or it may use initial labeled data to work with additional unlabeled data.

When the detector detects a keypoint and which was noisy then it transformed into triangulated in 3D using Multiview geometry or marked as outliers. The detector is trained on images with corresponding keypoint annotations, $(I^f, \{y_p^f\})$, where *f* denotes a particular image frame, and the set $\{y_p^f \in R^2\}$ includes all labeled keypoints for the image I^f. An initial training set T_0 having N_0 training pairs,

$$T_0 := \{(I^f, \{y_p^f\})\} \text{ for } f \in [1 \ldots N_0]$$

is used to train an initial detector d_0 with, e.g., stochastic gradient descent,

$$d_0 \leftarrow train(T_0).$$

Given the initial keypoint detector d_0 and a dataset of unlabeled Multiview images, our objective is to use the detector to generate a set of labeled images, T_1, which can be used to train an improved detector, d_1, using all available data:

$$d_0 \leftarrow train\ (T_0 \cup T_1).$$

To improve upon the detector d_0, we need an external source of supervision to ensure $_1$ contains information not

3.8 Triangulating Keypoints from Weak Detections

Given *V* views of an object in a particular frame *f*, we run the current detector d_i (trained on set Ti) on each image I_v^f, yielding a set *D* of *2D* location candidates:

$$D \leftarrow \{d_i\ (I_v^f) \text{ for } v \in [1 \ldots V]\}$$

For each Keypoint *p*, we have *V* detections (x_p^v, c_p^v), where x_p^v is the detected location of point *p* in view *v* and $c_p^v \in [0, 1]$ is a confidence measure (we omit the frame index for clarity). To robustly triangulate each point *p* into a 3D location, we use RANSAC [20] on points in *D* with confidence above a detection threshold λ. Additionally, we use a σ = 4-pixel reprojection error to accept RANSAC inliers. With this set of inlier views for point *p*, we minimize [21] the reprojection error to obtain the final triangulated position,

$$X_p^f = \arg.\min_x \sum_{v \in I_p^f} . \left\| P_v(\mathrm{X}) - x_p^v \right\|.{}_2^2$$

where I_p^f is the inlier set, with $X_p^f \in R^3$ the 3D triangulated keypoint p in frame f, and $P_v(X) \in R^2$ denotes projection of 3D point X into view v. Given calibrated cameras, this 3D point can be reprojected into any view (e.g., those in which the detector failed) and serve as a new training label.

To improve robustness specifically for hands, we reconstruct entire fingers simultaneously. We triangulate all landmarks of each finger (4 points) at a time, and use the average reprojection error of all 4 points to determine RANSAC inliers. This procedure is more robust because errors in finger detections are correlated: e.g., if the knuckle is incorrectly localized, then dependent joints in the kinematic chain— the inter-phalangeal joints and fingertip—are unlikely to be correct. This reduces the number of triangulated keypoints (because the entire finger needs to be correct in the same view) but it further reduces the number of false positives, which is more important so that we do not train with incorrect labels.

3.9 Scoring and Sorting Triangulated Frames

It is crucial that we do not include erroneously labeled frames as training data, especially if we iterate the procedure, as subsequent iterations will fail in a geometrically consistent way across views—a failure which cannot be detected using multi-view constraints. We therefore conservatively pick a small number of reliable triangulations. Our input is video and consecutive frames are therefore highly correlated. Instead of uniform temporal subsampling, we pick the "best" frame for every window of W frame (e.g., W=15 or W=30), defining the "best" as that frame with maximum sum of detection confidences for the inliers, i.e.,

$$Scorer\ (\{X_p^f\}) = \sum_{p\epsilon[1\ldots p]} \cdot \sum_{v\epsilon I_p^f} \cdot c_p^v$$

We sort all the remaining frames in descending order according to their score, to obtain an ordered sequence of frames, [s1, s2…sF'], where F' is the number of subsampled frames and s_i is the ordered frame index.

We manually verify that there are no obvious errors in the frames to be used as training data, and train a new detector. While visual inspection of the training set may seem onerous, in our experience this is the least time-consuming part of the training process. It typically takes us one or two minutes to inspect the top 100 frames. Crowdsourcing such a verification step for continuous label generation is an interesting future direction, as verification is easier than annotation. In practice, we find that this manual effort can be almost eliminated by automatically removing questionable triangulations using a number of heuristics: (1) average number of inliers, (2) average detection confidence, (3) difference of per-point velocity with median velocity between two video frames, (4) anthropomorphic limits on joint lengths, and (5) complete occlusion, as determined by camera ray intersection with body joints. Additionally, we require at least 3 inliers for any point to be valid.

3.9 Retraining with N-best Reprojections

We use the N-best frames according to this order to define a new set of training image-keypoint pairs for the next iteration $i+1$ detector,

$$T_{i+1} = \{(I_v^{s_n}, \{P_v\ (X_p^{s_n}) : v \epsilon [1\ldots V],\ p \epsilon [1\ldots P]\})\ for\ n \epsilon [1\ldots N]\}$$

where $P_v\ (X_p^{s_n})$ denotes projection of point p for frame index s_n into view v, and we aim for about N=100 frames for every 3 minutes of video. Note that 100 frames yield roughly $100\ \frac{V}{2} \approxeq 1500$ training samples, one

for each unoccluded viewpoint. Finally, we train a new detector using the expanded training set as d_{i+1} train $(T_0 \cup T_{i+1})$.

3.10 Algorithm Multiview Bootstrapping

Inputs:

• Unlabeled images: $\boldsymbol{I}_v^f$ *for* $v \in$ *views,* $f \in$ *frames*

• Keypoint detector: $d_0(I) = \{(x_p, c_p) \text{ for } p \in \text{points}\}$

• Labeled training data: T_0

for iteration i in *0* to *K*:

1. Triangulate keypoints from weak detections

for every frame *f* :

(a). Run detector $d_i\ (I_v^f)$ on all views *v*

(b). Robustly triangulate keypoints for X_p^f

2. Score and sort triangulated frames

3. Retrain with N-best reprojections

d_{i+1} train $(T_0 \cup T_{i+1})$.

Outputs: Improved detector $d_K(.)$ and training set T_K

Chapter 4

Implementation

4.1 Overview

We execute the method discussed in the previous chapter to estimate the hand pose and identify the five fingers separately. We detect all the 22 keypoints by implementing the Multiview Bootstrapping method and then we estimate the hand pose with the help of detected keypoints and finally we classify the five fingers separately. That was the final task of our implementation.

Implementation process:

1. Image preprocessing
2. Localize keypoints
3. Estimate Hand pose
4. Finger classification
5. Visualization

4.1. Image Preprocessing:

Preliminary processing of an image in order to prepare it for the primary processing or for further analysis. Here, we take an image and reduce its dimension by resizing. First, we find the original height and weight of given image and calculate the aspect ratio. Then, we set a default height 368 and weight is depending on aspect ratio and height. We calculate the weight as follows (Aspect ratio * Height).

We need four pieces of information of the image to process with the Multiview Bootstrapping method. The four parameters are 1. batch size, 2. channel, 3. height, 4. width. With the help of OpenCV and the function of OpenCV named “blobFromImage”, we combine all the information we need to provide for the network.

4.2. Localize Keypoints:

The output has 22 matrices with each matrix being the Probability Map of a keypoint. For finding the exact key-points, first, we scale the probability map to the size of the original image. Then find the location of the keypoint by finding the maxima of the probability map. This is done using the minmaxLoc function in OpenCV. We draw the detected points along with the numbering on the image.

We can see that 0 points in indicating wrist and 4, 8,12,16,20 is indicating the pic point of every finger.

4.3 Estimate Hand pose:

We define pairs of key points as like joint on human hand. We define 20 pose pairs to connect the detected keypoints. The pose pairs are like,

[[0, 1], [1, 2], [2, 3], [3, 4],

[0, 5], [5, 6], [6, 7], [7, 8],

[0, 9], [9, 10], [10, 11], [11, 12],

[0, 13], [13, 14], [14, 15], [15, 16],

[0, 17], [17, 18], [18, 19], [19, 20]]

We will use the detected points to get the skeleton formed by the key points and draw them on the image.

4.4. Finger Classification:

We noticed that all the fingers start from 0 point and end with 4, 8, 12, 16, 20 respectively.

When the detector d_i {if i ϵ [0…4]} It is indicating as number 1 finger,

When, d_i {if i ϵ [0, 5…8]} It is indicating as number 2 finger,

When, d_i {if i ϵ [0, 9…12]} It is indicating as number 3 finger,

When d_i {if i ϵ [0, 13…16]} It is indicating as number 4 finger,

When, d_i {if i ϵ [0, 17…20]} It is indicating as number 5 finger.

So, we need to identify the pixel value of the needed points and indicate with the indexing number.

4.5. Visualization:

Visualizing the image is a function of OpenCV module. With the help of OpenCV we show our result. Mainly we draw separate color for different fingers and indicate the fingers with straight line from the top of the image with index.

Chapter 5

Results and Analysis

5.1 Overview

None of the available hand pose estimation datasets we reviewed suited our target use case: general, in-the-wild images containing everyday hand gestures and activities. We therefore manually annotated two publicly available image sets: (1) The MPII Human Pose dataset [2], which contains images extracted from YouTube videos explicitly collected to reflect every-day human activities, and (2) Images from the New Zealand Sign Language (NZSL) Exercises of the Victoria University of Wellington [11], in which several people use NZSL to tell stories. We chose the latter because it contains a variety of hand poses as might be found in conversation (less common in MPII).

Figure 5.1: Result

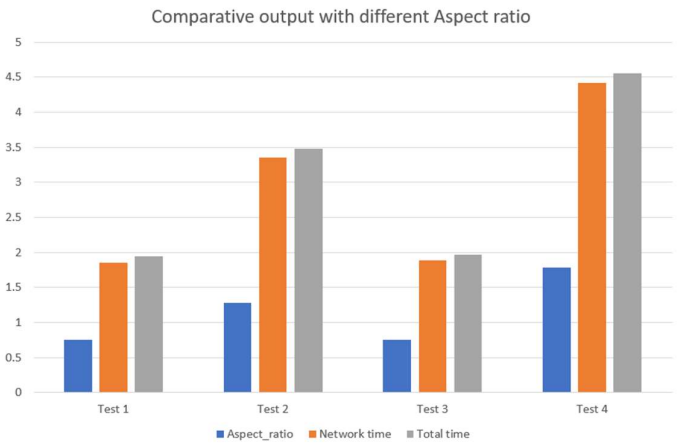

Figure 5.2: Comparative output with different Aspect ratio

5.2 Robustness to View Angle

We quantify improvement in the detector's robustness to different viewing angles by measuring the percentage of outliers during 3D reconstruction. As ground truth, we visually inspect our best 3D reconstruction result and select only correctly reconstructed frames. We define view angle as azimuth φ and elevation θ w.r.t a fixed hand at the origin. Intuitively, angles with $\varphi = \{-180, 0, 180\}$ (viewing the palm or backhand face-on) are easier because there is less self-occlusion. At $\varphi = \{-90, 90\}$, we are viewing the hand from the side, from thumb to little-finger or vice-versa, resulting in more occlusion. Similarly, at $\theta = \{90, -90\}$ the viewing angle is from fingertips to wrist, and vice-versa; these are the most difficult viewpoints. We compare the first iterations of the "Mix" detector, which quickly becomes robust to view diversity. We plot this as a heatmap, where we bin hand detections using each example's azimuth and elevation. The percentage of outliers is computed using all the examples that fall in each bin.

5.3 Comparison to Depth-based Methods

We quantify the performance of our method on a publicly available dataset from Tzionas et al. [26]. Although there exist several datasets often used to evaluate depth-based methods, many of them do not have corresponding RGB images, or their annotations are only valid for depth images. Datasets with RGB images and manual annotations that can be accurately localized are rare; the dataset from [26] is the best match to quantify our method. We run the 2D Keypoint detector "Mix 3" on the RGB images of the dataset. The sequences include single-hand motion, hand-hand interaction, and hand-object interaction. To allow direct comparison with [26], we use average pixel errors in the location of the keypoints provided, as shown in Table 1. Note that the method of [26] is based on a complex 3D hand template and uses depth data and tracking, taking several seconds per frame. Our result shows comparable performance for single-hand and hand-object scenarios, using only per-frame detection on RGB and can be run in Realtime using GPUs. Performance degrades for hand interaction: When one of the hands is very occluded, our detector tends to fire on the occluding hand. Detecting joints on both hands simultaneously would be advantageous in these cases, rather than treating each hand independently as our current approach does.

5.5 Summary

We have evaluated Multiview bootstrapping by applying the Algorithm on three initial detectors. All three detectors follow the architecture described in Sect. 4, but are trained on 3 different sets of initial training data T0: (1) “Render”: an initial set of synthetically rendered3 images of hands, totaling around 11000 examples, (2) “Manual”: the manual annotations in the MPII and NZSL training sets described above, and (3) “Mix”: the combination of rendered data and manual annotations. For Multiview bootstrapping, we use images from the Panoptic Studio dataset [8]. In particular, we use 31 HD camera views, and 4 sequences that contain hand motions, and we use the provided 3D body pose [8] to estimate occlusions and bounding boxes for hand detection. When performing bootstrapping iterations, we discard frames with an average number of inliers < 5 or an average reprojection error > 5, with detection confidence threshold λ=0.2. For no iteration did we have to manually discard more than 15 incorrectly labeled frames. One thing to note is that the detector expects a bounding box around the hand to predict the key points. Thus, for better results, the hand should be close to the camera or it should be cropped using a hand detector and then fed to the network. Also, the code given can detect only one hand at a time. You can easily port it to detecting multiple hands with the help of the probability maps and applying some heuristics.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This paper presents two innovations: (1) the first real-time hand Keypoint detector showing practical applicability to in-the-wild RGB videos; (2) the first marker less 3D hand motion capture system capable of reconstructing challenging hand-object interactions and musical performances without manual intervention. We find that rich training sets can be built using Multiview bootstrapping, improving both the quality and quantity of the annotations. Our method can be used to generate annotations for any Keypoint detector that is prone to occlusions (e.g., body and face). Building a large annotated dataset is often the major bottleneck for many machines learning and computer vision problems, and our approach is one way to refine weakly supervised learning by using Multiview geometry as an external source of supervision.

6.2 Future Recommendation

As future work, making the method robust enough to identify right hand or left hand and optimize algorithm and implement in a real world problem solving would allow the creation of even richer datasets that more closely reflect real-world capture conditions.

References

[1] S. Agarwal et al. Ceres solver. http://ceres-solver.org. 3

[2] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In CVPR, June 2014. 6

[3] L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys. Motion capture of hands in action using discriminative salient points. In ECCV, 2012. 2

[4] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Real-time multiperson 2d pose estimation using part affinity fields. In CVPR, 2017. 5

[5] M. de La Gorce, D. J. Fleet, and N. Paragios. Model-based 3D hand pose estimation from video. TPAMI, 2011. 2

[6] M. A. Fischler and R. C. Bolles. Random sample consensus. Communications of the ACM, 1981. 3

[7] T. M. Greiner. Hand anthropometry of US army personnel. Technical report, DTIC Document, 1991. 4

[8] H. Joo, H. Liu, L. Tan, L. Gui, B. Nabbe, I. Matthews, T. Kanade, S. Nobuhara, and Y. Sheikh. Panoptic studio: A massively multiview system for social motion capture. In ICCV, 2015. 6, 7

[9] C. Keskin, F. Kırac¸, Y. E. Kara, and L. Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In ECCV, 2012. 2

[10] S. Lu, D. Metaxas, D. Samaras, and J. Oliensis. Using multiple cues for hand tracking and model refinement. In CVPR, 2003. 2

[11] R. McKee, D. McKee, D. Alexander, and E. Paillat. NZ sign language exercises. Deaf Studies Department of Victoria University of Wellington, http://www.victoria.ac. nz/llc/llc_resources/nzsl/. 6

[12] M. Oberweger, G. Riegler, P. Wohlhart, and V. Lepetit. Efficiently Creating 3D Training Data for Fine Hand Pose Estimation. In CVPR, 2016. 2

[13] M. Oberweger, P. Wohlhart, and V. Lepetit. Training a Feedback Loop for Hand Pose Estimation. In ICCV, 2015. 2

[14] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Tracking the articulated motion of two strongly interacting hands. In CVPR, 2012. 2

[15] J. M. Rehg and T. Kanade. Digiteyes: Vision-based hand tracking for human-computer interaction. In Motion of NonRigid and Articulated Objects, 1994. 2

[16] T. Sharp et al. Accurate, robust, and flexible real-time hand tracking. In CHI, 2015. 2

[17] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015. 4

[18] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt. Fast and robust hand tracking using detection-guided optimization. In CVPR, 2015. 2

[19] 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 ECCV, 2016. 7

[20] Martin A. Fischler and Robert C. BollesSRI International, Random SampleConsensus: AParadigm for ModelFitting withApphcatlons to ImageAnalysis andAutomatedCartography

[21] Shawn Recker1, Mauricio Hess-Flores2, Kenneth I. Joy3University of California, Davis, Statistical Angular Error-Based Triangulation for Efficient and Accurate Multi-View Scene Reconstruction