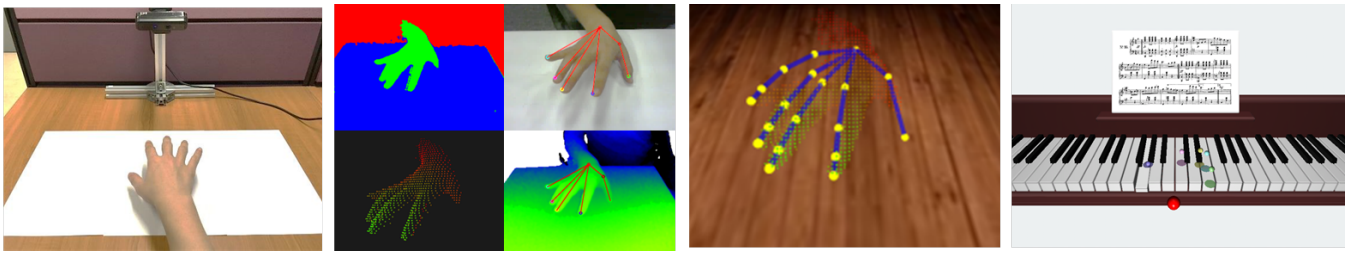
Barehanded Music: Real-time Hand Interaction for Virtual Piano

Hui Liang[†] * Jin Wang[†‡*] Qian Sun[†] Yong-Jin Liu[§] Junsong Yuan[†] Jun Luo[†] Ying He[†]
[†]Nanyang Technological University [‡]SAP Innovation Center, Singapore [§]Tsinghua University



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

Figure 1: *Our method is able to track fingertip and detect finger tapping in real-time using an RGB-D camera, allowing us to develop a virtual piano application. (a) Experiment setup with a DepthSense® 325 sensor. (b) Hand segmentation and fingertip tracking. (c) Hand pose estimation by inverse kinematics. (d) Virtual piano application with tapping detection.*

Abstract

This paper presents an efficient data-driven approach to track fingertip and detect finger tapping for virtual piano using an RGB-D camera. We collect 7200 depth images covering the most common finger articulation for playing piano, and train a random regression forest using depth context features of randomly sampled pixels in training images. In the online tracking stage, we firstly segment the hand from the plane in contact by fusing the information from both color and depth images. Then we use the trained random forest to estimate the 3D position of fingertips and wrist in each frame, and predict finger tapping based on the estimated fingertip motion. Finally, we build a kinematic chain and recover the articulation parameters for each finger. In contrast to the existing hand tracking algorithms that often require hands are in the air and cannot interact with physical objects, our method is designed for hand interaction with planar objects, which is desired for the virtual piano application. Using our prototype system, users can put their hands on a desk, move them sideways and then tap fingers on the desk, like playing a real piano. Preliminary results show that our method can recognize most of the beginner's piano-playing gestures in realtime for soothing rhythms.

Keywords: Fingertip Tracking, Finger Tapping Detection, Virtual Piano, RGB-D images, Human-Computer Interaction

Concepts: •**Computing methodologies** → *Graphics systems and interfaces;*

*H. Liang and J. Wang contribute equally to this paper. E-mail: yhe@ntu.edu.sg (Y. He)

I3D '16, March 01 2016, Redmond, Washington, USA.
ISBN: 978-1-4503-4043-4/16/03
DOI: http://dx.doi.org/10.1145/2856400.2856411

1 Introduction

Recent years have witnessed rapid progress of hand pose tracking and hand motion analysis using consumer depth sensors. State-of-the-art techniques [Tagliasacchi et al. 2015][Sun et al. 2015] are able to accurately track hand motion and handle intricate geometric configurations with complex contact patterns among fingers in real-time. However, most of them require that hands are in the air and cannot interact with physical objects. Such a requirement diminishes their utility for virtual instrument applications due to two reasons: First, users can quickly get tired when hands are not supported by some physical object. Second, mid-air interactions do not provide user any feedback, hence users may feel difficult to position their fingers and map them to the keys or strings of virtual instrument.

This paper aims at developing a virtual piano application, which allows users to put their hands on a desk, move them sideways and then tap fingers on the desk, like playing a real piano. There are two major technical challenges in this application. First, the system must track the positions of fingertips and detect their status, i.e., whether a finger is tapping or not. Due to frequent interaction between fingers and desk, the existing hand tracking algorithms often fail. Second, piano-playing gestures are usually fast and complex, involving highly flexible hand articulation and causing severe hand self-occlusion.

To tackle these challenges, we propose a virtual-piano tailored method to track fingertip and detect finger tapping using an RGB-D camera in real-time. We first collect a training dataset with 7200 RGB-D images, covering the most common finger articulation for playing piano. After manually labelling the positions of seven hand joints including five fingertips, thumb MCP joint and wrist center, we train a random regression forest to predict them using depth context features of spatial-voting pixels randomly sampled over the training images. During online testing, we first predict the positions of the hand joints from raw RGB-D images with the trained random forest. Then we use the trajectories of these joints to detect and locate finger tapping using support vector machine (SVM) classification. The virtual piano is registered onto the desk surface using pre-detected normal vector and centroid of the desk surface. Based on the locations of fingertips and the finger tapping status, the system can hereby determine which piano key is pressed and play the corresponding sound. Preliminary results show that our

method can recognize the basic piano-playing gestures in realtime for soothing rhythms. Figure 1 illustrates our virtual piano application with a DepthSense® 325 sensor on top of the desk and in front of the user. We render the hand and the piano based on the coordinates of the desk surface and the detected hand pose from the RGB-D images.

2 Related Work

Hand pose tracking and analysis is a fundamental problem in computer graphics and vision, and is central for many human-computer interfaces. Early gesture recognition applications resort to the use of data gloves or uniquely colored gloves/markers on hands or fingers [Aristidou and Lasenby 2010]. In recent years there has been a growing interest in *non-invasive* setup using a single commodity RGB-D sensor, such as Microsoft Kinect, Intel RealSense, or purpose-designed hardware, e.g., the Leap Motion Controller. Such single-camera acquisition does not impede user movements, hereby is particularly advantageous to VR applications. This section briefly reviews related work on hand pose tracking, finger action recognition and virtual musical instrument.

2.1 Hand Pose Tracking

Algorithms for vision-based hand pose tracking can be broadly categorized as generative model-fitting methods and discriminative methods. Each class of algorithms have their own merits and drawbacks. The model-fitting methods [Melax et al. 2013][Tagliasacchi et al. 2015] reconstruct hand poses by fitting a 3D articulated hand model to depth images. These methods work well in controlled environments, however, they usually require calibration and their results are sensitive to initialization. The discriminative methods require an annotated dataset to learn a regressor offline, and then use it to predict the hand pose online. Such methods are robust to initialization, but their accuracy heavily depends on the size of the training dataset. Therefore, the dataset must be reasonably large to cover the possible hand and finger articulations for a specific application.

The state-of-the-art methods (e.g.,[Tang et al. 2013; Sun et al. 2015; Xu and Cheng 2013]) require that the hand is in the air and not interacting with other objects. The reason is that, hand motion itself is of high degrees-of-freedom and thus requires lots of training data to represent such flexibility. Thus, if the hand is interacting with unknown objects, there will be more unpredictable complexity, e.g., hand self-occlusion and occlusion between object and hand, and the large appearance variations of both the hand and interacting objects. These difficulties hinder researches in this area. There are some previous work that can support hand interacting with objects, but they either assume that the geometrical details of the object is known so that object and hand can complement each other to improve pose estimation [Oikonomidis et al. 2011] via the physical constraints between them or confine the feasible hand articulations to be within a small set of templates [Rogez et al. 2014]. In [Oikonomidis et al. 2011] the type and exact size of the interacting object are assumed to be known in advance. The poses of the object and hand are jointly solved in a generative model-fitting framework using a multi-camera setting to reduce prediction ambiguity, which maximizes the models' compatibility to the image inputs and minimizes the intersection between hand and objects to find their pose. In [Rogez et al. 2014] the hand is allowed to interact with other objects, such as bottles, desk surfaces, etc., and the hand pose is predicted in a discriminative manner by training a multi-class cascade classifier on a dataset that covers many interacting examples between hand and objects. However, in their hand pose estimation framework, the hand posture is only assumed to belong to a small set of pre-defined templates. This is far from our need to play the piano in the proposed application, in which we need to track the accurate articulated fingertip positions and wrist positions, so that the system can detect whether a finger tap is performed by the performer or not.

Among the discriminative methods, random regression forest and its variants have proven effective to capture hand pose in depth images [Xu and Cheng 2013; Tang et al. 2013; Liang et al. 2015; Sun et al. 2015]. In [Xu and Cheng 2013], it is used to regress for hand joint angles directly. With a pre-trained forest, a set of voting pixels cast their votes for each joint angle, which are fused into several candidate hand poses. An extra model-matching stage is needed to find the optimal pose. In [Tang et al. 2013] a transductive regression forest is proposed to alleviate the discrepancy between synthesis and real-world data to improve prediction accuracy. In [Liang et al. 2015] a multi-modal prediction fusion algorithm is proposed to utilize hand motion constraints to resolve the ambiguous pose predictions from random regression forest, so that infeasible hand postures can be avoided. In [Sun et al. 2015], a hierarchical regression scheme is built upon the regression forest for hand pose estimation, in which the root joints of hand skeleton are predicted first and other joints are predicted subsequently based on the root joints, which proves to improve prediction accuracy largely. While these methods work only for in-air hands, we propose to utilize the regression forest for hand in interaction with planar objects.

2.2 Finger Action Recognition

To extract discriminative features and find effective learning models are the two key issues in every pattern recognition problem. Actions are spatio-temporal patterns, which requires comprehensive features gathering information from time domain as well as space domain to define the problem. It's generally known that absolute 3D joints positions are helpful in detecting actions of human body. Multi-camera motion capture (MoCap) systems[Campbell and Bobick 1995] has been widely used to obtain accurate 3D joint position of human body. Similar techniques include data glove (http://www.5dt.com), which provides accurate tracking and haptic feedback. Action recognition based on 3D joints positions retrieved from such devices has been well-studied. There have been many different temporal models in detecting human body actions. Lv and Nevatia [2006] used Hidden Markov Model over pre-defined relative positions obtained from the 3D joints. Han et al. [2010] used conditional random field over 3D joint positions. For actions with complex articulated structure, motions of individual joints are sometimes correlated. Relative positions between joints can be more discriminative features than the absolute position of individual joint [Zhu et al. 2008]. However, these techniques tracks human body motion involving many intermediate joints, and the motions are generally easily observant and has bigger differences in between than finger motions. Besides, for tapping gesture, y coordinate, namely moving direction of tapping finger, embeds more information compared to the remaining two directions. Yi et al. [2015] detected the falling edge of Y coordinate as a tap, and modified the method in [Palshikar et al. 2009] to detect the peak value of changes in Y coordinate. However, their techniques only consider the tapping action as one instant motion instead of two separate actions: up and down. Our approach makes use of the relative position of pair-wise fingertip and individual joints motion trends as features for tapping detection, and considers both up and down directions.

2.3 Virtual Musical Instruments

There are many research efforts to develop virtual and augmented musical instruments in the past decades. Virtual reality and/or aug-

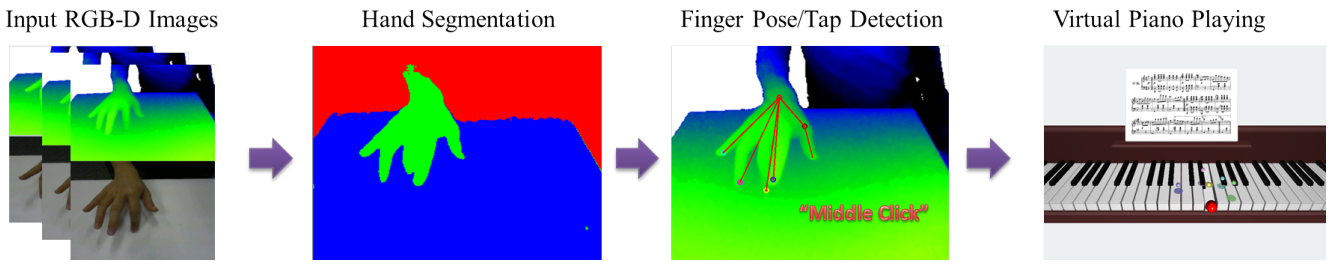


Figure 2: *Fingertip tracking and tapping detection for virtual piano playing.*

mented reality techniques are utilized to increase instrument accessibility, improve user's psycho-pleasure and provide performance guidance [Rogers et al. 2014][Dirkse 2009][Chow et al. 2013][Lin and Liu 2006]. Broersen and Nijholt [2002] developed a virtual piano, which allows multi-agents to play and is useful for educational purpose. However, it uses a real synthesizer or mouse/keyboard as input device, making it non-intuitive to play. Other applications involve instrument-like gestural controllers, such as video camera [Modler and Myatt 2008][Yeh et al. 2010], motion capture [Nymoen et al. 2011], multi-touch device [Ren et al. 2012], data glove [Mitchell et al. 2012], and more recently depth sensors.

Digito [Gillian and Paradiso 2012] is a gesturally controlled virtual musical instrument which utilizes 3D depth sensor to recognize hand gesture with machine learning algorithms and triggers the note to be played by using a "tap" style gesture with the tip of the index finger of the right hand. However, the user experience of Digito is too much different from real playing piano with different fingers. Some applications are developed using Leap Motion Controller to construct virtual piano using 3D positioning of fingers to detect the tapping [Heavers], but in these applications user's hands are not allowed to interact with any object, which is unnatural and uncomfortable for piano player. Han and Gold [2014] conducted a detailed examination on Leap Motion as the tracking device and algorithm for playing piano, which shows that although Leap Motion provide accurate tracking for free hand postures, when there is no interaction of hand with any object, it's difficult for player to determine the position and height of the virtual keyboard without prior practices. Our approach allows users to put their bare hand on a planar object and tap on it, like playing on a real piano.

3 Overview

As discussed in Section 1, our goal is to predict both the motion of the five fingers and the tapping action of each finger from the input RGB-D image sequence to play the virtual piano. Let $\mathbf{\Phi} = \{\phi_k\}_{k=1}^{K}$, $K = 7$, denote the seven joint positions of the hand including five fingertips, thumb MCP joint and wrist center, and $L = \{l_k\}_{k=1}^{5}, l_k \in \{0, 1\}$ denote the tapping action of the five fingers. $l_k = 1$ means the k_{th} finger tapping the desk and $l_k = 0$ means no tapping. Let us also denote by $I_{1:t}$ the input RGB-D sequence, where $I_t = \{I_t^c, I_t^d\}$ are the t^{th} RGB and depth images. The problem of fingertip tracking and tapping detection can thus be formulated as a joint regression and classification task, stated as follows:

$$\mathbf{\Phi}^*_{1:t}, L^*_t = \arg\max_{\mathbf{\Phi}_{1:t}, L_t} P(\mathbf{\Phi}_{1:t}, L_t | I_{1:t}). \tag{1}$$

Note that the action of finger tapping is highly correlated with the hand joint motion, we thus propose to first infer the joint positions from the RGB-D image inputs and then detect the finger tapping based on the predicted 3D trajectories of the joints. Therefore, the problem can be reformulated as:

$$\begin{aligned} \mathbf{\Phi}^*_{1:t}, L^*_t &= \arg\max_{\mathbf{\Phi}_{1:t}, L_t} P(\mathbf{\Phi}_{1:t}, L_t | I_{1:t}) \\ &= \arg\max_{\mathbf{\Phi}_{1:t}, L_t} P(L_t | \mathbf{\Phi}_{1:t}, I_{1:t}) P(\mathbf{\Phi}_{1:t} | I_{1:t}) \\ &= \arg\max_{\mathbf{\Phi}_{1:t}, L_t} P(L_t | \mathbf{\Phi}_{1:t}) P(\mathbf{\Phi}_{1:t} | I_{1:t}), \end{aligned} \tag{2}$$

where we assume that finger tapping can be detected based on the 3D joints trajectories only. Therefore, the joint trajectories and the finger tapping status can be estimated in a two-step manner. In the first step, the optimal trajectories $\mathbf{\Phi}^*_{1:t}$ are estimated by maximizing $P(\mathbf{\Phi}_{1:t} | I_{1:t})$, which can then be utilized to predict L^*_t.

We develop a virtual piano application enabling fingertip tracking and tapping gesture detection, which can let users play a virtual piano keyboard on any plane as a force feedback. We especially design the application for starter-level piano players, who start playing with slow and simple practice songs. In such cases, the fingertip motions can be accurately tracked, and tapping can be identified relatively robustly based on hand joint trajectories only. Our application is developed with DepthSense® 325 as the RGB-D sensor, which consists of three components:

- Fingertip Tracking (see Sections 4 and 5) takes RGB-D images as inputs, extracts the hand from the reference plane, and computes the positions of hand joints;
- Tapping Detection (see Section 6) converts five fingertip locations into global coordinate system, computes the height relative to the reference plane and the relative positions of each pair-wise fingertip, and finally generates tapping event based on the spatial-temporal features retrieved from motion trajectory data.
- Rendering and Feedback takes the tapping event as input, triggers virtual piano key event and sound system, and finally provides a visual and sound feedback to the user.

4 Hand Segmentation

To ensure high quality of hand pose estimation, the hand region needs to be segmented accurately from the background in the depth image. To find the hands, we perform per-pixel skin color detection [Hammer and Beyerer 2013; Li and Kitani 2013]. However, the detected skin mask is not always reliable and background pixels can be misclassified into the hand region, as illustrated in Figure 3. To improve the results, we propose to first fit a plane to the desk surface using the RANSAC algorithm [Fischler and Bolles 1981] in the depth image and then differentiate the points that do not fit the plane as the hand region. However, as the hand occupies a large portion of the foreground depth image, it introduces many outliers for 3D plane fitting. This large number of outliers can affect the

RANSAC algorithm as it will need much more iterations to find the best set of points that fit the plane. To address this problem, we find the hand region in the depth image with the skin color detection results, then use RANSAC to fit a plane with the remaining points. Based on the detected plane, the hand can be better segmented in depth images. In addition, we use the normal vector centered at the desk as the origin of the coordinate to construct the virtual piano for interaction.

The input RGB image is transformed into YCbCr space as it proves more robust to illumination variation for skin detection. The Gaussian Mixture Model is chosen to represent the distributions of skin and non-skin pixels in the color space:

$$P(\boldsymbol{c}|s) = \sum_{i=1}^{N} \rho_{i,s}\mathcal{N}(\mu_{i,s}, C_{i,s}), \tag{3}$$

where $\boldsymbol{c}$ is a color value, $s \in \{1, 0\}$ is the label of skin/non-skin regions, $\mathcal{N}(\mu_{i,s}, C_{i,s})$ is a single Gaussian component with mean $\mu_{i,s}$ and covariance $C_{i,s}$ and $\rho_{i,s}$ is the weight of each Gaussian component. The parameters in (3) are estimated for both skin and non-skin regions with the Expectation-Maximization algorithm using annotated skin mask training data. By assuming equal priors for the skin/non-skin regions, a pixel is classified as skin if $P(s = 1|c) = P(\boldsymbol{c}|s = 1)/\sum_{s\in\{0,1\}} P(\boldsymbol{c}|s) > 0.5$. The detected skin mask in RGB image is then mapped to the depth image coordinates with camera calibration parameters, as shown in Figure 3 (d). The cropped hand region with this hand mask is illustrated in 3 (e), which still contains many background pixels. Therefore, we further utilize RANSAC based 3D plane fitting to remove such pixels.

Let $\boldsymbol{d}_c$ and $\boldsymbol{v}_c$ denote the normal vector and center of the 3D plane, and $(,)$ the dot product. Denote by U the set of pixels outside skin mask in the depth image. We define an indicator function to check whether a point $\boldsymbol{v}$ is an outlier of the plane or not:

$$I(\boldsymbol{v}|\boldsymbol{d}_c, \boldsymbol{v}_c) = \begin{cases} 1 & if|(\boldsymbol{v} - \boldsymbol{v}_c, \boldsymbol{d}_c)| \leq \delta \\ 0 & otherwise \end{cases}. \tag{4}$$

In our implementation we set $\delta = 1.2cm$ to achieve both robustness to noisy depth measurement and accurate hand segmentation. The plane can then be estimated by the RANSAC algorithm. More accurate hand segmentation U_h is then obtained by finding the pixels that do not fit the plane well. We outline the plane fitting and hand segmentation in pseudocodes in Algorithm 1.

Algorithm 1 RANSAC based Plane Fitting & Hand Segmentation

$i = 1, N_{max} = -\infty$;
while $i \leq N_{it}$ **do**
  Randomly sample a set U_r of 3 points from U;
  Calculate the plane fitted to U_r:
  $\boldsymbol{d}_c^*, \boldsymbol{v}_c^* = \arg\max_{\boldsymbol{d}_c, \boldsymbol{v}_c} \sum_{\boldsymbol{v}_i \in U_r} |(\boldsymbol{v}_i - \boldsymbol{v}_c, \boldsymbol{d}_c)|^2$
  Get the consensus set $U_c = \{v|I(\boldsymbol{v}|\boldsymbol{d}_c^*, \boldsymbol{v}_c^*) = 1, \boldsymbol{v} \in U\}$;
  if $|U_c| > N_{max}$ **then**
    Calculate the plane fitted to U_c:
    $\boldsymbol{d}_c^*, \boldsymbol{v}_c^* = \arg\max_{\boldsymbol{d}_c, \boldsymbol{v}_c} \sum_{\boldsymbol{v}_i \in U_c} |(\boldsymbol{v}_i - \boldsymbol{v}_c, \boldsymbol{d}_c)|^2$
    $N_{max} = |U_c|$;
  end if
end while
$U_h = \{\boldsymbol{v}|(\boldsymbol{v} - \boldsymbol{v}_c^*, \boldsymbol{d}_c^*)| > \delta\}$;

5 Hand Pose Tracking

Once the hand is segmented, we can then use the random regression forest [Girshick et al. 2011] to predict the 3D positions of the seven joints of the hand. The regression forest is an ensemble of several random regression trees, each of which consists of a number of split nodes and leaf nodes. Each split node contains one split function learnt from the training data to branch to the child node based on the feature values of the descriptor of an input pixel i. Each leaf node contains the distributions over the 3D relative offsets to the joint positions, which are collected from the training samples.

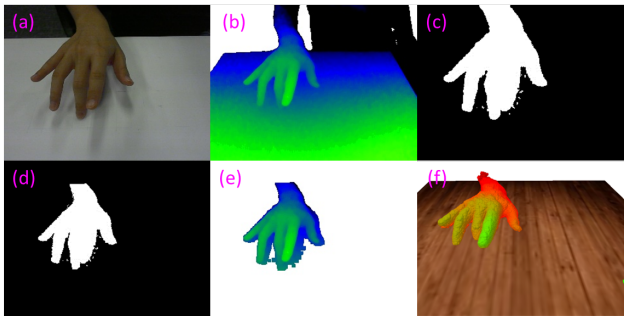


Figure 3: *Hand segmentation. (a) and (b): input RGB-D images; (c) skin mask in RGB image; (d) skin mask mapped to depth image; (e) hand segmentation without 3D plane fitting; (f) fitted 3D plane and final hand segmentation.*

Let us consider a forest with T trees. The depth context descriptor is adopted as the feature for regression [Liang et al. 2015]. Let it be $\boldsymbol{\mathcal{D}}$. To train the regression forest, a set of pixels are randomly sampled from each training image, and each sample pixel i is associated with the ground truth offsets between its 3D position and the seven joints, i.e., $\boldsymbol{\Delta}_{ik}, k = 1, ..., K$. Each split node of the tree has two child nodes and is associated with a learned split function to determine which branch the test sample goes into. It takes the following form:

$$\mathcal{D}^m \leq \tau, \tag{5}$$

where $\mathcal{D}^m$ is the m^{th} dimension of the depth context descriptor, and τ is a threshold to determine the branch to one of the child nodes, i.e., the left child for $\mathcal{D}^m \leq \tau$ and right child otherwise. To learn the split function at the split nodes, a set of candidate split functions $\{\psi\}$ are generated by sampling m and τ. The optimal split function is selected based on the following metric according to the pose annotation distribution of the training samples that reach the node:

$$\psi^* = \arg\max_{\psi} \left[H(A) - \sum_{s\in\{l,r\}} \frac{|A_s(\psi)|}{|A|} H(A_s(\psi)) \right], \tag{6}$$

where A is the samples reaching the current node and A_l and A_r are the two subsets of A split by ψ. To regress for the hand pose, the function $H(A)$ is defined as the sum of variances of the offsets to the seven joints among the samples in A to measure the pose uncertainty.

At the leaf nodes, the regression models for the relative votes to the seven joints are learned from the training samples. We define the regression model as a single relative vote $(\boldsymbol{\Delta}_k, w_k)$ for each joint, where $\boldsymbol{\Delta}_k$ represents the possible prediction of the relative offset based on the relative offsets $\{\boldsymbol{\Delta}_{ik}\}$ from the training samples, and w_k represents the confidence of the prediction. As in [Girshick et al. 2011], they can be obtained by mode-seeking with the ground truth offsets of the training samples reaching the leaf node.

In the testing phase, the trained forest is utilized to predict the hand joint positions. Figure 4 illustrates the prediction pipeline for a single joint, i.e., the middle fingertip, with one tree. First, a set

of voting pixels are sampled from the hand region U_h, and each voting pixel i will recursively branch down the tree and reach one leaf node in each regression tree in the forest based on its depth context descriptor. Since the forest consists of T trees, each pixel reaches in total T leaf nodes in the forest and thus retrieves T votes $\{\boldsymbol{\Delta}_{ijk}, w_{ijk}\}_{j=1}^{T}$ for a joint ϕ_k in total, where $\boldsymbol{\Delta}_{ijk}$ is the 3D relative offset between the 3D position of the pixel and the objective; w_{ijk} is the weight of the vote. Given the 3D position $\boldsymbol{v}_i$ of the pixel i, the relative votes can be converted to the absolute votes $\{\boldsymbol{v}_{ijk}, w_{ijk}\}_{j=1}^{T}$, where $\boldsymbol{v}_{ijk} = \boldsymbol{\Delta}_{ijk} + \boldsymbol{v}_i$. Similar to [Girshick et al. 2011], we use the weighted Parzen density estimator with Gaussian kernel to evaluate the single-frame posterior $P(\phi_k|I)$ for each joint independently, and the overall posterior $P(\boldsymbol{\Phi}|I)$ can thus be formulated as:

$$\begin{aligned} P(\boldsymbol{\Phi}|I) &= \prod_{k=1}^{K} P(\phi_k|I) \\ &= \prod_{k=1}^{K} \left[\sum_{i \in U_h, j \in [1,\dots,T]} w_{ijk} \mathcal{N}(\phi_k | \boldsymbol{v}_{ijk}, \delta_v^2) \right], \end{aligned} \tag{7}$$

where we assume an isotropic variance δ_v in all three dimensions for each joint in the Gaussian kernel. For single-frame prediction we can assume a uniform prior for the hand pose $\boldsymbol{\Phi}$ and thus the likelihood function can be taken as $P(I_t|\boldsymbol{\Phi}_t) \propto P(\boldsymbol{\Phi}_t|I_t)$. We utilize it to estimate $\boldsymbol{\Phi}^*_{1:t}$ in an iterative manner to approximate the optimal solution for formula (2). That is, in each iteration the optimal estimation $\boldsymbol{\Phi}^*_t$ for the current frame is predicted based on the current image observation I_t and the past optimal estimation $\boldsymbol{\Phi}^*_{1:t-1}$. The task is formulated as:

$$\begin{aligned} \boldsymbol{\Phi}^*_t &= \arg\max_{\boldsymbol{\Phi}_t} P(\boldsymbol{\Phi}_t, \boldsymbol{\Phi}^*_{1:t-1}|I_{1:t}) \\ &= \arg\max_{\boldsymbol{\Phi}_t} P(I_t|\boldsymbol{\Phi}_t) P(\boldsymbol{\Phi}_t|\boldsymbol{\Phi}^*_{t-1}) P(\boldsymbol{\Phi}^*_{1:t-1}|I_{1:t-1}) \\ &= \arg\max_{\boldsymbol{\Phi}_t} P(I_t|\boldsymbol{\Phi}_t) P(\boldsymbol{\Phi}_t|\boldsymbol{\Phi}^*_{t-1}), \end{aligned} \tag{8}$$

where the state transition function $P(\boldsymbol{\Phi}_t|\boldsymbol{\Phi}^*_{t-1})$ is defined as a Gaussian distribution $\mathcal{N}(\boldsymbol{\Phi}_t|\boldsymbol{\Phi}^*_{t-1}, C)$ with mean $\boldsymbol{\Phi}^*_{t-1}$ and covariance matrix C. We take C as a block diagonal matrix. Therefore, the above optimization problem can be solved for each hand joint ϕ_k independently by combining (7) and (8):

$$\phi^*_{k,t} = \arg\max_{\phi_{k,t}} \mathcal{N}(\phi_{k,t}|\phi^*_{k,t-1}, C_k) \sum_{i \in U_h, j} w_{ijk} \mathcal{N}(\phi_{k,t}|\boldsymbol{v}_{ijk}, \delta_v^2), \tag{9}$$

where C_k is a submatrix of C corresponding to joint k. Note that the above formula is essentially sum of Gaussians, this problem can thus be efficiently solved by the mean-shift algorithm [Comaniciu and Meer 2002].

6 Finger Tapping Detection

With the hand joint positions estimated from previous sections, we can detect the tapping event based on the motion trajectories of the five fingertips. In most cases, a tapping action is only an instant action when a finger moves down, reaches the local height minimum and followed by moving up. In the case of playing piano, the finger sometimes needs to stay at the plane for a period of time before moving up as a result of playing notes with different duration. Therefore, in our application, we deal with two tapping events, i.e., tapping down and tapping up. A tapping down event will trigger the press event of a piano key, while a tapping up event will trigger the release event of a piano key. The sound duration of each key pressed is thus determined by the time difference between each pair of tapping down and tapping up events.

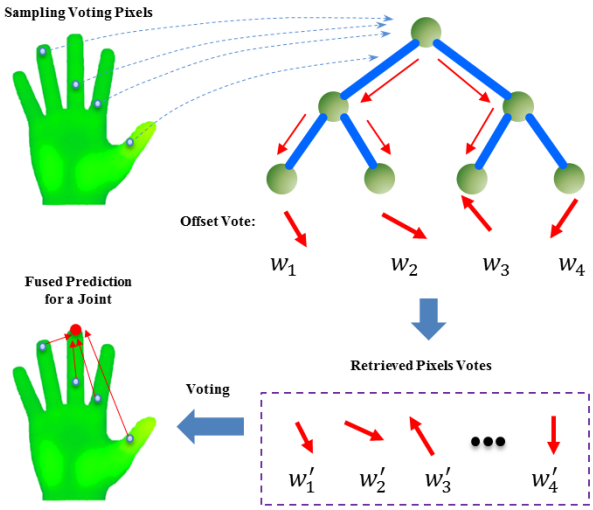


Figure 4: *Hand pose prediction with the random forest. The red arrows represent the votes of the relative offset pointing to the joint to be predicted. The red circle in the left-bottom denoted the predicted middle fingertip position.*

For beginner's level, single finger tapping is mostly performed in practice. Therefore we trained 2 classifiers, namely C_{down} and C_{up} to detect tapping down and tapping up separately, each of which has 6 classes including a non-action class and the tap-action classes for the five fingers.

A tapping down event TD_i is defined as the moment after the finger i moves down and finally reaches the table plane at frame F_k, and we label frame F_k and its previous 3 frames in C_{down} as TD_i. Similarly, a tapping up event TU_i happens when the finger i just leaves the table plane at frame F_k, and we label frame F_k and its following 3 frames in C_{up} as TU_i. For the rest cases, we label the frame in the classifier as NT. As tapping is an action over time, we need motion data over a time period to define a tapping event. We thus define a sliding window of size N, and use motion data from the previous $N-1$ frames and the current frame to determine whether the current frame triggers a tapping event or not. Based on experiments, we find the tapping down duration, when the fingertip moves down from its local maximum height till it reaches the minimum, is about 200ms. Given the FPS for depth camera is about 20, the number of frames over a typical tapping duration is 4-5, so we set window size $N = 5$ empirically. As there exist correlated movements between fingers, that one finger will passively move due to an active finger's movement, we need to consider the motion data from all fingertips in order to identify the tapping event for active finger. In our system, two sets of features are used for tapping event classification. First, as hands have complex articulated structure, relationship between joints has more information compared to individual ones. Based on the table normal vector estimate, we compute the heights of all fingertips toward the table plane by projecting the fingertip position to the normal vector. Then we compute the relative heights for all fingertips to each other. For any pair of heights of fingertips H_i and H_j, we define H_{ij} = H_i-H_j as the relative heights of finger j to finger i. For all five fingers, there are in total 25 pairs of relative heights. With a sliding window of size N, all relative heights pairs over the last N frames are set as the first feature set. Then we compute the mean, variance, interquartile, the slope of linear regression and peak range [Yi et al. 2015] of the relative heights over last N

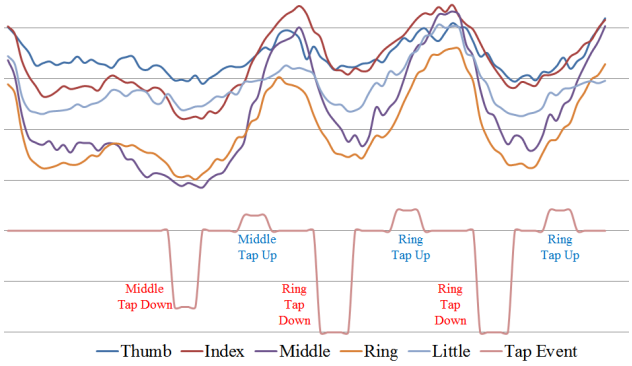


Figure 5: *Finger tapping detection: heights relative to table plane for all finger tips and action labelled*

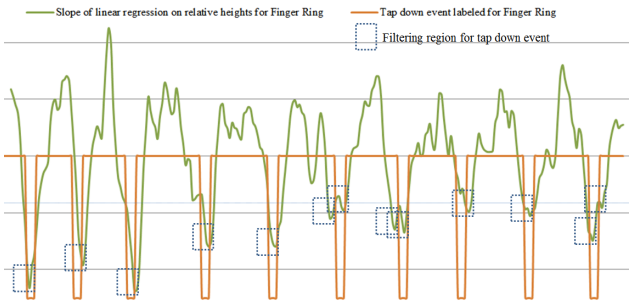


Figure 6: *Finger tap detection: filtering region based on height slope.*

frames for all the fingertips. We combine the two sets of features together and train 5 SVM classifiers. We classify all the frames into tapping-down, tapping-up or non-tapping.

As moving up and down are common finger actions, it is highly possible that during one tapping action, multiple frames are identified as tapping up or down events. The fluctuating tapping events detected will result in an unpleasant sound effect in virtual piano application. Therefore we further implemented a filter on actions detected for each finger. Observed that $0 \sim 2$ frames before the tapping down event moment have significant small values in slope estimated over that window as a result of fast moving down as shown in Figure 6, we capture the local minimum value less than a threshold $Thres_{slope_i}$ in slope estimated and conduct a vote among the last N frames: if number of the tapping down event TD_i is detected more than $Thres_{vote_i}$ times for finger i in the N frames, we treat the current frame as a tapping down event. Once we detect a tapping down event for a finger, we set a flag indicating that the finger is down now and only capture tapping up events in the next few frames. If there is no tapping up event detected, we release the key and search for the next tapping down event.

7 Experiments & Discussions

Experiment Setup. We implemented the fingertip tracking and tapping detection algorithm in C++/OpenCV and rendered the virtual piano using OpenGL. We adopted a DepthSense® 325 sensor on top of the desk and in front of the user. The system was tested on a PC with an Intel i7 3.3GHz CPU and 16GB RAM. It is worth noting that the time cost to process one frame is only $20ms$, which is efficient enough for realtime tracking. After training for 20 minutes, a user with little musical playground can play a simple adagio melody with our virtual piano. See the accompanying video.

Table 1: *Precision and recall for individual tapping down class.*

	Precision	Recall
Thumb	88.99%	97.00%
Index	100.00%	95.00%
Middle	96.77%	90.00%
Ring	85.34%	99.00%
Pinky	100.00%	87.00%

Training. To validate the effectiveness of the proposed hand pose tracking algorithm, we collect a dataset of real-world hand images consisting of around 7.2k depth images of two subjects performing various finger tapping postures to play the virtual piano. The resolution of these images is 320×240. The subjects can either put their hand over or on the desk. The hand poses collected cover the most frequent gestures for playing piano in the view of depth camera, and the poses are music score independent. In each of the image we manually annotate the 3D positions of the seven joints of the hand. In this experiment we set the number of trees in the forest to be 3. During training, we randomly sample 150 pixels from each training image and generated 6000 candidate split functions to learn the tree structure. The tree stops growing if its depth exceeds 20 or the node sample is less than 50. During testing, a number of 500 voting pixels are randomly sampled from the segmented hand region to predict the hand joint positions.

To collect the training data set for tapping, we manually label several sequences of RGB images, including over 100 taps for each finger. We label the tapping down moment frame and its previous 3 frames as TD frames, and label the tapping up moment frame and its following 3 frames as TU frames. The other frames are labeled as non-tapping frames. These annotated data are then used to train the SVM classifier for tap detection.

Performance. We perform 4-fold cross validation on this dataset to evaluate the performance of the proposed method. The prediction performance of a joint is evaluated in terms of the percentage of its predictions that are within a distance of D_T centimeters from the ground truth in the test images. This metric is averaged for all the seven joints to obtain the overall evaluation. To better understand the performance of the method, we present the results for different D_T so that the distribution of the predictions over different intervals of D_T can be observed, as shown in Figure 7. The average error between the ground truth hand joint positions and the predicted positions is $1.3cm$. Figure 8 shows the hand pose prediction results on some sample frames in the dataset. We can see that the proposed method can accurately recover the positions of the hand joints, when fingers are in the air and on the reference plane. In contrast, the commercial products, such as Leap Motion, Intel RealSense and SoftKinetic, are not able to detect the hand joints for those cases. To test tapping detection algorithm, we ask 2 users to perform 100 taps totally on each finger with around 1 second time difference in between. We consider a tapping down and tapping up event classified successfully if the finger which performed the action is correctly identified within 0.3 second. The result of tapping down detection is shown in Table 1.

Comparison. We compare our method with two state-of-the-art techniques, a model-based algorithm [Tagliasacchi et al. 2015] and Leap Motion Controller – the leading commercial product for hand tracking. These methods are able to accurately track (multiple) hands when they are in the air, however, they fail when hands are interacting with physical objects. In contrast, our algorithm is specifically designed for hand interaction with planar objects, hereby has

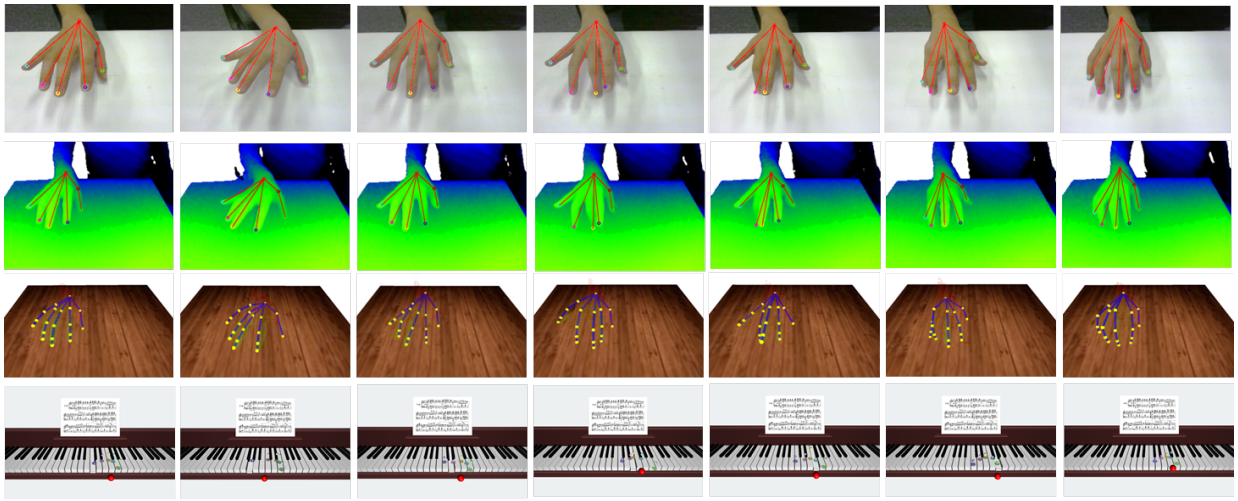


Figure 8: *Exemplar frames of hand pose prediction. Rows 1 & 2: input RGB and depth images overlaid with predicted 7 hand joints. Row 3: reconstructed hand skeleton via applying inverse kinematic to the joints. Row 4: playing virtual piano. Zoom in to see the details.*

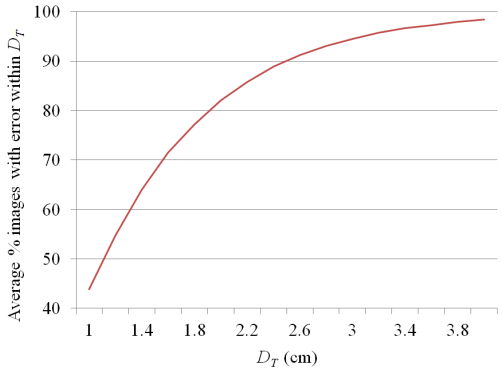


Figure 7: *Average hand pose prediction accuracy with different* D_T.

better performance and accuracy in the virtual piano application.

Limitations. Although our method can track most of the beginner's piano-playing gestures for soothing rhythms in realtime, our virtual piano has several limitations compared with playing *real* piano. (1) The proposed tracking algorithm is not quite robust to hand-shape variations, e.g., the prediction accuracy drops when the shape and/or size of player's hand are significantly different from the ones in the training dataset. (2) Thumb under is a common gesture, where the thumb is brought under the hand in order to pass the 3rd or 4th finger for playing the scale. Due to severe occlusion, the depth sensor is not able to capture the thumb and our tracking algorithm cannot detect it either. (3) Our current implementation is not efficient and accurate enough to detect the tapping event in a fast tempo. (4) Our method supports two-hand tracking. However, due to the limited viewing volume of the DepthSense® 325 sensor, users can only play with a single hand for about 2 octaves.

8 Conclusion & Future Work

This paper presented a virtual piano application that allows users to play with bare hands on or near a planar surface. Taking the RGB-D images as input, our method uses an offline trained random regression forest to track the fingertips and detect the finger tapping. Compared with the existing hand tracking algorithms, our method is designed for hand interaction with planar objects. Preliminary results show that our method can recognize most of the beginner's piano-playing gestures for soothing rhythms in realtime. The system can be further integrated with head-mounted display, such as Oculus Rift, to provide with user an immersive visual and audial environment, which may further support remote learning and gamification in musical instrument learning. In a broader sense, our work provides a pipeline to solve hand integration with planar objects and a general solution to such type of application, which draws the community's attention to the limitation of current mid-air hand tracking techniques.

In the future, we will expand the gesture database for intermediate and advanced players and improve the accuracy of our tracking algorithm for allegro rhythms. To detect self-occluded gestures, some graphical machine learning model will be applied to predict occluded finger position and tapping moment with the help of domain knowledge. We will also develop a hand normalization algorithm so that players whose hands are significantly different from those of the training dataset can use our system. Moreover, we will conduct a formal user study to evaluate the efficacy of the proposed system.

Acknowledgement

We thank the anonymous reviewers for their constructive comments. This project was partially funded by Singapore MOE2013-T2-2-011, MOE RG40/12, MOE RG23/15, the Economic Development Board and the National Research Foundation of Singapore, the NSFC Grants (61322206, 61521002) and the Foundation of TNList.

References

ARISTIDOU, A., AND LASENBY, J. 2010. Motion capture with constrained inverse kinematics for real-time hand tracking. In *International Symposium on Communications, Control and Signal Processing*, IEEE, 1–5.

BROERSEN, A., AND NIJHOLT, A. 2002. Developing a virtual piano playing environment. In *IEEE International conference on Advanced Learning Technologies (ICALT 2002)*, 278–282.

CAMPBELL, L. W., AND BOBICK, A. E. 1995. Recognition of human body motion using phase space constraints. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, IEEE, 624–630.

CHOW, J., FENG, H., AMOR, R., AND WÜNSCHE, B. C. 2013. Music education using augmented reality with a head mounted display. In *Proceedings of the Fourteenth Australasian User Interface Conference-Volume 139*, Australian Computer Society, Inc., 73–79.

COMANICIU, D., AND MEER, P. 2002. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. PAMI 24*, 5, 603–619.

DIRKSE, S. 2009. *A survey of the development of sight-reading skills in instructional piano methods for average-age beginners and a sample primer-level sight-reading curriculum*. University of South Carolina.

FISCHLER, M. A., AND BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24*, 6, 381–395.

GILLIAN, N., AND PARADISO, J. A. 2012. Digito: A fine-grain gesturally controlled virtual musical instrument. In *Proc. NIME*, vol. 2012.

GIRSHICK, R., SHOTTON, J., KOHLI, P., CRIMINISI, A., AND FITZGIBBON, A. 2011. Efficient regression of general-activity human poses from depth images. In *IEEE International Conference on Computer Vision*, IEEE, 415–422.

HAMMER, J. H., AND BEYERER, J. 2013. Robust hand tracking in realtime using a single head-mounted rgb camera. In *International Conference on Human-Computer Interaction*, Springer, 252–261.

HAN, J., AND GOLD, N. 2014. Lessons learned in exploring the leap motion tm sensor for gesture-based instrument design. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, 371–374.

HAN, L., WU, X., LIANG, W., HOU, G., AND JIA, Y. 2010. Discriminative human action recognition in the learned hierarchical manifold space. *Image and Vision Computing 28*, 5, 836–849.

HEAVERS, M. Vimeo video: Leap motion air piano. https://vimeo.com/67143314.

LI, C., AND KITANI, K. M. 2013. Pixel-level hand detection in ego-centric videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 3570–3577.

LIANG, H., YUAN, J., AND THALMANN, D. 2015. Resolving ambiguous hand pose predictions by exploiting part correlations. *IEEE Trans. Circuits and Systems for Video Technology 25*, 7, 1125–1139.

LIN, C.-C., AND LIU, D. S.-M. 2006. An intelligent virtual piano tutor. In *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, ACM, 353–356.

LV, F., AND NEVATIA, R. 2006. Recognition and segmentation of 3-d human action using hmm and multi-class adaboost. In *Computer Vision–ECCV 2006*. Springer, 359–372.

MELAX, S., KESELMAN, L., AND ORSTEN, S. 2013. Dynamics based 3d skeletal hand tracking. In *Proceedings of Graphics Interface 2013*, 63–70.

MITCHELL, T. J., MADGWICK, S., AND HEAP, I. 2012. Musical interaction with hand posture and orientation: A toolbox of gestural control mechanisms.

MODLER, P., AND MYATT, T. 2008. Video based recognition of hand gestures by neural networks for the control of sound and music. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Citeseer, 5–7.

NYMOEN, K., SKOGSTAD, S. A. V. D., AND JENSENIUS, A. R. 2011. Soundsaber-a motion capture instrument.

OIKONOMIDIS, I., KYRIAZIS, N., ARGYROS, A., ET AL. 2011. Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *IEEE International Conference on Computer Vision*, IEEE, 2088–2095.

PALSHIKAR, G., ET AL. 2009. Simple algorithms for peak detection in time-series. In *Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence*.

REN, Z., MEHRA, R., COPOSKY, J., AND LIN, M. 2012. Designing virtual instruments with touch-enabled interface. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, ACM, 433–436.

ROGERS, K., RÖHLIG, A., WEING, M., GUGENHEIMER, J., KÖNINGS, B., KLEPSCH, M., SCHAUB, F., RUKZIO, E., SEUFERT, T., AND WEBER, M. 2014. Piano: Faster piano learning with interactive projection. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, ACM, 149–158.

ROGEZ, G., KHADEMI, M., SUPANČIČ III, J., MONTIEL, J. M. M., AND RAMANAN, D. 2014. 3d hand pose detection in egocentric rgb-d images. In *ECCV Workshop on Consumer Depth Cameras for Computer Vision*, Springer, 356–371.

SUN, X., WEI, Y., LIANG, S., TANG, X., AND SUN, J. 2015. Cascaded hand pose regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 824–832.

TAGLIASACCHI, A., SCHRÖDER, M., TKACH, A., BOUAZIZ, S., BOTSCH, M., AND PAULY, M. 2015. Robust articulated-icp for real-time hand tracking. *Computer Graphics Forum 34*, 5, 101–114.

TANG, D., YU, T.-H., AND KIM, T.-K. 2013. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 3224–3231.

XU, C., AND CHENG, L. 2013. Efficient hand pose estimation from a single depth image. In *IEEE International Conference on Computer Vision*, IEEE, 3456–3462.

YEH, C.-H., TSENG, W.-Y., BAI, J.-C., YEH, R.-N., WANG, S.-C., AND SUNG, P.-Y. 2010. Virtual piano design via single-view video based on multifinger actions recognition. In *2010 3rd International Conference on Human-Centric Computing*, 1–5.

YI, X., YU, C., ZHANG, M., GAO, S., SUN, K., AND SHI, Y. 2015. Atk: Enabling ten-finger freehand typing in air based on 3d hand tracking data. In *Annual ACM Symposium on User Interface Software and Technology*.

ZHU, L. L., CHEN, Y., LU, Y., LIN, C., AND YUILLE, A. 2008. Max margin and/or graph learning for parsing the human body. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, IEEE, 1–8.