A Collaborative Filtering Approach to Real-Time Hand Pose Estimation

Chiho Choi, Ayan Sinha, Joon Hee Choi, Sujin Jang, Karthik Ramani
Purdue University
West Lafayette, IN 47907, USA
{chihochoi, sinha12, choi240, jang64, ramani}@purdue.edu

Abstract

Collaborative filtering aims to predict unknown user ratings in a recommender system by collectively assessing known user preferences. In this paper, we first draw analogies between collaborative filtering and the pose estimation problem. Specifically, we recast the hand pose estimation problem as the cold-start problem for a new user with unknown item ratings in a recommender system. Inspired by fast and accurate matrix factorization techniques for collaborative filtering, we develop a real-time algorithm for estimating the hand pose from RGB-D data of a commercial depth camera. First, we efficiently identify nearest neighbors using local shape descriptors in the RGB-D domain from a library of hand poses with known pose parameter values. We then use this information to evaluate the unknown pose parameters using a joint matrix factorization and completion (JMFC) approach. Our quantitative and qualitative results suggest that our approach is robust to variation in hand configurations while achieving real time performance ($\approx$ 29 FPS) on a standard computer.

1. Introduction

The easy availability of commercial depth cameras marked the advent of real time solutions to the human pose estimation problem [27]. However, the hand pose estimation problem has proved far more challenging. The manifold challenges to robust hand skeleton tracking are that (1) the hand is a highly articulated object, (2) has many degrees of freedom (DOF) with self-similar parts which often occlude each other, (3) all fingers are flexible, and (4) there exists intra-finger and inter-finger motion constraints [9]. Noise in data acquired from depth sensors further confounds all current methods for hand tracking. Consequently, a robust real time solution to the hand pose estimation problem remains elusive.

The success of hand tracking naturally depends on synthesizing our knowledge of the hand (e.g., geometric shape, constraints on pose configurations) and latent features of the RGB-D data stream (e.g., region of interest, key feature points like finger tips, and temporal continuity) [2]. In this paper, we propose a novel method to achieve this synthesis by drawing on collaborative filtering approaches for recommender systems [12]. Our main insight is that a recommender system (e.g., Netflix) [22] is very similar to a pose tracking system. Both systems have some *intrinsic* and *extrinsic* information about its constituent objects, the users in a recommender system and individual poses in a tracking system. The *intrinsic* knowledge of the hand in a tracking system corresponds to known user ratings in a recommender system. Similarly, the *extrinsic* RGB-D point cloud information corresponds to the metadata available about users (e.g., geographical locations, background, and interest). Specifically, the hand pose estimation problem is analogous to the cold-start problem in recommender systems.

The cold-start problem in recommender systems is to suggest personalized items to a new user with unknown preferences [23]. In analogy to a tracking system, the hand pose estimation problem is to evaluate the unknown pose parameters of the kinematic hand model for a new point clouds appearing at every instant of time via a RGB-D sensor. A common technique to alleviate the cold-start problem is to suggest items to a new user based on recommendations available for like-minded users [19]. The like-mindedness or similarity between users is evaluated using metadata such as age, gender, geographical location, interests, etc [24]. Following a similar approach, we efficiently find the nearest neighbors to an arriving point cloud using local shape descriptors from a large database of hand poses with known parameter values. Subsequently, the unknown pose parameters for this point cloud are estimated by *collaboratively* regressing the known parameters of all neighborhood poses. Our contributions include:

1. Our main contribution is a joint matrix factorization and completion (JMFC) algorithm to estimate the unknown pose parameters from the nearest neighbors on a per frame basis.

2. Construction of a hand pose library using a synthetic hand model which mimics real 3D hand gestures.

3. Efficient nearest neighbor retrieval from the pose library by using a combination of pose clustering, FAST feature point detectors and BRIEF descriptors.

4. Overall, a pragmatic solution to the real-time hand pose estimation problem devoid of training parameters and implementable on a standard computer.

This paper is organized as follows: We review relevant literature in section 2. We discuss the creation of a hand pose library using a synthetic 3D hand model and techniques for nearest neighbor retrieval using local shape descriptors. We propose our novel JMFC algorithm for estimating pose parameters and discuss the details of its implementation in Section 4. Section 5 demonstrates the quantitative performance and we qualitatively show the efficacy of our approach. Conclusions and future work are presented in section 6.

2. Related Work

A variety of approaches have been proposed over the last decade for hand pose estimation. These include, without claim of exhaustivity, wearable (e.g., camera, gloves) and marker based approaches, techniques reliant on RGB input from single or multiple cameras, and more recently depth camera or RGB-D input based approaches. We review some work relevant to our depth-camera based approach and readers are referred to [9] for a comprehensive review of literature.

Approaches for hand-pose estimation can be classified as either model-based (*generative*) methods, or appearance-based (*discriminative*) methods. An explicit hand model guides model-based methods to recover the hand pose, whereas appearance-based methods establish a map between image features and a library of hand pose configurations. Current model-based approaches use particle swarm optimization (PSO) [21] or a Gauss-Seidel solver [20] to resolve the hand configuration. Although straight forward to implement, these methods depend on prior motion for initializing the solvers and have high computational complexity. As a result, the pose estimates from these methods are poor for non-contiguous data and they often rely on a GPU for real-time processing.

Following the pioneering work in human-pose estimation [27], similar appearance based methods are proposed for hand pose estimation in [15, 16, 31]. Compared to a human body, the human hand is smaller, more flexible, and severely affected by self-occlusion. Consequently, these methods lose track under low resolution, output kinematically invalid solutions and lack robustness against occlusion. Discriminative approaches employing inverse kinematics (IK) for pose refinement are 6D Hands which performs nearest neighbor search from a hand pose database [34] and deep convolutional neural network based hand pose recovery system [33]. Nearest neighbor methods perform poorly when introduced to unseen poses not in the database, while training a deep convolutional network is notoriously time-consuming. Although we use nearest neighbors to estimate the pose parameters, our JMFC algorithm circumvents the errors due to unseen poses without any training. Some approaches for hand-tracking locally regress [8] the hand pose parameters to the input image [28, 30, 32]. However, these methods either require a large and comprehensive training dataset to ensure robust tracking [30, 32], or their heuristic initialization [28] causes it to lose track for poses with severe self-occlusion.

The methods in [35, 26] are hybrid methods similar in spirit to ours and leverage the paradigm of *'analysis by synthesis'*. These methods first create a population of hand poses and then select the hand pose that *best* fits the observed depth data by optimizing a scoring function. The heavy computational burden of this optimization means that the system either achieves low frame rates (12 FPS in [35]) or needs to be accelerated using a GPU (as in [26]). Unlike these methods which explicitly maximize the scoring function among all individual pose candidates, our approach implicitly optimizes a scoring function by *collectively* assessing the population of possible hand poses. We use fast algorithms for matrix factorization [1, 7] in our JMFC model to do this optimization.

Symbol	Description
$\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\phi})$	Synthetic hand model
$\boldsymbol{\theta}$	18 joint angle parameters
$\boldsymbol{\phi}$	3 global translation parameters
$\mathbf{s}$	Hand skeleton vertex coordinates
$\mathbf{v}$	Visible point cloud for a hand pose
$\mathbf{d}$	Euclidean distance to poses in basis
$\mathbf{c}$	Shape descriptor for a pose
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	Latent factor matrices
$\mathbf{D}$	Distance matrix
$\mathbf{P}$	Parameter matrix
k	Number of nearest neighbors

Table 1: Summary of key notations used in the paper.

3. Database creation

In this section, we first describe the 3D hand model and the procedure used to create a large library of hand poses. The pose library is annotated with labels we use for determining the hand pose from a depth map. We cluster the poses in the library to generate a set of pose exemplars use-

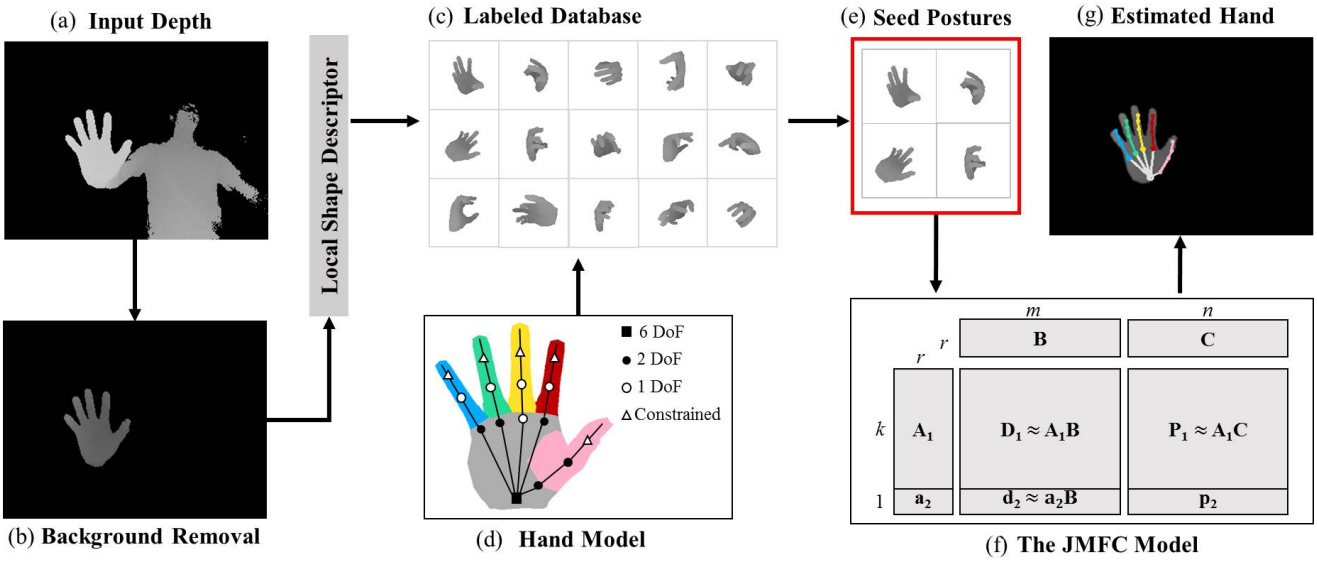


Figure 1: An overview of algorithm pipeline. Background noise in depth map is removed ((a) – (b)). We use a local shape descriptor to retrieve nearest neighbors from the labeled database of various hand configurations ((c) – (d)). The extracted neighbors serve as seed postures to a JMFC model, and unknown joint parameters are estimated using a matrix factorization and completion process ((e) – (g)).

ful for efficient nearest neighbor retrieval. Nearest neighbors are retrieved at runtime by evaluating the shape descriptor distance between the arriving depth data and simulated depth data of the pose exemplars.

3.1. Hand model and synthetic data generation

We statistically generate hand poses using a synthetic 3D hand model. The size of our synthetic hand model represents the median quartile of male hand sizes [11]. Our 3D hand model consists of 1,179 vertices and 2,126 triangular faces. This model is explicitly scaled for individual subjects. We adopt a kinematic hand model with 21 degrees of freedom (DOF), $\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\phi})$, as standard in hand pose estimation problems (see Figure 1d). $\boldsymbol{\theta}$ denotes the set of 18 joint angle parameters and $\boldsymbol{\phi}$ is the set of 3 global translation parameters (x, y and z) of the hand.

Manually creating a library of hand poses using different individuals is a tedious task. Instead, we (1) impose constraints for joint configurations and finger movement as discussed in [18] and [17]; and (2) uniformly sample each of the 18 joint parameters in this restricted configuration space, in order to automatically simulate 118K realistic hand poses. These hand poses are effectively mesh modeled with corresponding skeletal information. In order to synthetically generate point clouds consistent with those visible to a depth camera under occlusion, we process these mesh models using a hidden point removal [14] strategy. Thus, each pose instance in the database is a mesh model with labels $(\boldsymbol{\theta}, \mathbf{s}, \mathbf{v})$, where $\mathbf{s}$ are the coordinates of the skeletal vertices and $\mathbf{v}$ are coordinates of the visible vertices from the viewpoint of a depth camera.

3.2. Pose exemplars and basis

In order to reduce redundancy of poses in the library, we cluster the poses and extract pose exemplars. Density based approaches can automatically detect arbitrary shaped clusters in high dimensional data. To identify pose clusters, we use a combination of two density-based clustering approaches, OPTICS [3] and DBSCAN [10], on the shape descriptor distance described below. The OPTICS algorithm does not explicitly generate clusters, but instead provides an ordering of all hand poses based on their similarities. The density parameters (minimum number of cluster members and maximum cluster radius) are estimated by investigating the output of OPTICS, and these parameters serve as input to DBSCAN. We then extract clusters using DBSCAN, and set the pose with minimum average distance to other cluster members to be the pose exemplar. We identify 1,030 exemplars among the 118K poses in the library, thus greatly improving the efficiency of nearest neighbor retrieval while maintaining accuracy (see Figure 3b).

Additionally, we evaluate $(\boldsymbol{\theta}, \mathbf{s}, \mathbf{v})$ for a set of 15 poses from the alphabets of American Sign Language (see Figure 2). A 15 dimensional vector, $\mathbf{d}$, is calculated for each pose exemplar, wherein each element is the sum of all pairwise Euclidean distances between $\mathbf{v}$ of a pose in the basis

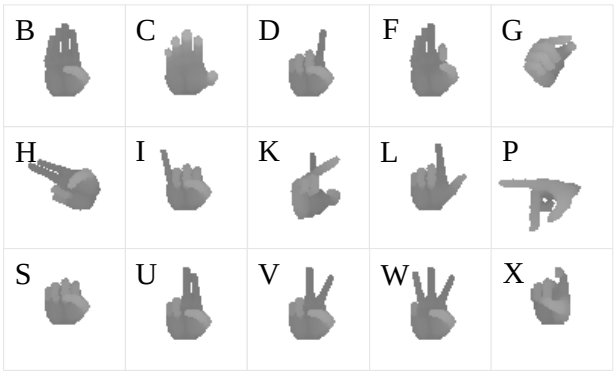


Figure 2: Illustration of 15 hand models used as basis adopted from American Sign Language.

and $\mathbf{v}$ of a pose exemplar. This vector serves as metadata for pose exemplars, akin to a feature vector for users in a recommender system.

3.3. Shape descriptor distance

We associate a local shape descriptor, $\mathbf{c}$, to each pose exemplar. Nearest neighbor retrieval at runtime, proceeds by first determining the shape descriptor of the arriving point cloud, calculating its shape descriptor distance of all pose exemplars, and then selecting the nearest neighbors less than a threshold. The computation of the shape descriptor distance between two depth maps is described next.

We use the FAST feature point detectors on a depth map to identify corner points [25]. For each detected FAST feature point, a BRIEF descriptor [6] is computed, which encodes information about surrounding regions. Details of FAST and BRIEF computation are skipped for brevity. Correspondences are established between FAST feature points of two depth maps by iteratively (1) finding the pair with minimum Hamming distance (bitwise XOR operation) between their corresponding BRIEF descriptors, and (2) removing this matched pair for evaluating subsequent correspondences. The shape descriptor distance is then the average Hamming distance between BRIEF descriptors of all matched pairs of FAST feature points. Note that this distance varies with the hand's orientation, and hence outputs similarly oriented hand poses from the library as nearest neighbors. This feature is desirable in our approach as the in-plane rotation angles can then be robustly estimated using these nearest neighbors in the JMFC algorithm. Also, the descriptors for all pose exemplars are pre-computed to reduce computational overhead and only the descriptor for the input depth map is evaluated at runtime for nearest neighbor computation.

We get a set of 1,030 pose exemplars with labels $\mathbf{r} = (\boldsymbol{\theta}, \mathbf{s}, \mathbf{v}, \mathbf{d}, \mathbf{c})$ after the above pre-processing steps. Next we discuss the steps of our solution at runtime.

4. Joint matrix factorization and completion

The pipeline of our approach is demonstrated in Figure 1. The input depth is first processed to remove the background and only contains the depth pixels of the hand. The global parameters, $\boldsymbol{\phi}$ are directly estimated from this processed depth map. Next, the local shape descriptor of this depth map is evaluated and the nearest neighbors are retrieved from the labeled database using the shape descriptor distance. These neighbors serve as seed postures to the JMFC model and the joint angle parameters, $\boldsymbol{\theta}$, are estimated, followed by some final post-processing to output the tracked hand skeleton.

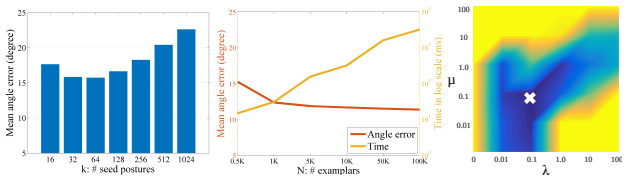
4.1. Model initialization

Background removal and estimation of $\boldsymbol{\phi}$: We use a simple heuristic to estimate the global translation parameters, $\boldsymbol{\phi}$. The depth map is pruned to exclude the background by only including points within the distance range of $(15, 50)$ cm to the depth camera, under the assumption that the hand lies in this region of interest. We determine the points corresponding to the hand in the depth map by considering the pixels enclosed in the longest continuous contour [29]. Extraneous noise in the detected blob is mitigated by using a median filter [13]. The translation parameters $\boldsymbol{\phi}$, are then set equal to the centroid of the remaining points in the depth map. Our experimental results suggest that this heuristic is fast and works well in practice. We propose to develop more sophisticated algorithms to estimate the translation parameters in future work.

Nearest neighbor retrieval and distance matrix: The k nearest neighbors [5] to depth map are calculated at each instant of time using the shape descriptor distance described in the previous section. The choice of parameter k is critical to the JMFC model. A small k compromises the robustness of the $\boldsymbol{\theta}$ estimation, whereas too large a k increases computational complexity making the model infeasible for real-time applications. Hence, we determine the $\hat{k}$ nearest neighbors below a threshold for the shape descriptor distance and set k equal to:

$$k = min(max(32, \hat{k}), 64); \tag{1}$$

This is because k between $[32, 64]$ ensures fast and robust parameter estimation (see Figure 3a). The distance threshold for the shape descriptor distance is set at 15 for all our experiments. Next, we impute two matrices $\mathbf{P_1}$ and $\mathbf{D_1}$ of dimensions $k \times n$ and $k \times m$ respectively, with the known joint angles, $\boldsymbol{\theta}$ ($n = 18$) and Euclidean distance vector, $\mathbf{d}$ ($m = 15$) for the k indexed neighbors in the preprocessed database. We also calculate the 15-dimensional distance vector, $\mathbf{d_2}$, as the sum of all pairwise Euclidean distances



(a) (b) (c)

Figure 3: (a) Choice of nearest neighbor, k. Joint angle error is minimum for $32 < k < 64$. (b) Choice of number of exemplars, N. $N \approx 1000$ optimally trades off between accuracy and computational time. (c) Choice of regularization parameters, μ, λ. Joint angle error color coded with blue denoting low error and yellow denoting high error. Best choice is $\mu = 0.1, \lambda = 0.1$ indicated by $\times$.

between $\mathbf{v}$ of each pose in the basis and points on the refined depth map. Our algorithm for estimating the joint angle parameters, $\mathbf{p_2}$, using $\mathbf{P_1}, \mathbf{D_1}, \mathbf{d_2}$ independently for each frame is discussed next.

4.2. The JMFC Model

As discussed previously, we use a joint matrix factorization and completion (JMFC) approach to estimate the unknown joint angles for a given depth map. Our rationale for using the JMFC model in analogy to a recommender system described in parenthesis is as follows: We have a matrix $\mathbf{P_1}$ with joint angles (known ratings) for a set of similar poses to the input depth (like-minded users to a new user). Additionally, matrix $\mathbf{D_1}$ contains auxiliary information about nearest neighbor poses relative to a basis (metadata about like-minded users) and vector $\mathbf{d_2}$ which contains the same auxiliary information about the new pose (metadata about new user) whose parameters $\mathbf{p_2}$ (unknown personalized ratings) are to be estimated. Our task is then to uncover the latent factors, $\mathbf{a_2}$ governing the parameters, $\mathbf{p_2}$ by determining the latent factors for (1) nearest neighbor poses, $\mathbf{A_1}$ (2) known joint angles, $\mathbf{C}$ and (3) known distances to basis models, $\mathbf{B}$. Mathematically, we find a factorization of matrices $\mathbf{P_1}, \mathbf{D_1}$ and vector $\mathbf{d_2}$ in terms of the latent factors $\mathbf{A_1}, \mathbf{a_2}, \mathbf{B}, \mathbf{C}$, and use these information to impute the unknown vector, $\mathbf{p_2}$. In other words, we simply find low rank approximations of known matrices in order to estimate the unknown pose parameters. Using the above intuition, our JMFC model is succinctly expressed as:

$$\underset{\mathbf{A_1},\mathbf{a_2},\mathbf{B},\mathbf{C}}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} \mathbf{D_1} \\ \mathbf{d_2} \end{bmatrix} - \begin{bmatrix} \mathbf{A_1} \\ \mathbf{a_2} \end{bmatrix} \mathbf{B} \right\|_F^2 + \frac{\mu}{2} \left\| \mathbf{P_1} - \mathbf{A_1 C} \right\|_F^2 . \tag{2}$$

where $\mathbf{B}$ and $\mathbf{C}$ are r-dimensional latent factors for the distances ($\mathbf{D}$) and joint angle parameters ($\boldsymbol{\theta}$), respectively; $\mathbf{A_1}$ and $\mathbf{a_2}$ are the r-dimensional latent factors for the k-nearest neighbors and input depth map respectively, and μ is regularization parameter which trades off the losses due to matrix factorization and accuracy of matrix completion. $\mathbf{P_1}$ decomposes as a product of latent factors $\mathbf{A_1}$ and $\mathbf{C}$, ($\mathbf{P_1} \approx \mathbf{A_1 C}$), $\mathbf{D_1}$ decomposes as a product of latent factors $\mathbf{A_1}$ and $\mathbf{B}$, ($\mathbf{D_1} \approx \mathbf{A_1 B}$), whereas the row $\mathbf{d_2}$ decomposes as $\mathbf{a_2 B}$ (see Figure 1f). To prevent overfitting, we add a regularization term, λ to the Frobenius norms of $\mathbf{A_1}, \mathbf{a_2}, \mathbf{B}$ and $\mathbf{C}$ which gives us the following minimization problem:

$$\underset{\mathbf{A_1},\mathbf{a_2},\mathbf{B},\mathbf{C}}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} \mathbf{D_1} \\ \mathbf{d_2} \end{bmatrix} - \begin{bmatrix} \mathbf{A_1} \\ \mathbf{a_2} \end{bmatrix} \mathbf{B} \right\|_F^2 + \frac{\mu}{2} \left\| \mathbf{P_1} - \mathbf{A_1 C} \right\|_F^2 + \frac{\lambda}{2} \left(\|\mathbf{A_1}\|_F^2 + \|\mathbf{a_2}\|_F^2 + \|\mathbf{B}\|_F^2 + \|\mathbf{C}\|_F^2 \right) . \tag{3}$$

We use the Alternative Least Squares (ALS) [4] to solve the above minimization problem, and it is summarized in Algorithm 1. Additional details about the objective function and the derivation of the algorithm are discussed in the supplementary material.

Algorithm 1: The JMFC algorithm

Input: $\mathbf{D}_1, \mathbf{d}_2, \mathbf{P}_1, \mu, \lambda$
Initialize: $\mathbf{A}_1, \mathbf{a}_2, \mathbf{B}, \mathbf{C}$
while *stopping criterion not met* **do**
  $\mathbf{A}_1 \leftarrow \left(\mathbf{D}_1\mathbf{B}^T + \mu\mathbf{P}_1\mathbf{C}^T\right)\left(\mathbf{B}\mathbf{B}^T + \mu\mathbf{C}\mathbf{C}^T + \lambda\mathbf{I}\right)^{-1}$
  $\mathbf{a}_2 \leftarrow \left(\mathbf{d}_2\mathbf{B}^T\right)\left(\mathbf{B}\mathbf{B}^T + \lambda\mathbf{I}\right)^{-1}$
  $\mathbf{B} \leftarrow \left(\mathbf{A}_1^T\mathbf{A}_1 + \mathbf{a}_2^T\mathbf{a}_2 + \lambda\mathbf{I}\right)^{-1}\left(\mathbf{A}_1^T\mathbf{D}_1 + \mathbf{a}_2^T\mathbf{d}_2\right)$
  $\mathbf{C} \leftarrow \left(\mu\mathbf{A}_1^T\mathbf{A}_1 + \lambda\mathbf{I}\right)^{-1}\left(\mu\mathbf{A}_1^T\mathbf{P}_1\right)$
end
$\mathbf{p}_2 \leftarrow \mathbf{a}_2\mathbf{C}$

The parameters λ and μ are empirically set to 0.1 and 0.1, respectively (see Figure 3c). The rank r of latent factors is set to 5 as it optimally trades off between accuracy and efficiency. The ALS procedure in Algorithm 1 repeats until the difference between output values of equation 2 for subsequent iterations is less than 10^{-6} or the number of iterations exceed 600. As a final step, the pose parameters $\mathbf{p_2}$ are estimated as $\mathbf{p_2} \approx \mathbf{a_2 C}$ and further refined by imposing the pose constraints mentioned in Section 3.1. This ensures that the final solutions comply with kinematically feasible hand configurations.

5. Experiments

In this section, we evaluate our approach for synthetic hand poses as viewed from a depth camera and real depth data. We perform quantitative analysis on a synthetic

dataset of hand poses generated by uniformly sampling in the constrained hand configuration space. This ensures adequate coverage, and hence an unbiased evaluation of our approach. Further, we perform the same quantitative analysis using realistic hand pose data captured from a commercial depth camera. The prime difference between real and synthetic data is the presence of noise in real depth streams. We first describe the datasets and set baselines before proceeding to the performance evaluation. All our experiments are performed on Intel Xeon E3-1240 CPU with 16GBs RAM.

5.1. Datasets

We generate a synthetic dataset of 1,000 randomized hand postures following the procedure in [26] as follows. The 18 joint angle parameters and 3 global translation parameters are uniformly sampled in the constrained hand configuration space to generate a synthetic hand configuration, and the depth map of this pose is rendered within the view frustum. All constraints for this configuration space simulating realistic hand poses are listed in the supplementary material. Consequently, we get varied poses with corresponding ground truth. Note that we can use this approach to evaluate performance because our algorithm does not depend on temporal information and re-initializes at every frame.

We capture depth streams using the SoftKinetic's DepthSense DS325 and use this information for evaluating our algorithm on real datasets. Four sequences are captured, each from a different person, and each sequence contains 300 frames ($\approx$ 10 seconds) of hand movement. The ground truth is first roughly initialized using FORTH [21] with 256 particles and 75 generations, followed by manual refinement. Even with the large number of particles and generations, FORTH contains subtle errors in the hand pose which we manually remove.

Furthermore, we evaluate ours against two state-of-the-art approaches [21, 28] on the large and challenging dataset released with [28] in order to demonstrate that our method is applicable in a general setting. The dataset consists of 76,500 depth images captured from 9 subjects, using a Intel's Creative Senz3D camera compatible with DepthSense camera resolution. The depth maps comprise of 17 hand gestures under large viewpoint changes and span diverse finger articulations and hand configurations.

5.2. Evaluation Metrics and Baselines

Metric Four standard metrics are used for our quantitative evaluation: (1) *individual joint angle error* averaged over all frames, (2) *individual joint distance error* averaged over all frames, (3) *proportion of correct frames* as a function of maximum allowed joint angle error, and (4) *proportion of correct frames* as a function of maximum allowed joint distance error described in [26, 31]. Metrics 1 and 2 indicate the estimation errors for individual joints whereas metrics 3 and 4 are indicative of overall robustness of an algorithm.

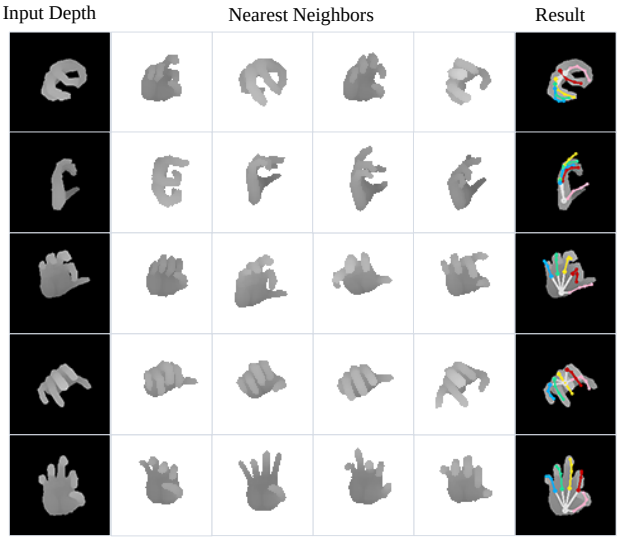


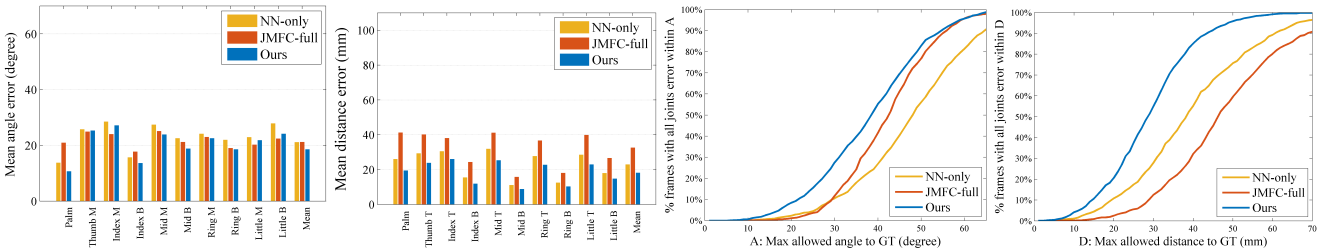
Figure 4: Qualitative analysis on the synthetic dataset. Left: randomly generated input poses. Middle: selected nearest neighbors (including outliers) from our pose exemplars. Right: the estimated hand pose.

Baselines We demonstrate the efficacy of our overall algorithm by comparing our method to the following baselines: (a) *NN-only* wherein we estimate pose parameters using a single nearest neighbor among the pose exemplars and (b) *JMFC-full* wherein all 1,030 pose exemplars are used for pose estimation, *i.e.*, nearest neighbors are not retrieved. We compare our algorithm to real-time implementation of FORTH on the realistic datasets by setting the parameters equal to 64 particles and 25 generations.

5.3. Experiments on Synthetic Dataset

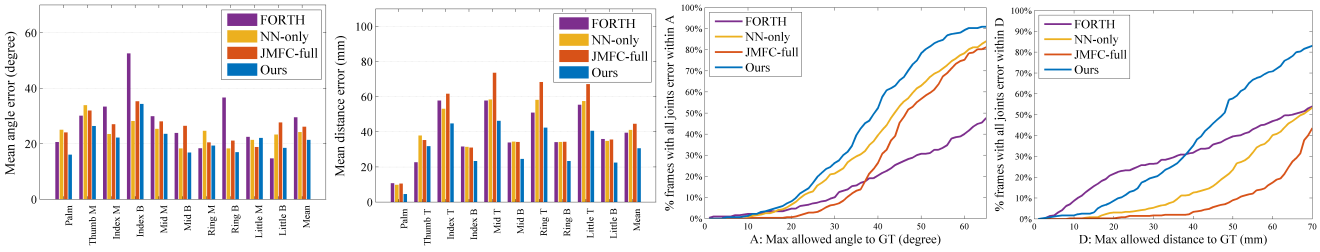
Quantitative Analysis We evaluated our approach on the generated synthetic poses. Figure 5 shows the quantitative evaluation of our algorithm in terms of the accuracy metrics, relative to the two baselines.

Figure 5a and 5b show the average error of estimated joint angles and distances relative to the ground truth. Our algorithm performs better than the two baselines with respect to both metrics. In Figure 5a we see that the errors in joint angles for JMFC-full are generally less than NN-only, except for the palm angle, meaning that the joint angles are robustly estimated by the JMFC model even in the presence of extraneous poses not similar to the input depth map. However, the high error in palm angle for JMFC-full makes the estimated pose very different from the ground truth. This error in JMFC-full propagates to other joints



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

Figure 5: Quantitative analysis on the synthetic dataset with respect to four metrics, relative to baselines (T: tip, M: mid, and B:base). (a) The average joint angle error in degrees. (b) The average joint distance error in millimeters. (c) and (d) show the proportion of depth maps (y-axis) with joint angle and distance error less than a threshold (x-axis).



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

Figure 6: Quantitative analysis on the realistic dataset with respect to four metrics, relative to baselines (T: tip, M: mid, and B:base). (a) The average joint angle error in degrees. (b) The average joint distance error in millimeters. (c) and (d) show the proportion of depth maps (y-axis) with joint angle and distance error less than a threshold (x-axis).

leading to large distance errors relative to NN-only as seen in Figure 5b. Figure 5c and 5d show that our algorithm performs better than NN-only and JMFC-full at all thresholds for maximum allowed joint angle and distance error. The proportion of correctly identified frames is about 90 percent when the threshold for the joint distance error is set to 40 mm as seen in Figure 5d. The comparative result can be found in [26] (figure 9c). Although we do not have access to their datasets, this qualitative comparison to their state-of-the-art method under the same experimental settings is very promising. Also unlike their approach, we do this without considering temporal information and without a GPU.

Qualitative Analysis We perform a qualitative analysis of our approach in Figure 4. The central sub-figures indicate the nearest neighbors retrieved from the pose library. We observe that even though some nearest neighbors share very little similarity to the input depth map, the final solution is robustly estimated. This robustness against outliers is attributed to the vector $\mathbf{d_2}$ (the vector of distances to basis models) in the JMFC model, which implicitly mitigates the effect of faulty nearest neighbors. Intuitively, the incorrect pose parameter values of these faulty neighbors are weighed less in the collaborative assignment of pose parameters to the unknown pose.

5.4. Experiments on Realistic Dataset

Quantitative Analysis We evaluate our approach on the generated realistic dataset affected by noise with respect to three baselines, NN-only, JMFC-full and FORTH[1].

Figure 6a and 6b show the average error of estimated joint angles and distances relative to the manually refined ground truth over all four sequences. We observe that overall our method is superior to all baselines with respect to all four error metrics. Unlike FORTH, our model does not need any temporal information, and hence, avoids errors accumulating over time. It is also interesting to note that noise in real datasets confounds nearest neighbor estimation leading to poorer performance than synthetic datasets. One solution to reduce the effect of noise is to use training for accurately generating pose hypothesis as done in [26] instead of using nearest neighbors, a possible direction for future work.

We observe that the performance of our algorithm to estimate joint angles on realistic dataset (Figure 6c) is very similar to the synthetic dataset. However, the performance as measured by error metric (d) deteriorates relative to synthetic dataset (Figure 6d). This hints at a compounded effect of poor nearest neighbor estimation and incorrect estima-

[1]The algorithm in [21] is reimplemented using our depth camera.

Figure 7: Qualitative comparison of our method with 3 baselines: FORTH, NN-only, JMFC-full in that order.

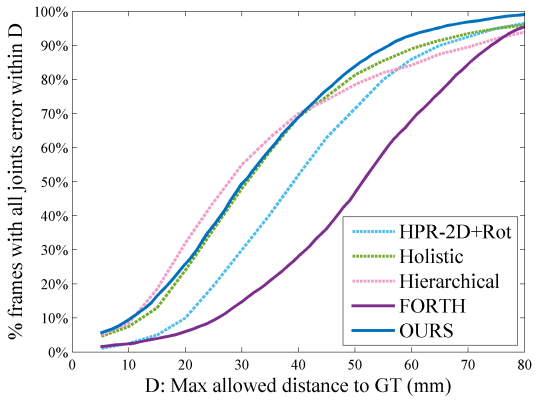


Figure 8: Quantitative comparison of our method with [21, 28] on a public dataset released with [28] with respect to proportion of depth maps (y-axis) with joint distance error less than a threshold (x-axis).

tion of global translation parameters. The latter problem, however, is easily solvable by replacing our heuristic based method by methods implemented in [33, 26] for accurate region of interest detection. However, the thrust of our contribution is the JMFC model for joint angle estimation which is effectively validated.

Qualitative Analysis Figure 7 qualitatively evaluates our approach against the baselines. All depth maps are centered for effective visualization. The top column shows the input depth map and each row corresponds a baseline method. We observe that our approach is robust to the various types of hand configurations under occlusion.

The average frame rate of our complete algorithm for hand pose estimation on the realistic datasets is $\approx$ 29Hz, and hence applicable in a real-time environment. In comparison, our implementation of FORTH with NVIDIA Quadro K4000 GPU resulted in an average frame rate of 16Hz. Additionally, we do not require temporal information as our algorithm proceeds on a per frame basis.

Quantitative Analysis on Public Dataset We compare our algorithm on the dataset of [28] with FORTH and the *Holistic*, *Hierarchical* and *HPR-2D+Rot* regression methods proposed in [28]. We indirectly compare our method with [30] as Hierarchical pose regression [28] has been shown to be better than [30] in [28] and with [35] which is similar in spirit to HPR-2D+Rot [28]. Figure 8 displays the proportion of depth maps (y-axis) with joint distance error less than a threshold (x-axis) for the 5 methods[2]. We see that our approach achieves better accuracy than FORTH and comparable performance to *Hierarchical* pose regression method of [28]. Our method has the highest fraction of frames with maximum allowed distance to ground truth in the $[0, 15]$ mm and $[40, 80]$ mm domain, validating that our approach is overall more robust to finger articulations and applicable to hand pose estimation in a general setting.

6. Conclusion

In this paper we present a novel approach for the hand pose estimation problem based on a joint matrix factorization and completion model. We present strong evidence of the applicability of our approach for hand tracking in a real-time environment. Although we demonstrate the efficacy of our approach for estimating joint angle parameters of the human hand, the overall idea is also applicable to the human pose estimation problem. More generally, our approach conclusively validates that advances in collaborative filtering approaches for recommender systems can be effectively synergized with pose estimation and tracking problems. This opens up several avenues for future work. One promising direction is the use of nuclear norm regularization instead of the Frobenius norm in the JMFC objective function to get low rank factors. We also wish to explore techniques for determining the best basis and effectively integrating RGB information in our future work. Overall, we believe our JMFC model based approach for hand pose estimation opens up new avenues for real-time solutions in computer vision.

Acknowledgements This work was partially supported by the NSF Award No.1235232 from CMMI and 1329979 from CPS, as well as the Donald W. Feddersen Chaired Professorship from Purdue School of Mechanical Engineering. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

[2]Performance of *Holistic*, *Hierarchical* and *HPR-2D+Rot* methods are estimated from figure 5a in [28] which displays the same error metric.

References

[1] E. Acar, T. G. Kolda, and D. M. Dunlavy. All-at-once optimization for coupled matrix and tensor factorizations. In *Mining and Learning with Graphs*, August 2011. 2

[2] J. R. Anderson and R. Milson. Human memory: An adaptive perspective. *Psychological Review*, 96(4):703, 1989. 1

[3] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: ordering points to identify the clustering structure. *ACM Conference on Management of data*, pages 49–60, 1999. 3

[4] R. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *International Conference on Data Mining*, pages 43–52, Oct 2007. 5

[5] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, Sept. 1975. 4

[6] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *ECCV*, pages 778–792. 2010. 4

[7] J. H. Choi and S. Vishwanathan. Dfacto: Distributed factorization of tensors. In *NIPS*, pages 1296–1304, 2014. 2

[8] W. S. Cleveland and S. J. Devlin. Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting. *Journal of the American Statistical Association*, 83(403):596–610, 1988. 2

[9] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108:52–73, 2007. 1, 2

[10] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *ACM SIGKDD*, pages 226–231, 1996. 3

[11] T. M. Greiner. Hand anthropometry of US army personnel. Technical report, DTIC Document, 1991. 3

[12] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *ACM SIGIR*, pages 230–237, 1999. 1

[13] T. Huang, G. Yang, and G. Tang. A fast two-dimensional median filtering algorithm. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 27(1):13–18, Feb 1979. 4

[14] S. Katz, A. Tal, and R. Basri. Direct visibility of point sets. *ACM Trans. Graph.*, 26(3), July 2007. 3

[15] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *ECCV*, pages 852–863. 2012. 2

[16] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun. Real time hand pose estimation using depth sensors. In *Consumer Depth Cameras for Computer Vision*, pages 119–137. Springer, 2013. 2

[17] J. Lee and T. Kunii. Model-based analysis of hand posture. *Computer Graphics and Applications*, 15(5):77–86, Sep 1995. 3

[18] J. Lin, Y. Wu, and T. S. Huang. Modeling the constraints of human hand motion. In *Proceedings of the Workshop on Human Motion*, pages 121–, Washington, DC, USA, 2000. 3

[19] J. McAuley and J. Leskovec. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *ACM RecSys*, pages 165–172, 2013. 1

[20] S. Melax, L. Keselman, and S. Orsten. Dynamics based 3d skeletal hand tracking. In *Graphics Interface*, pages 63–70, 2013. 2

[21] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC*, pages 1–11, 2011. 2, 6, 7, 8

[22] D. H. Park, H. K. Kim, I. Y. Choi, and J. K. Kim. A literature review and classification of recommender systems research. *Expert Systems with Applications*, 39(11):10059 – 10072, 2012. 1

[23] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl. Getting to know you: Learning new user preferences in recommender systems. In *International Conference on Intelligent User Interfaces*, pages 127–134, 2002. 1

[24] A. M. Rashid, G. Karypis, and J. Riedl. Learning preferences of new users in recommender systems: An information theoretic approach. *SIGKDD Explor. Newsl.*, 10(2):90–100, Dec. 2008. 1

[25] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *ICCV*, pages 1508–1515. IEEE Computer Society, 2005. 4

[26] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. K. C. R. I. Leichter, A. V. Y. Wei, D. F. P. K. E. Krupka, A. Fitzgibbon, and S. Izadi. Accurate, robust, and flexible real-time hand tracking. In *SIGCHI*, 2015. 2, 6, 7, 8

[27] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013. 1, 2

[28] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *In CVPR*, pages 824–832, 2015. 2, 6, 8

[29] S. Suzuki and K. be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32 – 46, 1985. 4

[30] D. Tang, H. J. Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. In *CVPR*, pages 3786–3793, 2014. 2, 8

[31] D. Tang, T. H. Yu, and T. K. Kim. Real-time articualted hand pose estimation using semi-supervised transductive regression forests. In *ICCV*, pages 3224–3231, 2013. 2, 6

[32] A. Thayananthan, R. Navaratnam, B. Stenger, P. Torr, and R. Cipolla. Multivariate Relevance Vector Machines for Tracking. In *ECCV*, pages 124–138. 2006. 2

[33] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Trans. Graph.*, 33(5):169:1–169:10, Sept. 2014. 2, 8

[34] R. Wang, S. Paris, and J. Popović. 6d hands: Markerless hand-tracking for computer aided design. In *UIST*, pages 549–558, 2011. 2

[35] C. Xu and L. Cheng. Efficient hand pose estimation from a single depth image. In *ICCV*, pages 3456–3462, 2013. 2, 8