Online Optical Marker-based Hand Tracking with Deep Labels

SHANGCHEN HAN, Facebook Reality Labs
BEIBEI LIU, Facebook Reality Labs
ROBERT WANG, Facebook Reality Labs
YUTING YE, Facebook Reality Labs
CHRISTOPHER D. TWIGG, Facebook Reality Labs
KENRICK KIN, Facebook Reality Labs

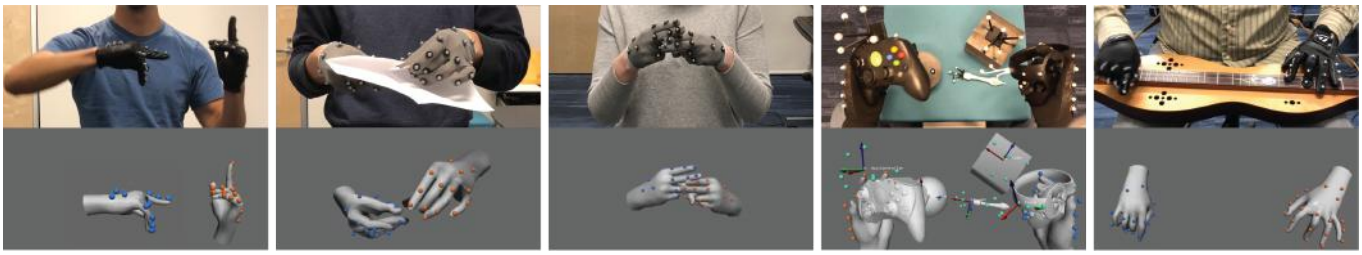


Fig. 1. **Snapshots of our online marker-based hand tracking system on sequences with two-handed and hand-object interactions.** We demonstrate a novel marker-labeling and tracking system that enables fully-automatic, real-time estimation of hand poses in challenging interaction scenarios with frequent occlusions. Markers labeled as left hand and right hand are rendered as orange and blue spheres respectively, while markers associated with predefined rigid bodies are rendered as green spheres.

Optical marker-based motion capture is the dominant way for obtaining high-fidelity human body animation for special effects, movies, and video games. However, motion capture has seen limited application to the human hand due to the difficulty of automatically identifying (or labeling) identical markers on self-similar fingers. We propose a technique that frames the labeling problem as a keypoint regression problem conducive to a solution using convolutional neural networks. We demonstrate robustness of our labeling solution to occlusion, ghost markers, hand shape, and even motions involving two hands or handheld objects. Our technique is equally applicable to sparse or dense marker sets and can run in real-time to support interaction prototyping with high-fidelity hand tracking and hand presence in virtual reality.

CCS Concepts: • **Computing methodologies** → **Modeling and simulation**; **Machine learning**;

Additional Key Words and Phrases: motion capture, hand tracking, marker labeling

Authors' addresses: Shangchen Han, Facebook Reality Labs, shangchen.han@oculus.com; Beibei Liu, Facebook Reality Labs, beibei.liu@oculus.com; Robert Wang, Facebook Reality Labs, rob.wang@oculus.com; Yuting Ye, Facebook Reality Labs, yuting.ye@oculus.com; Christopher D. Twigg, Facebook Reality Labs, chris.twigg@oculus.com; Kenrick Kin, Facebook Reality Labs, kenrick.kin@oculus.com.

0730-0301/2018/8-ART1 $15.00
https://doi.org/10.1145/3197517.3201399

ACM Reference Format:
Shangchen Han, Beibei Liu, Robert Wang, Yuting Ye, Christopher D. Twigg, and Kenrick Kin. 2018. Online Optical Marker-based Hand Tracking with Deep Labels. *ACM Trans. Graph.* 37, 4, Article 1 (August 2018), 10 pages. https://doi.org/10.1145/3197517.3201399

1 INTRODUCTION

Optical motion capture is the prevailing method for capturing high accuracy and high framerate motion. Applications across visual effects, gaming, VR/AR, usability research, and biomechanics are enabled by commercial systems that provide reliable sub-millimeter accurate tracking at up to 2000Hz. Notably, real-time full-body motion capture has enabled virtual production applications for games and movies [Antoniades 2016]. However, when it comes to detailed and dexterous finger motions, no commercial or academic software is able to produce comparable real-time tracking results, even in elaborate capturing environments.

The challenge of real-time capture lies in the identification or *labeling* of identical-appearing passive markers. While the labeling problem can be solved by active markers, their wired electronics or bulky sensors make them undesirable, and active systems are limited in the number of markers which can be tracked simultaneously. A common solution in full-body tracking for marker labeling is to start from a predefined T-pose or A-pose, then rely on high framerate tracking to propagate labels forward. This solution breaks down when tracking fails due to *occluded markers*, *ghost markers* from spurious reflections, or *dense marker sets* needed to capture subtle articulations. For body motion, tracking failures can be minimized by careful placement of cameras and marker layout design. Unfortunately, finger motions present additional challenges due to their high degree of articulation, self-similarity, and small scale. In

particular, frequent occlusions are inevitable even in common poses such as the fist.

Recent work has sought to reduce the marker density on a hand and optimize the marker layout, to disambiguate labels and minimize the impact on pose reconstruction [Alexanderson et al. 2017; Schröder et al. 2015; Wheatland et al. 2013]. However, methods using sparse marker sets rely on priors to hallucinate missing information about finger motion. They are especially susceptible to disruption from occluded or ghost markers. Although prior knowledge can regularize the results, they can lead to unresponsive animations or may not be accurate enough for capturing real-time manipulation tasks.

We contribute a technique that pushes the state-of-the-art of labeling and tracking for both sparse marker sets as well as dense marker sets that capture the full 26 degrees of freedom on a hand. We rely on a convolutional neural network (CNN) to predict labels given 3D marker positions detected by the motion capture system. Notably, we pose the problem as a keypoint estimation problem on 2D images, for which CNN's have proven to be particularly effective. To satisfy the need for high quality ground truth, we use synthetic data in a carefully designed data augmentation procedure for training (Section 3). At runtime, the output of the CNN is used to initialize labels whenever we need to reinitialize tracking of markers (Section 4.2). These online labels enable us to reconstruct hand poses in real-time (Section 4.3). Even though we train the network with synthetic data, we show it generalizes well to realistic data, including a variety of novel hand proportions and activities absent from the training set. Furthermore, we show our network is robust to occlusions and ghost markers that arise in sequences with object and environment interactions (see Figure 1). The flexibility of synthetic data also allows us to test different marker layouts. Our results show that the CNN labeling works just as well on a sparse marker set similar to Alexanderson and colleagues [2017]. While our marker labeling step is not user-specific, we can interactively calibrate a user's marker configuration at run-time to achieve the highest quality tracking (Section 4.5). We demonstrate the robustness of our real-time labeling and tracking system by capturing fascinating performances, such as object manipulation, hand-hand interaction, dancing, and playing musical instruments. Though not consumer-facing, our system can serve as a "time machine" to study the usability of AR/VR interaction protypes for future products with instantaneous feedback. In addition, our natural hand-hand and hand-object interaction data can be rendered as depth or RGB images to train deep learning models for solving markerless hand tracking. We publish our training dataset as well as our trained convolutional neural network to enable follow-up research. [1]

2 RELATED WORK

Labeling identical markers for motion capture remains an active problem in computer vision, computer graphics and robotics. Ringer and Lasenby [2002b] cluster pairwise distances between markers across frames to produce marker label hypotheses and propagate these hypotheses in a Bayesian multiple hypothesis tracking framework. Meyer et al.[2014] initialize marker labels by requiring users to start in a T-pose, so that arms and legs may be extracted from detected principal axes. Schubert and colleagues [2015] relax the requirement of a T-pose and match against a large database of previously observed poses instead. Schubert *et al.* [2016] further extend this work to support automatic calibration of the subject's skeleton.

To capture the hand's full articulation, several markers are needed on each finger and at least three markers are needed on the back of the hand [Kitagawa and Windsor 2008]. However, such a dense marker set is difficult to disambiguate, especially in a large space or in conjunction with full-body tracking [Wheatland et al. 2015]. Hence, recent work has explored the use of sparse marker sets [Alexanderson et al. 2016; Schröder et al. 2015; Wheatland et al. 2013], which attempt to capture as many as 20 degrees of freedom of the five fingers with as few as three markers. Alexanderson et al. [2016] in particular extend the multiple hypothesis tracking approach of Ringer and colleagues [2002a] to handle sparse marker sets. [Maycock et al. 2015] and [Aristidou and Lasenby 2010] further leverage inverse kinematics for online tracking and gap filling, respectively. Maycock et al. [2015] run the Hungarian method at each frame for labeling, which delays recovery from wrongly labeled markers resulting from the previous frame or occlusions. Its follow-up work [Schröder et al. 2017] utilizes dense points from 3*D* scanner for hand model fitting and marker calibration.

Our method pushes the state-of-the-art in labeling sparse marker sets, but also tackles the ambiguity problem of dense marker sets directly. Although sparse marker sets can be used to generate compelling hand-over animations [Kang 2012], dense marker sets capture additional nuances in finger pose and motion that can be used to drive real-time dexterous manipulation of physical objects and interaction prototyping tasks.

There is a rich body of work on markerless tracking of hand motion from either depth [Oikonomidis et al. 2012; Taylor et al. 2016; Tkach et al. 2016; Tompson et al. 2014] or RGB [Simon et al. 2017] cameras. Although high-end 3D scanners or depth cameras can generate high-fidelity geometry at high framerate such as [Taylor et al. 2017], they are more expensive and harder to set up and use compared to marker-based systems. Additionally, such markerless systems cannot yet achieve sub-millimeter accuracy and track the complex hand motions that optical motion capture can, which are required for our real-time dexterous hand interactions for complex tasks and subtle motions.

Convolutional neural networks have demonstrated state-of-the-art results for image classification [Krizhevsky et al. 2012], object detection [Ren et al. 2015], and semantic segmentation [He et al. 2017]. We transform the marker labeling into a keypoint estimation problem to leverage recent progress in this domain.

3 DEEP LABELING

In our proposed system, each glove is affixed with 19 markers (Figure 2). The geometric structure of the markers encode their labels implicitly through the kinematic structure of the hand. Alexanderson et al. [2016] decode this information with a Gaussian Mixture Model fitted from the spatial distribution of markers in a hand-local coordinate system. We propose transforming the 3D marker labeling (classification) problem into an image keypoint regression problem

[1] https://github.com/Beibei88/Mocap_SIG18_Data

Fig. 2. **Dense marker set layouts.** We attach 19 adhesive reflective markers on each off-the-shelf golf golve. Three markers are placed at the back of the glove, and a marker is attached to the center of each finger bone segment. We build five pairs of gloves of different sizes to capture the full range of adult hand sizes. These gloves are easy to put on and remove, which streamlines capture preparation.

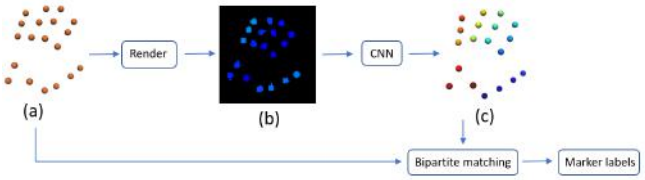


Fig. 3. **CNN deep labeling.** A list of 3D marker locations are fed into the labeling system (a). They are mean centered and rendered as a depth image (b). The CNN takes the depth image as input and outputs (in order) the 19 3D marker locations on the hand. Each marker is color coded by its entry in the marker set in (c). Bipartite matching is used to associate the input markers to their corresponding marker labels.

that can be efficiently solved with a convolutional neural network. The estimated keypoints are then corresponded with the actual 3D markers using a simple bipartite matching approach.

3.1 Labeling markers using a convolutional neural network

Deep neural networks have been widely applied to problems from image recognition [Krizhevsky et al. 2012] to automatic translation [Wu et al. 2016]. However, it is not obvious how to apply traditional architectures to the marker labeling problem. Regressing directly from a vector of 3D marker locations to labels is sensitive to the ordering of the markers in the vector. Despite recent work on sparse voxel representations, volumetric methods and 3D convolutions are still computationally expensive and memory-inefficient for real-time evaluation [Riegler et al. 2017]. A key insight is that we can capture the *spatial relationships* between the markers through 2D image projections, which are well handled by deep convolutional structures.

Therefore, to solve the labeling problem, we first render input $3D$ point cloud as a sparse 52×52 depth image (Figure 3) through an orthographic camera, whose principal axis passes through the center of point cloud. This camera is zoomed isotropically such that the projected points fit within a 10% margin of the image. Depth ("z") values are normalized between $[0.1, 1.0]$ and then rendered as pixel intensities. Critically, we preserve 3D information about the markers by splatting their relative depth, effectively creating a "sparse" depth image of the markers. We then feed this image into a conventional VGG-style neural network [Simonyan and Zisserman 2015], which consists of several layers of 3×3 convolutions, followed by a fully-connected layer, and finally output (in order) the 19 3D marker positions on the hand (see Table 1). During training, the principal axis of the orthographic camera is randomly picked. For run-time labeling, we generate 10 random principal axes and select the one with the largest spatial spread (that is, maximizes the eigenvalues of the covariance matrix of the $2D$ image positions).

To map these back to the original markers, we solve a bipartite matching problem. Given the output of the neural network $y_{1\ldots19}$, we match these to the original marker locations $x_{i\ldots n}$ with

$$\min_{\mathcal{M}} \sum_{j} \|y_j - x_{\mathcal{M}(j)}\|_2 \tag{1}$$

This can be posed as a minimum-cost flow problem where the edge weights represent distances, and solved using standard linear programming algorithms [Cormen et al. 2009].

3.2 Training data

The biggest bottleneck for supervised learning with a deep network is getting accurate, diverse and ample training data. To avoid tedious and error-prone manual labeling for dense marker sets, we use synthetic data instead. Thanks to the work of Tompson and colleagues [2014] on real-time hand pose recovery from depth input, we are able to generate large datasets with various valid hand poses. By referring to the marker set positions on the predefined hand model (Figure 2), a synthetic labeling dataset can be generated with diverse hand poses. In our framework, we use a Kinect camera to collect different hand gestures from five subjects exercising all the degrees of freedom of a single hand, such as gestures from sign language. 170330 frames of hand poses are used for training.

Table 1. **Structure of our neural network.** Conv is short for spatial convolution, BN is the batch normalization layer, ReLU is the activation function $f(x) = max(0, x)$ and FC is a fully connected layer. The input of our network is a 52×52 depth image, and the output is a 19×3 matrix which contains the predictions of the 19 3D marker locations on the hand model.

Layer id	Type	Filter shape	Input size
1	Conv + BN + ReLU	$64 \times 3 \times 3$	$1 \times 52 \times 52$
2	Conv + BN + ReLU	$64 \times 3 \times 3$	$64 \times 50 \times 50$
3	Maxpool	2×2	$64 \times 48 \times 48$
4	Conv + BN + ReLU	$128 \times 3 \times 3$	$64 \times 24 \times 24$
5	Conv + BN + ReLU	$128 \times 3 \times 3$	$128 \times 22 \times 22$
6	Conv + BN + ReLU	$128 \times 3 \times 3$	$128 \times 20 \times 20$
7	Maxpool	2×2	$128 \times 18 \times 18$
8	Reshape	N/A	$128 \times 9 \times 9$
9	FC + ReLU	2048×10368	10368
10	FC	2048×57	2048
11	Reshape	N/A	57

3.3 Robustness

In practice, we need to be robust to artifacts of real-world motion capture. First, stray reflections in the capture space are common

and may result in the system identifying additional spurious "ghost" markers. Second, a marker on the glove may be occluded in all of the camera views, making it impossible to reconstruct its location. In addition, there is significant variability between human hand shapes and sizes which will affect the resulting marker layout.

To ensure robustness to each of these factors, we augment our synthetic training data with several sources of randomness. First, to ensure robustness to variations in hand shape, we add random 3D jitter to each marker location during training. In addition, at each training step we drop between 0 and 5 random markers from each training sample. Finally, we add between 0 and 2 random ghost markers. Markers are omitted or added at random to model occlusions and ghost markers respectively. The effect of this data augmentation on the robustness of our networks can be seen in Table 2.

4 REAL-TIME HAND TRACKING

4.1 Overview

We adopt an off-the-shelf 16-camera Optitrack motion capture system [Opt 2018] for marker tracking (see Figure 5). It is configured to provide a 120Hz stream of marker positions $P^t \in R^{3\times n}, t \in \{t_0, t_1 \ldots\}$ at each time stamp where n is the number of observed markers in the current frame. Markers are assigned unique IDs as long as they are continuously tracked; if a marker disappears briefly, it returns with a new ID.

We use a model-based tracking approach for reconstructing the hand poses from the marker labels. Our model consists of a kinematic hand skeleton H, a mesh model S, and rest pose marker locations m_i for $i = 1 \ldots n$. Our hand model H has 26 degrees of freedom (DOF) where the first 6 DOF represent the global transformation $\{\mathbf{R}, \mathbf{t}\}$ and there are 4 rotational DOF per digit. We use the standard linear blend skinning/enveloping method. Given a hand pose θ, a rest-pose position v_k, and a set of bone weights ω_{ik}, we can transform a vertex on the rest mesh into world space using:

$$LBS(\theta, m_k) = \sum_i \omega_{ik} * T_i(\theta) * \left(T_i^{\text{rest}}\right)^{-1} * v_k$$

Here, T_i^{rest} is the bone transform in the rest space and T_i can be computed from the hand pose θ. We parameterize the markers in the same way $m = \{\omega, v\}$ to transform a marker to the global space.

4.2 Online marker labeling

The first stage of our algorithm is to group the markers into clusters, where each cluster potentially represents a single hand. We use a simple agglomerative clustering approach, iteratively joining clusters provided (a) the cluster centers are no further than 200 mm apart and (b) the combined cluster contains no more than 22 markers. Resulting clusters containing between 14 and 21 markers are considered to be possible hands, and we randomly assign the left/right hand to a cluster. The resulting left/right hand clusters are then fed into the deep labeling pipeline (Section 3), and then into the inverse kinematics solver. We then evaluate certain quality metrics on the resulting reconstruction (Section 4.4); if the result is deemed successful, then we can proceed with tracking using the marker IDs directly from the mocap system without the clustering and deep labeling steps.

Note that if the left/right hand are misassigned during the clustering phase, then the reconstruction quality tests will fail (Section 4.4) and the algorithm will simply retry at the next frame; since the system runs at 120fps, a few missing frames due to hand misassignment at the initiation of tracking will be barely noticeable to users.

During tracking, some markers will be inevitably occluded and appear again in the scene with a different ID. When new markers appear, we run two re-labeling processes simultaneously. In the first, we simply extrapolate the expected location for each missing marker based on the previous frame's hand pose, and assign markers that are less than 15 mm from their expected location. In the second, we re-run the deep labeling pipeline and use the labels from the matching step. We run IK using both labelings and select the one which has the lowest error under our reconstruction metric.

4.3 Inverse Kinematics

Given the labeled markers, we obtain the hand pose θ using inverse kinematics (IK) by optimizing the following least squares energy:

$$E_{\text{IK}} = \sum_i ||LBS(\theta, m_i) - p_i||_2^2$$

where p_i is the corresponding labeled marker for the marker m_i. Because the data from the motion capture system is already very smooth and accurate, we did not find that any regularization to ensure motion smoothness was necessary. Levenberg-Marquardt solver is used here and is initialized by valid hand pose from the previous frame when available, or a canonical relaxed hand pose otherwise. Generally this iterative solver converges within five steps. We do not consider missing markers in IK step currently and filling them using CNN's prediction is a promising future direction.

4.4 Validation

During a step of online tracking, we may have multiple proposed labelings: (a) the tracked labels from the previous frame (if avavailable); (b) the previously tracked labels plus additional labels based on proximity (Section 4.2); and (c) labels generated by the deep labeling step (Section 3). For each proposed marker labeling, we run the inverse kinematics step to fit the markers. The resulting RMS error provides an estimate of how well the hand model fits the data. Poses that generate RMS error more than 8 mm are discarded. If multiple proposals pass the RMS test, we select the pose using the most labeled markers.

4.5 User Marker Calibration

In a given tracking session, we select the glove from our set of five pre-constructed gloves (small through large) which most closely matches the user's hand size. Each glove has a corresponding hand model H which has been artist-constructed. However, due to the variability in human hand shape and mismatch between our predefined marker set and the markers on the gloves, we usually can't generate the most accurate poses. The most objectionable artifact that we notice is that the thumb and index finger in the tracked hand do not touch, even when they are touching in reality. We can correct for this using a quick calibration process. During the calibration

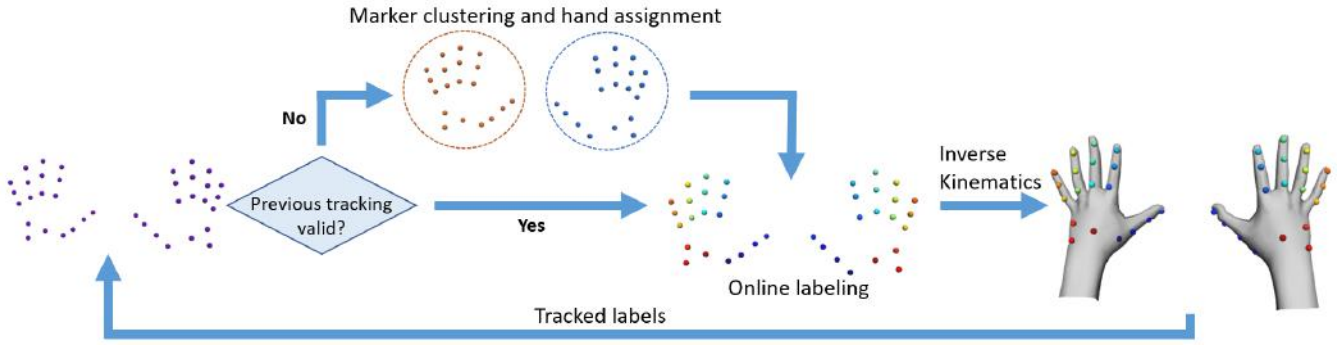


Fig. 4. **System overview.** The input to our system is a list of unlabeled 3D marker locations from the motion capture system. The tracker is initialized by clustering markers into groups and randomly assigning the left/right hands to clusters. The left/right hand markers are passed into the online marker labeling pipeline to obtain the label of each marker. Inverse kinematics is applied to reconstruct the hand poses from the labeled markers. The tracked labels and hand poses are fed into the next frame for online tracking if it passes our reconstruction metric.

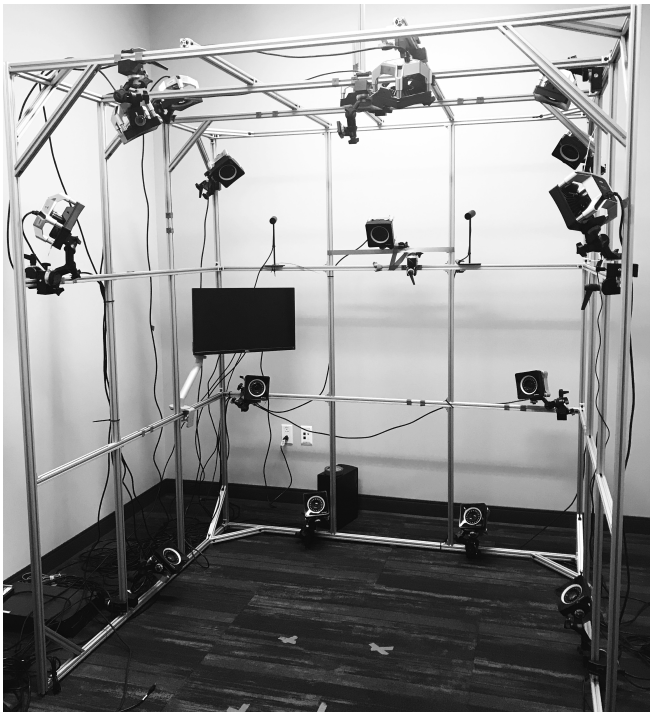


Fig. 5. **Camera setup.** We use 16 Optitrack cameras affixed to a rectangular grid to capture all of our sequences.

step, we capture a number of poses, including asking the user to touch their thumb to each of their other fingertips (Figure 6, left; see video for a sample session). We then solve an optimization problem which moves the markers to both reduce tracking error and ensure the fingers of the tracked hand touch as expected (Figure 6, right).

We formulate marker calibration as an optimization problem. The hand model is selected among five pre-defined hand models before capture and is fixed during optimization. We jointly solve for hand poses (θ), marker set (M) and contact points (C) with a set of pre-specified poses:

$$E(\theta, M, C) = E_{\text{data}}(\theta, M) + \lambda_1 E_{\text{prior}}(M) + \lambda_2 E_{\text{contact}}(\theta, C)$$

Data term. The data term measures how well the skinned marker matches the tracked marker locations.

$$E_{\text{data}} = \sum_{i,j} \|LBS(\theta_i, m_j) - p_{ij}\|_2^2$$

where j is the index of the markers on hand (i.e, $j \in [0, 18]$) and i is the index of the poses performed during calibration.

Marker location prior. We penalize deviation of the marker from its expected location relative to the hand. This helps regularize the problem. We split this term into two: one which measures movement along the surface, and one which measures movement normal to the surface. Movement normal to the surface is penalized more strongly because the marker radius is known to be fixed (3.2 mm).

$$E_{\text{prior}} = \sum_i (m_i - m_i')^T \left(\alpha_1 \mathbf{n}\mathbf{n}^T + \alpha_2(\mathbf{I} - \mathbf{n}\mathbf{n}^T)\right)(m_i - m_i')$$

where $\alpha_1 = 2.0$ and $\alpha_2 = 1.0$ in our setting.

Contact point error. We found it important for users to be able to see their virtual fingers touching when their physical fingers are touching. To enforce this in our calibration phase, we ask the user to touch their thumb to each of their fingertips, and then enforce this contact during the optimization using a least squares constraint.

$$E_{\text{contact}} = \sum_i \|LBS(\theta_i, c_i) - LBS(\theta_i, d_i)\|_2^2$$

Here, c_i represents the contact point on the thumb and d_i represents the contact point on the other finger. Note that we do not know *a priori* exactly where on each digit the contact occurs, so this location is also optimized. To ensure each contact point lies on the surface, it is parametrized using barycentric coordinates on the rest hand mesh and allowed to slide from a triangle to its neighbors during the course of the optimization.

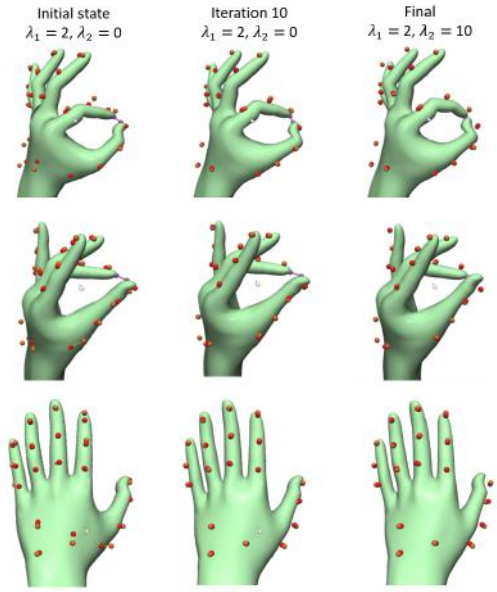


Fig. 6. **Marker calibration progress.** Three example poses are shown here: thumb touching index finger (top row), thumb touching middle finger (middle row), and spread pose (bottom row). During calibration, we jointly optimize all the hand poses, the marker locations on the rest hand model (red spheres), and the between-finger contact points (purple spheres). In the first 10 iterations, the optimizer adjust the locations of the markers on the hand to match the corresponding observed markers (orange spheres). Starting from iteration 11, the contact constraints are activated and the optimizer starts refining the contact points and their distance. The rightmost column shows the hand poses, marker positions, and contact points after the optimizer converges.

Implementation. We subdivide the original hand mesh once before optimization so that the contact points are able to slide more smoothly along the surface. All the error terms are in the sum of squares form and we use Levenberg-Marquardt method for optimization. After each iteration, we update the markers M and contact points C by looking for the closest point on the neighboring triangles of the mesh model. The progress of the calibration can be seen in Figure 6.

5 EVALUATION

Our CNN is trained using two NVIDIA Tesla M40 GPUs. We use stochastic gradient descent as the optimizer and train the network for 75 epochs. Each training iteration takes roughly 0.4 s using a batch size of 256. The learning rate starts at 0.6 and drops by a factor of 10 at the 50th epoch and the 70th epoch. Our runtime system was tested on an Intel Core i7 3.5GHz CPU with NVIDIA GTX 1080 Ti GPU running Windows 10. It takes 1 ms to run a forward pass of the CNN and 2 ms to run the inverse kinematics step. We only evaluate the CNN when tracking fails for at least one marker, and the entire real-time tracking system runs comfortably at 120Hz on average.

Robustness of data augmentation. We evaluate the network performance on a heldout dataset, and compare the different data augmentation schemes. The **Base** network is trained with exactly 19 markers. An occlusion-augmented network (**Base+O**) is trained by randomly omitting up to *five* markers from the input. A ghost-augmented network (**Base+G**) is instead trained by randomly introducing up to *two* additional markers. Finally, a combined network (**Base+O+G**) is trained with both occluded and ghost markers. We then apply the same data perturbations to the **Base** test dataset to generate **Base+O**, **Base+G**, and **Base+O+G** test datasets. As expected, data augmentation results in robust networks that generalize well to difficult scenarios, without sacrificing on the base case. Table 2 shows that while all networks are comparable on the **Base** test, the **Base+O+G** network performs best on all other more difficult tasks. We use this best-performing network in the rest of the paper.

Table 2. **Data augmentation schemes evaluated on synthetic test datasets**. We augment the Base network of 19 input markers with up to five occluded markers (Base+O) or up to two additional ghost markers (Base+G), or with both occluded and ghost markers (Base+O+G). Values reported here are labeling accuracy as the percentage of correctly predicted labels. While all networks are comparable on the Base test, the Base+O+G network outperforms the rest in the more challenging tests.

Train / *Test*	Base	Base+O	Base+G	Base+O+G
Base	97.76%	97.56%	**97.80%**	97.76%
Base+O	83.35%	92.87%	84.26%	**93.51%**
Base+G	92.31%	91.26%	94.87%	**95.30%**
Base+O+G	74.01%	81.88%	76.85%	**85.81%**

Evaluation on realistic test data. A more realistic test is to apply our labeling network to motion capture sessions of different people performing a wide range of tasks. For evaluation, we captured raw marker sequences of the following actions from a participant not in the training data: a single-hand-in-the-air motion without any interactions; two hands interacting with each other; a hand playing with a marker pen; and a hand playing with an Oculus Touch controller. Each sequence contains different numbers of occluded markers and ghost markers from the OptiTrack system. We first apply our method to produce initial labels, then ask a human labeler to manually correct for any mistakes to generate the ground truth labels. Table 3 shows a considerable number of occluded markers as the hand interacts with more complex objects.

We evaluate our network's performance on these test sequences against the ground truth. For comparison, we also show the same statistics of the synthetic test dataset **Base+O+G** in Table 4. For a simple sequence such as the single hand with few occlusion and ghost markers, our network makes accurate predictions for almost any pose. Performance drops on challenging hand-hand and hand-controller sequences, but is comparable to the augmented synthetic test data. The hand-pen sequences are particularly challenging since the subject is asked to point the pen tip at the markers on the glove, leading to confusion between the pen-tip marker and the hand markers that impacts the occlusion recall score. The network is

Table 3. **Test datasets.** #Frame is the total number of frames in a sequence. #Gap is the total number of gaps in the sequence, where each gap is a subsequence of one or more frames where at least one marker is missing. #AGL is the average number of (sequential) frames per gap. And #AMF is the average number of missing labels per frame, excluding frames with all markers present (i.e., #missing labels over #frames with missing labels).

Test set	#Frame	#Gap	#AGL	#AMF
Single hand	4785	44	2.11	1.35
Hand-hand	3312	331	4.01	3.30
Hand-pen	4511	481	3.39	2.31
Hand-controller	2645	382	2.65	1.62

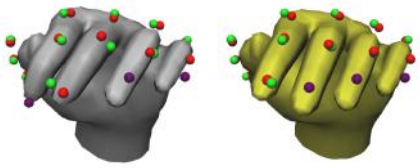


Fig. 7. **Ambiguity from missing markers.** On the right is a ground truth pose with the fingertip markers missing from the middle finger, the ring finger, and the pinky (marked in purple). On the left is the pose where the fingertip marker from the index finger has been incorrectly assigned to the middle finger. Although this pose has a higher RMS error, it has sufficiently low error that our system would consider it valid.

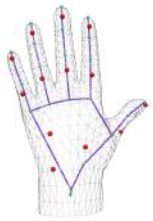
particularly robust to distractors, with high precision and recall numbers in realistic cases when the number of ghost markers is even larger than in the training set. However, it doesn't perform as well under a moderate number of occlusions, with the O recall score being particularly low. We hypothesize that this may be in part because occluded markers being selected randomly at train time (rather than using a visibility test), limiting its ability to handle real-world occlusions.

Another reason for the vulnerability to occlusion may be the overlapping motion range between fingers. When markers are missing, several plausible poses may explain the same observations under our RMS error check. For example, in Figure 7, the middle finger is missing a marker on the middle fingertip. This could be explained either by a missing marker on the middle fingertip, or by a missing marker on the index fingertip (and the index fingertip marker assigned to the middle finger). Incorporating the tracking history would help reduce such ambiguity. We could also train the neural network to output its confidence about a prediction, or even output a distribution over possible labeling [Bulat and Tzimiropoulos 2016; Shih et al. 2015].

Comparison of tracking result. In practice, we do not evaluate the network on every single frame, only those with failed tracking. A successful re-initialization of labels can then help recover tracking for subsequent frames. Therefore, when applied in real-time tracking, our system can accurately label all markers on over 90% of frames for all test sequences (Table 5), making it practical for realistic settings that are otherwise impossible without intense manual clean-up effort.

We compare our real-time tracking results with Alexanderson *et al.* [2017] against ground truth on the four captured test sequences (Table 5). Since Alexanderson *et al.* [2017] is designed for a sparse marker set requiring three markers on the back of the hand already labeled, it doesn't perform well on a dense marker set with a considerable amount of ghost markers or any of the three back markers is occluded. We also tried to compare with the commercial system Vicon [2018]. However, even with expert guidance, Vicon's calibrated skeleton was unable to label all the challenging frames from our test dataset. Nonetheless, it is clear that our system outperforms state-of-the-art real-time tracking methods on realistic hand motions.

Sparse marker set. One advantage of using synthetic data is that we can experiment with different marker layouts. To show our labeling pipeline also works for sparse marksets, we generate a training dataset using the sparse marker layout from Alexanderson *et al.* [2016], and follow our data augmentation procedure to train a network for labeling 13 markers (see right inset image) on a hand. When testing on the synthetic dataset and the same test sequences, we see a slight decline in performance (Table 6), but the overall accuracy is still quite satisfactory. This test demonstrates our network's capability to handle both dense and sparse marker sets.



User marker calibration. Our robust real-time tracking system allows for interactive fine-tuning of marker layout for a user's hand. Even though the labeling system doesn't require such knowledge, accurate marker positions result in more precise hand pose that permits finger contacts. Table 7 shows the performance comparison with and without user marker calibration. When not every hand marker is labeled, we use the predicted marker location from the hand model to assign labels to nearby markers. Without user-specific marker calibration, this process is prone to labeling ghost markers as hand markers, which causes a substantial drop in marker labeling performance on the hand-pen sequence when the pen tip marker is near the hand markers.

Reduced number of cameras. We use a 16-camera capture rig for real-time tracking to achieve high quality motions under heavy occlusion and noise. However, a greatly reduced setup is already sufficient for capturing interesting hand motions. We selectively turn off more than half of the cameras for tracking a single-hand-in-the-air sequence. With increased occlusion, the system can still successfully track almost all poses. In the 2875 frames captured, 95.55% of frames pass our preset IK error threshold, among which 92.32% have the correct labeling.

6 APPLICATIONS

We demonstrate the robustness and flexibility of our real-time hand tracking system on a wide range of activities, including dexterous manipulation and object interaction, sign language, dance, musical performance, as well as using the hands as an input device for VR applications (Figure 8). Our system is easy to set up and use for any user. It takes less than two minutes for the user to put on the gloves

Table 4. **Network performance on test datasets.** We compare network inference results on a synthetic dataset and four captured sequences, from a participant not in the training set. We report **Accuracy** as the percentage of correctly predicted labels. **O**cclusion **ratio** and **G**host marker **ratio** are respective fractions of the total expected number of labels. They provide some indication of the difficulty of a dataset. Our network is quite accurate when there are few occlusions or ghost markers. Because we perform data augmentation aggresively in training, the network performs comparably in real sequences to a synthetic one with similar numbers of occluded/ghost markers. Though our network is robust to a large number of ghost markers, it is more vulnerable to occlusions.

Test set	Synthetic	Single hand	Hand-hand	Hand-pen	Hand-controller
# Labels	957201	181720	129002	179752	103323
Accuracy	85.81%	97.64%	84.08%	77.81%	85.32%
O ratio	12.24%	0.07%	2.73%	2.03%	1.78%
O precision	73.95%	38.91%	59.73%	52.65%	63.17%
O recall	64.64%	37.89%	25.57%	17.85%	28.44%
G ratio	5.27%	0.01%	10.63%	7.55%	4.50%
G precision	56.21%	87.50%	79.46%	79.09%	70.89%
G recall	39.77%	87.50%	61.55%	63.65%	52.78%

Table 5. **Comparison of real-time tracking performance.** This table shows the number and ratio of correctly labeled frames.

Method	[Alexanderson et al. 2017]	Ours
Single hand	4330 (90.49%)	4785 (100.00%)
Hand-hand	981 (29.62%)	4477 (99.24%)
Hand-pen	402 (8.91%)	3101 (93.73%)
Hand-controller	1381 (52.21%)	2599 (98.26%)

and follow the marker calibration poses (see video). Afterwards, they can readily move their hands comfortably and freely in the capture volume, while their motion is tracked reliably in real-time.

Dexterous manipulation. Natural motions of hands interacting with the environment or manipulation of everyday objects provide invaluable insight for the reseach of biomechanics, usability, and animation. Our hand tracking system can capture hands manipulating objects in a cluttered environment, such as folding paper under severe occlusion as shown in the video. Strong interactions between hands or between hands and other body parts, such as finger spelling and sign language, are handled gracefully by our system.

Performance capture. Performance capture is integral to the creation of digital humans for entertainment and education. Its utility in complicated scenarios however is often limited by the capturing technology. For instance, dance movements may have frequent occlusion and interactions. Playing musical instruments requires close finger proximity and fast but subtle movements, as well as interacting with the instrument itself. Using our hand tracking system, we can successfully capture performances including tutting and playing percussion, string, and woodwind instruments. These performances can be captured using the same setup without special instructions to the performers, all in real-time (Figure 8; top left). Even though reflective surfaces on the musical intruments cause a considerable amount of noise near the hands (the drum sequence has up to 20 ghost markers per frame, please see video), our system still produces accurate hand motions.

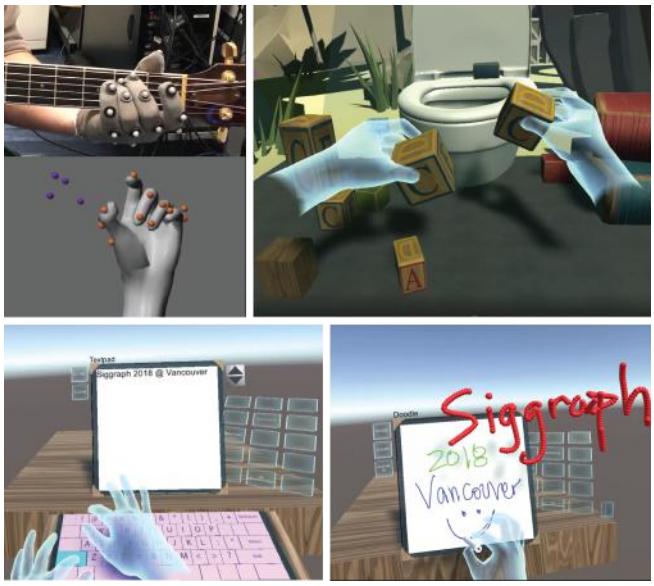


Fig. 8. **Our system enables a wide range of applications** Top Left: A view of a guitar chord from the player's perspective (ghost markers are rendered as purple spheres.) Top Right: Manipulation of virtual objects in VR. Bottom Left: Typing with a virtual keyboard. Bottom Right: 2D/3D drawing with a virtual stylus.

Virtual reality input. Hands are an attractive form of input in VR as we are accustomed to using our hands to interact in 3D environments and our hands are always readily available. With high-fidelity hand tracking, we can begin to explore the possibilty of using hand input in virtual environments, such as for virtual object manipulation (Figure 8; top right), interaction with virtual buttons and screens, typing on a virtual keyboard, and 2D/3D drawing (Figure 8; bottom). Key to the success of hand input is the accuracy and robustness of the hand motions. Only with a reliable hand tracking system can we forgo the questions on hand motion quality, and instead focus on developing the interaction model itself.

Table 6. **Test with a sparse marker set.** Similar to Table 4, we compare network inference results on four sequences, but here we use a network trained for a sparse marker set. Comparing to Table 4, we see a slight drop in performance, but overall similar characteristics.

Test set	Synthetic	Single hand	Hand-hand	Hand-pen	Hand-controller
# Labels	680680	124313	87292	126693	71778
Accuracy	84.89%	96.90%	82.14%	72.45%	86.59%
O ratio	10.96%	0.09%	1.67%	2.28%	1.83%
O precision	60.75%	59.29%	76.06%	41.93%	89.05%
O recall	66.85%	59.29%	17.45%	10.82%	29.68%
G ratio	7.64%	0.01%	10.74%	10.84%	6.09%
G precision	47.29%	81.25%	66.52%	65.47%	68.60%
G recall	54.10%	81.25%	57.64%	53.93%	54.84%

Table 7. **Performance comparison with/without user marker calibration.** First number is the percentage of frames which pass our preset IK threshold (8 mm RMS) (i.e., number of valid tracking frames / number of overall frames) while the second number is the fraction of correctly labeled frames as valid tracking frames (i.e., number of correctly labeled frames / number of valid tracking frames)

Test set	With calibration	Without
Single hand	100.00% / 100.00%	98.27% / 95.62%
Hand-hand	99.18% / 94.40%	95.56% / 85.18%
Hand-pen	99.38% / 98.95%	99.29% / 77.12%
Hand-controller	99.17% /99.09%	97.58% / 89.54%

7 CONCLUSION

We have proposed a fast and robust solution for tracking hands with an optical motion capture system. The crux of our approach is a novel labeling step that applies a neural network to resolve marker labeling by posing it as a keypoint estimation problem on 2D images. We have demonstrated that our deep labeling technique can be made robust to hand size, hand proportions, and different types of motions simply by generating a synthetic dataset with appropriate variation. Furthermore, data augmentation with artifacts such as occluded markers and ghost markers are effective at improving robustness to these artifacts on both real and synthetic data. Our technique is equally applicable to sparse and dense marker sets.

Taking advantage of efficient evaluation of neural networks on modern GPU hardware, our system easily runs at 120Hz, enabling new applications for hand motion capture such as usability research and real-time interaction prototyping in virtual reality. Our technique also provides a practical method for collecting large bodies of natural, accurate hand motion that can be used as ground-truth data for many research problems. We are releasing all of our synthetic training data and trained convolutional model with the publication of this paper.

8 DISCUSSION AND FUTURE WORK

Our current conventional VGG-style labeling network is easy to train, fast to evaluate and generates accurate predictions. There are definitely other applicable network architectures that may outperform our vanilla CNN structure. For instance, our method only utilizes the information from a single frame to determine the labeling of each marker. In effect, we are making a hard decision at the labeling step that cannot be overriden during tracking or inverse kinematics. Recent work in computer vision has explored predicting multiple hypotheses or regression to probability distributions. Multiple hypothesis tracking or a recurrent neural network, which take in account of temporal information from previous frames would lead to more accurate results. At the same time, clustering is required at the initialization of tracking. Although it fails on 50% of frames due to random hand assignment, at 120Hz a few missed frames at the start of tracking is barely noticeable to the user. Future work could explore labeling the two hands jointly when clustering fails.

We have addressed marker set calibration, but have not presented a general method for also calibrating the skeleton of the user. (The approximate size of the user's hand is selected a priori in our experiments.) Future work should be able to leverage priors on hand shape to calibrate the hand shape and marker placement simultaneously from motion sequences.

We have shown just a few examples of synthetic data augmentation to address artifacts such as occluded markers, but this could be extended to more robust labeling of hand-hand interaction, hand-body interaction or even multi-person interactions. Finally, nothing about this technique is specific to a particular marker layout or to human hands. We hope to motivate a new class of motion capture algorithms that make use of deep labeling for tracking dense marker sets on human bodies or even on animals.

ACKNOWLEDGMENTS

We would like to thank Simon Alexanderson for generating labeling results showed in Table 5 and Matt Longest for prototyping VR interactions. We also thank Albert Hwang, Cristine Cooper, Aaron Echeverria, and Mark Terrano for generous help with video recording.

REFERENCES

2018. OptiTrack Motion Capture Systems. (2018). https://www.optitrack.com
2018. Vicon Motion Systems. (2018). https://www.vicon.com/
Simon Alexanderson, Carol O'Sullivan, and Jonas Beskow. 2016. Robust Online Motion Capture Labeling of Finger Markers. In *Proceedings of the 9th International Conference on Motion in Games (MIG)*.
Simon Alexanderson, Carol O'Sullivan, and Jonas Beskow. 2017. Real-time labeling of non-rigid motion capture marker sets. *Computers & graphics* (2017).
Tameem Antoniades. 2016. Creating a Live Real-time Performance-captured Digital Human. In *ACM SIGGRAPH 2016 Real-Time Live! (SIGGRAPH '16)*.

A. Aristidou and J. Lasenby. 2010. Motion capture with constrained inverse kinematics for real-time hand tracking. In *2010 4th International Symposium on Communications, Control and Signal Processing (ISCCSP)*.

Adrian Bulat and Georgios Tzimiropoulos. 2016. Human Pose Estimation via Convolutional Part Heatmap Regression. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (Eds.). Springer International Publishing, Cham, 717–732.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms, Third Edition* (3rd ed.). The MIT Press.

Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. 2017. Mask R-CNN. In *The IEEE International Conference on Computer Vision (ICCV)*.

Wheatland N. Neff M. Zordan V. Kang, C. 2012. Automatic Hand-Over Animation for Free-Hand Motions from Low Resolution Input. In *Proceedings of Motion in Games (MIG)*.

M. Kitagawa and B. Windsor. 2008. *MoCap for Artists: Workflow and Techniques for Motion Capture*. Focal Press.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Jonathan Maycock, Tobias Rohlig, Matthias SchrÃűder, Mario Botsch, and Helge J. Ritter. 2015. Fully automatic optical motion tracking using an inverse kinematics approach.. In *15th IEEE-RAS International Conference on Humanoid Robots*. 461–466.

J. Meyer, M. Kuderer, J. Müller, and W. Burgard. 2014. Online marker labeling for fully automatic skeleton tracking in optical motion capture. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*.

Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A. Argyros. 2012. Tracking the Articulated Motion of Two Strongly Interacting Hands. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Gernot Riegler, Osman Ulusoy, and Andreas Geiger. 2017. OctNet: Learning Deep 3D Representations at High Resolutions. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Maurice Ringer and Joan Lasenby. 2002a. Multiple Hypothesis Tracking for Automatic Optical Motion Capture. In *The 7th European Conference on Computer Vision (ECCV)*.

Maurice Ringer and Joan Lasenby. 2002b. A Procedure for Automatically Estimating Model Parameters in Optical Motion Capture. In *In British Machine Vision Conference*. 747–756.

Matthias Schröder, Jonathan Maycock, and Mario Botsch. 2015. Reduced Marker Layouts for Optical Motion Capture of Hands. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games (MIG)*.

Matthias Schröder, Thomas Waltemate, Jonathan Maycock, Tobias RÃűhlig, Helge Ritter, and Mario Botsch. 2017. Design and evaluation of reduced marker layouts for hand motion capture. *Computer Animation and Virtual Worlds* (2017), e1751. https://doi.org/10.1002/cav.1751

T. Schubert, K. Eggensperger, A. Gkogkidis, F. Hutter, T. Ball, and W. Burgard. 2016. Automatic bone parameter estimation for skeleton tracking in optical motion capture. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*.

T. Schubert, A. Gkogkidis, T. Ball, and W. Burgard. 2015. Automatic initialization for skeleton tracking in optical motion capture. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*.

Kevin J. Shih, Arun Mallya, Saurabh Singh, and Derek Hoiem. 2015. Part Localization using Multi-Proposal Consensus for Fine-Grained Categorization. In *Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7-10, 2015*. 128.1–128.12. https://doi.org/10.5244/C.29.128

Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. 2017. Hand Keypoint Detection in Single Images Using Multiview Bootstrapping. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

K. Simonyan and A. Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jonathan Taylor, Lucas Bordeaux, Thomas Cashman, Bob Corish, Cem Keskin, Toby Sharp, Eduardo Soto, David Sweeney, Julien Valentin, Benjamin Luff, Arran Topalian, Erroll Wood, Sameh Khamis, Pushmeet Kohli, Shahram Izadi, Richard Banks, Andrew Fitzgibbon, and Jamie Shotton. 2016. Efficient and Precise Interactive Hand Tracking Through Joint, Continuous Optimization of Pose and Correspondences. *ACM Trans. Graph.* 35, 4 (July 2016), 143:1–143:12.

Jonathan Taylor, Vladimir Tankovich, Danhang Tang, Cem Keskin, David Kim, Philip Davidson, Adarsh Kowdle, and Shahram Izadi. 2017. Articulated Distance Fields for Ultra-fast Tracking of Hands Interacting. *ACM Trans. Graph.* 36, 6 (Nov. 2017), 244:1–244:12.

Anastasia Tkach, Mark Pauly, and Andrea Tagliasacchi. 2016. Sphere-meshes for Real-time Hand Modeling and Tracking. *ACM Trans. Graph.* 35, 6 (Nov. 2016), 222:1–222:11.

Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. 2014. Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks. *ACM Trans. Graph.* 33, 5 (Aug. 2014), 169:1–169:10.

Nkenge Wheatland, Sophie Jörg, and Victor Zordan. 2013. Automatic Hand-Over Animation Using Principle Component Analysis. In *Proceedings of Motion in Games (MIG)*.

Nkenge Wheatland, Yingying Wang, Huaguang Song, Michael Neff, Victor Zordan, and Sophie JÃűrg. 2015. State of the Art in Hand and Finger Modeling and Animation. *Computer Graphics Forum* (2015).

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR* abs/1609.08144 (2016).