Metaphoric Hand Gestures for Orientation-Aware VR Object Manipulation With an Egocentric Viewpoint

Youngkyoon Jang, *Member, IEEE*, Ikbeom Jeon, Tae-Kyun Kim, *Member, IEEE*, and Woontack Woo, *Member, IEEE*

***Abstract*—We present a novel natural user interface framework, called Meta-Gesture, for selecting and manipulating rotatable virtual reality (VR) objects in egocentric viewpoint. Meta-Gesture uses the gestures of holding and manipulating the tools of daily use. Specifically, the holding gesture is used to summon a virtual object into the palm, and the manipulating gesture to trigger the function of the summoned virtual tool. Our contributions are broadly threefold: 1) Meta-Gesture is the first to perform bare hand-gesture-based orientation-aware selection and manipulation of very small (nail-sized) VR objects, which has become possible by combining a stable 3-D palm pose estimator (publicly available) with the proposed static–dynamic (SD) gesture estimator; 2) the proposed novel SD random forest, as an SD gesture estimator can classify a 3-D static gesture and its action status hierarchically, in a single classifier; and 3) our novel voxel coding scheme, called layered shape pattern, which is configured by calculating the fill rate of point clouds (raw source of data) in each voxel on the top of the palm pose estimation, allows for dispensing with the need for preceding hand skeletal tracking or joint classification while defining a gesture. Experimental results show that the proposed method can deliver promising performance, even under frequent occlusions, during orientation-aware selection and manipulation of objects in VR space by wearing head-mounted display with an attached egocentric-depth camera (see the supplementary video available at: https://sites.google.com/site/fingergesture/fui/).**

***Index Terms*—Augmented reality (AR), computer vision (CV), egocentric viewpoint, self-occlusion, static–dynamic (SD) forest, 3-D hand gesture, virtual reality (VR).**

I. Introduction

RECENT advances of head-mounted display (HMD), such as HoloLens [1] and Oculus [2], promise several potential possibilities for augmented experiences, especially in interacting with virtual objects in wearable AR/VR environments. While wearing HMD, using bare hands is the most natural way of interacting with virtual objects. As an interaction technique in augmented reality/virtual reality (AR/VR) environments, research on bare-hand-based gesture recognition has produced many fascinating interaction scenarios, notable among them being scene navigation [32], object translocation [11], 3-D model assembly [46], and in-air annotation in a VR environment [19]. However, many of them were not meant for egocentrically viewed interactions, as a result of which, they often cause missing of visual information, because of occlusion by other parts of the hand (called self-occlusions). Moreover, sophisticated interactions, such as those involving selection of a very small (nail-sized) object or manipulation of rotatable object, still remain as challenging issues for conventional approaches, although the state-of-the-art machine-learning-based approach [14] can tackle the (fingertip-sized) problem of selecting small object in a wearable AR/VR environment.

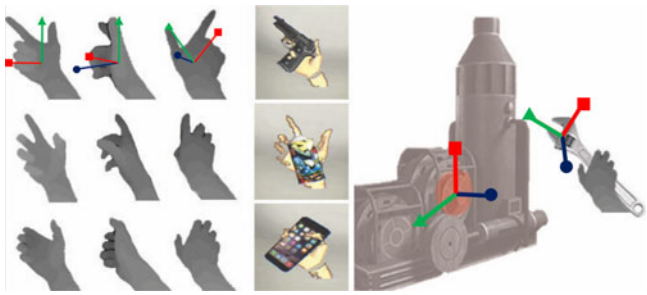

Fig. 1. Our system, which we refer to here as Meta-Gesture, supports orientation-aware selection and manipulation of small-sized AR objects by replicating the gestures of utilizing a tool. More technically, Meta-Gesture identifies partial static 3-D hand shapes for summoning a functional object up to the hand and then classifies 3-D movement patterns of hand parts (i.e., functions) for manipulating the functional AR object. The user can summon a tool intuitively by making a grabbing shape for that specific tool. Moreover, the user can manipulate the summoned AR object to activate the function of the tool.

Although there are many studies on bare-hand-based gesture recognition in the field of computer vision/human–computer interaction (CV/HCI), the most recent methods (discussed in Section II-A) cannot be directly repurposed to our interaction scenario, which involves orientation-aware selection and manipulation of objects in wearable AR/VR environment, because of the following challenges of the task.

1) *Sophisticated level of interaction:* Our target is selection of very small (nail-sized) object and orientation-aware manipulation (especially displacement). None of the existing methods can tackle this type of interaction scenario without utilizing an additional device (e.g., gloves, a wand).
2) *Variances of viewpoint:* When interacting with AR/VR objects in egocentric viewpoint, self-occlusion is the most challenging problem, because the finger bends under the back of the hand. During orientation-aware selection and manipulation, the hand must keep rotating, because of which, the shape of the hand keeps constantly changing.

Manuscript received October 2, 2015; revised February 8, 2016, May 16, 2016, and July 17, 2016; accepted August 9, 2016. This work was supported by National Research Foundation of Korea (NRF) Grant 2014R1A2A2A01003005. This paper was recommended by Associate Editor G. Serra.

Y. Jang is with Queen Mary University of London, London, E1 4NS, U.K., and also with the University of Cambridge, Cambridge, CB2 1TN, U.K. (e-mail: youngkyoon.jang@qmul.ac.uk).

I. Jeon and W. Woo are with the Korea Advanced Institute of Science and Technology, Daejeon 34141, South Korea (e-mail: ikbeomjeon@kaist.ac.kr; wwoo@kaist.ac.kr).

T.-K. Kim is with Imperial College London, London, SW7 2AZ, U.K. (e-mail: tk.kim@imperial.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/THMS.2016.2611824

To address the above challenges, we propose Meta-Gesture (see Fig. 1),[1] which performs simultaneous static–dynamic (SD) gesture estimation. The idea of Meta-Gesture germinated from the following question: "What if we could magically summon a tool up to the hand and manipulate it as occasion demands, in ways similar to those in [9], [12], and [47], but without the help of any additional device?" Because we have already learned the gesture of grasping and manipulating everyday tool, the gesture is intuitive and easy to memorize. Moreover, Meta-Gesture can also support sophisticated interactions, based on a summoned virtual tool, such as a pair of tweezers, rather than through direct bare hand contact with AR/VR devices. Thus, interestingly, even though Meta-Gesture is independent of any additional devices, it has the same effect as utilizing a physical tool.

The state-of-the-art CV methods, categorized mainly by discriminative (see Section II-C) and generative [13], [30], [31], [42] approaches, stably estimate 3-D hand posture. Nevertheless, they are insufficient to be directly repurposed for Meta-Gesture, which requires joint treatment for holding (static) and manipulating (dynamic) gestures, because of the following challenges.

1) *Imperfect feature representation in egocentric viewpoint:* It is difficult to estimate the exact hand pose when a finger's visual information is missing, because of self-occlusion, especially when the generative methods incorrectly estimate the skeletal pose of the completely occluded finger area. Thus, defining a gesture, based on an imperfect feature representation, causes an error in both static and dynamic gesture recognitions.
2) *Variances of SD parts for defining different gestures:* An array of inconsistencies exists in both static and dynamic gestures between different functional AR objects. For instance, the possible gestures of holding a stylus pen with the index finger pressing a button on the pen are different from those of holding and pressing buttons on a mobile phone with the thumb. Thus, the problem of estimating static and dynamic gestures cannot be solved in a straightforward manner with the conventional hand pose estimator.

Addressing the above challenges, we propose a novel SD forest utilizing the proposed layered shape pattern (LSP) feature. The SD forest is a hierarchical random forest, which classifies static and dynamic gestures. It is designed to take the characteristics of both static (e.g., 3-D LSP value changes along the spatial axis) and dynamic (e.g., 3-D LSP value changes along the temporal axis) features, with high stability in egocentric viewpoint. For that, the SD forest learns from the sequence of SD voxel features consisting of hand point clouds, which are captured in the local coordinates of the hand. By voxelizing a cuboid region surrounding the hand, one can define the SD voxel feature, based only on the position of the point clouds, but without requiring any preliminarily image analysis. Moreover, the SD forest can further define the best split function, utilizing the most important index of the feature and the offset time, both of which have to be considered when estimating a 3-D static gesture, or a manipulating action status as a dynamic gesture at each split node.

For our system, we foresee numerous potential AR/VR applications, which augment user's experience (see Section V). The contributions of our work are threefold.

1) *Novel NUI, supporting a sophisticated level of interaction:* For rotatable VR object selection and manipulation, we suggest Meta-Gesture, based on the results of SD forest. Summoning a function-equipped AR object and its manipulation help the user in interacting with very small (nail-sized) target objects in wearable AR/VR environments, besides providing better accuracy than conventional methods [10], [14].
2) *Joint spatial–temporal pattern treatment:* The proposed SD forest is designed to classify static and dynamic gestures hierarchically in a single tree, which does not require any specified rule for defining a gesture. Based on a set of labeled videos (containing replicated hand gestures of tool utilization), it automatically trains and selects an optimal feature index and temporal frame offsets to define the SD gestures in various viewpoints (including egocentric viewpoint).
3) *Novel 3-D shape feature representation:* The proposed LSP is defined by combining the results of 3-D palm pose estimator and hand point clouds. The proposed pattern encodes the holistic shape of the hand, which can be represented as a concatenation of the LSPs. Moreover, and very interestingly, the pattern is reusable to classify shapes from different viewpoints. For example, just as an LSP can be calculated from the egocentric viewpoint causing self-occlusions, the pattern can similarly be extracted from frontal viewpoint. Based on the proposed LSP, SD gesture is compressively encoded, and this necessitates minimal memory for training process.

II. Related Work

Hand gesture recognition has a long and diversified history across the research communities, engaged in working on such research topics as AR, VR, CV, HCI, and UI. With the advent of real-time RGB-D cameras, the research topics of hand gesture estimation received significant boost. In this section, we will discuss some of the more recent works on hand gesture recognition; for a detailed survey of earlier hand gesture recognition approaches, see [34] and [36].

A. Analyzed Feature-Based Gesture Estimation

For hand gesture recognition, most works require that image feature analysis (e.g., partial hand shape detection [35]; fingertip/palm center position detection [10], [20]; hand joints estimation [14], or per-pixel classification for hand [37] and its joint region segmentation [22], [23]) be carried out before defining and estimating a static or dynamic gesture. As already discussed, hand gestures captured in egocentric viewpoint had far more severe occlusions in our interaction scenarios, because of which we had to carry out orientation-aware manipulation also.

[1]Meta-Gesture is the name of the proposed framework; it enables the user to interact with virtual objects more intuitively, even though the user does not hold any tool or device.

Thus, the approaches that depend on feature extraction require exponentially more processing power to extract noise-tolerant features, even before defining and estimating a gesture, and this renders their direct application difficult. Furthermore, although there are previous studies on dynamic gesture recognition [5], [27], based on hand region trajectories or partial velocity values within the separated hand regions, their working environments are restricted to the front-facing viewpoint.

B. Hand-Related Interaction in Augmented Reality/Virtual Reality

Even before the current advances of HMD, hand-based interaction (such as fingertip positioning [7], [15], [21], [39]; clicking action [14]; or hand gesture [10], [33]) has been used as a natural user interface (NUI) in AR/VR environments. In addition to the bare-hand-based methods, some studies were carried out using haptic [4], wrist-worn [18], and finger-worn [28], [29] sensors, and combinations of multiple visual sensors [19] with the device's built-in touch panel [12] or motion-sensing mechanical keyboard [45]. Visual sensor-based approaches in AR/VR do not provide a more sophisticated interaction performance than the wearable sensor-based approaches, because the estimated hand joints or fingertips are not accurate enough, especially under self-occlusions from various viewpoints. On the other side, in wearable sensor-based interactions, the user feels inconvenienced to wear a sensor that supports only a few interaction modes, such as selecting and releasing, which even the sophisticated visual sensors can support just as well. To combine the benefits of both visual sensors and wearable sensors, we propose Meta-Gesture, which guarantees a sophisticated interaction performance in egocentric viewpoint, without wearing any device or sensor.

C. Hierarchical Random Forests in Hand Pose Estimation

As a discriminative framework, random-forest-based approaches have achieved remarkable success in hand pose estimation [16], [17], [38], [43], [44]. Specifically, for dealing with partial occlusions, Tang *et al.* [44] propose semisupervised transductive regression forests (TRFs), which hierarchically classify viewpoint, hand joints, and then conduct pose regression. However, their patch-based joint classification method fails to estimate the finger joints, which are often completely occluded in egocentric viewpoint, even though it can refine joint locations, based on the kinematic constraints. Keskin *et al.* [17] and Song *et al.* [40], [41] propose a similar concept of multilayered random forest for hand pose estimation and hand shape classification. However, for classifying a hand pose, representing the same gesture captured in various viewpoints, their method requires a huge synthetic dataset, generated from all possible viewpoints. That might require an inefficient training process for supporting our scenarios, because viewing angles frequently change during orientation-aware object manipulation in egocentric viewpoint.

In AR/VR community, Jang *et al.*'s [14] method can instantly detect clicking actions and occluded fingertip positions, based on spatiotemporal forest (STF), under self-occlusions in egocentric viewpoint, an environmental setting that is similar to that of ours. However, the method requires (manually labeled) ground-truth data of the occluded fingertip position for conducting a regression function at the training stage, which is cumbersome. In contrast, the proposed framework does not require the regression process; instead, it requires only short depth image sequences for training, while assuring instant-action status estimation. Moreover, the STF can detect only the clicking action of a specified finger, whereas our framework has no such constraint. Based on the SD forest for Meta-Gesture, we can define a gesture, as viewed from different angles, no matter how many fingers move simultaneously.

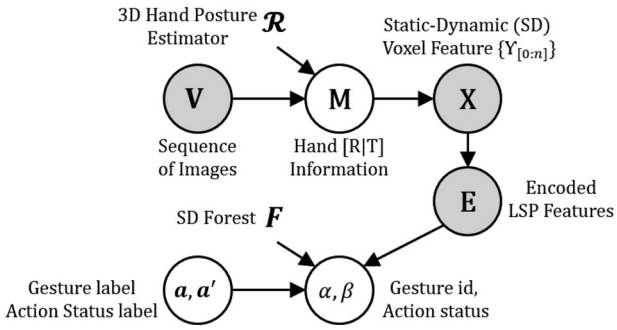

Fig. 2. Graphical representation of the proposed framework.

III. METHODOLOGY

A. Problem Formulation

In this paper, the simultaneous 3-D static and dynamic gesture estimation problem is formulated as shown in Fig. 2. Based on the given sequence of depth images, the transformation information M, representing rotation R and translation T of the palm of the hand, is estimated and tracked by a 3-D palm pose estimator $\mathcal{R}$ [25]. The translation and rotation invariant SD voxel feature X is composed of a set of static voxel features $\{\Upsilon_{[0:n]}\}$, which are defined based on the voxelized rectangular cube surrounding the hand point clouds in the local coordinate calculated by hand transformation information M. Then, the SD feature X is encoded into a novel feature E based on the proposed LSP value calculation. The encoded LSP E is utilized as an input for the proposed SD forest F.

Our approach simultaneously learns 3-D partial static gesture a for retrieving a functional AR object and 3-D partial dynamic gesture status a' for manipulating the AR object at the training stage. The proposed SD forest F is trained to separately capture different characteristics, such as the layer position showing LSP pattern, which never changes through the temporal sequence, and the discriminability between gestures or the tendency for changes in 3-D LSP value through the sequences at each node. After passing the feature E through SD forest F, both 3-D partial static gesture α and 3-D partial dynamic gesture status β are estimated by averaging the results stored at the leaf nodes of each tree.

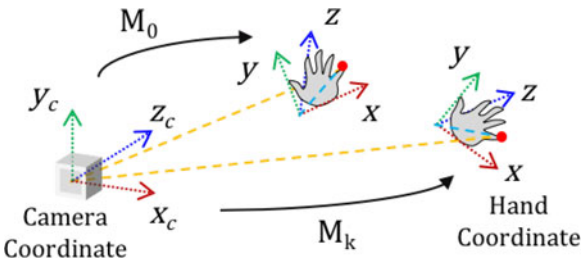

Fig. 3. Example of coordinate transformation from camera coordinates to hand coordinates.

B. Static–Dynamic Voxel Feature

To overcome the problem of imperfect feature representation (described in Section I), we propose a simple voxel-based coding scheme to represent a hand shape, as shown in Fig. 4. The proposed feature representation is very simple, but powerful for defining and estimating both 3-D static and dynamic gestures in different viewpoints. The proposed SD voxel feature X directly utilizes the source of raw data, namely the position of 3-D point clouds of the hand, rather than estimating hand joints from an image. Thus, it can define noise-tolerant features despite the failure of finger joint estimation by the utilized 3-D hand pose estimator $\mathcal{R}$ under self-occlusions in egocentric viewpoint.

Before making the SD voxel feature X, as described inxbrk 3-D Finger CAPE [14], utilizing the state-of-the-art 3-D palm pose estimator [25], we transform the hand point clouds located within the camera coordinates into the local coordinates of the hand, as shown in Fig. 3. For example, when two hand images are captured at different time slots (say 0, k), the coordinates of the pinky fingertips (red dots) are different from those within the camera coordinate system, while the transformed coordinates [based on (1)] are the same as those within the local (hand) coordinate system

$$v_l = M_0 v_{g\,0} = M_k v_{g\,k} = \begin{pmatrix} R_k & T_k \\ 0^\top & 1 \end{pmatrix} v_{g\,k} \tag{1}$$

where v_l is a local coordinate of a hand point cloud, $v_{g\,0}$ and $v_{g\,k}$ are global (camera) coordinates captured at different time slots (say 0, k, respectively), M is the palm transformation matrix composed of a 3×3 rotation matrix R, and a 3×1 translation matrix T [14].

1) Static–Dynamic Voxel Feature Configuration: First, static voxel feature is defined by concatenating the voxels $\Upsilon_i = (C_z, N_c)$, located within the local coordinates of the hand, as shown in Fig. 4(a). C_z is the mean depth value of the point clouds, included in the voxel, calculated within the camera coordinates, and N_c is the number (representing density) of the point clouds, which are included in voxel Υ_i, specified by an index i. By accumulating the static voxel features at every frame, the SD voxel feature X, which is a set of static voxel features $\{\Upsilon_{[0:n]}\}$, where n is the number of voxels defined at a frame, is composed.

a) Point cloud count normalization by depth: In contrast with the offset normalization of hand posture estimation methods [43], we normalize the point cloud counts, which are included in a voxel, for later use. When the hand gets closer to the camera, the single voxel includes larger depth pixel counts. To make the SD voxel feature invariant to scale changes, specifically, we normalize the counts of the point clouds $\Upsilon_i(N_c)$, which are stored in ith voxel Υ_i as

$$\Upsilon_i(\hat{N}_c) = b + \frac{c}{\Upsilon_i(C_z)} + \Upsilon_i(N_c) \tag{2}$$

where $\Upsilon_i(\hat{N}_c)$ is the normalized count number of the point clouds, $\Upsilon_i(C_z)$ represents the mean depth value (within camera coordinates) of the point clouds inside the voxel Υ_i, b is a constant value for shifting the numbers, so that a number is not negative, and c is a constant factor value, which is applied for normalizing the point cloud count. We could get the value simply by solving the equation as

$$c = \left(\Upsilon_i(N_c)_2 - \Upsilon_i(N_c)_1\right) \times \frac{\Upsilon_i(C_z)_1 \times \Upsilon_i(C_z)_2}{\Upsilon_i(C_z)_2 - \Upsilon_i(C_z)_1} \tag{3}$$

where $\Upsilon_i(\cdot)_1$ and $\Upsilon_i(\cdot)_2$ represent the same ith voxel Υ_i, captured in two different frames. We intentionally select the two frames that show large depth differences to get the factor value c.

2) Three-Dimensional Layered Shape Pattern

We propose the novel concept of LSP to capture partial finger movement, which is technically a change in the amount of point clouds in a voxel, as well as static shape pattern on a layered voxel plane. To make the voxel plane $\mathrm{LSP}^{\hat{u},j}$, we selectively specify the axis-dependent index j along the designated axis $\hat{u}$ within the local hand coordinates, as shown in Fig. 4(a). To avoid confusion in interpreting the variables, we clearly explain here the meaning of the variables we used for representing $(\mathrm{LSP}^{\hat{u},j})_{\mathrm{axis}}$. $\hat{u}$ is the designated axis, and j is an index along the axis for specifying the LSP plane; axis represents a label, which indicates that the LSP value is calculated along the indicated label direction, after configuring the LSP plane, based on $\hat{u}$ and j.

a) Layered shape pattern value: After configuring the voxel plane $\mathrm{LSP}^{\hat{u},j}$, we can extract the $\mathrm{LSP}_{\mathrm{axis}}$ value, based on the octal number system, as shown on the right-hand side of Fig. 4(b), because the density information is important to represent the significance of a bit. For instance, when a few noisy point clouds are included in a voxel, the binary number system represents complete inclusion (by giving 1 among [0:1]) of the point clouds in our scenario, while the octal number system represents the minimal inclusion (by giving 1 among [0:7]). Moreover, the region farther from the origin of the local coordinate represents more important information of the dynamic gestures, as shown in Fig. 4(b).

Based on the reconfigured voxel plane $\mathrm{LSP}^{\hat{u},j}$, two $\mathrm{LSP}_{\mathrm{axis}}$ values can be extracted along each designated axis, configuring a plane, as shown in Fig. 4(b). The changes in the $\mathrm{LSP}_{\mathrm{axis}}$ value between two voxel planes, captured randomly at two different times, p and q, represent the same hand part (finger) movement, as shown in Fig. 4(b). Thus, the values along an axis, representing the same dynamic gesture (finger movement),

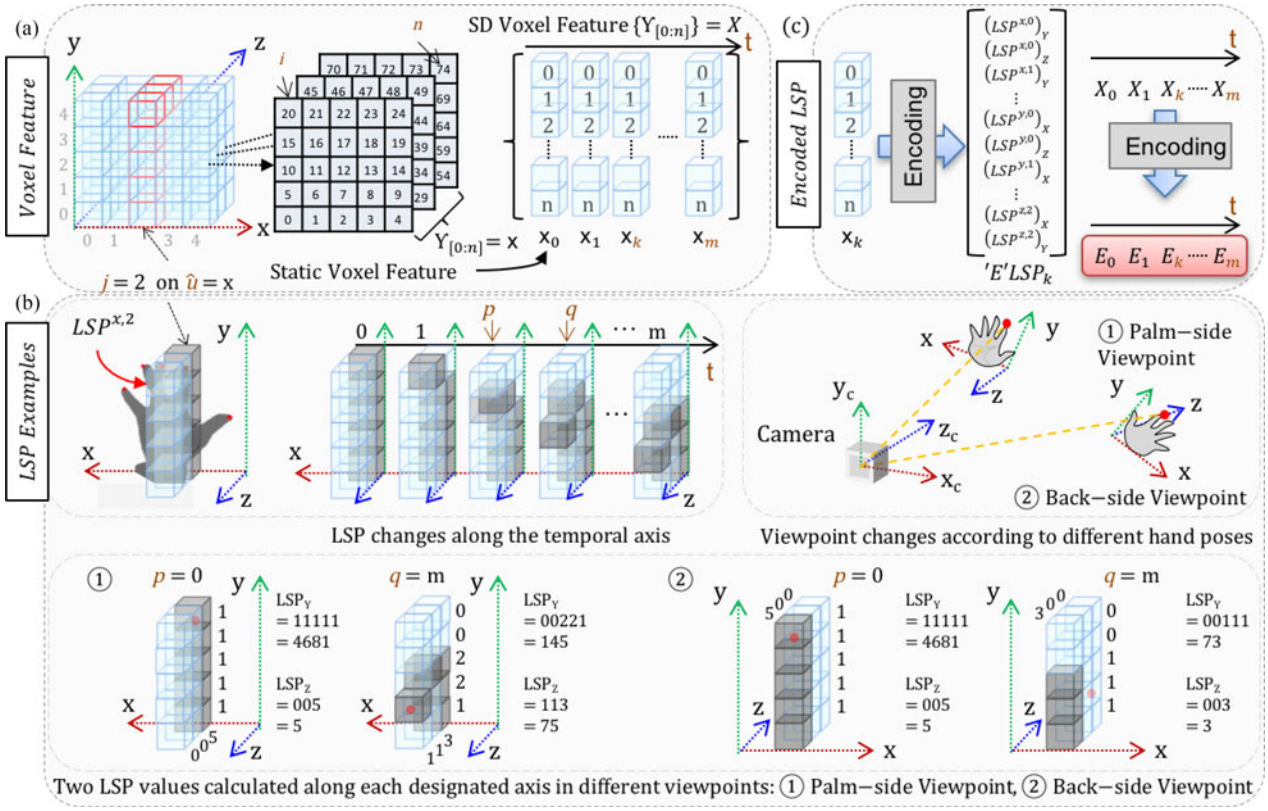

Fig. 4. Example of encoded LSP feature configuration. (a) Example of the proposed SD voxel feature configuration. (b) Example of a layered shape on a selected voxel plane (changed through the sequence) and LSP pattern values, calculated from two different frames, based on octal number system in different viewpoints. (c) Example of the encoded LSP feature, which is used as an input of the SD forest.

should correlate as they change, either both increasing or both decreasing, regardless of the variances in viewpoint.

For example, the LSP_Y value of Fig. 4(b) captures the correlating decreasing value changes in both palm-side and back-side viewpoints, although the index fingertip (red dot) is occluded, especially in back-side viewpoint. However, LSP_Z cannot capture value changes and determine whether or not they correlate, because of self-occlusions from the back-side viewpoint. Thus, finding the optimal axis that can capture a consistent correlation between LSP value changes is important. Our proposed SD random forest learns the optimal axis during its training stage (see Section III-C1 for a detailed description).

b) Encoding voxel feature: We finally encode all the possible LSP values to make an encoded LSP feature $E_{[0:L]}$ [see Fig. 4(c)], where $L = (\mathrm{DIM}_{\mathrm{axis:X}} \times 2) + (\mathrm{DIM}_{\mathrm{axis:Y}} \times 2) + (\mathrm{DIM}_{\mathrm{axis:Z}} \times 2)$ is the dimension of the encoded LSP feature. For example, the dimension of the encoded LSP feature E of Fig. 4(a) is 26, while the dimension of the voxel feature x_k is 75. Moreover, we expect that a $10 \times 10 \times 10$ cuboid $\mathrm{x} = \Upsilon_{[0:999]}$ can be encoded in 60 dimensions of features, representing $E_{[0:59]}$. This encoded LSP E is used for training and testing.

To access an element of the encoded LSP feature E, only an index of the element is required. The mechanism of assigning an index to an element of E is followed by taking parameters $\hat{u}$, j, and axis, in that order. For parameter $\hat{u}$, x, y, and z are assigned in that order to specify an LSP plane. For parameter j, the numbers from 0- to $\hat{u}$-dependent-axis dimension are assigned in order. Then, finally, for parameter axis, (Y and Z), (X and Z), and (X and Y) are assigned, when $\hat{u}$ is x, y, and z, respectively.

For example, all elements of the encoded LSP E of Fig. 4(c) are configured as follows, based on the indexing mechanism described: [**Index**: $\hat{u}$, j, axis], [**0**: $x, 0, Y$], [**1**: $x, 0, Z$], [**2**: $x, 1, Y$], [**3**: $x, 1, Z$], [**4**: $x, 2, Y$], [**5**: $x, 2, Z$], [**6**: $x, 3, Y$], [**7**: $x, 3, Z$], [**8**: $x, 4, Y$], [**9**: $x, 4, Z$], [**10**: $y, 0, X$], [**11**: $y, 0, Z$], [**12**: $y, 1, X$], [**13**: $y, 1, Z$], [**14**: $y, 2, X$], [**15**: $y, 2, Z$], [**16**: $y, 3, X$], [**17**: $y, 3, Z$], [**18**: $y, 4, X$], [**19**: $y, 4, Z$], [**20**: $z, 0, X$], [**21**: $z, 0, Y$], [**22**: $z, 1, X$], [**23**: $z, 1, Y$], [**24**: $z, 2, X$], [**25**: $z, 2, Y$]. A visualized LSP plane, based on the index, is shown in a split node in Fig. 5(b).

C. Static–Dynamic Forest

The aim of the proposed SD forest is to hierarchically estimate two different (but interdependent) types of 3-D gestures, which are static and dynamic, in a single classifier, despite self-occlusions and the variances in viewpoint. The proposed SD forest learns the static and dynamic gestures, based on the given gesture prior labels a, a', as shown in Fig. 2.

Similar to random forest in the basic structure [6], the proposed SD forest consists of randomized binary decision trees, containing two types of nodes: split and leaf. In the SD forest, split nodes perform the gesture type-specific test function, as determined by the layer (top: static, bottom: dynamic gesture) and

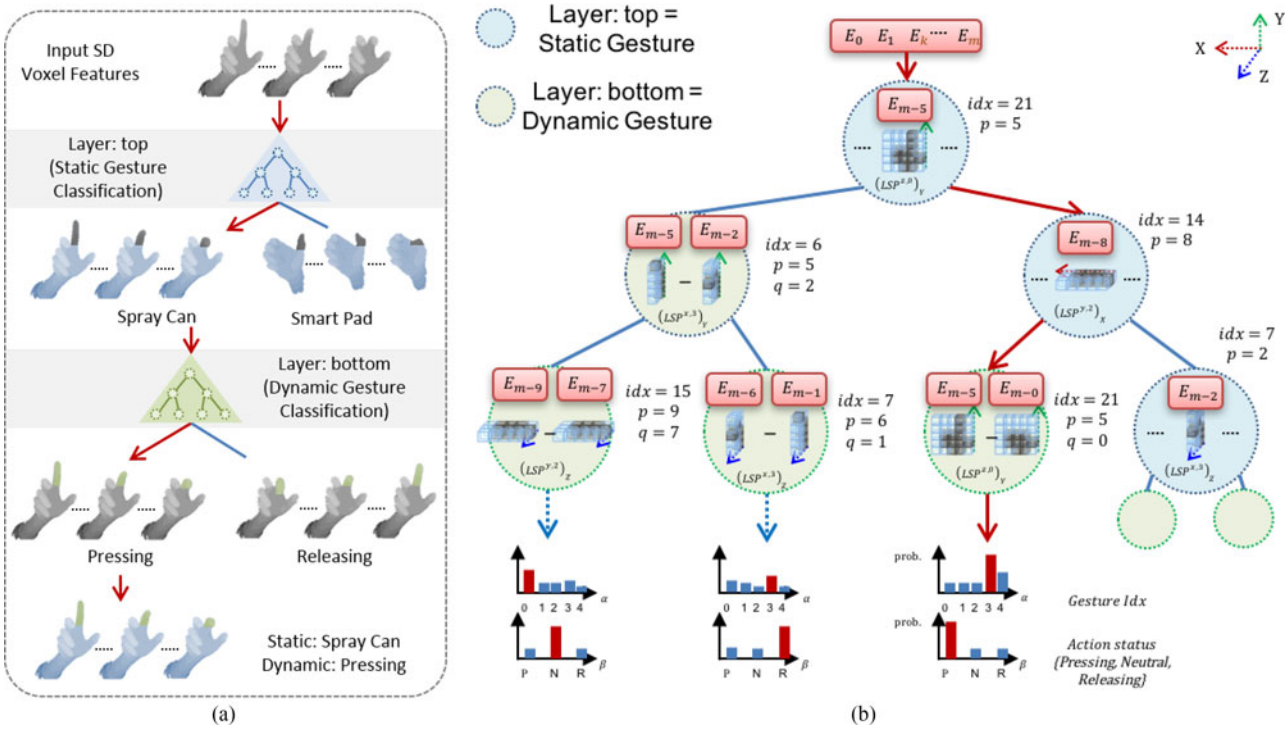

Fig. 5. Proposed hierarchical SD gesture learning model. (a) Layers in the gesture type hierarchy. (b) Illustration of the SD forest process. (Best viewed in color).

make the decision to route input data either left or right. The leaf nodes are terminating nodes, which represent the static gesture as well as the status of the manipulating dynamic gesture related to the static gesture; they store the probability of the static gesture ID and the status of the dynamic gesture. Moreover, the SD forest proposed by us is inspired by three different RFs: TRFs [44], multilayered randomized decision forest (MLRDF) [17], and STF [14]. Specifically, hierarchical classifications for different tasks are inspired by TRF [44], the concept of holistic hand-shape classification is inspired by MLRDF [17], and spatiotemporal (ST) information learning in the process of interaction scenario is inspired by STF [14]. Based on the ideas of previous workers, we could structure the proposed novel type of random forest specifically for hierarchical SD gesture classification, as shown in Fig. 5.

Overall, the proposed SD forest is composed of two layers, representing an interdependent hierarchy, as shown in Fig. 5(a). Static gesture classification is first performed at the top levels. In this layer, multiple static gestures are classified based on the static part of a hand. As shown in Fig. 5(a), blue-colored hand parts represent static point clouds along a sequence of frames that are discriminable between different grabbing gestures, while gray-colored hand parts vary over the process of time. Once the static gesture is determined at the top levels, the dynamic gesture (action status) is classified by checking the learnt pattern of partial finger movements along the timeline. As shown in Fig. 5(a), green-colored hand parts obviously produce different patterns along the sequence of frames, because gray-colored hand parts are static for a grabbing shape. Then, finally, static and dynamic gestures are classified through the layers of SD forests.

1) Static–Dynamic Forest Learning

While learning, each tree in the SD forest is grown by recursively splitting and passing the current training data to two child nodes, which are the same as random forests [6]. At each split node, we randomly generate splitting candidates $\Psi = (f_\iota, \tau_\iota)$, consisting of a split function f_ι and a threshold τ_ι. And, we find an optimal parameter set $\psi^\dagger{}_\iota \in \Psi$ for splitting a dataset at a split node, which gives us the largest information gain (IG) value, as described in the previous works. For the proposed SD forest, the function f_ι for a splitting candidate is defined as

$$f_\iota(\mathrm{E}) = \begin{cases} (E_{m-p})_{idx}, & \text{if layer:top.} \\ (E_{m-p})_{idx} - (E_{m-q})_{idx}, & \text{if layer:bottom} \end{cases} \tag{4}$$

where $(E_{m-p})_{idx}$ and $(E_{m-q})_{idx}$ are the LSP values, designated by random preceding offsets p, q (from the last frame m of a sequence), and index idx of the encoded LSP feature.

a) Static gesture classification: At layer:top, $(E_{m-p})_{idx}$ is the splitting function for capturing static gesture characteristics. Based on the split function f_ι and randomly generated threshold τ_ι, the given SD gesture dataset $\mathcal{D}$ is split into two subsets $\mathcal{D}_l$ and $\mathcal{D}_r$, such that $\mathcal{D}_l = \{(E_{m-p})_{idx} < \tau_\iota\}$ and $\mathcal{D}_r = \mathcal{D} \backslash \mathcal{D}_l$. Then, traditional IG [see (5)] is used to evaluate the classification performance of all the static gesture priors labeled a (of Fig. 2) in dataset $\mathcal{D}$ [6].

As mentioned, the processes of splitting and passing training data were recursively conducted until the layer-switching criteria are satisfied. Inspired by Tang *et al.* [44], adaptive switching was employed to move on to the next layer for dynamic gesture classification. The first layer:top was switched into the next layer:bottom, when "the difference between the highest poste-

rior of a gesture label and the second highest posterior in a node" was larger than the threshold. We set 0.9 as a threshold as in [44].

b) Dynamic gesture classification: At layer:bottom, representing dynamic gesture classification, the function returns the difference between the two encoded LSP values, which can be accessed by randomly selecting two offsets, p and q, in terms of the preceding time (ms) from the time m measured at the last frame of a sequence. This function captures the patterns of change in LSP values during hand part (e.g., finger) movements, such as pressing, neutral, and releasing. Similar to what was done in static gesture classification, the given SD gesture dataset $\mathcal{D}$ was split into two subsets $\mathcal{D}_l$ and $\mathcal{D}_r$, such that $\mathcal{D}_l = \{(E_{m-p})_{idx} - (E_{m-q})_{idx} < \tau_\iota\}$ and $\mathcal{D}_r = \mathcal{D} \backslash \mathcal{D}_l$. In addition, IG [see (5)] was used to evaluate the classification performance of all the dynamic gesture priors, labeled a' (in Fig. 2) in dataset $\mathcal{D}$ [6].

The IG is defined as

$$IG(\mathcal{D}_o, \psi^\dagger{}_\iota) = H(\mathcal{D}) - \sum_{e \in \{l,r\}} \frac{|\mathcal{D}^e|}{|\mathcal{D}|} H(\mathcal{D}^e) \tag{5}$$

where $H(\cdot)$ denotes Shannon's Entropy, which, in turn, defined thus

$$H(\mathcal{D}) = - \sum_{\forall g \in \{G^\dagger\}} p(g|\mathcal{D}) \log(p(g|\mathcal{D})) \tag{6}$$

where $p(g|\mathcal{D})$ represents the probability of a gesture with label g in dataset $\mathcal{D}$, and $G^\dagger$ a set of gesture labels, depending on the current layer related to a gesture type (e.g., static and dynamic). For example, label g at layer:top would be a label among $0, 1, 2, ..., N$, because $G^\dagger$ stands here for a set of multiple static gesture labels $\{0 : N\}$, and label g at layer:bottom would be a label among Pressing, Neutral, and Release, because $G^\dagger$ stands here for a set of dynamic gesture labels $\{P, N, R\}$.

c) Terminating the growing process: This growing process was repeated recursively on each split of the data, until it met the stopping condition. When the sample number of the dataset is less than the predefined minimum number (experimentally set as 10), the depth of the tree exceeds the predefined value (experimentally set as 20), or all the labels within a split subset assigned to the current node indicate an identical label among $\{P, N, R\}$.

2) Testing

As shown in Fig. 5(b), the proposed SD trees determine the optimal values of parameters to find the best split function at each split node by randomly testing the value of each parameter in the learning phase. Moreover, at the leaf node of an SD tree, both static and dynamic gesture probabilities are stored at the SD forest learning stage. Thus, using the given encoded LSP feature $\mathrm{E} = \{E_{[0:L]}\}$, the probabilities representing static gesture id and dynamic gesture status were retrieved from the proposed SD forest using

$$\alpha^*, \beta^* = \underset{\alpha \in \mathrm{A}, \beta \in \mathrm{B}}{\arg\max} P(Y_\alpha, Z_\beta|\mathrm{E}) \tag{7}$$

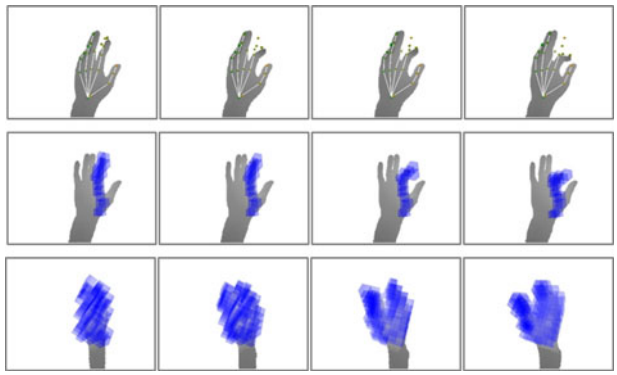

Fig. 6. Examples of the technical implementations. (First row) Three-dimensional hand posture estimation results show stable palm pose, while some partial finger joints are mislocated. (Second row) Layered shape changes while the finger bends. (Third row) Voxels represent the inclusion of the hand point clouds.

where $P(Y_\alpha)$ is the probability of a 3-D static gesture α, A being a set of static gestures defined, and $P(Z_\beta)$ is the probability of the status of dynamic gesture β, B being {Neutral, Pressing, Releasing}. The probabilities of static gesture Y_α and dynamic gesture Z_β were calculated by averaging the values stored at a leaf node of the trees in the SD forest as

$$P(Y_\alpha, Z_\beta|\mathrm{E}) = \frac{1}{T} \sum_{\forall \mathrm{tree} \in T} P_{\mathrm{tree}}(Y_\alpha, Z_\beta|\mathrm{E}). \tag{8}$$

IV. Implementation and Evaluation

A. Implementation

1) Static–Dynamic Voxel Configuration: In this paper, we configured an SD voxel feature within the local coordinates of the hand, based on the publicly available 3-D hand posture estimator [25] that provides a stable palm pose. To ensure that the palm pose estimator performs stably, we allocated a separate CPU thread for capturing images and conducting palm pose tracking, while configuring SD voxel features and estimating Meta-Gestures running in another thread. As shown in Fig. 6, Melax *et al.* [25] generally provides stable palm pose estimation of the hand, despite the mislocation of skeletal joints of the fingers, because of partially occluded fingers in egocentric viewpoint. The stable palm pose estimation results enable the SD voxel feature to be consistent so that the representative voxel planes are coherent, irrespective of the variance in viewpoint, as shown in Fig. 6.

2) Avoiding Side Effects of Using Time-of-Flight Camera: A time-of-flight camera provides a dense and smooth image, but the depth values it provides along the boundary of the hand outline are noisy. This noise adversely affects voxel feature configuration, because the feature is configured by counting the inclusions of the depth pixels. To avoid this, we eroded the hand region and utilized only the depth values within the eroded hand region.

3) Implementation and Experimental Environment: For experiencing a more immersive VR environment, the user was

requested to wear an HMD, with an attached camera, as shown in Fig. 1. We experimentally determined the position, in which the attached camera was aligned to the screen panel of HMD, because in this position, the HMD provides a more natural user experience for placing a hand location, while interacting with VR objects. We use Oculus Rift (Development Kit) [24] as the HMD and Intel's Creative Interactive Gesture Camera [25] as the depth camera. The VR environment, described in Section V, was implemented by Unity Engine [3]. As there are no conventional approaches using static and dynamic gestures together for manipulating a functional AR object, we show many plausible scenarios utilizing the proposed Meta-Gesture results in wearable AR/VR, as shown in Fig. 12.

B. Evaluation

Experiments were carried out to evaluate the effectiveness of the proposed approach in relation to the existing ones, using both static and dynamic gesture datasets. For static gesture, we used the publicly available source code of state-of-the-art static gesture (Chinese number gesture) recognition algorithm [22], [23], and for dynamic gesture, the 3-D Finger CAPE, the state-of-the-art dynamic clicking gesture estimation algorithm [14]. The focus of evaluation was on proving the performance of the 3-D static and dynamic gesture estimation, because proven performances straightforwardly guarantee the possibility of their later use for application scenarios (see the supplementary video available at: https://sites.google.com/site/fingergesture/fui/).

1) Static Gesture Estimation Results: The existing state-of-the-art algorithm [22], [23] for estimating static gesture, though based on the per-pixel classification of hand joint regions, still focuses on frontal-facing gesture recognition. Even though those approaches [22], [23] support small variations in terms of in-plane rotation, their estimation accuracies are easily degraded by variances in hand shape, size, partial finger movement, and rotation of the hand around a random axis. Thus, the static gesture dataset provided by those authors is inadequate to meet the main objective of the proposed approach, which is to be invariant to the viewpoint, and this may cause severe self-occlusions. Even though this is slightly different from what our framework focuses on (i.e., partial finger bending as well as the 3-D translation and 3-D rotation of hand), we collected two different datasets so that we can conduct a comparative study with the state-of-the-art method [22], [23] in frontal-facing viewpoint (by using Intel's Creative Interactive Gesture Camera [25]).

a) Dataset: The two static gesture datasets consist of ten-digit Chinese number gestures, varied with in-plane rotation. These gestures were varied with 3-D translation and rotation around a random axis. Each gesture comprised 1500 frames. The total number of different viewpoint-based static gesture datasets is $15\,000$. The palm pose tracking, however, which is based on depth image, does not always provide consistent results, in terms of position and orientation tracking, irrespective of variance in viewpoint, because the utilized tracking method [25] behaves like a magnet, which provides the hand position and orientation results that are stuck to the surface of the hand.

Thus, when the depth image is captured in frontal viewpoint with [25], the rectangular cube surrounding the hand region comes onto the palm. In egocentric viewpoint, on the other hand, the rectangular cube surrounds the back of the hand. To ensure that the dataset is more practical and covers a larger variety of viewpoints, we applied a perturbation factor of $\pm 5^{\circ}$ and ± 1 cm along each axis. By doing so, we could collect six times the usual number of samples for each gesture, which amounted to a total of $90\,000$ samples in the final dataset. We utilized half of these samples for training and the other half for testing.

We used our own datasets for evaluating the performance on static gesture estimation in various viewpoints. Because our proposed method is based on the palm pose estimation results of [25], our dataset also provides the tracking results, corresponding to the captured image. To make our evaluation results publicly available for reference,[2] we prepared a document of our dataset and thus formatted it: frame number, color image file path, depth image file path, IR image file path, 16 element values of a transformation matrix representing the palm pose, and gesture label.

Number of voxel selection: First, we determined the saturation point of accuracy by trying different numbers of voxels in the ST forest to get optimal performance with minimal number of voxels. As shown in Fig. 7(a), we confirmed that the static gesture estimation accuracies were saturated when the number of voxels exceeded 196 $(7 \times 7 \times 4)$. Thus, we set that 196 voxels are adequate to configure a voxel feature for generating an encoded LSP. This finding was used for the remaining part of the static gesture experiments in this paper. For the stage of splitting voxels, we fixed the size of the rectangular cube that surrounds the hand as 20 cm $\times$ 20 cm $\times$ 12 cm. Moreover, we aligned the cube to the center of the palm. Thus, it is possible to obtain a finer and accurate (91.36% accuracy) representation of the hand, as shown in Fig. 7(c), by increasing the number of voxels than by utilizing coarse voxel splits, as shown in Fig. 7(b) (75.80%).

The aim of this experiment is to make a comparative study of the performances of the classifiers, as shown in Fig. 7. For 3-D static gesture estimation, we determined the gesture by selecting the gesture id that has the highest probability among the ten-digit gesture set values. In static gesture dataset, captured in frontal viewpoint, the accuracy of our method (97.80%) is more or less the same as that of Liang's method (84.70%) [22], [23], as shown in Fig. 7(d). However, in the dataset of the same gestures, captured in various viewpoints, our method gives more accurate results (91.36%) than Liang's method (60.00%) [22], [23], as shown in Fig. 7(e). As expected, the errors in per-pixel-level hand joint classification, which are caused by variances in hand shape, size, and viewpoints, adversely affect the ability to define and estimate gestures.

2) Dynamic Gesture Estimation Results: For dynamic gesture estimation, we conducted a comparative study with 3-D Finger CAPE [14], the most relevant previous work that tackles the self-occlusion problem in various viewpoints. Its dynamic gesture estimation is based on a dataset of two sta-

[2]Project page: https://sites.google.com/site/fingergesture/fui/

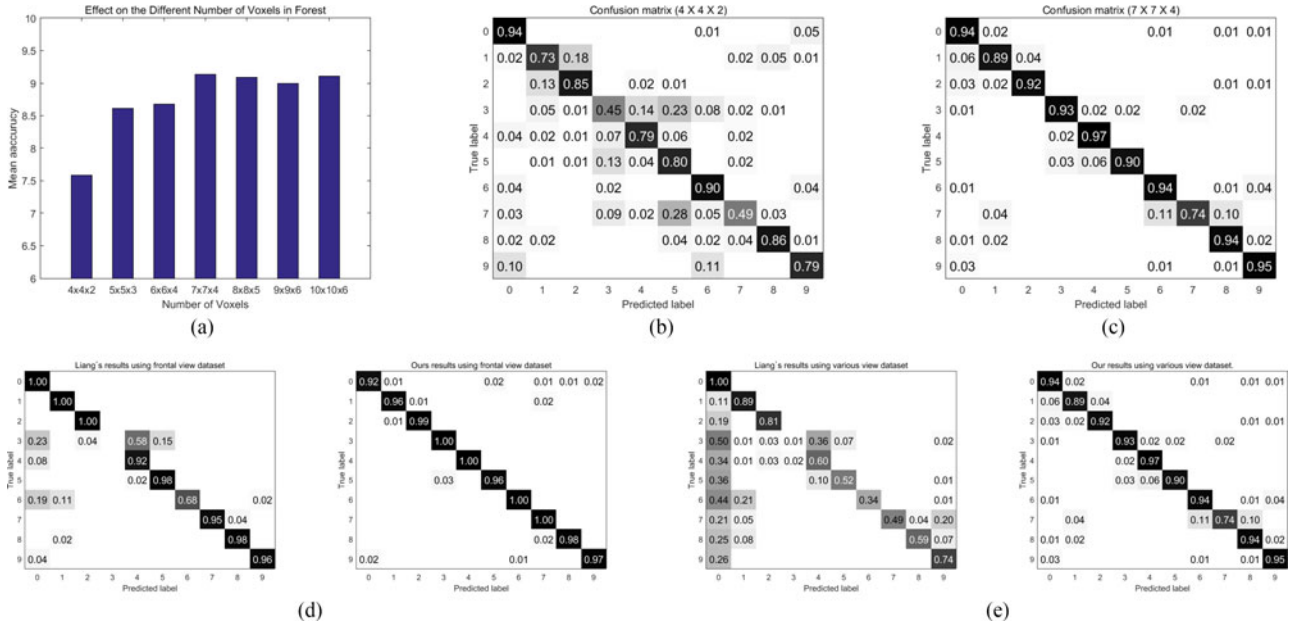

Fig. 7. Experimental results showing the effect of utilizing different voxel counts and a confusion matrix of ten-digit static gestures from the dataset captured at various viewpoints. (a) Static gesture estimation accuracies are saturated when voxel counts exceed 196 ($7 \times 7 \times 4$). (b) When voxel count is at 32 ($4 \times 4 \times 2$), estimation rates are at 75.80%. (c) When voxel count is at 196 ($7 \times 7 \times 4$), estimation rates are at 91.36%; (d) (left) Liang's results (84.70%, 94.70%, excluding bad cases [gesture 3]) [22], [23], and (right) our results (97.80%, 97.50%, excluding bad cases [gesture 3]) using a frontal viewpoint dataset. (e) (left) Liang's results (60.00%) [22], [23] and (right) our results (91.36%) using a dataset from various viewpoints.

tuses: pressing and neutral. In this paper, we made use of this dataset (see [14] for dataset description) for the purpose of comparison, because it is the only existing dataset that tackles dynamic gesture estimation under self-occlusions in egocentric viewpoint.

For training, we utilized only 20% of the dataset (580 actions, including 290 positive and negative samples); yet, interestingly, we achieved better performance in classification, based on our encoded LSP feature and SD forest. For testing, we utilized all the ten existing sequences (360 actions in total). In contrast to 3-D Finger CAPE [14], at training, our method does not require any specific voxel regions to separate a specific finger motion estimation, while ST forest [14] requires finger specification for training a specific finger motion. Thus, we can say that the proposed SD forest, together with SD voxel feature, is a more generalized method for training static and dynamic gestures, irrespective of finger index.

The receiver operating characteristic (ROC) curves, presented in Fig. 8(c), show that the proposed SD forest-based clicking action detection (93.87%) works as accurately as the 3-D Finger CAPE (93.91%) [14] for the use of multiple fingers. Moreover, interestingly, the proposed framework, utilizing SD voxel feature, provides more accurate (95.59%) results than those of 3-D Finger CAPE method [14] (89.80%) [see Fig. 8(a)], especially for detection by thumb-based clicking action. This is because the proposed SD voxel feature does not propagate the noise sources that occur in the feature extraction stage before defining a gesture, as the SD voxel feature is configured, based only on the positions of hand point clouds. Unlike our SD voxel feature, the 3-D Finger CAPE method [14] depends on the combination of 3-D skeletal joint data and 2-D fingertip data. Thus, thexbrk failure of 3-D skeletal joint extraction and 2-D fingertip detection affects the accuracy of detection by clicking action, while our method is independent of that kind of noise.

The accuracies of clicking action detection by SD forest, utilizing thumb, index finger, middle finger, ring finger, and pinky are 95.59%, 96.55%, 91.59%, 91.74%, and 93.92%, respectively. For Meta-Gesture, dynamic gesture estimation for the thumb and index finger is particularly important among the multiple fingers. In this paper, our problem domain is in the combination of static gesture (for summoning a functional AR object) and dynamic gesture (for action-based input). In many scenarios, the most commonly used fingers, while utilizing our 3-D hand gesture estimator, for making a dynamic gesture as that of holding a tool (for static gesture) are the thumb (e.g., holding a mobile phone and moving the thumb to press a button) and the index finger (e.g., holding a stylus pen and pressing button on the pen stalk). According to our Meta-Gesture scenarios, the superior estimations are meaningful, especially for the thumb and the index finger, as shown in Fig. 8(a) and (b).

The average processing time of the proposed framework was 21.02 ms (47.57 frames/s) in the environment of our experiment, which ran on an Intel Core-i7 2600 CPU processor with 8 GB of DRAM.

C. Evaluation via User Test

1) Subjects: We conducted the qualitative experiments with 12 graduate students (aged 24–29 years) with a mean age of 27.5 years. All the participants were aware of the concept of object selection and manipulation, and half of them had the experience of using an AR/VR application. The experiments

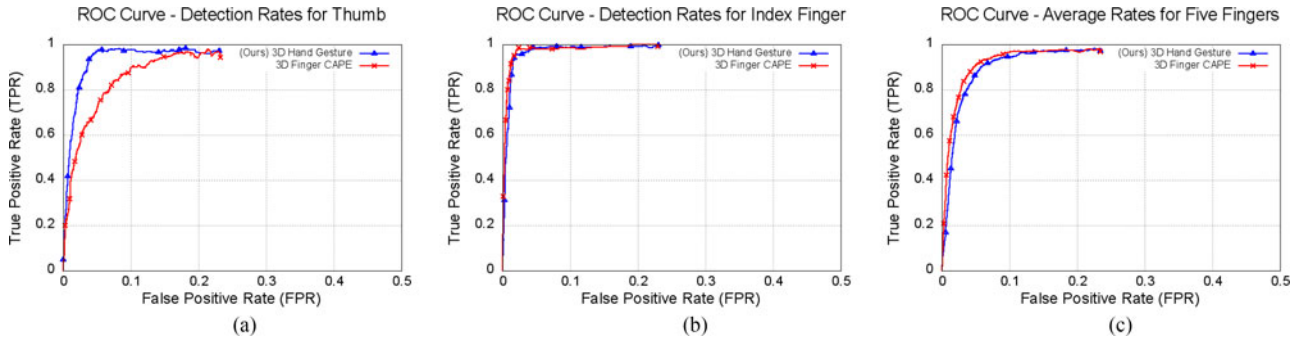

Fig. 8. ROC curves of clicking action classification comparing the proposed Meta-Gesture (our method) with 3-D Finger CAPE [14]. (a) thumb (Our method: 95.59%, 3-D Finger CAPE: 89.80%). (b) Index finger (Our method: 96.55%, 3-D Finger CAPE: 96.90%) . (c) ROC curves representing average accuracies for all five fingers (Our method: 93.87%, 3-D Finger CAPE: 93.91%).

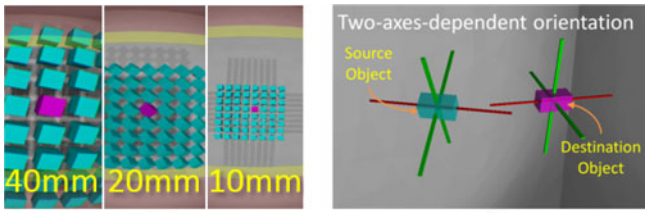

Fig. 9. VR environment setting. (a) Three different object sizes (40 mm, 20 mm, 10 mm) configuring a stage of selection task. (b) Example of a stage of manipulation task.

took approximately 45 min per person, including the presession training, and testing.

2) Experimental Design and Procedure: First, we compare the performances of selecting objects of different-sizes, based on three methods: the Grab-based method [10], 3-D Finger CAPE [14], and Meta-Gesture (Authors' method). For selecting an object, based on the Grab-based method, the user was requested to match the palm center position with the target object and then make a grabbing gesture for triggering the selection function [10]. The user was requested to make a clicking gesture by bending the index finger [14] to utilize 3-D Finger CAPE, and to make a combination of two SD gestures to utilize Meta-Gesture. Making a static gesture can summon a tool into the palm of the hand. Only when the tip of the summoned virtual object is matched with the target object, then the user conducts a dynamic gesture to trigger the selection function of the summoned object.

For the selection tasks, the user has a total of 216 stages, corresponding to the three methods (Grab, CAPE, and Meta-Gesture), multiplied successively by four scenarios [14], three object sizes (40 mm, 20 mm, 10 mm) [see Fig. 9(a)] and six stages. The three methods were independently activated. Using each method, the user was requested to select three objects of different sizes in four different scenarios: static-sparse, dynamic-sparse, static-dense, and dynamic-dense environments, as described in 3-D Finger CAPE [14] (see [14] for a detailed explanation for the environmental setting of the selection task). Each scenario had six stages in which an object was to be selected. At each stage, the users were requested to select a randomly positioned target object. No matter which of the three activating methods was used, the stage was considered passed when the distance between the selected position and the target object was less than 40 mm. Even if the stage did not pass, all the trials and the distances between the selected positions and the target objects were recorded for subsequent analysis.

The second task was evaluation of the manipulation performance of the proposed Meta-Gesture. As there are no previous studies on orientation-aware manipulation, utilizing bare hand, we compared the ground-truth performance (utilizing an AR multimarker-based tangible interface, called wand-based method) and the results obtained by the proposed method of utilizing Meta-Gesture.

For this manipulation task, the user was requested to select a source object and manipulate it in such a way that it matches the position and orientation of the destination object, as shown in Fig. 9(b). Each object was shaped like a cuboid (40 mm × 20 mm × 20 mm) and the visualized axes were laid over the objects, as shown in Fig. 9(b). Exceptionally, for this manipulation task, the sizes of the source and destination objects were slightly larger than nail-size, because of which some users could not perceive the object's orientation. This manipulation task included ten stages. When the distance between the source and the destination objects was less than 30 mm, and the difference between the angle values of the two axes was under 30°, the user was passed to the next stage. In addition, we recorded every tried position and orientation error at every trial and the time taken to complete the manipulation of the object at each stage for subsequent analysis.

Before testing the two tasks of selection and manipulation, the user was guided (supervised), through a 10-min training session, in getting accustomed to the VR environment and the three methods.

3) Statistical Analysis: Before analyzing the data, the outliers, recorded by the detection error of false movement, were removed, based on the distance threshold, where the calculated distance was more than the predefined threshold. For our experiment, we set the threshold to 80 mm.

We plotted the points, positioned relative to the position of the target object (center of the grid) (see Fig. 10), based on the data obtained from the three methods (red points: Grab-based method [10], green points: 3-D Finger CAPE [14], and blue points: Meta-Gesture). Each subfigure corresponds to 20 mm (fingertip-sized) and 10 mm (nail-sized) object selection

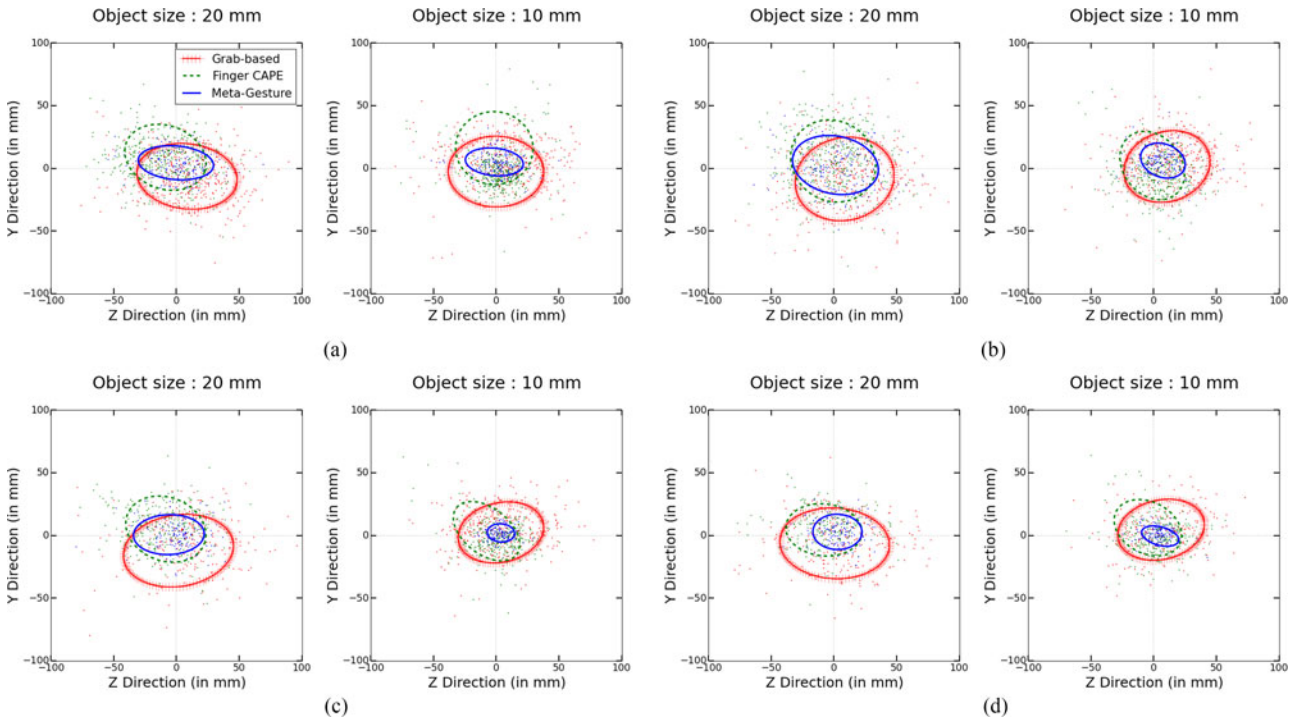

Fig. 10. Results of the distance errors between the target position (0, 0, 0) coordinates and the estimated position, based on Grab-based method (red dots) [10], the Finger CAPE (green dots) [14], and the proposed Meta-Gesture (blue dots), plotted within the camera coordinates, gathered from four different scenarios. (Best viewed in color). (a) Scenario 1: Sparse and static objects. (b) Scenario 2: Sparse and dynamic objects. (c) Scenario 3: Dense and static objects. (d) Scenario 4: Dense and dynamic objects.

sessions. We excluded the results of 40 mm object selection session, whose results can be predicted, based on the results of the 20 mm object selection session.

Fig. 11(a) shows the distance errors in selection tasks of different sessions, each session including four different environments and three different object sizes. The results show that the proposed Meta-Gesture stably selects the objects of all sizes, including those of even nail-size (10 mm), while the other methods fail to select smaller objects. This is because Meta-Gesture indirectly points the target object by utilizing a summoned object in the hand, whereas the other methods try to directly point the target object with the occluded fingertip estimation results. As already discussed, the summoned objects were stably augmented even where one or two fingers were totally occluded, because the augmentation was based on the tracking results of the whole hand region.

In the scenarios of sparsely placed objects, the results of 3-D Finger CAPE are not as good as those of the Grab-based method, because 3-D Finger CAPE, which has no guideline, is affected by human depth perception, as described in [14]. However, it can provide better results in environments of densely placed objects, in which the user can perceive the depth of the object correctly, based on their structure, as shown in Fig. 10(c).

Moreover, interestingly, the selection performances of 3-D Finger CAPE and Meta-Gesture in the environment of sparsely placed objects environment are worse than those in the environment of densely placed objects, as shown in Fig. 11(a). This is because, the randomly generated stages in the sparse environment include farther target objects that cause failure of hand tracking. Based on these results, we confirm that the 3-D Finger CAPE and Meta-Gesture can stably perform only in those environments that are within the reach of the arm.

For manipulation tasks, we also analyzed the orientation-aware displacement errors and the stage completion times, based on the recorded data. The marker-based tangible wand serves as the ground truth, because a vision-based system tracks the marker stably. Based on the foregoing results, we claim that our proposed Meta-Gesture can give stable results similar to those of the ground truth. The wand-based ground-truth method (13.36 s) and Meta-Gesture (15.27 s) show similar performance in terms of manipulation time.

V. Interaction Techniques and Head-Mounted-Display Use Cases

This section presents several compelling interaction scenarios, enabled by our technique of using HMD. For each scenario, we implemented a sample to highlight how our technique can be utilized to make the interaction not only possible, but also faster and easier. We could achieve this by leveraging the fact that an average person can skillfully manipulate various functional tools, such as scissors and stylus, because of his or her experience in using them during his or her lifetime. Thus, we could design and build a rich set of gestures for summoning interactable tools in AR/VR environment. In particular, our system can recognize the user's partial static hand gestures to instantiate the virtual tool, as if it was being grasped. The user can see the application in VR through HMD.

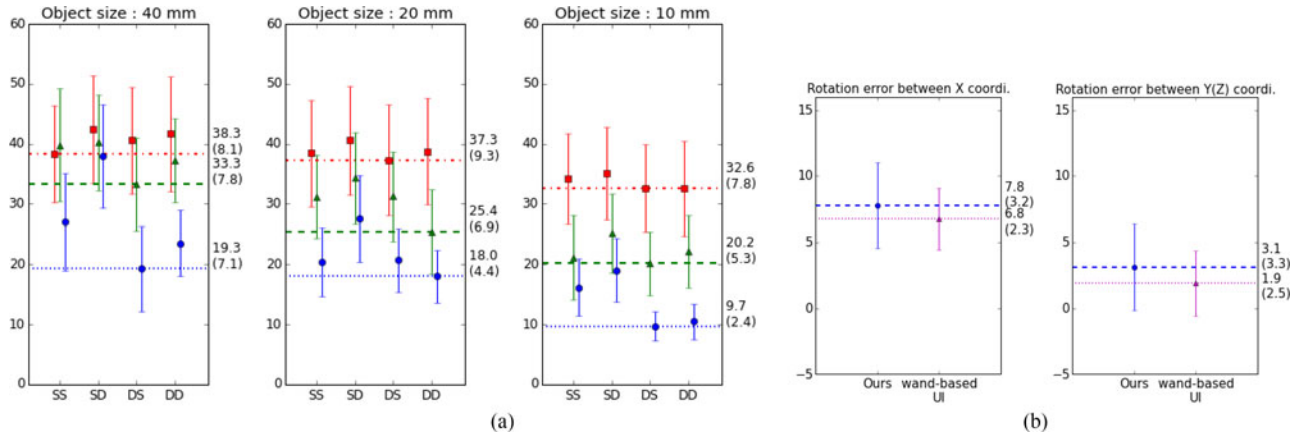

Fig. 11. Mean and standard deviation of measured error in orientation-aware selection (unit: mm) and manipulation (unit: degree) tasks. (a) Distance error between the target object and the estimated position, based on different methods (Grab-based method: red square dots [10], 3-D Finger CAPE: green triangle dots [14], and Meta-Gesture: blue circle dots) and in different scenarios (SS: sparse-static, SD: sparse-dynamic, DS: dense-static, and DD: dense-dynamic environments). (b) Angle differences between the orientations of the destination and the manipulated objects, based on Meta-Gesture (blue circle dots) and the wand-based ground-truth method (purple triangle dots).

A. *Complex Three-Dimensional Model Assembling Application*

We envision several scenarios wherein the users can easily perform different complex tasks, such as assembling machines or LEGO models by using our system [see Fig. 12(a)]. In such tasks, the components of different shapes and sizes need to be manipulated independently. By summoning different tools on to the hand, the user can easily manipulate them through translation, rotation, and other types of motions, as if they were using the physical counterparts. For example, a pair of pliers can be used for grabbing a small screw and putting it in position and a screwdriver for securing it. Thus, our approach provides fast and fluid mode-switching operations, without relying on external toggles or toolbars, which are considered more cumbersome, especially in interactive environments involving hand gestures.

B. *In-Air Drawing and Annotation Application*

With static gesture, it might be easy to overlay a virtual pen on the hand; however, there is no explicit signal/delimiter that differentiates between writing and nonwriting actions. Previous systems suffer from this limitation, which only allows for drawing a continuous line without separation between letters. With our hybrid approach, we can first summon a pen and then; with our dynamic gesture status recognition, we can activate the writing action, seemingly lifting the pen and thus avoiding the aforementioned continuous writing limitation [see Fig. 12(b)].

C. *Camera Application*

The possibility of using symbolic gestures for taking a photo has been explored in previous systems, such as in [8] and [26]. In contrast, our Meta-Gesture allows the users to first utilize a static gesture for pointing at a nice view, and then, once the user is satisfied, a dynamic gesture is performed to activate the camera shutter [see Fig. 12(c)]. We argue that, in this way, the users can hold their hand stably and then press the camera shutter. When only static gestures are used, the timing would be immature and the pictures blurred, because of unstable hands.

D. *Gesture-Based First Person Immersive Game*

Typical feature-rich games involve complex interactions, such as mode-switching or invoking different actions, which are typically mapped to different hot keys. We argue that mapping hand gestures for these tasks is more intuitive and faster, which in turn leads to a more immersive and enjoyable gaming experience. Several such examples are shown in Fig. 12(d), where different weapons (pistol, light-saber) can be switched instantly by just changing the static gesture, whereas actions such as shooting a bullet or activating a light-saber can be triggered by a dynamic finger gesture.

VI. Discussion

While most random forests learn a single task, our proposed SD forest learns hierarchically two tasks in a single classifier: static gesture (first hierarchy) and dynamic gesture (second hierarchy). Because only half of the dataset remains in the tree after a split, SD forest requires a rather large dataset for learning the tasks, but it provides computationally efficient results in testing time.

We have presented a novel view-independent feature representation and confirmed that the feature helps in providing better accuracies than those of the works that depend on joint detection/tracking. However, the SD forest proposed by us is still dependent on the publicly available palm pose estimator for setting the cuboid region, enclosing the 3-D hand region. Based on the learning process with the perturbed dataset, we could somehow overcome even coarse palm pose estimation results. However, when the palm pose was largely missestimated, then we could not encode the LSP. If we utilize a more stable palm pose estimator, then it would provide more stable results.

Overall, hand gesture recognition in egocentric viewpoint is still a challenging issue. More specifically, visual recognition

Fig. 12. Examples of using Meta-Gesture. (a) Elaborate AR object manufacturing (assembling) scenario showing selection of a thin and small component in a cluttered scene. (b) In-air pen annotation in 3-D space with split letters. (c) AR camera application taking a photograph of a VR scene. (d) In the immersive gaming, the user can summon different types of weapons, such as an AR gun or an AR sword, intuitively.

that depends on captured images fails easily when there is heavy occlusion, especially if the recognition task aims at multi-class classification. Because of this reason, our AR gun scenario, shown in Fig. 12(d), might fail when the static gun-grasping gesture hand stays a little longer in the perpendicular direction to the camera plane. In addition, sometimes, our proposed framework misclassified the static gestures when a static gesture (e.g., spray) was changed into another static gesture (e.g., gun). We call it an intergesture problem, and it can be overcome by considering sequential consistency and confidence level, based on a probability.

Although the dynamic gestures of our scenarios show only one-finger movements, it does not mean that the proposed method supports only such movements. Because our LSP pattern does not depend on a specified index of finger, the LSP value can be calculated, no matter how many fingers are moving in any given direction (for example, moving the fingers together and apart is possible and could be defined and estimated as a dynamic feature).

VII. Conclusion

This paper has presented a novel interface framework, called Meta-Gesture, which can simultaneously conduct static and dynamic gesture estimations, in various 3-D functional object manipulation scenarios, in wearable AR/VR. A major problem that arises from the user's free-hand environment, while wearing HMD, is partial self-occlusion (hidden) of fingers. Experimental results obtained by using our proposed LSP feature and SD forest demonstrate that, although they are similar to the methods of previous research, our approach of using a combination of voxelization of local cubes surrounding a hand, together with the help of 3-D palm pose estimation from noisy and occluded data, gives better accuracy and speed than the state-of-the-art methods. Experimental results show that SD forest gives higher accuracy in both 3-D static gesture (91.36%) and dynamic gesture (93.87%) estimation than that given by the state-of-the-art methods.

In static gesture, the results provided by the proposed SD forest are more stable than those provided by the previous method, because its estimation can get degraded when affected by the variance in hand shape (provoking per-pixel classification error). The same is the case with dynamic gesture, because the estimation by the previous method can similarly get degraded, when affected by unstable skeletal tracking results, as it heavily relies on partial finger and joint estimation. Moreover, we found that the proposed Meta-Gesture encourages more intuitive interactions in the scenarios of summoned function-equipped AR object manipulation, which are independent of rotation and translation of hand and are directly extendible into AR environment. For our future work, we plan to extend this work to design a more enhanced and augmented NUI and utilize real objects and different combinations of holding and finger gestures. Besides, we also envisage the transition of static and dynamic gesture estimations from function-equipped AR objects to objects that can give additional functionalities to real objects.

References

[1] Microsoft HoloLens. [Online]. Available: https://www.microsoft.com/microsoft-hololens/. Accessed on: Sep. 25, 2015.
[2] Oculus VR. [Online]. Available: https://www.oculus.com/. Accessed on: Sep. 25, 2015.
[3] Unity engine. [Online]. Available: http://unity3d.com/. Accessed on: May 4, 2015.
[4] M. Achibet, M. Marchal, F. Argelaguet, and A. Lécuyer, "The virtual mitten: A novel interaction paradigm for visuo-haptic manipulation of objects using grip force," in *Proc. IEEE Symp. 3D User Interfaces*, Minneapolis, MN, USA, Mar. 29–30, 2014, pp. 59–66.
[5] L. Baraldi, F. Paci, G. Serra, L. Benini, and R. Cucchiara, "Gesture recognition in ego-centric videos using dense trajectories and hand segmentation," in *Proc. IEEE Conf. Comput. Vision. Pattern Recognit. Workshops*, Jun. 2014, pp. 702–707.
[6] L. Breiman, "Random forests," *J. Mach. Learn.*, vol. 45, pp. 5–32, Oct. 2001.
[7] W. H. Chun and T. Höllerer, "Real-time hand interaction for augmented reality on mobile phones," in *Proc. Int. Conf. Intell. User Interfaces*, New York, NY, USA, 2013, pp. 307–314.
[8] A. Colaço, A. Kirmani, H. S. Yang, N.-W. Gong, C. Schmandt, and V. K. Goyal, "Mime: Compact, low power 3D gesture sensing for interaction with head mounted displays," in *Proc. 26th Annu. ACM Symp. User Interface Softw. Technol.*, New York, NY, USA, 2013, pp. 227–236.
[9] C. Corsten, I. Avellino, M. Möllers, and J. Borchers, "Instant user interfaces: Repurposing everyday objects as input devices," in *Proc. ACM Int. Conf. Interactive Tabletops Surfaces*, New York, NY, USA, 2013, pp. 71–80.

[10] T. Ha, S. Feiner, and W. Woo, "Wearhand: Head-worn, RGB-D camera-based, bare-hand user interface with visually enhanced depth perception," in *Proc. IEEE Int. Symp. Mixed Augmented Reality*, Sep. 2014, pp. 219–228.
[11] G. Hackenberg, R. McCall, and W. Broll, "Lightweight palm and finger tracking for real-time 3D gesture control," in *Proc. IEEE Virtual Reality Conf.*, Mar. 2011, pp. 19–26.
[12] C. Harrison, R. Xiao, J. Schwarz, and S. E. Hudson, "Touchtools: Leveraging familiarity and skill with physical tools to augment touch interaction," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, New York, NY, USA, 2014, pp. 2913–2916.
[13] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Efficient model-based 3D tracking of hand articulations using kinect," in *Proc. Brit. Mach. Vision Conf.*, 2011, pp. 101.1–101.11.
[14] Y. Jang, S.-T. Noh, H. J. Chang, T.-K. Kim, and W. Woo, "3D Finger CAPE: Clicking action and position estimation under self-occlusions in egocentric viewpoint," *IEEE Trans. Vis. Comput. Graph.*, vol. 21, no. 4, pp. 501–510, Apr. 2015.
[15] C. Kerdvibulvech and H. Saito, "Markerless guitarist fingertip detection using a Bayesian classifier and a template matching for supporting guitarists," in *Proc. 10th ACM/IEEE Virtual Reality Int. Conf.*, Apr. 2008, pp. 201–208.
[16] C. Keskin, F. Kirac, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *Proc. IEEE Int. Conf. Comput. Vision Workshops*, 2011, pp. 1228–1234.
[17] C. Keskin, F. Kirac, Y. E. Kara, and L. Akarun, "Hand pose estimation and hand shape classification using multi-layered randomized decision forests," in *Proc. 12th Eur. Conf. Comput. Vision*, 2012, pp. 852–863.
[18] D. Kim *et al.*, "Digits: Freehand 3D interactions anywhere using a wrist-worn gloveless sensor," in *Proc. 25th Annu. ACM Symp. User Interface Softw. Technol.*, 2012, pp. 167–176.
[19] D. Kim *et al.*, "Retrodepth: 3D silhouette sensing for high-precision input on and above physical surfaces," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, New York, NY, USA, 2014, pp. 1377–1386.
[20] P. Krejov and R. Bowden, "Multi-touchless: Real-time fingertip detection and tracking using geodesic maxima," in *Proc. 10th IEEE Int. Conf. Workshops Autom. Face Gesture Recognit.*, 2013, pp. 1–7.
[21] T. Lee and T. Höllerer, "Multithreaded hybrid feature tracking for markerless augmented reality," *IEEE Trans. Vis. Comput. Graph.*, vol. 15, no. 3, pp. 355–368, May/Jun. 2009.
[22] H. Liang and J. Yuan, "Hand parsing and gesture recognition with a commodity depth camera," in *Computer Vision and Machine Learning with RGB-D Sensors*. New York, NY, USA: Springer, 2014, pp. 239–265.
[23] H. Liang, J. Yuan, and D. Thalmann, "Parsing the hand in depth images," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1241–1253, Aug. 2014.
[24] P. Lubos, G. Bruder, and F. Steinicke, "Analysis of direct selection in head-mounted display environments," in *Proc. IEEE Symp. 3D User Interfaces*, 2014, pp. 11–18.
[25] S. Melax, L. Keselman, and S. Orsten, "Dynamics based 3D skeletal hand tracking," in *Proc. Graph. Interface Conf.*, 2013, pp. 63–70.
[26] P. Mistry and P. Maes, "Sixthsense: A wearable gestural interface," in *Proc. SIGGRAPH ASIA*, New York, NY, USA, 2009, Art. no. 11.
[27] P. Molchanov, S. Gupta, K. Kim, and K. Pulli, "Multi-sensor system for driver's hand-gesture recognition," in *Proc. 11th IEEE Int. Workshops Autom. Face Gesture Recognit.*, 2015, pp. 1–8.
[28] N. Bogdan, T. Grossman, and G. Fitzmaurice, "Hybridspace: Integrating 3D freehand input and stereo viewing into traditional desktop applications," in *Proc. 3D IEEE Symp. User Interfaces*, 2014, pp. 51–58.
[29] M. Ogata, Y. Sugiura, H. Osawa, and M. Imai, "Iring: Intelligent ring using infrared reflection," in *Proc. 25th Annu. ACM Symp. User Interface Softw. Technol.*, New York, NY, USA, 2012, pp. 131–136.
[30] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints," in *Proc. Int. Conf. Comput. Vision*, 2011, pp. 2088–2095.
[31] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Tracking the articulated motion of two strongly interacting hands," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 1862–1869.
[32] Z. Pan *et al.*, "A real-time multi-cue hand tracking algorithm based on computer vision," in *Proc. IEEE Virtual Reality Conf.*, Washington, DC, USA, 2010, pp. 219–222.
[33] N. Petersen, A. Pagani, and D. Stricker, "Real-time modeling and tracking manual workflows from first-person vision," in *Proc. IEEE Int. Symp.Mixed Augmented Reality*, Oct. 2013, pp. 117–124.
[34] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 1–54, Jan. 2015.
[35] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," *IEEE Trans. Multimedia*, vol. 15, no. 5, pp. 1110–1120, Aug. 2013.
[36] A. R. Sarkar, G. Sanyal, and S. Majumder, "Hand gesture recognition systems: A survey," *Int. J. Comput. Appl.*, vol. 71, no. 15, pp. 25–37, May 2013.
[37] G. Serra, M. Camurri, L. Baraldi, M. Benedetti, and R. Cucchiara, "Hand segmentation for gesture recognition in ego-vision," in *Proc. 3rd ACM Int. Workshop Interactive Multimedia Mobile Portable Devices*, New York, NY, USA, 2013, pp. 31–36.
[38] T. Sharp *et al.*, "Accurate, robust, and flexible real-time hand tracking," *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, Apr. 2015, pp. 3633–3642.
[39] Y. Shen, S. K. Ong, and A. Y. C. Nee, "Vision-based hand interaction in augmented reality environment," *Int. J. Human-Comput. Interact.*, vol. 27, no. 6, pp. 523–544, 2011.
[40] J. Song, F. Pece, G. Sörös, M. Koelle, and O. Hilliges, "Joint estimation of 3D hand position and gestures from monocular video for mobile interaction," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, New York, NY, USA, 2015, pp. 3657–3660.
[41] J. Song *et al.*, "In-air gestures around unmodified mobile devices," in *Proc. 27th Annu. ACM Symp. User Interface Softw. Technol.*, New York, NY, USA, 2014, pp. 319–329.
[42] S. Sridhar, A. Oulasvirta, and C. Theobalt, "Interactive markerless articulated hand motion tracking using RGB and depth data," in *Proc. Int. Conf. Comput. Vision*, Dec. 2013, pp. 2456–2463.
[43] D. Tang, H. J. Chang, A. Tejani, and T.-K. Kim, "Latent regression forest: Structured estimation of 3D articulated hand posture," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Jun. 2014, pp. 3786–3793.
[44] D. Tang, T.-H. Yu, and T.-K. Kim, "Real-time articulated hand pose estimation using semi-supervised transductive regression forests," in *Proc. IEEE Int. Conf. Comput. Vision*, Dec. 2013, pp. 3224–3231.
[45] S. Taylor, C. Keskin, O. Hilliges, S. Izadi, and J. Helmes, "Type-hover-swipe in 96 bytes: A motion sensing mechanical keyboard," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, New York, NY, USA, 2014, pp. 1695–1704.
[46] R. Wang, S. Paris, and J. Popović, "6D hands: Markerless hand-tracking for computer aided design," in *Proc. 24th Annu. ACM Symp. User Interface Softw. Technol.*, New York, NY, USA, 2011, pp. 549–558.
[47] A. Zerroug, A. Cassinelli, and M. Ishikawa, "Invoked computing: Spatial audio and video AR invoked through miming," in *Proc. Virtual Reality Int. Conf.*, Apr. 2011, pp. 31–32.

Youngkyoon Jang (S'15–M'15) received the Ph.D. degree from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2015.

He is a Postdoctoral Research Assistant with Queen Mary University of London, London, U.K., and has been a Visiting Researcher with the University of Cambridge, Cambridge, U.K., since 2016. His research interests include novel natural user interface technologies that aim to overcome challenges in interactions between humans and computers in a wearable augmented reality/virtual reality environment, specializing in understanding human affect/behaviors, understanding scene, and identifying users.

Dr. Jang received the Best Poster Award at the IEEE Computer Society Workshop on Observing and Understanding Hands in Action (in conjunction with the IEEE Conference on Computer Vision and Pattern Recognition) in 2016.

Ikbeom Jeon received the B.S. degree in media engineering from Soongsil University, Seoul, South Korea, in 2014. He is currently working toward the M.S. degree in the Graduate School of Culture Technology, Korea Advanced Institute of Science and Technology, Daejeon, South Korea.

His research interests include augmented reality, computer vision, machine learning, and graphics, with particular emphasis on systems that allow people to interact naturally with computers.

Tae-Kyun Kim (S'07–M'11) received the Ph.D. degree from the University of Cambridge, Cambridge, U.K., in 2008.

He has been an Associate Professor and Leader of the Computer Vision and Learning Laboratory, Imperial College London, London, U.K., since 2010. On the topics of hand pose, face recognition by image sets, 6-D object pose, active robot vision, he has published more than 40 top-tier journal and conference papers that have been highly cited. His coauthored algorithm is an international standard of MPEG-7 ISO/IEC for face image retrieval.

Dr. Kim is the Co-Chair of the IEEE Computer Society Workshop on Observing and Understanding Hands in Action (in conjunction with the IEEE Conference on Computer Vision and Pattern Recognition) and International Conference on Computer Vision/European Conference on Computer Vision Object Pose Workshops. He is an Associate Editor of *Image and Vision Computing*. He was the co-recipient of the KUKA Paper Award at the 2014 IEEE International Conference on Robotics and Automation.

Woontack Woo (M'93) received the Ph.D. degree from the University of Southern California, Los Angeles (USC), CA, USA in 1998.

He is a Professor with the Graduate School of Culture Technology, Korea Advanced Institute of Science and Technology, Daejeon, South Korea. In 1999, as an invited Researcher, he joined Advanced Telecommunications Research, Kyoto, Japan. From February 2001 to February 2012, he was a Professor with the School of Information and Communications and the Director of the Culture Technology Institute, Gwangju Institute of Science and Technology, Gwangju, South Korea. The main thrust of his research has been implementing ubiquitous virtual reality in smart space, which includes context-aware augmented reality, 3-D vision, human–computer interaction, and culture technology.