2013 IEEE International Conference on Robotics and Automation (ICRA)
Karlsruhe, Germany, May 6-10, 2013

A 3D-Point-Cloud Feature for Human-Pose Estimation

Kai-Chi Chan, Cheng-Kok Koh and C. S. George Lee
School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, U.S.A.
Email: {chan56, chengkok, csglee}@purdue.edu.

Abstract— **Estimating human poses is an important step towards developing robots that can understand human motions and improving their cognitive capabilities. This paper presents a geometric feature for estimating human poses from a 3D point cloud input. The proposed feature can be considered as an extension of the idea of visual features, such as color/edge, of color/grayscale images, and it contains the geometric structure of the point cloud. It is derived by arranging the 3D points into a tree structure, which preserves the global and local properties of the 3D points. Shown experimentally, the tree structure (spatial ordering) is particularly important for estimating human poses (i.e., articulated objects). The 3D orientation (pan, tilt and yaw angles) and shape features are then extracted from each node in the tree to describe the geometric distribution of the 3D points. The proposed feature has been evaluated on a benchmark dataset and compared with two existing geometric features. Experimental results show that the proposed feature has the lowest overall error in human-pose estimation.**

I. INTRODUCTION

Human-pose estimation is essential for robots to develop their cognitive capabilities for human-motion analysis, human-human/robot interaction, gait recognition, and fall detection. The two main components in human-pose estimation are pose initialization and pose tracking. In pose initialization, a human is first detected in the scene and joints are fitted under predefined criteria; pose is then tracked by matching the joint features over time.

The main difficulties of human-pose estimation are the high dimensionality of pose space and feature ambiguity. As a human is highly articulated, modeling the poses directly is intractable. Simply assuming independence among joints, however, could not reflect the reality, and hence results in errors in modeling. Prior probability of poses will also produce bias in estimating human poses. Meanwhile, visual features from color/grayscale images are ambiguous because of the change in illumination, occlusion, viewpoint, camera motion, cluttered background and the 3D-to-2D projection. For example, a silhouette can be formed from different poses; the same color may result in different color-intensity values in the images because of the change in illumination. These one-to-many mappings between visual features and human poses create additional burdens on the modeling of human poses.

†This work was supported in part by the National Science Foundation under Grants CNS-0958487, CNS-0960061 and IIS-0916807. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Several attempts have been reported to alleviate the problem of such mappings using visual or geometric features. By considering the temporal effect of visual features [1], 2D motion can reduce the ambiguity of difference of appearance. In addition, multiple visual features could be incorporated to reduce the ambiguity [2]. More comprehensive surveys can be found in [3] [4]. Recently, Time-of-Flight (TOF) sensors such as Swissranger and Microsoft Kinect become widely available. Geometric features from the 3D point cloud [5], which is a set of points containing only the 3D positions, can resolve the depth ambiguity directly and encode the 3D geometry of human poses. Although the 3D point cloud tends to be *noisy* and at least half of the surface of a human cannot be captured from a single TOF sensor in each frame, the geometric features could represent human poses geometrically using silhouette [6] and mesh [7], and hence assist the visual features in reducing the ambiguities of depth and illumination. Other constraints such as kinematic chain (skeleton) [8], joint limit and penetration [9] could be applied to reconstruct human poses.

In this paper, we study the performance of 3D human-pose estimation using geometric features from a 3D point cloud captured by a TOF sensor. We propose a geometric feature, namely VIewpoint and Shape feature Histogram (VISH), which is derived based on Viewpoint Feature Histogram (VFH) [10]. The novelty in our proposed geometric feature VISH is the spatial ordering of the VFH and shape features. Spatial ordering has been used for 3D-point-cloud preprocessing [11] and 3D-surface reconstruction [12]. This paper shows that the spatial ordering is important in resolving the ambiguity of symmetric poses. We have evaluated VISH on the Stanford TOF Motion Capture Dataset [8] using the k-nearest neighbor algorithm (k-NN) [13] and the support vector machines (SVM) [14]. Our results showed that VISH provides a more accurate 3D human-pose estimate, compared to the VFH and the Accumulative Geodesic EXtrema (AGEX) [15].

II. VIEWPOINT AND SHAPE FEATURE HISTOGRAM

Histograms of oriented gradients (HOG) is a common visual feature, based on the orientation histograms obtained from 2D color/grayscale images, for human detection and human-pose estimation [16] [17]. Since HOG is computed based on gradients, it is to some extent invariant to the change in illumination. The proposed VISH measures the orientation and shape responses from a 3D point cloud and hence can be considered as a 3D adaptation of HOG.

To handle the large number of 3D points in a 3D point cloud, the distribution of the 3D points is summarized into a nonparametric distribution using VFH and shape extractors. The summarization is performed on overlapping 3D regions of the 3D point cloud so that the spatial ordering can be captured.

There are three steps to extract VISH from a 3D point cloud, namely 3D-point-cloud pre-processing, hierarchical structuring and feature extraction. In the pre-processing of 3D point cloud, 3D points corresponding to a human are extracted and outliers are removed to retain the 3D points of interest. This step is important to reduce the number of 3D points by keeping only the 3D points from the human for further processing. In the hierarchical structuring, the pre-processed 3D point cloud is partitioned and replicated into a tree structure as nodes. VFH and shape features are extracted from each node in the tree to provide a descriptor to represent each node. As the features are obtained based on histograms, coarse-level details are highlighted in large regions and fine-level details are highlighted in small regions. Therefore, the features from the point cloud in the tree can capture coarse level to fine level information.

A. Pre-processing of 3D Point Cloud

We assume that the person, whose pose is to be estimated, performs different actions in a predefined 3D region. Therefore, 3D points corresponding to the person can be extracted by filtering away 3D points outside the 3D region. Let $\mathscr{P}$ and $\mathscr{R}$ be the 3D point cloud captured from a TOF sensor and the set of 3D coordinates (points) in the predefined 3D region, respectively. The set of 3D points corresponding to the person, $\mathscr{P}_A$, is given by the intersection of the sets $\mathscr{P}$ and $\mathscr{R}$,

$$\mathscr{P}_A = \mathscr{P} \cap \mathscr{R}. \tag{1}$$

Because of the measurement noise from the TOF sensor, there are outliers in the 3D point cloud $\mathscr{P}_A$. Smoothing or filtering techniques [18] can be used to remove the outliers by smoothing the spatial and temporal information of the 3D points. However, the size of the 3D points and the time interval for smoothing depend on the motion of the human and the sampling rate of the TOF sensor. Instead of estimating these two quantities, we assume the average distance between two neighboring points in the 3D point cloud $\mathscr{P}_A$ follows a continuous cumulative distribution function $F(\cdot)$. Hence, the outliers can be detected by the pseudo-residual method [19]. Let D be the random variable of the average distance. Then, the random variable $F(D)$ follows a uniform distribution U from 0 to 1; that is,

$$F(D) \sim U(0,1). \tag{2}$$

The auxiliary random variable, denoted as Z, for outlier detection is defined as

$$Z \triangleq \Phi^{-1}(F(D)), \tag{3}$$

where $\Phi^{-1}(\cdot)$ is the inverse of the cumulative distribution function of the standard normal distribution. Thus, the resulting point cloud, denoted as $\mathscr{P}_H$, after removing the outliers, can be derived by thresholding the deviation of the realization of the auxiliary random variable Z from the mean of the normal distribution.

B. Hierarchical Structuring

The 3D point cloud of a human in the predefined region $\mathscr{P}_H$ is organized into a tree structure such that a node in the tree represents a set of 3D points in a 3D region and an edge represents the process of duplicating 3D points. In this paper, the 3D region is represented by a rectangular region (cuboid) but it could be represented by other shapes [20] as well. Nodes at different levels of the tree capture the spatial ordering of sets of 3D points. At the zeroth level (root level), the root node represents the 3D point cloud $\mathscr{P}_H$.

Let M^0 be the set of the smallest cuboid that contains the points in the root node at the zeroth level. Let $S_n(\cdot)$ be the function that splits a cuboid into a set of n exhaustive, continuous, mutually exclusive, and equal-sized cuboids. Let M^i be the set of cuboids returned by the function $S_{n_i}(\cdot)$ at the ith level, where n_i is the number of cuboids returned by the function $S_{n_i}(\cdot)$ at the ith level and n_0 is defined as 1. The jth element in M^i is denoted as $\mathscr{M}^i_j$. The set M^i can be derived by the following recursive formula,

$$M^i = \bigcup_{j=1}^{|M^{i-1}|} S_{n_i}(\mathscr{M}^{i-1}_j), \quad \forall i = 1,2,\ldots,h, \tag{4}$$

where $\bigcup$ is the set union, $|\cdot|$ is the cardinality of the input set and h is the height of the tree, with the convention that the root node is at level 0.

In each cuboid $\mathscr{M}^i_j$, $i = 1,\ldots,h$, $j = 1,\ldots,|M^{i-1}|$, 3D points from the 3D point cloud $\mathscr{P}_H$ are extracted for feature extraction. Let $P_H(\cdot)$ be the function that takes such cuboid as input and outputs a set of 3D points from the 3D point cloud $\mathscr{P}_H$ that lies in the input cuboid. In the breadth-first fashion, we apply the function $P_H(\cdot)$ to the cuboid at each level of the tree structure to obtain a set of collections of 3D points, denoted as W, as follows,

$$W = \bigcup_{i=0}^{h} \bigcup_{j=1}^{|M^i|} \{P_H(\mathscr{M}^i_j)\}. \tag{5}$$

To illustrate the idea, Figure 1 shows an example of the hierarchical structuring when the height of the tree is 1 and the number of cuboids split (n_1) is 2. In the figure, the right side shows the smallest cuboid that contains the 3D point cloud $\mathscr{P}_H$ from a person. The root node in the tree represents the smallest cuboid as shown on the upper left region. M^0 is the set that contains the cuboid. The cuboid is then divided into 2 sub-cuboids with equal volume by the function $S_2(\cdot)$. The sub-cuboids are the children of the root node in the tree. M^1 contains the two sub-cuboids. Points from the 3D point cloud $\mathscr{P}_H$ in each node of the tree are extracted and stored in W for feature extraction. In the example, W has three elements, denoted as W_1, W_2, and W_3.

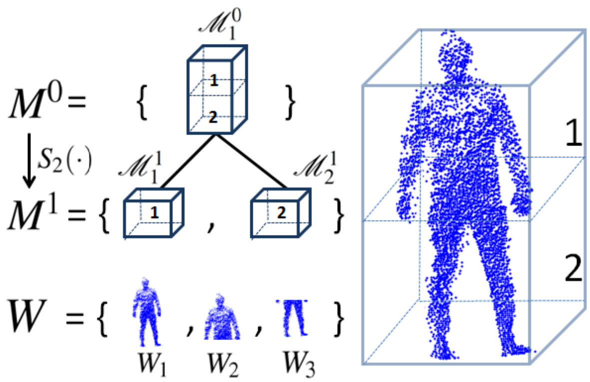

Fig. 1: An example of hierarchical structuring with the height of the tree being 1 and the number of cuboids split by $S_{n_1}(\cdot)$ being 2. The smallest cuboid, which contains the 3D point cloud $\mathscr{P}_H$ from the person, is shown on the right side. The cuboid is divided into two smaller sub-cuboids with equal volume by $S_{n_1}(\cdot)$. All the cuboids are arranged into a tree structure as shown on the left upper region. Points from the 3D point cloud $\mathscr{P}_H$ are extracted from each cuboid and grouped together in W for feature extraction.

C. Feature Extraction

Two features, namely VFH and shape features, are extracted from the set of 3D points in each cuboid. Let $\mathbf{f}(\cdot)$ be the mapping from a set of 3D points to the row vector of VFH and shape features. VISH can be arranged into a row vector $\mathbf{F}$ and is found by concatenating all the features extracted from each element of W; that is,

$$\mathbf{F} = (\mathbf{f}(W_1), \mathbf{f}(W_2), \ldots, \mathbf{f}(W_{|W|})), \tag{6}$$

where W_i is the ith element in W.

Note that in the tree structure, 3D points in a child node are duplicated from its parent node. It provides an illusion that 3D points are redundant among different levels of the tree. However, VFH and shape features are the summarization of 3D points in cuboids. The features extracted from different scales of cuboids provide different levels of details in the summarization. In addition, the order of the set W provides useful information of spatial ordering of the 3D points from the human surface. In the tree, each 3D point appears exactly $(h+1)$ times. As shown below, the time complexity of extracting VFH and shape features is $O(n)$. Thus, the proposed feature extraction is $O((h+1)n)$, where n is the number of 3D points. Moreover, experimental results show that h is typically small compared to n. Thus, the time complexity of the proposed feature extraction can be rewritten as $O(n)$. VISH is asymptotically more efficient than AGEX, which has the time complexity $O(n\log n)$ [15].

We next derive the VFH and shape features and analyze their time complexities.

1) VFH [10]: VFH is originally designed to estimate the 6 degree-of-freedom pose of rigid objects. Each 3D point in a 3D point cloud is first assigned a direction. The direction of a 3D point is found by three steps: 1) Find the k nearest neighbors of the 3D point; 2) Find the normal to the 2D plane that contains the 3D point and minimizes the average perpendicular distance between the nearest neighbors and the plane; 3) Set the normal to the 2D plane pointing towards the TOF sensor as the direction of the 3D point.

During the derivation of the normal (step 2), each 3D point, $\mathbf{y}$, in the k nearest neighbors is represented by the summation of three vectors that form an orthonormal set; that is,

$$\mathbf{y} = \sum_{i=1}^{3} a_i \psi_i, \tag{7}$$

where a_i is the weight corresponding to ψ_i, and ψ_1, ψ_2 and ψ_3 are vectors from the orthonormal set.

Without loss of generality, let ψ_1 be the normal to the 2D plane. Let ψ_2 and ψ_3 be the two vectors that span the 2D plane. Then, the direction of the 3D point is given by ψ_1 that is found by minimizing the average squared error distance between $\mathbf{y}$ and the 2D plane. The average squared error distance in the k nearest neighbors, $\varepsilon^2(b)$, is given by

$$\varepsilon^2(b) = E[(a_1 - b)^2], \tag{8}$$

where b is a constant weight corresponding to ψ_1, and $E(\cdot)$ is the expectation function.

Taking the derivative of Eq. (8) with respect to b and setting it to zero, the local minimizer b^* is given by

$$b^* = E[a_1] = \psi_1^T E[\mathbf{y}], \tag{9}$$

where T is the transpose operator.

The minimum squared error distance can be written as

$$\begin{aligned} \varepsilon_2(b^*) &= E[(a_1 - E[a_1])^2] = E[(\psi_1^T \mathbf{y} - \psi_1^T E[\mathbf{y}])^2] \\ &= \psi_1^T \Sigma \psi_1, \end{aligned} \tag{10}$$

where Σ is the covariance matrix of $\mathbf{y}$.

It can be shown that the optimal ψ_1 for the minimum squared error distance is the eigenvector of Σ corresponding to the smallest eigenvalue. Then, VFH is formed by combining two components. The first component is the histogram of the pairwise pan, tilt and yaw angles between every pair of 3D points. The second component is the histogram of the angles between the viewpoint direction and the direction of each 3D point. The time complexity of VFH is $O(n)$, where n is the number of 3D points [10].

2) Shape: The shape of a 3D point cloud $\mathscr{X}$ is measured around the 3D centroid of $\mathscr{X}$ on the range image, denoted as I, which is the 2D projection of $\mathscr{X}$. The 3D centroid, denoted as $\mathbf{c}$, of $\mathscr{X}$ is defined as

$$\mathbf{c} = \frac{1}{|\mathscr{X}|} \sum_{\mathbf{x} \in \mathscr{X}} \mathbf{x}. \tag{11}$$

Each pixel value in I represents the distance between the corresponding 3D point in $\mathscr{X}$ and the TOF sensor, with the convention that pixel values outside I are zeros. Let $I(x,y)$ be the pixel value of I at the position (x,y). Let (x^*, y^*) be the position on I where $\mathbf{c}$ is projected. The shape feature is defined as the row vector containing the pixels in the region of size $(2w+1)$ pixels by $(2w+1)$ pixels centered at the position (x^*, y^*) on I. Let $\mathbf{f}_s(\mathscr{X})$ be the mapping from $\mathscr{X}$

to the row vector containing the shape feature of $\mathscr{X}$. Then, $\mathbf{f}_s(\mathscr{X})$ can be expressed as

$$\mathbf{f}_s(\mathscr{X}) = (I(x^* - w, y^* - w), I(x^* - w, y^* - w + 1), \\ \ldots, I(x^* - w + 1, y^* - w), \ldots, I(x^* + w, y^* + w)). \quad (12)$$

To compute the shape feature, all the 3D points in $\mathscr{X}$ are accessed to compute the 3D centroid. Then, $(2w+1)^2$ pixels in the region on I are extracted. Typically, w is small compared to n. Thus, the time complexity of the shape feature is $O(n)$, where n is the number of 3D points.

D. Comparison between VISH and VFH

The main difference between VISH and VFH is the spatial ordering of 3D point cloud. In VISH, VFH and shape features are extracted from the 3D point clouds in a tree. In VFH, the feature is extracted from the input 3D point cloud directly. As VFH is a histogram-based feature, extracting the feature from the input 3D point cloud directly can only capture the global properties of the point cloud. Any local properties (fine details) are suppressed. On the other hand, VISH can capture both the global and local properties by partitioning the input 3D point cloud into regions. Figure 2 shows the limitation of only capturing the global properties. In the figure, the subject at the bottom left is raising his/her left arm. The subject pose is represented by the VFH and the closest matches from a database are returned by k-NN as shown in the other regions. Some of the closest matches are raising the other arm. This shows that VFH gives a similar description among symmetric 3D human poses and cannot distinguish them. When the subject pose is represented by VISH, this ambiguity is removed as shown in Figure 3.

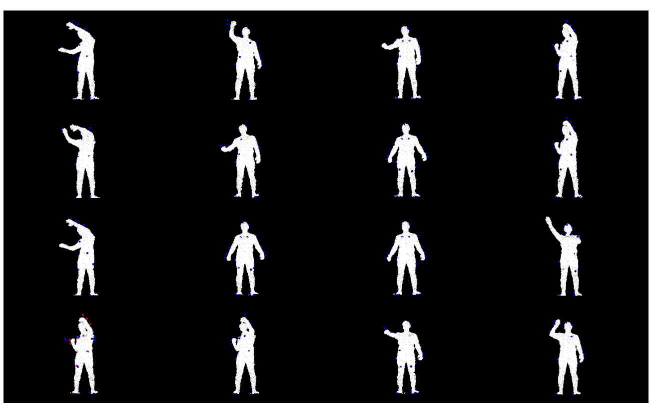

Fig. 2: Ambiguity of symmetric 3D human poses represented by VFH exists in some closest matches using k-NN. The 3D point cloud at the bottom left is the query pose. The others are the 3D point clouds returned by k-NN.

III. Experimental Results

The proposed geometric feature was implemented in C++ using PCL [21] and tested on the Stanford TOF Motion Capture Dataset [8], which contains 28 video sequences. The number of frames of the video sequences is shown in Table I. In the dataset, a subject performed different actions such as kicking and rotation, and was captured by a Swissranger SR4000 TOF sensor. The range images were captured at 25 fps at a resolution of 176 pixels $\times$ 144 pixels. The ground-truth 3D joint locations of the subject were recorded by a commercial motion-capture system. The error metric, ζ, for each video sequence was defined as

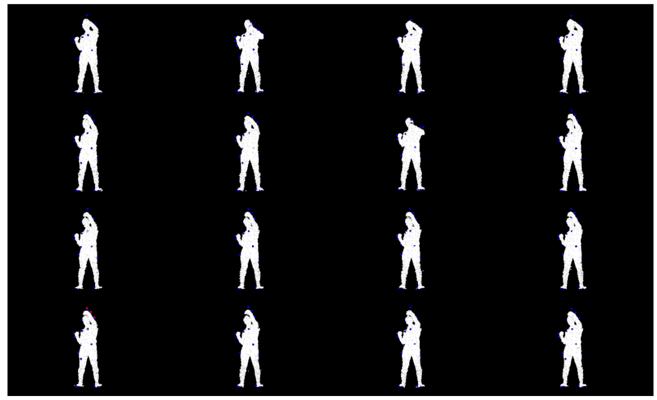

Fig. 3: Ambiguity of symmetric 3D human poses is removed using VISH.

$$\zeta = \frac{1}{N_f} \sum_{s=1}^{N_f} \frac{1}{N_s} \sum_{i=1}^{N_s} \left\| \mathbf{j}_{s,i} - \tilde{\mathbf{j}}_{s,i} \right\|_2, \quad (13)$$

where N_f is the number of frames of the video sequence for testing, N_s is the number of 3D joint locations measured by the motion-capture system at the sth frame, $\mathbf{j}_{s,i}$ is the ground-truth 3D location of the ith joint at the sth frame, $\tilde{\mathbf{j}}_{s,i}$ is the estimated 3D location of the ith joint at the sth frame and $\|\cdot\|_2$ is the Euclidean norm.

TABLE I: Total number of frames in each video sequence from the dataset.

Number of frames	Video index
100	0-5,8,9,14,16,18,
400	6,7,10-13,15,17,19,20-27

Two existing geometric features, namely VFH and AGEX, were implemented for comparison using k-NN and SVM. Each video sequence was divided into 5 equal parts randomly to perform 5-fold cross-validation.

In the pre-processing step, the predefined 3D region was set to be the 3D space with the depth ranging from -3 meters to -2 meters. 50 closest 3D points were used to estimate the average distance of each 3D point. The distribution function $F(\cdot)$ of the average distance was set to be normal distribution. 3D points were considered as outliers if their deviations from the mean of the normal distribution were larger than one standard deviation.

In the feature extraction step, two levels were used in the tree structure. Six cuboids were split from the smallest cuboid containing the root node. To estimate the direction of a 3D point, 3D points within 0.01 meters from that 3D point were used. The size of the 2D region used for shape feature was set to be 17 pixels $\times$ 17 pixels.

A. Evaluation using k-NN

The goal of the experiment is to evaluate the accuracy of 3D human-pose estimation using the proposed feature in a discriminative model. k-NN was used for learning the mapping from a geometric feature to a 3D human pose in the training database. The k nearest 3D human poses were then averaged to give the final estimate of the 3D human pose; that is,

$$\tilde{\mathbf{j}}_i = \frac{1}{k}\sum_{p=1}^{k}\mathbf{j}_i^p, \tag{14}$$

where $\tilde{\mathbf{j}}_i$ is the estimated 3D location of the ith joint and $\mathbf{j}_i^p$ is the ground-truth 3D location of the ith joint of the pth closest match.

Table II shows the quantitative result of the feature evaluation when $k = 3$. A bar chart of the errors using different features is shown in Figure 4. The first bar corresponding to the error of VISH was significantly lower than the other two bars and it showed the strength of spatial ordering of 3D point cloud in VISH. The second bar shows the error of using VFH. The error was higher than that of VISH because VFH suffered from the limitation of describing the local properties of input 3D point cloud. The third bar shows the error of using AGEX. In AGEX, when the limbs of the person were moved, the geodesic extrema of the limbs might be switched or lost. It failed to look up the 3D human poses in the dataset. The overall error of the proposed feature was 0.017m. In VFH and AGEX, the overall errors were about twice and 4.6 times as much as that of the proposed feature, respectively. It shows that the proposed feature is more discriminative than the other two features.

TABLE II: A quantitative result of feature evaluation using k-NN with k=3. Numbers on the left and the inside of parentheses are errors and standard deviations (in meters), respectively.

Test case	VISH	VFH	AGEX
0	0.015 (0.008)	0.030 (0.021)	0.042 (0.023)
1	0.016 (0.012)	0.039 (0.027)	0.073 (0.037)
2	0.013 (0.008)	0.028 (0.019)	0.053 (0.030)
3	0.021 (0.012)	0.026 (0.013)	0.084 (0.051)
4	0.015 (0.008)	0.032 (0.014)	0.047 (0.024)
5	0.014 (0.009)	0.039 (0.023)	0.051 (0.029)
6	0.012 (0.008)	0.038 (0.021)	0.055 (0.029)
7	0.016 (0.012)	0.033 (0.022)	0.054 (0.027)
8	0.012 (0.010)	0.038 (0.025)	0.055 (0.026)
9	0.015 (0.008)	0.040 (0.028)	0.065 (0.036)
10	0.019 (0.010)	0.033 (0.023)	0.078 (0.059)
11	0.013 (0.007)	0.024 (0.016)	0.056 (0.030)
12	0.012 (0.008)	0.025 (0.019)	0.077 (0.049)
13	0.012 (0.010)	0.026 (0.026)	0.068 (0.038)
14	0.020 (0.016)	0.025 (0.019)	0.085 (0.057)
15	0.014 (0.008)	0.034 (0.020)	0.061 (0.031)
16	0.017 (0.012)	0.065 (0.042)	0.071 (0.034)
17	0.023 (0.015)	0.058 (0.034)	0.086 (0.043)
18	0.015 (0.011)	0.034 (0.027)	0.079 (0.030)
19	0.016 (0.011)	0.042 (0.030)	0.071 (0.048)
20	0.020 (0.013)	0.040 (0.025)	0.082 (0.058)
21	0.022 (0.014)	0.043 (0.023)	0.085 (0.045)
22	0.014 (0.009)	0.042 (0.022)	0.073 (0.046)
23	0.015 (0.011)	0.048 (0.043)	0.089 (0.069)
24	0.021 (0.022)	0.049 (0.059)	0.141 (0.121)
25	0.020 (0.020)	0.032 (0.025)	0.096 (0.070)
26	0.018 (0.012)	0.030 (0.030)	0.097 (0.058)
27	0.023 (0.019)	0.060 (0.057)	0.177 (0.090)
Overall	**0.017 (0.012)**	**0.038 (0.027)**	**0.077 (0.046)**

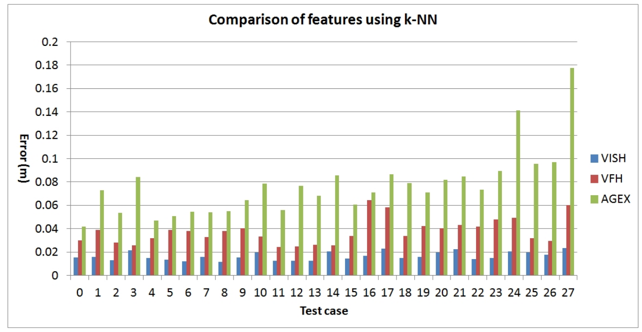

Fig. 4: Feature evaluation using k-NN.

The overall error of features under different values of k in k-NN is plotted in Figure 5. The overall error of VISH was the lowest among the three features when the value of k was changed. When the value of k started to increase from one, the error was decreased because the noise in the point cloud $\mathscr{P}_H$ was averaged out. As the value of k increased, the error was first decreased, but started to increase because the details from the point cloud $\mathscr{P}_H$ were also averaged out when k was too big. Overall, VISH is robust across a wide range of k in k-NN.

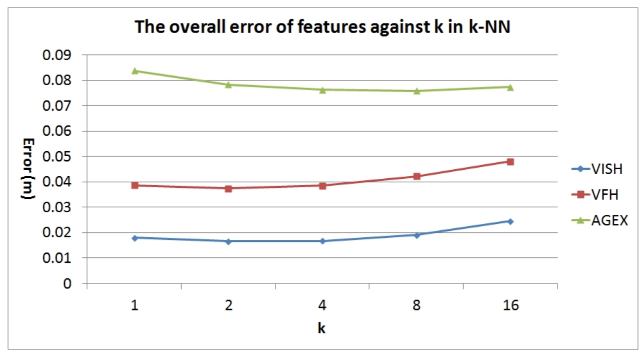

Fig. 5: Comparison of features using k-NN under different values of k.

B. Evaluation using SVM

The accuracy of the proposed feature was also evaluated using SVM. 1500 pose prototypes were used to model 3D human poses. They were found by running the k-means algorithm on the training video sequences.

Table III shows the quantitative result of the feature evaluation. A bar chart of the errors using different features is shown in Figure 6. The overall errors of VFH and AGEX were about 1.5 times and 2.4 times as much as that of VISH, respectively. It assures that the proposed feature performs better than the other two features. Note that the overall error incurred when using SVM was higher than the error incurred when using k-NN. The main reason is the quantization error in the pose prototypes when modeling the 3D human poses in a high-dimensional space.

IV. Summary and Conclusions

In this paper, we have proposed a geometric feature VISH, which is the 3D adaptation of HOG, for estimating human poses from a 3D point cloud. VISH is composed of pre-processing of 3D point cloud, hierarchical structuring and

TABLE III: A quantitative result of feature evaluation using SVM. Numbers on the left and the inside of parentheses are errors and standard deviations (in meters), respectively.

Test case	VISH	VFH	AGEX
0	0.036 (0.043)	0.038 (0.051)	0.069 (0.053)
1	0.072 (0.076)	0.126 (0.091)	0.122 (0.083)
2	0.031 (0.047)	0.033 (0.055)	0.074 (0.044)
3	0.071 (0.072)	0.077 (0.080)	0.101 (0.053)
4	0.021 (0.035)	0.064 (0.045)	0.103 (0.047)
5	0.039 (0.044)	0.065 (0.061)	0.075 (0.069)
6	0.040 (0.060)	0.065 (0.070)	0.101 (0.067)
7	0.034 (0.063)	0.047 (0.064)	0.133 (0.065)
8	0.055 (0.059)	0.104 (0.085)	0.164 (0.056)
9	0.019 (0.034)	0.040 (0.062)	0.123 (0.051)
10	0.059 (0.075)	0.075 (0.078)	0.138 (0.066)
11	0.015 (0.035)	0.022 (0.049)	0.164 (0.051)
12	0.030 (0.047)	0.056 (0.069)	0.146 (0.067)
13	0.030 (0.050)	0.030 (0.052)	0.099 (0.040)
14	0.048 (0.055)	0.054 (0.070)	0.103 (0.050)
15	0.027 (0.043)	0.056 (0.063)	0.144 (0.069)
16	0.023 (0.043)	0.082 (0.064)	0.084 (0.061)
17	0.079 (0.085)	0.128 (0.098)	0.202 (0.085)
18	0.056 (0.068)	0.071 (0.065)	0.077 (0.055)
19	0.051 (0.076)	0.085 (0.088)	0.151 (0.093)
20	0.071 (0.077)	0.114 (0.089)	0.140 (0.074)
21	0.101 (0.097)	0.130 (0.094)	0.173 (0.076)
22	0.064 (0.094)	0.136 (0.109)	0.194 (0.067)
23	0.071 (0.094)	0.133 (0.117)	0.143 (0.082)
24	0.138 (0.158)	0.173 (0.185)	0.204 (0.138)
25	0.098 (0.107)	0.106 (0.102)	0.149 (0.094)
26	0.072 (0.104)	0.078 (0.106)	0.158 (0.105)
27	0.135 (0.144)	0.139 (0.163)	0.201 (0.132)
Overall	**0.057 (0.071)**	**0.083 (0.083)**	**0.133 (0.071)**

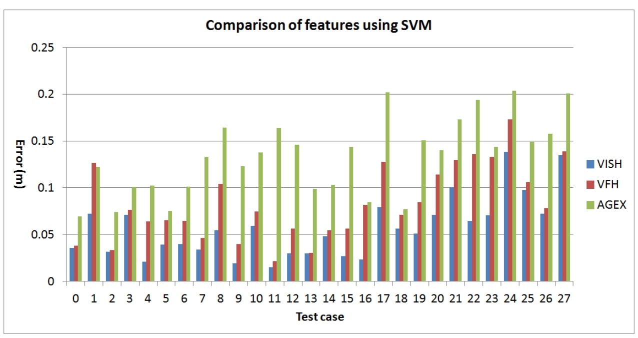

Fig. 6: Feature evaluation using SVM.

feature extraction. In the pre-processing step, region-based thresholding and pseudo-residual were used to stabilize the 3D point cloud from a person. The stable 3D point cloud was then organized into a tree structure in which VFH and shape features were extracted separately from each node in the tree. VISH was derived by combining all the features and therefore preserved the spatial ordering of the 3D point cloud. The spatial ordering was shown to be important in removing the ambiguity of symmetric 3D human poses. Two existing geometric features, namely VFH and AGEX, were implemented and compared with the proposed geometric feature. Both k-NN and SVM showed that the proposed geometric feature produces less errors than the other two features, suggesting that the proposed feature can describe the 3D point cloud more accurately for 3D human-pose estimation. The time complexity of the proposed feature extraction is asymptotically the same as that of VFH, which is $O(n)$, and VISH is more efficient than AGEX, which is of time complexity $O(n \log n)$. Future work will be focused on the use of the proposed feature for real-time applications.

REFERENCES

[1] A. Fathi and G. Mori. Human pose estimation using motion exemplars. In *IEEE Conference on Computer Vision*, pages 1–8, Oct. 2007.
[2] A. Agarwal and B. Triggs. 3D human pose from silhouettes by relevance vector regression. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 882–888, Jun. 2004.
[3] T.B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126, Nov. 2006.
[4] R. Poppe. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108(1-2):4–18, Oct. 2007.
[5] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1297–1304, Jun. 2011.
[6] J. Charles and M. Everingham. Learning shape models for monocular human pose estimation from the microsoft xbox kinect. In *IEEE International Conference on Computer Vision Workshops*, pages 1202–1208, Nov. 2011.
[7] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. Accurate 3D pose estimation from a single depth image. In *IEEE International Conference on Computer Vision*, pages 731–738, Nov. 2011.
[8] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real time motion capture using a single time-of-flight camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 755–762, Jun. 2010.
[9] Y. Zhu, B. Dariush, and K. Fujimura. Controlled human pose estimation from depth image streams. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, Jun. 2008.
[10] R.B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3D recognition and pose using the viewpoint feature histogram. In *IEEE International Conference on Intelligent Robots and Systems*, pages 2155–2162, Oct. 2010.
[11] L. Alboul and G. Chliveros. A system for reconstruction from point clouds in 3D: Simplification and mesh representation. In *IEEE International Conference on Control Automation Robotics Vision*, pages 2301–2306, Dec. 2010.
[12] W. Brink, A. Robinson, and M. Rodrigues. Indexing uncoded stripe patterns in structured light systems by maximum spanning trees. In *Proceedings of the British Machine Vision Conference*. British Machine Vision Association, 2008.
[13] R.O. Duda, D.G. Stork, and P.E. Hart. *Pattern classification and scene analysis. Part 1, Pattern classification*. Wiley, 2 edition, Nov. 2000.
[14] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.
[15] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun. Real-time identification and localization of body parts from depth images. In *IEEE International Conference on Robotics and Automation*, pages 3108–3113, May 2010.
[16] R. Poppe. Evaluating example-based pose estimation: Experiments on the humaneva sets. In *CVPR 2nd Workshop on Evaluation of Articulated Human Motion and Pose Estimation*, 2007.
[17] S. Wang, H. Ai, T. Yamashita, and S. Lao. Combined top-down/bottom-up human articulated pose estimation using Adaboost learning. In *IEEE International Conference on Pattern Recognition*, pages 3670–3673, Aug. 2010.
[18] D.T. Kuan, A.A. Sawchuk, T.C. Strand, and P. Chavel. Adaptive noise smoothing filter for images with signal-dependent noise. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(2):165–177, Mar. 1985.
[19] W. Zucchini and I.L. MacDonald. *Hidden Markov Models for Time Series: An Introduction Using R*. Chapman & Hall/CRC, 2009.
[20] G. Wasson, D. Kortenkamp, and E. Huber. Integrating active perception with an autonomous robot architecture. *Robotics and Autonomous Systems*, 29(2-3):175–186, 1999.
[21] R.B. Rusu and S. Cousins. 3D is here: Point cloud library (PCL). In *IEEE International Conference on Robotics and Automation*, pages 1–4, May 2011.