2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops

Accurate and Efficient 3D Human Pose Estimation Algorithm using Single Depth Images for Pose Analysis in Golf

Soonchan Park[1], Ju Yong Chang[2], Hyuk Jeong[1], Jae-Ho Lee[1], Ji-Young Park[1]
[1]Electronics and Telecommunications Research Institute
[2]Dept. of Electronics and Communications Engineering, Kwangwoon Univ.
parksc@etri.re.kr, jychang@kw.ac.kr, jayahyuk@gmail.com, {jhlee3, jiyp}@etri.re.kr

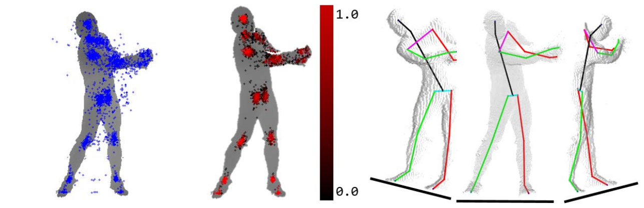
Abstract

Human pose analysis has been known to be an effective means to evaluate athlete's performance. Marker-less 3D human pose estimation is one of the most practical methods to acquire human pose but lacks sufficient accuracy required to achieve precise performance analysis for sports. In this paper, we propose a human pose estimation algorithm that utilizes multiple types of random forests to enhance results for sports analysis. Random regression forest voting to localize joints of the athlete's anatomy is followed by random verification forests that evaluate and optimize the votes to improve the accuracy of clustering that determine the final position of anatomic joints. Experiential results show that the proposed algorithm enhances not only accuracy, but also efficiency of human pose estimation. We also conduct the field study to investigate feasibility of the algorithm for sports applications with developed golf swing analyzing system.

1. Introduction

For sporting activities such as golf, the posture of the player and how his/her motions are formed can heavily impact the performance of the game. Given such importance, especially for golf, various studies and services have attempted to scrutinize golf swings with human pose [1, 19, 17] that would have been difficult to perceive with the naked eye. To obtain accurate human pose, human pose estimation technologies such as [2, 4, 7] are used. However, such technologies are difficult to be widely utilized because they all require either wearing a device or placing markers on the player's body.

On the other hand, in the computer vision research community, with proliferation of advanced depth cameras such as Kinect [3], various researches regarding marker-less 3D human pose estimation have been performed. Specifically, by analyzing depth images, the researches effectively localize human anatomic joints via tracking methods [8, 23, 12, 24] or utilizing a pre-trained machine like random forest [13, 20, 10, 21, 15, 16]. However, in spite of its convenience, the lack of precision has been a major obstacle for applications that require sufficient accuracy such as 3D pose analysis for sports.



(a) Casting 3d Votes (b) Vote-verification (c) Estimation Results

Figure 1. Verification forests evaluate votes and filter invalid votes to accurately and efficiently estimate human pose. (a) all votes cast by random regression forest, (b) only verified votes and their confidence values, and (c) result of 3D human pose estimation in three different views.

Among them, [13, 21] introduce random regression forest based methods that cast 3D votes and cluster them to localize joints even when the joints are occluded. However, examining the votes generated by random regression forest shows that a significant amount of the votes has noise as shown in Fig. 1-(a). Although the clustering algorithm such as mean shift might alleviate the noise of the votes, it is obvious that the invalid votes hinder accurate and efficient localization of the joints.

In this paper, we propose a 3D human pose estimation algorithm which enhances its precision and efficiency by verifying the votes. Initially, a random regression forest casts 3D votes to estimate position of joints as shown in Fig. 1-(a) which are then efficiently evaluated by a set of random verification forests as shown in Fig. 1-(b). By result of the verification, the verified votes maintain their weight, while the invalid votes lose their weight or even they are eliminated. With the enhanced conciseness of the votes, an algorithm

2160-7516/17 $31.00 © 2017 IEEE
DOI 10.1109/CVPRW.2017.19

to estimate clusters of the votes accurately and efficiently localizes 3D joints as shown in Fig. 1-(c). By performing the scheme over a 10,000 frames of golf swing data set, we notice that our proposed approach of validating the votes not only improves the accuracy but also reduces the computation time compared to the algorithm just utilizing the random regression forest.

Furthermore, we conduct a field study by developing a prototype system for golf swing analysis utilizing the proposed human pose estimation algorithm. With no additional devices but depth camera, the system evaluates golf swing by examining 3D human pose such as the angle of the joints, swing trajectories, etc. With a sequence of poses extracted from the swing, the system is capable of automatically identifying seven important phases of the swing such as backswing top or impact.

The organization of the paper is as follows: Sec. 2 contains a brief introduction to relevant work existing in literature, Sec. 3 elucidate the golf swing data set we acquired. Sec. 4 explains the random forest based algorithm and how to train it. Sec. 5 describes experiments and presents their results that verifies the performance of the algorithm, and sec.6 introduces a conducted field study by developing a prototype system for evaluating poses in golf swing using the proposed algorithm. Sec. 7 organizes the contributions of our work and introduces future work.

2. Related work

We will briefly mention previous work related to human pose estimation, and introduce existing applications for evaluating pose of golf swing.

2.1. Existing human pose estimation algorithms

2D human pose estimation with RGB images is a traditional problem that has been widely investigated. Currently researches like [18, 22, 9] aim to enhance the estimation performance by using deep learning approaches. Among them, [18] performs 3D analysis using 2D images; however, this strategy cannot replace 3D human pose estimation because of ambiguities between 2D and 3D human poses.

With the development of feasible depth cameras, various studies related to human pose estimation have been conducted by utilizing advantages of 3D information. Existing algorithms can be categorized into generative and discriminative approaches. Generative approaches fit articulated models to current depth observation frame by frame [8, 23, 12, 24]. The tracking approaches require delicate algorithms to build appropriate articulated models for each user [24] and complex optimization algorithms to minimize the difference between articulated model and depth information, such as Iterated Closest Point. Consequently, these approaches hardly achieve estimation results in real-time except [12]. Moreover, the tracking mechanisms, which utilize the pose of previous frame, exhibit limitations when the algorithm analyzes rapid movements, such as golf swing; that is, poses with considerable differences from sequential frames are difficult to optimize.

Discriminative approaches analyze human poses from a single depth image by training a machine in advance. [20, 10, 21] introduce algorithms based on random classification forest which examines that each visible pixel of foreground is belonging to which body part. Meanwhile, random regression forests can be used to estimate the entire human joints, including occluded joints, by learning relative offset [13, 21], or direction [15] from the foreground pixels to the ground truth joints. [16] enhances the accuracy of estimation by partitioning localization of joints into two sub-problems. In this study, joint localization and identification are sequentially performed. This strategy enhances the accuracy of pose estimation but exhibits low efficiency; as such, without verification, the entire votes cast by the regression forest should be considered by clustering algorithm such as mean shift. The low efficiency is one of the issues in implementing the application since the entire system should run in real-time including not only the pose estimation algorithm, but also related preceding and following algorithms, such as subtracting the background from the scene, rendering the user interface, and so on.

2.2. Pose analysis of golf swing

When playing golf, the driving distance and direction are considerably changed by minimal alterations in pose, such as trajectories of arm. Researches and tools like [1, 19, 17] investigate the human pose to scrutinize golf swing by utilizing motion capture system for acquiring accurate pose data. Motion capture systems [4, 7] can directly estimate and evaluate human poses without occlusions by tracking markers placed on subject's body. However, processes to operate the systems are complicated and expensive. For instance, operations, such as calibrating multiple cameras, wearing suit, and placing markers, must be preceded. Simpler wearable device like [2] is also developed but still cannot avoid forcing a user to wear the devices.

On the other hand, camera-based approaches conveniently provide pose analysis without difficult settings and any additional device that a user need to wear or hold. [6] just provides functionalities of capturing video and manual visualization tools to analyze a golf swing. [5] performs automated 3D analysis by using a depth camera. The system visualizes head, backbone, and hips, and provides information, such as weight balance and hip turn. This approach seems similar to a prototype system we will introduce, but we focus on precisely estimating raw-level pose data in real-time to allow service providers to design numerous functions what they need.

3. Ground truth data

A data set including the depth image and corresponding 3D position of joint is required to implement human pose estimation algorithm. An input depth image $I = \{p_0, p_1, p_2, ..., p_{wh}\}$ represents the distance from the depth camera to the scene for each pixel p. A set of 3D position of ground truth joints J is $J = \{J_1, J_2, ..., J_n\}$, where $J_i \in \mathbb{R}^3$.

3.1. Data acquisition

We utilize KinectV2 [3] and Vicon motion capture system [7] to acquire I and J respectively. Kinect is a time-of-flight depth camera that captures depth images I in 30 frames per second (fps). The installed Vicon system consists of 17 infrared cameras, which surround a subject to track markers without occlusion. By referring to Vicon's Plug-in-Gait Marker Placement, we extract 21 joints, including head, neck, shoulder L/R, elbow L/R, wrist L/R, hand L/R, hip L/R, knee L/R, ankle L/R, foot L/R, shoulder spine, middle spine, and base spine in 120 fps.

We synchronize acquired I and J in terms of their coordinates and captured time. In implementing the calibration tool based on OpenCV[14], two coordinates are matched so that the data from each system can be transferred to the other coordinates. To synchronize timings of I and J, we utilize the time stamps of I and J.

To maximize the variation of the poses in the data set, we define seven typical swing types and ask a subject to take a swing by following the swing types. The seven types of swing are Sway, Reverse spine angle, Hanging back, Chicken wing, Early extension, Wide stance, and Narrow stance. Adding one free swing, eight swings are collected from a subject. We also control the distance between Kinect and the subject from 1.5 m to 3.0 m. Twenty subjects (14 males and 6 females) are invited, and as a total, 10,476 frames of golf swing data are acquired.

3.2. Data set arrangement

We apply a background subtraction algorithm to I in advance, so that only pixels on user are remained as foreground; we then assign a large constant number to pixels for background. We also optimize the number of the target joints. Shoulder spine, middle spine, and base spine are excluded in the data set for ground truth, and as a result, set of joints J consists of 18 joints.

We define the visibility of each joints. In general, J penetrates a depth image I because J describes the position of anatomic joints below human skin, and the amount of penetration is various by the thickness of each joint. We denote half of the thickness of each joint j as ρ_j, and measure ρ_j by estimating the maximum penetration depth among entire cases in the data set I and J. Table 1 describes decided ρ_j. With ρ_j, the visibility V of the joint j denoted as

Name	Neck*	Head	Shoulder	Elbow	Wrist
	Hand	Hip	Knee	Ankle	Foot
ρ_j(mm)	80.00*	167.03	128.97	89.25	56.72
	68.17	159.59	83.35	82.79	64.80

Table 1. The maximum penetration depth for joint j in the data set is defined as ρ_j. In the case of neck, the penetration depth of neck is difficult to be measured due to occlusion by the head. We define ρ of neck as 80.00 mm.

$$V_j = \begin{cases} 0 & \text{if} \quad J_j.z - I(\mathrm{p}(J_j)) > \rho_j \\ 1 & \text{otherwise} \end{cases} \tag{1}$$

where $\mathrm{p}(J_j)$ is a 2D pixel position in the depth image where joint J_j is projected. Finally, all data set for human pose estimation algorithm in this paper is based on the arranged data set $S = (I, J, V)$.

4. Human pose estimation algorithm

A random forest is a well-known ensemble learning method that contains multiple decision trees and collects estimation results from the trees. Researches like [13, 20, 21, 15, 16] aim to estimate 3D human pose by using random forest. In particular, [13, 21, 16] utilize random regression forest, which learn the relative position of target joints from foreground pixels. In testing time, foreground pixels traverse the random forest and generate votes to estimate a location of target joints. Then, mean shift which examines clusters from a distribution of the votes decides final estimation result. However, the votes are not always appropriate as Fig. 1-(a) visualized. The algorithm we present in this paper aims to verify and optimize the votes in order to enhance accuracy and efficiency of estimating human pose.

In this section, we introduce details of proposed human pose estimation algorithm in this paper. We explain structure of the algorithm which has multiple types of random forest, and then describe what and how each random forests learns the acquired golf swing data set. At the end of following section, we describe procedures of the algorithm in testing time with its pseudocode.

4.1. Type of decision trees

Initially, we construct two types of decision trees, namely, regression decision tree T_{reg} and verification decision tree T_{ver}. T_{reg} learns offsets from foreground pixels to ground truth joints; in testing, the tree casts 3D votes to determine the position of target joints regardless of their visibility[13, 21, 16]. For training set for T_{reg}, we sample foreground pixels and measure offsets from the pixels to ground truth joint j. Finally, training set for T_{reg} is $S_{reg} = (P_r, \Delta_{p \to J})$, where P_r is set of sampled pixels and

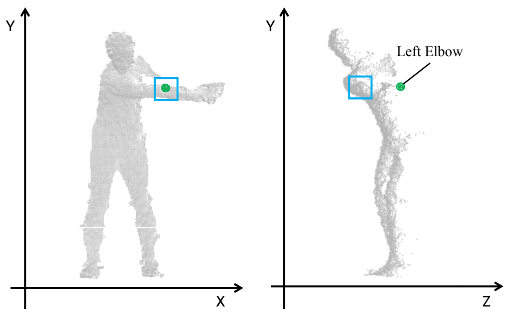


Figure 2. A left elbow is occluded by a right arm. In this case, $T_{ver,o}$ learns feature of the pixels of right arm(point cloud in blue square) which occlude a left elbow(green dot).

$\Delta_{p\rightarrow J}$ is a set of 3D vectors from the sampled pixels to each ground truth joints.

The verification decision tree T_{ver} evaluates votes cast by T_{reg} by analyzing foreground pixels placed where the votes are projected. Two verification decision trees are designed in this paper. The verification decision tree for visible joint, namely $T_{ver,v}$, investigates the only pixels for visible joints to estimate probabilities that the test pixels belong to certain joints. We employ classification methodologies like in [20]. For training data set for $T_{ver,v}$, we sample pixels distant within the pre-defined threshold γ from ground truth joint likely in [10]. The sampled pixels for visible joints P_v is denoted as

$$P_v = \left\{ p | \, \|p - \mathrm{p}(J_v)\|_2 < \frac{\gamma}{J_v.z} \right\} \tag{2}$$

where γ is a pixel distance threshold, and J_v is ground truth joints whose V is 1 mentioned in Eq. 1. γ is decided as six pixels through a prior experiment. We also compensate γ by the depth of the target joint, because the size of the joint in depth image will be decreased by its depth. At last, the training data set of $T_{ver,v}$ is arranged as $S_{ver,v} = (P_v, i_{P_v})$, where i_{P_v} is a integer index of the joint related to each pixel of P_v.

We then extend the concept of $T_{ver,v}$ to the case of occluded joints. Specifically, as $T_{ver,v}$ learns features of pixels near the ground truth joint, another verification decision tree for occluded joints $T_{ver,o}$ learns features of pixels covering the ground truth joints. For a specific example visualized in Fig. 2, when a left elbow is covered by pixels of a right arm(in a blue sqaure), features of those pixels are learned by $T_{ver,o}$. In testing time, when T_{reg} cast a vote for left elbow and the vote is placed behind input depth image, a pixel which covers the vote traverses $T_{ver,o}$ for examining probability that the pixel covers left elbow.

$$P_o = \left\{ p | \, \|p - \mathrm{p}(J_o)\|_2 < \frac{\gamma}{J_o.z} \right\} \tag{3}$$

where J_o is ground truth joint whose V is 0. Similarly, the coupled data of the sampled pixels and their indices of the related joint is training data set for $T_{ver,o}$ and it is denoted $S_{ver,o} = (P_o, i_{P_o})$.

4.2. Training decision trees

As shown in [13, 20, 21, 15, 16], the depth difference comparison is an efficient feature extracted from depth image. All decision trees in this paper utilize the depth difference f defined as follows.

$$f(p|\phi) = z(p + \frac{\delta_1}{I(p)}) - z(p + \frac{\delta_2}{I(p)}) \tag{4}$$

where p is the pixel on foreground, and $\phi = (\delta_1, \delta_2)$ is a set of two randomly sampled 2D offsets, δ_1 and δ_2. The sampled offsets ϕ are compensated by $I(p)$, similar to that in Eq. 2 and Eq. 3. Threshold of depth difference τ is also sampled to partition the pixels into two subsets by comparing τ and their f. Specifically, after $f(p|\phi)$ is evaluated for each training pixel, the pixels whose f is larger than τ, are classified to subset S_l, otherwise to S_r. In conclusion, for each sampled $\theta = (\phi, \tau)$, two subsets, namely, S_l and S_r, are decided.

The training at each node aims to determine parameter θ that optimizes the objective function E. The result of E is defined by the summation of the objective function for each subset, namely, $E(S_l)$ and $E(S_r)$. To balance a tree, we multiply a split ratio for each $E(S_l)$ and $E(S_r)$. The objective function for θ at certain node is defined as follows:

$$E(S, \theta) = \frac{|S_l|}{|S|} E(S_l) + \frac{|S_r|}{|S|} E(S_r). \tag{5}$$

The objective function E varies depending on the type of decision tree. T_{reg} employs sum-of-squared-difference as the objective function [21, 16] to measure how the offsets $\Delta_{p\rightarrow J}$ are converged. To minimize the effects of outliers, we set threshold λ for length of the offset, and exclude the offsets whose length is longer than λ when we evaluate E. λ is empirically decided as 500mm in order to allow T_{reg} to localize occluded joints generally placed further from foreground pixels. Finally, the objective function of E_{reg} is denoted by

$$\begin{aligned} E_{reg}(S) &= \sum_j \sum_{p \in P_j} \|\Delta_{p\rightarrow j} - \mu_j\|_2^2 \\ \mu_j &= \frac{1}{|P_j|} \sum_{p \in S} \Delta_{p\rightarrow j}, \quad \text{and} \\ P_j &= \left\{ p \in S | \, \|\Delta_{p\rightarrow j}\|_2 < \lambda \right\}. \end{aligned} \tag{6}$$

Like [20], for the objective function for the $T_{ver,v}$ and $T_{ver,o}$, the Shannon entropy is utilized like

$$E_{ver}(S) = -\sum_{j} h(j|S) \log h(j|S) \tag{7}$$

where $h(j|S)$ is normalized histogram of the training pixels for a joint j in set S.

After we sample and find θ which optimizes $E(S, \theta)$ for a node, the found θ is assigned at the node as learned parameters. Following two children nodes iterate identical procedures with each of the partitioned training data set, namely S_l and S_r.

4.3. Learing data at leaf nodes

When we visualize $\Delta_{p \to J}$ of the entire training pixels at a leaf node of T_{reg}, the offsets are distributed in 3D space. We utilize mean shift [11] to decide representative clusters of the offsets as the final training results. We evaluate the confidence value of training by analyzing the properties of the cluster. We consider two properties: how many offsets construct the cluster and how the offsets are converged. Hence, the confidence value for each cluster is proportional to the number of the related offset and inversely proportional to the average distance between the center of the cluster and each related offset. The confidence value w_r for joint j is denoted as

$$w_r(j) = |C_j| \frac{|C_j|}{\sum_{\Delta_{c_j} \in C_j} ||C_j - \Delta_{c_j}||_2} \tag{8}$$

where C_j is the cluster and Δ_{c_j} is the offset in C_j.

Like [20], $T_{ver,v}$ and $T_{ver,o}$ utilize the estimated normalized histogram $h(j|S)$ to decide the confidence value of verification w_v for joint j as follows.

$$w_v(j) = h(j|S) \tag{9}$$

4.4. Procedures of the algorithm

We build three types of random forests with the introduced decision trees. We will denote the random regression forest, the random verification forest for visible votes, and the random verification forest for occluded votes as F_{reg}, $F_{ver,v}$, and $F_{ver,o}$ respectively.

The purpose of the algorithm is estimating 3D position of joints using a single depth image. Specifically, the pose estimation algorithm start by casting votes with F_{reg}. The visibility of the votes is decided by ρ_j introduced in table 1. Depending on the visibility, $F_{ver,v}$ or $F_{ver,o}$, verifies the votes with pixels where the votes are projected. The algorithm decides the final weight of the votes by combining the confidence values of the random forests. Finally, considering the final weight, mean shift algorithm estimates clusters of the votes, and the center of each cluster describes the estimation result for a joint. The algorithm 1 is a pseudocode of the proposed algorithm. In following section, we discuss design of weight models which merge the results of each random forests.

Algorithm 1 Estimating 3D human pose

```
 1: for sampled foreground pixel $p$ in depth image $I$ do
 2: //Generate votes
 3:     run $F_{reg}$ to generate votes $v_{p,j}$ and their weight $w_r$
 4: //Verify the votes
 5:     for all $v_{p,j}$ do
 6:         acquire pixel position $p(v_{p,j})$ in $I$
 7:         acquire depth of the position $I(p(v_{p,j}))$
 8:         if $p(v_{p,j})$ is not on foreground then
 9: //Exclude the votes
10:             $W(v_{p,j}) = 0.0$
11: //$\rho_j$ is half of thickness of the joints, so double it
12:         else if $(v_{p,j}).z - I(p(v_{p,j})) \leqq 2\rho_j$ then
13:             run $F_{ver,v}$ and acquire $w_v$
14:         else
15:             run $F_{ver,o}$ and acquire $w_v$
16:         evaluate final weight $W(v_{p,j})$ with $w_r$ and $w_v$
17: //Clustering the votes
18: for all joint $j$ do
19:     perform mean shift with $v_{p,j}$ and $W(v_{p,j})$
20: return 3D position of all joints
```

4.5. Weight models

In order to define the final weight of each votes, we designed two weight models for each $F_{ver,v}$ and $F_{ver,o}$. Firstly, we employ a weight model defined as

$$W_1 = w_r w_v \tag{10}$$

where w_r and w_v are confidence values from F_{reg} and F_{ver} respectively. In Eq. 10, the result of verification w_v directly affects to the final W_1. For instance, if a vote is failed to be verified, W_1 becomes zero, and the vote is excluded when estimating clusters.

When we apply W_1 to $F_{ver,o}$, a significant number of the occluded votes are eliminated, and even some of joints cannot be localized due to a lack of related votes. $F_{ver,o}$ employs indirect verification scheme dissimilar to that of $F_{ver,v}$. Specifically, $F_{ver,o}$ evaluates the votes by using not features of the pixels for the joints, but features of the pixels which occlude the joints. Therefore, we need to design another weight model in order to alleviate the impact of $F_{ver,o}$ on the final weight. The weight model W_2 is denoted by

$$W_2 = w_r(\alpha w_v + (1 - \alpha)) \tag{11}$$

where α is a constant value. We obtain the best α through a prior-experiment. α is decided as 0.99 which maximizes an influence of w_v, and prevents W_2 from becoming zero

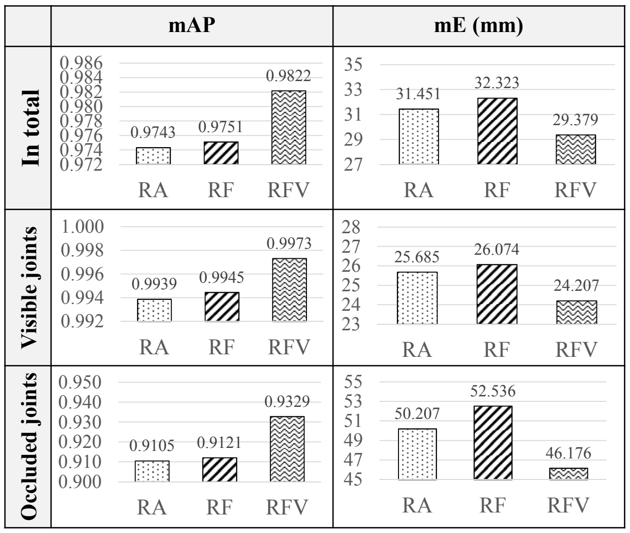


Figure 3. Mean average precision and mean error of the each human pose estimation algorithm in total(first row), in the case of estimating visible joints(second row), and in the case of estimating occluded joints(third row).

by w_v. In conclusion, a decided weight model of $F_{ver,v}$ and $F_{ver,o}$ is W_1 and W_2 respectively.

5. Experiments

We conduct experiments with the golf swing data set introduced in Sec.3. For cross-validation, we apply repeated learning-testing scheme that the data set is iteratively divided into training set and test set, and the data is partitioned by the subject in order to evaluate performance of the algorithm when the algorithm does not train swings of a particular subject. Specifically, we randomly select data of 16 subjects as training set, and the other data of 4 subjects as test set. We perform a couple of iterations to confirm a performance of the algorithm.

We define mean error(mE) as the average distance between estimation results and ground truth joints. If an error is less than 100mm, we determine the estimation as true [13, 20, 21, 15, 16]. The ratio of true cases among entire test cases is defined as mean average precision(mAP). We

type	# of tree	depth	avg # of sampled pixels per frame	#of sampled (ϕ,τ)
F_{reg}	2	20	2000.00	(1500,30)
$F_{ver,v}$	3	19	2171.35	(1000,30)
$F_{ver,o}$	2	18	1228.59	(1000,30)

Table 2. The training parameters for each random forest. In the case of $F_{ver,v}$ and $F_{ver,o}$, the number of sampled pixels of each frame is various because amount of sampled pixels depends on visibility of ground truth joint.

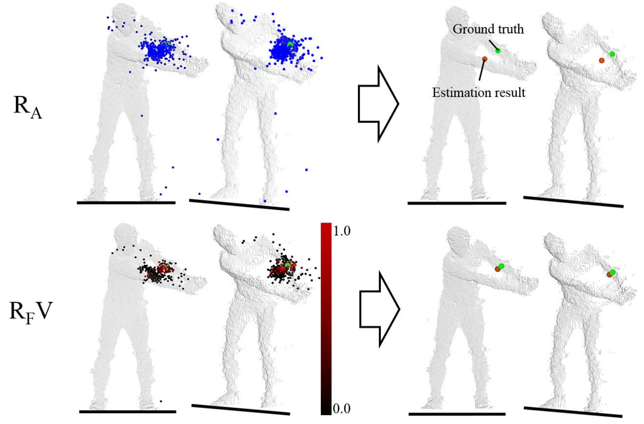


Figure 4. Estimation result of R_A(upper figures) and R_FV(lower figures) to localize a left elbow. The less votes exist in the result of R_FV because some of the votes are eliminated by the verification. The confidence value of votes visualized as color from red to black, and estimation result and ground truth are visualized in orange and green respectively at the figures right side.

run the algorithm on single-core computation with Intel i7-6850K CPU when measuring computation time for the algorithms.

5.1. Experimental setting

Various parameters exist for training each random forests. The training parameters are optimized empirically, and table 2 describes details. As a total, seven decision trees construct the proposed algorithm.

In this section, we monitor performance changes among three types of human pose estimation algorithm as follows:

- R_A: utilize F_{reg} and consider all votes
- R_F: utilize F_{reg} and consider the only votes projected on foreground
- R_FV: utilize F_{reg}, $F_{ver,v}$, and $F_{ver,o}$ which we proposed in this paper.

When we test the algorithms, about 750 pixels in average are sampled from the foreground per each frame, and the pixels traverse the random forests to localize 18 joints of a subject. The following sections will introduce the experimental results with respect to accuracy and computation time.

5.2. mAP and mE

Fig. 3 describes the measured performance of each algorithms in terms of mean average precision and mean error. As shown in first row of Fig. 3, supplementing the verification forests improves overall accuracy of the algorithm. We confirm the improvement in the case of not only visible joints(second row in Fig. 3) but also occluded joints(third row in Fig. 3). The enhancement is even remarkable when estimating occluded joints.

We also verify the effect of the random verification forests by visualizing the votes of R_A and R_FV as shown in Fig. 4. Specifically, as upper images in Fig. 4 illustrates, F_{reg} casts votes to localize left elbow of a user, but the votes are not converged enough to correctly localize ground truth joint(green) by clustering. Meanwhile, as shown in lower images of Fig. 4, when we verify the votes with the random verification forests, the only verified votes(red dot) maintain their weight considered by clustering process. Otherwise, the votes lose their weight. Consequently, mean shift can accurately estimate 3D position of left elbow as shown in right side of Fig. 4.

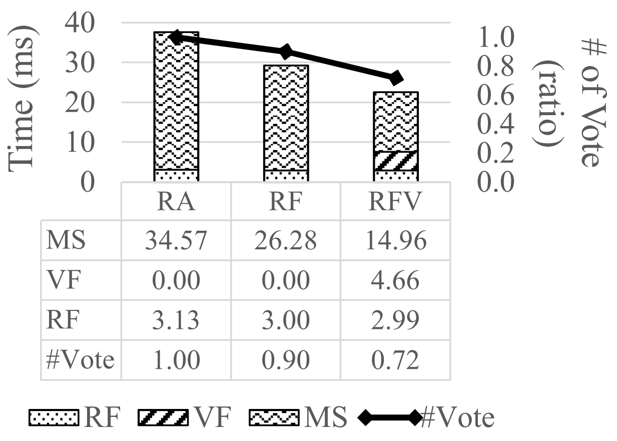
In the result of mean error, the mean error of R_F is increased in the entire cases. We believe that the result shows a side effect of just excluding the votes not in foreground. Specifically, the fundamental assumption of R_A is that a position of joints can be estimated by clustering the entire votes even if each of the votes has an error. However, the result becomes biased when we exclude the only votes out of foreground while the entire votes in foreground are considered whatever errors they have. On the other hand, proposed R_FV eliminates all invalid votes even if the votes stay on foreground, and therefore R_FV solves the bias which R_F has. As a result, R_FV has lower error compared to that of R_A and R_F.

5.3. Computation time

Employing the verification do not always guarantee enhancement in efficiency because the verification could require more time than what we earn by the verification. In order to elucidate the effect of the verification, we separate the algorithms in parts; random regression forest(RF), random verification forest(VF), and mean shift(MS), and measure the spent time of each sub-algorithms.

Fig. 5 shows the number of votes considered in estimation and computation times of each sub-algorithms. For the number of votes, we arrange them as ratio of the number of votes for each algorithm to that of R_A. R_F just decrease only 10% of the votes. The result indicates the number of votes not in foreground is 10% in the entire votes. On the other hand, about 28% of the votes are excluded in R_FV; consequently, for mean shift, R_FV spends less than half time that R_A spends while adding only 4.66ms for the verification. As a result, R_FV spends only 59.97% of the time R_A spent.

The result of the experiment shows that R_FV can efficiently and effectively optimize the votes by utilizing verification forests and improve efficiency of human pose estimation algorithm. When we also consider the improvement in accuracy discussed in section 5.2, we conclude that the



	RA	RF	RFV
MS	34.57	26.28	14.96
VF	0.00	0.00	4.66
RF	3.13	3.00	2.99
#Vote	1.00	0.90	0.72

Figure 5. Comparisons of computation time for each sub-algorithms and the ratio of the number of considered votes to localize the joints between the human pose estimation algorithms.

verification scheme applied to R_FV improves accuracy and efficiency of the human pose estimation algorithm.

6. Prototype system for pose analysis in golf: SWAN

We perform a field study to investigate feasibility of the proposed algorithm for pose analysis in golf. We develop a prototype system called "SWAN" which is abbreviation of "SWing ANalyzer". As Fig. 6 shows, SWAN consists of a display to visualize evaluation results, a depth camera, and a beam projector to show system states.

With the estimated human pose, we implement a swing detection algorithm to automatically notice user's swing and allow the system to proceed evaluations of the swing. When a swing is detected, we extract various information of the swing such as color images at each frame, 3D position of 14 joints[1] at each frame(head, neck, shoulder L/R, elbow L/R, wrist L/R, hip L/R, knee L/R, and ankle L/R), and frame indices of the seven important stages in golf swing(address, take-back, back swing top, down swing, impact, release, and finish).

Fig. 7 shows series of analysis by using estimated pose data. Fig. 7-(a) describes a page to compare the swing of the user to the reference swing performed by a professional golfer. With the extracted frame index for impact, we synchronize both video of the user's swing and the reference swing. The user can examine his/her head movement, knee alignment, swing rhythm, and balance of golf swing in this page. Fig. 7-(b) presents the results of a side view analysis. To acquire the side view of a user, traditional approaches use multiple cameras surrounding the user. Considering the

[1]Although the proposed algorithm can localize 18 joints, only 14 joints are utilized by the functions of SWAN. Four excluded joints are hand L/R and foot L/R.

depth information from the depth camera, we can acquire half of side view of the user, and visualize it as 3D point cloud. Head movement in depth, backbone alignment, and trajectory of wrist in up-swing and down-swing are visualized in this page. Fig. 7-(c) shows the user's pose and its evaluation result at the seven important stages compared to the reference swing. The page shows head movement, arm angle, weight balance, and structures of all 14 joints of a user. We evaluate the pose by comparing it to the reference swing, and visualize evaluation result with color. If serious error a swing has, the color of visualization gradually turns to green, yellow, and red.

We perform a field study by installing SWAN in a golf academy. Test users can conveniently utilize SWAN and check evaluation results of their swing. Among the functions provided by SWAN, the users are especially impressed by functions of visualizing side view for 3D analysis and automatically extracting and evaluating pose at the seven important stages of swing. However, upon taking a swing with a driver club fastest swing in golf, SWAN often misses to detect the impact time because the swing is too fast to be captured by the depth camera which operates in 30 fps.

7. Conclusion and future work

In this paper, we propose an accurate and efficient human pose estimation algorithm that has shown to be effective for pose analysis in sports. The algorithm utilizes a set of random verification forests to verify votes cast by an initial random regression forest for localizing joints. The enhanced conciseness of the votes improves overall performance of localizing joints in terms of accuracy and efficiency. 18 major anatomical joints including occluded ones are localized within 22.61ms with less than 30mm error. Furthermore, we expect the algorithm can be applied to the pose analysis in various applications, because the algorithm does not employs any constraint only for poses in golf.

Field study to demonstrate the feasibility and effectiveness of SWAN were conducted where the system was installed in local golf academy and utilized by golf enthusiasts. Without any additional device but depth camera, SWAN can estimate 3D pose information such as movement of head, trajectory of swing, arm angle, and so on. Moreover, the functions in SWAN is not restricted by the examples we introduced. They can be numerous since raw data of 3D pose is provided by the proposed algorithm.

In order to implement further improved application, we have a plan to merge the algorithm with an existing indoor golf system that accurately analyzes shot information such as shot direction, launching speed, and spin. We expect that the converged system can comprehensively analyze golf swing from the causes (pose) to the consequences (shot result).

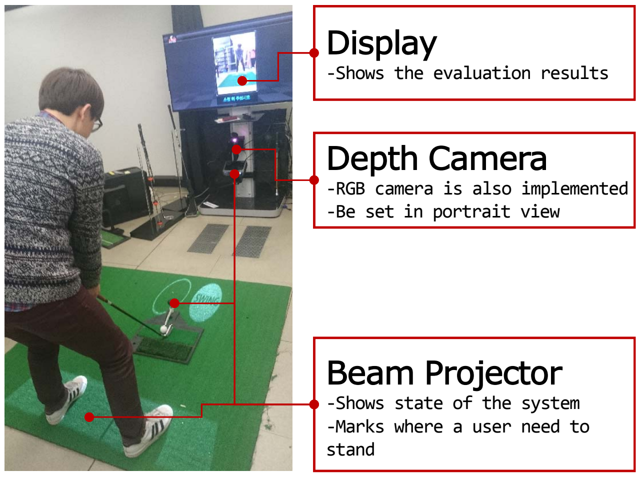
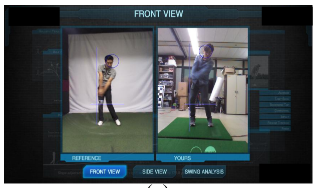
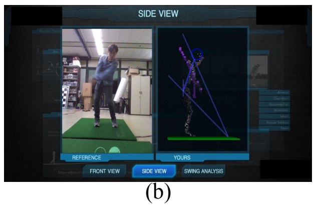

Display
-Shows the evaluation results
Depth Camera
-RGB camera is also implemented
-Be set in portrait view
Beam Projector
-Shows state of the system
-Marks where a user need to stand

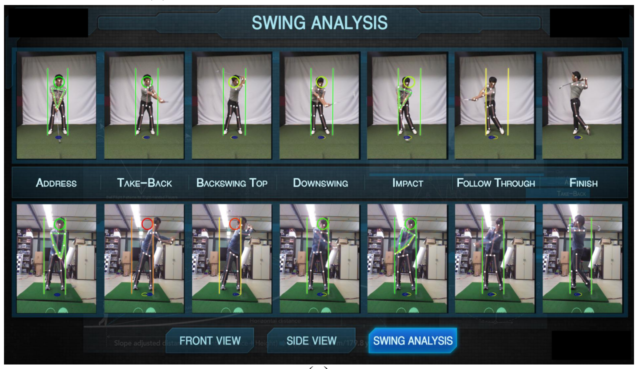
Figure 6. A prototype of SWAN. A user can do swing and evaluate his/her swing by following visualized instructions from SWAN.



(a)



(b)



(c)

Figure 7. Three pages of SWAN. (a) analysis of front view with synchronized video, (b) analysis of side view with visualization of point cloud, and (c) Pose analysis at the seven important stages.

Acknowledgement

This work was supported by the ICT R&D program of MSIP/IIPT. [R0118-17-1004, Development of Intelligent Interaction Technology based on Recognition of User's State and Intention for Digital Life]

References

[1] Gears golf. `https://www.gearssports.com/`. Accessed: 2017-04-23.

[2] K-vest. `http://www.k-vest.com/k-vest/`. Accessed: 2017-03-23.

[3] Microsoft kinect. `http://developer.microsoft.com/en-us/kinectforwindows/`. Accessed: 2017-

03-23.

[4] Optitrack. `http://www.optitrack.com/`. Accessed: 2017-03-23.

[5] Swinguru. `http://www.swinguru.com/`. Accessed: 2017-03-23.

[6] V1pro. `http://www.v1sports.com/`. Accessed: 2017-03-23.

[7] Vicon. `http://www.vicon.com/`. Accessed: 2017-03-23.

[8] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1092–1099. IEEE, 2011.

[9] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1611.08050*, 2016.

[10] J. Y. Chang and S. W. Nam. Fast random-forest-based human pose estimation using a multi-scale and cascade approach. *ETRI Journal*, 35(6):949–959, 2013.

[11] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[12] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real-time human pose tracking from range data. In *European conference on computer vision*, pages 738–751. Springer, 2012.

[13] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 415–422. IEEE, 2011.

[14] Itseez. Open source computer vision library. `https://github.com/itseez/opencv`, 2015.

[15] H. Y. Jung, S. Lee, Y. S. Heo, and I. D. Yun. Random tree walk toward instantaneous 3d human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2467–2474, 2015.

[16] H. Y. Jung, Y. Suh, G. Moon, and K. M. Lee. A sequential approach to 3d human pose estimation: Separation of localization and identification of body joints. In *European Conference on Computer Vision*, pages 747–761. Springer, 2016.

[17] Y.-H. Kwon, C. S. Como, K. Singhal, S. Lee, and K. H. Han. Assessment of planarity of the golf swing based on the functional swing plane of the clubhead and motion planes of the body points. *Sports biomechanics*, 11(2):127–148, 2012.

[18] S. Li and A. B. Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision*, pages 332–347. Springer, 2014.

[19] K. Mitchell, S. Banks, D. Morgan, and H. Sugaya. Shoulder motions during the golf swing in male amateur golfers. *Journal of Orthopaedic & Sports Physical Therapy*, 33(4):196–203, 2003.

[20] 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 a single depth image. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1297–1304. IEEE, 2011.

[21] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, et al. Efficient human pose estimation from single depth images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2821–2840, 2013.

[22] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660, 2014.

[23] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. Accurate 3d pose estimation from a single depth image. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 731–738. IEEE, 2011.

[24] M. Ye and R. Yang. Real-time simultaneous pose and shape estimation for articulated objects using a single depth camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2345–2352, 2014.