Building a skeleton-based 3D body model with angle sensor data

Qihan Wang, Gang Zhou, Zhenming Liu, Bin Ren

Computer Science Department, William & Mary, United States

Abstract

Human body activity recognition is beneficial for health monitoring, and wearable angle sensors are one of the most significant techniques to provide accurate body motion information, like joint angles and bone lengths. However, angle sensor data are partial and separated, which is inconvenient to describe a whole body motion. In the paper, we built a skeleton-based 3D body model to recover whole body motions with angle sensor data. We took advantage of two sub models: a multi-output regression model to expand body information and a skeleton recovery model to produce joint positions. The multi-output regression model included Gradient Boosting and Random Forest learning regressions. The skeleton recovery model made use of the law of cosines and binary quadratic equations. In addition, we developed an Android application prototype to show the practical scenarios of the 3D body model. In the evaluation, we measured the performance of the multi-output regression model by calculating statistic metrics, including mean absolute error (MAE), mean square error (MSE) and R2 score. Moreover, we computed the absolute errors and error percentages of the 3D body model for different joints and different motions. In addition, we evaluated the impact of bone lengths to make the evaluation complete. In total, the accuracy of this 3D body model achieves about eighty percent, while most joints' bias are limited into 10 centimeters in real world datasets.

Keywords: 3D body model, multi-output regressions, Gradient Boosting, Random Forest

1. Introduction

Wearable angle sensors have great significance in health treatments and human activity monitoring. Currently, much research aims to develop more accurate and functional angle sensors to monitor partial body motion status. For example, angle sensors have been designed to measure the gait system[1][2], while a wearable sensor badge and sensor jacket have been developed to measure upper limb and body movement[3]. After applying angle sensors, researchers needed to address the sensor data to recognize body motion. Many researchers implemented related frameworks to monitor the body status based on the sensor data. However, most existing research works focus on new hardware to detect a partial body movement and we need a complete body sensor network covering the whole body to analyze body motions.

Body motion recognition is a significant topic in computer vision, health monitoring and personal assistance. There are some research topics about body motion recognition based on Microsoft Kinect sensors[4], human body monitoring using wearable sensors[5][6], and building abstract body models based on adequate image data[7]. Microsoft Kinect sensors provide a number of image data to build body models or recognize body motions. However, the accuracy depends on the image pre-processing. Thus image-based recognition is complete but unstable, while sensor-based body recognition is stable, precise but incomplete. Our work is the first one to build a complete 3D body model based on the angle sensor data.

Since the angle sensors provide accurate but separated partial data, the main challenge is how to build a whole body model using limited data. To be more specific, the data produced from wearable sensors mainly consists of joint angles. For example, Fiber-optic sensors[8] built a low-cost wearable system to monitor the joint angles during a

Email addresses: qwang19@cs.wm.edu (Qihan Wang), gzhou@cs.wm.edu (Gang Zhou), lzhenming@gmail.com (Zhenming Liu), bren@cs.wm.edu (Bin Ren)

motion. Although these joint angles are able to cover the whole body, we do not know the relative positions between two joints. Therefore, our goal is to combine these separated data. Moreover, these angles are 2D angles, while the body motion is 3D. Thus, another question is how to use 2D angles to build a 3D body model. In this paper, we present a skeleton-based 3D body model based on the angle sensor data. We defined a skeleton-based action graph to represent the human body motion, then we made use of multi-outputs learning regressions to expand limited body information and utilized binary quadratic equations to recover the human body.

In the paper, we built a skeleton-based 3D body model to transform the angle sensor data into a constructed skeleton-based human body motion abstraction. The model produced the positions of human joints and skeletons, which then were used to form a motion. The skeleton-based 3D body model consisted of two sub-models: a multi-output regression model and a skeleton recovery model.

In order to expand the limited body information, we built a multi-output learning regression model. Joint angles are inputs, while the expected outputs are joint depths, like the depth images from Kinect sensors[7]. This problem was defined as a multi-output regression. Thus, we tested four classic regression models to be the candidates, compared their performance, and selected two of them to be applied in our 3D body model. They were Gradient Boosting and Random Forest learning regression models. We adjusted parameters, trained models, and analyzed overfitting and unbalanced categories problems.

After generating depth information, we built a skeleton recovery model. We utilized the law of cosines and binary quadratic equations to recover the skeleton abstraction from the joint angles and depths.

Moreover, we implemented an Android application prototype to show the skeleton 3D body model. According to this prototype, we analyzed practical scenarios, including visualizing real time sensor data, monitoring human activities and providing benefits for motion recognitions.

In the following sections, we introduce implementation and evaluation of our 3D body model. Section 2 introduces the whole picture of the skeleton-based body model. Section 3 and Section 4 introduce the multi-output regression model and skeleton recovery model. Section 5 introduces an Android application prototype. Section 6 introduces an evaluation to test the accuracy of the model. Section 7 discusses findings, limitations and future work. Section 8 gives an introduction of related works.

2. Skeleton-based Body Model

In this section, we introduce the whole concept of the skeleton-based body model. Firstly, we show the body skeleton abstraction. Next, we explain all the definitions in our model. Moreover, we demonstrate the model design including the input data, output data and two sub-models. Finally, we discuss the datasets of our model.

2.1. Body Skeleton Abstraction

A number of model-based methods are used to represent the human body, combining motion and shape information. The models include mesh models[9], cylinder based models[10], ellipsoids based models[5] and skeleton-based models[11]. The first three types of models are complicated, due to their inclusion of related shape information. But shape information like skin texture and muscle size is not necessary to describe a body motion. Additionally, limited wearable sensor data only provides joint angles and bone lengths. Considering both the limited input data and the efficient representative of body motion, the skeleton-based model is the best choice.

In this paper, the skeleton-based human body abstraction consisted of twenty joints, nineteen bones and eighteen joint angles. This body abstraction which came from the Microsoft 3D Motion Datasets[7], is more complicated than the 9 joints model[11]. Twenty joints are able to cover all important joints of the human body, including shoulders, elbows, hips and knees.

According to Figure 1, joints are represented as points. Joint angles are angles between two bones around one joint. For example, point C is the neck, point B is the right shoulder, and point I is the right elbow. Joint angle $\angle CBI$ is the right shoulder angle.

2.2. Definitions

Action graph[12] is a widely used body abstraction to represent dynamic human motions. According to the original action graph, we built a skeleton-based action graph to describe one motion in our 3D body model. Define

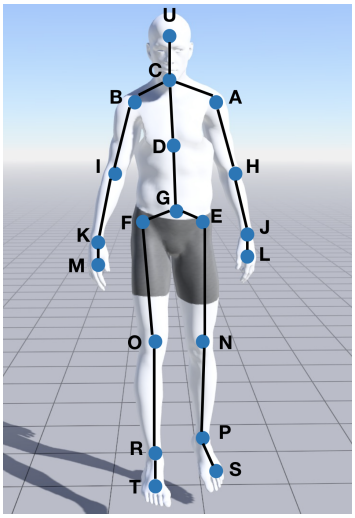

Figure 1: Angle data detection in wearable sensors

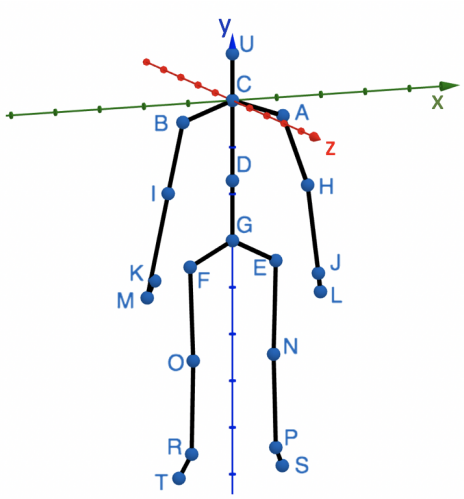

Figure 2: Transfer data to a mobile application

G as a skeleton-based action graph, which consists of three subsets: C, P and S. Set P is a set of postures, set S represents a set of bone lengths and set C describes the positions of the body center. G is represented by a triplet as follows:

$$G = \{C, P, S\} \tag{1}$$

$$C = \{c_x, c_y, c_z\},\ P = \{P_1, P_2, ..., P_h\},\ S = \{s_1, s_2, ..., s_k\} \tag{2}$$

To locate the position of every joint, we needed to build a 3D coordinate system. In Figure 2, the width of the body is the direction of the x axis, like left and right movements. The height of the body is the y axis, like up and down movements. The depth of the body joint in the 3D coordinate system is represented by the z axis, like forward and backward movements.

We defined body center to be the origin of the 3D coordinate system. The body center is one of the joints, which represents the center of the entire body. Set C is a set that describes the position of the body center. In the paper, we set the neck joint to be the body center, since it connects to all other four joints. Setting the neck joint to be the coordinate system origin can simplify the following computations. In set C, there are x, y, and z axis values to represent the position of the body center. Thus, $c_x = 0$, $c_y = 0$, and $c_z = 0$. In Section 4, we introduce how to use the center joint and bone lengths to calculate all other joints' positions.

Set P is a set containing postures of one motion. P has h subsets to describe one motion. According to the Microsoft 3D Motion datasets[3], there are 54 frames of the Kinect Camera that form one motion. Therefore, our model also utilized 54 continuous postures to describe a motion, then $h = 54$. We set i to be one moment during a motion. For moment i, the positions of all joints are in the set P_i, where $0 < i \leq h$.

Set S records all bone lengths in our abstract of body skeleton. We set the number of bones to be k, so set S consists of s_i, where $0 < i \leq k$. For one specific body, the bone lengths are fixed and accurate in our model. Based on Figure 1, there are nineteen segments in the body, then $k = 19$. We explain more about bone lengths in Section 6.3.

At a moment i, set P_i consists of four subsets: A_i, X_i, Y_i, Z_i. Joints are points, while joint angles are angles between two segments around one joint. Set A_i is a set of joint angles, which contains all the joint angles in the entire body. To describe the joint positions, we utilize X_i, Y_i and Z_i to represent x, y, and z axis values of joint locations. Here, $0 < i \leq h$. For moment i, the posture P_i can be described as a quadruplet:

$$P_i = \{A_i, X_i, Y_i, Z_i\} \tag{3}$$

$$A_i = \{a_1^i, a_2^i, ..., a_m^i\},\ X_i = \{x_1^i, x_2^i, ..., x_n^i\} \tag{4}$$

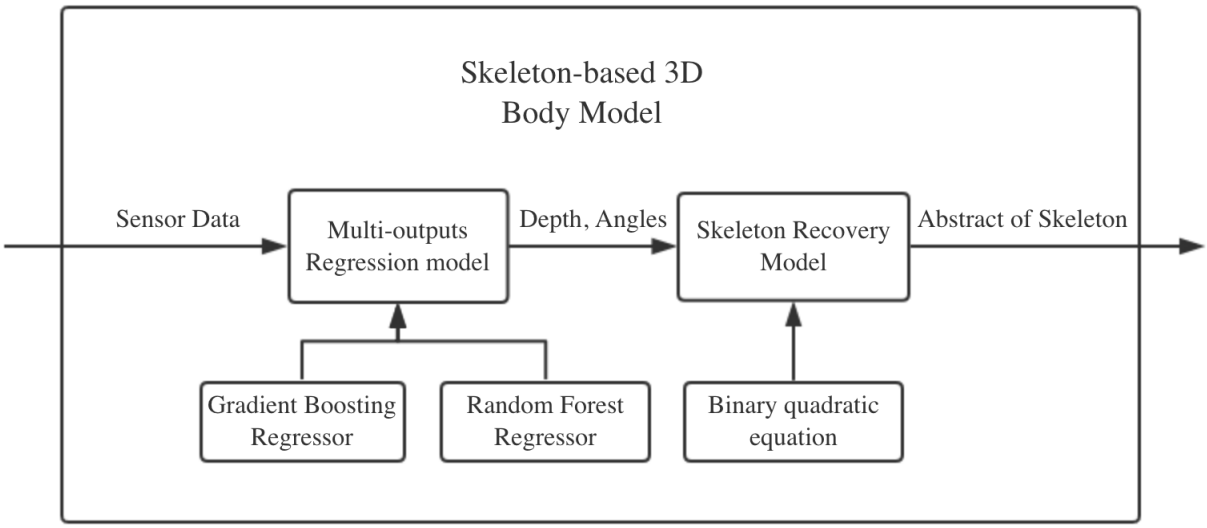

Figure 3: Construction of the skeleton-based model

$$Y_i = \{y_1^i, y_2^i, ..., y_n^i\},\ Z_i = \{z_1^i, z_2^i, ..., z_n^i\} \tag{5}$$

From Figure 1, we know that there are 20 joints and 18 joint angles in this skeleton. We use n to be the number of joints, and use m to represent the number of joint angles. Thus, $n = 20$ and $m = 18$. We use x_j^i to represent the x axis value for joint j at the moment i. y_j^i, and z_j^i are the same. In section 4.2, we introduce the solutions of computing x, y, and z values.

There exists another important definition called depth. Dep is a set of joint depths. Since the bone lengths will not change during the motion, only angles affect the motion type and depth information. In order to be independent of the bone lengths, the joint depths are defined as angle values, instead of z axis values. The definition of Dep is:

$$Dep = \{Dep_1, Dep_2, ..., Dep_h\},\ Dep_i = \{dep_1^i, dep_2^i, ..., dep_n^i\} \tag{6}$$

$$dep_j^i = arcsin(\frac{s_j}{z_j^i}) \tag{7}$$

Set Dep consists of all the subsets Dep_i and $0 < i \leq h$ during a motion. There are n joints in the body. One joint is related to one depth value, thus there are n joint depths in the body. In Section 3, we introduce the basic theory of learning regressions, and how to utilize joint angles to predict joint depths.

2.3. Design Overview

The skeleton-based 3D body model is shown in Figure 3. For the 3D body model, the input data includes set joint angles A, center position C and bone lengths S. Then, the output data is all the other data X, Y, Z and posture set P.

The whole body model consists of two sub-models. One is a multi-output regression model, mainly used to calculate the depth of every joint. The input data is the angle set A, while the output is depth set Dep. For the 3D body model, the added dimension is the z-axis. All the sensors could give us 2D angles of one joint during a motion. Thus, we need to put these input angles in a learning regression model in order to learn the joint depths. The key point here is the multi-output, for which the traditional regression model cannot work. Therefore, after trying several regression methods, we selected two of them to achieve the goal. They are Gradient Boosting and Random Forest models.

The other sub-model is the skeleton recovery model. After obtaining set Dep, A and S, we combined them to reconstruct the whole body skeleton. We took advantage of the law of cosines and binary quadratic equations to

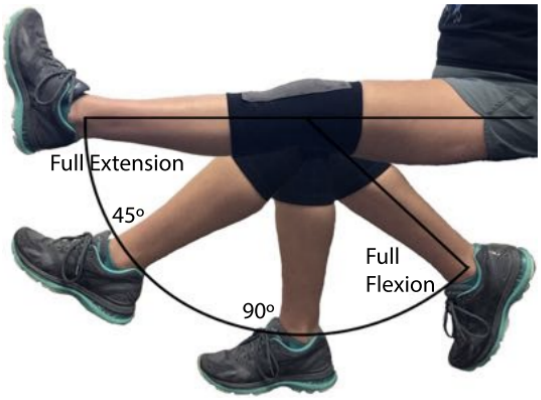

Figure 4: Angle data detection in wearable sensors

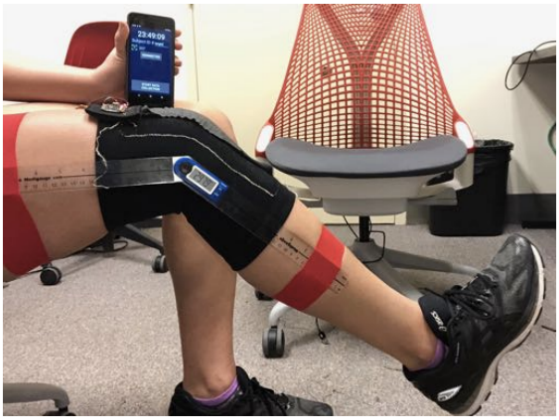

Figure 5: Transfer data to a mobile application

calculate the positions of every joints, and fix the skeleton positions, including x, y, and z values. During a motion, as long as we know several angles and bone lengths, we can produce a sequence of skeleton positions to form a motion.

2.4. Datasets

In this section, we mainly introduce how to pre-process the input data of our model. Firstly, we make use of an angle sensor example to show how angle sensors work. We explain the output data of angle sensors, and demonstrate the relationship between angle sensors and our body model. Next, we explain how we selected the 3D Microsoft Motion Datasets to train and test the body model. Finally, we show the simulation of angle sensor data when using the 3D Microsoft Motion Datasets. This section aims to discuss the input data preparation of our model.

2.4.1. Angle sensors

After exploring the current research[13] of wearable sensors, we have great interest in building a body model based on the wearable sensor data. However, current wearable sensors only generate angle data on the partial human body like knees[13]. Figure 4 and Figure 5 come from the Tracknee paper[13], and show an example of wearable angle sensors around knees. In Figure 4, users can wear angle sensors when moving their legs. Real-time knee angles can be detected from the wearable sensors. Figure 5 shows the process of transferring angle sensor data from wearable sensors to a mobile application. Our model can be applied in a mobile application, in order to visualize body motions.

Angle sensors produce joint angles, while the Kinect camera generates joint positions. Thus, our model aims to use joint angles to calculate joint positions. In addition, our model can also confirm that wearable sensor data are able to provide the same or more information when compared with Kinect camera data. Although we do not have wearable sensors to measure the entire body, our model can be applied with wearable sensors in the future. We focus on building a 3D body model instead of hardware developments and in this section we do a brief introduction of angle sensors.

2.4.2. Simulating angle sensor data

Since there is no sensor data, we need to simulate this data to be the input of our body model. We utilized Microsoft 3D Motion datasets[7] to calculate the joint angles. There are two skeleton datasets in Microsoft 3D Motion datasets[7]. In the 3D Microsoft Motion Datasets, they provided two sub-datasets named real world data and camera-based data. The two datasets recorded ten different people performing twenty motions. They utilized fifty-four frames to describe one motion, and provided x, y, and z positions of twenty body joints based on the fixed external coordinate system. We selected the real world skeleton datasets, which are closer to angle sensor data. There are ten people with twenty motions measured in the datasets, which we considered as ten sub-datasets. However, some sub-datasets missed important information. For example, sub-dataset 4 only contains four motions. Some other sub-datasets have low confidences. For instance, the sub-dataset 7 has five joints with zero confidence and only nine joints with high

confidence (larger than 0.7). Considering we need to use these data to simulate angle sensor data, their confidence will directly affect the accuracy of our model. Therefore, in view of the completeness, confidence and generalization, we selected two sub-datasets from the whole 3D Microsoft Motion datasets. More details will be explained in Section 6.

Mircosoft 3D Motion datasets provide the position of every joint, including x, y, and z axis values. We used these data to produce joint angles and bone lengths in the pre-process. Figure 6 shows an example of the right shoulder joint. According to Section 2, we knew that Point C is the center joint of the body, Point B is the right shoulder joint and Point I is the right elbow. The positions of these three joints can be described as set $\{x_c, y_c, z_c\}$, $\{x_b, y_b, z_b\}$ and $\{x_i, y_i, z_i\}$. Then we calculated the bone lengths CB and BI, as well as the joint angle $\angle CBI$.

The calculations of CB, BI are:

$$|CB| = \sqrt{(x_c - x_b)^2 + (y_c - y_b)^2 + (z_c - z_b)^2} \quad (8)$$

$$|BI| = \sqrt{(x_b - x_i)^2 + (y_b - y_i)^2 + (z_b - z_i)^2} \quad (9)$$

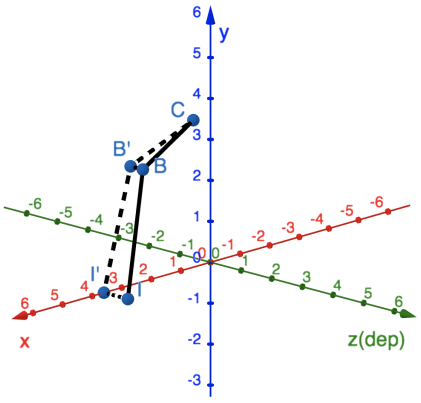

Figure 6: Shoulder example in 3D

Since we knew the x, y, and z axis values of C, B and I, we could directly calculate the distance between every two points.

We computed CI as follows:

$$|CI| = \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2 + (z_c - z_i)^2} \quad (10)$$

Then we utilized the law of cosines to calculate Angle $\angle CBI$:

$$|CI| = \sqrt{|CB|^2 + |BI|^2 - 2 \times |CB| \times |BI| \times \angle CBI} \quad (11)$$

$$\angle CBI = \frac{|CB|^2 + |BI|^2 - |CI|^2}{2 \times |CB| \times |BI|} \quad (12)$$

After the pre-process, we would get joint angles and bone lengths, which are set A and set S, respectively. These two sets will be the input of the body model.

3. Multi-output Regression Model

3.1. Overview

A basic regression model often supports only one output, of which the goal is to predict a continuous value. In the paper, we needed to produce all 20 joint depths at one time which is defined as a multi-output regression problem. Since the training data is attached with the ground truth, this is a supervised learning problem. The input data are joint angels, while the output data are joint depths. Before testing learning regression models, we needed to confirm the correlation between the inputs and the outputs, which is introduced in Section 3.2. Then we selected four classic regression models to test including linear regression, K-nearest neighbor (KNN) regression, Gradient boosting and Random forest. The first two regression models had relatively low accuracy. We focused on introducing the last two models in Section 3.4 and Section 3.5. Additionally, we analyze overfitting and unbalanced problems in Section 3.6. We adjusted the parameters to eliminate these problems, including increasing the test data size and decreasing the training iterations.

3.2. Correlation between joints and angles

Table 1: Descriptions of joint angles

Angle number	Angle in figure 1	Description
Angle 1	$\angle UCB$	Angle between right side of neck and right shoulder
Angle 2	$\angle UCA$	Angle between left side of neck and left shoulder
Angle 3	$\angle CBI$	Right shoulder joint angle
Angle 4	$\angle CAH$	Left shoulder joint angle
Angle 5	$\angle BIK$	Right elbow joint angle
Angle 6	$\angle AHJ$	Left elbow joint angle
Angle 7	$\angle IKM$	Right wrist joint angle
Angle 8	$\angle HJL$	Left wrist joint angle
Angle 9	$\angle CDG$	Angle of the joint in the middle of the spine
Angle 10	$\angle DGF$	Angle between the spine and right hip
Angle 11	$\angle DGE$	Angle between the spine and left hip
Angle 12	$\angle FGE$	Angle between right hip and left hip
Angle 13	$\angle GFO$	Right hip joint angle
Angle 14	$\angle GEN$	Left hip joint angle
Angle 15	$\angle FOR$	Right knee joint angle
Angle 16	$\angle ENP$	Left knee joint angle
Angle 17	$\angle ORT$	Right ankle joint angle
Angle 18	$\angle NPS$	Left ankle joint angle

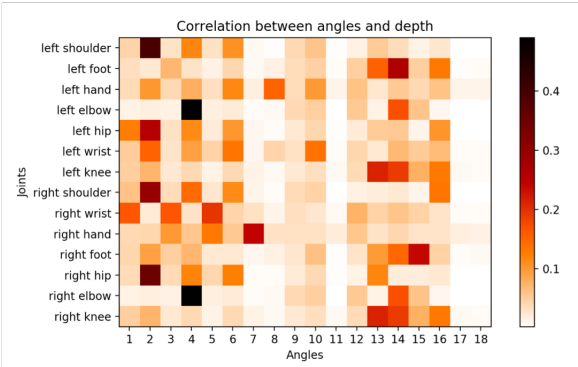

Figure 7: Correlation heatmap

Before starting regressions, it is necessary to confirm if there exists correlations between input data and output data. According to observations in daily life, joint angles should affect joint depths and body motions. To confirm this, we made use of the random forest regression model to calculate the feature importance. The feature importance measures how important joint angles are when predicting the depth values in the random forest regression. Figure 7 shows the correlation between angles and depths. The color means the correlation degree. When the color is deeper, the relationship between joint angles and joint depths is stronger. The range of the correlation degree is from 0 to 1. According to Figure 7, most of the correlation degrees between angles and depths are from 0.2 to 0.4. In the random forest regression, the importance measures[14] the mean of the regression trees' error reduction in the impurity criterion. Each independent variable has a feature importance. In our random forest regression model, the impurity criterion is Mean Square Error (MSE). Since it is not easy to describe a joint angle in the human body, we assigned each angle a number and detailed where the angle occurs in Table 1.

In Figure 7 and Table 1, we can see that Angle 2 comprises the left side of the neck and shoulder. Thus, the importance of the left shoulder joint depth is more than 40%. Another example is the left elbow. The correlation between the left elbow joint depth and left shoulder joint angle is about 50%. One interesting finding is the correlation between the right hip and Angle 2. Although Angle 2 involves the neck and shoulder, it affects the right hip depth by about 35%. Thus, we cannot simply think that angles only have correlation with close joints. Some other angles far from one joint may also affect this joint depth. Another finding is that nearly all the depths have correlations to all the angles. Most correlations range from 5% to 10%. Thus, we took all the correlations into consideration to avoid overfitting problems. Even if some correlations are not strong, we still were able to use all angles to predict depths.

3.3. Selection of regression model

For a multi-output regression model[15], the key idea is to combine several regression models together. For all learning regression models, gradient boosting and random forest are two popular ensemble models. They are

Table 2: Comparisons of four regression models

Regression model	R2 Score
Linear Regression	0.4273661
KNN Regression	0.5419863
Gradient Boosting	0.8131888
Random Forest	0.7916121

originally suitable for multi-output regressions, since they consist of several learners or regression trees, to implement the prediction together. Also generalization is the main advantage of the two ensemble models. Both bias and variance can be reduced. Although knowing their strengths, we still analyzed their generalization and measured the accuracy to confirm if they were suitable to solve our specific body model problem. To make it more complete, we tested another two classic regression models, linear and KNN models, which also supported multi-output regressions.

The results of the four tested regressions are shown in Table 2. We utilized Data 1 to test the four regression models. We calculated an R2 Score to evaluate their performance. We explain more about the R2 score in Section 6.1.2. Linear regression and KNN were dropped due to the low accuracy, which was about 50%. In this section, we briefly analyze the reasons why these two regressions failed.

Linear regression is based on the assumption that the relationship between angles and depths is linear. The correlation equation is shown below. For moment i, all the joint angles a_k^i compute one joint depth dep_j^i. Every independent value a_k^i has a related weight value w_k, where $0 < k \leq n$.

$$dep_j^i = w_1 a_1^i + w_2 a_2^i + ... + w_n a_n^i \tag{13}$$

Linear regression aims to train the values of weights w_i, but test results in Table 2 show that the correlation between angles and depths is not simply linear. A basic linear relation can not solve this problem. Even if linear regression could be expanded to a_i^2 or $log(a_i)$, the prediction still did not work well.

KNN regression is another regression model, used to solve the multi-output problems. The main idea of KNN is to calculate the distances between a new input point and all the existing sample points. After computing the distances and the average points' values, the model can predict the value of the new one. Additionally, KNN is able to deal with the non-linear regression problems. According to the face completion example in sciki-learn documents[16], KNN shows good multi-output regression results to process images. Here we utilized standard Euclidean distance, the most common one to test. Unfortunately, KNN is still not suitable to predict depth values based on the joint angles in the human body.

We summarize some reasons as to why KNN is not suitable: one is that it lacks independent variables. In the KNN problem, every independent variable is in one dimension and all independent variables form a multi-dimensional space. The related dependent variable is one point in the space with an output value. In image classification or regression, independent variables are the features of images, like pixel information. There are often more than 1000 features in images. Compared with images, the number of joint angles in body is only 18, which is much less than image features. Thus, having only 18 features makes it too difficult for KNN to do an accurate regression, especially a multi-output regression. Other potential reasons may include unbalanced categories, sensitivity of irrelevant features and not enough samples.

3.4. Gradient Boosting

Gradient boosting tree algorithms show great results in multi-output regressions[17][18]. The model consists of several weak regression models, which are typically decision trees. Prediction results from these decision trees will combine together to produce the outputs.

We define a training set $A' = a'_1, a'_2, ... a'_{m'}$, where m' is the number of training data. A' is a subset of A. In our model, we randomly selected eighty percent of all data in the original set A to be the training set A'. That is to say, m' equals to the floor of $0.8 * m$. Similarly, the output set is $\hat{Dep} = \hat{dep}_1, \hat{dep}_2, ..., \hat{dep}_{n'}$. Since there are n Dep sets, n' equals to the floor of $0.8 * n$. At moment j, the joint depth value is $\hat{dep}_j$, where $0 < j \leq n'$.

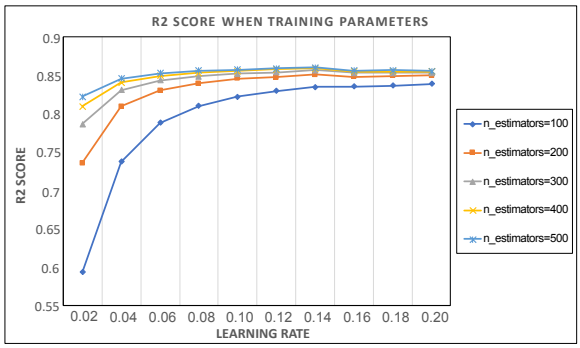

Figure 8: Training learning rate in gradient boosting

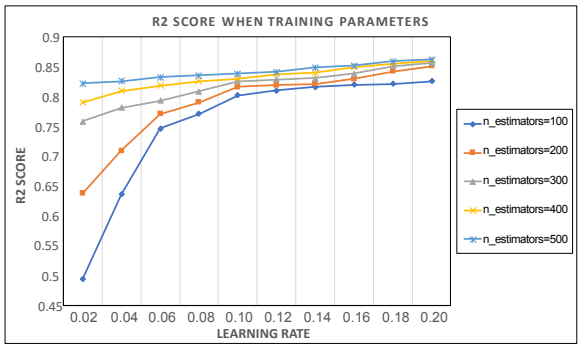

Figure 9: Training learning rate in random forest

There are B stages in the model. Every stage is b, $0 < b \leq B$. For every stage b, the model adds an estimator E to improve the model: $F_{b+1}(a'_i) = F_b(a'_i) + E(a'_i)$, where $0 < i \leq m'$. Therefore, giving $E(a'_i)$ a suitable value is important. The gradient boosting model aims to fit E to the $\hat{dep}_j - F_b(a'_i)$, $0 < j \leq n'$. We used L to define loss function. As for the multi-output regression, we utilized mean square error to be the loss value[19][20].

To be more specific, the gradient boosting model could be described in the following way:

$$F_b(a'_i) = F_{b-1}(a'_i) + \nu \cdot \gamma_b E_b(a'_i) \tag{14}$$

$$\gamma_b = argmin \Sigma_{j=1}^{n'} L(\hat{dep}_j, F_{b-1}(a'_i) + \gamma E_b(a'_i)) \tag{15}$$

$argmin$ means the values of $\hat{dep}_j$ and $F_{b-1}(a'_i) + \gamma E_b(a'_i)$ when minimizing the loss function L. γ_b is some value to minimize the loss function L, while ν is learning rate, $0 < \nu < 1$. The learning rate is to control learning improvements. Setting a proper learning rate is important. If the learning rate is too large, the mode will be overfitting. Meanwhile, if the learning rate is too small, the model will slowly improve and may not find the optimal solution. Thus, we needed to train this parameter to find the best value.

Before using the datasets to train the model, we needed to adjust parameter values to balance accuracy and overfitting. Since the amount of data is no more than 3500, there is no need to adjust some parameters like the maximum depth of the decision trees. Since the input features are only 18 joint angles, the maximum number of features is also not needed to be trained, but there are two important parameters to train. They are n_estimators and learning rate ν. n_estimators is the number of stages, defined as B before. Figure 8 shows the process of training learning rate when changing n_estimators. We can see that as the values of n_estimators and learning rate increases, the R2 score is increasing until the learning rate is 0.15. After that, the R2 score becomes stable and even lower. Thus, we set the learning rate to be 0.15. The light blue line is very close to the yellow line. To avoid overfitting, the parameter n_estimators was set to be 400.

3.5. Random Forest

Random Forest is another ensemble regression model[21][22], which is suitable to address multi-output nonlinear problems. It is also a bootstrap sample method, constructing multiple regression trees. Each regression tree is grown using a randomized subset of predictors. The basic idea of random forest regression is the feature bagging method[23]. As described in Section 3.4, the set $\hat{Dep}$ is predicted by the set A'. We let the bagging repeat B times, and the model is F_b at a stage b. The average of the values in set $\hat{Dep}$ is AVR_Dep. Here is the definition of AVR_Dep:

$$AVR\_Dep = \frac{1}{B} \Sigma_{b=1}^{B} F_b(a'_i) \tag{16}$$

Then the standard deviation σ of all the trees' predictions can be used to estimate the uncertainty:

$$\sigma = \sqrt{\frac{\Sigma_{b=1}^{B} (F_b(a'_i) - AVR\_Dep)^2}{B-1}} \tag{17}$$

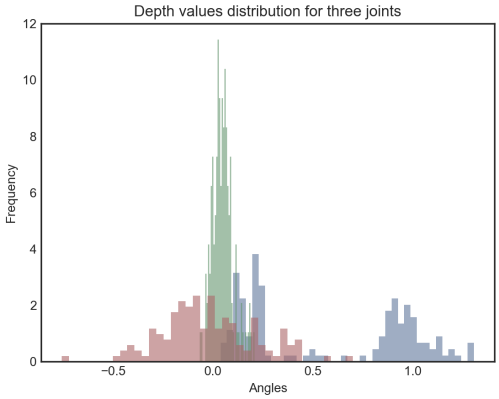

Figure 10: Depth distribution for upper body joints

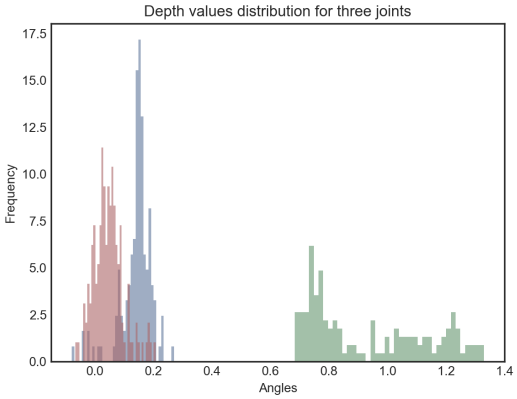

Figure 11: Depth distribution for lower body joints

The difference between bagging and random forest lies in the random selection of a subset of all the predictors, in order to find the best split of nodes[21]. Since the random forest provides the measurements of bias using the out of bag samples, there is no need to do the extra cross validation for random forest models. The bootstrapping procedure can reduce variance without increasing the bias when diminishing correlation among unpruned trees. We cannot control or test individual trees separately in the forest, but only adjust all the trees together. Thus, we trained the n_estimators and the learning rate in a similar way. According to Figure 9, the R2 score continues increasing as the learning rate becomes larger. Therefore, we set the learning rate to be 0.2, while the n_estimators was 500 to reach the peak accuracy.

3.6. Unbalanced categories and overfitting

1

Figure 10 and 11 show the distribution of depth values when randomly selecting three joints. For some joints, the distribution is separated and balanced. However, some joints may mainly present a small range of values. This leads to an aggregation of angle depth frequency, causing an apparent accuracy of prior predictions, which is nonetheless inaccurate. One explanation is, for some joints, the depths in some motions are truly limited to a small range. The results can not be directly considered wrong. Another way to eliminate this problem is to reduce the learning rate, and enlarge the testing datasets. These two methods are also suitable for the overfitting problem. Overfitting means the model can only predict very well using the input training data, but achieves low accuracy for other data. In the random forest problem, we also use OOB Score to measure the model. OOB Score represents the out of bags data prediction accuracy. Since the model is based on the bagging methods, there exists some data not used in the training. If there are N samplings, the probability of OOB is $(1-\frac{1}{N})^N$. When N is large:

$$(1-\frac{1}{N})^N = \frac{1}{(\frac{N}{N-1})^N} = \frac{1}{(1+\frac{1}{N-1})^N} \approx \frac{1}{e} \tag{18}$$

Thus, the OOB data makes up about 1/3 of the whole training data. These OOB data can be considered as new data to measure the overfitting problem. The OOB Score we measured is 0.768982, which achieves similar accuracy as to R2 score. Thus, we can conclude that the generalization of the random forest regression model is good.

4. Skeleton Recovery Model

In our skeleton recovery model, input data includes bone lengths, joint angles and joint depths, while output data are joint positions. Similar to how we computed in Section 2.2, we could calculate z axis values in an opposite way. Thus, in this section, we focus on calculating x and y positions of every joint. We used dimensional reduction to

generate a projection from a three-dimensional space to a two-dimensional plane. Additionally, the problem was translated to solve two binary quadratic equations. The equations produced two solutions and we utilized joint depths to select the correct one.

Take the three joints in Figures 6 and 12 as an example, we show how to calculate x and y axis values of Joint I below. We defined the input set to be IN and the output set to be OUT. x_c, y_c, x_c, y_b are x-axis and y-axis values. dep_b, dep_i are depth values. Then s_{bi}, s_{cb} are two segments between three joints. $angle_b$ is the $\angle CBI$.

The definitions of IN and OUT are:

$$IN = \{x_c, y_c, d_c, x_b, y_b, dep_b, dep_i, s_{cb}, s_{bi}, angle_b\} \tag{19}$$

$$OUT = \{x_i, y_i\} \tag{20}$$

Moreover, we reduced the 3D object to a 2D object by making predictions on the x-y axis plane. In Figure 6, we can see that points B' and I' are the projections of B and I in the x-y axis plane, while joint C is in the plane. We computed the lengths $C'B'$ and CI' based on the dep_b and dep_i.

In addition, we can build binary quadratic equations on two circles in Figure 12. One circle has center B' and radius $B'I'$, while the other circle has the center C and radius CI'. The positions of B' and C are given, and we have calculated CB' and $B'I'$. The intersection point of the two circles will be the I'. Therefore, we can build binary quadratic equations and the solutions are positions of two intersection points. Although equations produce two solution candidates, we can select the correct one based on the joint depths of I.

The segments $C'B$, CI' and CI' are calculated as follows:

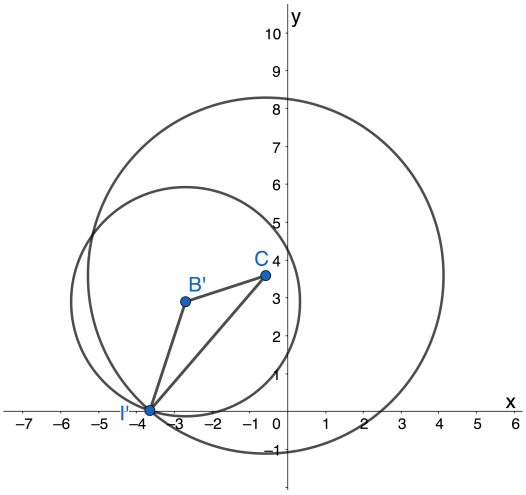

Figure 12: Shoulder example in 2D

$$|C'B'| = \sqrt{{s_{cb}}^2 - (sin(dep_i) * s_{bi})^2} \tag{21}$$

$$|CI'| = \sqrt{{s_{cb}}^2 + {s_{bi}}^2 - 2 \times s_{cb} \times s_{bi} \times \angle CBI} \tag{22}$$

$$|CI'| = \sqrt{|CI|^2 - (sin(dep_b) * s_{cb} + sin(dep_i) * s_{bi})^2} \tag{23}$$

The two binary quadratic equations can be constructed as:

$$(x_c - x_b)^2 + (y_c - y_b)^2 = |B'C'|^2 \tag{24}$$

$$(x_c - x_a)^2 + (y_c - y_a)^2 = |AC'|^2 \tag{25}$$

The prerequisites are the two existing points' positions in this example. As for the whole body, we need to know head and neck positions to calculate all other joints. The center position in the bottom of the neck is given as an input in our model. If we know the center position set C, the length of the neck, the right and left side angles as Angle 1 and Angle 2 in Table 1, then the position of the head can be computed as explained above. After that, the 3D body model can generate all other 18 joint positions by building several binary quadratic equations.

5. Android Application Prototype

5.1. Implementation

We implemented a simple Android Application prototype, in order to show the 3D body model in applications and explain real application scenarios. This Android application is implemented and simulated in the Android Studio platform. Figure 13 shows a welcome page, while Figures 14, 15 and 16 show three body motions in the generated

Figure 13: Welcome page

Figure 14: Standing

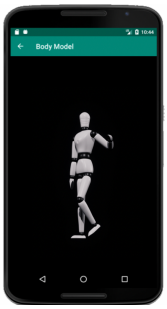

Figure 15: Waving hands

Figure 16: Picking up

body page in 3D views. The welcome page is designed to enter input data, and the body model page generates 3D human body skeletons. In the prototype application, users can enter a joint number and its angle. The bone lengths have been already set and cannot change. When clicking the submit button on the welcome page, the application will jump to the body motion page. We utilized a free application called Manikin to draw 3D body postures of all motions, then applied their figures in our application. These 3D figures consist of points and segments, which are suitable to show skeleton-based body motions that are defined in our body model.

However, this is only an application prototype. We did not integrate our model completely into the application. Since the application aims to show practical scenarios, a prototype is enough to achieve this goal, which is not the key part of our work. In addition, our work is based on the simulation of wearable sensor data of the whole body. Currently whole body wearable sensors have not been developed, so cannot obtain real input data from angle sensors for our model. Therefore, the application can not be implemented completely by the limited input data.

5.2. Potential applications

One of the most important practical scenarios is visualization. The main function of the 3D body model is to transform the real time angle sensor data to skeleton body motions. Since the human skeleton can be shown dynamically on a smart phone, we can do more work like motion recognition or monitoring human activities. A real time body model application is convenient to show human body motion based on the received sensor data. Thus users can observe the body motion in a remote way, which is beneficial for medical staff to monitor patients' behaviors and health status. Medical staff can see if patients' body motions are not suitable or not allowed. Additionally, an abnormal behavior can indicate a health emergency. Compared with partial sensor data, whole body motion is better to judge if a wearable sensor user needs medical help. In this case, the body model application could reflect whole body motion when receiving body sensor data, which is significant both for patients and medical staff.

6. Evaluation

We measured the performance of our 3D body model in three aspects: the multi-output regression models, the accuracy of the 3D body model, and the impact of bone lengths. In Section 6.1, we introduce the performance of the regression model including the Mean error and R2 Score. In Section 6.2, we show how we evaluated the accuracy of our 3D body model in different joints and different motions. In Section 6.3, to make the evaluation complete, we analyzed the impact of the bones lengths in our model.

In the paper, we selected two sub-datasets from the whole 3D Microsoft Motion datasets considering both the accuracy and completeness of the raw data. They are data 1 and data 2. To make the evaluation complete, we also tested them together, which is data 1&2 in Table 8. When combining these two datasets, we considered the influence of the trivial difference among different people. Therefore, there are three datasets in this paper including data 1, data 2 and data 1&2.

To train and test data, we divided all the data into two parts: training data and testing data. The percentage of testing data made up 30% of the whole data, which were randomly selected from all 20 motions. All the evaluation results about errors are the average values based on the testing data.

Table 3: Evaluation datasets

Dataset	Training data size	Testing data size
data 1	1223	525
data 2	1766	758
data 1&2	2989	1283

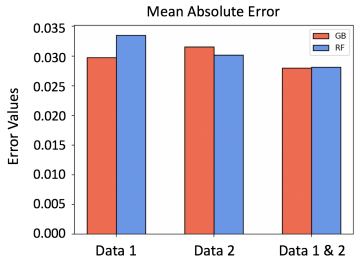

Figure 17: MAE bars

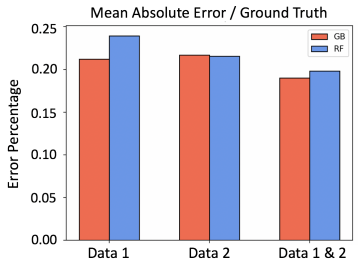

Figure 18: MAE/Ground Truth bars

Figure 19: MSE bars

6.1. Performance of Regression models

6.1.1. Mean Error

Mean absolute error (MAE) and Mean Square Error (MSE) are two important metrics widely used to evaluate a regression model's performance[23]. To give definitions of the two metrics, we assume there are $n_s p$ samples. The predicted depth value is $\hat{dep}_i$ while ground truth values are dep_i, where $1 \leq i \leq n_s p$.

The MAE and MSE are calculated as follows:

$$MAE = \frac{1}{n_s p} \Sigma_{i=1}^{n_s p} |dep_i - \hat{dep}_i| \tag{26}$$

$$MSE = \frac{1}{n_s p} \Sigma_{i=1}^{n_s p} |dep_i - \hat{dep}_i|^2 \tag{27}$$

MAE is a good indicator to evaluate the average error of a model, while MSE is better when errors are in Gaussian distribution. Since we cannot confirm which statistical metric is better, we show both of them to evaluate the performance. In addition, we also calculated MAE/Ground Truth to show the error percentage. Figure 17 shows the mean absolute error of Gradient Boosting and Random Forest regression models. We know the depth is an angle in z axis direction, so the range of depth is from $-\pi$ to π. The predicted mean absolute errors of 18 joint depths are mostly around 0.03. Here we can see that the influence of different people is positive and trivial by comparing among data 1&2 and data 1 and data 2. In addition, we evaluated the mean absolute error percentage based on the ground truth in Figure 18. Most error percentages range from 20% to 25%, while Gradient Boosting and Random Forest achieve every similar performances. Only in Data 1, the error percentage of Random Forest was worse than Gradient Boosting, but still lower than 25%. Although the error percentages in Figure 17 are a little high, the absolute error values of the final joint positions are not obvious, which can be limited into 10 centimeters in Section 6.2. Figure 19 shows the MSE results. Overall, the performance is satisfactory, and the errors are controlled from 0.0025 to 0.003.

6.1.2. R2 Score

R2 is also an important metric to evaluate the performance of non linear regression models. The definition of R2 is the square of the correlation coefficient between the ground truth and predicted values[24]. We have the similar assumptions as section 6.1.1, and $\bar{dep}_i$ is the average of the ground truth dep_i. R2 score is calculated as follows:

$$R^2 = 1 - \frac{\Sigma_{i=1}^{n} (dep_i - \hat{dep}_i)^2}{\Sigma_{i=1}^{n} (dep_i - \bar{dep}_i)^2} \tag{28}$$

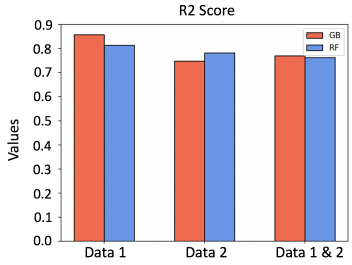

Figure 20: R2 bars

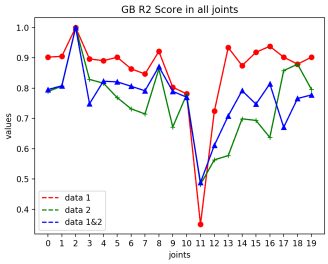

Figure 21: R2 Scores in Gradient boosting

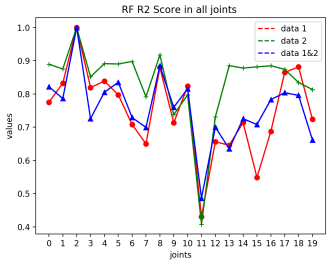

Figure 22: R2 Scores in Random forest

Table 4: Joint Descriptions

Joint Number	Descriptions	Joint Number	Descriptions
Joint 1	Right shoulder	Joint 2	Left shoulder
Joint 3	The bottom of the neck	Joint 4	Middle of the back
Joint 5	Right hip	Joint 6	Left hip
Joint 7	Middle of the waist	Joint 8	Right elbow
Joint 9	Left elbow	Joint 10	Right wrist
Joint 11	Left wrist	Joint 12	Right hand
Joint 13	Left hand	Joint 14	Right knee
Joint 15	Left knee	Joint 16	Right ankle
Joint 17	Left ankle	Joint 18	Right foot
Joint 19	Left foot	Joint 20	Head

R2 Score is calculated for the prediction values[24], thus we utilize the prediction values in testing data to measure the performance. In addition, it is an important but approximate metric to evaluate the accuracy. That is, R2 score is only an approximation but not a very precise description. When R2 score is closer to 1, the accuracy will be higher. If R2 score is 1, no error exists. Figure 20 shows the total R2 scores, which reflect the accuracy. R2 Score of Gradient Boosting and Random Forest are similar, ranging from 0.8 to 0.9 in three datasets. Moreover, Figure 21 and Figure 22 show the R2 score of every joint. We find that most R2 scores of joints achieve 0.7, indicating satisfied accuracy. According to the joints' descriptions in Table 4, joint 2 is the body center from set C. This center joint, as a given input, is absolutely correct. As for joint 11, the prediction accuracy is obviously lower than others. That is, the left wrist has great bias in body motion. Overall, the trends are similar in Gradient Boosting and Random Forest, and the accuracy achieves around 80% in three datasets.

6.2. Performance of 3D body model

We demonstrate the accuracy of our 3D body model from two aspects: one is to compare different joints based on all types of motions; the other is to compare different motions in all joints. Since 54 static positions form one motion, we utilized the average error of 54 frames to reflect the accuracy of motions.

Firstly, we show the error percentage and absolute error values in different joints. We implemented the 3D body model and then calculated the x, y, and z positions in data 1&2. We utilized error percentage and absolute error values of x, y, and z axis values in Table 4. We selected the 12 most important joints in Table 5. When calculating x and y positions, the errors are mainly propagated from joint depths to x positions. Take the left wrist as an example, depth has bad accuracy, as explained in section 6.1.2, while its error percentage of x is up to 480% and the absolute error is 0.55898 meters. We found that the accuracy of x positions obviously depended on the depths, but y positions were not much affected by joint depths. Another interesting finding is about the right ankle. Although the error percentage of z position is around 42%, the absolute error is less than 1 cm. In the real world coordinate system, the actual bias could be tolerated easily. Most of the absolute errors are limited within a small range, supporting the validation of our 3D body model.

Table 5: Errors in different joints

Joint	X error percent	X error(m)	Y error percent	Y error(m)	Z error percent	Z error(m)
Left shoulder	0.01773	0.04647	0.00881	0.00584	0.044821	0.01365
Left elbow	0.67626	0.27342	0.01128	0.01494	0.02389	0.01542
Left wrist	4.849508	0.55898	0.04282	0.02648	14.50363	0.02698
Left knee	0.02250	0.00732	0.03018	0.00196	0.04248	0.00320
Left ankle	0.10324	0.03907	0.07939	0.03312	0.15217	0.01299
Left foot	0.14716	0.06231	0.04477	0.00690	0.074535	0.02415
Right shoulder	0.01479	0.03237	0.00297	0.04820	0.03727	0.01535
Right elbow	0.35263	0.14754	0.02027	0.03888	0.01193	0.00634
Right wrist	0.84856	0.19203	0.02360	0.04429	6.02130	0.02169
Right knee	0.07854	0.00849	0.01068	0.00351	0.08040	0.00193
Right ankle	0.15628	0.00473	0.02422	0.00440	0.42031	0.00938
Right foot	0.09392	0.00105	0.01340	0.00108	0.09350	0.00213

Table 6: Errors in different motions

Motion type	Motion description	Error percent	Error absolute values(m)	Standard deviation
Motion 1	Horizontal arm wave	1.86433	0.10913	0.36942
Motion 2	Golf swing	0.24497	0.04098	0.17933
Motion 3	Hand clap	1.06313	0.084937	0.30408
Motion 4	Tennis swing	0.20052	0.02868	0.185259
Motion 5	Pickup & throw	0.319729	0.06680	0.27458
Total	Average of all motions	0.42031	0.04879	0.234557

Next, we compared different motions when calculating the average accuracy of all joints in Table 5. Since there are twenty different motions in the datasets, we selected five of them, then calculated the sum of all motions. To make it clear, we described the motion content and computed the error percentages and absolute errors. The errors are the average values of x, y, z for all joints. To show the deviation and stability of the performance, we also showed the average of the standard deviations in Table 6. We found that the upper body motions had worse performance than the whole body motions, which came from the bias of wrists. In sum, the absolute error values can be limited into 10 cm of different motions.

6.3. Impact of bone lengths

Bone lengths are important for simulating body motions. In our 3D body model, we can get precise bone lengths of one specific human body according to the Microsoft 3D Motion Datasets. Thus, above experiments are not affected by bone lengths. Our body model consisted of two sub-models: the multi-output regression model and the skeleton recovery model. The multi-output regression model made use of joint angles to predict joint depths, which did not utilize bone lengths to compute. For the skeleton recovery model, the solution is a mathematical method with no errors. That is to say, if the inputs of the skeleton recovery model are absolutely correct, the outputs must be correct. Errors only came from the multi-output regression model. Therefore in our body model, bone lengths have no impact on the accuracy. If the body model is applied in real wearable sensors, the accuracy of the body model can be guaranteed using precise bone lengths. To make the evaluation complete, we tested the impact of bone length errors. We calculated the average bone lengths of ten users in Microsoft 3D Motion Datasets, then utilized the average bone

Table 7: Errors from average bone lengths

Dataset	Error Percent in bone lengths	Error Percent in the body model
Data 1	0.16312	0.46992
Data 2	0.30093	0.53268

lengths in the skeleton recovery model. In Table 7, we show errors from bone lengths by using the average bone lengths in two datasets. We can see that the errors depend on the bias between average bone lengths and the bone lengths of a specific user.

7. Discussion

Our work is to build a 3D skeleton-based human body model, including a multi-output regression model and a skeleton recovery model. One interesting finding is that the performance of the combined dataset (data 1&2) is a little better than the separate datasets (data 1 or data 2). When enlarging the data size, the statistical evaluation metrics will produce higher accuracy[24]. In addition, the difference of people body information has trivial influence on the model accuracy, reflecting the generalization of our 3D body model. Another finding is about the bad prediction of the left hand. The motion flexibility of hands can be the reasons to explain this situation. The hand motions are often not connected tightly with the body motions. Moreover, compared with other joints, the accuracy of the left elbow is relatively low, since the errors are accumulated from all the previous joints.

For regression methods, there are some other potential machine learning models to solve the multi-output problems. Neural network is a possible good choice, since it is a very popular machine learning model currently. The challenges here are about regression, for neural networks are mainly used to solve the classification problems. In a classic neural network structure, the final layer will transform the continuous values, calculated from the previous layers, to a discrete value. Thus the possible solution is to remove the final layer and directly produce outputs using the second last layer. However, the difficulty of this solution is to train parameters and get high accuracy, which needs heavy work and great experience. In fact, neural networks are not traditional regression models, and that is why we did not select this in our model.

Our simulated and evaluation datasets only come from the Microsoft 3D Motion, which is one of the most popular datasets. However, there still exist many other datasets, and we did not include them into our 3D body model. The reason is that they often have different amount of joints. For instance, CMU mocap datasets[25] has 31 joints, which is not suitable for our body skeleton abstraction. Since simulating sensor data is only in the preprocessing section, we did not explore much about it. Thus to make our model more general, one direction for future work is to combine several different datasets and produce one human body.

8. Related Work

There are many related works about skeleton-based body motion recognitions. Mnier et al.[11] made use of a 3D skeleton model to recover the human body pose. The skeleton model used a statistic body model and mainly described the relationships between joints. Li et al.'s work[7] built an action graph based on a 3D bag of points. But this paper only came up with a theory body model and then extracted data from images. In Carranza et al.'s paper[9], they built a 3D mesh body motion model to represent body information extracted from videos. Li et al.[26] combined the sensor data and Kinect images to estimate joints angles. These two studies only focused on joint angles but not joint positions. In Qi et al.'s work[5], they built a ellipsoids based body model, which focused on the the lower extremity joint movements by using wearable sensors. Senior et al.[10] built a cylinder-based body motion model from video data. Dekker et al.'s work[27] introduced a whole body recovery model based on the image data and its main goal was to generate useful symbolic body information. In health care, body motion recognition was used for monitoring or therapy[4]. Since the Kinect camera is convenient to catch human body information, the image data become a popular base for the body motion recognitions. In Bian et al.'s work[28], they proposed a detection system to detect falls, which only recognizes one body motion. Alexiadis et al.'s[29] work aimed to reconstruct the real-time 3D body motions. Protopapadakis's work[30] focused on dance recognition by using depth information. Li et al.'s work[26] also used Random forest regression to recognize upper body motions. All these four works were based on the Kinect camera. Overall, most existing body motion models and recognitions focused on how to process the image data from Kinect datasets, instead of wearable sensor data.

9. Conclusion

In the paper, we built a skeleton-based 3D body model. The input data were joint angles by simulating angle sensor data, while the outputs were joint positions of the entire body. To implement the 3D body model, we divided the whole work into two parts: the multi-outputs regression models and the skeleton recovery model. Moreover, we developed an Android application prototype to show practical scenarios. To evaluate the model completely, we measured the performance of the regression models using MAE, MSE and R2 score in three datasets. In addition, we evaluated the accuracy of our 3D body model for different joints and motions. The impact of bone lengths were also tested. We achieved eighty percent accuracy and most joint position errors are less than 10 centimeters. The evaluation results demonstrate that our 3D body model has satisfied accuracy, stability and generalization.

10. Acknowledgments

This project was partially funded by NSF CSR EAGER (CNS-1841129). The authors appreciate that Xiaoying Zhai and Shuxin Zou provided great help for related work.

References

[1] W. Tao, T. Liu, R. Zheng, H. Feng, Gait analysis using wearable sensors, Sensors 12 (2) (2012) 2255–2283.
[2] T. Liu, Y. Inoue, K. Shibata, A wearable ground reaction force sensor system and its application to the measurement of extrinsic gait variability, Sensors 10 (11) (2010) 10240–10255.
[3] J. Farringdon, A. J. Moore, N. Tilbury, J. Church, P. D. Biemond, Wearable sensor badge and sensor jacket for context awareness, in: Digest of Papers. Third International Symposium on Wearable Computers, IEEE, 1999, pp. 107–113.
[4] R. Lun, W. Zhao, A survey of applications and human motion recognition with microsoft kinect, International Journal of Pattern Recognition and Artificial Intelligence 29 (05) (2015) 1555008.
[5] Y. Qi, C. Soh, E. Gunawan, K.-S. Low, R. Thomas, Lower extremity joint angle tracking with wireless ultrasonic sensors during a squat exercise, Sensors 15 (5) (2015) 9610–9627.
[6] N. Friedman, J. B. Rowe, D. J. Reinkensmeyer, M. Bachman, The manumeter: a wearable device for monitoring daily use of the wrist and fingers, IEEE journal of biomedical and health informatics 18 (6) (2014) 1804–1812.
[7] W. Li, Z. Zhang, Z. Liu, Action recognition based on a bag of 3d points, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops, IEEE, 2010, pp. 9–14.
[8] D. Z. Stupar, J. S. Bajic, L. M. Manojlovic, M. P. Slankamenac, A. V. Joza, M. B. Zivanov, Wearable low-cost system for human joint movements monitoring based on fiber-optic curvature sensor, IEEE Sensors Journal 12 (12) (2012) 3424–3431.
[9] J. Carranza, C. Theobalt, M. A. Magnor, H.-P. Seidel, Free-viewpoint video of human actors, Vol. 22, ACM, 2003.
[10] A. Senior, Real-time articulated human body tracking using silhouette information, in: Proc. of IEEE Workshop on Visual Surveillance/PETS, 2003, pp. 30–37.
[11] C. Menier, E. Boyer, B. Raffin, 3d skeleton-based body pose recovery, in: Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06), IEEE, 2006, pp. 389–396.
[12] W. Li, Z. Zhang, Z. Liu, Expandable data-driven graphical modeling of human actions based on salient postures, IEEE transactions on Circuits and Systems for Video Technology 18 (11) (2008) 1499–1510.
[13] A. Watson, M. Sun, S. Pendyal, G. Zhou, Tracknee: Knee angle measurement using stretchable conductive fabric sensors, CHASE 2019.
[14] C. Strobl, A.-L. Boulesteix, A. Zeileis, T. Hothorn, Bias in random forest variable importance measures: Illustrations, sources and a solution, BMC bioinformatics 8 (1) (2007) 25.
[15] Regression models with multiple target variables.
URL https://towardsdatascience.com/regression-models-with-multiple-target-variables-8baa75aacd
[16] Scikit learn 1.6 nearest neighbors.
URL https://scikit-learn.org/stable/modules/neighbors.html
[17] P. Li, Q. Wu, C. J. Burges, Mcrank: Learning to rank using multiple classification and gradient boosting, in: Advances in neural information processing systems, 2008, pp. 897–904.
[18] R. S. Zemel, T. Pitassi, A gradient-based boosting algorithm for regression problems, in: Advances in neural information processing systems, 2001, pp. 696–702.
[19] J. H. Friedman, Greedy function approximation: a gradient boosting machine, Annals of statistics (2001) 1189–1232.
[20] J. H. Friedman, Stochastic gradient boosting, Computational statistics & data analysis 38 (4) (2002) 367–378.
[21] A. M. Prasad, L. R. Iverson, A. Liaw, Newer classification and regression tree techniques: bagging and random forests for ecological prediction, Ecosystems 9 (2) (2006) 181–199.
[22] A. Liaw, M. Wiener, et al., Classification and regression by randomforest, R news 2 (3) (2002) 18–22.
[23] T. Chai, R. R. Draxler, Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature, Geoscientific model development 7 (3) (2014) 1247–1250.
[24] D. Alexander, A. Tropsha, D. A. Winkler, Beware of r 2: simple, unambiguous assessment of the prediction accuracy of qsar and qspr models, Journal of chemical information and modeling 55 (7) (2015) 1316–1322.

[25] Cmu motion capture database.
URL http://mocap.cs.cmu.edu/
[26] B. Li, B. Bai, C. Han, Upper body motion recognition based on key frame and random forest regression, Multimedia Tools and Applications (2018) 1–16.
[27] L. Dekker, I. Douros, B. Buston, P. Treleaven, Building symbolic information for 3d human body modeling from range data, in: Second International Conference on 3-D Digital Imaging and Modeling (Cat. No. PR00062), IEEE, 1999, pp. 388–397.
[28] Z.-P. Bian, L.-P. Chau, N. Magnenat-Thalmann, Fall detection based on skeleton extraction, in: Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry, ACM, 2012, pp. 91–94.
[29] D. S. Alexiadis, D. Zarpalas, P. Daras, Real-time, full 3-d reconstruction of moving foreground objects from multiple consumer depth cameras, IEEE Transactions on Multimedia 15 (2) (2013) 339–358.
[30] E. Protopapadakis, A. Grammatikopoulou, A. Doulamis, N. Grammalidis, Folk dance pattern recognition over depth images acquired via kinect sensor, 3D ARCH-3D Virtual Reconstruction and Visualization of Complex Architectures.