HMM-based state classification of a user with a walking support system using visual PCA features

Sajjad Taghvaei & Kazuhiro Kosuge

Department of Bioengineering and Robotics, Tohoku University, Sendai, 980-8579 Japan.

Published online: 23 Dec 2013.

To cite this article: Sajjad Taghvaei & Kazuhiro Kosuge (2014) HMM-based state classification of a user with a walking support system using visual PCA features, Advanced Robotics, 28:4, 219-230, DOI: 10.1080/01691864.2013.869482

To link to this article: http://dx.doi.org/10.1080/01691864.2013.869482
FULL PAPER

HMM-based state classification of a user with a walking support system using visual PCA features

Sajjad Taghvaei* and Kazuhiro Kosuge

Department of Bioengineering and Robotics, Tohoku University, Sendai 980-8579, Japan

(Received 1 February 2013; revised 1 May 2013; accepted 30 July 2013)

The improvement of safety and dependability in systems that physically interact with humans requires investigation with respect to the possible states of the user’s motion and an attempt to recognize these states. In this study, we propose a method for real-time visual state classification of a user with a walking support system. The visual features are extracted using principal component analysis and classification is performed by hidden Markov models, both for real-time fall detection (one-class classification) and real-time state recognition (multi-class classification). The algorithms are used in experiments with a passive-type walker robot called “RT Walker” equipped with servo brakes and a depth sensor (Microsoft Kinect). The experiments are performed with 10 subjects, including an experienced physiotherapist who can imitate the walking pattern of the elderly and people with disabilities. The results of the state classification can be used to improve fall-prevention control algorithms for walking support systems. The proposed method can also be used for other vision-based classification applications, which require real-time abnormality detection or state recognition.

Keywords: human state classification; PCA feature extraction; walking support systems; hidden Markov models

1. Introduction

With rapidly aging population worldwide, especially in developed countries, the development of safety and dependability in the systems that physically interact with humans has become more critical. Walking support systems such as walkers and canes that are used by elderly or people with disabilities such as poor eyesight, or a general lack of muscular strength,[1] require improvements in safety and dependability. In this highly vulnerable population, the frequent occurrence of injuries and hospital admissions due to falls associated with walking aids are reported.[2] This suggests that more research is required for realizing the development of safer and more dependable walking support systems.

Different researchers have already designed different types of walker robots by installing various sensory devices and implementing algorithms that control the motion of the robots. They usually try to improve safety and dependability by providing additional features such as navigation and guidance,[3] or adding extra parts to the mechanical frame.[4] Even the researchers who consider the fall prevention of the user mainly focus on the velocity control of the robot regardless of the possible states of user’s motion and different scenarios of falling accidents.[5,6] These approaches can be effective to some extent, but they lack the real-time estimation of the user’s state.

In this article, we categorize the problem to real-time fall detection, which is a one-class classification problem, and real-time state classification, which is a multi-class classification. In real-time fall detection, unlike the previous researches, we propose and test an algorithm that detects the falls before the person falls. In real-time state classification, we modify a previously used method and utilize it for the classification of possible states of a user with walking support system.

In this study, we focus on the possible states of motion for the user of a walker robot by proposing real-time fall detection (one-class classification [7]) to increase safety, and real-time state recognition (multi-class classification), which uses the depth images of the user’s upper body, to increase the dependability of walking support systems.

Vision-based falling detection algorithms are usually based on the geometric properties of human silhouettes in an image [8,9] or body-part detection and machine learning approaches.[10] However, in these cases, the falling detections are usually not fast enough and cannot be implemented in a fall prevention system because the detection occurs after the person has already fallen.[9,11] Therefore, we propose a real-time fall detection algorithm that can be utilized to control the motion of the robot for fall prevention.

On the other hand, visual human action recognition and behavior analysis have been actively researched for many

*Corresponding author. Email: sajjad@irs.mech.tohoku.ac.jp

© 2013 Taylor & Francis and The Robotics Society of Japan
years.[12] However, the studies have not specifically considered the recognition of human’s states of motion while using a walking support system. Here we consider eight states of motion for the user of a walker, and from the acquired experimental data we propose a hidden Markov model (HMM) classifier for real-time state recognition. The results can be used to increase the dependability of the system by developing fall prevention control algorithms for the robot.

Previously the fall detection algorithm had been implemented on RT Walker using laser rangefinders for sensory data and considering a support polygon and the walking characteristic of the user.[13] However, the system does not detect some types of falling such as fall right, fall left, or fall back. We have previously used Bayes classifiers for one-class classification in [14,15]. The methods for feature extraction, classification, and evaluation of the results are improved and modified in this research.

We use a principal component analysis (PCA)-based method for the extraction of the visual features from the depth map. Unlike the upper body centroid considered in [15], the method can be used for other vision-based classification applications that require real-time abnormality detection or state recognition. Moreover, in this approach, a similar classifier is used for different users. The classification for both fall detection and state recognition, referred to by one-class and multi-class classification, respectively, are performed using HMMs. The visual data are acquired by installing a Microsoft Xbox sensor, (also known as Kinect), on a walker robot equipped with servo brakes for controlling the motion. The performance of the algorithms are validated by the experimental data gathered from an experienced physiotherapist, who is very skillful at imitating the motions of the elderly and persons with disabilities, and another nine healthy subjects with various physical characteristics.

2. Framework
As with common classification problems, the framework consists of two main parts: feature extraction and classification. The classifier is actually a probabilistic model built by the accumulated data for different cases and conditions. We are going to use PCA in the feature extraction part and HMM in the classification part. We try to present an algorithm in which the state classification method is designed to be less dependent to user’s specifications such as height, weight, walking pattern, and clothing compared with the previously proposed method in [15].

The feature extraction and data classification methods are used to control the motion of a walker robot called ‘RT Walker.’ We used servo brakes to control the motion of a passive-type walker inspired by the proposed concept of passive robotics by Goswami et al. [16]. Because there is no chance of unintentional movements of the system, passive robots are intrinsically safe and suitable for systems that physically interact with humans.[1] The developed RT Walker is a prototype with a support frame, two passive casters, and a controller (Figure 1). The rear wheels are equipped with powder brakes, which can be used to change the brake torques of the rear wheels according to the input current.

The robot is equipped with the Kinect to obtain the user’s depth images (Figure 1). The sensor data are robust and reliable under various optical conditions. Moreover, it is considerably cheaper than the previously used laser rangefinders.[13]

Since the walker is passive, motion control involves the activation of brakes on the rear wheels for any non-walking state. On the basis of common accidents that involve walker users, these states can be classified into the types shown in Figure 2, including four falling scenarios, sitting, and standing. The figures are taken from an experienced physiotherapist who imitated typical accidents that befall real users of walkers.

The physiotherapist, Kawazoe [17], was asked to imitate the motion of an elderly person based on his long-term experience. He has tried to walk with roughly the same speed, step size, and walking pattern of an elderly person. Falling can occur in many ways, but it can be roughly classified as one of these typical cases. These falling types are considered on the basis of direction of the fall and common accidents that involve walker users according to the experienced physiotherapist. Although the falling posture varies for different body conditions, the direction of the fall can be classified roughly as either right, left, back, front, or down. However, we should mention that the eight states of motion might not cover all possible states of motion, especially when we are dealing with people with motion disabilities. For further reference, we use the notation C to
Figure 2. Human states while using a walker, as demonstrated by an experienced physiotherapist. Fall Side is categorized as fall right and fall left.

Table 1. Labels of the states.

<table>
<thead>
<tr>
<th>Label C</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Walk</td>
<td>Sit</td>
<td>Fall Down</td>
<td>Fall Right</td>
<td>Fall Left</td>
<td>Fall Forward</td>
<td>Fall Back</td>
<td>Stand</td>
</tr>
</tbody>
</table>

label the eight states of motion which we consider for the walker user. The label is shown in Table 1. We consider that these states also enable us to validate the accuracy of the state classification algorithm by indicating whether the algorithm detects all of them as non-walking states. It is believed that by stopping the walker in non-walking states, the occurrence of falling accidents can be significantly decreased, and the walker can be used as a support for standing and sitting.

We also propose algorithms for multi-class classification where we classify the state of motion to one of the eight states we consider in this research. The results are not used in the current motion control system, but they can be used for improving the dependability of the system by proposing a more flexible control algorithm. This is considered as a basis for the future work of this research.

3. PCA-based feature extraction

PCA is a method for extracting the important information from the data. It represents the original data as a set of new orthogonal variables called principal components. It has also been used in computer vision for the representation of shape, appearance, and motion, and one-class classification problems. The method has been specifically used for the classification of human actions on the basis of depth image data, and is shown to be a good choice for feature extraction. Here, we explain the user segmentation and propose PCA-based feature extraction method for one-class and multi-class classification. The proposed PCA-based method is a modification of the method in [21].

3.1. User segmentation

To extract the feature, the user first has to be segmented from the background. The user segmentation is performed using the distance slicing method. While the foreground may still contain some pixels from the environment, such as parts of the robot, they can be removed using simple image-processing filters. The segmentation results show that the system works well in different light conditions in indoor environments. This would be a difficult task had an RGB camera been utilized as the visual sensor. The segmented user’s silhouette is shown in Figure 3 for several different background light conditions and two subjects.

To present a quantitative measurement, we calculated the average pixel values of RGB images captured by Kinect’s RGB camera during the experiments with several light conditions. The values range from 2.2 to 131.5, where the images are saved with pixel values as integers between 0 and 255. For the whole range, the user segmentation was done with satisfactory results as shown in Figure 3.
Figure 3. The extracted color images (a)–(c) and the respective segmented user’s silhouettes (d)–(f) for different indoors light conditions.

Table 2. Characteristics of the subjects for the experiments.

<table>
<thead>
<tr>
<th>Subject</th>
<th>Gender</th>
<th>Height (cm)</th>
<th>Weight (kg)</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>M</td>
<td>188</td>
<td>74</td>
<td>28</td>
</tr>
<tr>
<td>B</td>
<td>F</td>
<td>149</td>
<td>64</td>
<td>34</td>
</tr>
<tr>
<td>C</td>
<td>M</td>
<td>164</td>
<td>50</td>
<td>27</td>
</tr>
<tr>
<td>D</td>
<td>M</td>
<td>178</td>
<td>73</td>
<td>29</td>
</tr>
<tr>
<td>E</td>
<td>M</td>
<td>183</td>
<td>72</td>
<td>28</td>
</tr>
<tr>
<td>F</td>
<td>M</td>
<td>173</td>
<td>85</td>
<td>49</td>
</tr>
<tr>
<td>G</td>
<td>F</td>
<td>169</td>
<td>57</td>
<td>28</td>
</tr>
<tr>
<td>H</td>
<td>F</td>
<td>160</td>
<td>52</td>
<td>30</td>
</tr>
<tr>
<td>I</td>
<td>M</td>
<td>182</td>
<td>51</td>
<td>28</td>
</tr>
<tr>
<td>J</td>
<td>F</td>
<td>155</td>
<td>48</td>
<td>26</td>
</tr>
</tbody>
</table>

3.2. Principal component analysis

The segmented depth image data for all states are gathered from all subjects and stored in the format of vectors with length \( n \), which is the number of pixels of the depth image (in our case, \( n = 480 \times 640 = 307200 \)). By applying the PCA method to any set of depth images, the average image (average vector) and eigenimages (eigenvectors) of the set are calculated. These eigenimages are sorted on the basis of their corresponding eigenvalues. The projection of a depth image, subtracted by the average image, on each eigenimage, would be the extracted PCA features associated with the data-set. We use different sets of eigenimages for one-class and multi-class classification.

3.2.1. Features for one-class classification

For one-class classification, the depth images from a group of users in all eight states of motion are gathered. The vectors of length \( n \) corresponding to depth images from all states of motion for each user are gathered in a matrix \( X_i \), where \( i \) is the label for the \( i \)th user among a total of \( J \) subjects. If we gather depth images for \( S \) different states of motion (\( F \) frames each) from \( J \) subjects in vectors denoted by \( x_{s,j}^{i} \) [21], we obtain

\[
X_j = \left[ x_1^{1,j}, x_2^{1,j}, \ldots, x_F^{1,j}, \ldots, x_1^{S,j}, x_2^{S,j}, \ldots, x_F^{S,j} \right]
\]

Then, the data from all the users are gathered in another matrix \( X \) defined as

\[
X = [X_1 | X_2 | \ldots | X_J]
\]
To find the principal components of the data, we first calculate the covariance matrix $\Sigma$ by

$$\Sigma = (X - \bar{x}1^T_F) (X - \bar{x}1^T_F)^T$$

(3)

where $\bar{x}$ is the rowwise mean vector of $X$ and $1$ is a vector with all entries equal to one. This would be like subtracting all the depth images from an average image ($\bar{x}$) before calculating the covariance matrix.

The covariance matrix $\Sigma$ can be decomposed as

$$\Sigma = U^T \Lambda U$$

(4)

where the rows of $U$ are unit eigenvectors and $\Lambda$ is a diagonal matrix of eigenvalues of $\Sigma$. The principal components of the data will be calculated by finding the projection of data vectors along these eigenvectors. However, the importance levels of the eigenvectors are not the same, and are directly related to the corresponding eigenvalues. Then, the rows of $U$ are sorted based on the corresponding eigenvalues in decreasing order. The first $0 < m \leq n$ eigenvectors construct $V_{m \times n}$. The principal components of a sample (depth image) data $x$ can be calculated by

$$y_{m \times 1} = V(x - \bar{x})$$

(5)

We see that using Equation (5), the number of variables corresponding to a depth image frame, $n = 480 \times 640 = 307200$, can be decreased to a chosen value $m$, in vector $y_{m \times 1}$.

For our problem, we gathered data from 10 subjects, whose characteristics are described in Table 2. The data from $S = 7$ states of motion, namely walk, sit, and five types of falling were gathered. The data from standing state is not mentioned here because the depth images for standing are already considered in the other non-walking states according to our definition for standing. For each state of motion, we require data that include various configurations of upper body during that state. We also need to consider the same number of frames $F$ for all states. According to the experimental results (Section 6), the state with the shortest time duration provides us with approximately 30 frames in average (Consider the frame rate of data gathering 30 fps, and the shortest time duration 0.98 s as shown in Table 3). By excluding the frames in the margin of state transitions for the sake of certainty, the remaining $F = 10$ frames would be sufficient for our purpose. The average image and first seven eigenimages of this set are shown in Figure 4. To choose the number of components that is sufficient for further analysis ($m$), we consider the weight of the eigenvalues,[15] and the adequate number of components for training an HMM for classification.

### 3.2.2 Features for multi-class classification

For one-class classification, we proposed the eigenimages of the data gathered from 10 subjects in different states altogether. The variations in the values of the principal components are expected to be sufficiently significant to detect whether the person is walking or not. However, these features did not show satisfactory results when used for multi-class classification. That is probably because we have used the depth images of all states of motion together for the PCA.

For multi-class classification, we propose that the eigenimages of different states of motion can be calculated separately, and the top principal component corresponding to each state of motion can be considered for classification.

Using the same notation as before, the depth images can be represented by vector $x_{s,j}$. Note that we obtained the depth images for $S$ different states of motion ($F$ frames each) from $J$ subjects in vectors. The vectors from all users during the $th$ state of motion are gathered in $X_s$. We have

$$X_s = [x_{s,1}^1, x_{s,1}^1, \ldots, x_{F}^s, \ldots, x_{s,J}^1, x_{s,J}^2, \ldots, x_{F}^s]$$

(6)

For our problem, the data are gathered from $J = 10$ subjects in $S = 7$ states. The subjects are asked to use RT Walker and perform all states of motion to provide sufficient
Table 3. Time duration of different fall states and detection duration.

<table>
<thead>
<tr>
<th>Fall type</th>
<th>Forward</th>
<th>Right</th>
<th>Down</th>
<th>Back</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fall duration (s)</td>
<td>1.40</td>
<td>1.52</td>
<td>0.99</td>
<td>0.98</td>
<td>1.22</td>
</tr>
<tr>
<td>Detection duration (s)</td>
<td>1.21</td>
<td>1.02</td>
<td>0.72</td>
<td>0.52</td>
<td>0.84</td>
</tr>
</tbody>
</table>

Figure 5. The eigenimages for multi-class classification associated with states of motion.

data for PCA. The experimental details are explained in Section 6. The average image, \( \bar{x} \) in Equation (5), and the top eigenvectors \( \{u_j, j = 1, 2, \ldots, 7\} \) for all seven states were calculated using the accumulated data from all subjects. The seven eigenvectors(eigenimages) are shown in Figure 5. The average image is similar to the one shown in Figure 4(a). Similar to one-class features, the average image and eigenimages would be represented by \( \bar{x} \) and \( V \) in Equation (5). Note that, unlike the feature for one-class classification, these eigenimages are not orthogonal to each other.

The eigenimages \( \{u_j, j = 1, 2, \ldots, 7\} \) refer to the top eigenvectors of walking, sitting, fall down, fall right, fall left, fall forward, and fall back, respectively. Although for the state classification we consider standing as another state of motion, it is not considered in the eigenimages. The state of standing is considered as either standing up from a sitting state or regaining stability after a fall incident. Therefore, the eigenimages for this state are already considered in other non-walking states. The only difference between standing and other non-walking states is the sequence of these eigenimages in the motion, which will be considered in the classification by HMMs.

3.3. Normalizing the features

For the case of one-class classification, the values obtained from Equation (5) are not used directly for the classification. Although the eigenimages contain the data from different users, we have already normalized the pixel values as described in Section 3.1; the values for \( y \) have significant differences among users. The values are normalized on the basis of each individual’s walking data. Therefore, despite these significant differences, the users will have similar feature values for walking. It is expected that the other states of motion for different users indicate similar patterns so that the proposed classifier(s) trained with the sample group of subjects obtain acceptable results for any user.

The normalized features are determined by

\[
\hat{y} = \frac{y - \mu_{\text{walk}}}{\sigma_{\text{walk}}}
\]

in which \( \mu_{\text{walk}} \) and \( \sigma_{\text{walk}} \) are the mean vector and standard deviation, respectively, of the \( y \) values calculated for walking. Therefore, for all subjects, the normalized values for walking will have zero mean and unit standard deviation. In addition, for the other states, we will use the normalized values for state classification.

4. HMM-based state classification

HMMs have been widely used by researchers in vision-based human action recognition.[20,22,23] With this approach, a time sequence of features (observations) is used to classify human actions. It should be noted that falling accidents cannot be accurately referred to as actions because they indicate a user’s unintentional behavior when falling. The method is also shown to be useful in human abnormal state detection and recognition.[15] However, the proposed classifier in [15] needs to be updated for a new user and it cannot be used for multi-class classification.

As discussed in Section 2, we use HMMs for both one-class and multi-class classification. After explaining the mathematical structure, the classification algorithm is described for both one-class and multi-class classification.
4.1. Mathematical model

The mathematical model used here is based on the one described in [24]. The observed data at frame $t$, $o(t) \in \mathbb{R}^T$ are the normalized features calculated by Equation (7):

$$o(t) = \tilde{y}(t)$$  \hspace{1cm} (8)

The time series of the observed data, $O = \{o(t)|1 \leq t \leq T\}$, comprises the data stored from $T$ successive frames. The model is composed of hidden states $S = \{s_1, s_2, \ldots, s_K\}$, which go through a transition at every time step with a Markov model. Note that these hidden states are mathematical terms and do not refer to states of motion. The observed data-set, $O$, is related to the hidden states through a set of parameters, which can be expressed as $\lambda = (\Pi, A, B)$. Parameter $\Pi$ is the probability distribution for the initial state, $A$ is the probability distribution for state transitions, and $B$ is the probability distribution for the observed data. Because the observed data are assumed to have a mixed Gaussian distribution, parameter $B = \{b_i(t)|1 \leq i \leq K, 1 \leq t \leq T\}$ can be formulated by

$$b_i(t) = \sum_{j=1}^{M} c_{ij} N(o(t), \mu_{ij}, \sigma_{ij})$$  \hspace{1cm} (9)

where $\mu_{ij} \in \mathbb{R}^D$ and $\sigma_{ij} \in \mathbb{R}^{D \times D}$ are the Gaussian mean vector and variance matrix for the $j$th mixture component in state $s_i$, respectively, and $D$ indicates the dimensions of the observed data.

The training of an HMM involves determining the parameter set values $\lambda = (\Pi, A, B)$, which are required to maximize the probability function of the observation sequence $P(O|\lambda)$. The initial values set for HMM parameters are important. In this paper, we have considered a left-to-right HMM for modelling the states of motion. [24] The number of hidden states, $K$, is set to be equal to the length of the time series $T$. The value for $T$ is chosen according to experimental results for the duration of a fall explained in Section 6.1. Although the classification is performed at every frame we need to obtain the whole sequence of a fall in the observation vector within the first quarter of a fall accident. Therefore, we choose $T$ by considering the minimum time duration of a fall accident (Table 3) and the achievable frame rate of our algorithm.

The parameters for a mixture of Gaussians are calculated using the EM algorithm. Elements of the transition matrix $A$ and initial probability distribution $\Pi$ are chosen to satisfy the conditions of Equation (10) and left-to-right HMMS. [24]

$$\sum_{j=1}^{K} a_{ij} = 1 \quad \sum_{i=1}^{K} \pi_i = 1$$  \hspace{1cm} (10)

The first condition has to be satisfied for all $1 \leq i \leq K$.

4.2. One-class classification

For one-class classification, we need to acquire the normal walking data. However, for the multi-class classification, the data for all states are needed. The models, which are specified by parameters $\lambda_i, 1 \leq i \leq 8$, respectively, refer to each of the eight states labeled in Table 1. The model parameters are updated using the gathered data and Baum-Welch algorithm. [24] The observation sequence is updated at each frame by replacing the last one with the current frame data and shifting all others one step back. Given the model parameters, $P(O|\lambda_i)$, the probability of the observation sequence is calculated at each frame by using the forward algorithm. [24] In this way, the state classification can be performed with a high frame rate, which is required for our application.

Considering $\lambda_1$ to be the model parameters trained with normal walking data for one-class classification, we simply have

$$\begin{cases} P(O(t)|\lambda_1) > P^c & C(t) = 1 \text{(walk)} \\ \text{otherwise} & C(t) = 2, 3, \ldots, 7 \text{(fall, sit, stand)} \end{cases}$$  \hspace{1cm} (11)

where $P^c$ is the critical threshold that separates normal walking from other states.

The way we determine the $P^c$ value is very important in the performance of one-class and multi-class classification. It should be low enough so that it does not hinder the walker during normal walking, and sufficiently high so that it does not miss any falling incident.

The new user is asked to walk normally with the walker to record $i = 1, \ldots, F$ successive frames and the probability distribution value for this data are shown by $P^\text{walk}_i$. The proposed method for calculating this value is

$$\log P^c = (1 + \beta) \min_{i=1}^{F} \log P^\text{walk}_i - \frac{\beta}{F} \sum_{i=1}^{F} \log P^\text{walk}_i$$  \hspace{1cm} (12)

The threshold value, $P^c$, is calculated using a safe distance from the minimum probability distribution value of normal walking. The distance is adjusted with the coefficient $\beta$, which is set on the basis of the experimental results and will be discussed in Section 6.

The threshold value is adjusted based on the walking pattern of the new user. The values of $P(O(t)|\lambda_1)$ might be different for a new user, especially for the patients who have different walking postures than healthy subjects. But, by adjusting the threshold $P^c$, for the new user, the walker will stop when the features deviate too much from the features for the normal state (walking) of the same user.

4.3. Multi-class classification

Eight models are trained for multi-class classification, and when the observation sequence is detected to be non-walking according to Equation (11) other model probabilities are also calculated to determine the state of motion.
The multi-class classification is performed on top of the one-class classifier by
\[
\begin{align*}
P(O|\hat{\lambda}_1) > \hat{P}^c & \quad C = 1 \text{(walk)} \\
\text{otherwise} & \quad C = \arg \max_{j=2}^8 P^i
\end{align*}
\] (13)
The parameters \(\hat{\lambda}_1\) and \(\hat{P}^c\) refer to the trained HMM and threshold for one-class classification of the user’s state. The reference probability, \(P^i\), is calculated by
\[
P^i = P(O|\lambda_i) \sum_{j=1}^{L} P(q(T) = s_{K-(L-j)}|O, \lambda_i) \tag{14}
\]

The first term shows the probability that the model \(\lambda_i\) generates the observed data. The value \(P(q(T) = s_{N-(L-k)}|O(t), \hat{\lambda}_i)\) shows the probability that the last hidden state, \(q(T)\), be any of the last \(L\) states in the set \(S = \{s_1, s_2, \ldots, s_K\}\). Since the HMMs are left-to-right models with same number of hidden states \(K\) and \(T\), it would be more probable for the correct model to take the transitions of hidden states to the last ones. We consider the second term to have a more strict evaluation of the observed data.[25]

These two terms are calculated by forward algorithm.[24]

5. Walker’s motion control

The control system is designed such that the brake force generates a maximum torque when the user’s state is shown to be non-walking, on the basis of Equation (11), for more than a certain time. The brake force \(f_b\) to control the passive walker is designed in a way similar to [13] as
\[
f_b = \frac{e^\alpha - 1}{\alpha} f_{\text{max}} \tag{15}
\]
where \(\alpha\) is a variable that decreases or increases by \(\Delta\alpha\) if the user is in walking or non-walking states, respectively. \(f_{\text{max}}\) is the maximum brake force generated when \(\alpha\) is at its maximum value, \(\alpha_{\text{max}}\). We use the term \(\alpha\) to be able to change the braking force between 0 and \(f_{\text{max}}\) with an exponential manner. It can be made by an increase or decrease in \(\alpha\) whether the state of motion is non-walking or not respectively. We use the notation \(C\) used in Equation (11) to refer to the states, and \(\alpha\) changes as follows:
\[
\alpha = \begin{cases} 
\alpha - \Delta\alpha & (C = 1; \alpha > 0) \\
\alpha + \Delta\alpha & (C = 2, 3, \ldots, 8; \alpha < \alpha_{\text{max}}) \\
\alpha_{\text{max}} & (\alpha \leq 0) \\
\alpha_{\text{max}} & (\alpha \geq \alpha_{\text{max}})
\end{cases} \tag{16}
\]
On the basis of Equation (16), the brake force \(f_b\) increases or decreases between 0.0 \([\text{N}]\) and \(f_{\text{max}}\)\([\text{N}]\) in an exponential manner.

6. Experimental results

Due to strict regulations and limitations on performing such experiments with patients or elderly people, we decided to evaluate our algorithm with healthy subjects for the current stage of the research. Since we consider the system adaptation for a new user, it is expected that the system be useful for such users as well. The above-mentioned algorithms are tested with 10 subjects using RT Walker. One of the subjects is an experienced physiotherapist, Kawazoe [17], who can skillfully imitate different walking abnormalities and falling accidents. The subjects’ characteristics are listed in Table 2, where the physiotherapist is labeled as user F. The gender is indicated by \(M\) and \(F\) for male and female subjects, respectively. The height of the subjects are within the range 149–188 cm, their weights vary from 48 to 85 kg, and their age range between 26 and 49 years. All the experiments were performed in indoor environments under different light conditions.

The subjects are first asked to walk using the walker for approximately 1 min (the time required to collect 1000 frames and calculate the mean and standard deviation). The acquired walking data will be used later for training the walking state HMM, normalizing the features (Equation (7)), and setting the threshold in Equation (12). They are not required to wear any special dress or clothing except for their regular clothing.

Regarding the falling experiments, the users are asked to start from a sitting position, then stand up, walk for a while, and pretend to fall. The users are asked to repeat each falling scenario for four successive times. In case of the physiotherapist, there was no pretending; the falling accidents included actual falling on the ground. The physiotherapist tried to imitate real accidents that happen to the elderly and disabled persons who regularly use walkers.

6.1. One-class classification

In one-class classification, the HMM was trained with the first six one-class features. The observation sequence, \(O\), consists of vectors, \(\hat{y}\), of length six, from \(T = 4\) successive frames. The HMM has \(N = 4\) hidden states. As mentioned in Section 6.1, these parameters are chosen according to the duration of fall accidents and the frame rate of the algorithm. The parameters are fixed so that a sequence of observation can be captured within less than a quarter of a fall accident.

The same trained HMM is used for all the 10 subjects. Therefore, this method does not require the online training of the classifier for a new user, unlike the method proposed in [14,15].

The experimental results for the HMM-based one-class classification of subject E, for two typical fall types, are shown in Figures 6. All the subjects were asked to start from a sitting position, then stand up, walk, and perform falling actions on four successive occasions. Prior to the falling experiments, the user is asked to walk normally to acquire enough data for the normalization of PCA features (Equation (7)) and to set the threshold \(P^c\) in Equation (11). The threshold is set by each user’s walking data according to Equation (12) with coefficient \(\beta = 0.3\) for all subjects.
We previously mentioned that the current fall detection algorithms usually occur after the person has already fallen on the ground [9,11] which is useless for a walker robot. To validate the performance of a real-time fall detection system, we need to realize whether the proposed algorithm can stop the walker before the falling is finished or not. Therefore, it is important to realize the required time for the system to detect the fall, stop the walker, and compare it to an average time length of the same type of fall accident.

In order to determine the start and end point of a fall, we use the human visual perception on the videos from the experiments recorded during the experiments. We used the videos from the experiments with the physiotherapist, who performed different types of fall accidents while walking with the walker without any brake control, for calculating the average duration of each type of fall. For each of these experiments, the videos are reviewed several times with 12.5% speed and segmented manually. The same process is repeated when a subject is performing the fall experiments using the walker with the brake control system. Note that the exact starting time of a fall cannot be determined, but the data can still be used to verify the performance of the system.

The time taken for fall detection and stopping the walker and the average duration of each fall accident based on the experiments with the physiotherapist are shown in Table 3. For all of the cases the results show that Fall Duration is greater than Detection Duration, which means that the fall detection and stopping the walker occur in the middle of a fall movement before the user has already fallen.

6.2. Multi-class classification

For the multi-class state classification, eight HMMs (for the eight states of motion) are trained by gathering data from the subjects. The same HMMs are used for all the subjects. The observation vectors consist of PCA feature values $\hat{y}$ from four successive frames ($T = 4$). Each model has four hidden states, and a Gaussian mixture model of order two is fitted on observation vectors ($N = 4$ and $M = 2$ in Equation (9)). We use 50

The experimental results of the HMM-based multi-class classification of subject D, for two typical fall types, are shown in Figure 7. The state of motion is indicated by the labels 1-8 (Table 1) on the left axis. It starts with state 2, which is sitting, followed by state 8, which is standing and so on.

The performance of the recognition algorithm can be evaluated by calculating the confusion matrix of the results. We created the training data-set with a size of 27568 gathered from the subjects. The data-set is split into training and testing sets. Table 4 shows the confusion matrix based on the test set. The row-column number refers to the state labels that are defined in Table 1. For better reference, the names of the states are abbreviated; e.g. fwd refers to fall forward.

The rows and columns refer to the true and predicted labels, respectively. For example, by looking at the fourth row of the table, we see that 99.2% of all of the frames for fall right, are classified correctly as fall right, and in 0.8% of these frames are classified as fall back.

7. Discussion

For one-class classification, the results from all 10 subjects show that the method detects 97.16% of falling accidents and all the sitting states. There is also a false positive fall detection rate of 5.68%. Apart from the detection rate, the speed of the algorithm has to be verified. The one-class classification and motion control of the robot is performed by an average frame rate of 18.75 fps.

We should mention that even after applying the brakes, there is a possibility that the user may fall; however, there is an increase in safety and it can be used by the user as a support to regain stability. As an example, consider...
Figure 7. Variations in $\log P(O|\lambda_1)$ and PCA-HMM-based state recognition results for subject D.

Table 4. Confusion Matrix for experimental results of multi-class classification.

<table>
<thead>
<tr>
<th></th>
<th>walk</th>
<th>sit</th>
<th>down</th>
<th>right</th>
<th>left</th>
<th>fwd</th>
<th>back</th>
<th>stand</th>
</tr>
</thead>
<tbody>
<tr>
<td>walk</td>
<td>98.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>0.3</td>
<td>0.7</td>
<td>0.4</td>
</tr>
<tr>
<td>sit</td>
<td>0</td>
<td>94.3</td>
<td>1.6</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>3.5</td>
<td>0.4</td>
</tr>
<tr>
<td>down</td>
<td>4.7</td>
<td>0</td>
<td>65.1</td>
<td>0</td>
<td>0</td>
<td>2.0</td>
<td>20.7</td>
<td>7.5</td>
</tr>
<tr>
<td>right</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>99.2</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0</td>
</tr>
<tr>
<td>left</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>98.4</td>
<td>0</td>
<td>1.6</td>
<td>0</td>
</tr>
<tr>
<td>fwd</td>
<td>2.0</td>
<td>2.2</td>
<td>5.6</td>
<td>0</td>
<td>0</td>
<td>68.7</td>
<td>10.1</td>
<td>11.4</td>
</tr>
<tr>
<td>back</td>
<td>1.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.1</td>
<td>0.3</td>
<td>97.2</td>
<td>0</td>
</tr>
<tr>
<td>stand</td>
<td>3.2</td>
<td>0</td>
<td>1.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>95.7</td>
</tr>
</tbody>
</table>

The results in Table 3 show that the fall detection and stopping the walker can occur in the midst of falling before the user has already fallen on the ground, unlike the methods proposed in [9,11]. Due to difficulties in measuring the exact starting and ending moments of falling, we calculated the time durations by careful manual segmentation of the videos in slow motion taken from the experiments with the users.

For multi-class classification, the confusion matrix indicates a total rate of 92.2% correct classifications, with the highest rates being for walk (98.6%), fall right (99.2%), and fall left (98.4%). The lowest rates are for fall down (65.8%) and fall forward (68.7%), which are mostly mistaken with standing or fall back. The silhouettes of the depth images from user’s upper body for fall down, fall forward, fall back, and standing are similar. This can also be noticed in Figures 5(c), (f), and (g), which are the eigenimages of each of these states. Therefore, the PCA features of these states are closer to each other than the other states. The observation sequences of falling down and forward are mostly mistaken with fall back and stand because the observation vectors of the sequence have close values. The results can be improved by using the depth image of the lower body so that the difference between these states become more significant.

Note that in calculating the confusion matrix, the definition of the true state of a frame is not trivial. In particular, it is difficult to decide the true state of the frames that are close to the transition point between two states. However, by assigning the true states and by seeing the raw visual data, and comparing the classification results to the true state, the calculated confusion matrix can give a useful quantitative measurement of the performance of the classification algorithm. The results can be improved by acquiring more training data from different subjects.

The system should be adaptable for various physical characteristics of the user. We have used the data from a various range of users with different heights, weights, and walking patterns in the experiments for this purpose. Moreover we considered an initial stage of data gathering for a new user and the adaptation of the threshold in classification algorithm. However, testing the system with patients or elderly people can provide a more accurate validation. We should also consider the limitations of the eight state classification which might not cover all possible states of motion, especially for the case of patients. The platform for the experiments, RT-Walker, can be also improved in the physical design to be adaptable for different heights so that the user can be in a natural and more stable position while using the walker.
8. Conclusions
We proposed algorithms for vision-based real-time classification of the states of the user. The classification was presented in two levels, namely one-class classification for fall detection and multi-class classification for state recognition. We considered eight possible states of motion, including five scenarios for falling accidents for the user of a walker. The proposed algorithm consisted of a PCA-based feature extraction and HMM-based classification, which could be used for real-time applications with more than 18 frames per second. The experiments were performed by 10 subjects, including an experienced physiotherapist who imitated the walking patterns of the elderly and persons with disabilities.

In the experiments, similar classifiers were used for all 10 subjects. The proposed method does not require the online updating of the classifiers for a new user. The new user is only asked to walk for one minute with the walker to acquire sufficient data for the normalization of the features. The real-time fall detection by one-class classifiers was achieved for all 10 subjects with a high accuracy of 97.16%. Unlike the previously proposed method, we achieved the real-time fall detection before the user fell on the ground. The time duration of a fall accident was calculated on the basis of the data from the physiotherapist, and the results were used to show that the fall detection system can stop the walker before the user falls on the ground. The real-time state recognition by multi-class classifiers was achieved with a rate of 92.2%.

The results of the real-time fall detection can be used to increase the safety of the system by activating the brakes in non-walking states. The results of the real-time state recognition can be used to increase the dependability of the system by controlling the walker according to the recognized states.

For the future work, we are going to use the classification results to propose more flexible control algorithms. We will also try to implement the framework for other vision-based human state classifications, which require real-time abnormality detection or state recognition.

Acknowledgments
The experiments were done with the help of Dr. Ryushiro Kawazoe, who is an experienced physical therapist and CEO of Kumasuma Inc, and Dr Takuro Hatsukari from the Paramount Bed Company.

Notes on contributors
Sajjad Taghvaei received his BSc in mechanical engineering from Shiraz University, Shiraz, Iran in 2006. He received his MSc in mechanical engineering from Amirkabir University of Technology, Tehran, Iran in 2009. He then received his PhD in bioengineering and robotics from Tohoku University, Sendai, Japan in 2012. His research interest includes assistive robotics and walking support systems.

Kazuhiro Kosuge is a Professor in the Department of Bioengineering and Robotics, Graduate School of Engineering, Tohoku University, Japan. He received the BS, MS, and PhD degrees in control engineering from the Tokyo Institute of Technology in 1978, 1980, and 1988 respectively. From 1980 to 1982, he was a Research Staff in the Production Engineering Department, DENSO Co., Ltd.

From 1982 through 1990, he was a Research Associate in the Department of Control Engineering at Tokyo Institute of Technology. From 1989 to 1990, he was a visiting scientist, Department of Mechanical Engineering, Massachusetts Institute of Technology. From 1990 to 1995, He was an Associate Professor at Nagoya University. From 1995, he has been at Tohoku University. For more than 25 years, he has been doing research on various robot control problems. He has over 150 technical publications in the area of robotics and its applications to the real world. He received the JSME Awards for the best papers from the Japan Society of Mechanical Engineers in 2002 and 2005, the RSJ Award for the best papers from the Robotics Society of Japan in 2005, the Original Paper Award, FANUC FA and Robot Foundation in 2004 and 2006, the Best Paper Award of IROS’97, and the Outstanding Scientific Achievement Award from the Robotics and Mechatronics Division of the Japan Society of Mechanical Engineers in 1999. He is an IEEE Fellow, a SICE Fellow, a JSME Fellow, a RSJ Fellow, and served as President of IEEE Robotics and Automation Society (2010–2011).

References


