Signal, Image and Video Processing (2019) 13:1431–1439
https://doi.org/10.1007/s11760-019-01490-9

ORIGINAL PAPER

Comprehensive evaluation of skeleton features-based fall detection from Microsoft Kinect v2

Mona Saleh Alzahrani[1] · Salma Kammoun Jarraya[2,4] · Hanêne Ben-Abdallah[3] · Manar Salamah Ali[2]

Received: 21 October 2018 / Revised: 13 February 2019 / Accepted: 30 April 2019 / Published online: 15 May 2019

Abstract
Most of the computer vision applications for human activity recognition exploit the fact that body features calculated from a 3D skeleton increase robustness across persons and can lead to higher performance. However, their success in activity recognition, including falls, depends on the correspondence between the human activities and the used joint/part features. To provide for this correspondence, we experimentally evaluate in this paper skeleton features-based fall detection by comparing fall detection performance for different combinations of skeleton features used in previous related works. We determine the skeleton features that best distinguish fall from non-fall frames, and the best performing classifier. In this endeavor, we followed the classical five steps of supervised machine learning: (1) we collected a learning data composed of 42 fall and 37 non-fall videos from FallFree; (2) we extracted and (3) preprocessed the skeleton data of the training set; (4) we extracted each possible skeleton feature; finally (5) we evaluated all extracted and selected features using two main experiments; one of them based on neighborhood component analysis (NCA). In this evaluation, we show that fall detection based on skeleton features has very encouraging accuracy that varies depending on the used features. More specifically, we recommend the following features: 12 features that resulted from NCA experiment, original and normalized distance from Kinect, and the seven features of the upper body part. These features ranked 1st, 2nd, 4th, and 8th on 22 feature sets, with accuracies 99.5%, 99.4%, 97.8%, and 94.5%, respectively. In addition, random forest is the best performing classifier.

Keywords Fall detection · Skeleton features · Feature selection · Kinect v2 · Neighborhood component feature selection

1 Introduction

Falls represent a major cause of morbidity and mortality among the elderly. In fact, statistics show that falls are the primary reason for injury-related death for seniors aged 79 or more, and the second leading cause of injury-related, unintentional deaths for all ages. These facts prompted the development of effective fall detection (FD) systems as a critical support means that would significantly reduce medical care costs associated with falls [1]. A FD system is an assistive device whose primary objective is to alert when a fall event has occurred. It is included in the core building blocks of systems under the umbrella of automatic human activity recognition (HAR), an important area in computer vision and pattern recognition research and applications.

Most of the computer vision applications for HAR recognize human activities through skeleton tracking by representing body parts as joints. They exploit the fact that body features calculated from a 3D skeleton increase robustness across persons and can lead to higher performance [2]. However, their success in human activity recognition, including falls, depends on the correspondence between the human activities and the used joint/part features. To provide for this correspondence, in this paper, we focus on skeleton features-based methods to detect the fall. We experimentally

✉ Salma Kammoun Jarraya
smohamad1@kau.edu.sa

Mona Saleh Alzahrani
mszahrani@ju.edu.sa

Hanêne Ben-Abdallah
hbenabdallah@hct.ac.ae

Manar Salamah Ali
mali@kau.edu.sa

1 College of Computer and Information Sciences, Jouf University, Sakaka, Saudi Arabia

2 Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

3 Higher Colleges of Technology, Dubai, UAE

4 MIRACL, Sfax, Tunisia

investigate which skeleton features best distinguish fall from non-fall frames, and which classifier produces the best performance.

More specifically, we investigate in this work the 25 skeleton joints among those offered by the recent version of Kinect v2 (SDK 2.0) [3]. In addition, we preprocess the body joints by two methods (original and normalization) to overcome the skeleton position, shape, and size. From the preprocessed skeletons, we gather all possible skeleton features (22 feature sets calculated and selected from 11 basic feature sets) that can be obtained from the Kinect v2 [4–8] and selected using the neighborhood component analysis (NCA). In addition, to assess experimentally the performance of the gathered features, we used the FallFree fall detection dataset [9] to conduct two main experiments where we used 79 scenarios/videos covering 42 fall scenarios/videos and 37 non-fall scenarios/videos. Furthermore, to identify the best performing classifier, we applied four supervised learning techniques: C4.5, random forest (RF), artificial neural networks (ANNs), and support vector machine (SVM).

Based on our herein presented experiments, we show fall detection based on the skeleton features has very encouraging accuracy that varies depending on the used features. In particular, we suggested the following features: 12 features that resulted from NCA experiment which ranked 1st features with 99.5% accuracy, original distance from Kinect which ranked 2nd features with 99.4% accuracy, normalized distance from Kinect which ranked 4th features with 97.8% accuracy, and the seven features of the upper body part proposed by Alzahrani et al. [8] which ranked 8th features with 94.5% accuracy. In addition, random forest is the best performing classifier.

The remainder of this paper is organized as follows: In Sect. 2, we overview works on Kinect-based fall detection using skeleton streams. In Sect. 3, we present the five stages of the evaluation framework, covering dataset collection, skeleton data extraction, skeleton preprocessing, feature extraction, and performance analysis (experiments evaluation). Finally, Sect. 4 summarizes the presented work and highlights its extensions.

2 Related works

Many vision-based approaches adopted in recent studies [4–7] use skeleton joints from Kinect to detect different fall scenarios. Kawatsu et al. [4] used the positions of all the 20 joints offered by Kinect v1 to calculate the floor plane equation and average velocity to detect falls. Using all joints, they can distinguish between falls and slowly laying down on the floor. Unlike [4], Lee and Lee [5] use only the hip center joint in their system to distinguish between fall and non-fall scenarios. If falls are detected, their system will notify health care services or the victim's caregivers to provide help. They track the hip center joint whose position and velocity are used to detect three fall scenarios: fall in open space from walking, standing, or lying in bed. Their system achieved a 90% accuracy rate. Also focusing on a subset of joints produced by Kinect v1, Le and Morel [6] use the distance and velocity features of both the head and spine to detect the following four fall scenarios (fall back, front, right, and left) and three non-fall scenarios (walk, pick an object, sit down the bed). First, they compute the room floor plane using the Kinect's floor plane equation. Second, they extract the joints coordinates and convert them to floor coordinates. Then, they calculate the distance from the floor and velocity features. After that, they classify the frames using the SVM. In this work, they detect a fall in a duration of time, and they get a 98.4% accuracy rate.

In the meantime, Kwolek and Kepski [7] added other sensors to Kinect which is a wearable smart device containing accelerometer and gyroscope sensors and was worn near the pelvis region. They use Kinect v1 to reduce the number of false alarms and employ it whenever it is only possible. A triaxial accelerometer is used to indicate the potential fall and motion of the monitored person. If the measured acceleration is higher than an assumed threshold value, the system extracts the person, calculates the features, and then executes the SVM-based classifier to authenticate the fall alarm. The system acquires depth images using the OpenNI library. It achieved a 98.33% accuracy rate when using both accelerometer and depth data, and 90% accuracy and 80% specificity when using depth only, which is the worst result compared to other techniques.

Overall, the above works used either a fixed set or all of the skeleton joints produced by Kinect. None of them tried to determine the subset of joints that is most effective in detecting various types of fall scenarios. In addition, only Maldonado et al. [10] tested some features that can be extracted from depth data (not skeleton data). In this paper, we will investigate the 25 skeleton joints among those offered by Kinect v2 (SDK 2.0) [3] as well as their features to identify those that most efficiently can detect various fall scenarios. Toward this end, we elaborated the evaluation framework shown in Fig. 1.

3 Evaluation framework

To identify the subset of Kinect v2 produced joints and their features that can be used efficiently in a FD method for various fall types, we built our evaluation framework in five stages following a typical machine learning process: (1) Find a suitable dataset that contains skeleton streams recorded using Kinect. For this purpose, we choose to use the FallFree fall detection dataset [9] because it covers all

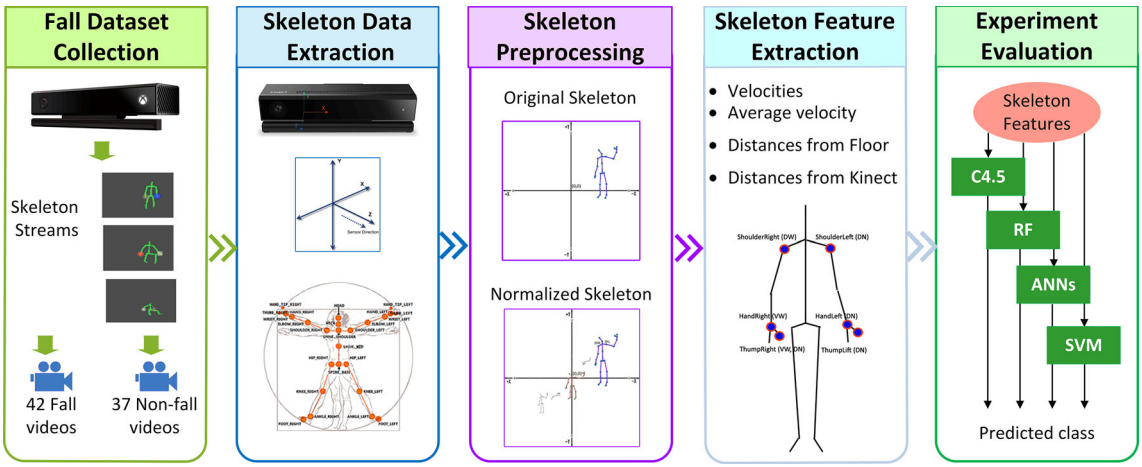

Fig. 1 Evaluation framework for elaborating a skeleton-based fall detection method

fall scenarios along with various non-fall scenarios; (2) For the dataset fixed in stage 1, extract the skeleton information that contains the 3D coordinates of the 25 joints of the human skeleton, the floor plane equation of the room, and the frames' timestamp; (3) Make two copies of the skeleton joints coordinates computed in stage 2; one copy is used as it is (original) and the second is preprocessed by normalization; (4) From each skeleton in the stage 3, compute all feature used in previous works [4–8] to distinguish fall from non-fall; (5) Experimentally evaluate the skeleton features using four supervised learning techniques: C4.5, random forest (RF) decision trees, artificial neural networks (ANNs), and support vector machine (SVM) [11].

3.1 Fall dataset collection

From the FallFree dataset [9], we used 42 true/positive fall videos and 37 non-fall videos. The true/positive fall videos contain: 25 forward falls, 5 backward falls, and 12 sideways falls. The sideways falls cover 6 lateral falls to the right and 6 lateral falls to the left. The non-fall videos contain 23 pseudo/negative falls videos (syncope or the previous falls with recovery) and 14 activities of daily living (ADL) such as sitting down, standing up, lying down, walking a few meters, catching something on the floor, and wearing shoes. Only the skeleton streams from these videos are used in our evaluation process. Figure 2 illustrates samples from FallFree dataset: (a) four postures of person before any action, (b) the postures after falling occurs, and (c) postures of the person when performing a non-fall action.

3.2 Skeleton data extraction

From the dataset fixed in the previous stage, we first develop an extraction software that extract the skeleton data, preprocess it, and then extract the skeleton features. The extracted

Fig. 2 Samples from FallFree dataset: **a** four postures of person before any action, **b** the postures after falling occurs, and **c** postures of the person when performing a non-fall action

skeleton data/properties from the skeleton steams are as follows:

– Body (joints) 3D coordinates
Each skeleton frame contains 3D position data for human skeletons that are visible in the depth sensor. The position of each of the skeleton joints is stored as (x, y, z) coordinates [12]. The (0, 0, 0) point is the center position of the IR sensor on Kinect. Every other point is calculated in terms of the position of the sensor [13]. Three axes thus define the coordinate system, with one unit of these axes equals to one meter. That is, each point in the space (including the skeleton joints) has three values (X, Y, and Z) [13]. X being the position in the horizontal axis, it grows to the sensor's left; Y being the position in the vertical axis, it grows up (note that this direction is based on the sensor's tilt); and Z being the position in the depth axis, it grows out in the direction the sensor is facing.

– Floor clip plane equation
  Each skeleton frame also contains a floor-clipping-plane vector, which contains the coefficients of an estimated floor-plane equation. The skeleton tracking system updates this estimate for each frame and uses it as a clipping plane for removing the background and segmenting players. In addition to joint information, the equation of the floor plane (in the same coordinate system as the joints) is acquired from the Kinect SDK. This provides the plane information in the form of the A, B, C, and D parameters defined in [12]. This equation is normalized so that the explanation of D is the height of the camera from the floor, in meters (the distance from the plane to the origin). Note that the floor may not always be visible or detectable. In this case, the floor clipping plane is a zero vector [12].
– The frame's relative time (timestamp)
  Kinect provides approximately 30 frames per second of data. From each frame, we use the timestamp (in milliseconds) which gets the timestamp of the body frame.

3.3 Skeleton preprocessing and feature extraction

Different users have different shapes and sizes that may not be relevant to the action performed. In addition, users may stand at any distance from the Kinect, which effects the captured skeleton size: If they are close to Kinect, their skeletons appear bigger, whereas if they are far from it, their skeletons appear small. Therefore, besides the original skeleton data, we need a normalized version of it.

Normalization compensates for anthropometric differences, by imposing the same limbs (skeleton segments) lengths for poses obtained from all users. Thus, a normalized skeleton is a skeleton of a fixed size and it has fixed distances between joints [14]. To normalize the skeleton, the torso-centered method used by Rhemyst and Rymix in their project [15] is used in this study. This method uses the reference of the left shoulder and right shoulder joints to normalize all the joints coordinates of the skeleton (cf. Fig. 3a, the skeletons before normalization are drawn in blue and gray colors and the normalized skeleton in red).

We performed normalization on the (X, Y) of each joint while fixing the Z value of each joint of the skeleton as 1m. The skeleton normalization bypasses two main problems that affect the FD performance: the size dependence on the user's distance from Kinect (the skeleton will have a fixed size independently of this distance); and the differences in sizes of the users (their skeleton will have the same size).

In this study, we investigate the 11 basic sets of features which were used in previous works and are shown in Table 1.

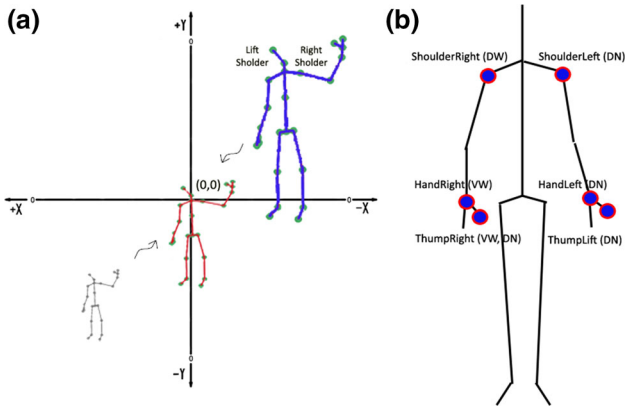

Fig. 3 Skeleton preprocessing and mapping: **a** Skeleton normalization using the torso-centered method and **b** mapping of the seven relevant features (7F) in [8]

3.4 Experimental evaluation

The last stage in our evaluation framework provides answers to our main research question:

– What are the best performing feature sets for the detection of various types of fall?
– What is the most appropriate classifier for these feature sets?

In this section, all the feature sets from Table 1 were investigated in two experiments as follows:

In Experiment 1: All the feature sets from 1 to 10 were calculated twice from the original and normalized skeletons which results in 20 feature sets in addition to the 11th feature set. And this makes us end with 21 feature sets to experiment. Each feature set of them was separately investigated in separated experiment until we obtain the most effective features by comparing their classification results.

In Experiment 2: All the 80 unique skeleton features from Table 1 were calculated as one original feature set contain: 25 velocities, 25 distances from the floor, 25 distances from the Kinect, joints average velocity, the three coordinates of the spine-base position (x, y, z), and the distance of the person's center from the floor. After that, we obtain the most effective features using NCA Feature Selection for Classification.

Finally, we discuss the results of our experiments to answer the previous questions.

3.4.1 Experiments settings

In this experiment, the following steps are implemented:

In Step 1, from the 79 videos that we took from the FallFree dataset, we take 4960 frames containing 2480 fall frames and 2480 non-fall frames. These frames cover all the scenarios of fall and non-fall in dataset.

Table 1 The 11 basic sets of skeleton features

Features ref.	Feature set	Skeleton
Kawatsu et al. [4]	1	The 25 joint velocities in the direction normal to the floor plane
	2	The joints average velocity over all joints and many frames
	3	The 25 joint distances from the floor
Lee and Lee [5]	4	The hip (spine-base) distance from the floor
	5	The hip (spine-base) position (x, y, z)
	6	The hip (spine-base) velocity
Le and Morel [6]	7	The head and spine-mid velocities and distances from the floor
Kwolek and Kepski [7]	8	The distance of the person's center from the floor
	9	The distance of the spine-mid (as the person's center) from the floor
Alzahrani et al. [8]	10	The 25 joints distances from the Kinect
	11	The seven relevant features (7F)

In Step 2, each frame was classified into two classes: 1 for fall and 0 for non-fall. The first class (1 = fall) labeled the frames that belongs to fall videos from the start of fall, while the second class (0 = non-fall) labeled the frames that belong to the non-fall videos.

In Step 3, from each frame, we extract the skeleton data of the users and take a copy of the original (O) joints positions as it is and construct a second normalized (N) copy, as explained in Sect. 3.3.

In Step 4, we extract the skeleton features for the two experiments 1 and 2 presented in Sect. 3.3, including the 11th feature set from every two copies of data (O and N) for Experiment 1, and the 80 unique skeleton features as one original feature set for Experiment 2.

In Step 5, we prepare the learning set by splitting the frames into 70% for training (3472 training frames, divided into 1763 frames in each class) and 30% for testing (1488 testing frames, divided into 744 frames in each class). We insure to make a balance between the number of frames in the two classes to prevent the effect of accuracy results.

In Step 6, for Experiment 1, we use four of the most common and classical supervised learning techniques to build prediction models. These algorithms are C4.5, RF, ANNs, and SVM. Both training data and learning techniques are used to train and generate the classifiers (prediction models). And based on the classification results, we choose the most effective features from the first experiment. For Experiment 2, we use the NCA feature selection for classification to select the most effective features. After that, we test them using the previous supervised learning techniques.

In Step 7, we assess the performance of the generated detection models on the testing data through the sensitivity (SE), specificity (SP), and accuracy (ACC) [7] rates.

3.4.2 Experiment results

In this section, we present the results of the two Experiments 1 and 2 in detail.

Experiment 1 Results. Experiment 1 includes 21 sub-experiments testing the 11 feature sets. The detail of the obtained results is presented in Table 2 (the best accuracy resulted of each feature set is shown in bold font). In Experiment 1.1 (Ex 1.1) using the original 25 joint velocities, we calculate the velocity of each 25 joints in the direction normal to the floor plane as in [4]. The best accuracy of this experiment is 78.8% using RF classifier. In Experiment 1.2 (Ex 1.2) using the 25 joint velocities with normalization, by applying skeleton normalization and then extracting features from Ex 1.1, we obtain 78.8% as the highest accuracy using RF classifier. In Experiment 1.3 (Ex 1.3) using the original joints average velocity, the velocities from Ex 1.1 are averaged over all joints and many frames to calculate the average velocity as in work [4]. The obtained accuracy is 91.3% using C4.5 classifier. In Experiment 1.4 (Ex 1.4) using the joints average velocity with normalization, we first normalized the skeleton and then extract the average velocity from Ex 1.3. These features produced the highest accuracy of 71.8% with the C4.5 classifier. In Experiment 1.5 (Ex 1.5) using the original 25 joint distances from the floor, these features were used by work [4]. They use only a single frame to take the joints positions, so they can calculate the distance from the floor for each joint. These features produced the highest accuracy of 98.1% with the RF classifier. In Experiment 1.6 (Ex 1.6) using the 25 joint distances from the floor with normalization, the features from the previous Ex 1.5 were used, but after we normalized the skeleton. These normalized features give the same accuracy of the original features with 98.1% accuracy by RF classifier. In Experiment 1.7 (Ex 1.7) using the original hip (spine-base) distance from the floor, the hip center joint was used by [5] to calculate different features. The hip center

Table 2 Experimental results of the 21 skeleton feature sets

Ref.	Ex #	Sk	# F	C4.5			RF			ANNs			SVM		
				SE (%)	SP (%)	ACC (%)	SE (%)	SP (%)	ACC (%)	SE (%)	SP (%)	ACC (%)	SE (%)	SP (%)	ACC (%)
[4]	1.1	O	25	73	71.3	72.2	75	82.8	**78.8**	58.1	81.1	69.2	53.8	86.5	70.2
	1.2	N	25	68.6	79.4	73.9	78.9	76.7	**78.8**	49	80.3	64.1	70.8	70.1	70.5
	1.3	O	1	88.5	94.3	**91.3**	86.7	88.8	87.7	85.7	96	90.7	86.2	96	90.9
	1.4	N	1	68	75.8	**71.8**	63.9	63.9	63.9	0	100	50	44.3	91.9	67.3
	1.5	O	25	94.9	96.8	95.8	96.6	99.6	**98.1**	92.8	96.7	94.7	80.5	97.5	80.5
	1.6	N	25	96	93.5	94.8	98.3	97.9	**98.1**	97.1	95.1	96.2	87.8	93.2	90.4
[5]	1.7	O	1	73.3	99.9	**86.2**	79.6	83.9	81.7	74.7	96	85	74.7	96.5	85.3
	1.8	N	1	46.2	84.7	**64.9**	62.5	62.2	62.4	34.2	94.9	63.6	33.9	95.6	63.7
	1.9	O	3	95.1	96.5	95.8	96.2	99.2	**97.6**	84.9	83.1	84	74.6	91.7	82.9
	1.10	N	3	90.2	81.3	85.9	91.4	93.5	**92.4**	68.5	92.5	80.1	71	92.2	81.3
	1.11	O	1	32.7	94.2	62.4	54.9	58.9	56.9	31.8	89.6	59.7	37.9	90.8	**63.5**
	1.12	N	1	63.8	57.2	60.6	55.5	53.8	54.6	0	100	50	54.9	70.8	**62.6**
[6]	13	O	2	93.1	94.7	93.9	93.2	96.5	**94.8**	91.7	94.3	92.9	88.9	95.4	92.1
	1.14	N	2	84.1	88.8	86.4	87.1	91.8	**89.4**	55.7	97.1	75.7	65.5	85.8	75.3
[7]	1.15	O	1	81.6	96.1	**88.6**	84.4	80.7	82.6	78.1	97.1	87.3	78.6	96.9	87.5
	1.16	N	1	59.8	94	76.3	71.6	69.3	70.5	62.5	91.5	**76.5**	61.3	92.2	76.3
	1.17	O	1	80.6	94.9	**87.5**	81.6	81.4	81.5	75	99	86.6	75.3	98.1	86.3
	1.18	N	1	70.8	85	**77.7**	69.5	67.8	68.7	51.6	94.6	72.4	53.9	94.3	73.5
[8]	1.19	O	25	95.3	96.1	95.7	99.3	99.4	**99.4**	94.8	96.4	95.6	74.2	90.6	82.1
	1.20	N	25	94.3	95.1	94.7	97.4	98.2	**97.8**	80.5	96.5	88.2	84.5	94	89.1
	1.21	O &N	7	91.8	91.3	91.5	93.1	96	**94.5**	72.3	95.7	83.6	48.3	98.5	72.6
This paper	2	O	12	99.6	99.2	99.4	99.3	99.7	**99.5**	99.1	100	**99.5**	95.7	99.9	97.7

Bold font represent the best accuracy resulted of each experiment (feature set). e.g: EX 1.1 has 78.7 as the best accuracy

joint in Kinect v1 matches the spine-base in Kinect v2. So, we instead used spine-base distance from the floor. This feature gives 86.2% accuracy by the C4.5 classifier. In Experiment 1.8 (Ex 1.8) using the hip (spine-base) distance from the floor with normalization, the same spine-base distance from the floor was used in this experiment but after we normalized its coordinates. This feature obtained 64.9% accuracy by the C4.5 classifier. In Experiment 1.9 (Ex 1.9) using the original hip (spine-base) position, this feature was used by work [5]. Since the position of each joint is represented by three coordinates values (X, Y, Z), we have three features. These original coordinates have 97.6% accuracy using RF classifier. In Experiment 1.10 (Ex 1.10) using the hip (spine-base) position with normalization, we normalized the positions of the hip (spine-base) coordinates from Ex 1.9. It gives us 92.4% accuracy using RF classifier. In Experiment 1.11 (Ex 1.11) using the original hip (spine-base) velocity, this feature was used as a vertical velocity of the hip by the work [5]. Using the hip (spine-base) velocity only gives us 63.5% accuracy by SVM classifier. In Experiment 1.12 (Ex 1.12) using the hip (spine-base) velocity with normalization, we normalized the hip (spine-base) before calculating the velocity. We obtained 62.6% accuracy using SVM classifier. In Experiment 1.13 (Ex 1.13) using the original head and spine-mid velocities and distances from the floor, these features were used by Le and Morel [6]. This combination of features gives us 94.8% accuracy by RF classifier. In Experiment 1.14 (Ex 1.14) using the head and spine-mid velocities and distances from the floor with normalization, using the normalized combination of features in Ex 1.13, we obtain 89.4% accuracy using RF classifier. In Experiment 1.15 (Ex 1.15) using the original distance of the person's center to the floor, this feature was extracted in work [7]. We calculate the center point between spine-mid and spine-base joints, and then calculate the distance of this center point to the floor. The highest accuracy is 88.6% using C4.5 classifier. In Experiment 1.16 (Ex 1.16) using the distance of the person's center to the floor with normalization, the center point used in Ex 1.15 was normalized first and then its distance to the floor is calculated. This experiment gives 76.5% as the highest accuracy using ANNs classifier. In Experiment 1.17 (Ex 1.17) using the original distance of the spine-mid (as the person's center) to the floor, the person's center in work [7] was considered as spine-mid and spine-base in Experiments 1.17 and 1.11, respectively.

Table 3 Common 12 features resulting from five runs of NCA feature selection

Feature index	Feature name	Feature weight				
		1st run	2nd run	3rd run	4th run	5th run
1	HeadDistanceFromKinect	0.7605	0.6075	0.1248	0.3022	0.0423
2	AnkleLeftDistanceFromKinect	1.0878	0.5108	1.1278	0.7655	0.8224
7	FootRightDistanceFromKinect	1.0369	1.5346	1.3562	1.487	1.2052
12	HipLeftDistanceFromKinect	0.3623	0.1041	0.0786	0.6193	0.326
19	SpineBaseDistanceFromKinect	0.3292	0.1813	0.1282	0.5817	0.4741
26	HeadDistanceFromTheFloor	0.8923	0.8126	0.6968	0.2363	0.746
38	HipRightDistanceFromTheFloor	1.067	0.6141	1.0981	1.3342	0.7306
76	AverageVelocity	1.4214	1.7427	1.4074	2.0686	1.5814
77	DistanceOfPersonCenterToTheFloor	1.3805	1.4088	1.8253	1.8955	1.2107
78	SpineBase-X	1.6818	1.7792	1.6581	2.1562	1.6362
79	SpineBase-Y	1.2366	1.4914	1.231	1.4858	1.3296
80	SpineBase-Z	0.7142	1.5169	1.0297	0.8303	1.0048

The highest accuracy of the spine-mid when considered as the center point is 87.5% with the C4.5 classifier. In Experiment 1.18 (Ex 1.18) using the distance of the spine-mid (as the person's center) to the floor with normalization, here the spine-mid will be normalized before calculating the distance to the floor. This experiment gives 77.7% as the highest accuracy with the C4.5 classifier. In Experiment 1.19 (Ex 1.19) using the original 25 joints distances from the Kinect (O-DfK) proposed in [8], the liner distances from the joints to the Kinect were calculated in this experiment. They produced the highest accuracy of 99.4% using RF. This is in fact the best result in all these sub-experiments. In Experiment 1.20 (Ex 1.20) using the 25 joints distances from the Kinect with normalization (N-DfK) proposed in [8], Ex 1.19 features were used but after we normalized the skeleton. These normalized features give 97.8% as the highest accuracy rate with RF. In Experiment 1.21 (Ex 1.21) using the seven relevant features (7F) in [8], in this experiment, from the original data, the extracted features are: right shoulder distance, right hand, and right thumb velocities. From the normalized data, the extracted features are left hand, left shoulder, left thumb, and right thumb distances (cf. Fig. 3b). This experiment gives 94.5% as the highest accuracy rate with RF.

Experiment 2 Results. Experiment 2 determines the most effective features by using nearest neighbor-based feature weighting algorithm (NCA). It learns a feature weighting vector by maximizing the expected leave-one-out classification accuracy with a regularization term where Lambda (Λ) is a regularization parameter which can be tuned via cross-validation [16]. Tuning means finding the value that produces the minimum classification loss.

In this experiment, the most effective features of the 80 unique skeleton features are selected based on their weights where the feature's weight represents how much each feature influence in a classification problem. We run the algorithms many time but we present the results of only five runs because the NCA algorithm is based on the best Λ value computed from five cross-validation and each run gives different five-folds so different Λ values. So, based on the best Λ value, we calculate the feature weights to select the most effective features.

From the results of five runs of NCA feature selection, we select the common 12 features as the most effective features illustrated in Table 3. And then, we assess their performance using the previously supervised learning techniques as shown in Table 2. These 12 features have 99.5% accuracy using RF or ANNs classifiers.

3.4.3 Experiment discussion

From these two experiments and as shown in Table 2, we can conclude that: (1) the skeleton features-based FD gives very encouraging results and (2) some features give better FD performance/accuracy than others.

In Table 4, we rank the skeleton features sets based on their FD performance (only the best 10 feature sets presented because the lack of space). As shown, the proposed feature sets in Alzahrani et al. work [8] are as follows: O-DfK, N-DfK, and 7F, as 2, 4, and 8 from 22 ranks, with accuracies: 99.4%, 97.8%, and 94.5%, respectively, where the results of NCA experiment ranked 1st with slightly increased accuracy of 99.5%.

Furthermore, for each learning algorithm/classifier, we calculate its gain and loss times. The gain represents how many experiments the classifier wins better FD performance. The loss represents how many experiments the classifier loses for another classifier by not giving the best results. Based on

Table 4 Ranking the best 10 skeleton feature sets based on their FD performance

Rank	Ex #	Features sets	Performance (%)
1	2	**Common 12 features from NCA**	**99.5**
2	1.19	**Original distances from Kinect**	**99.4**
3	1.5	Original distances from floor	98.1
4	1.20	**Normalized distances from Kinect**	**97.8**
5	1.6	Normalized distances from floor	98.1
6	1.9	Original spine-base position	97.6
7	1.13	Original head and spine-mid distances from floor and velocity	94.8
8	**1.21**	**Velocities and Distances from Kinect**	**94.5**
9	1.10	Normalized spine-base position	92.4
10	1.3	Original average velocity	91.3

Bold font represent the feature sets from our previously published work [8] and this work and there best resulted accuracies

the obtained results: C4.5 (gain = 7, loss = 15); RF (gain = 12, loss = 10); ANNs (gain = 2, loss = 20); and SVM (gain = 2, loss = 20), it shows that the RF classifier is the best performing classifier because it gives the higher wins with 12 gains and lower losses with 10 losses.

From these experiments, we conclude that adding some of the original distances from Kinect features with some of the features from previous works increases the classification accuracy and best distinguish fall from non-fall frames, and RF as the best performing classifier. In addition, using the NCA for skeleton feature selection is an efficient way to select the most effective features.

4 Conclusion

In this paper, we proposed an evaluation framework to answer two research questions: Among the various skeleton-based features, which subset best distinguishes various types of fall from non-fall scenarios? and which classifier has the best accuracy with this subset of features? We used the proposed framework to assess 12 basic sets of skeleton features that were used previously to detect the fall through two experiments. In addition, we used scenarios from the challenging FallFree dataset [9].

Our quantitative experimental results highlight that, in general, using skeleton features for fall detection gives very encouraging results. In addition, when comparing the proposed skeleton features by Alzahrani et al. work [8] with features from previous works [4,5,7], O-DfK, N-DfK, and 7F were ranked as 2nd, 4th, and 8th from 22 ranks, giving accuracy rates of 99.4%, 97.8%, and 94.5%, respectively. Furthermore, from the conducted experiments, we can conclude that adding some of the original distances from Kinect features with some of the features from previous works increases the classification accuracy, increases the rank to 1, and best distinguishes fall from non-fall frames, and RF as the best performing classifier.

In addition, using the NCA for skeleton feature selection is an efficient way to select the most effective features.

In our future works, we will focus on applying the proposed features in a real-time FD experiments to detect falls from frame sequences (not frame-by-frame) in order to increase the experiments efficiency and accuracy. Also, this work is extended for users that do not rely on a cane to study the robustness of the proposed features and the RF classifier.

References

1. Mubashir, M., Shao, L., Seed, L.: A survey on fall detection: principles and approaches. Neurocomputing **100**, 144–152 (2013)
2. Bulling, A., Blanke, U., Schiele, B.: A tutorial on human activity recognition using body-worn inertial sensors. ACM Comput. Surv. (CSUR) **46**, 33 (2014)
3. Microsoft. (2016). JointType enumeration. https://msdn.microsoft.com/en-us/library/microsoft.kinect.jointtype.aspx. Accessed 28 Oct 2016
4. Kawatsu, C., Li, J., Chung, C. J.: Development of a fall detection system with Microsoft Kinect. In: Kim, J.H., Matson, E., Myung, H., Xu, P. (eds) Robot Intelligence Technology and Applications 2012. Advances in Intelligent Systems and Computing, vol 208. Springer, Berlin, Heidelberg (2013)
5. Lee, C.K., Lee, V.Y.: Fall detection system based on kinect sensor using novel detection and posture recognition algorithm. In: Inclusive Society: Health and Wellbeing in the Community, and Care at Home. Springer, Berlin, pp. 238–244 (2013)
6. Le, T.-L., Morel, J.: An analysis on human fall detection using skeleton from Microsoft Kinect. In: 2014 IEEE Fifth International Conference on Communications and Electronics (ICCE), pp. 484–489 (2014)
7. Kwolek, B., Kepski, M.: Human fall detection on embedded platform using depth maps and wireless accelerometer. Comput. Methods Progr. Biomed. **117**, 489–501 (2014)
8. Alzahrani, M.S., Jarraya, S.K., Ali, M.S., Ben-Abdallah, H.: Watchful-Eye: a 3D skeleton-based system for fall detection of physically-disabled cane users. Presented at the 7th EAI International Conference on Wireless Mobile Communication and Healthcare (MobiHealth), Austria, Vienna (2017)
9. Alzahrani, M.S., Jarraya, S.K., Ali, M.S., Ben-Abdallah, H.: FallFree: Multiple fall scenario dataset of cane users for monitoring

applications using kinect. Presented at the 13th International Conference on Signal-Image Technology and Internet-Based Systems (SITIS), Jaipur, India (2017)
10. Maldonado, C., Ríos, H., Mezura-Montes, E., Marin, A.: Feature selection to detect fallen pose using depth images. In: 2016 International Conference on Electronics, Communications and Computers (CONIELECOMP), pp. 94–100 (2016)
11. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. Newsl. **11**, 10–18 (2009)
12. Microsoft. (2016). Coordinate spaces. https://msdn.microsoft.com/en-us/library/hh973078.aspx. Accessed 29 Oct 2016
13. Microsoft. (2016). Coordinate mapping. https://msdn.microsoft.com/en-us/library/dn785530.aspx. Accessed 29 Oct 2016
14. Zanfir, M., Leordeanu, M., Sminchisescu, C.: The moving pose: an efficient 3D kinematics descriptor for low-latency action recognition and detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2752–2759 (2013)
15. Rhemyst and Rymix. (2011/2017). Kinect SDK Dynamic Time Warping (DTW) Gesture Recognition. http://kinectdtw.codeplex.com/. Accessed 2 Jan 2017
16. Yang, W., Wang, K., Zuo, W.: Neighborhood component feature selection for high-dimensional data. JCP **7**, 161–168 (2012)