Automate Fall Detection using MediaPipe Keypoint-Extraction

Linh Tran
Dept. of Control Engineering and Automation
Ho Chi Minh University of Technology
Ho Chi Minh, Viet Nam

Thai Hoang Huynh
Dept. of Control Engineering and Automation
Ho Chi Minh University of Technology
Ho Chi Minh, Viet Nam

ABSTRACT
Fall accidents are becoming an increasingly common and serious issue worldwide, particularly among the elderly. These incidents are not only the leading cause of injuries and fatalities in older adults but also significantly impact their quality of life. Therefore, the research and development of automatic fall detection systems have become increasingly significant. Current fall detection devices often offer wearable solutions, which can cause unnecessary inconvenience and even reduce effectiveness. Elderly individuals, especially those in poor health, may forget to wear these devices. Therefore, developing an automatic fall detection system offers a more effective solution to address this issue. The proposed system uses Media Pipe for human body key-point extraction combined with a classification model of a Long-short term memory (LSTM) network or K-Nearest-Neighbor (KNN) algorithm. It is capable of identifying the fall actions of humans in real-time environment.

Keywords
Key-point extraction, Fall detection, MediaPipe, KNN, LSTM

1. INTRODUCTION
According to statistics from the World HealthOrganization (WHO) [1], there are 684,000 fall-related deaths annually, making falls the second leading cause of unintentional injury deaths after road traffic accidents. Over 80% of fall-related deaths occur in low- and middle-income countries, with the highest mortality rates found in adults over the age of 60.

Automating the fall detection process saves time and effort and creates a safer environment for the elderly. This can enhance their quality of life and provide them and their families peace of mind.This research topic not only addresses practical needs in improving safety and healthcare for the elderly but also reflects our commitment to leveraging technology to tackle significant societal challenges. Fall detection devices currently available on the market are often designed to be worn on the body[2][3][4], which can lead to unnecessary inconvenience and may even reduce their effectiveness. Older adults, particularly those with compromised health, may frequently forget to wear these devices. Therefore, the development of an automated fall detection system using cameras would address this issue more effectively.

Computer vision-based fall detection is a critical area of research, especially for elder care and home safety monitoring. Developing an automated fall detection system is challenging due to the short duration of fall events and the complex body postures involved, which distinguish falls from other activities. Furthermore, variations in background and real-time video angles pose significant hurdles.

Pose estimation models have emerged as a promising alternative for automated fall detection and classification. The research presented in [5][6] specifically investigated the efficacy of pose estimation for this purpose. Utilizing models such as OpenPose, PoseNet, DeepPose,and AlphaPose, the authors extracted key kinematic metrics, including body tilt angles and inter-joint distances. Their findings suggest that pose estimation-based fall detection achieves superior accuracy compared to traditional sensor-based methodologies. However, this approach remains susceptible to challenges related to video input quality and camera perspective.

Similarly, [7] proposes a fall detection system integrating pose estimation with a Gated Recurrent Unit (GRU) network to process sequential data. GRU is used to analyze the kinematic changes in joints across video frames. This method not only detects falls but also predicts the direction of the fall. Experimental results demonstrate high accuracy, particularly in complex scenarios such as falling while standing up or falling backward.

In addition to traditional pose estimation models like OpenPose and AlphaPose, [8] employs MediaPipe Pose, a recently introduced skeleton detection model, to extract real-time joint features. This is combined with an IoT platform to provide automated alerts. The system's advantages include continuous monitoring at a low cost and effective AI-IoT integration. Real-world experiments show that the system is suitable for deployment in homes or nursing facilities, though improvements are needed in low-light environments.

Beyond fall detection, [9] identifies the direction of falls, a critical factor in assessing the severity of incidents. By analyzing features such as the center of gravity and body orientation, the system classifies scenarios like lateral falls, forward falls, or backward falls. This method performs well across diverse testing environments but encounters difficulties with atypical postures.

Recognizing that a fall is a sequential motion, Long Short-Term Memory (LSTM) networks, particularly when combined with an attention mechanism as explored in [10], offer a promising approach. The authors propose leveraging LSTM's inherent capacity for processing sequential data, such as human movement, and integrating an attention mechanism to focus on the most salient portions of the motion sequence for fall recognition. This aims to improve both accuracy and robustness compared to traditional fall detection methods. The effectiveness of this LSTM-attention approach is evaluated through experiments benchmarked against established baselines. Key aspects of this research include the specific LSTM-attention architecture employed, the feature extraction techniques used, details of the dataset utilized, and the performance metrics considered. Ultimately, this work seeks to contribute to the development of more reliable and

automated fall detection systems, with a particular focus on applications within elderly care.

Motivated by the related research published in the literature, this study presents an automated fall detection system based on an LSTM architecture and the KNN algorithm, utilizing MediaPipe for human body keypoint extraction. The proposed real-time system will reduce the risk of late recognition of fall action.

The rest of the paper is organized as follows: Section 2 details the background of keypoint extraction and classification models. Section 3 presents the architecture, algorithm and data model of ourproposed fall detection solution. Section 4 presents the outcomes of the test experiments and evaluates the system's performance. Finally, the conclusion and future work are discussed in section 5.

2. BACKGROUND

2.1 Keypoint extraction

Keypoints are extracted from the video frames collected from the videos utilizing the MediaPipe Pose in the suggested method. It is a machine-learning pipeline comprising numerous models operating together. A Pose Landmark Model that returns high-accuracy human pose keypoints from the Region of Interest (ROI) of each frame determined by the pose detector.

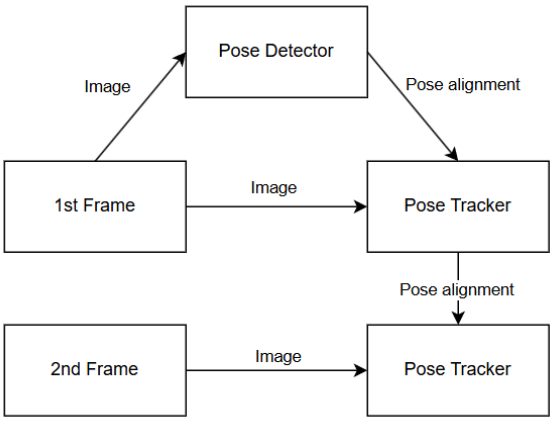

Fig 1: Pose detector and tracker of MediaPipe

In the domain of pose estimation, a two-stage machine learning pipeline, commonly referred to as the detector-tracker pipeline, is demonstrated in Figure 1. This pipeline comprises two distinct phases: detection and tracking. The initial detection phase employs a detector, typically a pre-trained object detection model or a coarse pose estimation model, to identify the Region of Interest (ROI) within the image frame. This ROI delineates the area containing the subject or individual whose pose is to be precisely estimated. Standard detectors employed in this context include YOLO, SSD, and Faster R-CNN for general object detection. At the same time, more specialized models like OpenPose and AlphaPose are utilized for human pose detection.

Subsequently, the tracking phase utilizes a tracker to predict the precise location of pose keypoints within the identified ROI. These keypoints represent anatomical landmarks or salient points on the human body, such as eyes, nose, elbows, and knees. The number of keypoints can vary depending on the specific application and desired level of detail (e.g., 17 keypoints for the COCO dataset, or 33 keypoints as mentioned). Standard tracking algorithms include Kalman filters, MeanShift, Kernelized Correlation Filters (KCF), and DeepSORT, each exhibiting distinct advantages and disadvantages in speed, accuracy, and robustness to challenging conditions such as occlusions or rapid movements.

A crucial aspect of video applications is the optimization of computational efficiency. Instead of executing the detector on every frame, it is typically applied only to the first frame to establish the initial ROI. For subsequent frames, the ROI is inferred from the predicted keypoint positions of the preceding frame. This approach relies on the assumption that pose changes between consecutive frames are minimal, allowing the tracker to efficiently predict the ROI without the computationally intensive detectionprocess. ROI inference can be performed through various methods, such as creating a bounding box encompassing the predicted keypoints with a defined margin or employing more sophisticated techniques based on keypoint motion estimation. However, this method is not without limitations. Particularly, inaccuracies in tracking and pose estimation may arise when the subject undergoes rapid movements, experiences occlusions, or exhibits abrupt pose changes. In such instances, re-detection, involving the re-application of the detector, can be employed to rectify the tracking process.

Thus, the extracted keypoints are stored NumPy array as Python files and get stored. Figure 2 represents the Keypoint extraction process with Mediapipe module.

a) No Fall

b) Fall

Fig 2: Keypoint extraction process

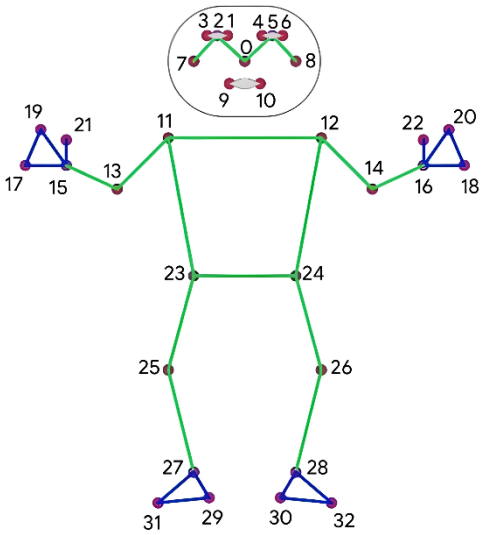

Fig 3: Keypoints detect by MediaPipe [11]

2.2 Classification model

2.2.1 LSTM model

LSTM is an extension of Recurrent Neural Networks (RNN); it is chosen because of the inability to remember a long sequences of memory and forgetting redundant information, which, in detail in this research is the "Fall" action. In this problem, we have used LSTM as one of our choices, where information is transferred through gates in a sequence chain, including the relevant information from earlier states. The figure of LSTM model is provided in Figure 4 for a better understanding of the flow and concept of LSTM. Here,

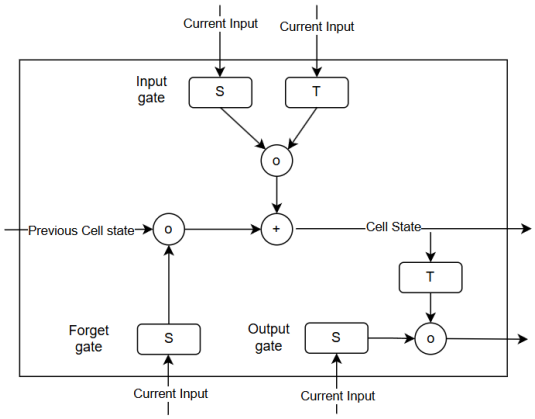

Fig 4: LSTM working process

- o denotes for Pointwise multiplication.
- + denotes for Pointwise addition.
- S denotes for Sigmoid activation
- T denotes for Tanh activation

The forget gate is an essential part of this model and is pre-trained to judge whether information from input data would be remembered/forgotten fully or partially. In this research, we only use one layer of LSTM (50 units, sigmoid activation function) and a Dense Layer (1 unit) as our problem only aiming to classify two classes Fall (1) and No Fall (0).

2.2.2 KNN model

The K-Nearest Neighbors (KNN) algorithm is used to classify the actions of identified objects. KNN is a machine learning algorithm for classification, where each new dataset collected from an object is compared to the k most similar datasets from the training set. The output is classified into two actions: "Fall" or "No Fall."

The primary limitation of KNN lies in its execution time due to the lack of a preprocessing step. Each new data point to be classified must be compared with the k most similar data points in the unprocessed training set, which can result in inefficiency. The Euclidean distance is one of common measures to find the nearest data, which is given by

$$d(K_1, K_2) = \sqrt{\sum_{i=1}^{n} (K_{1n} - K_{2n})^2}$$

where $K_1 = (k_{11}, k_{12}, \dots, k_{1n})$ and $K_2 = (k_{21}, k_{22}, \dots, k_{2n})$ [12].

3. PROPOSED FALL DETECTION SOLUTION

This proposed method uses a vision-based categorization system for "Fall" action. In contrast to a static pose, a fall pose motion comprises a complex motion of body gestures. A single frame can determine a static pose or action, whereas a sequence of bodies determines a dynamic action. MediaPipe module is used to extract the key points. The body pose is considered the region of interest, and the key points are extracted from these regions. Then, the key points extracted are given as input to the built LSTM model, and then the camera shows a fall alert message to indicate if there is a chain of fall action throughout the input frames.

3.1 Architecture

The biomechanics of falls are complex and exhibit significant variability depending on both the direction of the fall and the individual's reactive movements. The proposed fall detection system, depicted in Figure 5, is structured around three core components: video acquisition, keypoint extraction, and fall classification. In selecting a suitable model for keypoint extraction, several state-of-the-art pose detection/estimation

models were evaluated. MediaPipe was ultimately chosen due to its demonstrated accuracy and cross-platform compatibility, making it well-suited for both the current research and potential future development.

The extracted keypoints are structured into an array for subsequent training. To evaluate performance, two distinct action classification models are employed in parallel: a Long Short-Term Memory (LSTM) network and a K-Nearest Neighbors (KNN) algorithm. The LSTM network, with its gated architecture comprising input, output, and forget gates, is well-suited to capture the temporal dependencies inherent in human actions and address the short-term memory requirements of fall detection. Simultaneously, the KNN algorithm provides a computationally less complex and faster alternative, potentially more amenable to real-time implementation.

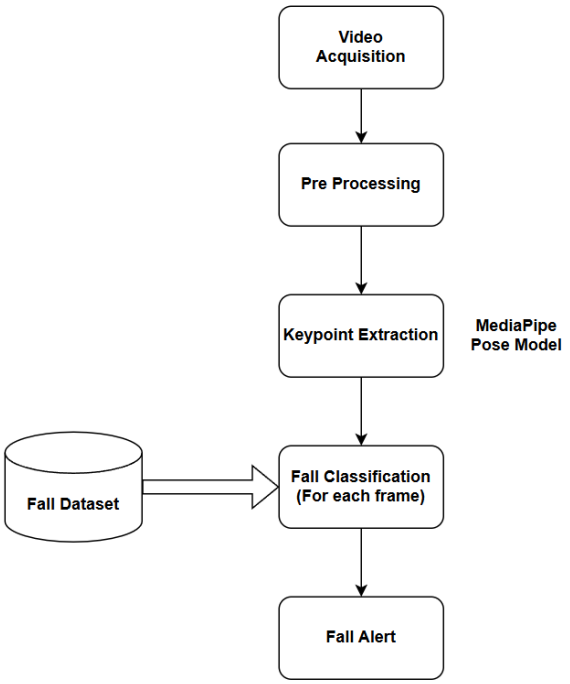

Fig 5: Architecture of the proposed fall detection system

Below is the step-by-step algorithm for an overall process of the proposed system:

BEGIN

1. Dataset videos are created for fall activity.

2. The recorded videos get converted to the image frames using OpenCV.

3. Training.

(3.1) Keypoint Extraction done with Mediapipe.

(3.2)LSTM/KNN classification for each frame (model, input shape).

4. Fall detection

(4.1) Fall detection using classification results from a set of 8 frames continuously when 4 out of 8 are classified as “Fall”

END

3.2 Features for Fall Detection

In this research, the features used to detect the fall action are proposed as follows. In Figure 6, recognizing the fact that the elbow joint angles (α_1,α_2), hip joint angles (α_3,α_4), and knee joint angles (α_5,α_6) are usually changed significantly during the fall action, the changes of these angles in 4 consecutive video frames are adopted as features to detect the fall. These joint angles can be easily calculated from the keypoints extracted using MediaPipe. There are 24 features calculated from the joint angles as follows:

$$f_{ij} = a_i(k+1-\ j) - a_i(k-\ j), (i = 1,..,6; j = 1,..,4)$$

Besides the changes in joint angles, the change of Shape Aspect Ratio (SAR), which has been proven to be essential for “Fall” judgment [13], is also used as an additional feature. The SAR value can be obtained by dividing the height by the width of the bounding box of the human body detected in each video frame.

We can form a feature matrix *F*fromthe mentioned features as below:

$$F = \begin{bmatrix} f_{11} & \cdots & f_{61}SAR_1 \\ \vdots & \ddots & \vdots \quad \vdots \\ f_{14} & \cdots & f_{64}SAR_4 \end{bmatrix}$$

In total, a matrix with 28 elementsis used as input data for Fall Detection.

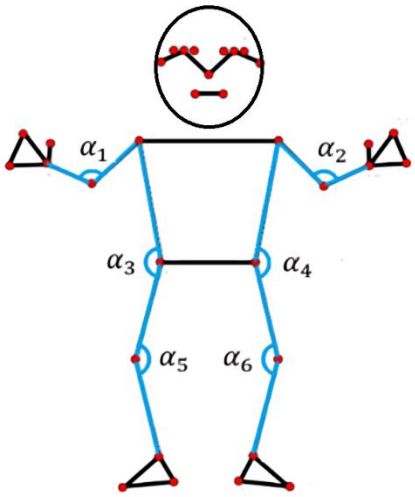

Fig 6: Joint angles used in fall detection

3.3 Dataset Acquisition

The CAUCAFall Dataset [6], supplemented by our own collected data, was used to train the fall classification models. In the CAUCAFall Dataset, body poses were captured in video format and converted into sequences of image frames as in Figure 7. Recognizing that falls typically occur quickly, videos were recorded for a short duration (5-7 seconds) to minimize the imbalance between "Fall" and "No Fall" instances. A total of 40 videos were recorded, with keypoints extracted from each frame and stored in a data file for training the classification model. To further enrich the dataset, we self-collected an additional 12 videos. The combined dataset comprises 29 "Fall" videos, depicting various fall types (forward, backward, left, right, sitting), and 23 "No Fall" videos, showcasing other actions such as hopping and kneeling.

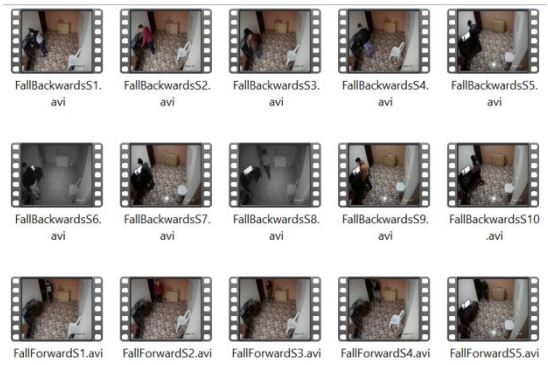

Fig 7: Dataset use for training and validation

After frame-by-frame extraction across all videos in the keypoint dataset, a total of 17,301 frames were used for training and validation, employing a 70/30 split ratio. Each input data point was manually labeled as either "Fall" (1) or "No Fall" (0).

4. TEST RESULT AND PERFORMANCE EVALUATION

4.1 Model training

A total of 17,301 sets of input data were used for training purposes for both LSTM and KNN models, with variable Camera position, background, and lighting conditions.

For KNN classification model, we process training in 3 different values of$k$(3, 5 and 7)to assess the impact of this parameter on model accuracy.

Its Precision, Recall and F1-Score evaluate the performance of the built method.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positvie}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 1 provides the performance of KNN classification model for each frame with input support by MediaPipe Pose keypoint extraction.It is clear that k = 5 provide the highest accuracy with nearly 91.99%. It is evident that a k value that is too large or too small is not suitable; a small k value leads to less reference data to compare with, or a large k value can increase noise data.

Similarly, for LSTM, we evaluate model performance with epoch = 50, batch size =16. Table 2 provide the performance result of MediaPipe-LSTM model.

Table 1. Performance result of MediaPipe-KNN model

k value	Class	Precision	Recall	F1-score	Accuracy
k=3	0 (No Fall)	0.93	0.99	0.96	91.63%
	1 (Fall)	0.29	0.60	0.60	
k=5	0 (No Fall)	0.92	0.99	0.96	**91.99%**
	1 (Fall)	0.33	0.30	0.28	
k=7	0 (No Fall)	0.92	1.00	0.96	92.19%
	1 (Fall)	0.42	0.20	0.10	

Table 2. Performance result of MediaPipe-LSTM model

Class	Precision	Recall	F1-score	Accuracy
0 (No Fall)	0.9324	0.9886	0.9597	92.34%
1 (Fall)	0.5217	0.1481	0.2308	

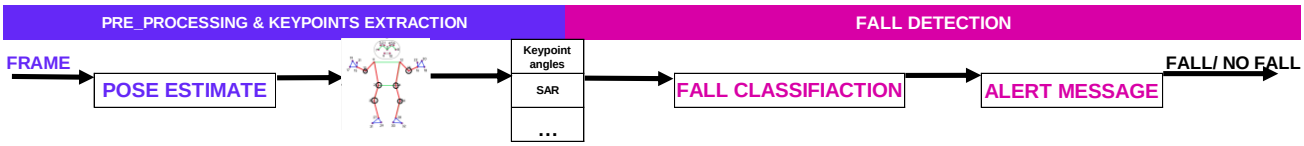

Fig 8: Fall alert raising flow

4.2 Result Analysis

Following model training, a "Fall Alert" mechanism was implemented, as depicted in Figure 8. This mechanism triggers a fall alert if three out of eight consecutive frames are classified as "Fall" by the pre-trained model. Recognizing that a single frame is insufficient for reliable fall detection, a set of 30 short videos, comprising 11 fall events, was used to evaluate the algorithm's performance. Evaluation was conducted on a system with an Intel Core i7 processor, 16GB RAM, and an NVIDIA GeForce RTX 3050 Ti GPU. In addition to accuracy, precision, and recall, the Frames Per Second (FPS) metric was also considered to assess the method's processing speed. The evaluation results are presented in Table 3.

Table 3. Performance result of model

Methods	Pre.	Rec.	F1.	Acc.	FPS
MediaPipe+ LSTM	0.73	1.00	0.84	90.00%	12.76
MediaPipe+ KNN	0.82	0.84	0.83	83.33%	25.12

The comparative evaluation of the LSTM network and the K-Nearest Neighbors (KNN) algorithm reveals key insights into

their respective performance in fall detection tasks. Overall, the LSTM model demonstrates slightly higher accuracy than the KNN algorithm. A closer look at class-wise accuracywhere class 0 denotes "No Fall" and class 1 denotes "Fall" shows that the LSTM model significantly outperforms KNN in correctly identifying "Fall" instances. In contrast, both models achieve comparable accuracy for the "No Fall" class, with each reaching approximately 90%.

This discrepancy in classification performance between the two classes can be largely attributed to the nature of the training data. The dataset comprises short video sequences, wherein fall events typically span only in a few frames. Conversely, the majority of frames depict non-fall activities. This inherent class imbalance biases the models towards the "No Fall" class, making it more challenging to accurately detect the comparatively rare "Fall" frames. The LSTM model, with its temporal learning capabilities, is better equipped to capture the sequential patterns associated with fall events, enabling improved detection of these short, transient actions.

In terms of computational efficiency, however, KNN holds a substantial advantage. The KNN algorithm operates at an average frame rate of approximately 24 frames per second (FPS), making it highly suitable for real-time deployment. In contrast, the LSTM model achieves a lower processing speed of about 12 FPS, which may pose limitations in latency-sensitive applications. This notable difference highlights KNN's suitability for resource-constrained or performance-critical environments, especially when additional features or sensors are to be integrated without significantly impacting responsiveness.

In summary, while the LSTM model provides better accuracy,particularly in identifying fall events,its higher computational cost makes it less ideal for real-time applications. KNN, on the other hand, offers a balanced trade-off between speed and accuracy, positioning it as a more pragmatic choice in scenarios where rapid decision-making and low-latency processing are critical.

5. CONCLUSION

An advantage of this method is that it is designed to be a real-time interface and easy to access. The keypoint extraction model effectively extracts critical information for action classification based on human pose, while the proposed combination of techniques ensures faster making it suitable for real-world applications. Implementing the LSTM network and KNN algorithm increases the option for Fall detection and proposes an interface that can reduce the risk of Fall, especially for elderly people. The challenge is enhancing the method's accuracy. Unlike other approaches, it achieves superior speed, which is crucial in real-time fall detection scenarios.

The system's accuracy remains a challenge to improve, but the solution can be embedded in a compact, convenient device, ensuring accessibility and practicality. This project's potential is vast, with many possibilities for future enhancements. For example, the dataset could be expanded to include different backgrounds and fall poses, while model adjustments could fine-tune performance. The input image frame size could also be standardized to match the training dataset for improved results.

Looking ahead, the developed method can be adapted for specific embedded computers such as Raspberry Pi or Jetson Nano, optimizing device performance. Furthermore, it holds great promise for expansion into mobile or web applications, increasing accessibility to a wide range of users.

6. REFERENCES

[1] World Health Organization (WHO) 2021. Fact-sheets: Falls

[2] Rihana, S. and Mondalak, J. 2016. Wearable Fall Detection System. Middle East Conf. Biomed. Eng. (Nov. 2016), 84-87.

[3] Saha, B., Islam, M. S., Riad, A. K. I., Tahora S., Shahriar H. and Sneha S., 2023. BlockTheFall: Wearable Device-based Fall Detection Framework Powered by Machine Learning and Blockchain for Elderly Care. IEEE 47th Annual Computers, Software, and Applications Conference (Jun. 2023), 1412-1417.

[4] Amir, N. I. M., Dziyauddin, R. A., Mohamed, N., Ismail, N. S. A., Zulkifli, N. S., Din, N. M. 2022. Real-time Threshold-Based Fall Detection System Using Wearable IoT. 4th International Conference on Smart Sensors and Application (Jul. 2022), 173-178.

[5] Serpa, Y. R., Noqueira, M. B., Neto, P. P. M. and Rodrigues, M. A. F. 2020. Evaluating Pose Estimation as a Solution to the Fall Detection Problem. IEEE 8th International Conference on Serious Games and Applications for Health (Aug. 2020),1-7.

[6] Ziwei, C., Yiye, W. and Wankou, Y. 2021. Video Based Fall Detection Using Human Poses.

[7] Kang, Y., Kang, H. and Kim, J. 2021. Fall Detection Method Based on Pose Estimation Using GRU. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (Feb 2021), 169-179.

[8] Bugarin, C. A. Q., Lopez, J. M. M., Sambrano, M. F. C. and Loresco, P. J. M. 2022. MachineVision-Based Fall Detection System using MediaPipe Pose with IoT Monitoring and Alarm. IEEE 10th Region 10 Humanitarian Technology Conference (Sep. 2022), 269-274

[9] Yuan, C., Zhang, P., Yang, Q. and Wang, J. 2022. Fall detection and Direction Judgement Based on Posture Estimation. Discrete Dynamics in Nature and Society (June 2022), 1-12.

[10] Guerrero, J. C. E., Espana, E. M., Anasco, M. M. and Lopera, J. E. P. 2022. Dataset for human fall recognition in an uncontrolled environment.

[11] Valentin, B. and Ivan, G. 2020. On-device, Real-time Body Pose Tracking with MediaPipeBlazePose.

[12] Asha, G. K., Jayaram, M. A. and Manjunath, A. S. 2010. Combining Akaike's Information Criterion (AIC) and the Golden-Section Search Technique to find Optimal Numbers of K-Nearest Neighbors. International Journal of Computer Applications (May. 2010), 80-87.

[13] Min, W., Zou, S. and Li, J. 2018. Human fall detection using normalized shape aspect ratio. Multimedia Tools and Applications (Nov. 2018), 14331-143.