IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 May 2025, accepted 8 June 2025, date of publication 11 June 2025, date of current version 23 June 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3578683

APPLIED RESEARCH

Multi-Person Fall Detection Using Data Assimilation Method With Kalman Filter

JINMO YANG, YE JIN JIN, AND R. YOUNG CHUL KIM

SE Laboratory, Hongik University, Sejong-si 30016, Republic of Korea

Corresponding author: R. Young Chul Kim (bob@hongik.ac.kr)

This work was supported in part by Korea Creative Content Agency (KOCCA) Grant funded by the Ministry of Culture, Sports and Tourism (MCST), in 2025 through the Project "Artificial Intelligence-Based User Interactive Storytelling 3-D Scene Authoring Technology Development," under Project RS-2023-0022791730782087050201; in part by Korea Research Foundation's Brain Korea 21 Fostering Outstanding Universities for Research (BK21 FOUR) funded by the Ministry of Education (MOE), in 2025, through the Project "Ultra-Distributed Autonomous Computing Service Technology Research Team," under Project 202003520005; and in part by the 2025 Hongik University Innovation Support Program Fund.

This work involved human subjects or animals in its research. The authors confirm that all human/animal subject research procedures and protocols are exempt from review board approval.

ABSTRACT Fall detection is an essential technology for ensuring the safety of elderly individuals, as falling accidents are critical and can cause significant functional damage in old age. Our previous work focused on fall detection with just a single individual, using simple statistical aggregation to achieve low model complexity and moderate accuracy. However, the pose estimation model may score low confidences on individual body parts (landmarks), affecting the aggregated statistics, thereby resulting in incorrect fall detection status. To solve this problem, we propose an enhanced fall detection method that adopts the Kalman filter for improved landmark and fall detection and object tracking for multi-person fall detection. Specifically with the Kalman filter, we reduce noise in network model's heatmaps and landmarks with an adaptability across different input video sources. Compared to other methods that analyze AI models' hidden layers and the layer outputs for providing confidence of measurements, this approach has a big advantage of plug-and-play for the pose estimation models and other streaming models that provide confidence. Tested on a computer with i7-13700HK, RTX 4070, 32 GB RAM, and a full HD camera, our method achieved an F1-score of 0.944 in the multi-person setup and 0.933 in the single-person setup.

INDEX TERMS Adaptive filtering, artificial intelligence, data assimilation, fall detection, Kalman filter, pose estimation.

I. INTRODUCTION

Living standards have improved over the years. This affects the human life expectancy level, causing it to continue to rise. According to the World Health Organization, the global average human life expectancy has increased from 66.9 years in 2000 to 73.1 years in 2019 [1]. Many factors contribute to this, such as medicine, nutrition, health services, and other improvements [2]. However prolonged the life expectancy has been, accidents may still happen, such as fall injuries. In 2021, fall injury placed second in the leading causes of death by unintentional injuries [3]. This is a significant concern because older individuals receive severe functional damage (functional decline) from falling accidents even if he or she has survived it [4], [5]. Therefore, immediate medical care is needed to prevent this damage from being a death sentence.

Various research in fall detection uses wearable, vision, ambient, and/or fusion-based systems [6]. To accurately capture a falling motion, most works use and develop state-of-the-art artificial intelligence (AI) models for detecting the state of the human subject as fall or non-fall. In those approaches, however, increasing the performance in correctness comes with a drawback; the performance in model complexity decreases as a trade-off. This can be seen with most neural networks that process spatio-temporal input [7],

The associate editor coordinating the review of this manuscript and approving it for publication was Yu-Da Lin.

[8]. Our previous work focused on developing a single-person fall detection model with low model complexity and moderate correctness performance [9]. This was done by capturing the state of the human subject as a single parameter in every frame and simply aggregating these parameters as a single decision parameter. This also differed from most models that accumulated the landmark distribution every frame and continuously processed the data in resource-intensive spatio-temporal networks. As an achievement, our model scored the lowest model complexity of 426 FLOPs/f (The corrected calculation is found in [10].) among many other models [9].

Having achieved the lowest model complexity, we now focus on increasing the performance in correct detection as well as extending the detection capability to multi-person fall detection. Increasing the model's correctness performance is always desired. Distinguishing fallen individuals from one another is also crucial for accurately tracking each individual's state of fall or non-fall. All of this is to be done while the cost remains low. To do this, we propose the simple multi-person fall detection model (SMFDM), which uses the computer vision pose estimation technique combined with the Kalman filter for increased correctness, object tracking for multi-person identification, and a series of simple calculations to detect human falls. Specifically for the filter, the landmarks parsed from the network's feature maps–heatmaps–are used to evolve the maps, combining each person's movement into the maps for enhanced results. Also, the confidence of the pose estimation model in inferencing landmarks is applied to the measurement covariance matrix of the Kalman filter for adaptability. The main contributions of this paper can be summarized as follows.

1) The Kalman filter is used to estimate the position and velocity state variables of 2D and 3D landmarks. For the 2D space, the difference between the 2D estimated and raw landmark positions is calculated, which is used to evolve 2D heatmaps accordingly. Then the 3D landmarks are filtered. This process effectively combines the pose estimation model and the filters to increase the correctness of the fall detection.
2) The heatmaps contain the confidence of each landmark. We use this information as the uncertainty in the measurement, providing adaptability of the filter on different input source environments as well as model exchangeability.

This paper is organized as follows: Section II presents related works, Section III describes SMFDM, Section IV presents the experiment, Section V discusses the results, and Section VI concludes the paper.

II. RELATED WORKS

This paper uses the pose estimation model to detect falling accidents. The Kalman filter method improves correctness, and object tracking extends single-person detection to multi-person detection. As such, this section presents works related to our paper: fall detection systems, the Kalman filter, and object tracking.

A. FALL DETECTION SYSTEMS

Most fall detection systems use wearable, vision, ambient, and/or fusion techniques [6], [11].

1) WEARABLE-BASED SYSTEMS

For wearable-based systems [12], individuals wear wristbands or belts with accelerometers that process the body's movements. If the wearable senses an abrupt downward motion of great magnitude, it is highly likely that the individual has fallen.

Yu et al. [13] proposed a lightweight yet robust neural network system, TinyCNN, that can detect falls from the inertial data of wearable sensors. They used the publicly available KFall [14] and SisFall [15] datasets, which contain diverse kinds of falls and distinct subjects, which helped generalize their model. Furthermore, they used a quantization technique to reduce the model size on top of low model complexity [16]. Through this, they achieved 0.9983 sensitivity and 0.9917 specificity on the KFall dataset and 0.9829 sensitivity and 0.9849 specificity on the SisFall dataset. However, their system constantly consumed battery power, which they mentioned to address in future work.

Hashim et al. [17] proposed a method of combining multiple wearable sensors to detect falls in patients with Parkinson's disease. The accelerometer node was composed of tilt switches and an accelerometer. The tilt switches detected extreme tilt of the body orientation of the patient in a falling accident, and the accelerometer detected falling motion. Myoware muscle sensor node measures force pressure and the muscle activity of the patient. The results are received by the receiver node for the alert. For test cases, they created thirteen sequences of activities of daily living (ADL) and fall events, and scored 0.9305 to 1.0 accuracy with various acceleration thresholds. A substantial advantage of this system is that sensors of the system wake and sleep at an appropriate time step of the algorithm flow for low battery consumption. However, detection cannot be made if the accelerometer node or the Myoware node failed to collect or transmit data to the receiver node, regardless of the cause.

Overall, the main concern in these systems is whether the wearable can be worn or turned on at all times. These devices run on battery and must be taken off for recharge. Also, the wearable must feel comfortable on the wearer's wrist or waist. Sweating from the wrist or waist may lead to the individual taking the wearable off [18], [19]. A woman wearing a dress may choose not to wear the device on the waist because the device may ruin the beauty of the outfit [20].

2) VISION-BASED SYSTEMS

Vision-based systems do not have these downsides, as there is no need for the wearable. Instead, a camera captures the motion of the individual in the view, detects sudden movement, and decides if a fall has occurred. However, the primary concern with the vision method is that the individuals may be identified, possibly leading to a privacy violation. Because of this, minimal features from the video are extracted to recognize the individuals' orientations without direct identification via silhouette segmentation or pose estimation.

Lee et al. [21] introduced a novel deep-learning model designed to process dynamic vision sensor (DVS) data. Since a DVS captures pixel-level RGB changes and processes this information into a compressed stream, privacy is enhanced compared to that with traditional cameras. The authors adapted the original temporal segment networks [22] to better handle DVS data. Specifically, they removed the network layers that handle spatial information but retained those for temporal information. This allowed for minimizing spatial processing for optical flow, which involves minimal RGB variations while preserving the model's ability to analyze changes in time. With additional optimizations to improve performance, their model achieved an F1-score of 0.955 at 31.25 FPS on the GPU. One downside with their approach is the affordability of a DVS, with low-grade products being around €1,600.00 [23].

Some systems use pose estimation techniques to identify the landmarks on the human body [24]. This landmark data represents body orientation and detects movement while respecting privacy. Chang et al. [25] introduced the pose estimation-based fall detection methodology, which combines a pose estimation model with a deep-learning model for human action recognition to detect falls from video streams. For pose estimation, they proposed OpenPose-light, which replaces VGG-19 with MobileNetV2 from 2D Lightweight OpenPose [26]. This allows for faster landmark inference and fewer parameters without compromising performance. For fall detection, they incorporated an LSTM module [27], which processes the input from OpenPose-light pose estimation map. Furthermore, they applied the centroid tracker [28] to identify each individual in the frame. The tracker followed the human head as it was tested to be the most visible part of the human body, but there was no mention of the case where the head was hidden from the camera's view. The overall detection achieved an accuracy of 0.981.

3) AMBIENT SENSOR SYSTEMS

Whereas most vision-based systems implement techniques to protect individuals' privacy in footage, ambient sensor systems are inherently more robust in respecting privacy. Chen et al. [29] introduced a low-resolution infrared sensor system designed to monitor the movement of elderly individuals for fall detection from temperature. Their system used two synchronized sensors to capture the scene, and the scene images were preprocessed for reduced noise and elimination of the background. They applied the angle of arrival positioning algorithm to track the individuals and used the K-nearest neighbor classifier to identify human actions as either a fall or a non-fall. Low resolution reduced the computational cost of the system, but the accuracy of 0.93 is low among other infrared fall detection systems.

Tanaka et al. [30] built a virtual home with a motion sensor on the ceiling with an aging person modeled as an autonomous agent to simulate smart homes [31]. Having produced nine years of data collected on virtual sensors, they trained and compared a rule-based method, HMM, random forest, and decision tree. For the decision tree, the scores were 0.96 sensitivity, 0.29 false alarms per day, and 83.43 hours per false alarm. The simulated environment was that the elderly person lived alone and performed ADLs with no awkward motion. However, the usual occurrences of fall accidents differed from those of the real-world environment.

4) FUSION SYSTEMS

Fusion-based systems combine inputs from multiple sensors to use all available information, resulting in improved detection correctness compared to single-device systems [6]. Abro et al. [32] designed a multi-modal fall detection method that processed vision and inertial sensor data from the UR Fall Detection (URFD) dataset [33]. With the vision sensor data, human silhouette and subsequently skeleton were modeled, from which features were extracted. With the inertial data, a sliding window captured both static and dynamic motion, Parseval's energy was estimated, and the Gaussian mixture model was applied for inertial feature extraction. The resulting vision and inertial features were concatenated and optimized, and a multi-layer perceptron classification model was trained. This model scored an accuracy of 0.88. The use of an optimization technique to select the most relevant features lowered the computational cost of the classification model. However, the cost may further be reduced by modeling the skeleton from raw vision data.

Xu et al. [34] proposed a fall detection algorithm that combined a threshold-based method (TBM) and CNN. For the sensor, they designed a wearable that used an accelerometer and a gyroscope. TBM used sensor information to suspect a fall of the wearer, and if it was suspicious, the accelerations were aggregated and encoded into an RGB image, which the CNN module took as its input to decide if a fall accident had happened. Training and testing on the SisFall [15] and the FARSEEING [35] datasets, the algorithm achieved 0.9746 accuracy. The CNN module had high complexity, leading to their subsequent research on a combined TBM and recurrent neural network–LSTM–method on the accelerometer only [36]. When the TBM module decided a suspicious activity had happened, three axial LSTM modules of the same structure processed respective dimensional accelerations for a final decision, scoring a 0.9907 accuracy.

Table 1 summarizes the works presented above.

TABLE 1. The summary of fall detection methods.

Category	Authors	Scores	Datasets	Advantages	Disadvantages
Wearable	Yu *et al.* [13]	0.9829-0.9983 sensitivity, 0.9849-0.9917 specificity	KFall, SisFall	Quantization of the deep-learning model to reduce model complexity	Constant battery drainage
Wearable	Hashim *et al.* [17]	0.9305-1.0 accuracy	Created dataset of 13 sequences	Wake and sleep of the sensors to minimize battery usage	A single sensor failure rendering system non-functional
Vision	Lee *et al.* [21]	0.955 F1-score	Created dataset of 1860 sequences	DVS for High privacy	DVS not financially viable
Vision	Chang *et al.* [25]	0.981 accuracy	Created dataset of 800 sequences	Using a different backbone for pose estimation model for lower complexity	Occlusion of the head not considered
Ambient	Chen *et al.* [29]	0.93 accuracy	Created dataset of 160 sequences	Low computational cost from low-resolution infrared sensors	Low detection score compared to other ambient sensor systems
Ambient	Tanaka *et al.* [30]	0.96 sensitivity, 0.29 false alarms per day, 83.43 hours per false alarm	Created dataset of 9 virtual years	Realistic ADL motions and sequences	Unrealistic fall timings
Fusion	Abro *et al.* [32]	0.88 accuracy	URFD	Feature optimization for low model complexity	Redundant human modeling of silhouette and skeleton creation in series
Fusion	Xu *et al.* [34], [36]	0.9746-0.9907 accuracy	SisFall, FARSEEING	Lower complexity of LSTM over CNN	Losing gyroscope information from previous work

B. THE KALMAN FILTER

To improve the detection scores from sequential data, developers also use filtering algorithms. One such algorithm is the Kalman filter, which combines noisy observations about a system to estimate the actual state of the system by recursion [37]. There are two main steps: prediction and update. For the prediction, the current state is projected forward based on the filter's process model to form the prior estimate. For the update, measurements with noise are combined with the prior and minimize error in the process and the measurement to form the posterior estimate [38].

The basic Kalman filter can effectively model linear systems, such as a projectile motion in a friction-free environment, as the underlying mathematics is linear algebra. However, the filter can diverge if the system is nonlinear, such as air drag on a projectile. To address this problem, mathematicians have developed a technique to linearize the system. The extended Kalman Filter computes the Jacobian matrix at the current estimate to approximate the system as linear [39]. If the nonlinearity is strong, sampling from the probability distribution can be done, which is used in the unscented Kalman filter and the particle filter [40]. Also, if the selected filter model is weak in targeting the system due to constant changes in the conditions, algorithms that can adapt to these conditions can be applied [41], [42], [43], [44].

For practical use in pose estimation, Buizza et al. [45] used filters to model the motion of the landmarks. They applied filters and conducted a comparison study with and without a filter; zero-, first-, and second-order, and velocity-as-input Kalman filters; and the particle filter. Among these models, the second-order acceleration and the velocity-as-input models performed best. Although they declared their approach real-time, they obtained the noise information using the ground truth from a single dataset–PoseTrack 2018 [46]–which may not be well-adaptive to real-world video or significantly different datasets.

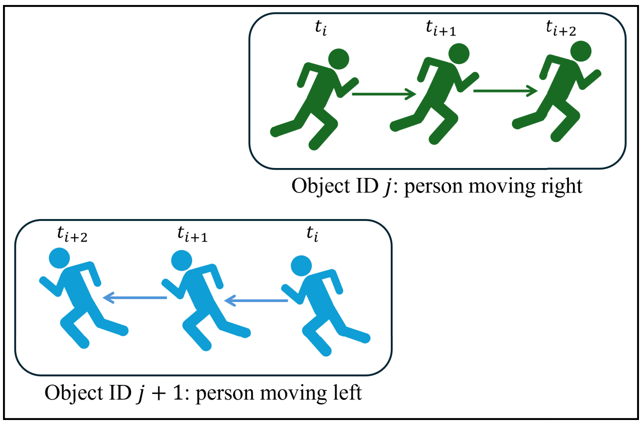

FIGURE 1. Schematic process of object tracking.

C. OBJECT TRACKING

To recognize the actions of multiple individuals, the necessary features must be distinguished from one individual to another. Object tracking is a computer vision technique to track objects with their movements in a video sequence to differentiate them from one another [47]. Fig. 1 shows the schematics of this process. For increasing productivity in aquaculture, Shreesha et al. [48] tracked the movement of the fish to analyze the behavior and visualize the stress in the fish. To effectively track the fish, they computed the temporal and spatial scores for tracking the motion as well as the similarity score for the fish in similar appearances. Chang et al. [25] also used a tracker, namely the centroid tracker, to distinguish the individuals in the input frames from one another [28].

As different as the cost or score computation algorithms may be, it is the data association algorithm that assigns each tracked object with an ID. This algorithm can be divided into two categories: global and greedy. The global assignment algorithms find the IDs that minimize the cost or maximize the score the most from all available combinations of assignments [49]. As the best group with the lowest cost must be found, the time complexity is generally high, approaching $O(n^2 \log n)$ [50]. The greedy algorithms find the minimum-cost match for the first object, remove the match from the pool, find the best match with the leftover objects, and repeat the process until a solution is found [51]. Many greedy assignment algorithms have different time complexities and solution ranges, but all are generally faster than the global assignment algorithms [52]. A threshold for the cost comparison must be applied in case of unbalanced assignment when the object with the highest costs is first selected, which results in wrongful assignment [51].

III. METHODOLOGY

Our algorithm, SMFDM, can best be presented in the following order: single-frame (SF) pose estimation, the Kalman filter, object tracking, SF fall detection, multi-frame (MF) fall detection, and dataset and final detection implementation. Fig. 2 shows the overall process of SMFDM.

A. SINGLE-FRAME POSE ESTIMATION

The OpenPose variant 3D Lightweight OpenPose [53] produces landmarks of multiple people in an image frame. This model works by inferencing 2D landmarks from the 2D image frame and using the 2D landmarks to generate 3D landmarks. More precisely, the 2D landmarks are first represented with heatmaps from 2D image encoding. A single heatmap may contain a single corresponding landmark region or multiple regions, depending on the number of people in the image frame [54]. Secondly, the 2D encoding, the heatmaps, and the part affinity fields (PAFs) are processed with a 3D encoding layer to produce 3D landmark maps. A part affinity field is a vector field that contains information about the connection for each joint [55]. Finally, the 2D heatmaps, PAFs, and 3D landmark maps are parsed and grouped into 2D and 3D landmarks per person, with 19 landmarks in each set.

B. THE KALMAN FILTER APPLICATION

To increase the performance in correctness of landmark detection–and thereby increase the correctness in fall detection–the Kalman filter is applied. We model the movement of people–landmarks–in 2D and 3D space as our state variables. The transformation from 2D space to 3D space, as well as the reverse process, is a nonlinear transformation. Modeling a nonlinear system requires a nonlinear method, which is computationally more expensive than linear methods [56]. Therefore, we model 2D and 3D landmarks as separate variables and use the linear method on each variable set.

The state $\vec{x}_t$–without a superscript–for 2D and 3D landmarks and their respective velocities–with superscripts–is represented as follows:

$$\vec{x}_t = \begin{bmatrix} \vec{x}_t^{(2n)} \\ \dot{\vec{x}}_t^{(2n)} \\ \vec{x}_t^{(3n)} \\ \dot{\vec{x}}_t^{(3n)}, \end{bmatrix} \tag{1}$$

where t denotes time step, $\vec{x}^{(\text{dim})}$ denotes positional vector of corresponding dimension, $\dot{\vec{x}}^{(\text{dim})}$ denotes velocity vector. The superscripts indicate whether the base is a 2D vector or a 3D vector. If there are n number of landmarks, the vectors with superscripts $(2n)$ and $(3n)$ belong to the spaces $\mathbb{R}^{2n}$ and $\mathbb{R}^{3n}$, respectively; therefore, for the stacked vector, $\vec{x}_t \in \mathbb{R}^{10n}$. Note that the variables x and $\dot{x}$ in this subsection denote Kalman filter's position and velocity state variables as standard notation. They are not to be confused with the directional variables x, y, and z in Section III-D.

We also keep track of the uncertainty of the system with the state covariance matrix $\mathbf{P}_t \in \mathbb{R}^{10n\times10n}$, which quantifies the uncertainty of the individual state variables as variances–diagonal terms–or in between the variables as covariances–off-diagnoal terms [57].

The state and the state covariance matrix undergo prediction and update for prior and posterior estimates, respectively. To model the prediction, we introduce the second equation of motion from physics [58],

$$p_t = p_{t-1} + v_{t-1}\Delta t + \frac{1}{2}a_{t-1}(\Delta t)^2, \tag{2}$$

where p, v, a, and Δt are position, velocity, acceleration, and small change in time, respectively. We assume constant acceleration during each time step and use up to the first order terms–positions and velocities–in the transition matrix $\mathbf{F}_t \in \mathbb{R}^{10n\times10n}$:

$$\mathbf{F}_t = \begin{bmatrix} \mathbf{F}_t^{(2n)} & \mathbf{0} \\ \mathbf{0} & \mathbf{F}_t^{(3n)}, \end{bmatrix} \tag{3}$$

with

$$\mathbf{F}_t^{(2n)} = \begin{bmatrix} \mathbf{I}^{(2n)} & \mathbf{T}^{(2n)} \\ \mathbf{0} & \mathbf{I}^{(2n)} \end{bmatrix}$$
$$\mathbf{F}_t^{(3n)} = \begin{bmatrix} \mathbf{I}^{(3n)} & \mathbf{T}^{(3n)} \\ \mathbf{0} & \mathbf{I}^{(3n)}, \end{bmatrix} \tag{4}$$

where $\mathbf{I}^{(\text{dim})}$ is the identity matrix of dimension corresponding to the superscript and $\mathbf{T}^{(\text{dim})} = \mathbf{I}^{(\text{dim})}\Delta t$. We model the leftover second order term–acceleration–as the process noise $w_t \sim \mathcal{N}(0, \mathbf{Q}_t)$ with the process covariance matrix $\mathbf{Q}_t \in \mathbb{R}^{10n\times10n}$. Assuming independence across dimensions,

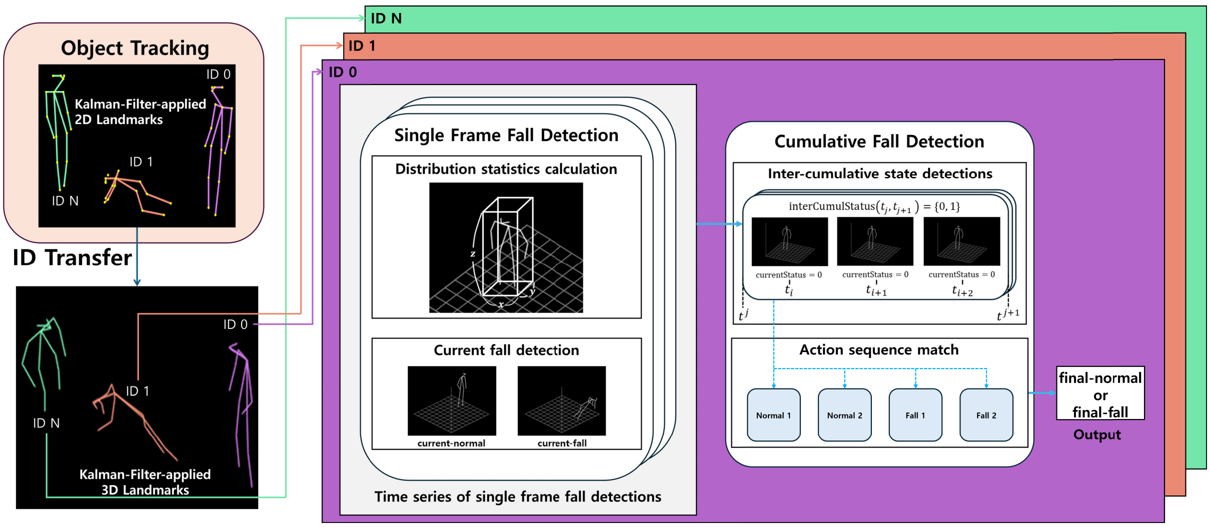

FIGURE 2. The overall schematics of the SMFDM process.

we write the unit process covariance matrix along a single dimension as follows–with the scaling, continuous noise term Q_c [57]:

$$\mathbf{Q}_t^{(\text{unit})} = \begin{bmatrix} \frac{1}{4}(\Delta t)^4 & \frac{1}{2}(\Delta t)^3 \\ \frac{1}{2}(\Delta t)^3 & (\Delta t)^2 \end{bmatrix} Q_c. \tag{5}$$

The components of $\mathbf{Q}_t^{(\text{unit})}$ are appropriately placed in the complete $\mathbf{Q}_t$ for all dimensions and landmarks. Then the predicted prior state and state covariance matrix can be written as follows:

$$\begin{aligned} \vec{x}_t^- &= \mathbf{F}_t \vec{x}_{t-1}^+ + \vec{w}_t \\ \mathbf{P}_t^- &= \mathbf{F}_t \mathbf{P}_{t-1}^+ \mathbf{F}_t^{\mathrm{T}} + \mathbf{Q}_t, \end{aligned} \tag{6}$$

where $-$ and $+$ on the superscript denote prior and posterior, respectively.

Now, we model the update step. For a valid measurement set $\vec{z}_t$, the measurement covariance matrix $\mathbf{R}_t$ and the measurement matrix $\mathbf{H}_t$ are constructed and used to compute the Kalman gain $\mathbf{K}_t$, which incorporates the uncertain state prediction and measurement together to form a good estimate [39]. Compared to the work by Buizza et al. [45], which used ground truth information for uncertainty, we use the network's confidence in the inference to model the measurement covariance matrix [53], effectively modeling inputs from different data sources at different time steps accordingly. With the Kalman gain, the posterior state and state covariance matrix can be expressed as follows:

$$\begin{aligned} \mathbf{K}_t &= \mathbf{P}_t^- \mathbf{H}_t^{\mathrm{T}} \left(\mathbf{H}_t \mathbf{P}_t^- \mathbf{H}_t^{\mathrm{T}} + \mathbf{R}_t \right)^{-1} \\ \vec{x}_t^+ &= \vec{x}_t^- + \mathbf{K}_t \left(\vec{z}_t - \mathbf{H}_t \vec{x}_t^- \right) \\ \mathbf{P}_t^+ &= \left(\mathbf{I}^{(5n)} - \mathbf{K}_t \mathbf{H}_t \right) \mathbf{P}_t^- \left(\mathbf{I}^{(5n)} - \mathbf{K}_t \mathbf{H}_t \right)^{\mathrm{T}} \\ &\quad + \mathbf{K}_t \mathbf{R}_t \mathbf{K}_t^{\mathrm{T}}. \end{aligned} \tag{7}$$

The 3D Lightweight OpenPose detects only the positional variables [53], meaning that they are the only observed variables and the velocity variables are hidden. This restricts the dimension of the measurement to $5n$. Additionally, the heatmap confidence is assimilated into the measurement covariance matrix $\mathbf{R}_t$ for optimal results. If the confidence is below a certain threshold, the corresponding measurement is ignored. This further reduces the dimension of $\vec{z}_t$ to be $\vec{z}_t \in \mathbb{R}^{5m}$ for $m <= n$. Subsequently, $\mathbf{R}_t \in \mathbb{R}^{5m \times 5m}$, and $\mathbf{H}_t \in \mathbb{R}^{5m \times 10n}$. State variables with missing measurements will undergo only the prediction step for a threshold of timesteps if previously measured.

Now, experimenting with the scaling terms and the initial covariance will complete the formulation of the Kalman filter application on the final output; however, to combine with 2D feature maps, we break up the 2D and 3D components. The Kalman filter prediction and update are applied to 2D landmarks, and the 2D heatmaps and the PAFs are evolved with the difference between the measurement and the posterior estimate as follows:

$$\begin{aligned} \vec{x}_{t,+\mathrm{z}} &= \vec{x}_{t,\mathrm{obs}}^+ - \vec{z}_t \\ \mathcal{H}_t^+ &= \mathcal{H}_t + \mathcal{X}_{t,+\mathrm{z}}^{(\mathcal{H})} \\ \mathcal{P}_t^+ &= \mathcal{P}_t + \mathcal{X}_{t,+\mathrm{z}}^{(\mathcal{P})}, \end{aligned} \tag{8}$$

where +z on the subscript means displacement from measurement to posterior; $\vec{x}_{t,\mathrm{obs}}^+$ is posterior observed variables; $\mathcal{H}$ and $\mathcal{P}$ are original 2D feature maps (heatmaps and PAFs); $\mathcal{H}^+$ and $\mathcal{P}^+$ are evolved feature maps; and $\mathcal{X}_{t,+\mathrm{z}}$ is $\vec{x}_{t,+\mathrm{z}}$ mapped onto the heatmaps or PAFs space according to the superscript. Note that prediction values are used for missing $\vec{z}_t$ components.

The resulting feature maps are entered in place of the original measurements, along with the 2D encoding, to produce 3D landmark maps. Then the 3D landmarks parsed from the landmark maps are filtered for enhanced estimates. Fig. 3 shows the inference process with data assimilation.

C. MULTI-FEATURE OBJECT TRACKING

The 3D Lightweight OpenPose provides an additional feature of greedy tracking with similar landmarks between the previous and next ones over time. This similarity is treated as intersection over union (IoU) as follows [53]:

$$\mathrm{IoU} = \frac{\text{number of overlaps}}{\text{union of overlaps and non-overlaps}}. \tag{9}$$

As the 3D Lightweight OpenPose makes single-shot inferences on multiple people, tracking both 2D and 3D landmarks fully and separately is redundant [54]. This is reflected in the provided greedy tracking algorithm and, thus, remains unchanged [53].

D. SINGLE-FRAME FALL DETECTION

After each of the landmark sets is numbered with object ID j, SF fall detections are made. This is done by comparing the distributions among the landmark components in 3D axial directions. If the distribution in one direction is different from those in other directions, the posture of the labeled set can be determined. For the distribution statistic, we use standard deviation, which measures the extent of scattering in a set of values [59]. This is shown in the following equation:

$$\sigma_{p,j} = \sqrt{\frac{\sum_k^n \left(p_{k,j} - \mu_{p,j}\right)^2}{N}}, \tag{10}$$

where σ is the standard deviation; p is the position in the direction of width x, depth y, or height z; μ_p is the average position in p direction; and N is 19 as the total number of landmarks [53]. j denotes j-th landmark set from the image frame. If the value of the standard deviation is large, then the coordinates are far apart. If it is small, then the coordinates are close together. Since z is the coordinate in height, if the standard deviation in height is less than that of either of the horizontal directions x or y, then the person is considered currently fallen. This current status curSt_j is expressed by the following equation:

$$\mathrm{curSt}_j = \mathrm{int}\left(\sigma_{z,j} < \sigma_{x,j} \text{ or } \sigma_{z,j} < \sigma_{y,j}\right) \tag{11}$$

The value of curSt_j is either 0 for the current state of normal (c-normal) or 1 for the current state of fall (c-fall). We also describe the variables that are used throughout this work, i.e., normal N and fall F:

$$N = 0, \quad F = 1 \tag{12}$$

Using (12), we can represent the possible values of curSt_j in the following manner:

$$\mathrm{curSt}_j \in \{N, F\}. \tag{13}$$

E. MULTI-FRAME FALL DETECTION

Lastly, a series comprising curSt_js of consecutive temporal sets of landmarks is processed. This ensures that the actual status does not change, meaning there should be no false positives in single images. For example, a person may temporarily get close to the ground to pick up something, which may be marked as an actual fall. We adopt the two-stage check from our previous work: intermediate cumulative detection and final detection [9]. In the intermediate cumulative detection, if most of curSt_js are equal to 0, then the person is in the intermediate cumulative state of normal (ic-normal). If most of them are equal to 1, then the person is in the intermediate cumulative state of fall (ic-fall). The following equation represents this:

$$\mathrm{icSt}_j\left(t_s, t_f\right) = \mathrm{int}\left(\frac{\sum_{t=t_i}^{t_f} \mathrm{curSt}_j(t)}{\#_{t=t_i}^{t_f} t} > 0.5\right) \in \{N, F\} \tag{14}$$

where t is a variable representing time, s and f denote the temporal sequence's start and end, int converts a type boolean argument to type int, and # denotes the count of occurrences. This is the average of all curSt_js in the time interval $\left[t_s, t_f\right]$ compared with the threshold of 0.5. This follows the current status curSt_j convention.

A schematic process of intermediate cumulative detection is shown in Fig. 4.

Although the intermediate cumulative state is used to prevent the false detection of fall status on a single frame, a falling sequence from upright to fallen must be recognized. Therefore, to produce the final status finSt_j from the final detection finDet_j, different sequences of icSt_j are arranged as before [9]:

$$\mathrm{finSt}_j = \mathrm{finDet}_j\left(\left(\mathrm{icSt}_j\left(\left(t_s, t_f\right)_k\right)\right)_{k \in \mathbb{W}}\right) \in \{N, F\}, \tag{15}$$

where $\left(t_s, t_f\right)_k = \left(t^{2k}, t^{2k+1}\right)$. The sequence rule indicates that the time intervals–the sequence of icSt_js–are in an immediate, non-overlapping consecutive order, shown for ic-normal to ic-fall in Fig. 5. Note the different usages of subscripts and superscripts on the base t; the subscripts will denote the frame numbers equal to the numbers in the respective subscripts–as in every tick–and the superscripts will denote the ordering of selected frames–as in the order of the intervals. As before, the possible values of final detection are 0 for final-normal and 1 for final-fall.

F. DATASET AND FINAL DETECTION IMPLEMENTATION

There are many datasets for studying fall accidents as well as ADLs. URFD [33], Fall Detection Dataset (FDD) [60], and Multicam Fall Dataset [61] are popular datasets that provide videos for vision-based systems. Whereas these provide valuable resources, our previous work primarily focused on achieving low model complexity. To treat the problem simply

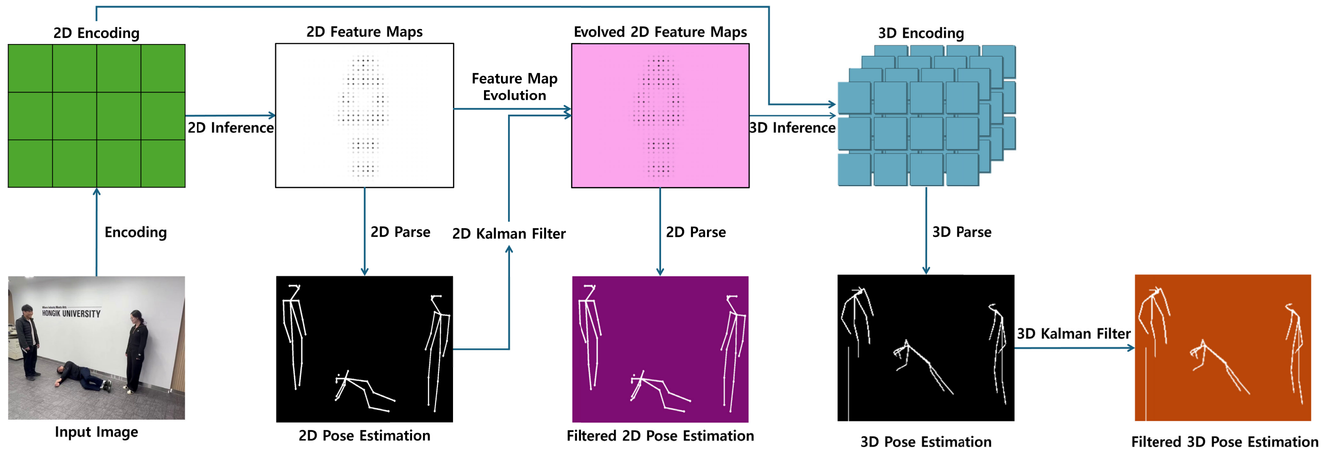

FIGURE 3. The pose estimation process with the Kalman filter.

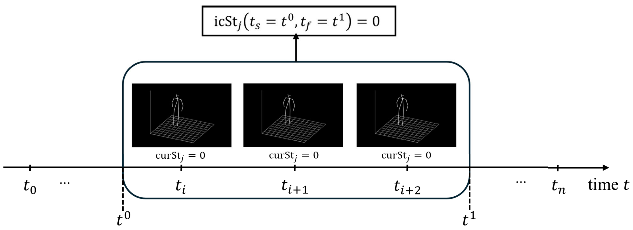

FIGURE 4. Intermediate cumulative state detection example–ic-normal.

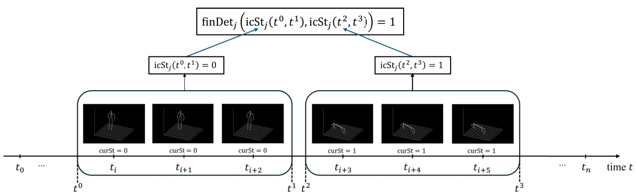

FIGURE 5. Final state detection example–final-fall.

as our model would require, we had created our own dataset of ADLs and falls. From Fig. 6, Normal 1 represents a standing action (ADL), Fall 1 is falling from a standing position, Normal 2 is recovery from a fall, and Fall 2 represents the failure to recover. As this study enhances the algorithm and focuses on the individual landmarks' confidence, we adopt our previous dataset scheme.

For the actual implementation of finDet_j in (15), icSt_js in (14) are calculated every frame with an averaging window of 5 seconds, and a run-length encoding (RLE) algorithm is used to group icSt_js. An RLE algorithm compresses a sequence by counting the number of consecutive elements in the sequence and replacing the occurrences of those elements with the count and a single element representation for all elements [62]. To read the order of consecutive icSt_js, the count of the occurrences is unnecessary and thus dropped. This is done using the Python library Pandas [63], [64]. An example usage of RLE on a sequence $\mathrm{seq}_{\mathrm{icSt}_j}$ to encode it as $\mathrm{encSeq}_{\mathrm{icSt}_j}$ and generate the order $\mathrm{ordSeq}_{\mathrm{icSt}_j}$ is shown below:

$$\begin{aligned}
\mathrm{seq}_{\mathrm{icSt}_j} &= (N, N, N, F, N, N, F, F, F) \\
\mathrm{encSeq}_{\mathrm{icSt}_j} &= \mathrm{RLE}\left(\mathrm{seq}_{\mathrm{icSt}_j}\right) \\
&= (3N, 1F, 2N, 3F) \\
\mathrm{ordSeq}_{\mathrm{icSt}_j} &= \mathrm{ord}\left(\mathrm{encSeq}_{\mathrm{icSt}_j}\right) \\
&= (N, F, N, F), \qquad (16)
\end{aligned}$$

where RLE and ord represent functions that process RLE and the ordering from given, respective sequences.

This final detection is done for every tracked object with ID j in the webcam footage. A three-person example of this multi-person (MP) detection is shown in Fig. 7.

G. ALGORITHM

To summarize, our method applies the Kalman filter method on pose estimation to obtain landmarks with reduced noise. The landmarks are used to decide if the person has fallen with aggregation and run-length encoding. With g_{2D} (image) and $g_{3D}\left(\mathcal{H}_t^+, \mathcal{P}_t^+\right)$ as 2D and 3D landmark generators, the algorithm is outlined in Algorithm 1.

IV. EXPERIMENT

A. THE SETUP

For the experiment, we have prepared three case studies: all standing, all fallen, and mixed. Ten trials are conducted with the following procedure:

1) Set up the computer and webcam system.
2) Three human subjects enter the webcam view. All limbs must remain inside the view.
3) Start the SMFDM program with a window of 30 seconds.
4) The subjects perform the action sequences in Fig. 6 that correspond to the case study.
5) Record the detection result.

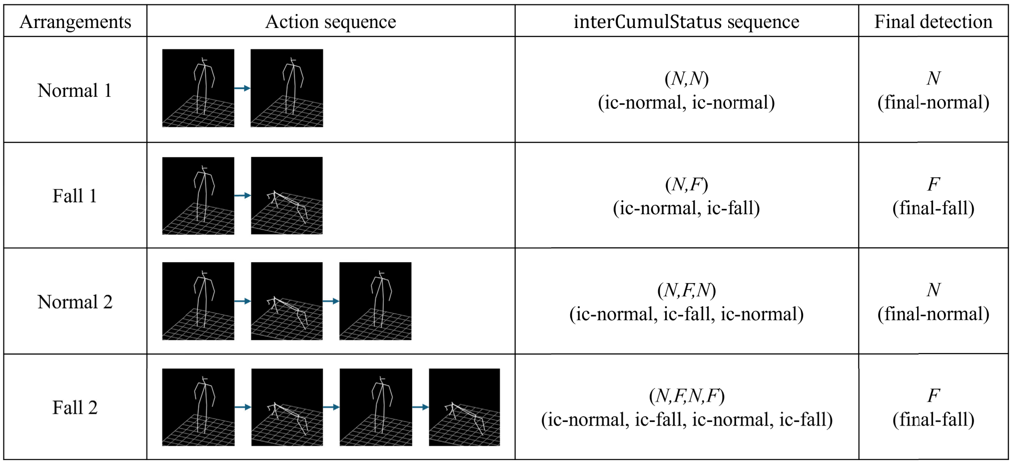

FIGURE 6. Final detections from arrangements of icSt_js.

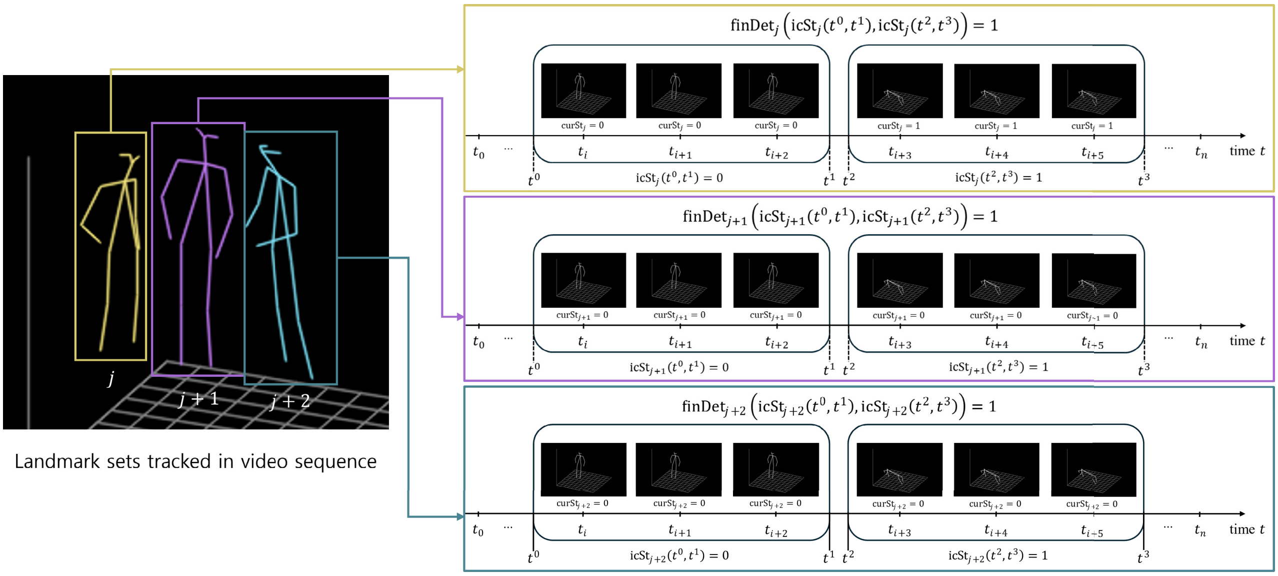

FIGURE 7. Schematics of multi-person fall detection showing the Fall 1 sequence (one of four action sequences).

TABLE 2. The input data of the case studies. $N1$, $N2$, $F1$ and $F2$ indicate Normals 1 and 2 and Falls 1 and 2 in order.

Case Studies	Person 0				Person 1				Person 2			
	$N1$	$F1$	$N2$	$F2$	$N1$	$F1$	$N2$	$F2$	$N1$	$F1$	$N2$	$F2$
1	10	0	0	0	5	0	5	0	3	0	7	0
2	0	10	0	0	0	5	0	5	0	7	0	3
3	5	5	0	0	3	4	2	1	3	1	4	2

6) Repeat Steps 4 and 5 ten times.
7) Repeat Steps 4-6 for different case studies.

The case studies were conducted on a system with the i7-13700HK, RTX 4070, 32 GB of RAM, and a Full HD webcam in a light environment. Case Study 1 represents the scenario of ADLs with no fall accidents. Therefore, all subjects perform Normal 1 from start to end. Case Study 2 represents the scenario of all humans falling down. All subjects perform either the action of Fall 1 or 2. Finally, Case Study 3 is a mix of Case Studies 1 and 2; some subjects perform Normal 1 or 2, and the rest perform Fall 1 or 2.

The experimentation setup is recorded in Table 2.

B. SCORE METRICS

We use accuracy and F1-score as the classification metrics and the micro-averaging method for multi-class classification [65]. We start from binary to multi-class classification and list four cases of a binary model's detection.

Algorithm 1 SMFDM With Kalman Filter

Initialize: template $\mathbf{F}_{t'}^{(2D)}$, $\mathbf{Q}_{t'}^{(3D)}$,$\mathbf{F}_{t'}^{(2D)}$, $\mathbf{Q}_{t'}^{(3D)}$, $\vec{x}_{t'}^{(2D),+}$, $\vec{x}_{t'}^{(3D),+}$, $\mathbf{P}_{t'}^{(2D),+}$, $\mathbf{P}_{t'}^{(3D),+}$; t' with initial time value; durations T_{icSt},T_{RLE}

while true **do**
  $t' \leftarrow t$
  Store current time into t
  2D Kalman Filter ▷ (2D) superscripts omitted
    $\{\vec{z'}_t\}_j, \{\vec{c}_t\}_j, \mathcal{H}_t, \mathcal{P}_t \leftarrow g_{2D}\left(\text{image}_t\right)$ ▷ $\vec{c}_t$: conf.
    $\{\vec{z}_t\}_j \leftarrow$ Run object tracking on $\{\vec{z'}_t\}_j$
    for j in length($\{\vec{z}_t\}_j$) **do** ▷ index j omitted
      $\mathbf{F}_t,\mathbf{Q}_t \leftarrow$ Update $\mathbf{F}_{t'},\mathbf{Q}_{t'}$ with t
      $\vec{x}_t^- \leftarrow \mathbf{F}_t\vec{x}_{t'}^+$
      $\mathbf{P}_t^- \leftarrow \mathbf{F}_t\mathbf{P}_{t'}^+\mathbf{F}_t^{\mathrm{T}} + \mathbf{Q}_t$
      Compute $\mathbf{H}_t$, $\mathbf{R}_t$ with $\vec{z}_t, \vec{c}_t$
      $\mathbf{K}_t \leftarrow \mathbf{P}_t^-\mathbf{H}_t^{\mathrm{T}}\left(\mathbf{H}_t\mathbf{P}_t^-\mathbf{H}_t^{\mathrm{T}} + \mathbf{R}_t\right)^{-1}$
      $\vec{x}_t^+ \leftarrow \vec{x}_t^- + \mathbf{K}_t\left(\vec{z}_t - \mathbf{H}_t\vec{x}_t^-\right)$
      $\mathbf{P}_t^+ \leftarrow (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\,\mathbf{P}_t^-\,(\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)^{\mathrm{T}} + \mathbf{K}_t\mathbf{R}_t\mathbf{K}_t^{\mathrm{T}}$
    end for
  end 2D Kalman Filter
  Index observed variables $\vec{x}_{t,\text{obs}}^+$
  $\vec{x}_{t,+z} \leftarrow \vec{x}_{t,\text{obs}}^+ - \vec{z}_t$
  $\mathcal{H}_t^+, \mathcal{P}_t^+ \leftarrow$ Evolve $\mathcal{H}_t, \mathcal{P}_t$ with mapped $\vec{x}_{t,+z}$
  3D Kalman Filter ▷ (3D) superscripts omitted
    $\{\vec{z}_t\}_j, \{\vec{c}_t\}_j, \leftarrow g_{3D}\left(\mathcal{H}_t^+, \mathcal{P}_t^+\right)$
    for j in length($\{\vec{z}_t\}_j$) **do**
      Similar to **2D Kalman Filter**
    end for
  end 3D Kalman Filter
  $\{\vec{x}\}_j, \{\vec{y}\}_j, \{\vec{z}\}_j \leftarrow$ Assign components of $\{\vec{x}_t^{(3D,+)}\}_j$
  for index j in $\{\vec{x}\}_j$ **do**
    for $\vec{p}_j = \vec{x}_j, \vec{y}_j, \vec{z}_j$ **do**
      $\sigma_{p,j} \leftarrow \sqrt{\frac{\sum_k^n (p_{k,j} - \mu_{p,j})^2}{N}}$
    end for
    $\text{curSt}_j \leftarrow \text{int}\left(\sigma_{z,j} < \sigma_{x,j} \text{ or } \sigma_{z,j} < \sigma_{y,j}\right)$
    $\text{icSt}_j(t) \leftarrow \text{int}\left(\frac{\sum_{\tau=t-T_{\text{icSt}}}^{t} \text{curSt}_j(\tau)}{\#_{\tau=t-T_{\text{icSt}}}^{t}\tau} > 0.5\right)$
    $\text{seq}_{\text{icSt}_j}(t) \leftarrow$ Sequence of $\{\text{icSt}_j\}_{\tau=t-T_{\text{RLE}}}^{t}$
    $\text{encSeq}_{\text{icSt}_j}(t) \leftarrow \text{RLE}\left(\text{seq}_{\text{icSt}_j}\right)$
    $\text{ordSeq}_{\text{icSt}_j}(t) \leftarrow \text{ord}\left(\text{encSeq}_{\text{icSt}_j}\right)$
  end for
end while

1) TP (True Positives): The number of positive instances (Fall) correctly detected as positives.
2) TN (True Negatives): The number of negative instances (Normal) correctly detected as negatives.
3) FP (False Positives): The number of negative instances (Normal) incorrectly detected as positives.
4) FN (False Negatives): The number of positive instances (Fall) incorrectly detected as negatives.

TABLE 3. The results of the case studies in confusion matrix.

		True			
		$N1$	$F1$	$N2$	$F2$
Predicted	$N1$	29	0	0	0
	$F1$	0	32	0	0
	$N2$	1	1	16	0
	$F2$	0	2	1	8

Accuracy and F1-score can be explained using these terms. Accuracy is the proportion of detections that the classification model correctly detected. It is the most basic metric that indicates the model's overall correctness. The formula for accuracy is given below:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{17}$$

whereas it is simple to compute, accuracy can be misinterpreted in cases where one class outnumbers the others. If the model treats all inputs as fall and there are 95 falling instances and 5 normal instances, the accuracy will be 0.95, which is very high regardless of false detections. Therefore, it is often provided in conjunction with other metrics, such as precision, recall, and F1-score. Precision is the proportion of correct positive detections over all positive detections, and recall is the proportion of correct positive detections over all positive instances. The formulas for precision and recall are given below:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$
$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{18}$$

As precision is sensitive to false positives and recall is sensitive to false negatives, these metrics can be combined to give a balanced score metric of F1-score. F1-score is the harmonic mean of precision and recall, given below:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{19}$$

So far, the metrics have been explained using binary classification concepts. In multi-class, all classes are positive in their respective point of view. Therefore, a method to average the results is needed. Micro-averaging is a strategy to give a single metric score by counting all detections separately in four cases and calculating the overall metric score. The micro-averaged precision, recall, and F1-score are given by the following equations:

$$\text{Precision}_{\text{micro}} = \frac{\sum_c \text{TP}_c}{\sum_c (\text{TP}_c + \text{FP}_c)}$$
$$\text{Recall}_{\text{micro}} = \frac{\sum_c \text{TP}_c}{\sum_c (\text{TP}_c + \text{FN}_c)}$$
$$\text{F1-score}_{\text{micro}} = 2 \times \frac{\text{Precision}_{\text{micro}} \times \text{Recall}_{\text{micro}}}{\text{Precision}_{\text{micro}} + \text{Recall}_{\text{micro}}}. \tag{20}$$

The micro F1-score defined above is used for our model's multi-class detection of Normal 1, Fall 1, Normal 2, and Fall 2.

TABLE 4. Comparison among fall detection models.

Method	Accuracy	Remarks
Conv1D [67]	0.9983	• Determine landmarks that contribute to detection
PEFDM [25]	0.981	• Used different backbone for low model complexity • Does not handle head occlusions
SFDM [9]/**SMFDM**	0.925 / 0.933	• Plug and play: Final refinement can be applied to all other methods in this table for increase in performance • Interoperability: Lifting models can use our method for heatmap evolution • Increased accuracy by 0.008
ST-GCN [68]	0.924	• Substantial complexity of 16.32 B FLOPs [69]
BP-2RF [70]	0.8999	• No multi-person detection
Conv3D [71]	0.805	• Low detection score

We run our model and obtain the confusion matrix in Table 3. Using the above metrics, micro-averaged accuracy and F1-score are calculated to be 0.944.

V. DISCUSSIONS

On the multi-person detection data, our model scored 0.944 micro-averaged F1-score. For a direct comparison, we applied SMFDM on our previous dataset [66]. The performance in correctness has increased from the micro-averaged accuracy of 0.925 of the single-person detection algorithm [9] to 0.933. This improvement is most likely due to the Kalman filter, which would correct noisy measurements to give more consistent curSts, and thus more correct icSts and finSts over time.

As for the performance in speed, detections were made at 27.4 FPS, compared to 28.2 FPS from the single-person model [9]. Object tracking is applied on each set of landmarks, and the fall detection algorithm part performs detections on all subjects in the webcam view; therefore, this decrease is well expected but is too small to be safely ignored.

We also compare the accuracy of selected pose-estimation-driven machine- and deep-learning fall detection methods, as shown in Table 4.

Our fall detection method, with 0.933 accuracy, falls in the middle. Our previous single-person approach had scored 0.925 accuracy. By applying the Kalman filter methodology, the score increased from 0.925 to 0.933.

The above results indicate that fall detection methods can adopt the adaptive Kalman filter method to gain correctness performance. Specifically for pose estimation, the Kalman filter can readily be used to adaptively change the measurement covariance matrix on the final output to reduce noise as plug-and-play. Models that lift 2D landmarks to 3D landmarks can adopt our method of evolving the hidden layer heatmaps to refine intermediate data for high-confidence 3D landmarks.

Whereas our study demonstrates enhancement in correctness, it is important to consider other critical factors. The concern of data privacy grows, as does any other data technology. As mentioned in Section II, extracting key features from videos and respecting privacy is highly encouraged. It is also true that identifiable information may be temporarily cached or stored permanently for data collection. Therefore, an effort to anonymize vision data is necessary. Some notable works are face blurring [72], use of an IR camera to remove facial region on an RGB image [73], and a first-person perspective wearable camera [74].

As diverse information can be retrieved in a multi-sensor approach, researchers focus on developing fusion systems. However, the system must use efficient methods to connect the sensors and handle the input rate. As previously mentioned, Xu et al. conducted research series on developing a sensor-based fall detection system [34], [36]. The first research [34] used the fusion-sensor approach by combining gyroscope and accelerometer sensor data. In their follow-up research [36], however, they removed the gyroscope sensor as they focused on optimizing their model. Therefore, further research is necessary to develop systems that process multi-modal data efficiently.

The works mentioned so far fall in the category of fall detection, where falling accidents are spotted as they happen [75]. There are other categories, namely fall prevention and fall protection [76]. Fall prevention is a category of methods to prevent falls, such as exercising, cognitive training, and removing environmental hazards [77]. Fall protection prevents injury from fall accidents by installing handrail for the person to grab onto [78] and inflating a mobile airbag when the person is falling [79]. As technology progresses, designing edge devices that encompass all these categories will be highly advantageous.

Having mentioned the limitations and future directions, we address the remaining challenges and points to further enhance our proposed system. We have examined the enhancement of pose estimation and fall detection with the basic Kalman filter. However, given the nature of human movements as having both linear and nonlinear aspects, filters that handle nonlinear systems can be used instead of the basic linear filter. This will effectively estimate sudden movements of the limbs, and more importantly, the sudden falling actions in accidents.

Also, we have designed our adaptive functionality by applying confidence scores of pose estimation on the measurement covariance matrix for robustness in different input data sources for pose estimation. Whereas this works, other advanced adaptive algorithms can also be applied for the measurement and the process covariance matrix.

These algorithms can be used to increase our model's performance or compared against our confidence-based adaptive algorithm.

Finally, the current approach detects only the falling individuals. When the accident happens, there may be other people willing to help the fallen person recover or paramedics who provide medical care when the accident has been reported. These individuals will likely get close to the ground, which may result in another fall detection. For a precise accident report, our algorithm can be modified to use a different distribution statistic to allow the detection of the assisting people. Survey and analysis of datasets containing multi-class labels of injured, bystanders, and assisting individuals will be conducted. A request for nursing home data collection has been filed to the Institutional Review Board of Hongik University Industry-Academic Cooperation Foundation.

VI. CONCLUSION

Human fall detection is crucial for a healthy life, especially for elderly people. In this paper, we have proposed the simple multi-person fall detection model for fall accident detection in live streams. This was done by inferencing landmarks on people from the 3D Lightweight OpenPose, applying the Kalman filter with confidence scores from the pose estimation for increased correctness and adaptability in landmark detection, using object tracking to track and label multiple landmark sets, and calculating the state of fall or non-fall from a series of calculations. This model achieved an F1-score of 0.944 in the multi-person setup and, for the single-person setup, 0.933 which has minimal speed loss from our previous single-person fall detection model. In the future, we will evaluate the performance of this model in real nursing environments with identification of assisting individuals for precise reporting of the accident, implement the extended or unscented Kalman Filter for the robustness in nonlinear motion, conduct a comparison study on different adaptive algorithms, and expand the detection algorithm to process sound that is made when a falling accident occurs for multi-modality.

REFERENCES

[1] WHO. *GHE: Life Expectancy and Healthy Life Expectancy*. Accessed: Mar. 25, 2025. [Online]. Available: https://www.who.int/data/gho/data/themes/mortality-and-global-health-estimates/ghe-life-expectancy-and-healthy-life-expectancy

[2] J. M. Aburto, F. Villavicencio, U. Basellini, S. Kjærgaard, and J. W. Vaupel, "Dynamics of life expectancy and life span equality," *Proc. Nat. Acad. Sci. USA*, vol. 117, no. 10, pp. 5250–5259, Mar. 2020, doi: 10.1073/pnas.1915884117.

[3] WHO. *Global Health Estimates: Leading Causes of Death*. Accessed: Mar. 25, 2025. [Online]. Available: https://www.who.int/data/gho/data/themes/mortality-and-global-health-estimates/ghe-leading-causes-of-death

[4] S. Ek, D. Rizzuto, W. Xu, A. Calderón-Larrañaga, and A.-K. Welmer, "Predictors for functional decline after an injurious fall: A population-based cohort study," *Aging Clin. Experim. Res.*, vol. 33, no. 8, pp. 2183–2190, Aug. 2021, doi: 10.1007/s40520-020-01747-1.

[5] K.-P. Yeh, M.-H. Lin, L.-K. Liu, L.-Y. Chen, L.-N. Peng, and L.-K. Chen, "Functional decline and mortality in long-term care settings: Static and dynamic approach," *J. Clin. Gerontol. Geriatrics*, vol. 5, no. 1, pp. 13–17, Mar. 2014, doi: 10.1016/j.jcgg.2013.08.001.

[6] X. Wang, J. Ellul, and G. Azzopardi, "Elderly fall detection systems: A literature survey," *Frontiers Robot. AI*, vol. 7, pp. 1–23, Jun. 2020, doi: 10.3389/frobt.2020.00071.

[7] Z. Ma and G. Mei, "A hybrid attention-based deep learning approach for wind power prediction," *Appl. Energy*, vol. 323, Oct. 2022, Art. no. 119608, doi: 10.1016/j.apenergy.2022.119608.

[8] G. Menghani, "Efficient deep learning: A survey on making deep learning models smaller, faster, and better," *ACM Comput. Surv.*, vol. 55, no. 12, pp. 1–37, Mar. 2023, doi: 10.1145/3578938.

[9] J. Yang and R. Y. C. Kim, "Simple single-person fall detection model using 3D pose estimation mechanisms," *IEEE Access*, vol. 12, pp. 174640–174653, 2024, doi: 10.1109/ACCESS.2024.3496992.

[10] J. Yang. (2025). *Flops Correction*. [Online]. Available: https://gist.github.com/yjmd2222/e52f91dc14fbd3256a1718c313bd3375

[11] S. K. Gharghan and H. A. Hashim, "A comprehensive review of elderly fall detection using wireless communication and artificial intelligence techniques," *Measurement*, vol. 226, Feb. 2024, Art. no. 114186. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224124000708

[12] A. Ramachandran and A. Karuppiah, "A survey on recent advances in wearable fall detection systems," *BioMed Res. Int.*, vol. 2020, no. 1, Jan. 2020, Art. no. 2167160. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2020/2167160

[13] X. Yu, S. Park, D. Kim, E. Kim, J. Kim, W. Kim, Y. An, and S. Xiong, "A practical wearable fall detection system based on tiny convolutional neural networks," *Biomed. Signal Process. Control*, vol. 86, Sep. 2023, Art. no. 105325. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1746809423007589

[14] X. Yu, J. Jang, and S. Xiong, "A large-scale open motion dataset (KFall) and benchmark algorithms for detecting pre-impact fall of the elderly using wearable inertial sensors," *Frontiers Aging Neurosci.*, vol. 13, pp. 1–14, Jul. 2021. [Online]. Available: https://www.frontiersin.org/journals/aging-neuroscience/articles/10.3389/fnagi.2021.692865

[15] A. Sucerquia, J. López, and J. Vargas-Bonilla, "SisFall: A fall and movement dataset," *Sensors*, vol. 17, no. 1, p. 198, Jan. 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/1/198

[16] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2704–2713.

[17] H. Ali Hashim, S. L. Mohammed, and S. K. Gharghan, "Accurate fall detection for patients with Parkinson's disease based on a data event algorithm and wireless sensor nodes," *Measurement*, vol. 156, May 2020, Art. no. 107573. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S026322412030110X

[18] S. B. Khojasteh, J. R. Villar, E. de la Cal, V. M. González, J. Sedano, and H. R. Yazğan, "Evaluation of a wrist-based wearable fall detection method," in *Hybrid Artificial Intelligent Systems*. Cham, Switzerland: Springer, 2018, pp. 377–386.

[19] H. Fang, L. Wang, Z. Fu, L. Xu, W. Guo, J. Huang, Z. L. Wang, and H. Wu, "Anatomically designed triboelectric wristbands with adaptive accelerated learning for human–machine interfaces," *Adv. Sci.*, vol. 10, no. 6, Feb. 2023, Art. no. 2205960. [Online]. Available: https://advanced.onlinelibrary.wiley.com/doi/abs/10.1002/advs.202205960

[20] J. Chen, K. Kwong, D. Chang, J. Luk, and R. Bajcsy, "Wearable sensors for reliable fall detection," in *Proc. IEEE Eng. Med. Biol. 27th Annu. Conf.*, May 2005, pp. 3551–3554.

[21] H. Lee, J. Kim, D. Yang, and J.-H. Kim, "Embedded real-time fall detection using deep learning for elderly care," 2017, *arXiv:1711.11200*.

[22] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. Comput. Vis. (ECCV)*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham, Switzerland: Springer, Jan. 2016, pp. 20–36.

[23] IniVation. *Dvxplorer Lite—Academic Rate*. Accessed: May 19, 2025. [Online]. Available: https://shop.inivation.com/collections/dvxplorer-lite-1/products/dvxplorer-lite-academic-rate

[24] C. Zheng, W. Wu, C. Chen, T. Yang, S. Zhu, J. Shen, N. Kehtarnavaz, and M. Shah, "Deep learning-based human pose estimation: A survey," 2020, *arXiv:2012.13392*.

[25] W.-J. Chang, C.-H. Hsu, and L.-B. Chen, "A pose estimation-based fall detection methodology using artificial intelligence edge computing," *IEEE Access*, vol. 9, pp. 129965–129976, 2021.
[26] D. Osokin. *Real-time 2D Multi-person Pose Estimation on CPU: Lightweight Openpose*. [Online]. Available: https://github.com/Daniil-Osokin/lightweight-human-pose-estimation.pytorch
[27] G. Chevalier. *Lstms for Human Activity Recognition*. [Online]. Available: https://github.com/guillaume-chevalier/LSTM-Human-Activity-Recognition
[28] J. C. Nascimento, A. J. Abrantes, and J. S. Marques, "An algorithm for centroid-based tracking of moving objects," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Jul. 1999, pp. 3305–3308, doi: 10.1109/ICASSP.1999.757548.
[29] W.-H. Chen and H.-P. Ma, "A fall detection system based on infrared array sensors with tracking capability for the elderly at home," in *Proc. 17th Int. Conf. E-Health Netw., Appl. Services (HealthCom)*, Oct. 2015, pp. 428–434, doi: 10.1109/HealthCom.2015.7454538.
[30] K. Tanaka, M. Kudo, and K. Kimura, "Fall detection by ambient sensors on years-long simulation data," in *Proc. Int. Conf. Activity Behav. Comput. (ABC)*, vol. 53, May 2024, pp. 1–8, doi: 10.1109/abc61795.2024.10651854.
[31] D. J. Cook, "How smart is your home?" *Science*, vol. 335, no. 6076, pp. 1579–1581, Mar. 2012, doi: 10.1126/science.1217640.
[32] I. A. Abro and A. Jalal, "Multi-modal sensors fusion for fall detection and action recognition in indoor environment," in *Proc. 3rd Int. Conf. Emerg. Trends Electr., Control, Telecommun. Eng. (ETECTE)*, Nov. 2024, pp. 1–6, doi: 10.1109/etecte63967.2024.10823705.
[33] B. Kwolek and M. Kepski, "Human fall detection on embedded platform using depth maps and wireless accelerometer," *Comput. Methods Programs Biomed.*, vol. 117, no. 3, pp. 489–501, Dec. 2014.
[34] T. Xu, H. Se, and J. Liu, "A fusion fall detection algorithm combining threshold-based method and convolutional neural network," *Microprocessors Microsyst.*, vol. 82, Apr. 2021, Art. no. 103828. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141933121000089
[35] J. Klenk, L. Schwickert, L. Palmerini, S. Mellone, A. Bourke, E. A. F. Ihlen, N. Kerse, K. Hauer, M. Pijnappels, M. Synofzik, K. Srulijes, W. Maetzler, J. L. Helbostad, W. Zijlstra, K. Aminian, C. Todd, L. Chiari, and C. Becker, "The FARSEEING real-world fall repository: A large-scale collaborative database to collect and share sensor signals from real-world falls," *Eur. Rev. Aging Phys. Activity*, vol. 13, no. 1, pp. 1–7, Dec. 2016.
[36] T. Xu and J. Liu, "A low-power fall detection method based on optimized TBM and RNN," *Digit. Signal Process.*, vol. 126, Jun. 2022, Art. no. 103525. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1051200422001427
[37] J. Mochnac, S. Marchevsky, and P. Kocan, "Bayesian filtering techniques: Kalman and extended Kalman filter basics," in *Proc. 19th Int. Conf. Radioelektronika*, Apr. 2009, pp. 119–122, doi: 10.1109/RADIOELEK.2009.5158765.
[38] G. Welch and G. Bishop, "An introduction to the Kalman filter," in *Proc. SIGGRAPH, Course*, 2001, vol. 8, no. 23175, p. 41.
[39] M. I. Ribeiro, "Kalman and extended Kalman filters: Concept, derivation and properties," *Inst. Syst. Robot.*, vol. 43, no. 46, pp. 3736–3741, 2004.
[40] K. György, A. Kelemen, and L. Dávid, "Unscented Kalman filters and particle filter methods for nonlinear state estimation," *Proc. Technol.*, vol. 12, pp. 65–74, Jun. 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212017313006427
[41] S. D. Brown and S. C. Rutan, "Adaptive Kalman filtering," *J. Res. Nat. Bur. Standards*, vol. 90, no. 6, p. 403, Nov. 1985, doi: 10.6028/jres.090.032.
[42] J. Li, X. Xu, Z. Jiang, and B. Jiang, "Adaptive Kalman filter for real-time visual object tracking based on autocovariance least square estimation," *Appl. Sci.*, vol. 14, no. 3, p. 1045, Jan. 2024. [Online]. Available: https://www.mdpi.com/2076-3417/14/3/1045
[43] M. Sun, M. E. Davies, I. K. Proudler, and J. R. Hopgood, "Adaptive kernel Kalman filter," *IEEE Trans. Signal Process.*, vol. 71, pp. 713–726, 2023, doi: 10.1109/TSP.2023.3250829.
[44] H. Wang, Z. Deng, B. Feng, H. Ma, and Y. Xia, "An adaptive Kalman filter estimating process noise covariance," *Neurocomputing*, vol. 223, pp. 12–17, Feb. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231216312176
[45] C. Buizza, T. Fischer, and Y. Demiris, "Real-time multi-person pose tracking using data assimilation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 438–447.
[46] M. Andriluka, U. Iqbal, E. Insafutdinov, L. Pishchulin, A. Milan, J. Gall, and B. Schiele, "PoseTrack: A benchmark for human pose estimation and tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5167–5176.
[47] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, p. 13, Dec. 2006, doi: 10.1145/1177352.1177355.
[48] S. Shreesha, M. M. M. Pai, U. Verma, and R. M. Pai, "Fish tracking and continual behavioral pattern clustering using novel sillago sihama vid (SSVid)," *IEEE Access*, vol. 11, pp. 29400–29416, 2023.
[49] M. Zhou, J. Li, C. Wang, J. Wang, W. Zhai, and V. Puig, "Global round-up strategy based on an improved Hungarian algorithm for multi-robot systems," *J. Intell. Robotic Syst.*, vol. 110, no. 4, p. 168, Dec. 2024, doi: 10.1007/s10846-024-02190-4.
[50] L. Ramshaw and R. E. Tarjan, "On minimum-cost assignments in unbalanced bipartite graphs," HP Labs, Palo Alto, CA, USA, Tech. Rep. HPL-2012-40R1, 2012, vol. 20, p. 14.
[51] A. J. Robertson, "A set of greedy randomized adaptive local search procedure (grasp) implementations for the multidimensional assignment problem," *Comput. Optim. Appl.*, vol. 19, no. 2, pp. 145–164, 2001, doi: 10.1023/a:1011285402433.
[52] R. Duvignau and R. Klasing, "Greediness is not always a vice: Efficient discovery algorithms for assignment problems," *Proc. Comput. Sci.*, vol. 223, pp. 43–52, Aug. 2023, doi: 10.1016/j.procs.2023.08.212.
[53] D. Osokin. *Real-time 3D Multi-person Pose Estimation Demo*. Accessed: Mar. 25, 2025. [Online]. Available: https://github.com/Daniil-Osokin/lightweight-human-pose-estimation-3d-demo.pytorch
[54] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, S. Sridhar, G. Pons-Moll, and C. Theobalt, "Single-shot multi-person 3D pose estimation from monocular RGB," 2017, *arXiv:1712.03453*.
[55] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2D pose estimation using part affinity fields," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1302–1310.
[56] C. Montella, "The Kalman filter and related algorithms: A literature review," *Res. Gate*, vol. 10, pp. 1–17, Jun. 2011.
[57] R. R. Labbe. (2015). *Kalman Bayesian Filters Python*. Github. [Online]. Available: https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python
[58] W. Bauer and G. Westfall, *University Physics With Modern Physics* (Connect learn succeed). New York, NY, USA: McGraw-Hill, 2011.
[59] S. El Omda and S. R. Sergent, "Standard deviation," in *StatPearls [Internet], Treasure Island (FL)*. Treasure Island, FL, USA: StatPearls, Aug. 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK574574/
[60] U. K. Kandagatla. (2022). *Fall Detection Dataset*. Accessed: May 18, 2025. [Online]. Available: https://www.kaggle.com/datasets/uttejkumarkandagatla/fall-detection-dataset
[61] E. Auvinet, C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Multiple cameras fall dataset," DIRO - Université de Montréal, Montréal, QC, Canada, Tech. Rep. 1350, 2025. Accessed: May 18, 2025.
[62] M. Arif and R. S. Anand, "Run length encoding for speech data compression," in *Proc. IEEE Int. Conf. Comput. Intell. Comput. Res.*, Dec. 2012, pp. 1–5, doi: 10.1109/ICCIC.2012.6510185.
[63] The pandas development team, "Pandas-dev/pandas: Pandas," Zenodo, Apr. 2024, doi: 10.5281/zenodo.10957263.
[64] W. McKinney, "Data structures for statistical computing in Python," in *Proc. 9th Python Sci. Conf.*, S. van der Walt and J. Millman, Eds., Jun. 2010, pp. 56–61, doi: 10.25080/majora-92bf1922-00a.
[65] Scikit-Learn Developers. (2024). *Model Evaluation: Quantifying the Quality of Predictions*. Accessed: May 19, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html
[66] J. Yang, "Fall detection dataset for SFDM," *IEEE Dataport*, Sep. 2024, doi: 10.21227/4hmh-sq38.
[67] Z. Chen, Y. Wang, and W. Yang, "Video based fall detection using human poses," in *Big Data*, X. Liao, W. Zhao, E. Chen, N. Xiao, L. Wang, Y. Gao, Y. Shi, C. Wang, and D. Huang, Eds., Singapore: Springer, 2022, pp. 283–296.
[68] C. Zampino, F. Biancospino, M. Brienza, F. Laus, G. Di Stefano, R. Romano, A. Pennisi, V. Suriani, and D. D. Bloisi, "Fall detection using nao robot pose estimation in RoboCup SPL matches," in *Proc. 9th Italian Workshop Artif. Intell. Robot. (AIRO)*, Udine, Italy, Sep. 2022, pp. 88–95.

[69] Y.-F. Song, Z. Zhang, C. Shan, and L. Wang, ''Constructing stronger and faster baselines for skeleton-based action recognition,'' 2021, *arXiv:2106.15125*.
[70] Y. Wang and T. Deng, ''Enhancing elderly care: Efficient and reliable real-time fall detection algorithm,'' *Digit. Health*, vol. 10, pp. 1–11, Feb. 2024, doi: 10.1177/20552076241233690.
[71] A. Osigbesan, S. Barrat, H. Singh, D. Xia, S. Singh, Y. Xing, W. Guo, and A. Tsourdos, ''Vision-based fall detection in aircraft maintenance environment with pose estimation,'' in *Proc. IEEE Int. Conf. Multisensor Fusion Integr. Intell. Syst. (MFI)*, Sep. 2022, pp. 1–6, doi: 10.1109/MFI55806.2022.9913877.
[72] S. Liu, T. Mangla, T. Shaowang, J. Zhao, J. Paparrizos, S. Krishnan, and N. Feamster, ''AMIR: Active multimodal interaction recognition from video and network traffic in connected environments,'' *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 7, no. 1, pp. 1–26, Mar. 2023, doi: 10.1145/3580818.
[73] J. Liu, R. Tan, G. Han, N. Sun, and S. Kwong, ''Privacy-preserving in-home fall detection using visual shielding sensing and private information-embedding,'' *IEEE Trans. Multimedia*, vol. 23, pp. 3684–3699, 2021.
[74] X. Wang, E. Talavera, D. Karastoyanova, and G. Azzopardi, ''Fall detection with a nonintrusive and first-person vision approach,'' *IEEE Sensors J.*, vol. 23, no. 22, pp. 28304–28317, Nov. 2023.
[75] S. Hu, S. Cao, N. Toosizadeh, J. Barton, M. G. Hector, and M. J. Fain, ''Radar-based fall detection: A survey [Survey],'' *IEEE Robot. Autom. Mag.*, vol. 31, no. 3, pp. 170–185, Sep. 2024, doi: 10.1109/MRA.2024.3352851.
[76] A. Purwar and I. Chawla, ''A systematic review on fall detection systems for elderly healthcare,'' *Multimedia Tools Appl.*, vol. 83, no. 14, pp. 43277–43302, Oct. 2023.
[77] J. Hamm, A. G. Money, A. Atwal, and I. Paraskevopoulos, ''Fall prevention intervention technologies: A conceptual framework and survey of the state of the art,'' *J. Biomed. Informat.*, vol. 59, pp. 319–345, Feb. 2016, doi: 10.1016/j.jbi.2015.12.013.
[78] P. Gosine, V. Komisar, and A. C. Novak, ''The effect of handrail cross-sectional design and age on the speed and quality of reach-to-grasp reactions to recover balance,'' *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 66, no. 1, pp. 56–70, Jan. 2024.
[79] G. Shi, C. S. Chan, W. J. Li, K.-S. Leung, Y. Zou, and Y. Jin, ''Mobile human airbag system for fall protection using MEMS sensors and embedded SVM classifier,'' *IEEE Sensors J.*, vol. 9, no. 5, pp. 495–503, May 2009, doi: 10.1109/JSEN.2008.2012212.

JINMO YANG received the B.S. degree from Korea University, Seoul, South Korea, in 2019. He is currently pursuing the M.S. degree in software engineering with Hongik University, Sejong-si, South Korea. His research interests include natural language processing and software quality measurement, both based on AI approach.

YE JIN JIN received the B.S. and M.S. degrees from Hongik University, in 2023 and 2025, respectively, where she is currently pursuing the Ph.D. degree. Her research interests include generating cartoon and code based on natural language processing and validating reinforcement learning models.

R. YOUNG CHUL KIM received the Ph.D. degree in software engineering from the Department of Computer Science, Illinois Institute of Technology (IIT), USA, in 2000.

He worked at LG Research Center, South Korea, in 2001. He is currently a Professor with Hongik University. His research interests include test maturity model, model-based testing, meta-modeling, software process model, and software visualization.