OptiTrip: Detecting Falls Using Optical Flow

Kinjal Shah, Catalina Gomez, Erica Tevere, Maia Stiber

I. INTRODUCTION

Fall detection is an active area of research that has major potential impact. In the US, the aging population is constantly growing. Along with this growth, the fall death rate is increasing - up 30% from 2007 to 2016 [1]. Every year, over 3 million elderly people are treated in emergency departments as a result of fall injuries, and one in five falls causes severe injury, such as broken bone or head injury [1]. Accurate detection of falls is vital to timely intervention. Current state-of-the-art focuses on video- and sensor-based detection. Video based monitoring can provide critical support for detection of injury and falls, enabling improved response times and mitigating long term effects. We focused our project scope in fall detection using video due to the ubiquitous video monitoring in hospitals and elderly care homes where such applications can be most helpful. We aim to use optical flow and human pose tracking to localize falls from standard RGB video feed. We performed our analysis with videos from a standard falls database labeled with fall onset and termination, and our own collected data with simulated falls. While we specifically mention the impact to older populations, accurate fall detection can extend to other vulnerable populations including people affected by seizures, people with limited mobility, and patients recovering in hospital.

Project Goals	Achieved
Able to classify falls from nonfalls	✓
Able to detect onset of falls	✓
Able to detect termination of falls	✓

II. METHODS

A. Database

We initially leveraged the URFD dataset [2], which contains 30 fall and 40 daily life activity videos associated with three labels: person not lying on the ground, falling, and lying on the ground. We merge the second and third labels to denote a fall, having an average fall length of 1.97s. In addition to this dataset, we collected our own videos of 18 falls and 19 challenging daily life activities, like crouching and sitting, to evaluate the model's performance on more diverse and realistic videos. The average fall length of our videos was 1.07s. Both datasets had a sampling rate of 30 fps.

B. Optical Flow (OF)

We used optical flow (OF) information to identify the onset of a fall. We processed each video as grayscale images, and estimated the optical flow between adjacent frames using OpenCV's function *calcOpticalFlowFarneback* for a dense calculation. We specified a pyramid with three levels (three iterations at each level), a scale of 0.5 to compensate for large motions, and a window size of 25 pixels for a more robust estimation. After computing the optical flow vectors, we calculated the magnitude and orientation at each pixel. We filtered artifacts in the motion vectors due to brightness variations in the background using a threshold defined as the magnitude standard deviation at each frame. Then, we computed the average OF across frames, and considered a moving average to smooth this signal over a temporal window corresponding to 5% of frames. To determine the fall onset, we were interested in identifying large variations in the average OF magnitude over time. Thus, we calculated an approximate derivative as the difference in magnitude between consecutive frames, as we can assume a small time-step between each frame (dt), and defined a threshold as one standard deviation from the mean derivative. The first frame above that threshold was defined as the fall onset.

C. Human Pose (HP)

Using FacebookAI's Detectron2 Library [3], we explored pose tracking based fall detection. Poses are extracted for each RGB frame of a video in the COCO Keypoints format [4], detecting up to 17 keypoints per frame. For every frame, if the keypoint is detected in the current frame and the prior frame, the magnitude of the difference in position is calculated. Like with OF, since we assumed a small time-step between each frame (dt), this difference can be assumed to be the velocity of the point. We then filtered for outliers and computed the average velocity for the frame. As in optical flow based detection, we initially performed a coarse threshold-based prediction. This threshold could be trained with more data using a Support Vector Machine (SVM); however, that is outside the scope of our work.

D. Combined: OF + HP

As seen in Table I, our initial results from independently performing optical flow analysis and human pose detection were sub-optimal. Therefore, we implemented a combined approach. The overall workflow can be seen in Figure 1.

1) Image Processing: To address the varying image sizes across our self-made dataset, taken with a variety of cameras, we performed an area based normalization to ensure that optical flow values were not artificially

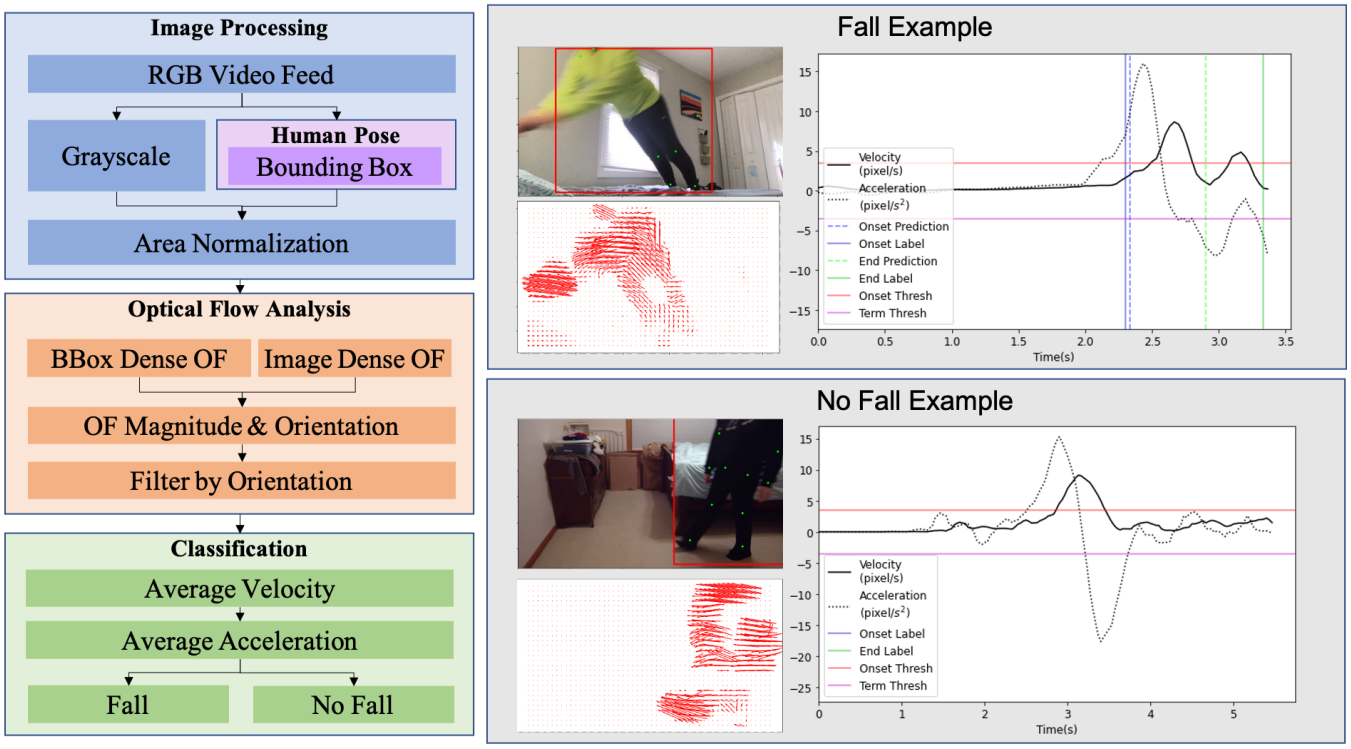

Fig. 1. Overview of algorithm (left), sample performance on fall (right-top) and no fall (right-bottom) videos.

inflated/deflated due to varying image sizes. Image dimensions were re-scaled based on Equation 1.

$$scale = \sqrt{\frac{target_{area}}{Img_{area}}} \tag{1}$$

2) Human Pose: As before, we computed the human pose and bounding box for each frame where a person is detected. In the combined implementation, we extracted only the bounding box as a method for focusing our optical flow calculation to reduce background noise. The bounding box was scaled using the scale factor in Eq. 1 to match the rescaled image.

3) Optical Flow Refinement: Given the optical flow from the bounding box region, or whole image if person is not detected, we calculated the magnitude, $|\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t}|$, and filter based on orientation, $\arctan(\frac{\partial y}{\partial t}, \frac{\partial x}{\partial t})$. Since falling motion has a large downward component for the optical flow vector, we filtered out directly horizontal motion and upwards motion from our optical flow to refine our signal (Fig. 2).

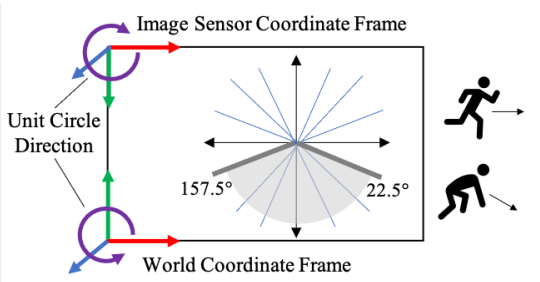

Fig. 2. As falls are primarily downward motion, we filtered magnitudes of optical flow based on orientation.

4) Classification: The refined OF signal provided us with the magnitude of downward motion across frames. We then computed the derivative to obtain the acceleration of motion. We smoothed the signal across a window of 15 frames. As falls in our dataset are estimated to take between 0.5 to 1.5 seconds, a signal displaying acceleration averaged across every 0.5 seconds provided improved fall detection. We performed classification of our output through a two-thresholding process. Our first threshold classified falls given the acceleration signal. This initial threshold intentionally allows for false positives over false negatives. This concern of increased false positives was addressed in our second classification step. As people generally go from upright ($H > W$) to horizontal ($W > H$) positions during falls (Fig. 3), we implemented a bounding box ratio (Height:Width) threshold. If the ratio changes from > 1 to ≤ 1 between the beginning and end of the video, we confirmed the classification of a fall. However, if we observed that the ratio remains > 1, we reclassify as a not fall. This check is skipped if bounding boxes are not detected in the first 6 or last 6 frames of a provided time window. This method successfully removed false positives from our classification. For both steps we performed hyperparameter tuning on a training set to determine appropriate thresholds to most accurately classify falls across datasets.

E. Evaluation

To determine how well our system performs, we compared the fall detection output from our method

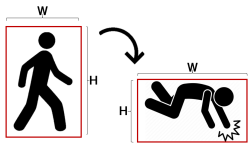

Fig. 3. Height and Width of Bounding Box Pre and Post-Fall

with the annotations provided by the datasets. Initially, we reported the difference between the true fall onset and the detected one. Later, we aimed to localize the complete fall, and computed the accuracy, precision, recall, and F1 score to describe the performance of our classification model.

III. RESULTS

TABLE I

RESULTS OVERVIEW ACROSS DATASETS AND DETECTION METHODS.

Optical Flow Based Approach				
Dataset	F1	Precision	Recall	Accuracy
URFD	0.6	0.43	1	0.43
OWN	0.65	0.49	1	0.49
Human Pose Based Approach				
Dataset	F1	Precision	Recall	Accuracy
URFD	0.71	0.66	0.77	0.73
OWN	0.7	0.64	0.78	0.68
Combined Approach				
Dataset	F1	Precision	Recall	Accuracy
URFD	0.78	1	0.63	0.83
OWN	0.78	0.95	1	0.97

Our final method (combined optical flow and human pose approach) performed the best of all of the algorithms with an accuracy of 83% for URFD and 97% for our own dataset. This was further reflected in the precision, recall and F1 for both datasets. See Table I for a complete summary of the evaluation metrics for the three methods.

TABLE II

AVG. DIFFERENCE BETWEEN LABELED AND DETECTED START & END FRAMES OF FALLS FOR TRUE POSITIVES (STANDARD DEVIATION).

Optical Flow Based Approach		
Dataset	Start Fall Diff. (s)	End Fall Diff. (s)
URFD	0.18 (0.76)	—
OWN	-0.6 (0.67)	—
Human Pose Based Approach		
Dataset	Start Fall Diff. (sec)	End Fall Diff. (sec)
URFD	0.32 (0.72)	—
OWN	-0.35 (0.66)	—
Combined Approach		
Dataset	Start Fall Diff. (sec)	End Fall Diff. (sec)
URFD	0.23 (0.15)	-1 (0.6)
OWN	-0.35 (0.57)	-0.71 (0.47)

In addition to fall detection, we localize the fall and identify the start and end frame. Refer to Table II for the average performance of each method to localize falls. Negative numbers mean that the fall was detected earlier than the labeled falls in seconds, and positive numbers mean they were detected later than labeled in seconds.

On average, our combined algorithm, for the URFD dataset, was 0.25 of a second late on detecting the start of the fall as compared to the annotations. When detecting the end frame, however, we identified the end frame one second early. For our own dataset, we noticed that our algorithm was, on average, 0.35 of a second early in detecting the start and 0.71 of a second early in detecting the end of the fall. However, we would like to highlight that annotating fall start and stop times is a highly subjective process.

IV. DISCUSSION

Our hypothesis was correct; combining both methods performed better than each method alone.

A. Optical Flow based approach

The major drawback of the OF approach is that the signal used to determine the fall onset is computed at all spatial locations in each frame, which makes it sensitive to background noise. Artifacts in the video background or movements that did not correspond to falls caused the optical flow based method to yield significant false positives, which is reflected in a low precision but high recall, as shown in Table I. In addition, the algorithm will find the most salient change in the average OF magnitude as the fall onset even if the video does not contain a fall. OF based analysis was highly successful during periods of high motion, when the motion signal magnitude was significantly larger than background noise or other motion type.

B. Human Pose based approach

The human pose detection method provided inconsistent signals, where keypoint and bounding box detection fails, especially during periods of high motion such as during a fall. It was very accurate during slower motion.

C. Combined: OF + HP

The complementary nature of the signals from both methods above led us to combine them to produce a more robust algorithm.

The accuracy of our final combined algorithm performed better on our own dataset than URFD. We believe this is due to our own dataset having the person around uniform depth whereas the URFD dataset has some large variations in depth of video subject across the datset. By looking at the breakdown for accuracy as well as precision and recall, we can see our predictions are conservative. In the context of this problem space, it is better to detect false positives than to label a false negative. It is important to note that all of the previous methods cannot determine the end frame; however, with our combined algorithm we are able to determine the ending of the fall.

As confirmed by our results, depth plays a significant role on observed optical flow from images, a person

moving at the same speed in the foreground of an image will be observed as "faster" than the person moving at the same speed further "into" the image. Therefore, future steps for our method include normalizing our OF with respect to relative depth of the person. When a person is falling parallel to the camera, the effect of this is not felt as their depth profile is overall consistent; however, the impact of depth on OF can be observed when a person falls perpendicular to the image plane, as the person's depth within the image space changes over time. To normalize for depth, we could calibrate cameras to obtain focal length and place an object of known dimensions in the frame; however, this method is not particularly feasible given the setting of where we like to do fall detection and would require generation of datasets with these constraints. Another option is simply replacing the RGB camera with an RGB-D camera, such as the Intel RealSense D455 [5], that can provide depth information localized to our image, however these cameras fail in the presence of natural light. Therefore the most optimal solution would be to use novel RGB image based depth reconstruction methods [6], which uses deep learning approaches to estimate depth of a person given monocular camera images. Given the compute resources required for such tasks currently, we did not implement this method.

D. Dataset Limitations

We evaluated our solution on the two datasets mentioned. The URFD dataset is a standard across related work in our problem space. We created our own dataset to add greater variability and focused edge cases to test our approach on. Although we achieved high performance across both datasets, we did find the limitations of the datasets to be one of the biggest challenges in crafting our solution.

Falls often occur spontaneously and can cause injury. Beyond the difficulty of capturing spontaneous events, causing people to purposefully fall (especially the targeted populations in this problem space such as elderly people or hospitalized patients) to produce a realistic dataset is highly unethical. As such, a dataset consisting of a large number of "true" falls does not currently exist, and is not used as the standard in this problem space. We also note that because the datasets are comprised of fake falls, we observed a deviation from velocity of real falls. The fake falls are often longer and the persons body movement can sometimes be not representative of a fall. Videos where this was clearly evident tend to be the fail cases of our method. The current datasets are pretty uniform with consistent background and similar controlled environments. A potential extension for dataset production is the use of physics engines to create simulated fall datasets as seen in Fig. 4.

In addition, these datasets require individuals to manually label the start and end of the fall. Each person labeling has their own criteria for when a fall starts and ends. The difference in labeling could easily be ± 10 frames given a sampling rate of 30 fps.

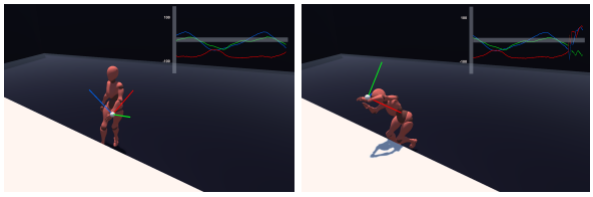

Fig. 4. Simulated forward fall using physics engine [7].

E. Linear Boundary Limitation

The method by which we classify falls assumes that a linear boundary exists between falls and non-falls. This worked well until dealing with edge cases that fall close to the boundary. If we wanted better performance on our edge cases, we would use a non-linear boundary for classification. An extension of our project would be learning this boundary using machine learning approaches such as a SVM. This limitation is probably why many state-of-the-art methods do not strictly use classical computer vision techniques and are concentrated in the machine learning sphere.

F. Lessons Learned and Advice

Through this project we learned the importance of good quality datasets. A good, properly labeled dataset can greatly affect the efficiency of the project, performance, and ease of implementation. Before jumping into a project make sure to properly look through the datasets you want to use. Initial exposure through a paper or website may indicate the dataset as fitting for your problem, but that does not necessarily mean that the quality is what is needed for your approach.

Another lesson is how sensitive OF is with regards to noise, image artifacts, and camera movements. We knew that using OF made assumptions that the movements were small between frames, but we did not anticipate how much camera movement and artifacts would affect this. This ties into with finding a good quality dataset. If one is attempting to use fundamental OF, make sure the dataset is appropriate with little camera movement and image artifacts to make it easier; otherwise, recognize that the problem will be harder if the data is not ideal.

V. CONCLUSION

Based on the results of our approach, we see satisfactory performance across different datasets. Although our results detect falls fairly well, we struggled to get top performance on either of our datasets. The limitations on our performance was due to thresholding, which presents a natural segue into why machine learning methods are popular in this problem space. Further extensions of our project would point us towards training a network given our optical flow output to detect and classify falls in video.

REFERENCES

[1] [Online]. Available: https://www.cdc.gov/falls/facts.html

[2] B. Kwolek and M. Kepski, "Human fall detection on embedded platform using depth maps and wireless accelerometer," *Computer methods and programs in biomedicine*, vol. 117, no. 3, pp. 489–501, 2014.

[3] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[4] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[5] [Online]. Available: https://www.intelrealsense.com/depth-camera-d455/

[6] T. Alldieck, M. Magnor, B. L. Bhatnagar, C. Theobalt, and G. Pons-Moll, "Learning to reconstruct people in clothing from a single rgb camera," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[7] A. Collado-Villaverde, M. Cobos, P. Muñoz, and D. F Barrero, "A simulator to support machine learning-based wearable fall detection systems," *Electronics*, vol. 9, no. 11, p. 1831, 2020.