A Deep Learning Approach to 3D Human Walking Pose Estimation From SensFloor Capacitive Signals

Anonymous Author(s)

Abstract

While camera-based systems are the dominant approach for human pose estimation, they face challenges in terms of privacy concerns and occlusion problems. These issues are of particular relevance in domains such as elderly care, where pose estimates can be used to monitor residents health or analyze incidents retrospectively. To assess an alternative to camera-based pose estimation, this paper aims to predict 3D walking poses using SensFloor: a capacitance-based floor which registers movement activity. We analyze the potential to utilize the floor's low-resolution signals to estimate poses and to what extent certain joint positions can be predicted accurately.

For this purpose, we collected synchronized SensFloor signals and video data, from which we extracted 3D human poses using MediaPipe to serve as ground-truth labels for training. These signals and their corresponding labels were then used for supervised training of an LSTM neural network. To estimate the person's position on the floor, a Kalman filter was applied to smooth the noisy SensFloor measurements.

Our results demonstrate that it is possible to predict human walking poses using the proposed methods, establishing a proof-of-concept for an alternative way of activity monitoring.

Keywords

SensFloor, Human pose estimation, Deep Learning, LSTM, Kalman Filter, Capacitive Floor, MediaPipe

Code: https://github.com/sensfloor

Create README for public organization view

1 Introduction

2 Methods

The final goal of this project is to design a system that reads SensFloor signals, processes them, and visualizes the person's gait within a virtual simulation. To achieve this goal, we split up the problem into two components: pose estimation and localization. The process of both components is illustrated in figure 1. By separating these tasks, the system is able to use a small localized active area of the floor for pose estimation. Afterward this pose is mapped across the global floor coordinates. This enables the system to be ported to various floors of different dimensions.

For the pose estimation component, we implemented a supervised fine-tuning pipeline. For that we collected reference pose estimates from a MediaPipe to serve as labels. Using these labels, we trained a Long Short-Term Memory (LSTM) to predict human poses based on SensFloor activations within an extracted Region of Interest (ROI). To localize the estimated pose, a Kalman Filter processes the noisy sensor signals to determine the person's global coordinates. Finally, the combined data of pose and position is streamed to a frontend application for real-time visualization.

[?]

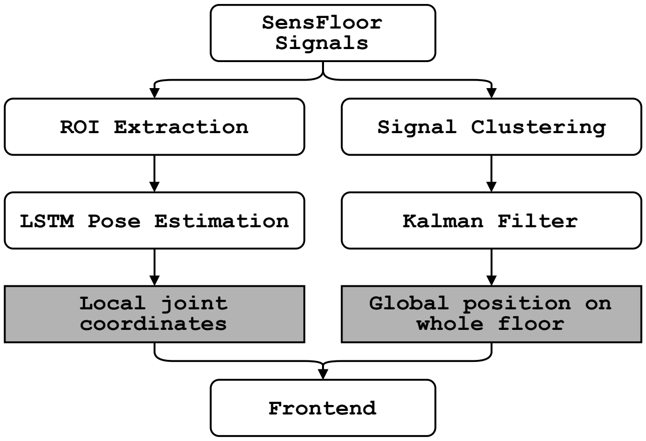

Figure 1: Overview of the SensFloor-based pose estimation pipeline — The left branch estimates the pose in a local, position-independent coordinate system, while the right branch localizes the person in global floor coordinates.

2.1 Data collection

The model developed in this paper takes SensFloor signals as input to predict the 3D joint coordinates of a person walking on the floor. To train and evaluate such a model, we recorded in two sessions a total of eight hours of synchronized SensFloor signals and video from three participants. To synchronize the data, we mapped the SensFloor signals to the corresponding video frame number at their time of occurrence. In post-processing, we used MediaPipe to extract 3D pose estimates from the video to serve as training targets. It is important to mention here that the origin of MediaPipe's pose estimates coordinate system is located at the center of the hips. This allows us to decompose the paper's problem into the previously mentioned two components of pose estimation relative to the hip and localization on the whole floor.

Look up actual duration

mention 15fps? 10Hz?

cite

training targets or reference labels?

2.2 Training Data preprocessing

Before giving the data to our model we preprocess it in several ways. First we filter signals that are beneath a threshold and likely correspond to noise from the floor. We defined the threshold to be $\tau = 140$, but it can vary by the floor setup. . Then we filter labels that do not contain SensFloor signals and vice versa ending up with a set of frames that contain both significant signals as input and a reference label. Next, we take the first 80% of this list as the training data, 10% as validation and 10% as test. When we load one item of this set we create a ROI history, which is a sequence of areas of the floor. We defined the sequence length as 50 and the ROI is a 4×4 grid of the floor with the signal values for this frame. The history is craeted by going back 50 frames and for each frame taking the signals it received or no signals if no signals were in this frame. All

equation format?

Patch signals should be explained

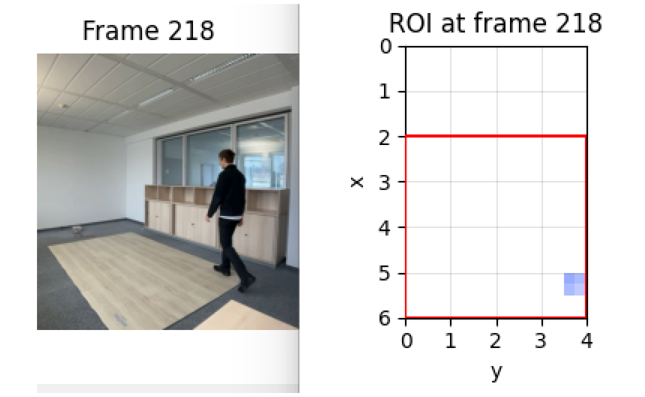

Figure 2: Example of one ROI in the sequence given to the model. The highlighted rectangle is the area of the ROI and the signals of the floor are in blue, showing the step of a person

signals are normalized to the range [0, 1] (127 will be 0 and 255 will be 1). Given the translation invariance of the data, we rotate the ROI history by 0, 90, 180, 270 randomly for a more robust training. Finally we use 13 of the 33 joints given by MediaPipe, excluding eyes, ears, mouth and fingers because they are irrevelant for our goal. Additionally we exclude the Foot heel and index, because MediaPipe predicted the foot unreliably to lengths of 3 cm to 18 cm.

2.3 Training Setup

We used a LSTM architecture for training our model. It is based on the model from [1]. We instead use a CNN to account for the Translation invariance. The full architecture is in figure 3. Instead of giving the model all signals of the floor, we construct a Region of Interest (ROI) which is a part of the floor containing the most signals in a defined sequence. A sequence is a collection of frames. For each frame we have a state of the floor. The state is constantly updated by the signals received from the patches. The CNN-LSTM model receives the signals in the ROI frame by frame and produces a vector with continuous values, the coordinate predictions for each joint. These coordinates are given into the Loss function. The total loss is defined as

$$L_{total} = L_{\text{WMSE}} + \lambda \cdot L_{\text{link}}, \tag{1}$$

where λ is a fixed scalar for L_{link}. We set λ to 0.1. The L_{WMSE} calculates the squared difference between predictions and the reference labels, the joint coordinates, weighted by specific landmark importance and is defined as

$$L_{\text{WMSE}} = \frac{1}{B \cdot J \cdot 3} \sum_{i=1}^{B} \sum_{j=1}^{J} \sum_{k=1}^{3} w_j (\hat{y}_{i,j,k} - y_{i,j,k})^2, \tag{2}$$

where B is the batch size, J the number of joints, k x,y and z coordinates of each joint, $\hat{y}_{i,j,k}$ the model's predictions and $y_{i,j,k}$ the reference labels. We increased the weight for the knees and feet to 10 compared to 1 for everything else, because they are the most active components in a walking scenario Furthermore we used the link loss from [2] defined as

$$L_{\text{link}}(d) = \begin{cases} k_{\min} - d, & \text{if } d < k_{\min} \\ d - k_{\max}, & \text{if } d > k_{\max} \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where d is the length of a link (e.g. shoulder width) and $k_{\min}$ and $k_{\max}$ are the top and bottom 0.03 percentiles of the link lengths in the training set. We assume an upright position, this is why the upper body and the head will only rotate and the focus is more on the knee and feet MediaPipe extracts 33 Joints, we only used 13, excluding the fingers, face expression, heels and foot index. For each epoch we calculate the following metrics: Mean joint position error (MJPE), percentage correct keypoits (PCK) with a 5 and 10 threshold

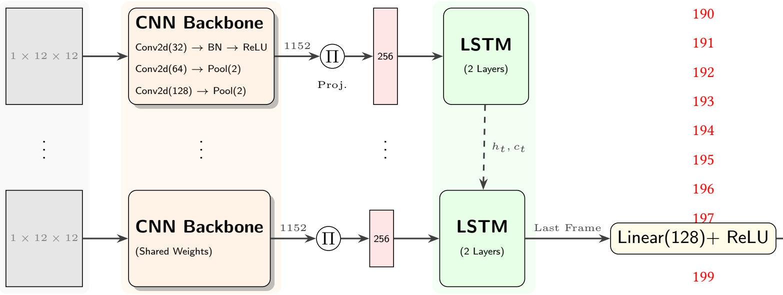

Figure 3: Overview of the model. We used a CNN-LSTM architecture with a self attention layer

We assume only one person is walking on the floor

2.4 Localization

To integrate the local pose estimate from the model into the global coordinate system of the floor, we developed a pipeline that tracks the subject's hip position relative to the SensFloor. We do this by extracting a raw pose estimate through clustering the measured sensor activations and applying a Kalman filter to smoothen the trajectory.

We identify the subject's contact points with the floor by grouping adjacent active sensor fields into clusters. The hip position is then calculated the following way: during a step, when both feet touch the ground, two clusters can be measured. In this case, we define the hip position as the midpoint between the centroids of the two clusters. If only one cluster is detected, meaning one foot is off the ground or both feet are close together, we assume the hip is located directly above that single centroid. This logic is illustrated in Figure 4 and allows us to extract the hip position from the received signals.

Because the raw position estimates are noisy and lead to jittery, unnatural movement in the visualization, we apply a Kalman Filter to ensure a smooth trajectory. Similar to [, 255 ff], the filter's state vector is defined as $\mathbf{x} = [x, y, \dot{x}, \dot{y}]^T$, where x and y are the position coordinates and $\dot{x}$ and $\dot{y}$ the corresponding velocities. Due to the absence of ground-truth tracking data for exact calibration, we tuned the filter's noise parameters empirically until the trajectory visually matched the subject's actual movement in the recorded

Is this diagram valuable enough for the space?

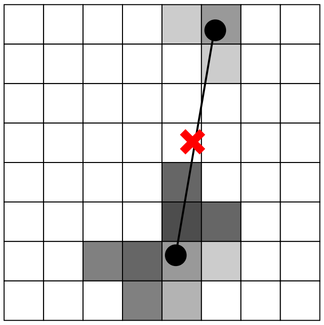

Figure 4: Calculation of the global position — This visualization shows two clusters of SensFloor activations. For both the centroid calculated. The final position is then determined by taking the mean position between both clusters.

videos. A comparison of the Kalman-filtered position estimates versus the raw positions is shown in Figure 5

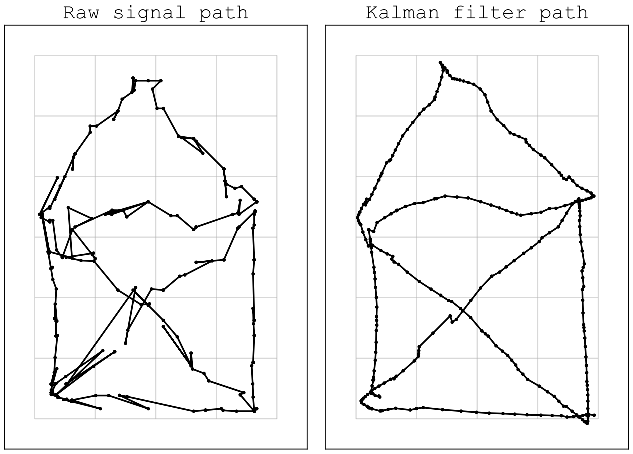

Figure 5: Effect of the Kalman filter — The left part of the image shows the raw signals that we obtain from the clustering of the activated fields from the walk of a person on the floor. On the right side are the signals processed by the Kalman filter which show a much smoother and realistic trajectory.

3 Results

4 Discussion and Conclusion

References

[1] Raoul Hoffmann, Hanna Brodowski, Axel Steinhage, and Marcin Grzegorzek. 2021. Detecting Walking Challenges in Gait Patterns Using a Capacitive Sensor Floor and Recurrent Neural Networks. *Sensors* 21, 4 (2021). doi:10.3390/s21041086

[2] Yiyue Luo, Yunzhu Li, Michael Foshey, Wan Shou, Pratyusha Sharma, Tomás Palacios, Antonio Torralba, and Wojciech Matusik. 2021. Intelligent Carpet: Inferring 3D Human Pose from Tactile Signals. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 11250–11260. doi:10.1109/CVPR46437.2021.01110