TU Delft

Delft University of Technology

Document Version
Final published version

Licence
CC BY

Citation (APA)
Parlitsis, A., Vaidya, G., & Zuniga, M. (2026). Don't Pry Into My Personal Space: Exploiting mmWave Sensors to Provide Privacy for Areas With Glass Walls. *International Conference on Embedded Wireless Systems and Networks*, *2026*, 1-12. https://doi.org/10.3217/9vs0-nt86

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states "Dutch Copyright Act (Article 25fa)", this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Don't Pry Into My Personal Space: Exploiting mmWave Sensors to Provide Privacy for Areas With Glass Walls

Asterios Parlitsis
s.parlitsis@gmail.com
Delft University of Technology
Delft, Netherlands

Girish Vaidya
vaidya.girish@gmail.com
Delft University of Technology
Delft, Netherlands

Marco Zuniga
M.A.ZunigaZamalloa@tudelft.nl
Delft University of Technology
Delft, Netherlands

Abstract

Glass walls reduce illumination costs, enhance thermal comfort and improve social safety. However, they raise privacy concerns. People do not like passersby looking into personal areas, such as computer monitors. Our solution combines two upcoming technologies, mmWave sensors and transparent screens. The mmWave sensors detect passersby, and the transparent screen allows us to turn opaque only a small area instead of the entire wall. We build upon studies performing human pose estimation (HPE) from radar point clouds while addressing their two important shortcomings. They (i) focus on single static users, and (ii) estimate only the right pose, not the exact body location. We provide a solution that works for multiple mobile people, while simultaneously providing accurate location and pose estimation, enabling a robust identification of head coordinates. Our live evaluation shows that, for three passersby, dynamic opaque areas of 30 cm × 30 cm block the line of sight over 95% of the time. Furthermore, our method outperforms a competitive SoA alternative for HPE, reducing the RMSE error (worst-case error) for the head by a factor of 17. Our code and dataset will be publicly available to study human pose estimation in challenging mobile scenarios with multiple people.

CCS Concepts

• **Human-centered computing** → *Ubiquitous and mobile computing*; *Human computer interaction (HCI)*; • **Security and privacy** → **Privacy protections**.

Keywords

mmWave sensing, privacy protection, multi-person pose estimation

1 Introduction

1.1 Motivation

Social safety has been a major concern with reports suggesting increasing incidents of workplace bullying and harassment [16]. Threatened individuals feel safer when there is less opportunity for concealment [7]. One simple approach to address this concern is by using transparent materials in the construction of offices [7]. Using glass is a simple design intervention to increase social safety even during closed-door meetings. In addition to transparency, improved lighting conditions in glass offices add to the perception of safety. Additionally, transparent offices also help reduce the energy costs. However, transparent offices also raise privacy concerns. An unnoticed passerby could look into private devices, gaining access to personal details that could be later exploited. A trivial solution would be to use a presence detection sensor, such as an infrared sensor, to turn the entire glass wall opaque when movement is detected. However, that would negate the benefits of transparent offices. Thus, an adaptive system is needed – the solution must detect the presence of any passerby and selectively turn a small part of the glass opaque, such that any sensitive object is blocked from the line of sight. The opaque area must be kept to a minimum to maintain the benefits of transparent offices.

The solution requires a mechanism that detects and tracks the passersby and then turns the selective areas of the glass screen opaque. To implement the proposed solution, *firstly, we require a transparent screen that can selectively block specific areas.* Recently smart window panels with programmable opacity have been commercially introduced [29]. Similar to regular displays, these panels can project text, pictures, or videos; but in their default state, they are transparent like a glass wall. An important feature of these panels is that the pixel darkness controls the opacity at the corresponding point – the light pixels make the part of the window transparent while the dark pixels make it opaque. In our implementation, we project dark squares to block the view of passersby. *Secondly, the system requires a mechanism to detect passersby.* If multiple people are passing across, the system must simultaneously estimate their head positions and block their line of sight. A few sensing alternatives are available for human detection, and camera-based solutions may appear as the best choice since they excel at estimating human pose. With the easy availability of cameras and advanced image processing algorithms, it is possible to identify humans in a scene and precisely localize their head positions. However, the use of cameras is highly discouraged, especially in workplaces. Cameras convey a culture of surveillance. In Europe, a new legal framework, the General Data Protection Regulation (GDPR), mandates collecting data in the least privacy-invasive manner. Citizens are following through,

recently universities were compelled to withdraw the installation of cameras due to protests from students [1, 2]. Thus, there is a strong legal, societal, and ethical motivation to adopt a privacy-friendly alternative for detecting passersby in our application.

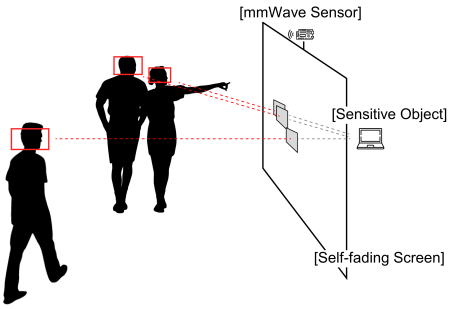


Figure 1: Main concept: First, a mmWave sensor estimates the location of the heads of various passersby. Second, a smart window projects black squares at the appropriate positions to block the line of sight between the passersby and the sensitive object.

In this work, we present a privacy-aware window that protects sensitive contents from the gaze of passersby. The key to selectively blocking the smart window lies in correctly estimating the head positions of individuals. Our system uses millimeter wave (mmWave) radar as an alternative to cameras for detecting the presence and posture of humans. The system performs multi-person posture estimation (MPPE) in real-time, identifies the head positions and projects black squares to selectively block a part of the window. Figure 1 shows the concept of our proposed application. We also provide a demonstration video at https://youtu.be/ZsH1TucxXQ to demonstrate the functionality of our system. The video shows a transparent screen and an mmWave sensor mounted on a table. Without loss of generality, we place a tag behind the screen that emulates a sensitive area. When a first passerby appears, a square appears on the screen and blocks his view. When a second passerby appears, a corresponding second square appears. The blocking squares follow the passersby as they move across attempting to view the sensitive area. Also, note that the square dimensions increase as the person gets closer to the screen. We encourage the reader to watch the video to attain a better understanding of our system.

1.2 mmWave radar for passersby detection

The current literature has explored the use of mmWave radars for human sensing across diverse applications such as gesture identification [18, 21], human activity recognition [14, 24], gait classification [13, 32] and crowd estimation [28, 30]. Radar sensors are a promising candidate for our application, but the scenarios we target pose unique challenges that the current state-of-the-art fails to address:

- **Mobile scenarios:** Current research in pose estimation focuses predominantly on actions performed on a *constrained area* and the sole interest is on *estimating the pose, not the location* [4–6, 22, 23]. We consider scenarios where people are mobile and a correct pose estimation is necessary but not sufficient. For example, the pose of a person may be correctly estimated as standing up, but if the location is wrong, the head coordinates will be incorrect. Thus, the opaque region on the glass panel will be wrongly positioned, failing to block the intended line of sight. Our application requires estimating the posture and location of individuals moving freely in the scene. These requirements introduce challenges associated with varying and sparse point clouds, especially when people move farther from the sensor.
- **Multiple People:** Another limitation of current studies is that pose estimation is done on single individuals [4–6, 15, 22, 23]. The simultaneous detection and pose estimation of multiple people creates additional challenges. When people walk across one another, it may cause temporary occlusions causing erratic estimation. Additionally, detecting more individuals translates to a need for higher compute and memory implying careful resource management. Since our system must detect multiple passersby in real-time, we require a pose and tracking estimation method that is light weight.
- **Real time system:** Our system must detect the passersby in real-time. This implies that our approach for human detection and tracking must be light-weight.

Our work overcomes these challenges through the following contributions.

- Our application requires precise estimation of head positions. To achieve this, we introduce what we believe to be the first real-time Multi-Person Pose Estimation (MPPE) system operating on mmWave radar point clouds. Our system first localizes multiple dynamic targets, i.e., identifies their positions in space and then estimates the posture for each localized target.
- We present a fully integrated system for a smart shielding application. The system successfully blocks the view of passersby by projecting black squares over smart window panels. In our evaluation, the system achieves 95% mean blocking accuracy for scenarios involving up to three simultaneous passersby through squares with dimensions $30\,cm \times 30\,cm$. For the localization error of the head position, our system

Table 1: Values of major radar configuration parameters used in our work.

Description	Values
Range Resolution (m)	0.044
Max. unambiguous Range (m)	8.02
Max. Radial Velocity (m/s)	1
Radial velocity resolution (m/s)	0.13
Frame Duration (msec)	100
CFAR threshold scale	1130
Range Peak Grouping	enabled
Doppler Peak Grouping	enabled
Static clutter removal	enabled

reports a Mean Average Error (MAE) and Root Mean Square Error (RMSE) of 13 *cm* and 16 *cm*, outperforming the selected baseline by more than 20% (17 *cm*) for the mean and 90% (279 *cm*) for the RMSE.

- We create a new radar dataset for many mobile people, further explained in Section 6. It comprises 88,851 labeled frames containing 5D mmWave point vectors and ground truth data for 19 human joints, collected from movements of 16 individuals. We plan to make the dataset and the code publicly accessible to promote further research on mmWave posture estimation.

2 System overview

The three components of our system are: a window panel, the sensor for detecting passersby, and the methodology to accurately project the blocking squares. Figure 2 shows an overview of our system. Next, we describe each of these components.

Window panel. Our implementation requires a window panel with adjustable opacity, in terms of pixel area, that can be dynamically controlled by the system. In case of Polymer Dispersed Liquid Crystal (PDLC) panels, it is not possible to change their opacity for only a part of the panel, either the entire panel is transparent or opaque. Unlike PDLC [20], with a smart window panel, we can control the opacity of every pixel. Thus, our prototype uses a smart window panel from VideowindoW [29] with dimensions $1.6\,m \times 1.1\,m$.

Sensor. As discussed in Section 1, we choose a mmWave sensor for our application. Specifically, we select the IWR1443 radar by Texas Instruments [26]. Prior to its deployment, we configure the radar to set various operational parameters such as its range and sensing resolution. Table 1 captures the configuration of the mmWave sensor used in our application.

Methodology. Our methodology adopts a top-down approach – we first identify and track all the human targets in the scene, identify the points corresponding to each target, then accurately estimate their postures and finally localize the heads.

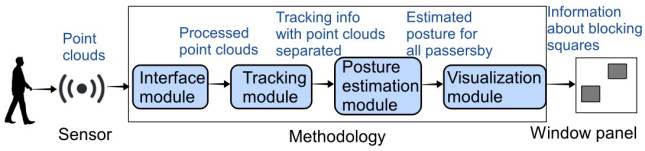


Figure 2: System overview. The point cloud collected by the sensor is processed by the system using the outline methodology. The system outputs the location and dimensions of the squares to be projected on the window panel.

We divide our methodology into four primary modules as follows:

- Interface module: The interface module configures the radar for optimum performance and receives the point cloud transmitted by the radar. By default, the point clouds obtain their coordinates assuming the radar as the origin, making the location of point clouds a function of the sensor's location (height and angle). We noticed that using this 'relative' sensor origin complicated mathematical tractability. The interface module translates the data into a global coordinate system, with origin located at the ground level and the xy plane parallel to the ground. That process is described in detail in Appendix A.
- Tracking Module: The first task for the top-down system is to track moving targets. This is done through a tracking algorithm using a motion model. Tracking is a well-known method that has been implemented in many state-of-the-art works. We use the GTRACK algorithm from Texas Instruments as our baseline [27] and adapt it for our application. Appendix B discusses the implementation of the tracking module.
- Posture Estimation Module: Our primary contributions and improvements are achieved through the posture estimation module. After the targets and their locations are identified, the posture estimation module generates a list of estimated 3D coordinates of human joints for all the detected targets, focusing on the head location. Section 4 discusses the details of our posture estimation approach.
- Visualization Module: After finding the positions and postures of the human targets in the scene, the system must block their view. This is achieved through the visualization module discussed in Section 5.

Next, we discuss the major challenges (not tackled by the SoA), and choose the most appropriate baseline.

3 Challenges and Baseline

3.1 Challenges

Noise. Environmental noise is one of the major challenges for mmWave sensing. Unlike stationary targets, spatial filtering is not effective to contain noise in our application. In our case, the region of interest is much larger, proportionately increasing the noise. The presence of multiple people causes more reflections contributing further to the noise. Our approach addresses multi-person scenarios through a tracking-first design that isolates individual targets and applies posture estimation independently per track. This design allows the system to generalize to multiple people even when trained on single-person data. We explain this concept further in Section 4.1.

Errors in location. In most posture estimation methods, obtaining an accurate location of the subject is not relevant, so long as the estimated pose is correct. In our application, a small error in estimating the location of the person would translate to a corresponding error in the location of the head. Thus, even if the posture estimation system is performing correctly, inaccuracies in location would result in erroneous projection of squares. Integrating tracking into posture estimation methods helps us identify the location and pose with higher accuracy, as explained in Section 4.2.

Occlusions and Sparsity of point clouds. Despite correctly computing the location of the person, the MPPE system with moving people faces additional challenges. Varying point cloud shapes and density across the frames cause discontinuity in the frames and makes it tricky to estimate the posture. Under such circumstances, a static posture estimation system exhibits abrupt changes in the posture. We address this challenge by integrating temporal learning to improve the continuity of postures across the frames. We explain this method further in Section 4.3.

3.2 Baseline

Our most critical requirement is having a lightweight method to enable a real-time system. Section 9 provides a detailed analysis of the SoA, showing that, for our purposes, the best SoA alternative is MARS [5]. MARS sorts the input point vectors according to their spatial coordinates, and then adjusts a constant input size, either by dropping data points or padding zeros to reach it. Finally, it reshapes the 64-entry array into an 8x8 matrix which is used as the input to the Convolutional Neural Network (CNN). The architecture of MARS maintains a straightforward structure, following a sequential execution of layers. It first passes the preprocessed 8x8 matrix input through two stages of CNN with 16 and 32 output channels and intermediate dropout layers. It then applies Batch Normalization (BN) and flattens the CNN's output. The flattened vector goes through two Fully Connected (FC) layers, reducing the vector's size to 512 and 57 neurons respectively. Each FC layer also includes intermediate dropout layers and Batch Normalization to avoid internal covariate shift and achieve more robust training. The model's output corresponds to the estimated 3D coordinates of 19 human joints with format $[\ x_1,\ x_2,\ \ldots x_n,\ y_1,\ y_2,\ \ldots y_n,\ z_1,\ z_2,\ \ldots z_n\ ]$. The total number of parameters in the model is 1,095,115 which is approximately half compared to another state-of-the-art model [22], further emphasizing its computational and performance gains.

4 Posture Estimation Module

Our MPPE system follows a top-down approach. The tracking module identifies and tracks all the individuals in the scene, isolates the points corresponding to individual tracks and passes this information to the posture estimation module. The posture estimation module separately predicts the posture of every individual. While the baseline MARS model performs well for the static postures, it fails to address the challenges posed by moving people, as discussed in Section 3.1. We now discuss the major improvements we implement over the baseline model to enhance its performance. We compare the performance of our improved system over the baseline model in Section 7.

4.1 Posture estimation on individual tracks

Unlike MARS which operates on the entire point cloud obtained from the scene, we perform posture estimation separately on the points for each track. *This is one of the key improvements of our work over the baseline.* Grouping points for individual subjects through tracking and then performing posture estimation provides several benefits. Firstly, random points corresponding to environmental noise are rejected better. They do not belong to any of the identified tracks and thus are not considered as part of the point cluster used for posture estimation. Secondly, performing tracking and then posture estimation for each track simplifies and generalizes the model. It breaks down the multi-person posture estimation problem into multiple single-person posture estimation problems. This enables MPPE through a simpler model architecture. Additionally, breaking down the problem also simplifies the model training. The model needs to be trained only on single-person samples but can be used on an arbitrary number of targets.

4.2 Position vector elimination

The tracking module monitors the position of every passerby. We exploit this tracking information to improve the position and ultimately reduce the absolute error of the system. *This approach, which integrates tracking information into posture*

estimation, is the next key improvement of our system. We call this approach *position vector elimination* and explain next the details of our approach.

The system's absolute error is a combination of two distinct factors:

- **Position error** ϵ_l: This is the error contribution due to incorrect position, measured as a difference between the estimated and true position of the spine base of a person. In other words, when the spine base between the prediction and ground-truth is aligned, ϵ_l equals zero.
- **Posture estimation error** ϵ_p: This error captures the difference between the true and estimated posture of a person assuming the spine bases are aligned. This is represented by the average error (MAE) and root mean square error (RMSE) computed across all the joints. We discuss this further in Section 7.2.2.

The baseline, MARS, is not designed to differentiate between the above two errors. MARS does not consider tracking, hence, it is completely vulnerable to location errors. As a result, if there are momentary occlusions or sparse point clouds, the system will be affected in two ways, the location error ϵ_l will increase and the posture estimation of MARS will also suffer from the noisy data, increasing ϵ_p. This double effect results in a higher absolute error for the head position.

In our proposed approach, we improve the estimation by isolating the two functions – position estimation and posture estimation. The position error is attributed to the tracking system, while the modified neural network contributes only to the posture estimation error. Continuously monitoring the target through the tracking system ensures that the centroid of the person is estimated correctly even if there are momentary occlusions. This reduces the position error and results in a lower absolute error. As shown in Figure 3, our *position vector elimination* has four steps. First, we track the centroid of the point cloud. Second, we translate the tracked centroid to the coordinate system's origin, becoming agnostic to the subject(s) location. Third, the translated point cloud is passed through the posture estimator. Fourth, the created human skeleton is placed back to its original position. A comparative analysis in Section 7.3.2 shows that our approach of isolating the errors provides better accuracy.

4.3 Creating temporal connections

The sequence of frames corresponding to a moving person exhibits a temporal pattern. The MARS model does not recognize and learn from this pattern. *Through temporal processing, our proposed approach demonstrates improved overall system performance. Furthermore, temporal processing is expected to improve robustness under sparse observations and temporary*

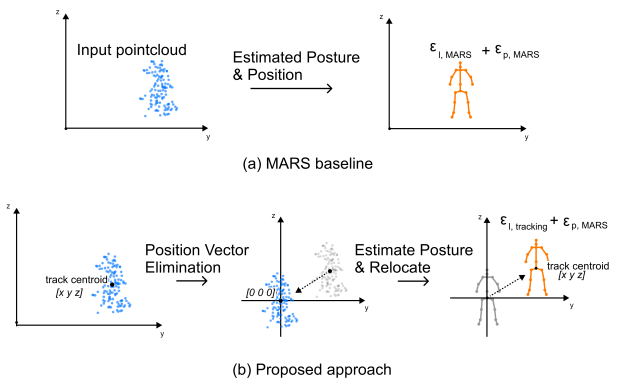


Figure 3: Comparison of the position errors between the MARS baseline and our approach. Our approach eliminates the position vector of the point cloud before estimating its posture.

occlusions by exploiting temporal consistency across consecutive frames. A dedicated evaluation under controlled sensing degradations remains part of future work. Following the insights of mmWave studies on other applications, which suggest combining frames to exploit temporal information [6, 18, 21], we implement two different methods to combine frames, viz., **frame fusion** and **frame stacking**. Frame fusion combines multiple frames into a single frame entity merging their temporal information to achieve higher point cloud density. Frame stacking processes multiple frames as an ensemble of layers, preserving their temporal information. Considering our system's sensing frequency (10 fps), we combine 3 frames to avoid introducing noise during human movements as suggested in [6]. Introducing frame fusion and frame stacking helps the system in the following ways:

Improvement in clustering and tracking. We utilize frame fusion for improved clustering and it is triggered only to identify new subjects. When new point clouds are detected in the scene, we first need to define if they belong to a person or if they are noise. This step is particularly important at longer ranges since the points are sparse resulting in their possible exclusion during cluster formation. Frame fusion improves the density of point clouds mitigating the impact of sparsity and leading to more efficient clustering, and hence, detection of passersby. During frame fusion, random noise does not accumulate in a dense area and is thus filtered out.

Improved posture estimation. Once a new person (track) is detected, *frame fusion* stops, after that, we only perform *frame stacking*. Our system extracts the last three frames corresponding to each track and stacks them. After removing the position vector (Section 4.2), we limit or pad the stacked frames to 64 points each and reshape the resulting arrays into

a 8x8x3 vortex. The vortex serves as the model's new input with 5 channels [$x\ y\ z\ \dot{r}\ i$]. To include temporal learning, we change the convolutional layers in the network architecture. We use a sliding 3D kernel of dimensions 3x3x3 at each convolutional layer. The kernel's first two dimensions refer to the spatial characteristics of the frames while the third is used to extract temporal connections from the data. The dimensions of the two convolutional layer output channels are kept at 16 and 32, and the output size of the flattening layer and the first Fully Connected (FC) layer is tripled to simultaneously handle the data of all the frames. Figure 4 illustrates the revised architecture of the posture estimation system including the 3D kernel modifications.

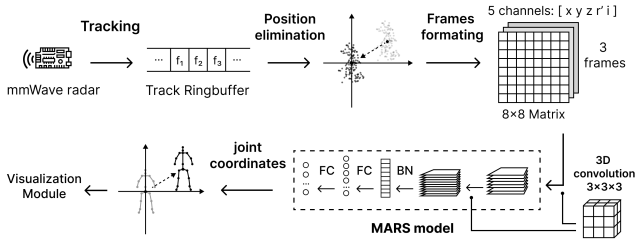


Figure 4: Multi-frame point cloud pre-processing per track and model architecture with 3D kernels for temporal connections.

5 Visualization Module

After estimating the position of the heads, the system must correctly project the black squares on the smart panel to block the gaze. The visualization module achieves this while maintaining a balance between privacy and transparency by controlling parameters such as the number of fading squares, their positions and size. In addition to the head position, the module takes the scene information as input through a configuration file.

5.1 Configuring the scene

To enable easy adaptability of the system in new environments, we capture the scene dependencies in a configuration file and provide it as one of the inputs to the visualization module. The module reads this information and correlates the various elements of the scene. Table 2 lists the various parameters captured through the scene configuration file. This parameterization allows the system to adapt to different installation geometries by adjusting sensor height, tilt, and panel position, without modifying the sensing or inference pipeline.

Table 2: List of scene configuration parameters.

Scene parameters	Description
h_{sensor}	Sensor installation height (m)
t_{sensor}	Sensor down tilt (degrees)
h_{screen}	Smart panel installation height (m)
$size_{screen}$	Smart panel height & width (m)
x_o, y_o, z_o	Sensitive objects coordinates (m)

5.2 Projecting the fading squares

The projection of the opaque squares over the screen requires two pieces of information, location and size. Next, we explain these two components.

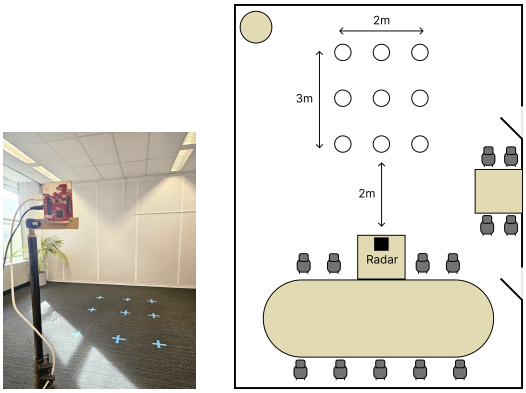
Location of squares. The posture estimation module outputs the location estimates of every human joint. Using this information, the visualization module combines the head coordinates with the scene parameters and calculates the gaze vector – a line extending from an individual's eyes to the sensitive object behind the smart panel – of each person. Through elementary trigonometry, the module determines the coordinates of points where the gaze vectors intersect with the smart panel. These points act as the centers of the projected fading squares. This formulation captures potential visibility rather than precise gaze direction. Estimating fine-grained gaze from mmWave sensing is challenging; therefore, we adopt a conservative strategy that blocks any plausible line of sight, prioritizing privacy over transparency.

Size of squares. To balance the tradeoff between privacy and transparency, we need to obtain the minimum block sizes that still provide privacy. An adaptive mechanism is designed to project bigger squares when the person is close to the panel and vice versa. Equation 1 defines the size of the projected squares, where y_{head} represents the y-coordinate of the head joint, indicating the head's distance from the window. We set a minimum $size_{min}$ and maximum $size_{max}$ threshold for the size of the squares, a distance threshold d_{min} that defines the minimum distance to the screen, corresponding to the maximum square dimensions and a size reduction rate β. The mechanism ensures a smooth transition in the dimension of the squares when the person is approaching towards or moving away from the window.

$$size_{rect} = \max \begin{cases} size_{min} \\ \min \begin{cases} size_{max} \\ size_{max} - \beta * (y_{head} - d_{min}) \end{cases} \end{cases} \quad (1)$$

6 Dataset creation

Another major contribution of our work is the creation of a new dataset for solving the challenge of real-time posture estimation for dynamic targets. We now explain the details regarding the setup, collection and post-processing of the dataset.



(a) Experimental scene defined by markers **(b) Top view of experiment room**

Figure 5: Setup to gather the data set

6.1 Preparation for gathering the dataset

Hardware and Devices. We use the IWR1443 mmWave sensor with the configuration discussed in Section 2 to capture the point cloud data.

Participants and experimental process. We engage 16 volunteers, 10 males and 6 females. The range of the participants' height was 35 cm with a mean of 177 cm, and a standard deviation of 8.6 cm. The participation was voluntary and the experiments were approved by the Ethics Committee of the Institution. The dataset's scope is limited to capturing one person at a time. This design is intentional: our system adopts a top-down approach in which tracking first isolates individual targets, and posture estimation is performed independently per track. As a result, training on single-person data is sufficient, while multi-person scenarios are handled at inference time through tracking and temporal modeling. Each participant was asked to participate alone in five experiments representing different types of movement, each experiment lasting approximately 2 - 2.5 minutes. The five experiments correspond to different movements: Static waving, Normal walking, Combined walking and waving, Static movements such as leaning and squatting, and Free movements.

Environment setup. Figure 5 shows our environmental setup and layout of the space for data gathering. We mounted the Kinect sensor at the height of 1 m with no tilt. The mmWave sensor was mounted at the height of 165 cm with a down-tilt of 5 degrees and in the same vertical plane as the Kinect with a small offset of 5 cm along the x-axis.

6.2 Capturing the Dataset

We perform simultaneous capture using mmWave sensor and the Kinect sensor using the system developed by [25] for data logging through Kinect. The data from the two sources is synchronized via the Network Time Protocol (NTP).

6.3 Dataset postprocessing

After capturing the data, we take steps to clean the data and pair the data between the two sensors. This helps create a reliable dataset that we use further for training. To pair the mmWave and Kinect frames, we use their timestamps, creating 1-on-1 associations between the frames that have the smallest absolute time difference between the timestamps. Further, we spatially align the co-ordinates by translating the data to a common ground reference.

Data partitioning. After performing all the processing steps, we have a dataset that includes 88,851 labeled frames. We split the dataset into training, validation, and testing sets in approximately a 60:20:20 ratio such that all the data from one individual stays within the same set. We will make this dataset publicly available for the benefit of the research community.

7 Evaluation

7.1 Implementation Details

First, we configure the tracking system. For the DBSCAN algorithm we choose an *eps* of 0.3, and the minimum number of points for cluster formation is 40. We use euclidean distances as the distance metric, based on the global origin discussed in Appendix B.

We use the same values as the baseline MARS model for the neural network parameters set during the training phase. Specifically, we use the Adam optimizer with a learning rate of 0.001, batch size of 128, and set the number of epochs during training to 150. On average, the validation loss converges to 0.01 after the 130^{th} epoch for our model, and a little later for the MARS baseline. The Mean Square Error (MSE) in Equation 2 defines the validation loss, where x_i, y_i, z_i are the values estimated by the system and $\hat{x}_i$, $\hat{y}_i$, $\hat{z}_i$ are the ground truth values for the joint coordinates.

$$loss_{MSE} = \frac{\sum_{i=0}^{18}(x_i - \hat{x}_i)^2 + \sum_{i=0}^{18}(y_i - \hat{y}_i)^2 + \sum_{i=0}^{18}(z_i - \hat{z}_i)^2}{3 * 19} \quad (2)$$

7.2 Performance

We first evaluate the performance of our final application, evaluating its real-time effectiveness to block the gaze for multiple individuals. After that, we demonstrate that our system also achieves significant improvements in human pose estimation.

7.2.1 Online Evaluation. We perform a live evaluation of our final application. *It is important to highlight that the baseline, MARS, cannot be tested in this scenario because it cannot cope*

with multiple mobile people due to the multiple occlusions and missing frames that occur. In the next subsection, we compare our approach with MARS using our curated dataset. That evaluation will show that our approach performs better even in the scenarios for which MARS was originally designed.

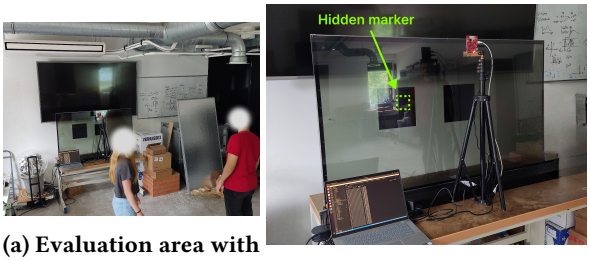
To emulate a realistic setting, we conduct the evaluation in a new environment, different from the one used to gather the dataset. To mimic sensitive content, we place a marker at a 1 m distance behind the smart screen. Figure 6 shows the evaluation area with participants and the smart screen setup with a marker. The volunteers move freely in an area of $8\,m \times 4.5\,m$, attempting to view the marker while the screen tries to prevent it by projecting the dark squares. The configuration parameters used during our online evaluation are listed in Table 3.

Table 3: Values of scene configuration parameters.

Scene parameters	Values
h_{sensor} (m)	1.8
t_{sensor} $(degrees)$	-5
h_{screen} (m)	1.3
$size_{screen}$ (m)	1.6×1.1
$x_{marker}, y_{marker}, z_{marker}$ (m)	(0.32, 0, 1.3)
$size_{min}$	Varying
$size_{max}$ (m)	0.4×0.4
d_{min} (m)	0
β	-0.08

We conducted nine experiments, varying the number of participants in the scene and the minimum size of the squares. Each experiment is conducted for three minutes and is repeated five times with different combinations of volunteers. We restrict the maximum number of participants to three participants. The limitation to three simultaneous individuals arises from the physical size of the smart window rather than sensing or inference constraints. While the radar and tracking pipeline can detect more individuals, projecting a large number of opaque regions reduces transparency within the available display area. Every person carries a smartphone with a simple application that allows them to record the state as **Blocked** and **Not Blocked** using a simple touch. We define the system's accuracy as the average percentage of time the marker is blocked from the people in the scene. While this evaluation relies on participant-reported feedback, it captures perceived privacy, which is inherently user-centric. To reduce bias, each experiment is repeated multiple times with different participants and averaged across trials. Camera-based visibility verification would provide valuable complementary evaluation and remain an important direction for future work.

Table 4 captures the results of our online evaluation. We observe that the system accuracy improves with increasing



(a) Evaluation area with two participants in the scene. **(b) mmWave and smart screen setup**

Figure 6: Online evaluation setup

Table 4: Accuracy of the system based on the minimum square size ($size_{min}$) and the number of people in the scene.

Size of squares	1 person	2 persons	3 persons
30 cm	*96%*	*95%*	*98%*
20 cm	*93%*	*92%*	*90%*
15 cm	*88%*	*81%*	*79%*

square dimensions. This aligns with our expectations, as larger squares compensate for any errors introduced during the positioning and posture estimation. Conversely, the accuracy declines as the number of people increases. The increase in number of participants increases the system lag and also the number of occlusions. However, even with square sizes of 20 cm, we observe that the performance degradation is minimal, demonstrating the resilience of our system. In the outlier case of 30 cm squares tested with 3 people, the large squares covered a major percentage of the screen's surface. The significant coverage blocked people's sight, even when the originally assigned squares failed to do so themselves.

7.2.2 Offline evaluation for posture estimation of a single person. Having evaluated the efficacy of our system in our final application, we now measure its pose estimation accuracy. Figure 7 provides a qualitative indication of the model's performance by comparing the postures estimated by our approach against the ground truth. We quantify the system accuracy through the Mean Absolute Error (MAE) given by Equation 3 and the Root Mean Squared Error (RMSE) given by Equation 4, computed across the entire testing set. MAE and RMSE together provide the absolute error and the variance in error. If the RMSE is significantly higher than MAE, it implies a higher error variance which translates to a higher worst-case error. In our application, a high RMSE implies that the posture estimation error would vary across frames resulting in abrupt changes in posture. *As we describe next, a key strength of our approach is not only that it reduces the average absolute error (MAE) by a factor of 1.3×, but it reduces*

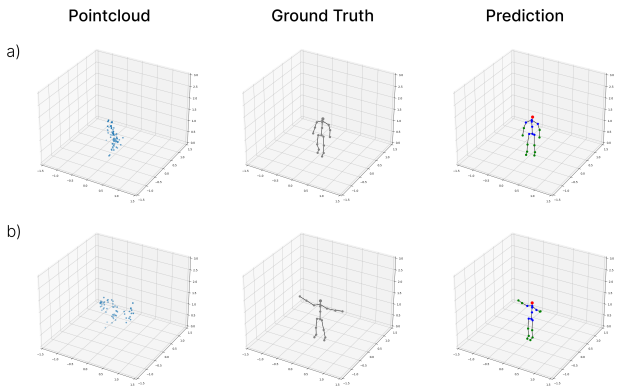
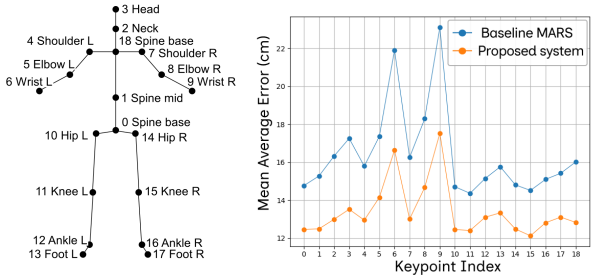


Figure 7: Samples of pose prediction. The movements presented are a) walking, b) waving arms



(a) Skeleton diagram illustrating the mappingbetween indexes andcorresponding joints (b) MAE comparison between the end-to-end baseline MARS and the proposed system approach over a single dynamic target.

Figure 8: MAE comparison and the corresponding keypoint indices.

the variance of the worst case scenarios (RMSE) by a factor of 17×.

We use a method similar to 10-fold cross-validation for computing the MAE and RMSE. We split the dataset randomly into ten groups and perform ten iterations of training and evaluation. During each iteration, six random groups of the dataset are used for training, two for validation and two for testing. The final performance metrics are obtained by averaging the results across all the iterations. The final application requires a precise position of the head. However, for a fair comparison with the MARS model, we use performance metrics in which all 19 human joints contribute equally to the loss.

$$loss_{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3}$$

$$loss_{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{4}$$

Figure 8b plots the MAE error across the 19 joint indices comparing our solution with the baseline model. The various joints and the corresponding indices are shown in Figure 8a. Table 5 summarizes the average MAE and RMSE computed across all joints and separately for the head for both models. We observe that our system outperforms the baseline MAE by approximately 20%, achieving a mean average error of 13 cm for the head and 13.1 cm across all joints. Furthermore, the RMSE error improves significantly by a factor of 17×. This implies a significant reduction in worst-case error which would translate to better continuity across frames in the final application.

Table 5: Average MAE and RMSE error (in cm) for the combined 19 joints and the head joint.

	All joints		Head	
	MAE	RMSE	MAE	RMSE
Baseline MARS	16.43	251.87	17.26	279.32
Our approach	13.15	16.21	13.00	15.74

7.2.3 System Performance and Latency. Our system operates at a sensing frequency of 10 frames per second, enabling near real-time tracking and projection. The inference latency is measured to be below 100 ms, which is sufficient for interactive applications. The use of a lightweight model architecture, adapted from MARS, ensures efficient processing for multiple tracked individuals. The sensing range (approximately 8 m) covers typical indoor environments.

7.3 Ablation Study

As described earlier, we perform three main enhancements over a state-of-the-art pose estimation method. In this section, we present an ablation study to demonstrate the impact of our improvements. The analysis is performed using our dataset with 10-fold cross-validation, as described in Section 7.2.2. We note that our ablation study is conducted on clean and well-aligned data to ensure reliable comparison. In real-world deployments, additional challenges such as occlusions, missing frames, and environmental noise are addressed through tracking and temporal modeling. While not explicitly isolated in this evaluation, these components are validated through the online multi-person experiments.

7.3.1 Improvement-1: Tracking system. We discussed in Section 4.1 that including a tracking system prior to the posture estimation module helps with filtering out random noise and ghost targets. The comparison of Improvement-1 with the baseline (MARS) is in Table 6. The results illustrate that the tracking system lowers the RMSE error for the head by a

factor of 14× and thus improves the accuracy of the worst-case predictions. The filtered input allows the CNN model to perform more accurately and consistently, improving the overall performance.

7.3.2 Improvement-2: Position vector. One of the key contributions of our work is exposing the intertwined effect of location and pose on error estimation and the ability to decompose both using a position vector. Table 6 shows the results of Improvement-2. We observe an additional improvement of approximately 10%. While this value may seem like an incremental improvement, the functionality is critical for our application. As explained in 4.2, this enables consistent positioning of the person if there are momentary occlusions.

7.3.3 Improvement-3: Temporal connections. Similar to what happens with Improvement-2, the final system (which includes Improvement-3) suggests an incremental gain of 4% in RMSE for the head position. However, as explained in Section 4.3, establishing temporal connections helps to achieve continuity of posture across the frames. Our ablation studies are conducted on the postprocessed and filtered data set. There are no missing frames or occlusions in our data and anomalies were removed. Our objective was to create a dataset with good correspondence between the point clouds and the ground truth so that the extended research community could use it. Under such ideal conditions, the contributions of Improvement-2 and Improvement-3 seem incremental. However, in real-world deployments, the impact of missing frames due to occlusions and sparse point clouds is mitigated through these improvements.

7.4 Results Summary

We summarize all results from the proposed approaches and improvements. Table 6 presents the average MAE and RMSE errors, separately for the head and all 19 human joints. Figure 9 plots the MAE error of every posture estimation approach discussed during the ablation studies across all joints.

Table 6: Average MAE and RMSE error (in cm) for the combined 19 joints and the head joint over all the analyzed approaches.

	All joints		Head	
Approaches	MAE	RMSE	MAE	RMSE
Baseline MARS	16.43	251.87	17.26	279.32
Improvement 1	14.55	19.08	15.51	20.03
Improvement 2	13.42	16.68	13.52	16.37
Proposed System (All Improvements)	**13.15**	**16.21**	**13.00**	**15.74**

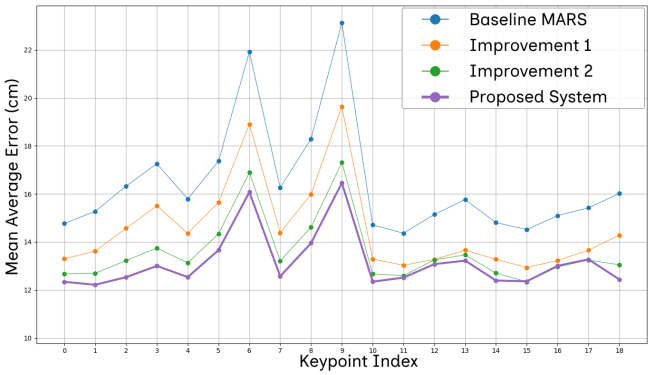


Figure 9: MAE comparison of all analyzed approaches.

8 Limitations and Deployment Considerations

Environmental complexity: Our evaluation is conducted in controlled indoor environments. In more complex settings, clutter and reflections may introduce sensing artifacts. The tracking module helps mitigate spurious reflections as transient detections are less likely to form stable tracks. However, remaining sensing artifacts may result in additional or larger blocking regions, leading to conservative behavior that preserves privacy at the cost of reduced transparency. Evaluating system performance under more challenging deployment scenarios, including higher pedestrian speeds, denser crowds, complex occlusions, varying environmental conditions, and other sources of sensing uncertainty, remains an important direction for future work.

Merged targets and adaptive blocking: In scenarios where closely spaced individuals are represented as a single track, the visualization module can be extended to adapt blocking regions based on the spatial extent of the detected point cloud. This would lead to conservative behavior where ambiguous detections result in increased blocking.

Evaluation methodology: Our online evaluation relies on participant-reported blocking success, which may introduce subjectivity. However, since privacy perception is inherently user-dependent, this provides meaningful insight into system effectiveness.

Despite these limitations, our results demonstrate that selective, privacy-preserving shielding is feasible and effective in realistic scenarios.

9 Related Work

To the best of our knowledge, this is the first implementation which demonstrates selective blocking of a glass wall with a privacy-friendly sensor such as mmWave.

HPE using other technologies. The high-resolution images captured by cameras and the advances in computer vision enable HPE with high accuracy. The works in [11, 19] use the top-down approach whereas the authors of [8, 12] propose the bottom-up approach. However, as discussed earlier,

camera-based approaches raise serious privacy concerns and are discouraged in public places. WiFi-based approaches have been suggested in [3] and [31]. However, these WiFi systems use a large setup with multiple antennas raising scalability concerns.

HPE using mmWave. HPE with mmWave is challenging due to the sparsity of point clouds. Many of the earlier works with mmWave-based HPE used heat maps for data processing. mmPose [22] was proposed in 2020 and was the first mmWave based system capable of detecting more than 15 human keypoints from a single target. Similarly, mPose [23] first performed human detection to remove environmental noise and introduced a domain discriminator to remove subject-specific characteristics. HuPR [15] proposed a novel model as well as a new preprocessing approach at the on-chip data path to better extract velocity information. The authors in [9] introduced a model architecture of two streams combining temporal and spatial data, one learning the joint constraints and the other finding patterns of joint movements between frames. While the results were encouraging, all these works followed a heat-map-based approach that requires a high computation and memory which are not suitable for our application. To address this problem for real-time applications, researchers proposed MARS [5], a simple yet effective model which operates on raw point cloud data. We select MARS as our baseline due to its lightweight architecture and ability to operate directly on point cloud data, making it suitable for real-time deployment. Other approaches relying on heatmap representations or multi-sensor setups impose higher computational or hardware requirements, limiting their applicability in our setting.

Pose Estimation Datasets. Our dataset is an important contribution of our work. For our privacy-shielding application, the appropriate dataset would require raw point cloud data and include dynamic movements, such as walking, which is the primary action of interest. However, the existing datasets either utilize heat-map formatting or fail to address dynamic movements adequately. These observations highlight the necessity for a new dataset. Table 7 provides an overview of all the proposed posture estimation datasets comparing them with our dataset.

10 Conclusions

Through this work, we demonstrate a novel system that selectively blocks only a part of the glass wall by accurately localizing the heads of passersby. Our system integrates tracking and posture estimation to create the first-of-its-kind multi-person posture estimation (MPPE) with privacy-friendly mmWave sensor in mobile scenarios. Our application shows that privacy can be maintained for more than 95% of cases by exploiting small opaque areas.

Table 7: Comparison of existing mmWave datasets for the task of posture estimation. (*) In the mRI dataset only 1 of the 12 movements features global displacement in a strictly straight line (**) Our dataset includes 11 distinct actions at different orientations, as well as unsupervised, free movements.

	Movement	#Movements	Representation	# Participants	# Frames	Public
mmPose	Static	4	heatmap	2	15k	No
MARS	Static	10	point cloud	4	40k	Yes
MiliPoint	Static	49	point cloud	11	213k	Yes
mRI	Static*	12	point cloud	20	160k	Yes
HuPR	Dynamic	3	heatmap	6	141k	Yes
Our dataset	Dynamic	11**	point cloud	16	89k	Yes

References

[1] [n. d.]. Leiden University students fight against classroom surveillance cameras. https://dutchreview.com/news/protests-cameras-leiden-university///.

[2] [n. d.]. Universiteit Utrecht stopt met 'druktecamera's' in collegezalen na studentenprotest in Leiden. https://www.rtvutrecht.nl/nieuws/3245218///.

[3] Fadel Adib et al. 2015. Capturing the Human Figure through a Wall. *ACM Trans. Graph.* (2015).

[4] Sizhe An, Yin Li, and Umit Ogras. 2022. mRI: Multi-modal 3D Human Pose Estimation Dataset using mmWave, RGB-D, and Inertial Sensors. arXiv:2210.08394 [cs.CV]

[5] Sizhe An and Umit Ogras. 2021. MARS: mmWave-based Assistive Rehabilitation System for Smart Healthcare. *ACM Transactions on Embedded Computing Systems* (2021).

[6] Sizhe An and Umit Y. Ogras. 2022. Fast and scalable human pose estimation using mmWave point cloud. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*. ACM.

[7] Christine Boomsma and Linda Steg. 2014. Feeling safe in the dark: Examining the effect of entrapment, lighting levels, and gender on feelings of safety and lighting policy acceptability. *Environment and Behavior* (2014).

[8] Z. Cao et al. [n. d.]. OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. *IEEE Transactions on Pattern Analysis & Machine Intelligence* ([n. d.]).

[9] Lin Chen et al. [n. d.]. Spatial-temporal Multi-scale Constrained Learning for mmWave-based Human Pose Estimation. *IEEE Transactions on Cognitive and Developmental Systems* ([n. d.]).

[10] FmmW-Group. 2024. IWR1443-Python-API. https://github.com/FmmW-Group/IWR1443-Python-API. Accessed: 2024-06-02.

[11] Kaiming He et al. 2018. Mask R-CNN. arXiv:1703.06870 [cs.CV]

[12] Eldar Insafutdinov et al. 2016. DeeperCut: A Deeper, Stronger, and Faster Multi-Person Pose Estimation Model. arXiv:1605.03170 [cs.CV]

[13] Prabhu Janakaraj et al. 2019. STAR: Simultaneous Tracking and Recognition through Millimeter Waves and Deep Learning. In *2019 12th IFIP Wireless and Mobile Networking Conference (WMNC)*.

[14] Feng Jin et al. 2019. Multiple Patients Behavior Detection in Real-time using mmWave Radar and Deep CNNs. 1–6.

[15] Shih-Po Lee et al. 2022. HuPR: A Benchmark for Human Pose Estimation Using Millimeter Wave Radar. arXiv:2210.12564 [cs.CV]

[16] Jose M León-Pérez et al. 2021. The presence of workplace bullying and harassment worldwide. *Concepts, approaches and methods* (2021).

[17] MathWorks. 2024. *DBSCAN - Density-Based Spatial Clustering of Applications with Noise*. MathWorks. https://ww2.mathworks.cn/help/stats/dbscan.html

[18] Sameera Palipana et al. 2021. Pantomime: Mid-Air Gesture Recognition with Sparse Millimeter-Wave Radar Point Clouds. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* (2021).
[19] George Papandreou et al. 2017. Towards Accurate Multi-person Pose Estimation in the Wild. arXiv:1701.01779 [cs.CV]
[20] PDLC Glass. [n. d.]. Why choose smart pdlc glass? https://www.pdlcglass.com/.
[21] Dariush Salami et al. 2022. Tesla-Rapture: A Lightweight Gesture Recognition System from mmWave Radar Sparse Point Clouds. *IEEE Transactions on Mobile Computing* (2022).
[22] Arindam Sengupta et al. 2020. mm-Pose: Real-Time Human Skeletal Posture Estimation using mmWave Radars and CNNs. *IEEE Sensors Journal* (2020).
[23] Cong Shi et al. 2021. mPose: Environment- and subject-agnostic 3D skeleton posture reconstruction leveraging a single mmWave device. *Smart Health* (2021).
[24] Akash Deep Singh et al. 2019. Radhar: Human activity recognition from point clouds generated through a millimeter-wave radar. In *Proceedings of the 3rd ACM Workshop on Millimeter-wave Networks and Sensing Systems.*
[25] Swapnil. 2019. Kinect Skeleton Tracking. https://github.com/swapnil0908/Kinect-Skeleton-Tracking Accessed: 2024-06-09.
[26] Texas Instruments. [n. d.]. IWR1443. https://www.ti.com/product/IWR1443.
[27] Texas Instruments. 2024. Tracking with GTRACK. https://dev.ti.com/.
[28] Girish Vaidya and Marco Zuniga. 2024. Exploiting mmWave and Deep-Learning Models to Estimate People Count in Urban Scenarios. In *2024 IEEE/ACM Ninth International Conference on Internet-of-Things Design and Implementation (IoTDI).* IEEE, 73–84.
[29] VideowindoW. [n. d.]. Transforming Windows into Screens. https://www.videowindow.eu/.
[30] Ehsan Yavari et al. 2014. Is There Anybody in There?: Intelligent Radar Occupancy Sensors. *IEEE Microwave Magazine* (2014).
[31] Mingmin Zhao et al. 2018. Through-Wall Human Pose Estimation Using Radio Signals. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition.*
[32] Peijun Zhao et al. 2019. mID: Tracking and Identifying People with Millimeter Wave Radar. In *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS).*

A Interface module

The interface module reads the data from the radar and converts it into a format that simplifies processing by the other modules. The radar senses the scene as a point cloud with each point representing position information, radial velocity, and signal intensity. We use an available API [10] for reading the point cloud from the radar sensor. The point cloud default format is $[x\ y\ z\ \dot{r}\ i]$, where x, y, z are the point coordinates, $\dot{r}$ the radial velocity and i the signal intensity. The tracking system operates on Cartesian coordinates and thus we translate the velocity output to Cartesian form. Translating the velocity in Cartesian simplifies the calculations of the Kalman filter in the tracking module.

The point cloud data received from the radar provides the scene information with the sensor as the origin of the coordinate system and the xy plane perpendicular to the plane of the sensor board. We eliminate the impact of sensor setup on received data by transforming the data to a global coordinate system. Using basic principles of linear algebra, we define a Translation Matrix T, to offset the sensor height and a Rotation Matrix R_{inv}, to rotate (roll) the axis system over the x-axis. The transformed values refer to a global coordinate system with the origin at the ground level and the xy plane parallel to the ground plane.

B Tracking module

Figure 10 illustrates our implementation of the tracking module based on GTRACK proposed by Texas Instruments [27]. However, we have made certain changes to GTRACK to adapt it to our application.

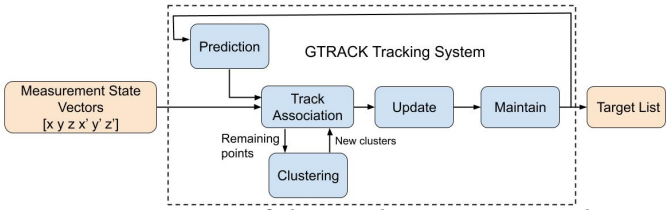


Figure 10: Overview of the Tracking system pipeline.

Track Association. The Track Association submodule gets measurement state vectors as input together with the predicted position of targets from the prediction submodule. Based on the information from the prediction submodule, it associates the points to each of the tracks and marks the remaining points as unassociated.

Clustering. The points that are not associated with any tracks undergo Clustering. GTRACK utilizes a clustering algorithm almost identical to the DBSCAN (Density Based Spatial Clustering of Applications with Noise) [17]. *We redefine the distance metric of DBSCAN improving the efficacy of clustering.* Similar to the work in [32], we introduce a diminishing weight z_{weight} over the z-axis' contribution, where $z_{weight} \in (0, 1)$. We set z_{weight} to 0.4 in our implementation. Further we introduce a damping factor $\alpha(y_\mu)$, that takes a value in the range (0,1) and is inversely proportional to the mean range value of the points. This improves clustering at longer range. The final revised distance metric is presented in Equation 5.

$$d = \alpha(y_\mu)\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + z_{weight}(z_A - z_B)^2} \tag{5}$$

Update and Maintain. The update step estimates the true location of the objects based on the predictions and the actual measurements. Further, the attributes of each track are updated. Each track is labeled as either *Static* or *Dynamic* based on the velocity of the corresponding centroids, and they are assigned a maximum lifetime accordingly.

Prediction. The current locations of the tracks predict the likely location of the targets at the next instance based on a motion model. We use a Kalman filter with a constant acceleration model for prediction in our implementation.