

G³R: Generating Rich and Fine-grained mmWave Radar Data from 2D Videos for Generalized Gesture Recognition

Kaikai Deng, *Student Member, IEEE*, Dong Zhao, *Member, IEEE*, Wenxin Zheng, Yue Ling, Kangwen Yin, and Huadong Ma, *Fellow, IEEE*

Abstract—Millimeter wave radar is gaining traction recently as a promising modality for enabling pervasive and privacy-preserving gesture recognition. However, the lack of rich and fine-grained radar datasets hinders progress in developing generalized deep learning models for gesture recognition across various user postures (e.g., standing, sitting), positions, and scenes. To remedy this, we resort to designing a software pipeline that exploits wealthy 2D videos to generate realistic radar data, but it needs to address the challenge of simulating diversified and fine-grained reflection properties of user gestures. To this end, we design G³R with three key components: (i) a *gesture reflection point generator* expands the arm’s skeleton points to form human reflection points; (ii) a *signal simulation model* simulates the multipath reflection and attenuation of radar signals to output the human intensity map; (iii) an *encoder-decoder model* combines a *sampling module* and a *fitting module* to address the differences in number and distribution of points between generated and real-world radar data for generating realistic radar data. We implement and evaluate G³R using 2D videos from public data sources and self-collected real-world radar data, demonstrating its superiority over other state-of-the-art approaches for gesture recognition.

Index Terms—Generalized sensing, Synthetic radar data, Cross domain translation, Gesture recognition, 2D videos

1 INTRODUCTION

In recent years, millimeter wave (mmWave) radar has gained growing interest among researchers as a reliable sensing modality for supporting pervasive and privacy-preserving human sensing [1–7], as it remains unaffected by adverse conditions (e.g., foggy weather, poor illumination) due to its strong permeability while not revealing texture information of users. Radar-based studies are also gradually developing from *coarse-grained* (i.e., sensing tasks typically focus on users’ whole-body movements) activity recognition [5] and object detection [1] to *fine-grained* (i.e., sensing tasks typically focus on fine-grained changes in user gestures) gesture recognition [4, 8], enabling applications such as assisting users to effortlessly control common household appliances according to their gestures under poor illumination conditions [9, 10], providing users with a better interactive experience in AR/VR according to recognized gestures [11, 12], and assisting workers to operate machines according to recognized gestures in an industrial plant [6, 13].

In practice, it is difficult to achieve a generalized gesture recognition system, as it necessitates the consideration of various practical factors, such as user postures (standing, sitting, etc.), user positions, and collection scenes of gesture data. For example, the gesture recognition accuracy

significantly drops from 96.7% to 47%, when training on a large-scale radar dataset of users performing gestures while standing, and testing with radar data of users performing gestures while sitting; similarly, changes in user positions and scenes also result in a drop in recognition accuracy by 18.08% and 5.26%, respectively (see Section 2). To address this problem, a direct method is to employ rich data across various user postures, positions, and scenes to train a generalized deep learning model. Unfortunately, compared to video and sound datasets, there are very few corresponding radar gesture datasets [3, 4], which are further constrained by a single human body posture, data collected at fixed positions, and limited scenes. Meanwhile, collecting and labeling a large-scale radar dataset require substantial human efforts, slowing down the research and development [7, 14]. Fortunately, there are many public 2D video datasets [15–19] and websites (e.g., YouTube¹, Bilibili²) that offer abundant human gesture data across various user postures, positions, and scenes. Furthermore, existing studies have utilized 2D videos to generate other sensor data, such as Inertial Measurement Unit (IMU) [20, 21] and sound data [22, 23], for training deep learning models, which motivates us to explore the possibility of using 2D videos to generate realistic radar data.

Exciting studies have explored the generation of realistic radar data from different data sources, such as motion capture (MoCap) data [24, 25] and depth camera data [26, 27]. However, either the generated radar data is coarse or it lacks some common gestures. The generative adversarial

• K. Deng, D. Zhao, W. Zheng, Y. Ling, K. Yin, and H. Ma are with Beijing Key Laboratory of Intelligent Telecommunication Software and Multimedia, School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, 100876, China. E-mail:{dkk, dzhao, zhengwenxin, lingyue, 393473185, and mhd}@bupt.edu.cn

(Corresponding authors: Huadong Ma and Dong Zhao.)

1. <https://www.youtube.com/>

2. <https://www.bilibili.com/>

networks (GANs) [28, 29] are widely used to augment existing radar datasets, but it could easily lead to data confusion when performing similar gestures. Recently, some studies [5–7] explore using 2D videos to generate radar data. However, these methods primarily focus on generating coarse-grained radar data by just capturing users’ whole-body movements, which makes it difficult to characterize fine-grained changes in their gestures. In contrast, we focus on utilizing wealthy 2D videos to generate fine-grained radar gesture data, yet we will face a unique challenge.

How to simulate diversified and fine-grained reflection properties of user gestures? Existing works [5–7] can generate 2D Range-Doppler signals or point clouds using 2D videos. However, either the signal is difficult to accurately characterize the 3D spatial representation of gestures or the point cloud is coarse-grained. Given the corresponding regions of each human body part, we can calculate the necessary components of radar data, i.e., the radar cross-section (RCS) and radial velocity of every vertex, in each region with respect to a radar sensor. However, there are three different influencing factors: user postures, user positions, and various scenes, all of which would lead to diversified differences in the gesture data produced by users. If each vertex is directly extracted to obtain the corresponding RCS and radial velocity without considering the multipath reflection and attenuation of radar signals during the transmitting and receiving process, this diversified and fine-grained difference will be ignored, resulting in lower fidelity of the generated radar data.

To address this challenge, we design a unique data generation system, G^3R , that allows converting wealthy 2D videos into fine-grained radar data. Specifically, a *human parsing* model, a *skeleton extraction* model, and a *depth prediction* model extract human constituent parts, skeleton points, and depth information, respectively, to prepare for subsequent modules. Considering that the main reflection points of the human body focus on the arm, we design a *gesture reflection point generator*, which expands the reflection points of the arm by using random interpolation based on the arm’s skeleton points. However, interpolated reflection points may experience depth value shifts during arm movement. To this end, we map human constituent parts to the generated reflection points, correcting any overflow points and ensuring that the generated points completely belong to the arm.

Based on the obtained reflection points, we can calculate their corresponding RCS and radial velocity, but directly using them will result in lower quality of the generated radar data due to neglecting the multipath reflection and attenuation of radar signals. To address this problem, a *signal simulation model* is designed, which takes RCS and depth information as inputs to simulate the propagation characteristics of radar signals during the transmitting and receiving process, followed by outputting the human intensity map. Due to the differences in various user postures, positions, and scenes, the generated radar data differs from the real-world radar data in number and distribution of points. Therefore, an *encoder-decoder model* combines a *sampling* module and a *fitting* module to generate realistic radar data through graph convolution and matrix transformation. Besides, we utilize both a large amount of generated and

a small amount of real-world radar data to train gesture recognition models to further enhance their generalization.

In summary, our contributions are as follows:

- To our knowledge, this is the first work on a system called G^3R that utilizes wealthy 2D videos to generate rich and fine-grained radar data for developing a generalized gesture recognition model across various user postures, positions, and scenes.
- We propose a suit of novel and effective techniques: (i) a *gesture reflection point generator* utilizes the extracted skeleton points to expand the reflection points of the arm; (ii) a *signal simulation model* simulates the multipath reflection and attenuation of radar signals during the transmitting and receiving process, followed by outputting human intensity map; (iii) an *encoder-decoder model* combines a *sampling* module and a *fitting* module to generate realistic radar data.
- We implement and extensively evaluate G^3R for gesture recognition with 2D video data from five public datasets [15–19], YouTube, and Bilibili, as well as self-collected real-world radar data from 32 volunteers (a total of 23,040 samples collected over 3 months). The experimental results show that G^3R achieves 90.51% accuracy, which surpasses three state-of-the-art approaches, *Vid2Doppler* [5], *SynMotion* [7], and *Midas* [6] by 67.41 percentage points (pp), 66.28 pp, and 51.36 pp, respectively, when training a model solely with all generated radar data; if we add a small amount of real-world radar data for training, G^3R achieves 97.32% accuracy, 61.80 pp, 54.67 pp, and 34.45 pp higher than that of *Vid2Doppler*, *SynMotion*, and *Midas*, respectively. Moreover, when facing a new user, the model trained on all generated data only requires 6 samples per gesture to achieve the recognition accuracy of 96.76%, while it only requires 6 samples per gesture to achieve 98.53% accuracy by combining a small amount of real-world radar data for training. Besides, we deeply evaluate the impact of various factors using self-collected real-world radar data from 5 volunteers (a total of 8400 samples). G^3R achieves 90.06% and 96.99% accuracy across various user postures, positions, and scenes when training with all generated radar data only and all generated radar data with a small amount of real-world radar data, respectively.

2 MOTIVATION AND CHALLENGES

2.1 Necessity of Generating Rich Radar Data

Liu et al. [4] have collected and openly shared a large-scale radar dataset of users performing gestures with a single posture (standing). To verify the necessity of generating rich radar data, we collect a radar dataset of users performing gestures while sitting, including 32 volunteers, 5 gestures (pull a hand (PL), push a hand (PS), draw a circle (CR), lift up a hand (UP), and knock a virtual table twice (KO), as shown in Fig. 1), 9 positions, and 2 scenes. Each gesture is performed 8 times, resulting in a total of $32 \times 5 \times 9 \times 2 \times 8 = 23040$ samples. Using this dataset, we evaluate the influence of various user postures, positions, and scenes. Note that we utilize the control variable method to maintain two factors

constant while verifying the impact of the remaining one factor.

Impact of User Postures. Different user postures would cause the reflection of radar signals to change. Fig. 2 shows the data distribution of a user performing the same gesture while standing and sitting, respectively. To verify the influence of different user postures on gesture recognition accuracy, we train a state-of-the-art radar-based gesture recognition model, *mTransSee* (hereinafter referred to as the model) [4], using a large-scale radar dataset of users performing gestures while standing, and test it with radar data of users performing gestures while sitting. The gesture recognition accuracy significantly drops from 96.7% to 47%, which indicates significant disparities in the radar data produced by different user postures. Moreover, we augment the existing gesture recognition dataset with real-world data of users performing gestures while sitting to train the model. As depicted in Fig. 3, we observe that the recognition accuracy has an upward trend with the increase of data. When the number of samples increases to 8000, the recognition accuracy reaches 94.4%. Therefore, it is imperative to continuously add a substantial amount of sitting posture samples for training to further improve the model's performance.

Impact of User Positions. The number of points and signal intensity are different when a user performs gestures in various positions, and this observation has also been validated by *mTransSee* [4]. Based on the collected dataset, we conduct corresponding experiments to demonstrate that the recognition accuracy is affected by user positions when a user performs gestures while sitting. We evaluate the recognition accuracy of user gestures at deviations of 50 cm from 9 positions. As shown in Fig. 4, we observe that: (i) the recognition accuracy is continuously improved with increased training data from different positions; (ii) even when training the model with data from 9 positions, the recognition accuracy only reaches a maximum of 76.66%. Note that the model can achieve 94.74% accuracy when training and testing on radar data from 9 positions. Therefore, the model requires more data from different positions for training to achieve higher accuracy that satisfies users' requirements.

Impact of Different Scenes. There are various reflectors in different scenes. When a user performs different gestures close to a reflector (i.e., within 60 cm), the reflector generates multipath reflection points that cannot be effectively processed using common noise filtering methods, such as CFAR [30] and DBSCAN [31], making it difficult to differentiate between the signal reflection generated by user gestures and that of the reflector in various scenes. In this case, if the model does not learn sufficient knowledge, it will suffer a drop in recognition accuracy. To clearly show this, we perform a leave-one-scene-out cross validation. As shown in Fig. 5, we observe that: (i) when training using data from scene 1 and testing with scene 2, the recognition accuracy drops by 5.26 pp; (ii) when training using data from scene 2 and testing with scene 1, the recognition accuracy drops by 3.77 pp; the main reason is that the scene 2 is more complex, enabling the model to learn more about reflectors. Therefore, it is necessary to mix a large amount of data from different scenes for training to improve the model's performance.

Data Collection Cost. To collect the above dataset, we spend 3 months. Similarly, Liu et al. [4] collect a gesture dataset, enabling the model to accurately recognize gestures performed by users while standing, but it takes approximately 6 months. Apart from time cost, we must consider other complex factors that impact costs: (i) researchers need possess professional expertise and adeptly operate equipment while executing proficient data processing tasks. Note that some areas, particularly remote and underdeveloped areas, may suffer from a scarcity of professional talents, rendering data collection more difficult; (ii) data quality may be poor during collection, necessitating re-collection; (iii) some volunteers may quit after performing gestures for several hours due to boredom and fatigue, resulting in the interruption of data collection; (iv) unconscious gesture noise can be easily produced during data collection process; (v) labeling data is burdensome and prone to errors; (vi) there are many more user postures, user positions, and scenes in real life, which requires collecting more data to make deep learning models more generalized. Moreover, according to a survey [14], all researchers surveyed agree that collecting data is very difficult.

Summary. Based on the above analysis, we observe that the performance of gesture recognition is affected by many factors, requiring rich training data to continually enhance the model's performance, but collecting large-scale radar data is costly. Therefore, this motivates us to explore an efficient approach to address the lack of radar data and high collection cost.

2.2 Opportunities

To obtain rich radar data, we tend to use other data sources to generate radar data. Inspired by this idea, we investigate existing data generation efforts and find that radar data can be generated from various data sources. However, different data sources have different pros and cons. *First*, some works [24, 25] utilize MoCap data to generate radar data, but such data is relatively sparse, resulting in very coarse radar data. *Second*, some works [26, 27] use depth camera data to generate radar data, but these datasets only have few gesture classes. In contrast, the ubiquity of mobile devices has produced vast amounts of 2D videos daily [32], which are either collected as public datasets [15–19] or accessible on some websites (e.g., YouTube, Bilibili), covering rich gesture content. Meanwhile, the computer vision technology enables the accurate extraction of 3D postures and skeletal points of humans from 2D videos, offering a foundation for exploring and generating large-scale radar data. Therefore, we set out to design a software pipeline that utilizes wealthy 2D videos to generate rich radar data, addressing the lack of radar data and reducing the cost of collecting real-world radar data.

2.3 Design Challenges

The main challenge in designing the G^3R system lies in simulating diversified and fine-grained reflection properties of user gestures. Although recent works [6, 7] consider the reflection properties of radar signals when generating radar data, they mainly simulate the coarse-grained motion reflections of humans, and the generated data is difficult

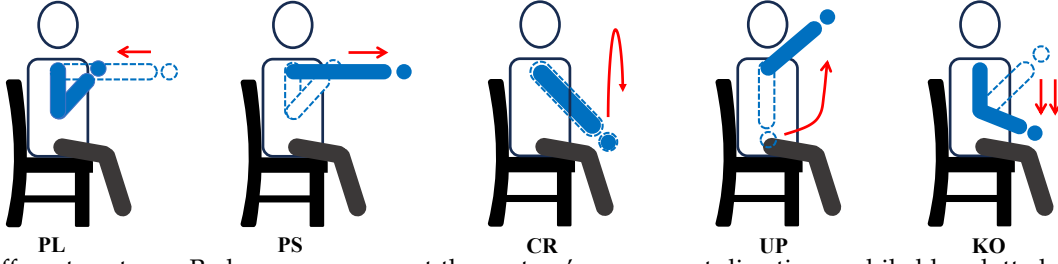


Fig. 1: Five different gestures. Red arrows represent the gesture’s movement directions, while blue dotted and solid lines represent the starting and ending of different gestures, respectively.

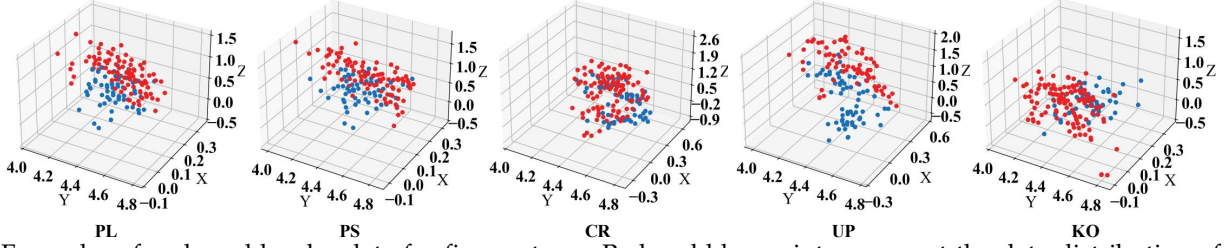


Fig. 2: Examples of real-world radar data for five gestures. Red and blue points represent the data distribution of a user performing the same gesture while standing and sitting, respectively.

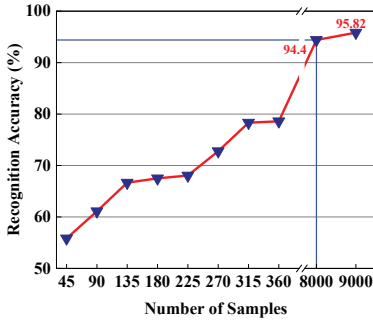


Fig. 3: Recognition accuracy as sitting posture samples increase.



Fig. 4: Recognition accuracy as the number of positions increases.

to accurately characterize the diversified and fine-grained gesture features. We evaluate the quality of data generated by existing works under various factors, as shown in Fig. 6. Note that each gesture sample typically comprises dozens or hundreds of reflection points, with each one being linked to a corresponding signal intensity value. The signal intensity range for each reflection point (each chirp) typically lies within the range of -150 to +150 dB. However, it is meaningless to compare the signal intensity for each reflection point, so we calculate the average cumulative errors in the signal intensity and radial velocity for all collected gesture samples. Compared to the real values, there are significant average cumulative errors in the signal intensity and radial velocity obtained through three state-of-the-art methods [5–

Train Scenes	Test Scenes	
	1	2
2	91.44%	96.30%
1	96.70%	92.53%

Fig. 5: Recognition accuracy with different scenes.

7] under 2 user postures (standing and sitting), 9 user positions, and 2 scenes. For example, for the latest method *Midas* [6], the errors for signal intensity/radial velocity reach 16408 dB/20.12 m/s, 19608 dB/35.96 m/s, and 13472 dB/51.35 m/s, under three different factors, respectively. Note that according to our experience, we need these two errors to be less than 4000 dB and 15 m/s, respectively, to satisfy users’ requirements (see Section 5.2). This shows that it is an important problem to simulate diversified and fine-grained reflection properties of user gestures. If the errors are not effectively addressed, it will hinder the model from learning the real features of each gesture, causing a drop in recognition performance.

3 SYSTEM OVERVIEW

As shown in Fig. 7, G^3R consists of six modules. Specifically, there are differences in the reflections of various human body parts when a user performs gestures, with the primary reflection concentrated on the arm (including the hand) rather than other human body parts. To this end, we utilize a *human parsing* model to obtain different human constituent parts and accurately output gesture regions, followed by characterizing the reflection differences in various regions of the human body. Meanwhile, a *skeleton extraction* model is used to extract the skeleton points of the human body. Since the main reflection points are focused on the arm, we design a *gesture reflection point generator*, which exploits the random interpolation method to expand the skeleton

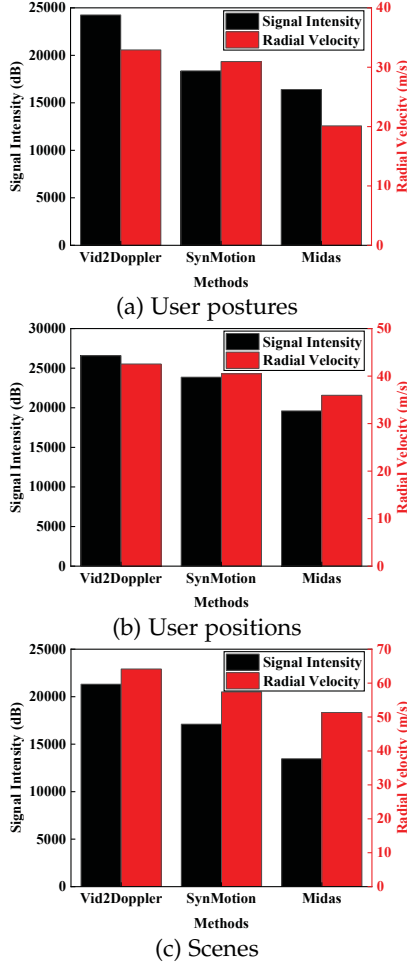


Fig. 6: Average cumulative errors of signal intensity and radial velocity for different methods with various factors.

points of the arm (the number of interpolation points can be obtained according to the statistical results of real-world radar data), followed by concatenating skeleton points of other human body parts to obtain the human reflection points. Note that since the extracted skeleton points are 2D data, a *depth prediction* model is used to supplement their depth information.

Based on the human reflection points, we can calculate their RCS and radial velocity. However, there will be serious multipath reflection and attenuation during the transmitting and receiving process of radar signals, especially when there are many reflectors. To address this problem, a *signal simulation model* is designed, which utilizes RCS and depth information as inputs to simulate the propagation characteristics of radar signals to output the human intensity map, followed by concatenating the radial velocity to generate the initial radar data. Considering the changes in user postures, positions, and scenes, there are differences in number and distribution of points between the generated and real-world radar data. Therefore, an *encoder-decoder model* is designed, which consists of a *sampling* module and a *fitting* module; the former utilizes graph convolution to obtain the number and distribution correlation of points between generated and real-world data, followed by using sampling layers to output generated points, and the latter employs matrix transformation to fit radial velocity and intensity, thus

generating realistic radar data. Besides, we train gesture recognition models using both a large amount of generated and a small amount of real-world radar data to further improve their generalization in practical scenes.

4 DETAILED DESIGN OF G³R

Now we introduce the detailed design of G³R, which consists of *human parsing*, *skeleton extraction*, *depth prediction*, *gesture reflection point generator*, *signal simulation model*, and *encoder-decoder model*.

4.1 Human Parsing

In general, when a user performs gestures, the arm moves more than the rest of the human body, causing more reflections. Therefore, to obtain fine-grained gesture data, it is crucial to precisely parse human constituent parts. The goal of human parsing is to partition a human in 2D videos into different constituent parts. However, existing methods [33, 34] either rely on predefined human body hierarchies or accurate human postures, making it difficult to ensure generalization in scenes involving multiple humans or unexpected occlusions of human parts. Considering the hierarchical structure of the human body, each body part in an image can possess its unique position distribution characteristic. Therefore, we employ a state-of-the-art *human parsing* model (CDGNet) [35], which generates instance class distributions by accumulating raw human parsing labels in horizontal and vertical directions, thereby providing valuable supervisory information. By leveraging these horizontal and vertical class distribution labels, CDGNet is guided to mining the intrinsic position distribution of each class. Finally, CDGNet combines two guided features into a spatial guidance map, which is subsequently superimposed on the baseline network through multiplication and concatenation to accurately parse human body parts.

4.2 Skeleton Extraction

Based on the input 2D videos, we adopt a *skeleton extraction* model (RSN) [36] to extract the skeleton points of different human body parts. This process can avoid using highly complex human mesh models [37, 38]; the main reason is that they are difficult to extract fine-grained hand features. As shown in Fig. 8, we visualize the extraction result of a state-of-the-art human mesh model (BEV) [37]. When the hand posture is a fist, the output of the model is an open hand, making it difficult to accurately map gesture features. Therefore, we use RSN to extract the skeleton points of different human constituent parts. RSN efficiently aggregates features with the same spatial size to obtain refined local representations while using an attention mechanism to weigh the output local and global features, enabling accurate localization and extraction of skeleton points.

4.3 Depth Prediction

Given an input 2D video, we utilize a state-of-the-art *depth prediction* model (ZoeDepth) [39] to obtain the depth information of humans. ZoeDepth first pre-trains an encoder-decoder architecture that is trained using relative depth

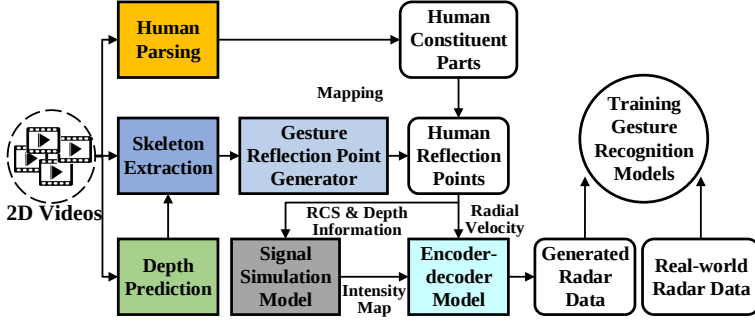


Fig. 7: System overview of G^3R .

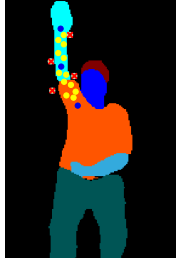


Fig. 9: Illustration of reflection point mapping. Blue, yellow, and crossed yellow dots represent the three initial skeleton points, randomly interpolated points, and overflow points, respectively.

information gathered from various datasets, and then adds domain-specific heads for metric bins module to the encoder-decoder architecture, followed by fine-tuning them on metric depth datasets. Moreover, in the inference process, ZoeDepth uses a classifier on encoder features to automatically route an image to the appropriate head for further improving the performance of depth estimation. In G^3R , considering the propagation characteristics of radar signals, the reflection points are mainly focused on the arm with a large movement amplitude. Therefore, we only need to accurately obtain the depth information of the arm. To this end, we average the depth values near the three skeleton points on the arm to further reduce the prediction error.

4.4 Gesture Reflection Point Generator

Based on the obtained depth information, we perform a coordinate transformation and concatenate it to the 2D skeleton points to form 3D skeleton points, characterizing the corresponding reflection points of the human body. However, there are 19 human skeleton points extracted with only 3 skeleton points on the arm, making it difficult to characterize the real number and distribution of radar reflection points for each gesture. Therefore, we design a *gesture reflection point generator*, which expands the reflection points of the arm based on skeleton points on the arm. Specifically, we randomly interpolate the skeleton points of the arm to expand the reflection points on it to realistically simulate the movement characteristics of the arm. Note that as the default configuration for the radar sensor sets the number of points collected per frame to 64, we uniformly expand the original 3 skeleton points to 64 reflection points. During the random interpolation process, we observe that there is a possibility of encountering reflection point overflow, which causes an incorrect correspondence between the reflection points and the arm. The main reason is that



Fig. 8: Illustration of hand mesh extraction.

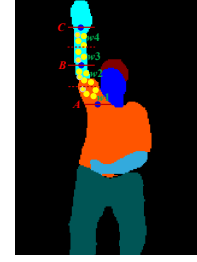


Fig. 10: Illustration of radial velocity of gesture reflection points.

the depth prediction will have some deviations with the movement of user gestures, resulting in significant errors for the interpolation points, as shown in Fig. 9. To solve this problem, we map human constituent parts partitioned by the *human parsing* model with the reflection points of the human body, followed by checking the interpolation points to remove these reflection points that do not belong to the arm. Note that for the removed points, we will also re-execute the interpolation to supplement.

4.5 Signal Simulation Model

Based on the obtained human reflection points, we can calculate their RCS and radial velocity with respect to a radar sensor. For RCS, we convert the obtained reflection points into surface triangles, and the normal of all triangles is also calculated during the conversion process. Based on the obtained surface area and normal, we can calculate the RCS corresponding to each reflection point. Meanwhile, we observe from the real-world radar data that the radial velocity of each gesture basically concentrates on a few values. Therefore, we perform a windowing strategy on the reflection points to calculate their radial velocities. As shown in Fig. 10, we can obtain their radial velocities based on the arm's three initial skeleton points (A , B , C). Note that the radial velocity of these 3 points can be obtained by looking back at the movement history (previous frames). Considering that during the movement of the arm, the speed of points A , B , and C increases in order. Based on the statistical results of real-world data, we divide the points on the arm into four windows (w) on average, where the velocity of points within $w1$ aligns with point A , while the velocity of points within $w2$ and $w3$ corresponds to point B . Similarly, the velocity of points within $w4$ corresponds to point C .

To simulate the multipath reflection and attenuation of radar signals, inspired by the ray tracing [40] in computer graphics, we design a *signal simulation model*. A typical

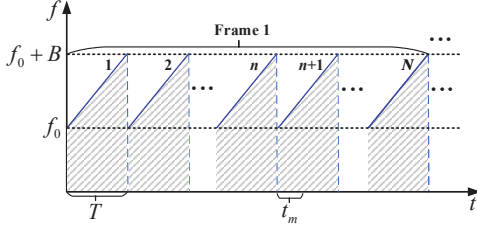


Fig. 11: Illustration of chirps in one frame. The shaded area represents the number of periods experienced since the starting of a frame.

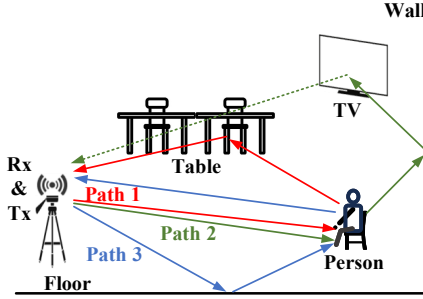


Fig. 12: Illustration of multipath reflection and attenuation of radar signals. Paths 1 and 3 represent the reflection process of radar signals among multiple reflectors. Path 2 represents an invalid radar signal due to signal attenuation.

application of ray tracing is to track the movement of visible light. Existing works [41, 42] have utilized the principle of multiple reflections of light to achieve object tracking and positioning. In addition, light propagation also involves common physical phenomena, i.e., reflection and attenuation, which are similar to the multipath reflection and attenuation of radar signals during the transmitting and receiving process. Therefore, the *signal simulation model* takes depth information and RCS as inputs to simulate the propagation characteristics of radar signals. Specifically, for each transmitting (Tx) and receiving (Rx) antenna pair, the signal formed by a radar is sinusoidal-like [43]. As shown in Fig. 11, to model it, we should consider the time t across multiple chirps. Suppose n chirps have been transmitted ($0 \leq n \leq N-1$, where N is the total number of chirps in a frame), and the current $(n+1)$ th chirp has been transmitted for t_n time. Therefore, we can calculate the time t and the instant frequency $f(t)$ at which the chirp is transmitted, i.e.,

$$t = nT + t_n, \text{ and } f(t) = f_0 + \frac{Bt_n}{T}, \quad (1)$$

where T represents the period of each chirp, B represents the sweep bandwidth. Based on this, we can mathematically express the radar emission signal $T_{Tx}(t) = Ae^{j\varphi}$, where A and φ represent amplitude and phase, respectively.

As shown in Fig. 11, we can calculate the shaded area under all chirps in a frame, i.e., $\int_0^{nT+t_n} f(x)dx$, which represents the number of periods experienced. The signal phase φ varies by 2π per period, which is calculated as follows:

$$\varphi = 2\pi \left(\int_0^{nT+t_n} f(x) \cdot dx \right) + \varphi_0 = 2\pi \left(f_0 + \frac{B(t_n^2 + nT^2)}{2T} \right) + \varphi_0, \quad (2)$$

where φ_0 represents initial phase. Then, $T_{Tx}(t)$ can be expressed as:

$$T_{Tx}(t) = Ae^{j2\pi \left(f_0 + \frac{B(t_n^2 + nT^2)}{2T} \right) + \varphi_0}. \quad (3)$$

The transmitted signal $T_{Tx}(t)$ will bounce back to the receiving antenna $R_{Rx}(t)$ at the reflection point. The received signal can be seen as a delayed version of the transmitted signal and the latency is τ . Therefore, $R_{Rx}(t)$ is expressed as:

$$R_{Rx}(t) = A' e^{j(2\pi(f_0(t-\tau) + \frac{B((t_n-\tau)^2 + nT^2)}{2T}) + \varphi_0)}, \quad (4)$$

where A' represents the attenuated amplitude, which can be calculated according to the radar communication principle [44]:

$$A' = \frac{G_{Tx}G_{Rx}\lambda\sqrt{PR}}{(4\pi)^{1.5}D^2}, \quad (5)$$

where $G_{Tx/Rx}$ represents antenna gains of Tx/Rx . λ , P , R , and D represent wavelength, transmission power, RCS, and the distance between reflection points and the Rx , respectively.

As shown in Fig. 12, to better simulate the propagation characteristics of radar signals in real-world scenes, we add some existing real reflectors, such as tables and TVs, when designing the *signal simulation model*. Specifically, the voxels we selected are cubes with 0.05 m on each side, and the 3D space we constructed is divided into a grid with dimensions of $128 \times 128 \times 64$, representing its length, width, and height, respectively. During the simulation process, we will incorporate some common reflectors into the grid within the 3D space, followed by calculating the average signal intensity of all triangles associated with each vertex to derive the vertex's signal intensity information. Finally, by averaging the signal intensity of all vertices within each voxel grid, we can obtain the signal intensity information corresponding to the reflector in the grid. Note that the position of reflectors can be freely adjusted to facilitate the simulation of diverse scenes; meanwhile, there will be a bias so that the model can better adapt to various scenes. Apart from the reflection, we should also consider the energy loss of radar signals during propagation. Existing works [6, 41] have also demonstrated that energy is essentially lost when a signal undergoes reflection more than three times. Therefore, when designing the *signal simulation model*, we set the maximum threshold for the signal attenuation coefficient (α) to 0.3. Moreover, we experimentally verify the impact of different α on the quality of generated radar data (see Section 5.3.5), where the interval is set to 0.05. Finally, in G^3R , α is set to 0.15, and the *signal simulation model* outputs the human intensity map.

4.6 Encoder-decoder Model

Based on the human intensity map and radial velocity, we can generate the initial radar data. However, such data seriously differs from real-world radar data in number and distribution. Therefore, an *encoder-decoder model* is designed to generate realistic radar data, which contains a *sampling module* and a *fitting module*. Since point clouds are essentially an unordered point set and lack topological information, convolutional neural networks are unsuitable for processing this data type. To this end, PointNet [45] is designed to process point cloud data without converting it to other data formats (e.g., voxel), but it ignores the geometric relationships between points, making it difficult to capture local features of gestures. DGCNN [46] incorporates a new

neural network module (EdgeConv) based on PointNet, which explicitly constructs a locally connected graph and learns the embedding of edges, supporting the capture of local features while maintaining permutation invariance. This network architecture can produce richer contextual information and better learn the semantic information of the point set by dynamically updating the graph structure between layers, followed by accurately finding the mapping relationship between the input point cloud and the ground truth.

Inspired by the DGCNN, as shown in Fig. 13, we design a *sampling module*. The module takes the x , y , and z coordinates of N points as inputs and calculates an edge feature set of size k for each point at an EdgeConv layer, where $N=1920$, $k=20$. After constructing the local connection graph, we use a multi-layer perceptron (MLP), whose parameters in parentheses are the number of output channels of the 2D convolutional layer. Note that a batch normalization (BN) and a leaky rectified linear unit (Leaky ReLU) are used after convolution. Shortcut connections are employed to incorporate the local features from all EdgeConv outputs to obtain (64×3) -dimensional point cloud features, followed by transforming them into 512-dimensional features by MLP. Meanwhile, max pooling and average pooling are performed on the obtained local features to obtain the global feature vector of point clouds $(1 \times (512 + 512) = 1024$ dimensions). Moreover, the module encodes the gesture labels and distances into 32-dimensional feature vectors, respectively, followed by concatenating them into the global vector to obtain a 1088-dimensional feature vector. Finally, we repeat the obtained feature vectors for N times to get the aggregated point-wise features, which are concatenated with local features (1920×1280) , followed by employing farthest point sampling (FPS) layers to obtain the generated M_{ij} points corresponding to different positions i and different gestures j .

To guarantee near-optimal convergence, we construct two loss functions to measure the correlation between generated and real-world point clouds in number and distribution of points. To this end, we first use the Chamfer Distance (ChD) to measure the number difference between generated and real-world point sets [47]. ChD is expressed as:

$$L_{ChD}(P_G, P_R) = \frac{1}{N_G} \sum_{x \in P_G} \min_{y \in P_R} \|x - y\|_2^2 + \frac{1}{N_R} \sum_{y \in P_R} \min_{x \in P_G} \|x - y\|_2^2, \quad (6)$$

where P_G and P_R represent generated and real-world point sets, respectively. N_G and N_R represent the number of points in P_G and P_R , respectively. However, relying solely on the ChD loss does not effectively guarantee the accurate distribution of points; the main reason is that ChD only measures the distance between the nearest points, which may fail to capture the distribution differences between points. Therefore, to accurately obtain the point distribution, we introduce an Earth Movers' Distance (EMD) in the loss function, which is expressed as:

$$L_{EMD}(P_G, P_R) = \frac{1}{N_G} \min_{\phi: P_G \rightarrow P_R} \sum_{x \in P_G} \|x - \phi(x)\|_2, \quad (7)$$

where $\phi: P_G \rightarrow P_R$ represents a bijection function to guarantee that every element of P_G is paired with precisely one element of P_R . Therefore, the final loss function is:

$$L = \lambda L_{ChD} + (1 - \lambda) L_{EMD}, \quad (8)$$

where $\lambda=0.5$, which is a hyper-parameter to balance the weight of different components.

In addition, considering the existing gesture recognition models will splice the radial velocity and intensity together to form a matrix similar to a grayscale map so that the model can better learn different gesture features. Therefore, we design a *fitting module*, which adopts a *U-Net* model with an encoder-decoder architecture to fit radial velocity and intensity between generated and real-world radar data for generating realistic radar data. As shown in Fig. 14, the *U-Net* has a total of 23 convolutional layers and about 1.91M parameters. The encoder uses double convolution blocks (3×3 kernel, stride=1, padding=1) for feature extraction. The decoder uses deconvolution layer (2×2 kernel, stride=2, padding=0) and skip-connection for feature fusion. A Leaky ReLU and a BN are used for each convolutional layer. We employ the corresponding pairs of generated and real-world matrices to train the *U-Net* model for 1000 epochs, using the *Adam optimizer* [48] with a learning rate of 0.001.

5 EVALUATION

5.1 Implementation and Experimental Setup

Implementation. The hardware platform for data collection has been implemented, as depicted in Fig. 15. The platform includes a tripod to simulate the sensor installation position and a Dell notebook (XPS13, i7-1250U CPU, 16GB memory) for data storage and processing. The notebook is connected to both a commercial *Frequency Modulated Continuous Wave* (FMCW) radar sensor (TI IWR1443BOOST) operating at 77 GHz and a global shutter camera (DOH805) for data collection via UART USB cables. By default, we employ the following FMCW parameters: idle time $7\mu s$, ramp end time $114.29\mu s$, range resolution 4 cm, Doppler resolution 0.34 m/s, maximum unambiguous range 8.19 m, frame duration 100 ms, and maximum radial velocity ± 2.67 m/s. In addition, we mount a camera next to the radar sensor to capture footage. The training of G^3R is performed on a server running on Ubuntu 20.08 OS with two RTX3080 GPUs. We implement different modules in G^3R using Python to facilitate integration with various radar-based sensing applications.

Dataset Preparation. *Dataset1.* We place the hardware platform on a tripod with a height of 1.7 m to collect real-world radar data. We recruit 32 volunteers (23 males, 9 females, i.e., users) aging from 21 to 30 with heights ranging from 158 cm to 186 cm and weights ranging from 45 kg to 98 kg. Users sit and perform different gestures at a linear distance of 0.5-4.5 m in front of the collection device, resulting in a total of 720 samples per user. Meanwhile, we utilize a camera to record user gestures and label the radar dataset. Note that we link the frame of the camera with the nearest radar frame for temporal alignment and exploit coordinate transformation to achieve spatial alignment. Each user performs 5 gestures (PL, PS, CR, UP, and KO) in front of the collecting device (see Section 2.1). Each

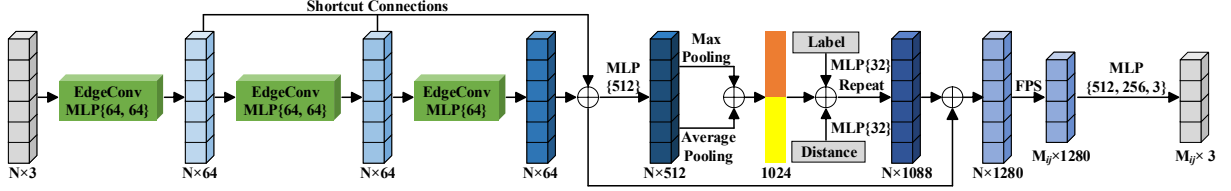


Fig. 13: The architecture of our *sampling* module.

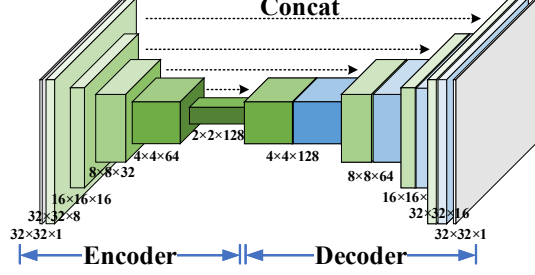


Fig. 14: The architecture of our *fitting* module.



Fig. 15: Hardware platform for data collection.

gesture takes about 1-2 s to complete. Thus, in total we collect roughly 9 hours of real-world radar data. For the collected dataset, we process them into point cloud data via 3D fast Fourier transform (FFT) for training gesture recognition models. Moreover, we aggregate 21 hours of 2D video data from five public datasets [15–19], which are structured with gesture labels, and unstructured available 2D video sources (YouTube, Bilibili) using queries related to our gesture set. These 2D videos serve as inputs of G^3R , enabling the generation of realistic radar data for model training. Note that when preparing the 2D video dataset, we manually eliminate some low-quality 2D videos (e.g., blurred samples).

Dataset2. To deeply evaluate the impact of various factors on recognition performance, we recruit 5 new users (3 males, 2 females) again. As shown in Fig. 16, we select three new scenes (meeting room, work room, and hotel room) for validation, where: (i) users perform 8 times per gesture in different postures (standing and sitting); (ii) users perform gestures in different positions (P1-P5) and the radar is also placed in different positions (R1 and R2). Thus, in total we collect $5 \text{ gestures} \times 5 \text{ users} \times 3 \text{ scenes} \times 8 \text{ times} \times 2 \text{ postures} \times 7 \text{ positions} = 8400 \text{ samples}$.

Baselines. We verify the effectiveness of G^3R by comparing the following three state-of-the-art methods:

- *Vid2Doppler* [5], a software pipeline that allows 2D videos to be transformed into radar data, which employs a neural network to directly extract human meshes, followed by generating signal reflection of each vertex with respect to a radar. Note that *Vid2Doppler*



(a) Meeting room (Scene 1)



(b) Work room (Scene 2)



(c) Hotel room (Scene 3)

Fig. 16: Three different scenes. $\star$ represents different radar positions (R1 and R2), and $\blacktriangledown$ represents various user positions (P1-P5).

ignores the multipath reflection and attenuation of radar signals while generating data that only represents the occupancy information of humans, resulting in a notable disparity from real-world radar data.

- *SynMotion* [7], a method for generating radar data using human skeleton points. It employs a *variant tracker* to obtain accurate skeletal point coordinates of humans, followed by generating radar signals reflected from these points. Note that *SynMotion* does not consider the multipath reflection and attenuation of radar signals during the transmitting and receiving process, significantly compromising the quality of generated radar data.
- *Midas* [6], a method for generating radar data using 2D videos. It exploits several key modules, *human region indexing*, *human mesh fitting*, and *multi-human reflection* model to simulate the multipath reflection and attenuation of radar signals, followed by using a *Transformer* model to generate convertible and realistic radar data for various human sensing tasks. Note that while *Midas* considers the multipath reflection and attenuation of radar signals, the point cloud data generated is coarse-grained and solely represents the occupancy informa-

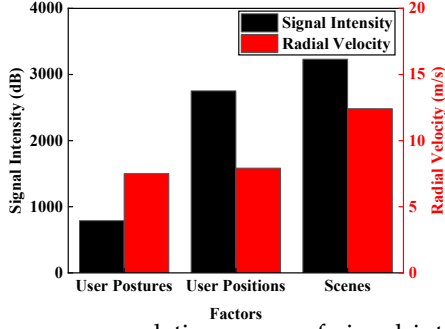


Fig. 17: Average cumulative errors of signal intensity and radial velocity for G^3R with various factors.

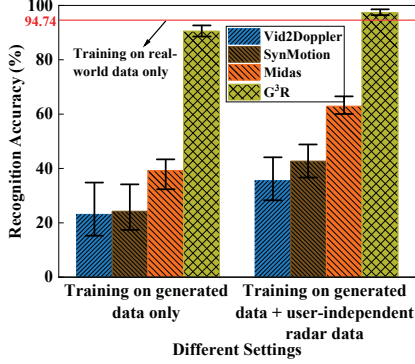


Fig. 18: Gesture recognition accuracy of different methods across different train/test settings.

tion of humans, making it difficult to apply for fine-grained gesture recognition.

Experimental Settings. To fairly verify the accuracy of G^3R and three baselines, we evaluate their performance on dataset1 using three different experimental settings: (i) training on all generated radar data only and testing on real-world radar data from users, (ii) training on four folds of real-world radar data and testing on one fold of real-world radar data (all combinations, results averaged), and (iii) training on all generated data and four folds of real-world data, testing on one fold of real-world data (all combinations, results averaged) (see Section 5.2). Note that all generated radar data refers to 21 hours of 2D videos, and the real-world data is approximately 9 hours of gesture samples collected. Meanwhile, we also use the first setting to conduct ablation experiments to demonstrate the effectiveness of G^3R (see Section 5.3). Moreover, to deeply evaluate the effectiveness of G^3R under various factors, we utilize the same training data from the first and third settings for model training, and use dataset2 for testing, further demonstrating the advantage of G^3R under various user postures, positions, and scenes (see Section 5.4). We use *mTransSee* [4] as the gesture recognition model.

Evaluation Metrics. To quantify the performance of G^3R and three baselines, the following evaluation metrics are defined:

- **Average Cumulative Error.** We run 2D videos through G^3R and compare the generated radar data to the real-world radar data by quantifying the difference values of signal intensity/radial velocity under various user postures, positions, and scenes, respectively, followed by averaging them.
- **Recognition Accuracy.** The recognition accuracy is de-

True Gestures	KO	0.00%	0.17%	0.17%	0.52%	99.13%
	UP	8.51%	5.73%	1.39%	83.33%	1.04%
	CR	0.17%	0.87%	96.53%	1.91%	0.52%
	PS	0.00%	97.57%	0.00%	2.26%	0.17%
	PL	97.22%	0.00%	0.17%	2.09%	0.52%
		PL	PS	CR	UP	KO
		Predicted Gestures				

Fig. 19: Gesture recognition accuracy using real-world radar data.

defined as the probability that the model [4] correctly recognizes a gesture.

- **Confusion Matrix.** Each row corresponds to the true gestures in the confusion matrix, while each column corresponds to the predicted gestures. The entry of the p -th row and the q -th column represents the recognition accuracy.

5.2 Evaluation on Gesture Recognition

5.2.1 Quality of Generated vs. Real-world Radar Data

To demonstrate the effectiveness of G^3R , we use our collected 2D videos and corresponding real-world radar data for verification. Specifically, we measure the average cumulative errors between generated and real-world radar data regarding signal intensity/radial velocity under various user postures, positions, and scenes. Fig. 17 shows that the errors are 789 dB/7.5 m/s, 2752 dB/7.92 m/s, and 3232 dB/12.41 m/s under three different factors, respectively. G^3R reduces the whole average cumulative errors of signal intensity/radial velocity by 10.65x/5.02x, 8.76x/4.63x, and 7.31x/3.86x compared with *Vid2Doppler*, *SynMotion*, and *Midas* (see Section 2.3), respectively, demonstrating that the radar data generated by G^3R is closer to real-world data.

5.2.2 Gesture Recognition Accuracy

As shown in Fig. 18, G^3R achieves 90.51% accuracy, 67.41 pp, 66.28 pp, and 51.36 pp higher than that of *Vid2Doppler*, *SynMotion*, and *Midas*, respectively, when using the first experimental setting, demonstrating the advantage of G^3R . The second setting achieves 94.76%. While a 4.25 pp difference exists in recognition accuracy, the model trained on all generated radar data can be comparable to a model trained solely on real-world radar data. Moreover, G^3R achieves the best average recognition accuracy, 97.32% under the third setting; meanwhile, it improves the accuracy by 61.80 pp, 54.67 pp, and 34.45 pp compared with *Vid2Doppler*, *SynMotion*, and *Midas*, respectively.

We then refine the recognition accuracy of each gesture under three different experimental settings. Fig. 20 shows that: (i) G^3R achieves an accuracy of over 90% for four gestures, and only one gesture is below 90%; (ii) the recognition accuracy is below 50% for each gesture in *Vid2Doppler*, *SynMotion*, and *Midas*; the main reason is that they are difficult to simulate diversified and fine-grained gesture reflection properties. Moreover, we refine the recognition accuracy of each gesture for the second setting, as shown

True Gestures	KO	52.44%	6.46%	17.34%	2.09%	21.66%
	UP	32.13%	27.76%	34.96%	1.49%	3.67%
	CR	64.21%	10.77%	12.86%	1.67%	10.49%
	PS	23.32%	46.51%	18.30%	0.88%	10.99%
	PL	33.01%	51.41%	14.60%	0.88%	0.09%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(a) Vid2Doppler

True Gestures	KO	14.17%	4.91%	34.25%	14.20%	32.47%
	UP	17.41%	16.13%	48.05%	16.41%	2.01%
	CR	24.12%	29.47%	23.98%	19.00%	3.43%
	PS	13.99%	28.83%	41.12%	12.46%	3.60%
	PL	19.50%	6.01%	62.31%	11.43%	0.75%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(b) SynMotion

True Gestures	KO	3.91%	7.07%	27.67%	12.27%	49.07%
	UP	28.67%	29.15%	10.07%	22.56%	9.55%
	CR	17.49%	12.42%	31.44%	14.64%	24.02%
	PS	36.03%	50.34%	5.93%	6.38%	1.32%
	PL	42.38%	36.34%	7.45%	11.27%	2.56%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(c) Midas

True Gestures	KO	0.23%	1.16%	0.23%	0.93%	97.45%
	UP	12.50%	2.31%	6.94%	77.31%	0.93%
	CR	0.69%	0.00%	90.28%	7.64%	1.39%
	PS	1.39%	95.37%	0.23%	3.01%	0.00%
	PL	92.13%	0.00%	0.69%	7.18%	0.00%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(d) G^3R

Fig. 20: Gesture recognition accuracy using all generated radar data.

in Fig. 19. We observe that the recognition accuracy of the model trained on generated data only closely approximates the performance of a model trained on real-world radar data. Similarly, we also refine the recognition accuracy of each gesture for the third setting. Fig. 21 shows that: (i) G^3R achieves an accuracy of over 93% for all gestures, and even approaches 100% accuracy for some gestures; (ii) G^3R improves the accuracy by 57.41 pp/48.79 pp/65.39 pp/74.10

True Gestures	KO	11.42%	6.06%	36.71%	9.21%	36.60%
	UP	22.97%	17.62%	29.65%	19.77%	10.00%
	CR	22.27%	25.35%	30.99%	2.09%	19.30%
	PS	16.41%	49.53%	16.06%	16.71%	1.29%
	PL	40.78%	19.61%	23.84%	10.21%	5.57%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(a) Vid2Doppler

True Gestures	KO	3.10%	1.43%	33.09%	18.78%	43.59%
	UP	24.33%	12.83%	26.22%	31.15%	5.47%
	CR	14.35%	11.00%	40.89%	15.82%	17.94%
	PS	8.89%	54.38%	18.55%	10.42%	7.76%
	PL	43.19%	5.24%	27.04%	23.23%	1.30%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(b) SynMotion

True Gestures	KO	5.24%	1.69%	6.53%	16.90%	69.64%
	UP	32.27%	6.69%	14.30%	44.13%	2.62%
	CR	13.49%	1.40%	61.51%	15.17%	8.43%
	PS	2.63%	69.10%	9.58%	17.93%	0.76%
	PL	70.01%	0.35%	4.35%	24.88%	0.41%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(c) Midas

True Gestures	KO	0.00%	0.00%	0.00%	0.14%	99.86%
	UP	2.51%	2.65%	0.84%	93.87%	0.14%
	CR	0.14%	0.14%	96.38%	3.34%	0.00%
	PS	0.14%	98.32%	0.14%	1.40%	0.00%
	PL	98.19%	0.28%	0.14%	1.39%	0.00%
		Predicted Gestures				
		PL	PS	CR	UP	KO

(d) G^3R

Fig. 21: Gesture recognition accuracy using all generated data + user-independent radar data.

pp/63.26 pp, 55.00 pp/43.94 pp/55.49 pp/62.72 pp/56.27 pp, and 28.18 pp/29.22 pp/34.87 pp/49.74 pp/30.22 pp for PL/PS/CR/UP/KO gestures compared with *Vid2Doppler*, *SynMotion*, and *Midas*, respectively, demonstrating the advantage of G^3R in generating fine-grained radar data. Furthermore, Fig. 22 shows the detailed precision-recall curves under three experimental settings, demonstrating that G^3R is a promising system to achieve higher accuracy for gesture

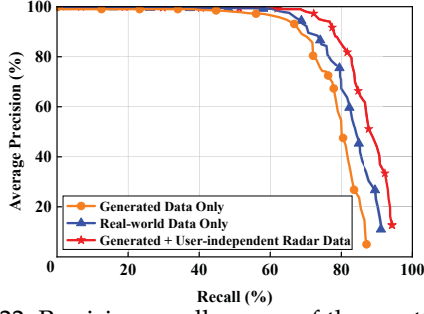
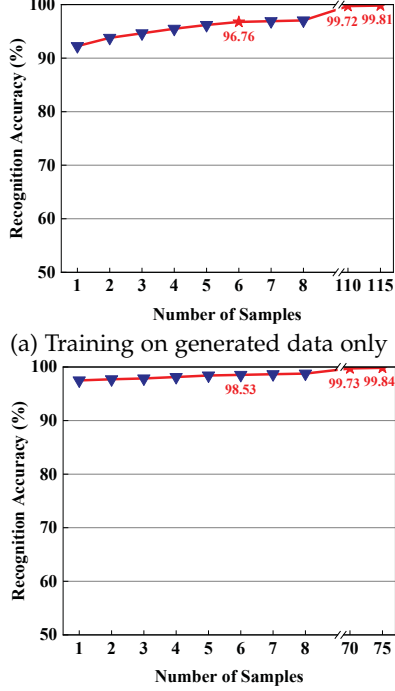


Fig. 22: Precision-recall curves of three settings.



(b) Training on generated data + user-independent radar data
Fig. 23: Gesture recognition accuracy changes with the increase of real-world samples from new users.

recognition.

Note that in the above three settings, the users in the test set are entirely excluded from the model's training process. However, it is common for gesture recognition systems to collect some real-world radar data from new users. Therefore, we verify the first and third settings respectively, and continuously add samples from new users to train the model. From Fig. 23(a) we observe that once the number of new user samples reaches 6, the recognition accuracy (96.76%, solely training with generated data) exceeds the claimed result of the existing work, *mTransSee* (96.7%, training with large-scale real-world data, at least 8 new user samples) [4]. Meanwhile, when users utilize the gesture recognition system, the corresponding radar data can be stored to iteratively update the model, thus achieving more accurate gesture recognition. Once the number of samples reaches 115, the recognition accuracy is infinitely close to 100%, demonstrating the effectiveness of G^3R . Note that stored radar data can be automatically annotated according to the recognition results. Moreover, Fig. 23(b) shows that when adding 6 samples, the recognition accuracy can reach 98.53% under the third setting; meanwhile, as new user samples continue to increase, the recognition accuracy can

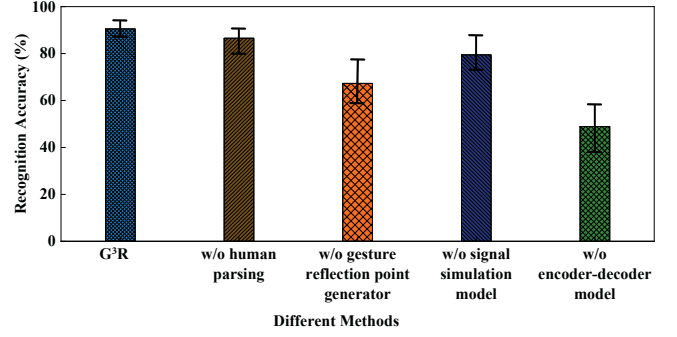


Fig. 24: Impact of individual modules on the recognition accuracy.

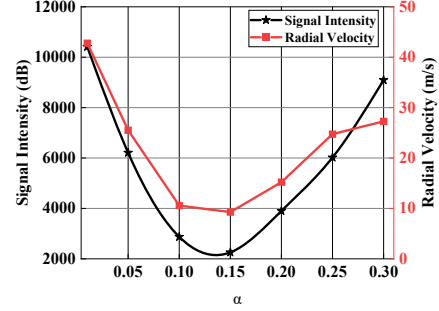


Fig. 25: Average cumulative errors of signal intensity and radial velocity with various attenuation coefficients.

quickly approach 100%, further demonstrating that G^3R provides data support for improving the performance of gesture recognition.

5.3 Ablation Study

5.3.1 Impact of the Human Parsing

To evaluate the contribution of the *human parsing* module, we do not parse and map human constituent parts. Fig. 24 shows that G^3R improves the recognition accuracy by 3.97 pp compared to G^3R w/o *human parsing* module. The results show that *human parsing* module can effectively filter out overflow points and guarantee accurate mapping of reflection points to enhance the quality of generated radar data.

5.3.2 Impact of the Gesture Reflection Point Generator

We then evaluate the benefits of the *gesture reflection point generator* module. Fig. 24 shows that G^3R improves the recognition accuracy by 23.22 pp compared to G^3R w/o *gesture reflection point generator* module. The main reason is that there are few reflection points on the arm, making it difficult to characterize fine-grained gesture features. If the reflective points on the arm are not expanded, it will make the generated radar data tend to the reflection properties of the whole human body, which makes it difficult to simulate the fine-grained gesture data.

5.3.3 Impact of the Signal Simulation Model

To evaluate the benefits of the *signal simulation model*, we directly use the calculated RCS and radial velocity. From Fig. 24 we observe that G^3R improves the recognition accuracy by 11.04 pp compared to G^3R w/o *signal simulation model*. The results show that *signal simulation model* can effectively

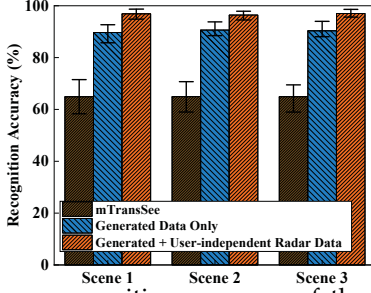


Fig. 26: Gesture recognition accuracy of three settings for different user postures under three different scenes.

simulate the multipath reflection and attenuation of radar signals during transmission to improve the quality of generated radar data.

5.3.4 Impact of the Encoder-decoder Model

We also conduct extensive experiments to verify the necessity of the *encoder-decoder model*. We directly concatenate human intensity map and radial velocity to obtain the generated radar data without performing any fitting operation. Fig. 24 shows that G^3R improves the recognition accuracy by 41.49 pp compared to G^3R w/o *encoder-decoder model*. The results show that *encoder-decoder model* can effectively simulate real-world data characteristics and improve the fidelity of generated radar data.

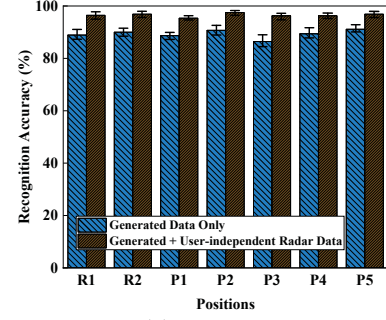
5.3.5 Impact of the Attenuation Coefficient

Following the propagation characteristics of radar signals, we vary the attenuation coefficient within the range of 0 to 0.3 to evaluate its impact on the generated radar data. From Fig. 25 we observe that average cumulative errors of signal intensity and radial velocity almost reach the minimum when $\alpha = 0.15$. Meanwhile, we find that: (i) when α is too small, the signal that should be lost reaches the receiver, resulting in excessive noise; (ii) when α is too large, the signal that should be retained does not reach the receiver, making the generated radar data too sparse.

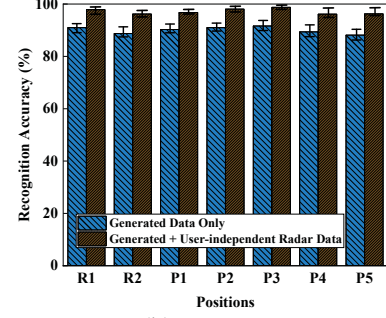
5.4 In-depth Evaluation on Impact of Various Factors

5.4.1 Impact of User Postures

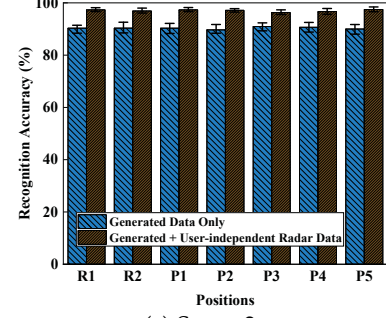
To evaluate the performance of G^3R for different postures, we conduct extensive experiments under the first and third settings; meanwhile, we also test the performance of *mTransSee* [4] that leverages a transfer learning strategy to learn the knowledge of large-scale users who perform gestures while standing for improving generalization. From Fig. 26 we observe that: (i) *mTransSee* only achieves an average recognition accuracy of 64.93% for standing and sitting postures across three different scenes; the main reason is that different user postures would cause differences in radar data due to the reflection properties of signals; (ii) G^3R achieves an average recognition accuracy of 90.22% for standing and sitting postures across three different scenes under the first setting, still maintaining a similar accuracy (90.51%, see Section 5.2.2); meanwhile, it improves the average recognition accuracy by 25.29 pp compared with *mTransSee*; the main reason is that the generated radar data covers more various user postures, enabling the model to learn more knowledge; (iii) G^3R achieves the best average recognition accuracy,



(a) Scene 1



(b) Scene 2



(c) Scene 3

Fig. 27: Gesture recognition accuracy of different settings for various positions under three different scenes.

96.78% under the third setting; meanwhile, it improves the accuracy by 31.85 pp compared with *mTransSee*, demonstrating the advantage of G^3R under different postures.

5.4.2 Impact of User and Radar Positions

We then evaluate the impact of user and radar positions on recognition performance under different scenes. We select 5 different user positions and 2 different radar positions for testing to verify the effectiveness of G^3R under the first and third settings. Fig. 27 shows that: (i) G^3R achieves an average recognition accuracy of 89.31%/90.02%/90.33% and 96.50%/97.14%/97.02% across all positions under scene 1/scene 2/scene 3 under the first and third settings, respectively, demonstrating that G^3R can fully adapt to changes in user and radar positions; (ii) the recognition accuracy will experience slight fluctuations in different user and radar positions; the main reason is that changes in radar's heights and user's distances will have a certain impact on the quality of generated radar data.

5.4.3 Impact of Different Scenes

We measure the overall average recognition accuracy under three different scenes. Fig. 28 shows that: (i) G^3R achieves an average accuracy of 90.06% and 96.99% across various

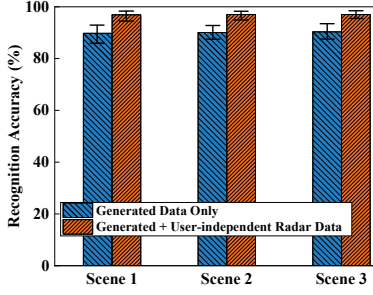


Fig. 28: Gesture recognition accuracy of two settings under three different scenes.

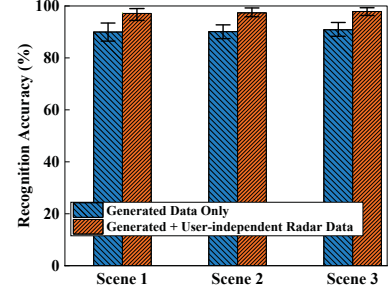


Fig. 30: Gesture recognition accuracy of two settings for two different gestures under three different scenes.

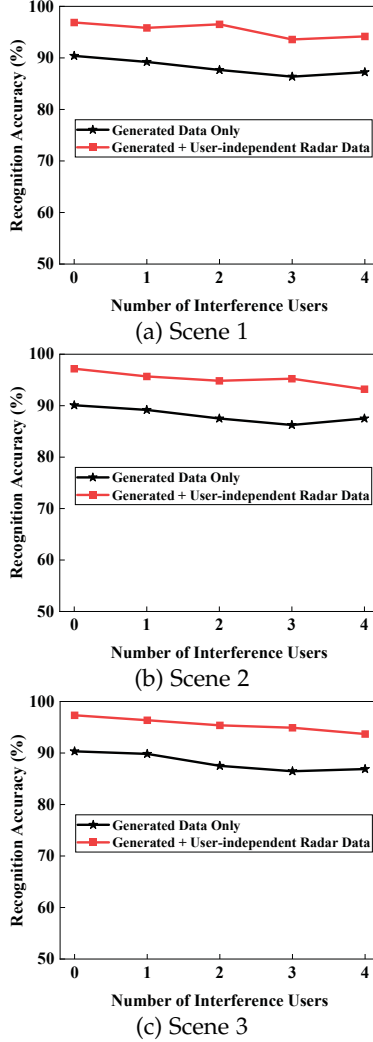


Fig. 29: Gesture recognition accuracy of different settings for multi-user coexistence under three different scenes.

factors under the first and third settings, respectively, still achieving a similar accuracy (90.51% and 97.32%, see Section 5.2.2); (ii) the model's accuracy remains relatively stable across three scenes, with a maximum difference of only 0.58 pp and 0.11 pp under the first and third settings, respectively, demonstrating that G^3R is a prospective system to achieve generalized gesture recognition.

5.4.4 Impact of Multi-user Coexistence

We further evaluate the recognition performance of G^3R under multi-user coexistence scenes. Specifically, we allow the interference users (1-4 persons) to freely move within

the radar sensing range while a target user performs corresponding gestures. Meanwhile, we ask the target user to perform each gesture in turn in the case of 1-4 interference users (5 gestures $\times$ 8 times $\times$ 2 postures $\times$ 3 positions $\times$ 4 interference cases $\times$ 3 scenes = 2880 samples in total), followed by testing the recognition accuracy. Similarly, we also adopt the first and third settings for evaluation. From Fig. 29 we observe that: (i) the recognition accuracy only drops by an average of 2.55 pp and 2.18 pp when there are 1-4 interference users across three scenes under the first and third settings, respectively; (ii) as the number of interference users increases, the recognition accuracy slightly jitters; the main reason is that interference users may perform some activities close to gestures during free movement; (iii) even if 5 users coexist, the recognition accuracy still surpasses 86% and 93% across three scenes under the first and third settings, respectively, demonstrating the effectiveness of G^3R .

5.4.5 Impact of Different Gestures

We further evaluate the recognition performance of G^3R under different user gestures. Specifically, we additionally collect two different gesture data, i.e., users waving from left (right) to right (left), to verify the generalization of G^3R . Thus, in total we collect 5 users $\times$ 2 gestures $\times$ 3 scenes $\times$ 8 times $\times$ 2 postures $\times$ 3 positions = 1440 samples. Fig. 30 shows that: (i) G^3R achieves an average accuracy of 90.32% and 97.45% across various factors under the first and third settings, respectively, still achieving an accuracy comparable to other gestures (90.51% and 97.32%, see Section 5.2.2); (ii) the model's accuracy remains relatively stable across three scenes, with a maximum difference of only 0.87 pp and 0.76 pp under the first and third settings, respectively, demonstrating that G^3R is a prospective system to achieve generalized recognition for arbitrary gestures.

6 RELATED WORK

Contact-based Gesture Recognition. Some early works on gesture recognition mainly utilize wearable sensors (such as MoCap [49, 50], IMUs [51, 52], and RFID tags [53, 54]). STMT [49] employs a novel hierarchical transformer architecture that encodes intrinsic and extrinsic representations, incorporating both intra- and inter-frame attention to enhance spatial-temporal mesh modeling to better distinguish between nuanced gestures. GPOGR [51] recognizes user gestures by extracting motion information from IMU data. RF-Dial [53] attaches two antennas and two RFID tags to users, followed by using translation and rotation to track their trajectories to perform gesture recognition. However,

these methods rely on physical contact, necessitating users to wear or attach sensors or RFID tags, significantly impacting user experience. In contrast, we focus on accurate gesture recognition using contactless radar.

Contactless Gesture Recognition. In recent years, contactless gesture recognition is gaining traction due to the fact that it does not require attaching sensors/tags to users. These sensors can be sound [55, 56], RGB camera [57], depth camera [58], WiFi [59, 60], and radar [4]. SpeakerGesture [55] adopts smart speakers to realize room-scale gesture recognition without any hardware deployment. AO-Finger [55] proposes a fast fine-grained gesture recognition system based on acoustic-optic sensor fusion. However, sound-based gesture recognition methods are easily affected by environmental noise while involving user privacy. FDT [57] enables fast recognition by detecting and segmenting user gesture features in 2D videos. ICRS [58] tracks gesture movements through 3D information to achieve precise human-computer interaction. However, vision-based gesture recognition methods will face problems of user privacy leakage, environmental lighting, and non-line-of-sight conditions. Compared to vision and sound technologies, WiFi technologies have a larger operating area and can work in environments with poor illumination. Widar3.0 [59] extracts domain independent features of gestures to explore a zero-effort cross-domain recognition system. However, it is difficult for WiFi to capture fine-grained gesture features due to the larger wavelength; meanwhile, existing WiFi-based gesture recognition methods [60, 61] cannot adapt well to changes in user positions and scenes. Recently, mTransSee [4] deploys a radar to collect data, followed by utilizing transfer learning to adapt to environmental changes to support accurate gesture recognition. A commonality in previous work is the need for a large amount of real-world data to train gesture recognition models with the changes in user postures, user positions, and scenes. However, collecting and labeling data is a time-consuming and labor-intensive task.

Generating Radar Data. Some works have generated multiple types of sensing data, such as IMU data [20, 21, 23], sound data [22], depth camera data [62], and radar data [6, 7, 25], for human sensing models using other data, such as 2D videos, MoCap data, and animated 3D models. The idea of generating radar data using other data sources is not new. In particular, DRM [25] and FRS [27] use MoCap and depth camera data to generate Doppler signals for activity recognition, respectively. However, either the generated radar data is coarse or they are missing some common gestures. Moreover, some works [63, 64] utilize GANs to augment existing radar datasets, but these methods may generate confusing radar data when users perform similar gestures. Recently, Vid2Doppler [5] employs a neural network to directly extract human meshes from 2D videos, followed by generating signal reflection of each vertex to output Doppler radar data. SynMotion [7] adopts a signal synthesizer to emulate the radar sensing procedure during the transmitting and receiving process of radar chirps, followed by using a signature generator to generate Doppler signals. Similarly, Midas [6] further simulates multipath reflection and attenuation of radar signals to generate realistic radar data. Although Midas can generate some points

to characterize the occupancy information of users, these points are randomly selected and coarse-grained, which cannot accurately characterize the spatial features of fine-grained gestures. In contrast, G^3R is designed for generalized gesture recognition, which addresses the challenge of simulating diversified and fine-grained reflection properties of user gestures.

7 DISCUSSION

Generality. Our G^3R is a unique software pipeline that converts wealthy 2D videos into realistic radar data to address the gap of data limitation for training a generalized gesture recognition across various user postures, positions, and scenes. The results show that training the model with the generated radar data can achieve comparable performance in gesture recognition as training the model with real-world radar data. Meanwhile, G^3R guarantees the generalization of the gesture recognition model, enabling it to effectively adapt to changes in user postures, user positions, and scenes. If the video data only focuses on the gesture part in a near distance rather than the whole body, G^3R can still generate large-scale radar data by simply replacing the human parsing model with a hand detection model, further demonstrating the advantages of modular design. Moreover, G^3R is compatible with different hardware and operating systems, and the gesture recognition model has a low complexity, occupying only 20 M of memory, which makes it effortless to deploy on end devices with weak computing power. Besides, with advances in *human parsing*, *skeleton extraction*, and *depth prediction* techniques, it is possible to generate higher-quality radar data.

Limitations. There exist several key technical limitations that need to be addressed. *First*, G^3R has not yet attempted to generate radar data beyond distances of 4.5 m; the main reason is that current test distances universally satisfy the needs of normal households. For longer distance scenes, the performance of G^3R could experience a decrease. *Second*, G^3R cannot yet recognize who a user gesture corresponds to in different positions, but it is possible to achieve the correspondences by exploiting user behavioral features. *Third*, G^3R may fail when a user or the radar is in motion, but it is possible to recognize fine-grained gestures by leveraging radial velocity and signal intensity to partition the reflection points of different human body parts. *Fourth*, some postures (e.g., users perform gestures while lying down) have fewer 2D video sources, it is possible to address it by employing the latest AI-generated content (AIGC) model [65], thereby enhancing the capabilities of G^3R . *Fifth*, for unseen gestures (videos), G^3R may fail to generate realistic radar data, but it is possible to generate rich and fine-grained radar data using few-shot domain adaptation. *Sixth*, although the radar data generated by G^3R protects user privacy, recording gesture data in itself may bring privacy concerns. This issue is a long-standing research topic in the sensing computing community, and radar sensors will also face unique scrutiny due to their increasing popularity.

8 CONCLUSION

In this work, we design a software pipeline that exploits wealthy 2D videos to generate rich and fine-grained radar

data to support research for gesture recognition. In G^3R , a *gesture reflection point generator* expands reflection points of the arm; a *signal simulation model* simulates the multipath reflection and attenuation of radar signals to output the human intensity map; an *encoder-decoder model* combines a *sampling module* and a *fitting module* to generate realistic radar data. We implement and evaluate G^3R for gesture recognition using 2D videos from five public datasets, YouTube, and Bilibili, as well as self-collected real-world radar data from 32 volunteers. The results demonstrate that G^3R achieves 90.51% accuracy when training with all generated radar data; if we add a small amount of real-world radar data, G^3R can achieve 97.32% accuracy. Moreover, when facing a new user, the model only requires 6 samples per gesture to achieve 96.76% and 98.53% accuracy when training with all generated radar data and all generated radar data with a small amount of real-world radar data, respectively. Besides, based on the above two training settings, G^3R achieves 90.06% and 96.99% accuracy across various user postures, positions using self-collected real-world radar data from 5 volunteers, respectively.

ACKNOWLEDGEMENT

The work is supported by the Innovation Research Group Project of NSFC (61921003), in part by the National Natural Science Foundation of China under No. 62222202, No. 62232004, Beijing Natural Science Foundation (L223002), and in part by the 111 Project under No. B18008.

REFERENCES

- [1] K. Deng, D. Zhao, Q. Han, S. Wang, Z. Zhang, A. Zhou, and H. Ma, "Geryon: Edge assisted real-time and robust object detection on drones via mmwave radar and camera fusion," *Proc. of ACM IMWUT*, vol. 6, no. 3, pp. 1–27, 2022.
- [2] K. Deng, D. Zhao, Q. Han, Z. Zhang, S. Wang, and H. Ma, "Global-local feature enhancement network for robust object detection using mmwave radar and camera," in *Proc. of IEEE ICASSP*, 2022, pp. 4708–4712.
- [3] H. Liu, Y. Wang, A. Zhou, H. He, W. Wang, K. Wang, P. Pan, Y. Lu, L. Liu, and H. Ma, "Real-time arm gesture recognition in smart home scenarios via millimeter wave sensing," *Proc. of ACM IMWUT*, vol. 4, no. 4, pp. 1–28, 2020.
- [4] H. Liu, K. Cui, K. Hu, Y. Wang, A. Zhou, L. Liu, and H. Ma, "Mtranssee: Enabling environment-independent mmwave sensing based gesture recognition via transfer learning," *Proc. of ACM IMWUT*, vol. 6, no. 1, pp. 1–28, 2022.
- [5] K. Ahuja, Y. Jiang, M. Goel, and C. Harrison, "Vid2doppler: Synthesizing doppler radar data from videos for training privacy-preserving activity recognition," in *Proc. of ACM CHI*, 2021, pp. 1–10.
- [6] K. Deng, D. Zhao, Q. Han, Z. Zhang, S. Wang, A. Zhou, and H. Ma, "Midas: Generating mmwave radar data from videos for training pervasive and privacy-preserving human sensing tasks," *Proc. of ACM IMWUT*, vol. 7, no. 1, pp. 1–26, 2023.
- [7] X. Zhang, Z. Li, and J. Zhang, "Synthesized millimeter-waves for human motion sensing," in *Proc. of ACM SenSys*, 2022, pp. 377–390.
- [8] H. Liu, A. Zhou, Z. Dong, Y. Sun, J. Zhang, L. Liu, H. Ma, J. Liu, and N. Yang, "M-gesture: Person-independent real-time in-air gesture recognition using commodity millimeter wave radar," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3397–3415, 2021.
- [9] S. Palipana, D. Salami, L. A. Leiva, and S. Sigg, "Pantomime: Mid-air gesture recognition with sparse millimeter-wave radar point clouds," *Proc. of ACM IMWUT*, vol. 5, no. 1, pp. 1–27, 2021.
- [10] X. Shen, H. Zheng, X. Feng, and J. Hu, "MI-hgr-net: A meta-learning network for fmcw radar based hand gesture recognition," *IEEE Sensors Journal*, vol. 22, no. 11, pp. 10 808–10 817, 2022.
- [11] K. Ling, H. Dai, Y. Liu, A. X. Liu, W. Wang, and Q. Gu, "Ultragesture: Fine-grained gesture sensing and recognition," *IEEE Transactions on Mobile Computing*, vol. 21, no. 7, pp. 2620–2636, 2020.
- [12] A. Waghmare, Y. Ben Taleb, I. Chatterjee, A. Narendra, and S. Patel, "Z-ring: Single-point bio-impedance sensing for gesture, touch, object and user recognition," in *Proc. of ACM CHI*, 2023, pp. 1–18.
- [13] Z. Xia and F. Xu, "Time-space dimension reduction of millimeter-wave radar point-clouds for smart-home hand-gesture recognition," *IEEE Sensors Journal*, vol. 22, no. 5, pp. 4425–4437, 2022.
- [14] W. Chen, S. Lin, E. Thompson, and J. Stankovic, "Sensecollect: We need efficient ways to collect on-body sensor-based human activity data!" *Proc. of ACM IMWUT*, vol. 5, no. 3, pp. 1–27, 2021.
- [15] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *Proc. of IEEE ICCV*, 2011, pp. 2556–2563.
- [16] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arXiv preprint arXiv:1609.08675*, 2016.
- [17] A. G. Perera, Y. Wei Law, and J. Chahl, "Uav-gesture: A dataset for uav control and gesture recognition," in *Proc. of ECCV Workshops*, 2018, pp. 0–0.
- [18] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.
- [19] H. J. Escalante, V. Ponce-López, J. Wan, M. A. Riegler, B. Chen, A. Clapés, S. Escalera, I. Guyon, X. Baró, P. Halvorsen *et al.*, "Chalearn joint contest on multimedia challenges beyond visual analysis: An overview," in *Proc. of IEEE ICPR*, 2016, pp. 67–73.
- [20] H. Kwon, C. Tong, H. Haresamudram, Y. Gao, G. D. Abowd, N. D. Lane, and T. Ploetz, "Imutube: Automatic extraction of virtual on-body accelerometry from video for human activity recognition," *Proc. of ACM IMWUT*, vol. 4, no. 3, pp. 1–29, 2020.
- [21] H. Kwon, B. Wang, G. D. Abowd, and T. Plötz, "Approaching the real-world: Supporting activity recognition training with virtual imu data," *Proc. of ACM IMWUT*, vol. 5, no. 3, pp. 1–32, 2021.
- [22] D. Liang and E. Thomaz, "Audio-based activities of

- daily living (adl) recognition with large-scale acoustic embeddings from online videos," *Proc. of ACM IMWUT*, vol. 3, no. 1, pp. 1–18, 2019.
- [23] P. S. Santhalingam, P. Pathak, H. Rangwala, and J. Kosecka, "Synthetic smartwatch imu data generation from in-the-wild asl videos," *Proc. of ACM IMWUT*, vol. 7, no. 2, pp. 1–34, 2023.
- [24] Y. Lin and J. Le Kernec, "Performance analysis of classification algorithms for activity recognition using micro-doppler feature," in *Proc. of IEEE CIS*, 2017, pp. 480–483.
- [25] M. S. Seyfioglu, B. Erol, S. Z. Gurbuz, and M. G. Amin, "Diversified radar micro-doppler simulations as training data for deep residual neural networks," in *Proc. of IEEE radarConf*, 2018, pp. 0612–0617.
- [26] B. Erol and S. Z. Gurbuz, "A kinect-based human micro-doppler simulator," *IEEE AESM*, vol. 30, no. 5, pp. 6–17, 2015.
- [27] J. Li, A. Shrestha, J. Le Kernec, and F. Fioranelli, "From kinect skeleton data to hand gesture recognition with radar," *The Journal of Engineering*, vol. 2019, no. 20, pp. 6914–6919, 2019.
- [28] B. Erol, S. Z. Gurbuz, and M. G. Amin, "Gan-based synthetic radar micro-doppler augmentations for improved human activity recognition," in *Proc. of IEEE radarConf*, 2019, pp. 1–5.
- [29] M. M. Rahman, S. Z. Gurbuz, and M. G. Amin, "Physics-aware design of multi-branch gan for human rf micro-doppler signature synthesis," in *Proc. of IEEE radarConf*, 2021, pp. 1–6.
- [30] H. Rohling, "Radar cfar thresholding in clutter and multiple target situations," *IEEE TAES*, no. 4, pp. 608–621, 1983.
- [31] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. of ACM SIGKDD*, vol. 96, no. 34, 1996, pp. 226–231.
- [32] Statista. (2020, February) Hours of video uploaded to youtube every minute as of february. [Online]. Available: <https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute/>
- [33] X. Zhang, Y. Chen, B. Zhu, J. Wang, and M. Tang, "Blended grammar network for human parsing," in *Proc. of ECCV*, 2020, pp. 189–205.
- [34] X. Zhang, Y. Chen, M. Tang, J. Wang, X. Zhu, and Z. Lei, "Human parsing with part-aware relation modeling," *IEEE TMM*, 2022.
- [35] K. Liu, O. Choi, J. Wang, and W. Hwang, "Cdgnnet: Class distribution guided network for human parsing," in *Proc. of IEEE CVPR*, 2022, pp. 4473–4482.
- [36] Y. Cai, Z. Wang, Z. Luo, B. Yin, A. Du, H. Wang, X. Zhang, X. Zhou, E. Zhou, and J. Sun, "Learning delicate local representations for multi-person pose estimation," in *Proc. of ECCV*, 2020, pp. 455–472.
- [37] Y. Sun, W. Liu, Q. Bao, Y. Fu, T. Mei, and M. J. Black, "Putting people in their place: Monocular regression of 3d people in depth," in *Proc. of IEEE CVPR*, 2022, pp. 13 243–13 252.
- [38] S. Guan, J. Xu, M. Z. He, Y. Wang, B. Ni, and X. Yang, "Out-of-domain human mesh reconstruction via dynamic bilevel online adaptation," *IEEE TPAMI*, vol. 45, no. 4, pp. 5070–5086, 2022.
- [39] S. F. Bhat, R. Birkel, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," *arXiv preprint arXiv:2302.12288*, 2023.
- [40] T. Whitted, "An improved illumination model for shaded display," in *ACM Siggraph 2005 Courses*, 2005, pp. 4–es.
- [41] S. Yue, H. He, P. Cao, K. Zha, M. Koizumi, and D. Katabi, "Cornerradar: Rf-based indoor localization around corners," *Proc. of ACM IMWUT*, vol. 6, no. 1, pp. 1–24, 2022.
- [42] N. Scheiner, F. Kraus, F. Wei, B. Phan, F. Mannan, N. Appenrodt, W. Ritter, J. Dickmann, K. Dietmayer, B. Sick *et al.*, "Seeing around street corners: Non-line-of-sight detection and tracking in-the-wild using doppler radar," in *Proc. of IEEE CVPR*, 2020, pp. 2068–2077.
- [43] H. Ling, R.-C. Chou, and S.-W. Lee, "Shooting and bouncing rays: Calculating the rcs of an arbitrarily shaped cavity," *IEEE TAP*, vol. 37, no. 2, pp. 194–205, 1989.
- [44] S. Rao, "Introduction to mmwave sensing: Fmcw radars," *Texas Instruments (TI) mmWave Training Series*, pp. 1–11, 2017.
- [45] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. of IEEE CVPR*, 2017, pp. 652–660.
- [46] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM TOG*, vol. 38, no. 5, pp. 1–12, 2019.
- [47] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proc. of IEEE CVPR*, 2017, pp. 605–613.
- [48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of ICLR*, 2015.
- [49] X. Zhu, P.-Y. Huang, J. Liang, C. M. de Melo, and A. G. Hauptmann, "Stmt: A spatial-temporal mesh transformer for mocap-based action recognition," in *Proc. of IEEE CVPR*, 2023, pp. 1526–1536.
- [50] M. Barnachon, S. Bouakaz, B. Boufama, and E. Guillou, "Ongoing human action recognition with motion capture," *Pattern Recognition*, vol. 47, no. 1, pp. 238–247, 2014.
- [51] V. Villani, C. Secchi, M. Lippi, and L. Sabatini, "A general pipeline for online gesture recognition in human-robot interaction," *IEEE Transactions on Human-Machine Systems*, vol. 53, no. 2, pp. 315–324, 2023.
- [52] A. Sharma, C. Salchow-Hömmen, V. S. Mollyn, A. S. Nittala, M. A. Hedderich, M. Koelle, T. Seel, and J. Steimle, "Sparseimu: Computational design of sparse imu layouts for sensing fine-grained finger microgestures," *ACM TOCHI*, vol. 30, no. 3, pp. 1–40, 2023.
- [53] Y. Bu, L. Xie, Y. Gong, C. Wang, L. Yang, J. Liu, and S. Lu, "Rf-dial: An rfid-based 2d human-computer interaction via tag array," in *Proc. of IEEE INFOCOM*, 2018, pp. 837–845.
- [54] H. Li, C. Ye, and A. P. Sample, "Idsense: A human object interaction detection system based on passive uhf rfid," in *Proc. of ACM CHI*, 2015, pp. 2555–2564.

- [55] D. Li, J. Liu, S. I. Lee, and J. Xiong, "Room-scale hand gesture recognition using smart speakers," in *Proc. of ACM SenSys*, 2022, pp. 462–475.
- [56] C. Xu, B. Zhou, G. Krishnan, and S. Nayar, "Ao-finger: Hands-free fine-grained finger gesture recognition via acoustic-optic sensor fusing," in *Proc. of ACM CHI*, 2023, pp. 1–14.
- [57] S. Mukherjee, S. A. Ahmed, D. P. Dogra, S. Kar, and P. P. Roy, "Fingertip detection and tracking for recognition of air-writing in videos," *ESWA*, vol. 136, pp. 217–229, 2019.
- [58] M. S. Alam, K.-C. Kwon, and N. Kim, "Implementation of a character recognition system based on finger-joint tracking using a depth camera," *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 3, pp. 229–241, 2021.
- [59] Y. Zhang, Y. Zheng, K. Qian, G. Zhang, Y. Liu, C. Wu, and Z. Yang, "Widar3. 0: Zero-effort cross-domain gesture recognition with wi-fi," *IEEE TPAMI*, vol. 44, no. 11, pp. 8671–8688, 2021.
- [60] C. Feng, N. Wang, Y. Jiang, X. Zheng, K. Li, Z. Wang, and X. Chen, "Wi-learner: Towards one-shot learning for cross-domain wi-fi based gesture recognition," *Proc. of ACM IMWUT*, vol. 6, no. 3, pp. 1–27, 2022.
- [61] R. Xiao, J. Liu, J. Han, and K. Ren, "Oneifi: One-shot recognition for unseen gesture via cots wifi," in *Proc. of ACM SenSys*, 2021, pp. 206–219.
- [62] B. Planche, Z. Wu, K. Ma, S. Sun, S. Kluckner, O. Lehmann, T. Chen, A. Hutter, S. Zakharov, H. Kosch *et al.*, "Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5 d recognition," in *Proc. of IEEE 3DV*, 2017, pp. 1–10.
- [63] M. M. Rahman and S. Z. Gurbuz, "Self-supervised contrastive learning for radar-based human activity recognition," in *Proc. of IEEE RadarConf*, 2023, pp. 1–6.
- [64] M. M. Rahman, S. Z. Gurbuz, and M. G. Amin, "Physics-aware generative adversarial networks for radar-based human activity recognition," *IEEE TAES*, 2022.
- [65] H. Du, R. Zhang, D. Niyato, J. Kang, Z. Xiong, D. I. Kim, X. S. Shen, and H. V. Poor, "Exploring collaborative distributed diffusion-based ai-generated content (aigc) in wireless networks," *IEEE Network*, no. 99, pp. 1–8, 2023.



Kaikai Deng (Student Member, IEEE) is currently pursuing the Ph.D. degree with the School of Computer Science and the Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, China. He serves as a reviewer for many top conferences and journals, such as IMWUT/UbiComp (Outstanding reviewer), Knowledge-Based Systems, Signal Processing, and Frontiers of Computer Science. His research interests include Internet of Things,

edge computing, and multimodal sensing computing.



Dong Zhao (Member, IEEE) received the B.S. degree from the Department of Computer Science and Technology, Henan University, Kaifeng, China, in 2008, and the Ph.D. degree from the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China, in 2014. He is a Professor with Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications. He was a visiting Ph.D. student with Illinois Institute of Technology, Chicago, IL, USA, from 2012 to 2013, and was a Visiting Scholar with Rutgers University, New Brunswick, NJ, USA, from 2019 to 2020. His research interests include Internet of Things, mobile crowd sensing, urban computing, and data science, and he has published over 60 papers and two books on these fields. Dr. Zhao was awarded the China Computer Federation (CCF) Outstanding Doctoral Dissertation Award in 2015, the ACM Beijing Doctoral Dissertation Award in 2015, and the Natural Science Award of the Ministry of Education, China, in 2017.



Wenxin Zheng is currently working toward the M.S. degree with the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, China. Her research interests include Internet of Things, data generation, and infrastructure-assisted autonomous driving.



Yue Ling is currently working toward the PhD degree with the School of Computer Science and the Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, China. Her research interests include Internet of Things, AIGC, and multimodal data augmentation.



Kangwen Yin is currently working toward the Master degree with the School of Computer Science and the Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, China. His research interests include Internet of Things and multimodal data generation.



Huadong Ma (Fellow, IEEE) received the B.S. degree in mathematics from Henan Normal University, Xinxiang, China, in 1984, the M.S. degree in computer science from Shenyang Institute of Computing Technology, Chinese Academy of Science, China, in 1990, and the Ph.D. degree in computer science from the Institute of Computing Technology, Beijing, Chinese Academy of Science in 1995. He is currently a Professor in School of Computer Science, Beijing University of Posts and Telecommunications (BUPT), China. From 1999 to 2000, he held a Visiting Position with the University of Michigan, Ann Arbor, MI, USA. His current research interests include Internet of Things and sensor networks, multimedia computing, and he has published more than 300 papers in prestigious journals (such as ACM/IEEE Transactions) or conferences (such as ACM SIGCOMM, ACM MobiCom/MM, IEEE INFOCOM) and five books. Dr. Ma received the first class prize of the Natural Science Award of the Ministry of Education, China, in 2017. He received the 2019 Prize Paper Award of IEEE TRANSACTIONS ON MULTIMEDIA, 2018 Best Paper Award from IEEE MULTIMEDIA, Best Paper Award in IEEE ICPADS2010, and Best Student Paper Award in IEEE ICME2016 for his coauthored papers. He received the National Funds for Distinguished Young Scientists in 2009. He was/is an Editorial Board Member of the IEEE Transactions on Multimedia, IEEE Internet of Things Journal, ACM Transactions on Internet of Things, and Multimedia Tools and Applications. He serves as Chair of ACM SIGMOBILE China.