A. Artifact Appendix

A.1 Abstract

This artifact shows two major software sections of **MIMSID**, a high-resolution mmWave imaging system using metasurface and diffusion. We provide our end-to-end optimization model and signal-to-diffusion model, as well as some corresponding data for training and evaluating our models.

A.2 Artifact check-list (meta-information)

- **Algorithm: We present our end-to-end optimization model and signal-to-image model.**
- **Program: A set of data collected by our system is provided in our artifact, including channel matrix A and evaluation data.**
- **Model: We utilize diffusion models DDPM and DDIM in our system, which are included and implemented in our code.**
- **Data set: We use FashionMNIST dataset for model training, the download process is executed automatedly in our code. We also provide some real-world measurement data for model training and evaluating.**
- **Run-time environment: Our codes are implemented and tested on Windows 11 OS.**
- **Hardware: Our codes are tested on an NVIDIA GeForce RTX 3060.**
- **Execution: The training process may run for 1 hour. The evaluation process using the pre-trained model takes less than a second per sample.**
- **Metrics: The RMSE of the imaging system and the inference time are evaluated.**
- **Output: For the end-to-end optimization, a codebook of phase-shifting for the mmwave radar is the output. For the signal-to-image diffusion model, the outputs are image pairs of reconstructed and ground-truth images.**
- **Experiments: Please set up the experiment environment based on the requirements in Installation or run script setup.py for a Windows OS meeting our requirements.**
- **How much disk space required (approximately)?: This artifact is less than 3GB in total.**
- **How much time is needed to prepare workflow (approximately)?: The time for preparing the workflow should be less than 1 hour.**
- **How much time is needed to complete experiments (approximately)?: The full experiments should be completed in less than 3 hours.**
- **Publicly available?: No**

A.3 Description

A.3.1 How to access

The whole code and evaluation dataset can be downloaded from GitHub:

https://github.com/dozenw/MIMSID.git

A.3.2 Hardware dependencies

- CPU: No explicit requirements, but should be able to run the required software.
- RAM: At least 8 GB
- GPU: A CUDA-enabled GPU (our codes are tested on an RTX 3060 with CUDA 12.4. The Nvidia driver of our computer is **version 551.32**.)

A.3.3 Software dependencies

Windows or Linux based operating system with the latest conda and Nvidia driver (version number 470 or later). We use Windows 11 and conda 23.1.0.

Requirements:

The detailed package requirements are described on our github repository.

We recommend to use cuda 12.4, torch 2.6.0, and torchaudio 2.6.0, and torchvison 0.21.0.

The **fashion-mnist_train.csv** and **fashion-mnist_test.csv** are provided in

https://pan.baidu.com/share/init?surl=tk_yZyuzoA26Vy_VeNzsQg&pwd=v16u

or

https://drive.google.com/drive/folders/1p8zrBXasdzsQb-eiKvT9usp=sharing

A.4 Installation

Step 1. Extract the artifact repositories via git clone and cd into it.

```
git clone https://github.com/dozenw/MIMSID.git
cd MIMSID
```

Step 2. Create Conda environment and activate it.

```
conda create -n MIMSID python=3.9.13
conda activate MIMSID
```

Step 3. Install CUDA 12.4. (Note this is one command line.)

```
conda install
-c "nvidia/label/cuda-12.4" cuda-toolkit
```

Step 4. Install requirements.

```
pip install -r requirements.txt
```

Step 5. Place the downloaded fashion-mnist_train.csv and fashion-mnist_test.csv into the dataset folder.

A.5 Experiment workflow

A.5.1 End-to-end optimization

To run the end-to-end optimization, simply run

```
python e2e_optimization.py
```

The optimized codebook will be printed out and the corresponding channel model will be saved in **mts_model** folder.

A.5.2 Signal-to-image diffusion

To test the Signal-to-image diffusion model, simply run

```
cd sig2imgDM
python main.py
```

The default program is running in evaluation mode. To activate the training process, run

```
python main.py --if_Train=True
```

When activating training, the model is stored in the **sig2imgDM** folder with the name **conditional_ddpm_width192_fmnist_Aytrain.py**.

To disable evaluation, run

```
python main.py --if_Eval=False
```

A.6 Evaluation and expected results

The example output is included in our github page. **MIMSID/ sig2imgDM/ output** shows image pairs of reconstructed (named with suffix "_est") and ground truth (named with suffix "_gt") images. The evaluated RMSE should be under 0.06 with mean inference time less than 0.2 seconds.

The detailed results for running **python main.py** on our device are:

```
PSNR: 47.2251
mean RMSE tensor: 0.0430
mean infer time: 0.155576
```

For running **python main.py –if Train=True** on our device, the results are:

```
PSNR: 21.7930
mean RMSE tensor: 0.2572
mean infer time: 0.1169613
```

A.7 Experiment customization

End-to-end optimization To customize, you can alter the iteration steps of the initialization and the learning rates of the models in the "if __name__ == "__main__": section. You can also modify the training steps in function **optimize_step**. In addition, you can modify the parameters of class **mmWaveSimulator**, where the target's distance, image resolution, and metasurface size can be altered.

Signal-to-image diffusion To customize, you can modify the training epochs **n_epoch** and learning rate **lr** in the training section.

A.8 Notes

A.9 Methodology

Submission, reviewing and badging methodology:

- `https://www.acm.org/publications/policies/artifact-review-and-badging-current`
- `https://cTuning.org/ae`