A Additional Evaluation Results

A.1 End-To-End Inference on More GPUs

Figure 18, Figure 19, Figure 20, and Figure 21 show the end-to-end inference performance of all SpC engines across different networks and datasets, measured on GTX 1060, Quadro RTX 5000, H100, and Jetson Orin AGX GPUs, respectively. For these benchmarks, all evaluations on the GTX 1060 are performed in 32-bit float precision, since 16-bit float precision in this architecture is inefficiently supported, i.e., it achieves 64× lower compute throughput than that of 32-bit float.

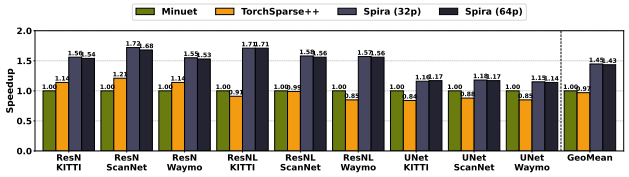

Figure 18. End-to-end inference performance of all SpC engines using various point cloud networks and datasets on GTX 1060 GPU.

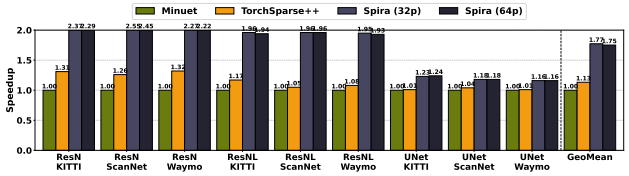

Figure 19. End-to-end inference performance of all SpC engines using various point cloud networks and datasets on Quadro RTX 5000 GPU.

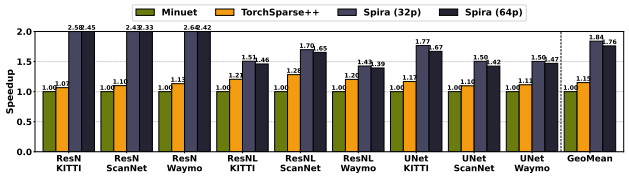

Figure 20. End-to-end inference performance of all SpC engines using various point cloud networks and datasets on H100 GPU.

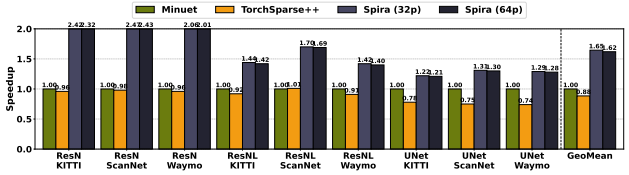

Figure 21. End-to-end inference performance of all SpC engines using various point cloud networks and datasets on Jetson Orin AGX platform.

Spira significantly outperforms Minuet and TorchSparse++ across all GPUs, with 32-bit packing delivering consistent average speedups of 1.47×, 1.67×, 1.71×, and 1.75× on GTX 1060, Quadro RTX 5000, H100, and Jetson Orin AGX, respectively. These results demonstrate Spira's ability to provide robust performance gains regardless of the underlying GPU architecture.

A.2 Memory Footprint Evaluation

Table 4 presents the peak memory footprint (including the memory footprint of kernel maps, weights, and features) and performance of all SpC engines measured during end-to-end inference on the Waymo Dataset. We evaluate UNet (42 layers) with large kernel size (**K**) up to 13 and batch size (**B**) up to 4 using the RTX 3090 GPU (24GB).

Configuration	TorchSparse++	Minuet	Spira
K=3; B=1	353MB / 1.00	405MB / 1.06×	360MB / 1.33×
K=7; B=1	1.6GB / 1.00	1.7GB / 1.30×	1.4GB / 2.14×
K=13; B=1	9.2GB / 1.00	9.6GB / 1.15×	9GB / 2.20×
K=3; B=4	1.1GB / 1.00	1.3GB / 1.05×	1.1GB / 1.28×
K=7; B=4	3GB / 1.00	3.5GB / 1.28×	2.7GB / 1.75×
K=13; B=4	17GB / 1.00	OOM / –	16GB / 1.81×

Table 4. Peak memory footprint and performance of all SpC engines in end-to-end inference for different kernel size (**K**) and batch size (**B**) configurations.

Our results show that the peak memory footprint of all SpC engines is similar. While Spira pre-computes all kernel maps upfront at the network start, TorchSparse++ and Minuet generate them progressively, as layers are executed, and retain them in memory to reuse them across SpC layers that share the same kernel maps (See §5.5). Hence, all SpC engines converge to have similar memory footprints for kernel maps, as well as all other inference data including features and weights, which remain identical across the engines. Additionally, Spira outperforms Minuet and TorchSparse++ by up to 1.30× and 2.20×, respectively. In these experiments, the minimum and maximum kernel map memory footprint of Spira is 19MB (K=3 & B=1) and 6.4GB (K=13 & B=4), respectively.

B Artifact Appendix

B.1 Abstract

The Artifact Appendix describes how to reproduce the main results of this paper. It includes the source code of Spira, benchmark scripts, and step-by-step instructions for the key evaluation results. The experiments require an NVIDIA GPU with an up-to-date NVIDIA driver installed. We provide a `README.md` file that describes the required hardware and software dependencies and provides step-by-step instructions. Note that the datasets used in our evaluations are licensed. This artifact is used to support our major claims (See §B.6), demonstrating Spira's performance benefits in Figure 8, Figure 9, Figure 10, Figure 12, and Figure 16. We expect the full evaluation pipeline, including setup to take approximately 2–3 hours. We also provide a Dockerfile to automatically set up the runtime environment for running the artifact. We **recommend** using the docker environment for running the experiments.

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

- **Program:** *Spira_Artifact*: In this artifact, we compile and evaluate three comparison points Minuet, TorchSparse++ and Spira.
- **Compilation:** CMake build system; GNU compilers (gcc/g++); NVIDIA CUDA compiler (nvcc); PyTorch. The artifact is compiled as a Docker image for ease of deployment and reproducibility.
- **Data set:** SemanticKITTI (KITTI), ScanNet and Waymo. All datasets are licensed.
- **Run-time environment:** Linux Ubuntu 22.04 with Python 3.10, requiring CUDA 12.4.1 and PyTorch 2.5.0, all provided via a Docker image.
- **Hardware:** A system with an NVIDIA GPU device with a minimum compute capability of 7.5 and at least 16GB GPU memory should be used to validate the results.
- **Metrics:** Execution time in milliseconds normalized as relative performance speedup.
- **Output:** Output files containing raw results. Figures similar to Figure 8, Figure 9, Figure 10, Figure 12, and Figure 16 of the main paper.
- **Experiments:** End-to-end inference performance, layer-wise performance, mapping performance, scene density ablation study, correctness check.
- **How much disk space required (approximately)?:** 256 GB.
- **How much time is needed to prepare workflow (approximately)?:** 60 minutes (build the code and download datasets).
- **How much time is needed to complete experiments (approximately)?:** 90 minutes.
- **Publicly available?:** Yes.
- **Code licenses (if publicly available)?:** Apache 2.0.
- **Data licenses (if publicly available)?:** ScanNet: ScanNet Terms of Use; Waymo Open Dataset: Waymo Open Dataset License; SemanticKITTI: CC BY-NC-SA 4.0.
- **Archived (provide DOI)?:** 10.5281/zenodo.18879475.

B.3 Description

B.3.1 How to Access

Download the compressed file Spira_Artifact.zip from the Zenodo archive https://doi.org/10.5281/zenodo.18879475 or our GitHub repository at https://github.com/SPIN-Research-Group/Spira.

B.3.2 Hardware Dependencies

The artifact should be tested on a host machine with:

- x86-64 CPU with at least 64GB main memory and 256GB disk storage.
- NVIDIA GPU device with a compute capability (SM) of 7.5–9.0 and at least 16GB GPU memory.

B.3.3 Software Dependencies

The artifact requires the following software for installation:

- Ubuntu 22.04 (or newer)
- Python 3.10
- GNU compilers (gcc/g++) 11.4.0 (**strict** requirement)
- CUDA 12.4.1 (or newer)
- PyTorch 2.5.0 (or newer)
- CMake 3.27.0 (**strict** requirement)
- Pybind 2.11 (**strict** requirement)
- libsparsehash-dev 2.0.3 (or newer)
- libopenblas-dev 0.3.20 (or newer)

To simplify setup and ensure reproducibility, we **recommend** building the artifact as a Docker image. The Dockerfile installs all necessary software dependencies, including additional Python packages for dataset preparation and figure generation. To build and run the artifact as a Docker image, users should have a Linux-based operating system with an up-to-date NVIDIA driver (supporting at least CUDA 12.4), Docker Engine and NVIDIA Container Toolkit installed. For reference, our environment uses:

- Debian GNU/Linux 12
- NVIDIA driver 550.54.14
- Docker Engine 29.2
- NVIDIA Container Toolkit 1.13.5

B.3.4 Data Sets

For our evaluation, we use scenes from 3 licensed real-world datasets. We provide detailed instructions for getting access, downloading and preparing the datasets in the `README.md` file of the artifact. The real-world datasets are the following:

- SemanticKITTI (KITTI)
- ScanNet
- Waymo

B.4 Installation

Download the zip file containing the artifact source code in §B.3.1. We provide detailed instructions in the `README.md` file under the root of source code directory to build and install Spira and baselines.

We provide a Dockerfile to setup the runtime environment for all the experiments.

1. Install Docker Engine following the instructions provided in https://docs.docker.com/engine/install/ubuntu/.
2. Install NVIDIA Container Toolkit following the instructions provided in https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/latest/install-guide.html.
3. Download the source code of Spira_Artifact.
4. Export the variable `CUDA_ARCHS` with the targeted GPU architecture for evaluation and build the Docker image by executing the following command at the root directory

of the source code:

```
$ docker build --build-arg
CUDA_ARCHS=$CUDA_ARCHS -t spira .
```

B.5 Experiment workflow

The artifact contains 5 experiments to evaluate the performance and correctness of the SpC engines, with each experiment executed by a script located in the `automate/` directory. Each script runs the experiment, parses the raw results, and generates a corresponding figure. The first four experiments 1-4 measure execution time and save the raw results under the `results/` directory, and generate figures under the `figures/` directory. Next, we describe in detail how to run each experiment.

1. End-to-End Inference Performance: The following script evaluates the end-to-end inference performance of all SpC engines across different datasets and networks (Figure 8 and Figure 9):

```
$ bash automate/end_to_end.sh
```

2. Layerwise Performance: The following script evaluates the layerwise performance of all SpC engines averaged across all datasets for different SpC layer configurations (Figure 10):

```
$ bash automate/layerwise.sh
```

3. Mapping Performance: The following script evaluates the mapping performance in voxel indexing step of all SpC engines for various input coordinate counts and layer kernel sizes (Figure 12):

```
$ bash automate/mapping.sh
```

4. Scene Density Ablation Study: The following script evaluates end-to-end inference performance averaged across all networks for synthetic scenes of varying sparsity (Figure 16):

```
$ bash automate/ablation.sh
```

5. Correctness: The following command verifies the correctness of all SpC engine outputs (coordinates and features), including all threshold selections for Spira:

```
$ bash automate/correctness.sh
```

All: The following command will execute *all* experiments and generate *all* figures:

```
$ bash automate/run_all.sh
```

B.6 Evaluation and expected result

Major Claims. For each of the first four experiments 1-4, we expect the reproduced results to be similar to those reported in the paper, given the same input configuration. We next clarify our major claims:

1. Spira achieves significant end-to-end point cloud inference speedup over prior state-of-the-art SpC engines on modern GPUs. As indicative results, we report speedups in the range of 1.5×–2.1× on the RTX 3090 and A100 GPUs.
2. Spira achieves significant layerwise speedup over prior state-of-the-art SpC engines on modern GPUs. As indicative results, we report speedups in the range of 1.9×–2.6× on the RTX 3090 and A100 GPUs across diverse SpC layer configurations.
3. Spira's z-delta search delivers significant mapping part acceleration over Minuet, TorchSparse++, and Simple BSearch on modern GPUs. As indicative results, we report speedups in the range of 2.7×–7.8× on the RTX 3090 and A100 GPUs for the mapping part of voxel indexing step.
4. When evaluated on randomly generated scenes with densities ranging from 0.12% to 12.50%, Spira achieves significant end-to-end point cloud inference speedup over prior state-of-the-art SpC engines on modern GPUs. As indicative results, we report speedups in the range of 1.5×–3.1× on the RTX 3090 and A100 GPUs.

B.7 Experiment customization

Each bash script can be configured to modify the input arguments (e.g., dataset, network, scenes) of the corresponding experiment. The input arguments are defined at the beginning of each script and can be modified to test different input configurations. For example, in the end-to-end inference performance script, lines 3–5 can be modified to test different datasets, models (i.e., different networks) and/or SpC libraries (i.e., different engines):

```
DATASETS=(...)
MODELS=(...)
LIBS=(...)
```