Design and Deployment of a Multi-Modal Multi-Node Sensor Data Collection Platform

Shiwei Fang
shiweifang@cs.umass.edu
University of Massachusetts Amherst

Ankur Sarker
ankursarker@g.ucla.edu
University of California, Los Angeles

Ziqi Wang
wangzq312@g.ucla.edu
University of California, Los Angeles

Mani Srivastava
mbs@ucla.edu
University of California, Los Angeles

Benjamin Marlin
marlin@cs.umass.edu
University of Massachusetts Amherst

Deepak Ganesan
dganesan@cs.umass.edu
University of Massachusetts Amherst

ABSTRACT

Sensing and data collection platforms are the crucial components of high-quality datasets that can fuel advancements in research. However, such platforms usually are ad-hoc designs and are limited in sensor modalities. In this paper, we discuss our experience designing and deploying a multi-modal multi-node sensor data collection platform that can be utilized for various data collection tasks. The main goal of this platform is to create a modality-rich data collection platform suitable for Internet of Things (IoT) applications with easy reproducibility and deployment, which can accelerate data collection and downstream research tasks.

CCS CONCEPTS

• **Computer systems organization → Sensors and actuators**; **Sensor networks**; **Embedded and cyber-physical systems**; • **Human-centered computing → Ubiquitous and mobile computing systems and tools**.

ACM Reference Format:
Shiwei Fang, Ankur Sarker, Ziqi Wang, Mani Srivastava, Benjamin Marlin, and Deepak Ganesan. 2022. Design and Deployment of a Multi-Modal Multi-Node Sensor Data Collection Platform. In *The Fifth International Workshop on Data: Acquisition To Analysis (DATA '22), November 6–9, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3560905.3567770

1 INTRODUCTION

Researchers in multiple research communities have shown interest in multi-modal sensing, ranging from video and audio in computer vision and natural language processing to the camera, LIDAR, and radar in autonomous vehicles. The possibility of combining multi-modal sensing capabilities to harness each sensor type's advantages while minimizing their drawbacks excites researchers, especially when there are significant advancements in machine learning with each modality. Furthermore, with the tendency to require a large amount of data to train neural network models, multi-modal datasets become increasingly essential to fuel the advancements of such research. However, existing multi-modal datasets tend to target specific applications, such as video-and-language datasets [16] [18], or rich in multi-view but lack in sensor varieties, such as autonomous vehicle datasets [10] [14] [12] [11] generally have multiple copies cameras and LIDARs, but lack other sensing modalities such as sound and RF.

To address the lack of sensor varieties in multi-modal datasets, we set to build a multi-modal sensor data collection platform. This platform should be able to reproduce and deploy in various locations and scenarios to collect datasets for different research projects, especially in the area of the Internet of Things (IoT). For example, the same types of multi-modal sensors should support easy deployment to collect data for either human activity recognition or object re-identification. We note there are benefits to collecting data with a specific set of sensors to demonstrate the feasibility of achieving maximum performance with a minimum number of sensors. However, there is an argument that creating a modality-rich dataset can further the research and provide more insights into how each modality can perform and complement each other given a specific situation. Furthermore, selecting a subset of sensors in a dataset is always easier than collecting more data with a different set of sensors.

To this end, we designed a multi-modal sensor data collection platform that consisting sensors of various modalities to enable modality-rich dataset collections with easy reproducibility and deployment. Our platform's easy reproducibility and management (discussed in later sections) enables multi-node deployment that can be configured to provide broader coverage, multi-view capability, or a combination of both for the intended research tasks. The sensor platform is suitable for IoT applications where low-cost off-the-shelf sensors are normally utilized in contrast to highly advanced, specialized, and expensive sensors used in autonomous vehicle datasets.

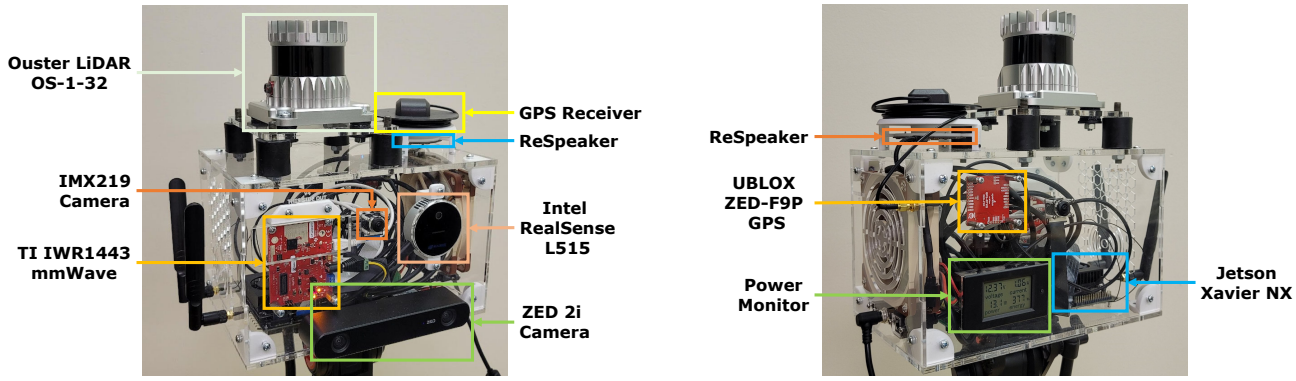
2 SYSTEM OVERVIEW

2.1 Hardware

For our multi-modal sensor data collection platform, we start our design by choosing common sensing modalities used by different research communities with off-the-shelf hardware availability, namely vision (visible light), range, motion, and sound. Our sensors of choice are shown in Table 1 to fulfill these modalities. While

DATA '22, November 6–9, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9886-2/22/11…$15.00
https://doi.org/10.1145/3560905.3567770



(a) Front of the multi-modal sensor platform.

(b) Rear of the multi-modal sensor platform.

Figure 1: Multi-Modal sensor platform with most sensors attached to the front of the enclosure for same direction sensing.

Sensor	Modality
Intel RealSense L515 [2]	Vision, Range
ZED 2i [8]	Vision, Range, Motion (self)
TI IWR1443 mmWave Radar [9]	Range, Motion
ReSpeaker Mic Array V2.0 [6]	Sound
Ouster LIDAR OS-1-32 [5]	Range
SparkFun ZED-F9P GPS [7]	Position

Table 1: Sensors used in our multi-modal sensor platform and their corresponding modalities.

some sensors cover the same or similar modalities, they have different strengths and weaknesses. For example, Intel RealSense L515 can work in darkness but not under heavy ambient light, whereas ZED 2i can only work when there is sufficient light but becomes nonfunctional in darkness.

To control and collect data from these sensors, we use the Jetson Xavier NX Developer Kit [4] as our computing device. This Jetson model has better port configuration and processing power than the Jetson Nano but a lower cost than the more capable AGX models. Moreover, the low power consumption can also enable the deployment of such a platform in an outdoor environment with a portable battery power station. With our current implementation, the power consumption with all the sensors, excluding Ouster LIDAR, collecting is around 29W, and Ouster LIDAR requires an additional 20W.

All the sensors and the Jetson are packaged in a custom laser-cut acrylic enclosure with custom 3D-print mounting brackets, as shown in Figure 1. Most sensors are attached to the front of the enclosure, as shown in Figure 1a, for sensing the same direction. Each sensor has its own custom-designed cutout on the acrylic panel to eliminate potential interference. The ZED-F9P GPS module is attached to the back of the enclosure (shown in Figure 1b) as its location does not affect the collected data. The GPS antenna is mounted at the top of the enclosure, and ReSpeaker is right below. The ReSpeaker is placed outside the enclosure for better sound sensitivity without the blockage of the acrylic panels. We note that while the IMX219 CSI camera is shown in the figure, we are not utilizing it at the moment as both Intel RealSense L515 and ZED 2i provide RGB vision data. This platform can be easily reproduced and deployed simultaneously, and each copy is referred to as a node in a muli-node setup in this paper.

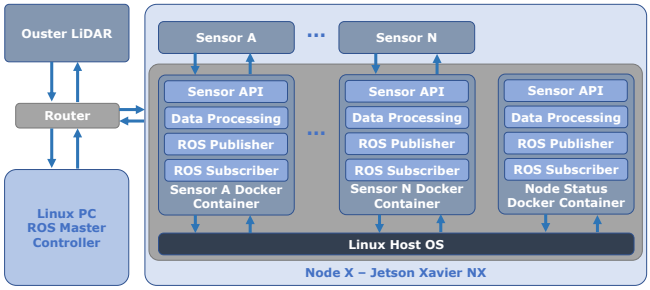


Figure 2: Multi-modal sensor platform data flow and architecture.

2.2 Software

The software architecture and data flow are shown in Figure 2. The software is the same on all the nodes (if multiple copies are produced and deployed), where they are deployed through Docker containers. For each sensor type, we build a separate container with ROS subscriber for command inputs, ROS publisher for periodic data publishing to enable real-time monitoring, sensor API for physical sensor interaction, and data processing for encoding and storing the sensor data. The node status container does not have a dedicated sensor but collects data from the system for resource usage. Containers generally have low resource overhead, ensure the software performs the same on all the nodes and allow easy deployment for multiple copies. Each sensor is isolated in separate containers to improve system stability. We also note that the Ouster LIDAR is connected to the router, and the corresponding data collection node (e.g., node 1 has the Ouster LIDAR installed) will execute the Ouster LIDAR container to record the LIDAR data. A Linux PC is connected to the router through either WiFi or LAN to act as the ROS master, and control the operations on all the nodes simultaneously.

3 IMPLEMENTATION

3.1 Intel RealSense L515

The initial sensor choice for our vision and depth sensing is Intel RealSense L515 [2]. It can provide vision and depth simultaneously, and the active IR depth camera enables continuous sensing in situations with poor lighting. The RealSense L515 also performs computation onboard. The host system (Jetson Xavier NX) only requires

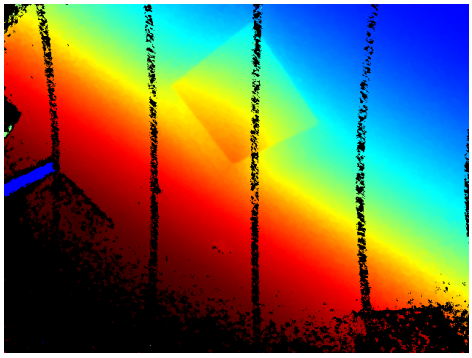


Figure 3: Intel RealSense L515 experience banding issue when used in motion capture lab with infrared tracking system.

to interface with the sensor and acquire the data for storage, thus reducing the host system's computation load. However, while testing our system in the motion capture lab with the Qualisys system, we encountered an interference issue as both the L515 and Qualisys utilize infrared around the same wavelength. The interference is shown in Figure 3, where dark bandings can be seen (the depth image is colorized using the Intel RealSense SDK [3]). This interference issue can be alleviated by adjusting the capturing frequency of the Qualisys system, but it can not be eliminated. We added ZED 2i to our multi-modal sensor platform as a complementary vision and depth sensor to reduce the interference's effect and improve the datasets' richness.

Intel RealSense SDK [3] is used to communicate with the sensor to acquire RGB images and depth frames. After acquiring each frame, the RGB image is directly sent to a GStreamer [1] pipeline for storing the images in H264 format. For the depth frames, we first convert the depth information to colormap using Turbo [15] and then send it to a separate GStreamer pipeline to store the depth information in H264 format. They are stored in H264 encoding for storage efficiency, as raw images and depth arrays can take up significant storage space. For example, to store the depth arrays in a ROS bag file at a frame rate of 15 frames per second (fps), the raw data can take around 1-2 GB per minute, depending on the resolution and bit depth. In addition, storing RGB images in either PNG or JPEG files can take several hundreds of Megabytes (MB) to several Gigabytes (GB) depending on the compression ratio, the complexity of the scene (more complex scene results in less compression), and the resolution of the image. Whereas encoding both RGB image and depth frames in H264 video encoding, the file size can be reduced to around 100-200 MB per minute at the same frame rate. While encoding and decoding the RGB image is relatively simple and relatively lightweight, the encoding and decoding of the depth frame can be computationally intensive as each pixel needs to be computed. To reduce the time required to decode each depth frame, we utilize PyTorch [17] and GPU to perform the computation in matrix form.

3.2 Stereolabs ZED 2i

For RGB and depth measurement, we also include a Stereolabs ZED 2i [8] depth camera for complement sensor characteristics and to mitigate the interference on the Intel RealSense L515, as mentioned earlier. Unlike Intel RealSense L515, the ZED 2i depth camera generates depth images by comparing the difference between an RGB camera pair spaced by 12 cm. There are different strengths and limitations for the L515 and ZED 2i, as shown in Table 2. However, their complement characteristics should provide better depth information coverage in different scenarios than utilizing only one. It's worth noting that while the ZED 2i consumes less power through the USB port, which can be beneficial to the Jetson device, NVIDIA GPU is required for real-time depth generation, which can pose significant computation challenges for embedded devices.

	RealSense L515	ZED 2i
Depth Range (m)	0.25 - 9	0.2 - 20
Computation	Onboard	GPU
Limitation	Ambient Light	Darkness

Table 2: Difference between Intel RealSense L515 and Stereolabs ZED 2i. While both provide depth measurement, they have different strengths and usage scenarios.

ZED SDK is used in the software implementation of the ZED container. Instead of acquiring the RGB and depth frames as with Intel RealSense L515, we record the ZED data and save them into the Stereolabs' proprietary SVO files. Internally, the data are stored similarly to H264 MP4 encoding. Utilizing the SVO file for data recording can significantly reduce the file size as each minute of data requires around 100 MB with a frame rate of 15 frames per second. Furthermore, because the data is recorded in the SVO file, the computation on the Jetson is minimal, as no depth calculation is performed. After data collection, the SVO file can then be processed on any computation system as long as it has ZED SDK support and NVIDIA GPU. However, while decoding the SVO file and generating depth information is fast on desktop GPU, the I/O bottleneck associated with saving individual RGB images and depth arrays (as Python numpy array) is a limiting factor, and there is no significant performance difference between utilizing HDD or SSD. We also collect IMU (inertial measurement unit) data from the ZED onboard IMU sensor for datasets completeness and situations where the node is moving.

3.3 TI mmWave IWR1443

The TI mmWave IWR1443 mmWave radar is selected for sensing the environment through an alternative modality, i.e., radio frequency. RF sensing poses different characteristics when compared with traditional vision-based sensor systems. For example, mmWave radar can see through objects, simultaneously measure range and speed, and operate in bright daylight and total darkness. However, the advantages also have clear disadvantages, such as low angular resolution and limited see-through capability. The angular resolution of a mmWave radar is determined by the number of antennas available. For example, the low-cost TI IWR1443 mmWave radar only has two azimuth Tx antennas and four Rx antennas, which corresponds to an azimuth angular resolution of 15°(detailed explanation of mmWave angular resolution can be found at [20]). The see-through capability also depends on the frequency and transmitted RF power – lower frequency and higher transmitting power can penetrate more objects.

For the software, we modified code from [13] to record the range azimuth heat map, range doppler heat map, range profile, noise

profile, and point cloud. The mmWave radar configuration file is updated to suit our data collection environment better. Due to the limited bandwidth through the USB connection, the radar can only achieve around 3-4 fps. We note that one can add a DCA1000EVM real-time data-capture adapter for the TI mmWave radar [19], which can enable the collection of raw mmWave data at a higher frame rate. The collected data are directly stored in ROS bag files as arrays. During our data collection experiments, we observed data corruption due to data speed and USB data buffer. The data corruption can present as a wrong range azimuth heat map and no range doppler heat map. We remove such corrupted data frames during post-processing. The mmWave radar configuration file is also stored in the ROS bag file for future reference. The configurations are required to calculate the mapping between the arrays to the corresponding angles and ranges in real-world coordinates.

3.4 ReSpeaker

To monitor the sound modality, we choose the ReSpeaker microphone array. The ReSpeaker has four microphone inputs with an onboard processor for speech recognition, degree of angle calculation, and more. The four microphones are separated by 90 degrees from each other. It also has 12 LED lights that can be used to indicate where the sound is coming from. We built a custom standoff where the ReSpeaker can be attached and integrated with the enclosure. The standoff is placed at the top of the enclosure, where the intuition is the sound signal should not be obstructed by the acrylic panels (even with cutouts). The four microphones face the enclosure's front, rear, left, and right.

To record the sound data from the ReSpeaker, we use pyaudio Python library to record all channels and encode the data in FLAC file format. Once the predefined length of each recording is reached, the data is encoded and stored as a byte string in the ROS bag. The byte string is extracted and converted back to the FLAC file during data processing. In addition, the ReSpeaker-calculated direction of arrival (DOA) is also recorded and stored in the ROS bag file. The ReSpeaker does not update the DOA data if no new direction is detected and requires the software to periodically poll the data for acquisition. Because the sound modality is not discrete in time as RGB image is, the DOA data is polled at the same frame rate as other vision sensors. In addition, timestamps are recorded for each DOA data, the start and the end of each recording, which then can be used to splice the sound files into shorter lengths to match the timestamps of other sensors.

3.5 ZED-F9P GPS

For accommodation of possible outdoor data collection, a SparkFun GPS-RTK2 Board with ZED-F9P [7] is integrated into our data collection platform. This ublox ZED-F9P chip on this GPS board can receive RTCM 3.x format correction data for GPS Real Time Kinematics (RTK), which can significantly improve GPS accuracy. While the theoretical accuracy can be in the magnitude of centimeters, the actual performance can be different given the actual environment where the board is deployed. For example, our platform's low-cost magnetic GPS antenna needs a metal ground plate to attach to and improve performance, which can introduce variance between nodes. In addition, the distance between the GPS device and the base station can also significantly affect the RTK accuracy, as a fixed base station's effective range is around 10km. The metal ground plate is mounted on top of the ReSpeaker standoff, which can be seen in Figure 1. This position provides an unobstructed view of the sky.

To perform RTK-corrected GPS data collection, we have two software components running in a single container. The first software component polls GPS data from the sensor with GPS-reported latitude, longitude, height, accuracy, and more. This component also broadcasts the received GPS coordinates through the ROS topic. The second component subscribes to the ROS topic and monitors the GPS coordinates. Once the GPS has a coordinate, it goes through a list of nearby base stations and finds the closest one, then retrieves the RTK data from the base station through the internet and sends the data to the GPS sensor. We note that the GPS will require additional time after continuously retrieving the RTK data before converging to improve accuracy.

3.6 Ouster LIDAR

An Ouster OS-1-32 LIDAR is integrated into our multi-modal sensor platform, enabling long-distance range measurement day and night. While the LIDAR can provide high-accuracy range measurement in a large area, the cost for such a sensor is high. Therefore, a single LIDAR unit is deployed with our data collection nodes after factoring in the sensor cost and coverage area. A rubber mount is used in attaching the Ouster LIDAR to the enclosure, as shown in Figure 1. The rubber mounts are used to minimize the vibration and sound from the LIDAR, which may affect other sensors as the LIDAR is mechanically rotating. The height of rubber mounts serves two purposes: first, to minimize the sound blockage to the ReSpeaker mounted nearby, and second, to minimize the interference to other components of the system due to the heat generated by the LIDAR. The Ouster LIDAR is powered separately from other components by directly connecting to the power source to improve stability with the LIDAR's included cable and power supply.

The interface between the sensor platform and the Ouster LIDAR is through its SDK and its modified ROS example. Instead of directly communicating between the Ouster LIDAR and the Jetson, we found it easier to connect the LIDAR to the router through ethernet and retrieve data through the local network. The data is stored as Ouster's custom lidar packets into ROS bag files with associated timestamps. The lidar packets can be read and replayed as virtual LIDAR for data processing and decoded into desirable formats (e.g., range depth map, point cloud, etc.). The raw data collected enables flexibility after the data collection and minimize the computation required during collection time, thus reducing the computation load of the Jetson device. Because the LIDAR is not directly connected to the Jetson device and broadcasts its data through the local network, the docker container is executed on the node where the LIDAR is attached to enable the correct data file naming convention.

3.7 Jetson Xavior NX

Jetson Xavier NX Developer Kit is chosen to be the computing module of our data collection platform. Different from Jetson Nano, the Xavior NX platform has much higher computing power in both CPU and GPU while still maintaining a low power consumption. Compared to the AGX platform, the price of the Xavior NX is more manageable, and the port selection is better suited for a multi-modal

sensor platform. While the OS runs on the microSD card, a separate SSD is installed for data storage and better I/O performance (the Jetson Nano developer kit does not have an SSD port). The Jetson's plastic standoff and WiFi antennas are removed to allow the Jetson to be integrated with the enclosure by attaching to the panel. Plastic standoffs are used to create the necessary spacing between the Jetson PCB board and the bottom panel. Aftermarket antennas are mounted on the side panel for better WiFi performance.

The Jetson Xavier NX runs Nvidia's custom Ubuntu 18.04 operating system with docker installed. In addition, a node status container is implemented to collect and store real-time performance metrics, such as CPU, RAM, storage utilization, temperature, and NTP status. With the containerization of all sensor collection software, each copy of the sensor platform can have the same configuration and performance.

3.8 Enclosure & Power Supply

For reproducibility and more accessible transportation, we designed our custom enclosure and power supply system for the multi-modal sensor platform. The enclosure design is aimed for small in size but provides good thermal performance. For cost reduction and better visuals, we choose laser-cut acrylic panels with 3D-printed mounting hardware for the enclosure. The cost of the laser-cut manufacturing process is significantly less than 3D printing. The full enclosure and the transparent property of the acrylic can allow us to visually inspect the system's state with the LEDs onboard the sensors and the Jetson. In addition, we added a 120 mm Noctua NF-S12A ULN case fan with Low-Noise Adapter (L.N.A.) attached. We chose this fan for the optimized airflow design with a low 6.8db noise when L.N.A. is used. As the Jetson device has an onboard fan for cooling, it's more important for our design to remove the hot air accumulated in the case and supply it with fresh air.

The power supply requirements for the sensors are relatively straightforward, as most can be powered through USB cables. However, TI IWR1443 mmWave radar requires a 5V power supply. In addition, the case fan requires a DC power supply with 12V DC. Given the Jetson Xavier NX development kit can be powered from 9V - 20V DC, we choose a 12V AC to DC power adapter as the main power source. The 12V input power is connected to a power distribution board, which then distributes power directly to the Jetson Xavier NX development kit and 120mm case fan. A DC-DC step-down converter converts the 12V to 5V for TI mmWave radar. We also included a power meter for real-time voltage and energy consumption monitoring, which can provide a good indication of the system's state.

4 LESSONS ALONG THE WAY

Hardware Availability One major hurdle in our hardware design and deployment is the availability issue. This project coincides with the COVID-19 pandemic, which resulted in severe supply chain disruptions. During the initial project phase, we only purchased a small number of various sensors and NVIDIA Jetson Xavier NX development boards, which posed significant issues in later stages. The Jetson devices are sold-out in almost all online stores, and we are limited by the number of Jetson devices on-hand. In addition, during the development, Intel announced the "winding down" of its RealSense computer vision division, which may result in future scarcity of the RealSense depth sensors.

Cost Overrun With the initial intended purpose of the multi-modal sensing platform, the proposed sensor selections in the proposal have fewer varieties. Thus, the budget per node is set to the cost of those sensors combined. However, new sensors are added during our development to compensate for different limitations. Along with the cost of new sensors, we encountered rising prices on devices and additional costs for the enclosure and power supply system, which were not in the proposal. Thus we experienced cost overrun, resulting in either fewer nodes for the fixed total budget or more nodes with additional funding.

Verification Overhead While during the hardware and software development phase, we can test each sensor and measure their performance, their actual behavior in the data collection environment is different as the environment changes. For example, our initial design does not include the ZED stereo depth camera. However, the motion capture lab uses infrared with the Qualisys system, thus presenting interference with our Intel RealSense L515 camera. The new sensor addition added development time to our project with sensor procurement, software development, updated enclosure design, and additional testing. Each cycle of hardware and software development, motion capture lab data collection, and verification can take several days or weeks. Furthermore, numerous such cycles are usually needed to iron out issues with the data collection platform.

Data Management We designed our system to store the collected data as small in file size as possible, increasing the ease of storing and transferring such data. However, decoding the raw data can take extra time and space, especially when file system I/O can limit the performance of the data processing pipeline. For example, while the generation of depth images of the ZED stereo camera is relatively fast, writing both left and right RGB images with depth values to disk slows the process even when using an SSD. In addition, the decoded data is also large in size, which may pose challenges to data storage for downstream tasks.

5 CONCLUSION

In this paper, we propose a multi-modal multi-node sensor data collection platform that can be deployed to collect multi-modal datasets in various scenarios. Implementation and lessons learned through the journey are provided for future reference to the community. We hope such contributions can spur more designs that can streamline the data collection process to enable high quality datasets and free up researchers to focus more on utilizing the data instead of building data collection platforms.

ACKNOWLEDGMENTS

The research reported in this paper was sponsored by the CCDC Army Research Laboratory (ARL) under Cooperative Agreement W911NF-17-2-0196 ARL IoBT CRA. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ARL or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

REFERENCES

[1] GStreamer. URL: https://gstreamer.freedesktop.org/.

[2] Intel RealSense L515. URL: https://www.intelrealsense.com/lidar-camera-l515/.

[3] Intel RealSense SDK. URL: https://www.intelrealsense.com/developers/.

[4] Nvidia Jetson Xavier NX Developer Kit. URL: https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit.

[5] Ouster OS-1-32 lidar. URL: https://ouster.com/products/scanning-lidar/os1-sensor/.

[6] ReSpeaker Mic Array v2.0. URL: https://wiki.seeedstudio.com/ReSpeaker_Mic_Array_v2.0/.

[7] SparkFun GPS-RTK2 Board - ZED-F9P (Qwiic). URL: https://www.sparkfun.com/products/15136.

[8] Stereolabs ZED 2i. URL: https://www.stereolabs.com/zed-2i/.

[9] TI IWR1443BOOST. URL: https://www.ti.com/tool/IWR1443BOOST.

[10] Dan Barnes, Matthew Gadd, Paul Murcutt, Paul Newman, and Ingmar Posner. The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6433–6438. IEEE, 2020.

[11] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[12] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8748–8757, 2019.

[13] Manfred Constapel, Marco Cimdins, and Horst Hellbrück. A practical toolbox for getting started with mmwave fmcw radar sensors. In *Proceedings of the 4th KuVS/GI Expert Talk on Localization*, 2019.

[14] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.

[15] Google. Turbo, An Improved Rainbow Colormap for Visualization. URL: https://ai.googleblog.com/2019/08/turbo-improved-rainbow-colormap-for.html.

[16] Jingzhou Liu, Wenhu Chen, Yu Cheng, Zhe Gan, Licheng Yu, Yiming Yang, and Jingjing Liu. Violin: A large-scale dataset for video-and-language inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10900–10910, 2020.

[17] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[18] Riko Suzuki, Hitomi Yanaka, Koji Mineshima, and Daisuke Bekki. Building a video-and-language dataset with human actions for multimodal logical inference. *arXiv preprint arXiv:2106.14137*, 2021.

[19] TI. DCA1000EVM. URL: https://www.ti.com/tool/DCA1000EVM.

[20] TI. MIMO Radar. URL: https://www.ti.com/lit/an/swra554a/swra554a.pdf.