CPSP - Dataset

- Riccardo Giani
- Tianyi Tan
- Mennatalla Hassan
- Zainab Zahara

Presentation’s Overview

1. Project’s Motivations & Objectives
2. System Architecture
3. Technical Implementation
4. Challenges & Solutions
5. Results & Validation
6. Future Steps

1 - Project’s Motivations & Objectives

Motivations:

- Lack of synchronized DVS + stereo vision datasets for dynamic environments
- Growing demand for neuromorphic vision in autonomous systems

Key Objectives:

✔️ Develop ROS2-based synchronization framework

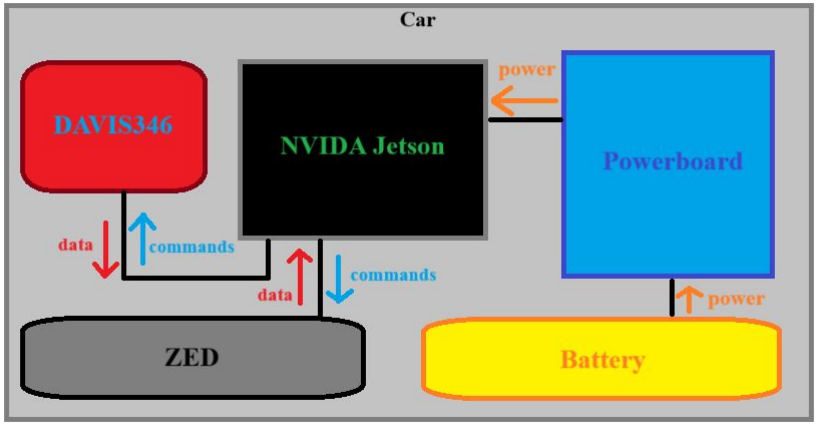
✔️ Capture ZED ground truth data with <5ms temporal alignment

✔️ Implement sync_saver node for unified dataset packaging

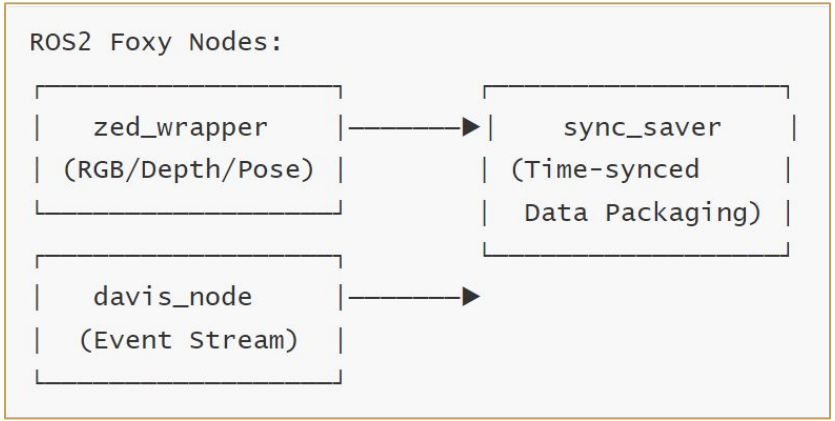
⚠️ DAVIS346 integration pending hardware repair

2 - System Architecture

Hardware Architecture



Software Architecture



3 - Technical Implementation (Hardware)

Computing elements

Data Acquisition:

- DAVIS346 (event camera)
- ZED Mini (stereo camera)

Synchronization & Elaboration:

- NVIDIA Jetson Orin Nano

Power Supply:

- Powerboard

Physical support & motion elements

Moving Around:

- Traxxas Slash 4X4 "Ultimate"
- car’s stock motor and servo
- car’s stock radio command

Power Supply:

- Lipo Battery

3 - Technical Implementation (Software)

Development Environment

- Core Stack: Jetpack 6.1 | CUDA 11.4 | Ubuntu 20.04
- SDKs: ZED SDK 4.0.8 | ROS2 Foxy | OpenCV 4.5

ROS2 Node Development

- Custom Nodes: sync_saver (Core synchronization node)

Precision Synchronization:

- ROS2 message_filters API

4 - Challenges & Solutions (#1)

Critical Issue 1: ROS2 Compilation Conflict

Conflict between ZED SDK 4.0.8 & ROS2 Foxy and OpenCV 4.5

- Uninstall cv_bridge that comes with ROS
- Compile cv_bridge from source code

4 - Challenges & Solutions (#2)

Critical Issue 2: Device selection and ROS version selection

- NIVIDIA Jetson only supported ubuntu up to 20.04 which only supports ROS versions until ROS foxy
- The available library interfacing the DVS uses ROS2 humble and its nodes aren't compatible with ROS2 foxy
- The stereo sensor library needs ROS2

Decision was made to use NIVIDIA Jetson with ROS2 foxy and solve the incompatibility issues with the library used for DAVIS camera interface

4 - Challenges & Solutions (#3)

Critical Issue 3: Incompatibility issues with ROS foxy and the DAVIS repo

Solutions included the following:

1. Several dependencies were needed for a successful installation, build, and launch
2. libcaer_driver repo
    - the repo was installed from source and the issues with incompatibility led to build errors but they were solved
3. Changes were made to cmake file, package.txt file, and package.xml

- Commands only supported in ROS humble were deleted or replaced with ROS foxy compatible commands

4 - Challenges & Solutions (#4)

Critical Issue 4: Receiving and recording data from DAVIS camera

1. Jetson unable to recognize DAVIS camera
   - Debugging process included checking camera datasheet, and reinstalling USB drivers on Jetson
2. Node for DAVIS camera functional but recieves no data
   - Some dependencies were installed to allow subscribing to the node and therefore recieves data

4 - Challenges & Solutions (#5)

Critical Issue 5: ROS2 Message Synchronization Not Triggering

- Topics publishing, node builds, but callback not firing

4 - Challenges & Solutions (#6)

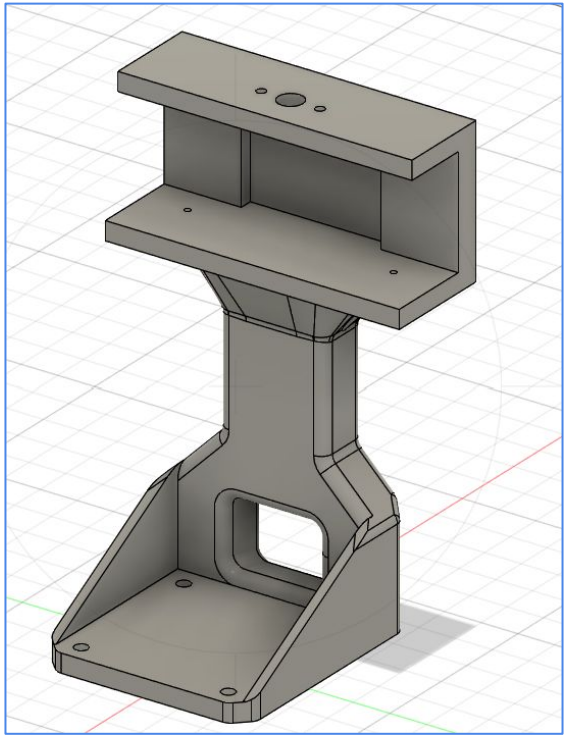
Critical Issue 6: acquisition of materials and **variations** from the original hardware design.

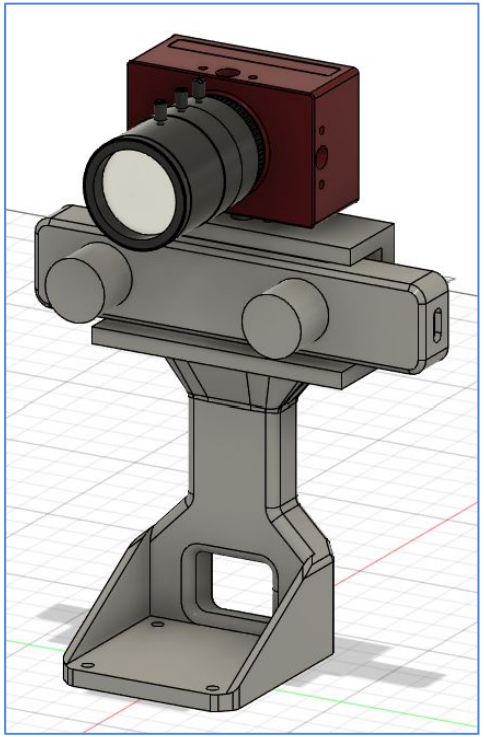
- Revision of the original bill of materials → no need for VESC and Lidar
- Some pieces not purchasable → found CAD files for platform deck (lasercut) and antenna mount (3D print)
- Powerboard was not complete → components soldered on the circuit board
- Car's motor and computing devices are now independent → crafted a 3-headed cable to power up the car and the Jetson separately
- WiFi module not present on the Jetson → acquired it separately

4 - Challenges & Solutions (#7)

Critical Issue 7: must add **DAVIS346** and **ZED** on the car.

Customized and 3D printed a **camera mount**.





5 - Results & Validation

Data structure

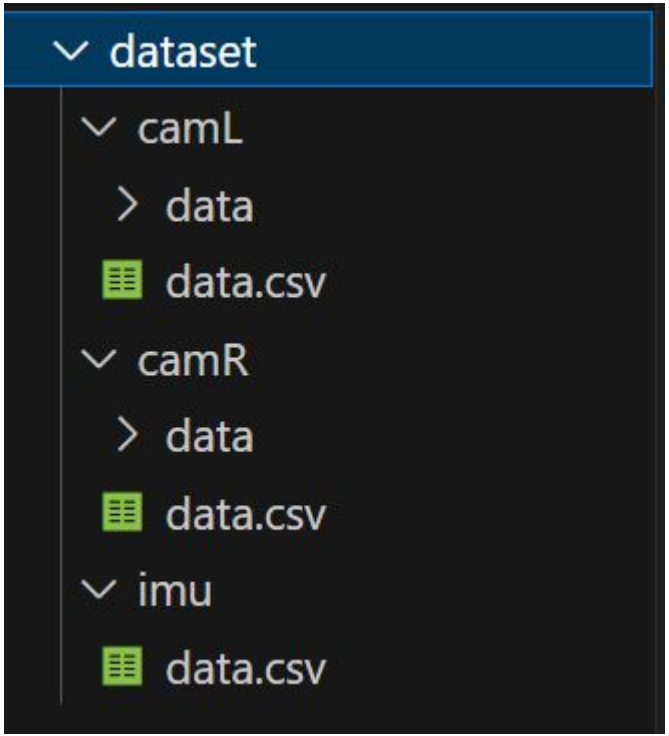
(Target)

```
dataset/
├── 0001/
|    ├── events/          # DVS events (*.??)
|    |    ├── data/             # events data file
|    |    ├── data.csv          # timestamp; filename
|    |    ├── data.yaml         # Sensor calibration
|    ├── rgb/             # ZED images (timestamped)
|    |    ├── data/             # image file
|    |    ├── data.csv          # timestamp; filename
|    |    ├── data.yaml         # Sensor calibration
|    └── stereo/          # ground-truth data (CSV)
|    |    ├── data.csv          # timestamp; ground-truth data
|    |    ├── data.yaml         # Sensor calibration
```

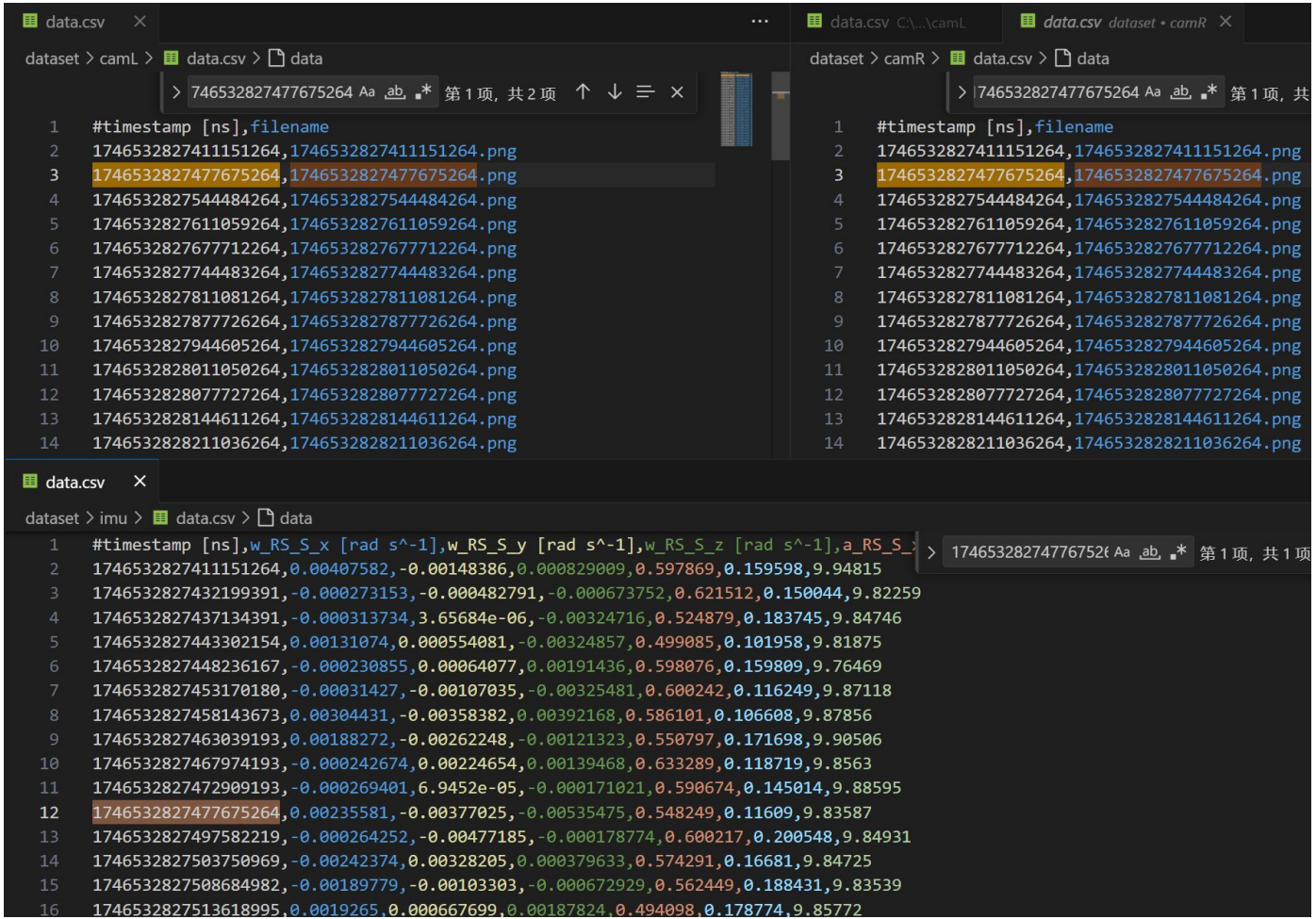
5 - Results & Validation

Data structure

(stereocamera)



Synchronization Validation:



data.csv

dataset > camL > data.csv > data

1746532827477675264 Aa ab .* 第 1 项，共 2 项

```
#timestamp [ns],filename
1746532827411151264,1746532827411151264.png
1746532827477675264,1746532827477675264.png
1746532827544484264,1746532827544484264.png
1746532827611059264,1746532827611059264.png
1746532827677712264,1746532827677712264.png
1746532827744483264,1746532827744483264.png
1746532827811081264,1746532827811081264.png
1746532827877726264,1746532827877726264.png
1746532827944605264,1746532827944605264.png
1746532828011050264,1746532828011050264.png
1746532828077727264,1746532828077727264.png
1746532828144611264,1746532828144611264.png
1746532828211036264,1746532828211036264.png
```

data.csv C:\...\camL

data.csv dataset • camR

dataset > camR > data.csv > data

1746532827477675264 Aa ab .* 第 1 项，共

```
#timestamp [ns],filename
1746532827411151264,1746532827411151264.png
1746532827477675264,1746532827477675264.png
1746532827544484264,1746532827544484264.png
1746532827611059264,1746532827611059264.png
1746532827677712264,1746532827677712264.png
1746532827744483264,1746532827744483264.png
1746532827811081264,1746532827811081264.png
1746532827877726264,1746532827877726264.png
1746532827944605264,1746532827944605264.png
1746532828011050264,1746532828011050264.png
1746532828077727264,1746532828077727264.png
1746532828144611264,1746532828144611264.png
1746532828211036264,1746532828211036264.png
```

data.csv

dataset > imu > data.csv > data

```
#timestamp [ns],w_RS_S_x [rad s^-1],w_RS_S_y [rad s^-1],w_RS_S_z [rad s^-1],a_RS_S_
1746532827411151264,0.00407582,-0.00148386,0.000829009,0.597869,0.159598,9.94815
1746532827432199391,-0.000273153,-0.000482791,-0.000673752,0.621512,0.150044,9.82259
1746532827437134391,-0.000313734,3.65684e-06,-0.00324716,0.524879,0.183745,9.84746
1746532827443302154,0.00131074,0.000554081,-0.00324857,0.499085,0.101958,9.81875
1746532827448236167,-0.000230855,0.00064077,0.00191436,0.598076,0.159809,9.76469
1746532827453170180,-0.00031427,-0.00107035,-0.00325481,0.600242,0.116249,9.87118
1746532827458143673,0.00304431,-0.00358382,0.00392168,0.586101,0.106608,9.87856
1746532827463039193,0.00188272,-0.00262248,-0.00121323,0.550797,0.171698,9.90506
1746532827467974193,-0.000242674,0.00224654,0.00139468,0.633289,0.118719,9.8563
1746532827472909193,-0.000269401,6.9452e-05,-0.000171021,0.590674,0.145014,9.88595
1746532827477675264,0.00235581,-0.00377025,-0.00535475,0.548249,0.11609,9.83587
1746532827497582219,-0.000264252,-0.00477185,-0.000178774,0.600217,0.200548,9.84931
1746532827503750969,-0.00242374,0.00328205,0.000379633,0.574291,0.16681,9.84725
1746532827508684982,-0.00189779,-0.00103303,-0.000672929,0.562449,0.188431,9.83539
1746532827513618995,0.0019265,0.000667699,0.00187824,0.494098,0.178774,9.85772
```

17465328274776752(Aa ab .* 第 1 项，共 1 项

6 - Future Steps

✔️ **Completed tasks**:

- Test the car assembly and write down the list of missing parts (**Riccardo**)
- Re-asseble the existing ESC module, test the car with the RC control, impose safe speed limit (**Zainab**)
- Assemble the car with missing pieces and add the two cameras (**Riccardo**)
- Testing DVS with laptop (**Riccardo**)
- Python script to read DVS data (**Riccardo**)
- Read stereocamera from laptop (**Tianyi**)
- Setup NVIDIA Jetson (**Mennatalla**)
- Read DVS from Jetson using ROS2 (**Mennatalla + Tianyi**)
- Read sterocamera (ground-truth) from Jetson using ROS2 (**Tianyi**)
- In ROS2 sychronize the acquisition of DVS and stereo data (**Zainab - ongoing**)

➡️ **Next steps**:

- Choose/setup the data acquisition environment
- Drive around car with onboard sensor to acquire the dataset

Thanks for your attention!