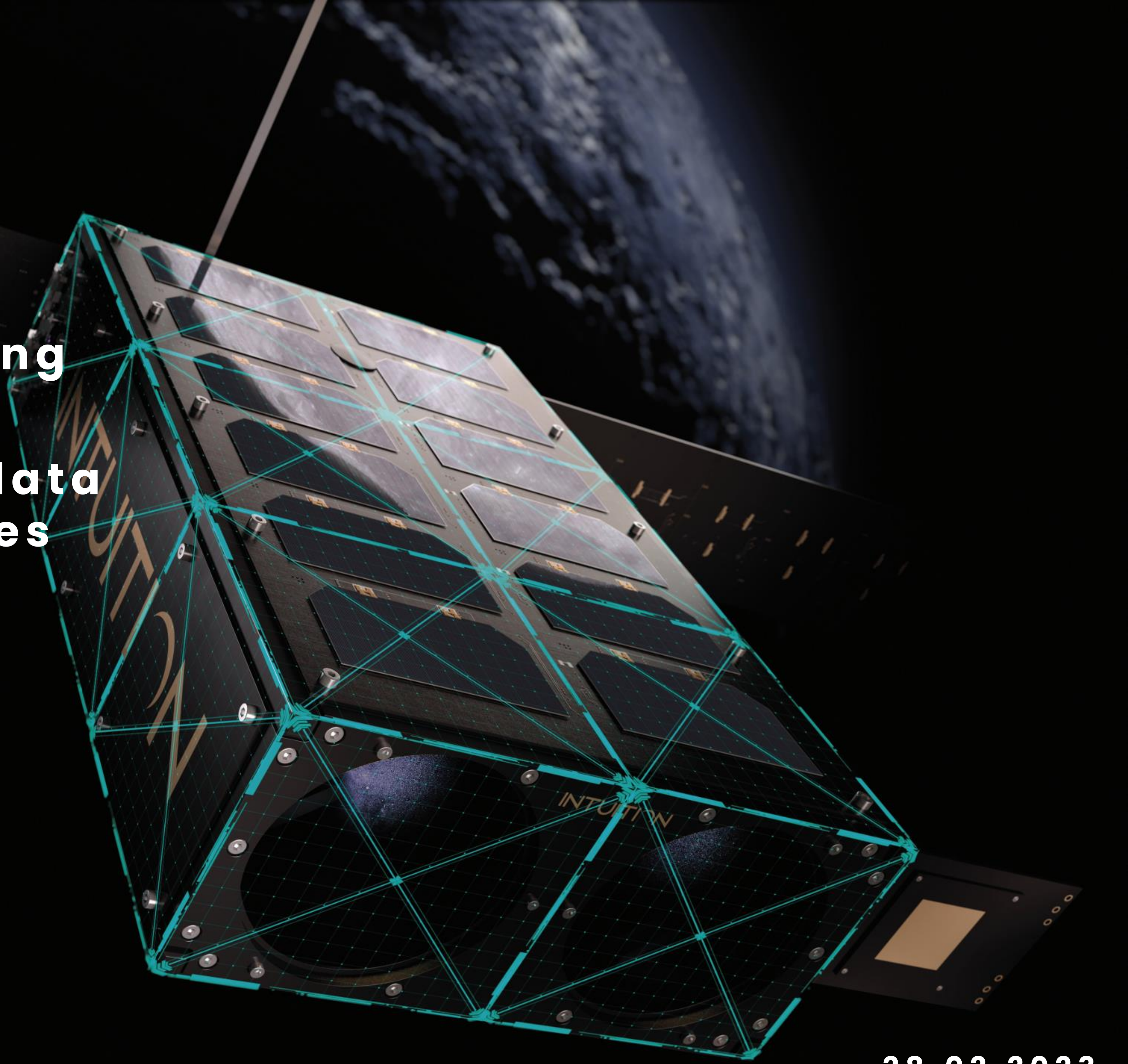
Final presentation meeting

Cognition - distributed data system for lunar activities processing



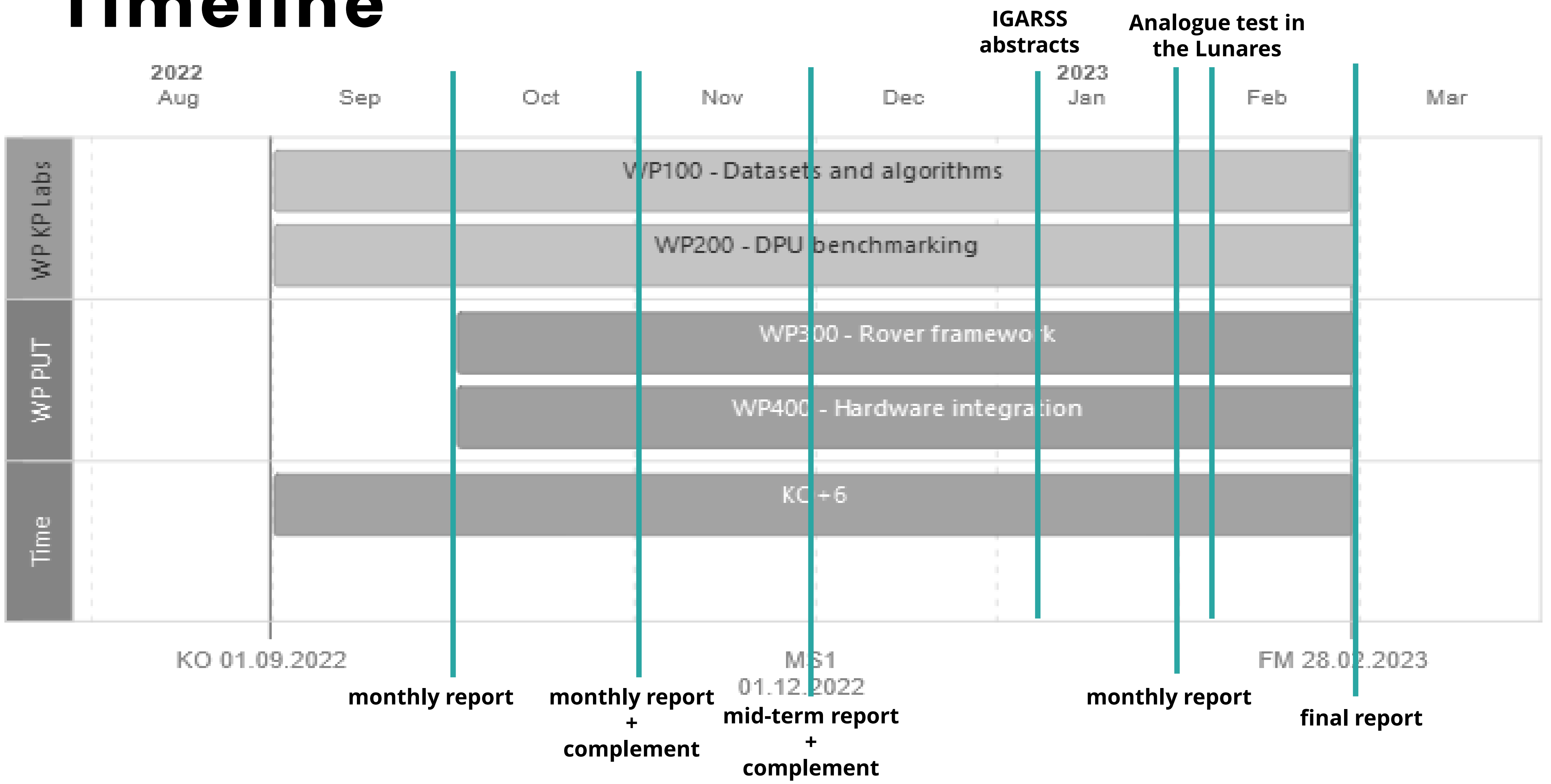
www.kplabs.pl

28.02.2023

Meeting Agenda

- Timeline
- Background and Objectives
- Work logic diagram
- Achieved outcomes:
  - WP100
  - WP200
  - WP300
  - WP400
- Q/A

Timeline


IGARSS abstracts
Analogue test in the Lunares
2022
Aug
Sep
Oct
Nov
Dec
2023
Jan
Feb
Mar
WP KP Labs
WP PUT
Time
WP100 - Datasets and algorithms
WP200 - DPU benchmarking
WP300 - Rover framework
WP400 - Hardware integration
KO +6
KO 01.09.2022
MS1
01.12.2022
FM 28.02.2023
monthly report
monthly report
+
complement
mid-term report
+
complement
monthly report
final report

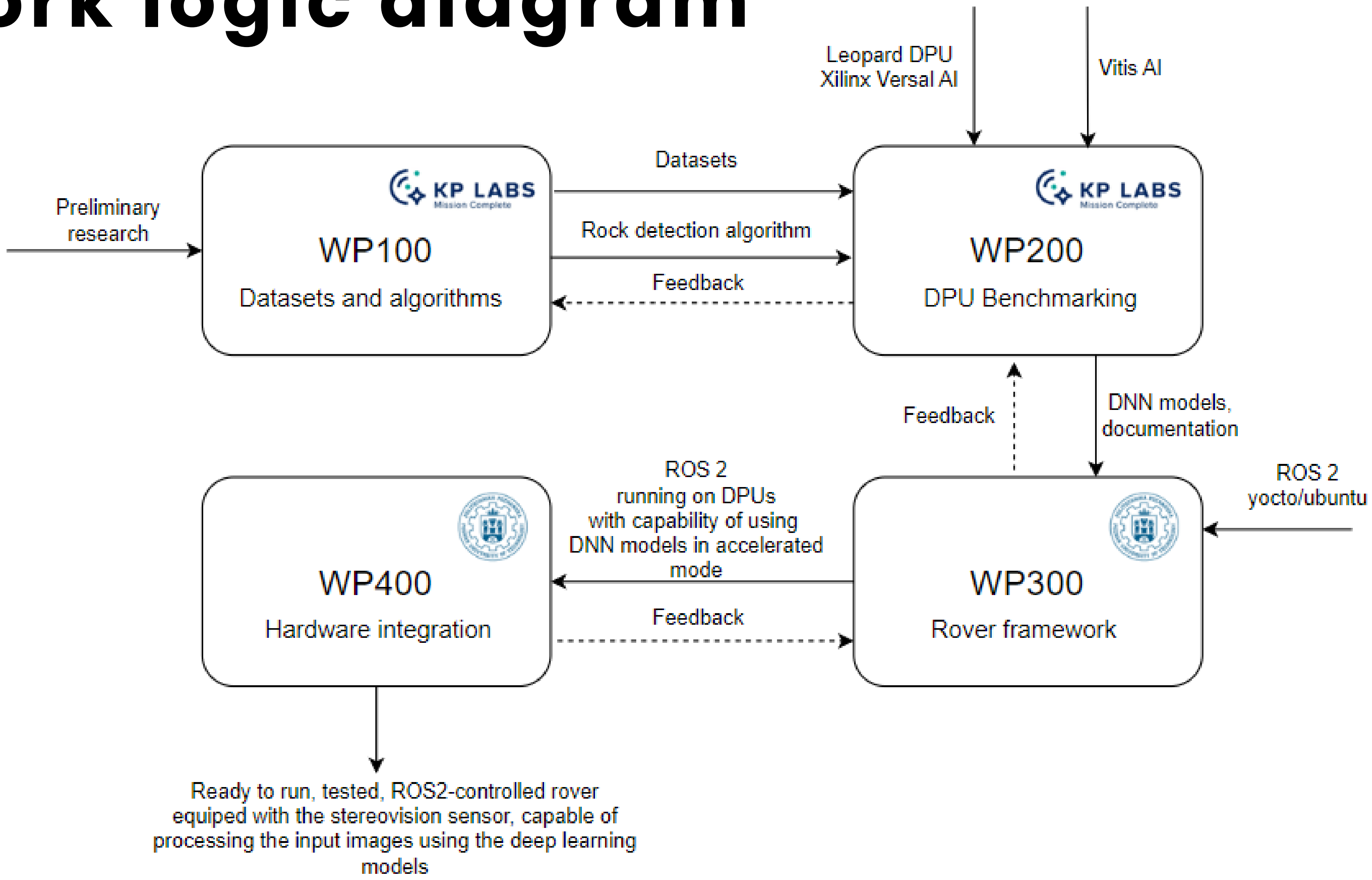
Background

The growing number of lunar missions and limitations in Moon-Earth communications create the need for a DPU capable of processing at least some of the data on the lunar surface, thereby reducing data transfer to Earth and increasing rover autonomy.

Objectives

- #1 To explore the capabilities of the AI development environment from Xilinx and benchmark two architectures (Leopard DPU and Versal AI)
- #2 To analyse the possibility of running ROS on limited resources
- #3 To perform analogue tests with a DPU and a stereovision camera
- #4 To define the architecture for a future distributed processing system

Work logic diagram



Achieved outcomes

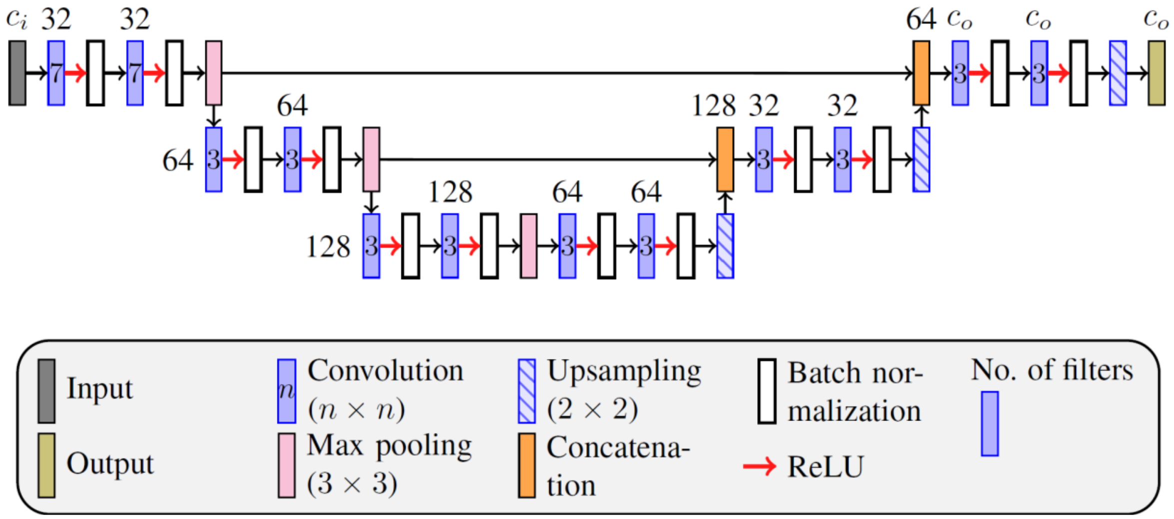
WP100 – Datasets and algorithms

- The goals:
  - To prepare a dataset for training and validating rock segmentation and detection
  - To select and train a deep model for rock detection suitable for further deployment
- The algorithms considered in our approach:
  - A lightweight U-Net model for rock segmentation
  - YOLO v5 models (X-sized, S-sized, Nano-sized)
  - A sequential approach (U-Net segmentation maps fed into YOLO)
    (an abstract submitted to IGARSS 2023)
  - Data preprocessing and augmentation based on adaptive histogram equalization (CLAHE) and gamma corrections
- The datasets used for training and validation
  - Artificial Lunar Landscape Dataset (ALLD) – almost 10k images, split into training, validation and test subsets
  - Real-world images (several hundreds annotation masks) collected at Lunares Research Station (quite different from ALLD images and from actual Lunar images)

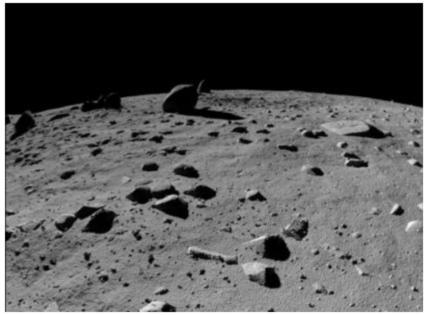
WP100 – Datasets and algorithms

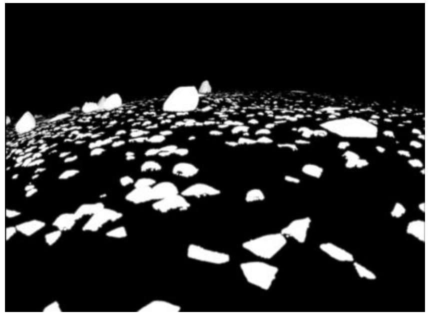
- The lightweight U-Net architecture adapted from:

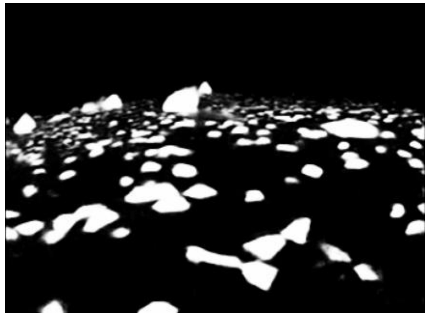
Grabowski, B., Ziaja, M., Kawulok, M., & Nalepa, J. (2021). Towards robust cloud detection in satellite images using U-Nets. In *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS* (pp. 4099-4102). IEEE.



- Results for ALLD:


Input image


Ground-truth mask


Segmentation outcome

Stage	Loss	Precision	Recall	Dice	Jaccard	Dataset
float	0.3097	0.6764	0.7552	0.6977	0.5678	ALLD
quant	0.3101	0.6929	0.7428	0.7009	0.5722	ALLD
compiled	0.3085	0.6966	0.7399	0.7017	0.5733	ALLD
float	0.8527	0.1724	0.6884	0.2199	0.1291	Lunares

We can appreciate there is no difference in the quality of floating and compiled models for the ALLD dataset

WP100 – Datasets and algorithms

- The results obtained for images acquired at Lunares Research Station:

Input image

Ground-truth mask

Segmentation outcome (model trained from ALLD images)

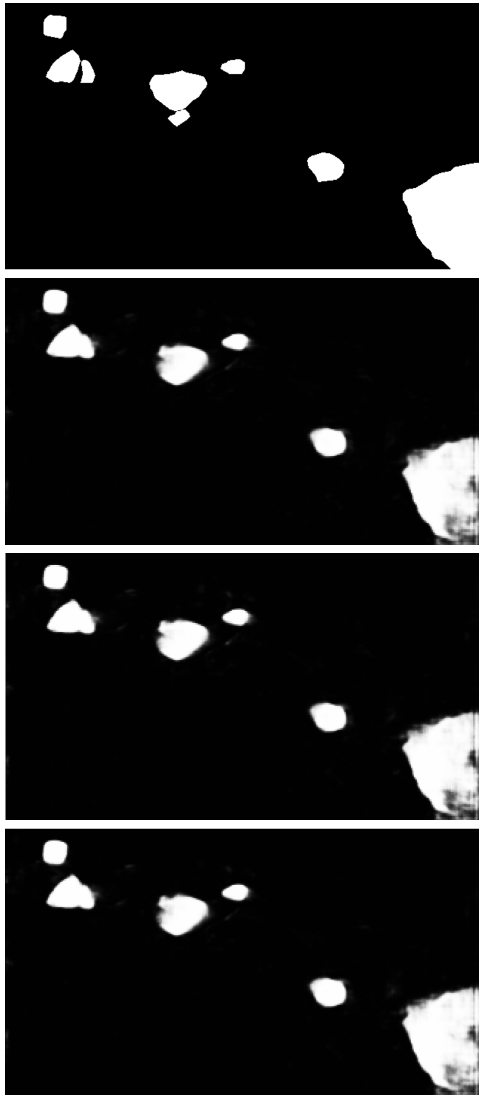
Segmentation outcome (after fine-training)

- Conclusions:
  - The segmentation maps produced by U-Net can be used for rock detection
  - The model can be easily adapted to different lighting conditions (Dice coefficient: 0.22 $\rightarrow$ 0.64)

WP100 – Datasets and algorithms

- Quantization-oriented degradation (based on real-life Lunares dataset):
  - From top to bottom: ground-truth, floating model, quantized model, compiled model
  - Deployment-oriented degradation is visually negligible, yet it can be spotted in metrics
  - Results seem to be very consistent between quantized and compiled models

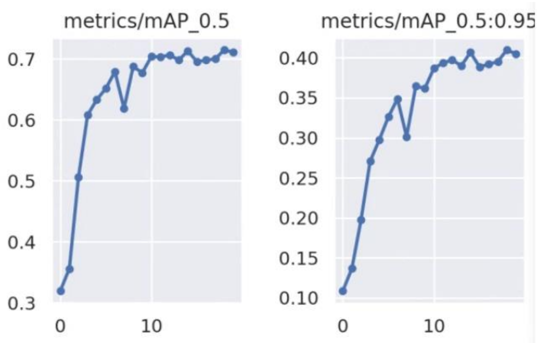
Stage	PowerJaccard	Precision	Recall	DiceCoeff	JaccardIndex	Dataset
float	0.4328	0.7932	0.6322	0.6527	0.5290	Lunares
quant	0.4449	0.7607	0.6397	0.6421	0.5160	Lunares
compiled	0.4434	0.7749	0.6339	0.6426	0.5174	Lunares



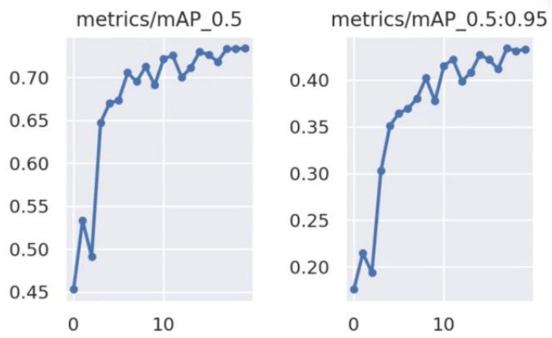
WP100 – Datasets and algorithms

- YOLO v5 (Nano size) fed with input images with U-Net segmentation maps

Results for the validation set during training:

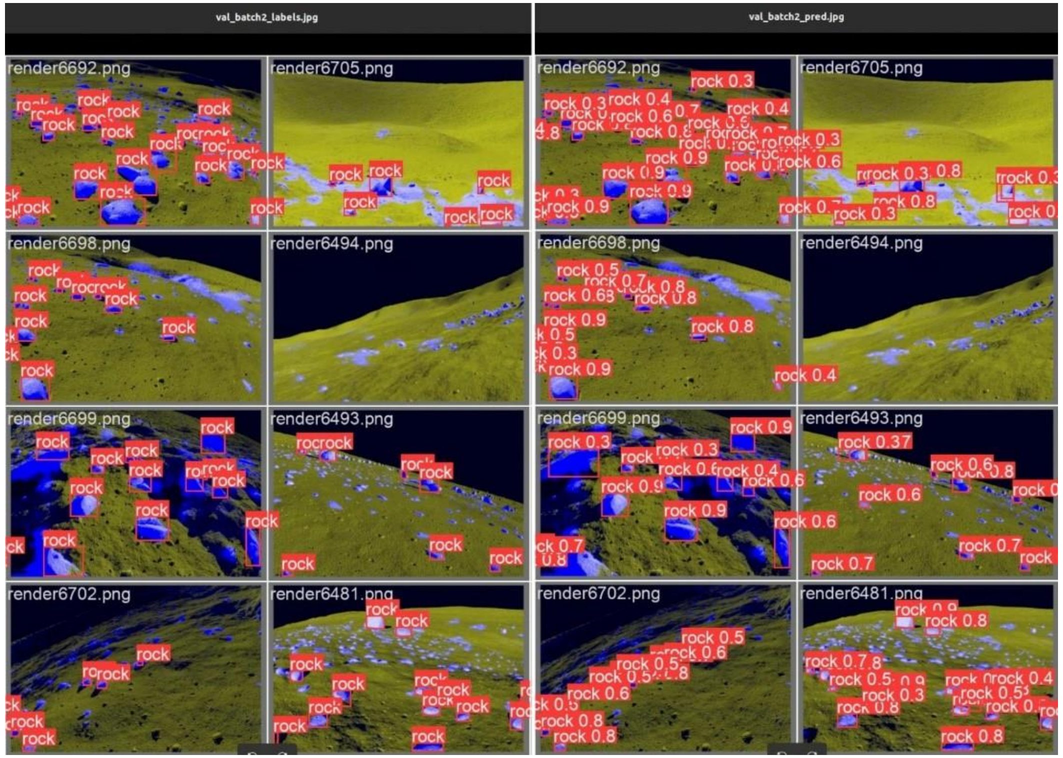


Input image without auxiliary information



Input image with U-Net segmentation maps

Qualitative results (with U-Net segmentation maps)

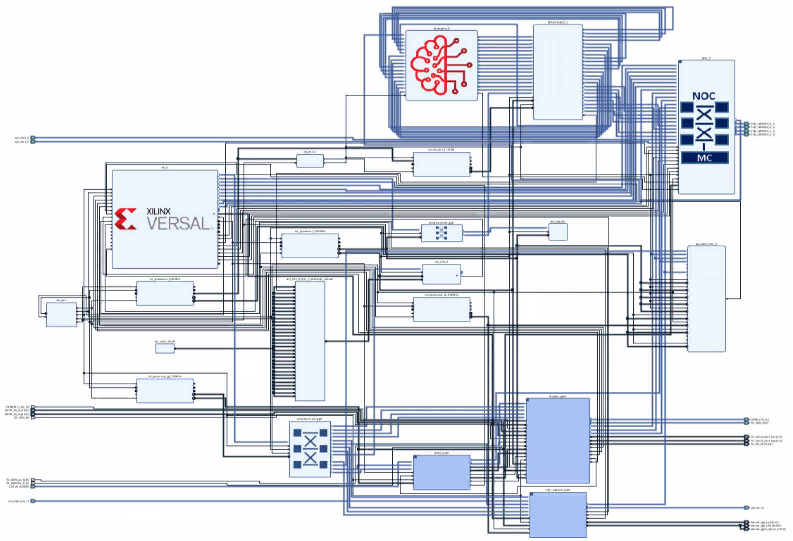


WP200 – DPU Benchmarking

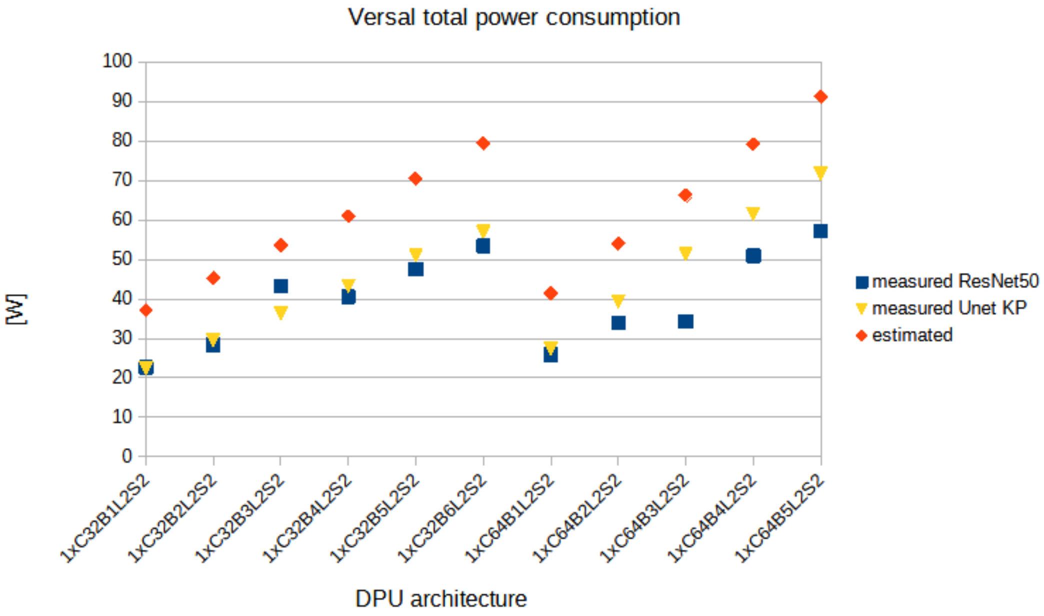
- The goals and tasks of WP200:
  - DPU synthesis and system setup for Versal vck190 device
  - Model deployment from pytorch to xmodel
  - Benchmarking Versal vck190 and Leopard EBB devices focusing on power consumption and inference throughput

WP200 – DPU Benchmarking

- Versal system design is based on a Xilinx reference design.
- A range of DPU configurations integrated:
  - 1xC32B1L2S2, 1xC32B2L2S2, 1xC32B3L2S2, 1xC32B4L2S2, 1xC32B5L2S2, 1xC32B6L2S2
  - 1xC64B1L2S2, 1xC64B2L2S2, 1xC64B3L2S2, 1xC64B4L2S2, 1xC64B5L2S2



- Power consumption estimations were extracted and compared with benchmark results.

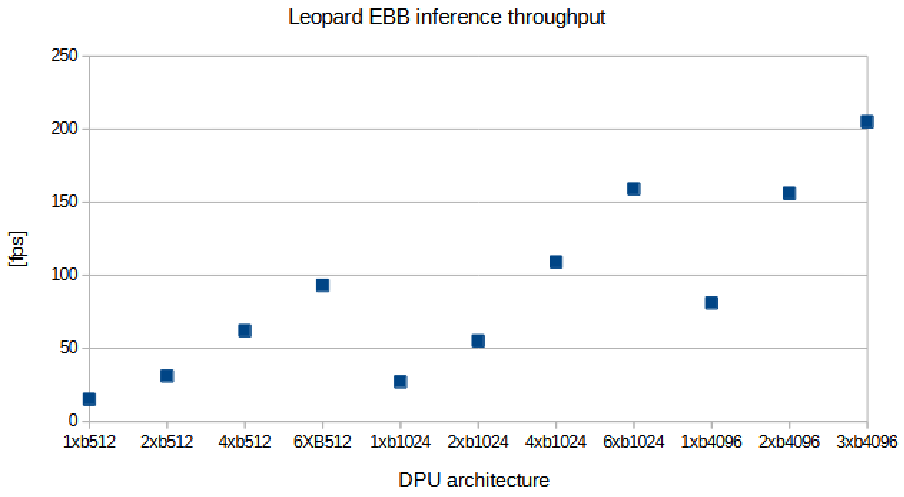


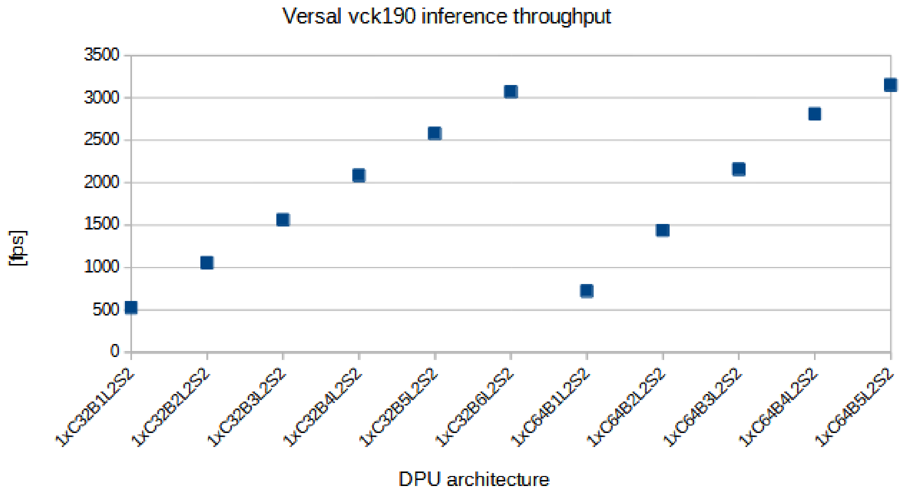
WP200 – DPU Benchmarking

- Deployment of a custom UNet model was done in Vitis AI 2.5 environment.
- The quantization step has proven to be error prone. Quantization of some of the model layers would compromise inference results or even make compilation impossible. The original UNet model had to be modified to accommodate Vitis AI restrictions.
- The quantized model was compiled for all Versal architectures:
  - 1xC32B1L2S2, 1xC32B2L2S2, 1xC32B3L2S2, 1xC32B4L2S2, 1xC32B5L2S2, 1xC32B6L2S2
  - 1xC64B1L2S2, 1xC64B2L2S2, 1xC64B3L2S2, 1xC64B4L2S2, 1xC64B5L2S2

WP200 – DPU Benchmarking

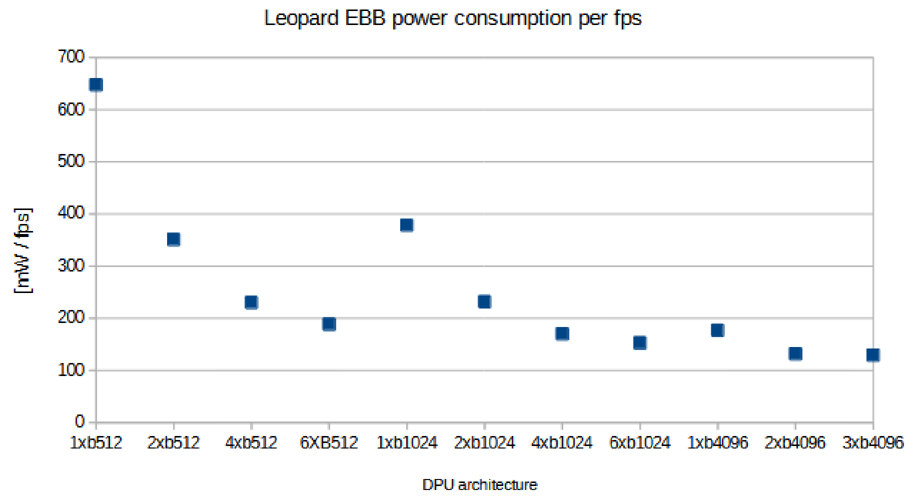
- Inference throughput
  - Initial benchmarks were performed with a reference ResNet50 (224x224) network.
  - Benchmarks were performed with xdputil utility tool.
  - Versal's maximum throughput reached 15 times higher value, than Leopard's.

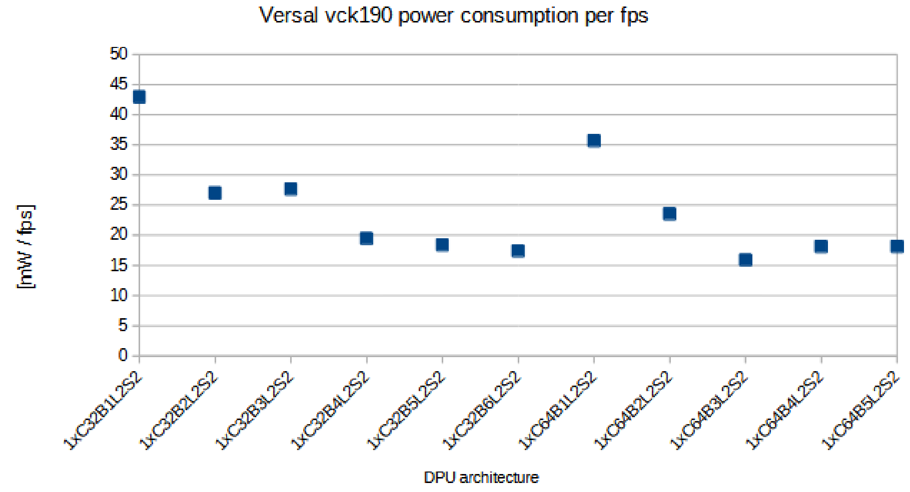




WP200 – DPU Benchmarking

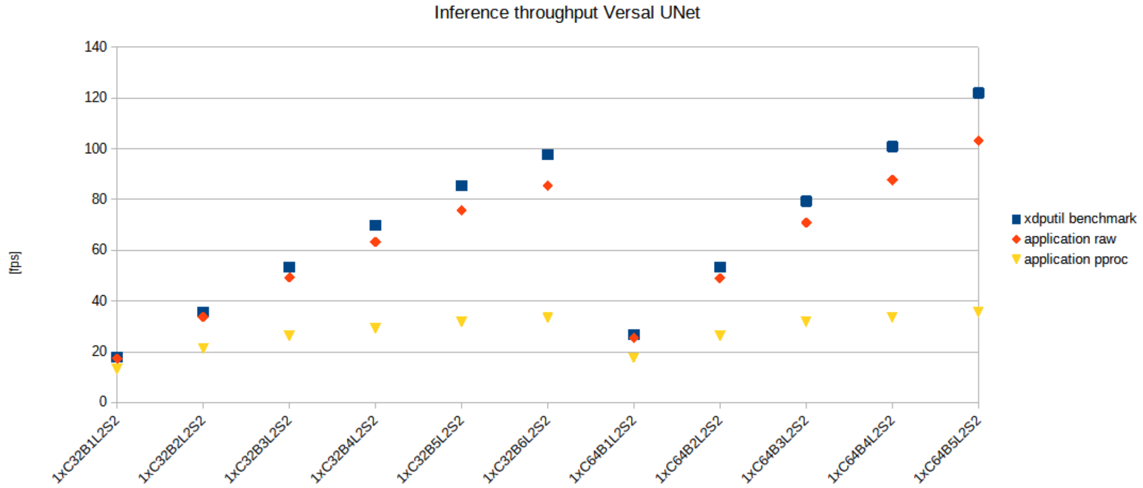
- Power consumption per fps
  - Initial benchmarks were performed with a reference ResNet50 (224x224) network.
  - Power was measured with vck190 Power Tool.
  - Versal's minimum power consumption per fps was 7 times lower than for Leopard EBB.

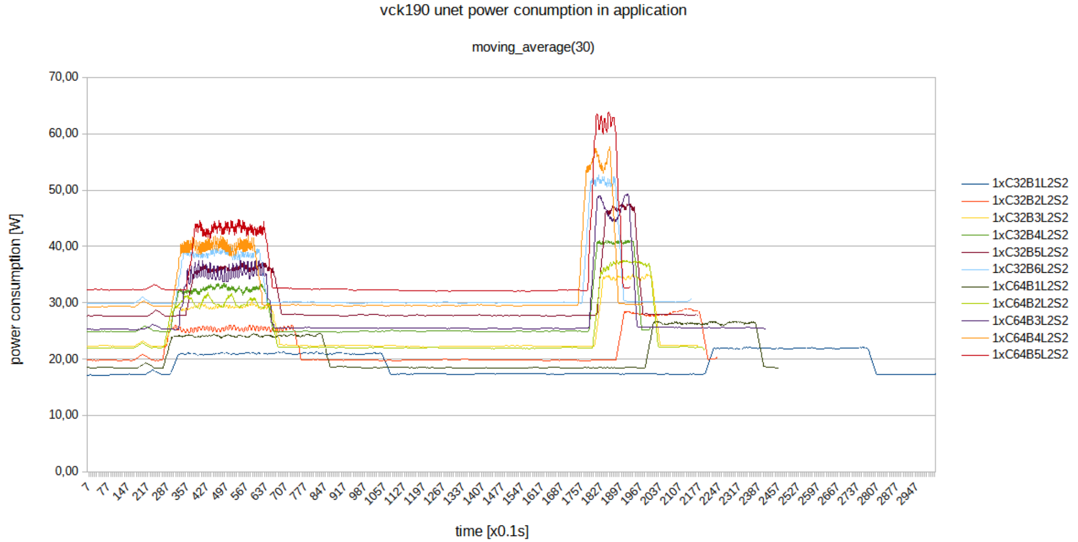




WP200 – DPU Benchmarking

- A python module for inference was developed and tested with a standalone application.
- The application yielded correct inference results.
- Inference throughput and power consumption was measured for the application in modes with and without pre- and postprocessing.



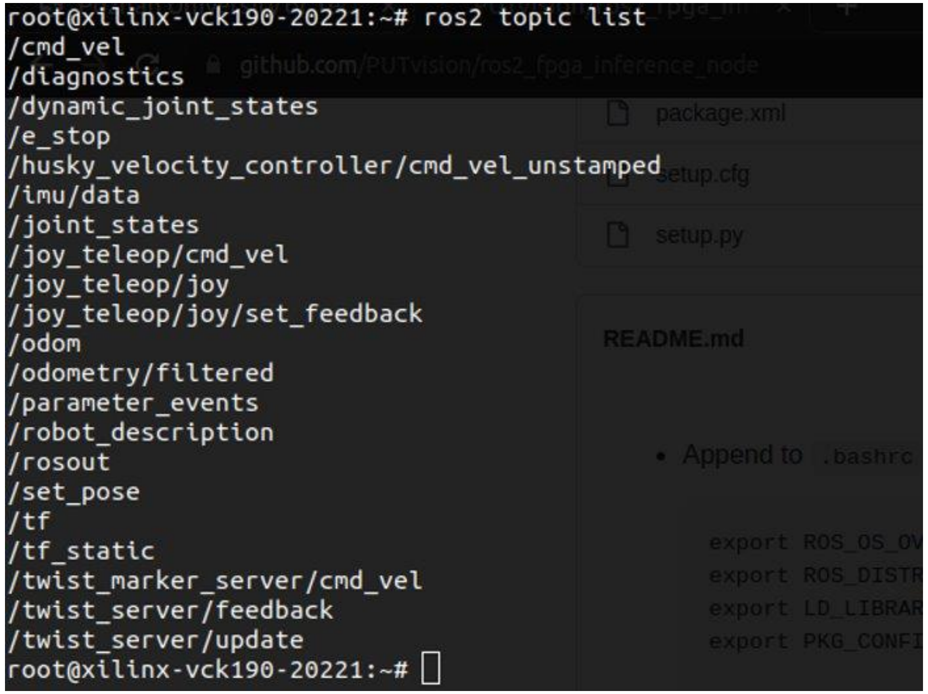


WP300 – Rover framework

- The TR#3 states the goals for WP300:
  - To perform an analysis of existing lightweight ROS 2 variants (forks), possibly designed to run on embedded hardware (micro-ROS, riot-ros2). The forks will be analyzed in the context of the feasibility of running on the selected DPU platform. To test the integration of deep learning coprocessors implemented using programmable hardware as ROS2 endpoints within the ROS-native publisher-subscriber communication model.
- So, to meet the requirement, we need to satisfy two output goals:
  - Make ROS2 run on the DPU processors
  - Integrate the capability of using the deep neural network models running in accelerated mode using the programmable logic part of the DPU as ROS 2 native mechanisms

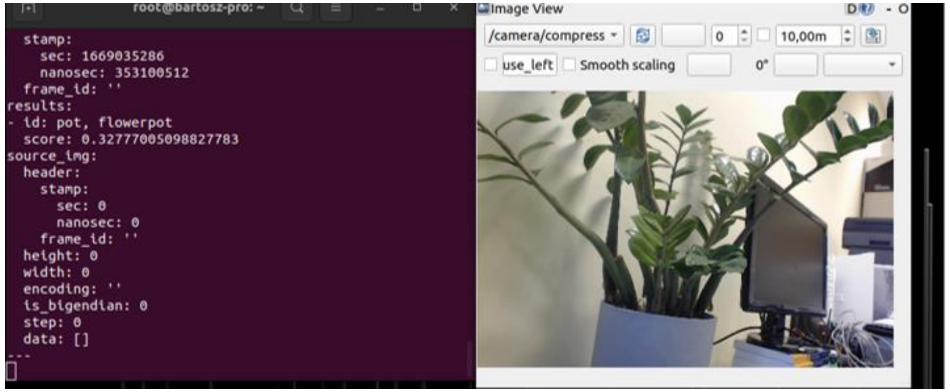
WP300 – Rover framework

- ROS 2 integration - obtained results:
  - It was possible to run the fully functional ROS 2 without problems, so other variants were not tested
  - The Petalinux-based image was equipped with Robot Operating System (ROS 2 Foxy) and successfully handle all sanity checks on the VCK190 prototype board.
  - ROS components such as publishing, subscribing, script running and launch executing were verified.
  - Petalinux's ROS-compatible meta-layers for the Husky robot, Luxonis camera, and their dependencies have been created.
- This satisfies the requirements for the first stated goal



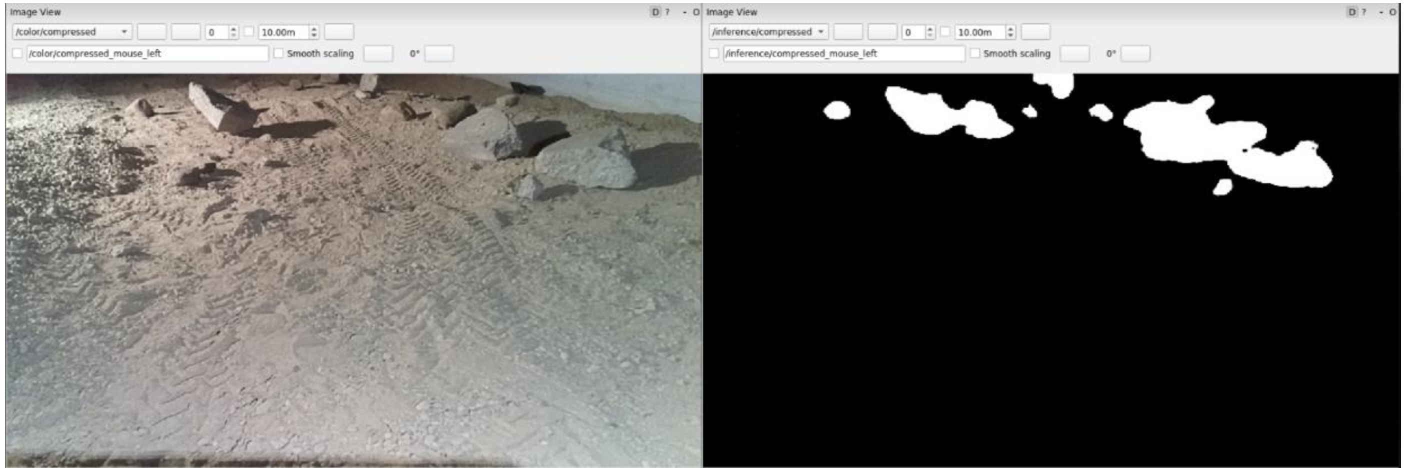
WP300 – Rover framework

- DPU coprocessor integration - obtained results:
  - Wrappers facilitating the use of neural network hardware coprocessors running inference as native ROS 2 services were prepared.
  - They can use the camera publisher as the source of image data for neural network inference.
  - ResNet50 wrapper was developed for initial testing, and UNet wrapper was developed as a key part of the final application.
  - Final goal was to detect and segment rocks on the simulated lunar surface during analogue mission, so all the components (sensor input, rover control, DPU hardware coprocessors) were integrated into a complete application.
- This satisfies the requirements for the second stated goal



WP300 – Rover framework

- Everything worked as intended during the analogue mission, as confirmed by the preview from the developed visualization tools



- All the goals and technical requirements envisioned for WP300 were met

WP300 – Rover framework

- Lessons learned:
  - A dedicated machine for embedded Linux and ROS 2 development is very useful; in some cases, 2 or 3 such machines would be a welcome addition to test multiple approaches simultaneously.
  - Neural networks prepared specifically for embedded hardware introduce an additional interface layer due to optimizations. This calls for careful handling of data types, output activations, input and output data normalization and denormalization and digging deep in VITIS AI documentation.
  - ROS 2 provides a significant upgrade when compared to ROS 1 in terms of development tools and standardization, but still lags in terms of several readily available components.

WP400 – Hardware integration

- The main goal for WP400:

  Preparation of a mobile platform with the necessary sensors and devices to perform analog tests.

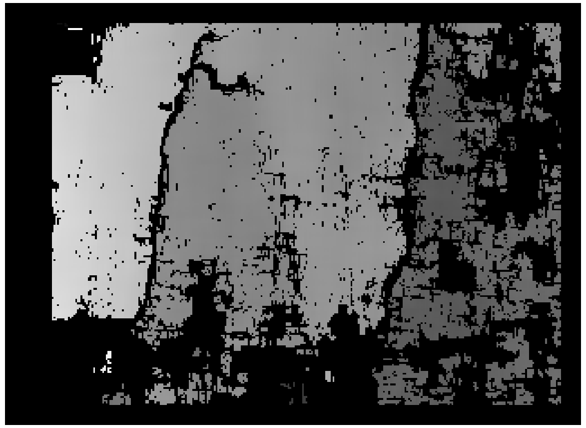
- This task consists of several parts:
  - modification of the Clearpath Husky A200 robot to mount the necessary devices, connect and power them
  - preparation of software to control robot remotely using ROS2
  - preparation of software for sensors: stereo camera and IMU

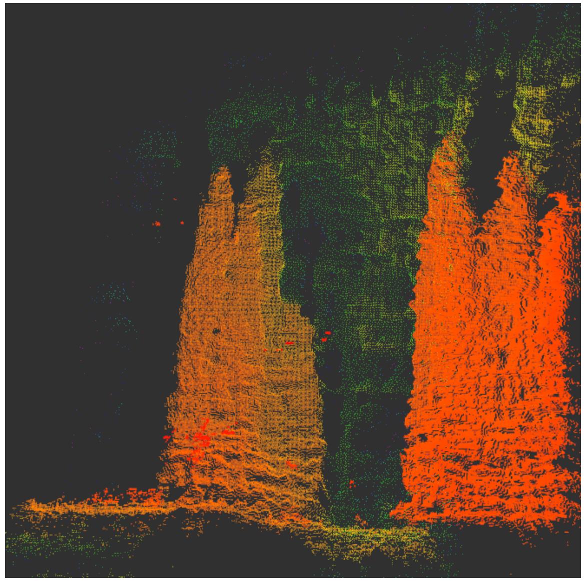
WP400 – Hardware integration

- A frame was built on the robot to mount the following devices: Wi-Fi/Ethernet router, orange emergency light, antennas, receiver for remote safety button and stereo camera.
- Additional power outputs were prepared, for all mounted components .
- The new software stack of Ubuntu 20.04. with ROS2 Foxy was set up to meet the project requirements.

WP400 – Hardware integration

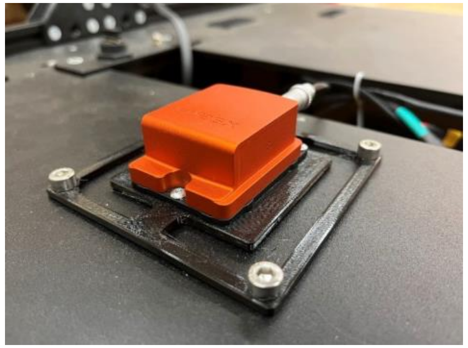
- Stereo camera:
  - Two stereo cameras were tested.
  - ROS2 nodes were prepared to publishing RGB and depth image, and point cloud.
  - For analog tests, the OAK-D-Lite camera was used.
  - Camera was calibrated to determine cameras' intrinsic parameters and relative positions of lenses.
  - In order to speed up the operations on images, some of them were performed on the VPU on camera.
  - RGB image was resized and compressed, which allowed to eliminate delays in data transfer.
  - The prepared stack ensures continuous and stable image transmission and worked well during tests.

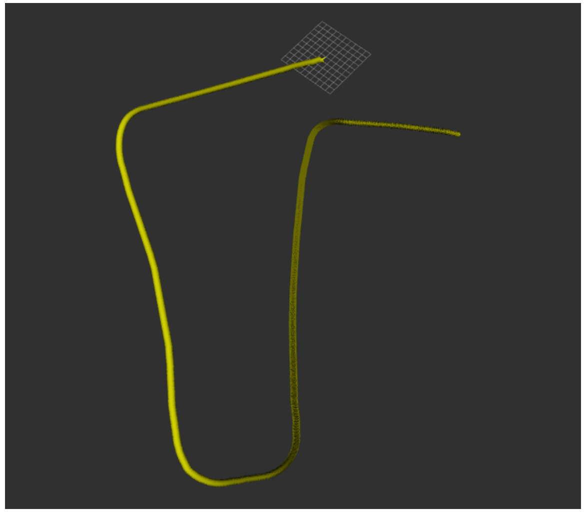




WP400 – Hardware integration

- The XSENS 300MTI IMU was mounted on the robot base, and the data from the unit were fed into an Extended Kalman Filter alongside the wheel odometry to improve localization in sloping and unstable terrain.
- During tests proved that IMU sensor is resilient in the sloppy terrain and keeps track of the localization of the rover.





WP400 – Hardware integration

- Lessons learned:
  - The software stack requires a proper startup procedure, considering the difference in booting times of different components and dependencies between them. It is also important to avoid undervoltage on the router and sensors.
  - It is important to adjust frequency, resolution and format of images to provide smooth data transfer.
  - To get a better quality of depth image, it is necessary to ensure proper lighting (passive stereo camera), so additional lighting was mounted on the robot.
  - The rover performed well during the tests in the LunAres Research Station– no issues with locomotion in rough terrain.
  - The software of robot, camera and IMU worked flawlessly.

Q&A