Real-Time Landform Feature Detection and Segmentation Based on Heterogeneous MPSoCs for Lunar Robotic Exploration

Qichen Fan, Ran Duan, Bo Wu*, Hao Zhou, Siqing Zhang, Yuan Ma

Research Centre for Deep Space Explorations | Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong – bo.wu@polyu.edu.hk

Keywords: Lunar robots, Hardware accelerator, Heterogeneous Computing, Edge Computing

Abstract

Real-time feedback and operation are crucial for the next generation of lunar rovers and robots, enabling more powerful and intelligent exploration. Traditional ground control methods face challenges in meeting the demands of real-time navigation and exploration for lunar rovers and robots due to issues such as latency and communication inefficiencies. The electronic systems of lunar rovers and robots are constrained by limited power supply, necessitating energy-efficient solutions. Multiprocessor System-on-Chips (MPSoCs) are considered capable of balancing the power consumption and performance requirements for lunar robotic exploration missions. MPSoCs integrate multiple processor cores and other components, such as memory and peripheral interfaces, and can include various processors such as ARM cores and Field-Programmable Gate Arrays (FPGAs). FPGA-based MPSoCs are highly customizable and energy-efficient, making them advantageous for addressing the challenges of edge computing tasks in lunar robotic exploration. In this paper, we present neural network inference accelerators using Vitis-AI on an FPGA heterogeneous MPSoC for lunar landform analysis, including landform feature detection and segmentation. Feature detection is implemented by the YOLOv5s-MobileNetV2 network, and segmentation is based on a Feature Pyramid Network (FPN). The dataset consists of images of the lunar surface taken by various lunar exploration missions. By utilizing inference acceleration and an improved neural network design, our accelerators demonstrate superior performance in energy efficiency, inference speed, and accuracy. The developments have been deployed on an edge computing device, the Xilinx Kria KV260 MPSoC platform, and experimental results show that the accelerators achieved a feature detection frame rate of 52 FPS and a segmentation frame rate of 27 FPS. The accuracy of the project meets or exceeds that of popular deep-learning models. The system operates at around 6.3 W of power for the detection task and 7.6 W for segmentation, making it energy efficient. The results suggest that such a system can be deployed in future lunar rovers and robots to enhance the effectiveness of exploration tasks.

1. Introduction

The Moon serves as an outpost for deep space exploration. In recent years, various countries have been actively advancing their plans for lunar landing missions (De Rosa et al., 2012; Wu et al., 2014, 2020; Wang et al., 2024). Among these initiatives, NASA's Artemis Program encompasses a series of robotic and manned missions, aiming to establish a sustainable research base at the Moon's south pole (Smith et al., 2020). China's forthcoming lunar exploration missions, including Chang'e 7 and Chang'e 8, will deploy lunar surface rovers and robots to explore the lunar south pole and initiate the construction of a future scientific research station there (Wang et al., 2024). Additionally, other countries, such as India and Russia, have also planned or launched their own lunar landing programs.

For future lunar robotic exploration missions, traditional ground control methods encounter challenges in meeting the demands of real-time navigation and exploration for lunar rovers and robots, primarily due to latency and communication inefficiencies. Real-time feedback and operation are essential for the next generation of lunar rovers and robots, facilitating more powerful and intelligent exploration. Among the key technologies, the real-time analysis of lunar landform features to identify interesting targets and obstacles is a critical aspect (Wu et al., 2021).

Considering the limited power supply and the need for hardware reconfigurability, energy-efficient FPGA-based Multiprocessor System-on-Chips (MPSoCs) are favorable choices for implementing intelligent landform analysis. Despite their advantages in energy efficiency and reconfigurability, this approach presents several engineering challenges. Firstly, such hardware typically supports only a limited variety of operators, imposing many design constraints on networks. For instance, some hardware introduced by Xilinx does not support 3D convolution directly. Secondly, the computational power of this hardware is generally limited, making it suitable only for deploying lightweight models. Additionally, deploying models on such hardware platforms is more complex than on other computational platforms, posing a challenge for developers (Mohaidat et al., 2024).

To address these issues, this paper presents an edge computing system for real-time lunar landform feature detection and segmentation, utilizing MPSoCs. The system uses a YOLOv5s-based lightweight network for landform feature detection, adapted to the Xilinx KV260 MPSoC. For feature segmentation, it chooses the Feature Pyramid Network (FPN) due to its lower computational overhead. After training the detection and segmentation models, the network was further compressed. The resulting network was then deployed to the DPUCZDX8G Deep Learning Processor Unit (DPU) of the KV260, thereby realizing a hardware neural network accelerator for landform detection and segmentation.

The next section introduces the detailed approach including network construction and the methods for model deployment. The third section presents experimental results. The fourth section provides an analysis of these results and concluding remarks.

2. Lunar Landform Feature Detection and Segmentation Based on MPSoCs

2.1 Overview of the Approach

Fig. 1 presents the framework of this study. In the dataset preparation phase, we integrate datasets from diverse sources, encompassing both simulation data and actual lunar mission data. These datasets are meticulously annotated and prepared separately for detection and segmentation tasks. Subsequently, we improved and trained lightweight neural networks, ensuring that the network operators align with those supported by the selected MPSoC. Utilizing Xilinx's dedicated toolchain, these models are then subjected to pruning, quantization, compression, and cross-platform compilation. Ultimately, the optimized models are deployed on the chosen MPSoC platform for testing.

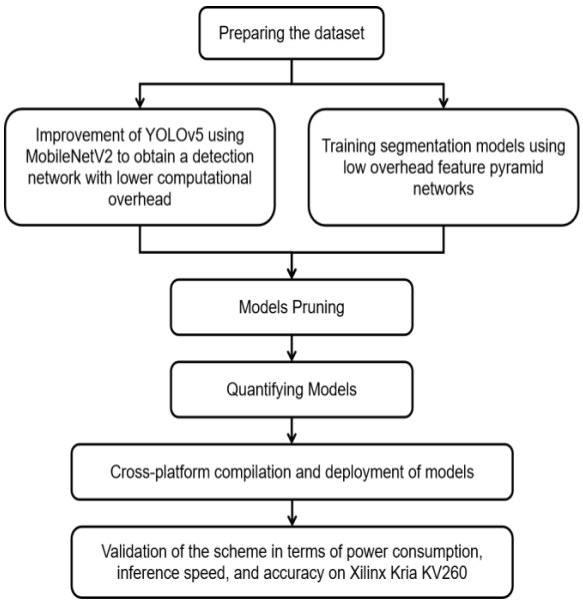

Figure 1. Framework of the approach.

2.2 Network Improvement for Feature Detection

Landform feature detection is a crucial task in planetary surface analysis, aiming to identify and locate features from images or digital terrain models (e.g., Wu et al., 2013; Liu and Wu, 2020). Traditional object detection methods rely on manually designed features and complex classifiers (Wang et al., 2021), whereas the introduction of deep learning techniques has significantly improved the performance and efficiency of object detection. The following techniques are representative of deep-learning based object detection tasks:

- R-CNN: This method first applied Convolutional Neural Networks (CNNs) to object detection by generating candidate regions, then using CNNs to extract feature and classify them (Girshick et al., 2014).
- Fast R-CNN: Accelerated the detection process by sharing convolutional feature maps (Girshick et al., 2015).
- Faster R-CNN: Introduced the Region Proposal Network, enhancing detection speed and accuracy (Ren et al., 2015).
- YOLO: Treated object detection as a regression problem, predicting object locations and classes in a single forward pass (Redmon et al., 2016).
- YOLOv2: Introduced anchor boxes and multi-scale training, improving the detection of small objects (Redmon et al., 2017).
- YOLOv3: Adopted Darknet-53 as the backbone network, further enhancing detection accuracy and speed (Farhadi et al., 2018).
- YOLOv4: Optimized the model performance by introducing Bag of Freebies and Bag of Specials strategies (Bochkovskiy et al., 2020).
- YOLOv5: Improved the anchor frame generation method with a more dynamic approach, resulting in improved detection accuracy (Redmon et al., 2016).

To date, the YOLO series of detection networks remain among the best-performing and most popular in both academia and industry. YOLO models have the following notable advantages in the field of object detection: they can predict the locations and classes of all objects in an image through a single forward pass, significantly improving detection speed and enabling real-time detection. By utilizing the entire image during prediction, YOLO models can better leverage contextual information to predict bounding boxes and classes, enhancing detection accuracy. Additionally, YOLO can detect multiple objects in an image simultaneously, making it suitable for complex scenes. The balance between speed and accuracy makes YOLO one of the top choices for real-time object detection.

In our detection tasks, we chose YOLOv5s as the baseline model for improvement. Firstly, YOLOv5s is a single-stage object detection model that offers high detection accuracy and real-time performance. Compared to other complex multi-stage detection models, YOLOv5s provides accurate landform feature detection results while maintaining fast inference speed. Secondly, the YOLOv5s model structure is relatively lightweight, with fewer parameters and lower computational requirements, making it suitable for deployment on resource-constrained hardware platforms. Thirdly, YOLOv5s has excellent scalability and flexibility, making it easy to improve and optimize. Lastly, and most importantly, except for the SiLU activation function, all operators in YOLOv5s are supported by Vitis-AI. To deploy the original YOLOv5s on the KV260, we could simply modify the activation function to Leaky ReLU.

Although YOLO family is already a kind of relatively lightweight model, its computational overhead is still high for FPGAs. To address the computational demands of edge computing, the creators of the YOLO family of detection algorithms developed the YOLOv3-Tiny model, designed for resource-constrained environments. This model significantly simplifies the original YOLOv3 by reducing the DarkNet-53 backbone to a CNN with just seven layers, thereby greatly decreasing computational overhead and enhancing inference speed. However, the reduction in convolutional layers results in a trade-off with detection effectiveness. To strike a balance between detection performance and computational efficiency, we opted for an alternative approach by replacing the original YOLOv5s backbone with MobileNetV2.

MobileNetV2 is a lightweight convolutional neural network specifically designed for mobile and embedded devices (Sandler et al., 2018). Its core concept is the use of depthwise separable convolutions to reduce the number of parameters and computational complexity. Compared to traditional convolution operations, depthwise separable convolutions decompose standard convolutions into depthwise convolutions and pointwise convolutions, significantly reducing computational complexity. Specifically, depthwise separable convolutions reduce the computational load by approximately 8 to 9 times. In MobileNetV2, downsampling of input feature is accomplished by increasing the stride of the depthwise convolution.

Additionally, the outputs of each network layer undergo batch normalization before activation with the ReLU6 function, which accelerates the model's convergence. This lightweight design makes MobileNetV2 particularly suitable for deployment on resource-constrained hardware platforms.

Table 1 gives the network structure of MobileNetV2 and the operators in each category. Another important reason for choosing MobileNetV2 is that Vitis-AI supports all these operators, allowing them to be deployed on the DPU.

Input	Operator
224×224×3	Conv2d
112×112×32	Bottleneck
112×112×16	Bottleneck
56×56×24	Bottleneck
28×28×32	Bottleneck
14×14×96	Bottleneck
7×7×160	Bottleneck
7×7×320	Conv2d 1×1
7×7×1280	AvgPool 7×7
1×1×1280	Conv2d 1×1

Table 1. The network structure of MobileNetV2

MobileNetV2 introduces inverted residuals and linear bottlenecks, further enhancing the model's expressive power and computational efficiency. Fig. 2 shows the structure of MobileNetV2. MobileNetV1, the predecessor of MobileNetV2, primarily utilizes stacked depthwise separable convolutions (Howard et al., 2017). In the design of MobileNetV2, in addition to continuing the use of depthwise separable convolutions, it introduces Expansion layers and Bottleneck layers. The Bottleneck layer is capable of mapping high-dimensional feature to a lower-dimensional space. The inverted residual structure uses expanding and compressing convolutions within residual blocks to increase the network's non-linear representational capacity while maintaining low computational cost. When designing a network structure, to reduce the amount of operations, it is necessary to design the network dimension as low as possible but if the dimension is low, the activation transform ReLU function may filter out a lot of useful information. The linear bottleneck, by employing a linear activation function at the end of residual blocks, prevents information loss and over-fitting. These design characteristics enable MobileNetV2 to maintain a lightweight structure while providing high feature extraction capabilities.

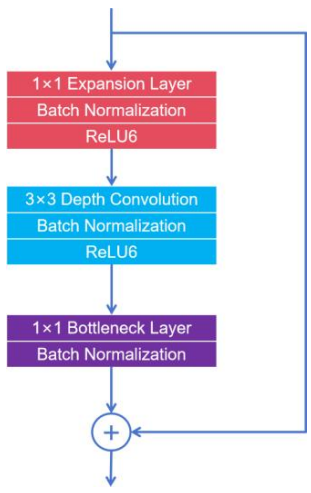

Figure 2. The bottleneck residual structure of MobileNetV2

Additionally, regarding the loss function (Hu et al., 2016), we use the VFL loss function:

$$\mathrm{VFL}(p,q)=\begin{cases}-q(q\log(p)+(1-q)\log(1-p)) & q>0\\ -\alpha p^{\gamma}\log(1-p) & q=0\end{cases}$$

When $q>0$, VFL has no hyperparameters for positive samples and no decay. However, when $q=0$, VFL introduces hyperparameters for negative samples, where γ reduces the contribution of negative samples, and α prevents excessive suppression (Zhang et al., 2021).

2.3 FPN for Feature Segmentation

We pay special attention to the efficiency and performance of models when selecting semantic segmentation models for edge computing platforms. Feature Pyramid Networks (FPN) is particularly well-suited for edge computing platforms, especially highly efficient hardware platforms like the Xilinx Kria KV260 (Lin et al., 2017). Therefore, we chose the classical FPN as our segmentation network.

Firstly, FPN effectively fuses feature at different scales by constructing a feature pyramid. This multi-scale feature fusion allows FPN to capture detailed information while retaining global semantic information when processing high-resolution images. This is particularly important for lunar landform feature segmentation, as the lunar surface has complex topographic feature that require high-precision segmentation results. Secondly, the architecture design of FPN is relatively simple and efficient, with low computational complexity, making it very suitable for running on resource-constrained edge computing platforms. The KV260 platform, with its powerful hardware acceleration capability, can fully leverage the performance advantages of FPN to achieve real-time image segmentation processing. FPN's multi-scale feature fusion, efficient architectural design, and excellent segmentation performance make it an ideal choice for semantic segmentation tasks on edge computing platforms.

The structure of the Feature Pyramid Network (FPN) used for semantic segmentation is illustrated in Figure 3, 256 and 128 denote the number of channels of the feature map, fractions 1/4, 1/8, etc. denote the ratio of the current feature map to the size of the original picture, and C represents the total number of categories.. When employing FPN for semantic segmentation, each feature map undergoes convolution and upsampling operations, enhancing the resolution to one-quarter of the original image. These feature maps are then summed together. Subsequently, a upsampling operation is applied to match the resolution of the original image. The final output is a feature map with the same dimensions as the original image, where the number of channels corresponds to the number of classes.

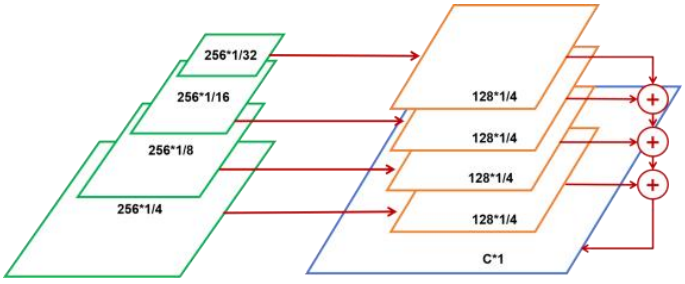

Figure 3. Network architecture of FPN for semantic segmentation

2.4 Model Compression and Deployment

Vitis-AI, developed by Xilinx, is a comprehensive AI inference development platform designed to help developers efficiently deploy AI applications on Xilinx's FPGAs and MPSoCs. Vitis-AI provides a complete set of tools and libraries that support the entire workflow from model optimization to deployment, significantly simplifying the complexity of implementing deep learning inference on hardware. A general framework for building such accelerators is given in Figure 4. Vitis-AI supports leading deep learning frameworks such as TensorFlow, PyTorch and Caffe, enabling developers to port existing models to FPGA-based MPSoCs.

Xilinx's MPSoCs perform deep learning tasks through FPGA-based deep learning processing units (DPUs). Vitis-AI offers a specialized toolchain, including compiler, optimizer, quantizer, and runtime, for deploying trained models on DPUs. These DPUs support only a subset of deep learning operators, so we typically select models from the official model zoo as the basis for improvement and training to avoid deployment issues caused by unsupported operators.

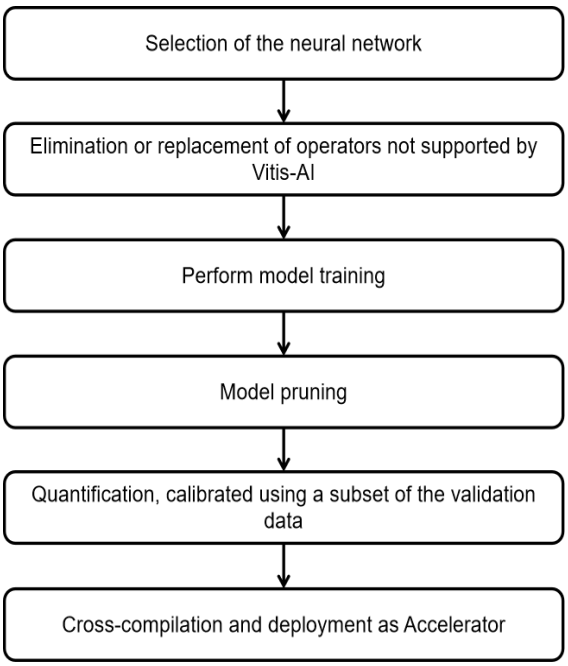

Figure 4. The framework of constructing a neural network accelerator on Xilinx Kria KV260

To ensure speed of inference, we must prune and quantify the model before deployment, by using optimizer and quantizer of Vitis-AI. Pruning involves removing redundant or less significant parameters from the neural network to reduce its size and complexity. This process helps in improving the model's inference speed and energy efficiency. Quantization is the process of converting a model's weights and activations from high-precision floating-point numbers to lower-precision fixed-point numbers. This conversion significantly reduces the model's memory footprint and computational requirements, leading to faster inference and lower power consumption. We quantized the model parameters into 8 bits.

We start by calibrating the model using a representative dataset. This step involves running the model on sample data to collect statistics on the distribution of activations and weights. These statistics are used to determine the optimal scaling factors for quantization. We need a test set of about two hundred images for this step. After pruning and quantization, we use the Vitis-AI compiler to convert the quantized model into a hardware-specific executable. This executable is then deployed on the KV260 MPSoC using the Vitis-AI runtime, enabling efficient inference on the edge device.

3. Experimental Analysis

3.1 Dataset Preparation and Model Training

We utilized lunar surface images from the Chang'e 3 mission, Chang'e 4 mission, Apollo 11 mission, Apollo 14 mission, Apollo 15 mission, and Apollo 17 mission to construct the dataset. The images were co-registered or geo-referenced (Wu et al., 2015) as necessary for dataset construction and validation. Additionally, a portion of the artificial lunar landscape dataset created by Romain Pessia and Genya Ishigami of the Space Robotics Group at Keio University, Japan, was used to supplement the dataset. This resulted in a comprehensive dataset for segmentation and detection training. With the exception of the segmentation task of the artificial lunar landscape dataset, most of the data were labelled by the research team itself.

For the detection tasks, the dataset includes three categories: large rocks, small rocks, and impact craters. For the segmentation tasks, the dataset covers three categories: sky, large rocks, and small rocks. The constructed dataset encompasses multiple real lunar missions and various imaging conditions. We divided the entire dataset into 80% for training, 10% for validation, and 10% for testing.

3.2 Experimental Environment

The project utilized Vitis AI version 3.5. Network model training was conducted using an NVIDIA 4090 GPU, while the Heterogeneous MPSoC used was the Xilinx Kria KV260. The deep learning framework employed was PyTorch 2.2.1. The images in the test set have a resolution of 640×480 pixels and are 8-bit color images with three channels.

3.3 Evaluation of Landform Feature Detection

To evaluate the performance of YOLOv5s-MobileNetV2 on the MPSoC, we deployed YOLOv5s and YOLOv3-Tiny networks as benchmarks. We conducted evaluations in terms of accuracy, inference speed, and power consumption, with the results presented in Table 2.

Models	Precision (%)	Recall (%)	FPS	Power consumption (W)
YOLOv3-Tiny	78.37	61.54	57	6.1
YOLOv5s	84.43	69.63	31	8.4
YOLOv5s-MobileNetV2	84.12	69.47	**52**	**6.3**

Table 2. Evaluation results of the detection task

The experimental results confirmed that YOLOv5s-MobileNetV2 achieved accuracy very close to that of YOLOv5s while maintaining power consumption and frame rates similar to YOLOv3-Tiny. This demonstrates that YOLOv5s-MobileNetV2 can effectively reduce power consumption and improve inference speed while ensuring accuracy in relevant tasks. The actual detection performance of YOLOv5s-

MobileNetV2 is illustrated in Figure 5. The experimental results indicate that the selected network performs well in detection tasks, exhibiting strong detection capabilities for dense targets, small targets, and multi-scale targets. It can fully meet the needs of lunar robots in obstacle avoidance and autonomous driving.

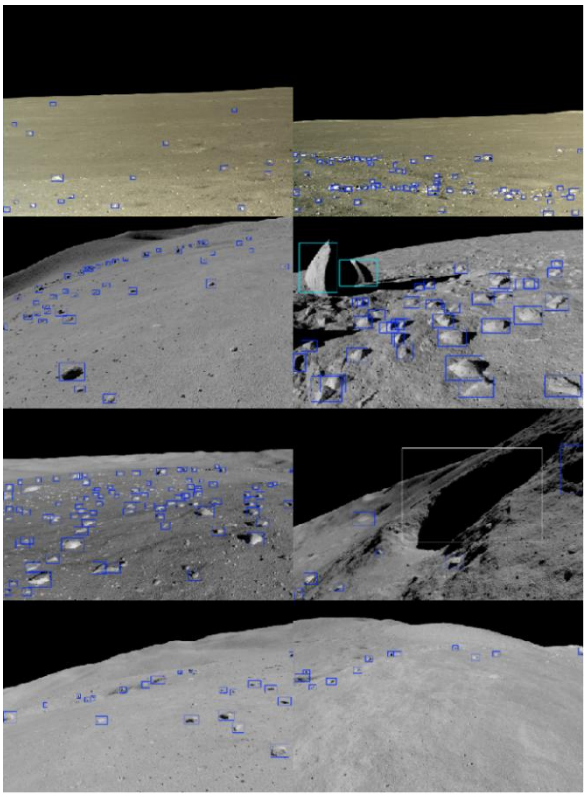

Figure 5. Examples of landform feature detection results, blue for small rocks, green for large rocks, and white for craters

3.4 Evaluation of Landform Feature Segmentation

We conducted evaluations in terms of accuracy, inference speed, and power consumption, with the results presented in Table 3.

Class	IoU (%)	FPS	Power consumption (W)
Large rocks	71.34%	**27**	**7.6**
Small rocks	64.15%		

Table 3. Evaluation results of the segmentation task

The validation results of the semantic segmentation task on the KV260 are shown in Figure 6.

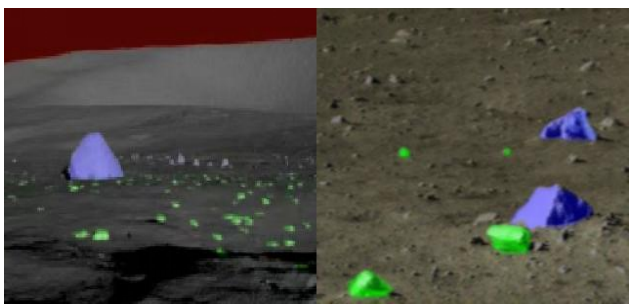

Figure 6. Examples of landform feature segmentation results, red for the sky, blue for the big rocks, green for the small rocks

3.5 Simulation Validation of the Scenario

To visually verify the inference speed of the landform feature detection task and its capability to support autonomous lunar rover navigation, we conducted a simulation using a small vehicle equipped with the KV260 and connected to a CMOS camera. The simulation was performed in a laboratory setting with a constructed planetary surface. The vehicle used was the SCOUT MINI from AgileX Robotics, and the CMOS camera was the Sony IMX-291. The verification environment is depicted in Figure 7. Additionally, parallel light sources were arranged to simulate the varying lighting conditions on the Moon.

Figure 7. Rover for dynamic conditional validation

The test results of the detection task in the laboratory simulation environment are shown in Figure 8, with images from different lighting conditions displayed on the left and right. The experiment demonstrates that the detection model can identify all major obstacles in the laboratory simulation environment under various lighting conditions, effectively simulating the Moon's environment (Duan et al., 2024). During the experiment, data was transmitted and streamed back to the host computer via RTSP using a USB-WiFi adapter module. The operator then navigated through the images received by the host computer. The model's high inference speed was evaluated to better support the autonomous movement of the rover on the lunar surface, particularly in avoiding larger obstacles such as rocks at higher speeds.

Figure 8. Effectiveness of detection tasks in a laboratory simulation environment

4. Conclusions and Discussion

In this paper, we propose a practical approach to real-time detection and segmentation of lunar landform features using a heterogeneous MPSoC platform. By leveraging the energy-efficient and customizable characteristics of FPGA-based heterogeneous MPSoCs, we have developed a neural network inference accelerator that balances energy efficiency, inference speed, and accuracy. Our experimental results demonstrate the effectiveness of this approach, achieving a detection frame rate of 52 FPS and a segmentation frame rate of 27 FPS with an input size of 640x480, while maintaining a low power consumption of approximately 6.3 W for detection task and 7.6 W for segmentation. These results indicate that our design can

support autonomous lunar rover missions by balancing computational power and energy efficiency.

The deployment of a neural network accelerator on the Xilinx KV260 MPSoC for such missions underscores the potential of FPGA-based heterogeneous MPSoCs for intelligent landform feature sensing in lunar exploration. However, several challenges and future research directions remain:

Application of Multitasking Networks: Future research could explore the use of multitasking networks instead of two separate networks. Multi-task learning can share feature extraction layers, thereby reducing computational overhead, improving resource utilization, and potentially enhancing inference speed and energy efficiency. Additionally, multitasking networks can improve overall model robustness and generalization by jointly optimizing multiple tasks.

Integration with Robot Control: Combining heterogeneous processor accelerators with robot control modules on the same MPSoC could maximize the advantages of reconfigurability. This integration would streamline the deployment process and enhance the overall efficiency of the system.

In summary, our work demonstrates the feasibility of using FPGA-based heterogeneous MPSoCs for the real-time detection and segmentation of lunar landform feature. It is demonstrated that such accelerators meet the requirements of the task in terms of energy consumption, inference speed, and accuracy.

Acknowledgement

This work was supported by grants from the Research Grants Council of Hong Kong (Project No: PolyU 15236524, Project No: PolyU 15215822, RIF Project No: R5043-19). The authors thank all individuals who worked on the dataset to make them publicly available.

References

Bochkovskiy, A., Wang, C. Y., Liao, H. Y. M., 2020. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934. https://doi.org/10.48550/arXiv.2004.10934.

De Rosa, D., Bussey, B., Cahill, J., 2012. Characterization of potential landing sites for ESA’s Lunar Lander project. Planetary and Space Science, 74(1), 224-246.

Duan, R., Wu, B., Chen, L., Zhou, H., Fan, Q., 2024. AI-Driven Dim-Light Adaptive Camera (DimCam) for Lunar Robots. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 48, (pp. 141-146).

Farhadi, A., Redmon, J., 2018. Yolov3: An incremental improvement. In Proceedings of the IEEE conference on computer vision and pattern recognition, (Vol. 1804, pp. 1-6).

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, (pp. 580-587).

Girshick, R., 2015. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, (pp. 1440-1448).

Girshick, R., 2015. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, (pp. 580-587).

Howard, A. G., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.0486. https://doi.org/10.48550/arXiv.1704.04861.

Hu, H., Ding, Y., Zhu, Q., Wu, B., Xie, L., Chen, M., 2016. Stable least-squares matching for oblique images using bound constrained optimization and a robust loss function. ISPRS Journal of Photogrammetry and Remote Sensing, 118, 53-67.

Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, (pp. 2117-2125).

Liu, W. C., Wu, B., 2020. An integrated photogrammetric and photoclinometric approach for illumination-invariant pixel-resolution 3D mapping of the lunar surface. ISPRS journal of photogrammetry and remote sensing, 159, 153-168.

Mohaidat, T., Khalil, K., 2024. A survey on neural network hardware accelerators. IEEE Transactions on Artificial Intelligence, 8(5), (pp. 3801-3822).

Pessia, R., Ishigami, G., 2019. Artificial lunar landscape dataset. Kaggle, v6. https://www.kaggle.com/datasets/romainpessia/artificial-lunar-rocky-landscape-dataset.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, (pp. 779-788).

Redmon, J., Farhadi, A., 2017. YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition, (pp. 7263-7271).

Ren, S., He, K., Girshick, R., Sun, J., 2016. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence, 39(6), 1137-1149.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L. C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, (pp. 4510-4520).

Wang, C., Jia, Y., Xue, C., Lin, Y., Liu, J., Fu, X., Zou, Y., 2024. Scientific objectives and payload configuration of the Chang'E-7 mission.National Science Review, 11(2), nwad329.

Wang, Y., Wu, B., Xue, H., Li, X., Ma, J., 2021. An improved global catalog of lunar impact craters (≥ 1 km) with 3D morphometric information and updates on global crater analysis. Journal of Geophysical Research: Planets 126 (9), e2020JE006728.

Wu, B., Guo, J., Hu, H., Li, Z., Chen, Y., 2013. Co-registration of lunar topographic models derived from Chang’E-1, SELENE, and LRO laser altimeter data based on a novel surface matching method. Earth and Planetary Science Letters, 364, 68-84.

Wu, B., Li, F., Ye, L., Qiao, S., Huang, J., Wu, X., Zhang, H., 2014. Topographic Modeling and Analysis of the Landing Site

of Chang’E-3 on the Moon. Earth and Planetary Science Letters, 405, (pp. 257-273).

Wu, B., Tang, S., Zhu, Q., Tong, K., Hu, H., Li, G., 2015. Geometric integration of high-resolution satellite imagery and airborne LiDAR data for improved geopositioning accuracy in metropolitan areas. ISPRS Journal of Photogrammetry and Remote Sensing, 109, 139-151.

Wu, B., Li, F., Hu, H., Zhao, Y., Wang, Y., Xiao, P., Li, Y., Liu, W. C., Chen, L., Ge, X., Yang, M., Xu, Y., Ye, Q., Wu, X., Zhang, H., 2020. Topographic and Geomorphological Mapping and Analysis of the Chang’E-4 Landing Site on the Far Side of the Moon. PE&RS, 86(4), 247-258.

Wu, B., Li, Y., Liu, W. C., Wang, Y., Li, F., Zhao, Y., Zhang, H., 2021. Centimeter-resolution topographic modeling and fine-scale analysis of craters and rocks at the Chang'E-4 landing site. Earth and Planetary Science Letters, 553, 116666.

Zhang, H., Wang, Y., Dayoub, F., Sunderhauf, N., 2021. Varifocalnet: An iou-aware dense object detector. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, (pp. 8514-8523).