An FPGA-SoC Camera Payload Processing Unit with Soft-GPU Acceleration for Onboard Vision

Mohamed Nadhem Mraihi[1], Ulas Sezgin[1], Chedi Fassi[1], Sebastian Stang[1], Jintin Frank[1], Markus Plattner[2]

[1]*Engineering Minds Munich GmbH, Gollierstr. 70, 80339 Munich*
[2]*University of Applied Sciences Munich, Lothstraße 34, 80335 Munich*

This paper presents an FPGA-SoC-based camera payload processing unit for onboard vision on resource-constrained spacecraft platforms. The system combines Engineering Minds Munich's Smart Power and Processing Module (SPPM), a remote camera interface based on MIPI-CSI-2, and a GPU-accelerated embedded processing chain using a soft-GPU concept. The architecture is intended for image-based monitoring and recognition tasks directly on the payload electronics, enabling local interpretation of visual data instead of relying exclusively on transmission of full-frame imagery. A lightweight semantic-segmentation network identifies relevant scene structures, while deterministic post-processing kernels convert the dense output into compact geometric descriptors and other image-derived outputs for subsequent interpretation. In the current engineering implementation, the processing chain is distributed between the FPGA-based camera payload platform and an external GPU@SAT accelerator used as a prototype soft-GPU environment. Experimental validation in an EGSE environment demonstrates the feasibility of the end-to-end processing chain and highlights current challenges related to illumination variability, background complexity, memory-aware implementation, and execution time. The presented approach shows how soft-GPU acceleration can extend FPGA-based payload electronics toward onboard vision with compact, decision-relevant outputs.

1 Introduction

Onboard image processing is becoming increasingly relevant for modern spacecraft, especially for small and medium-class platforms operating under limited downlink bandwidth, delayed ground interaction, and tight power budgets. Instead of transmitting full-frame imagery for later interpretation on ground, future payload electronics are expected to generate higher-level outputs directly onboard, such as classifications, contours, compact descriptors, or event flags. This trend is reflected in recent work on onboard artificial intelligence and image-selection strategies, including demonstrations such as ESA's OPS-SAT SmartCam, where onboard machine learning was used to support image selection and downlink prioritization [1–3].

At the same time, FPGA-based payload electronics already provide substantial parallel processing capability and are widely used in space systems for interfacing, control, and deterministic signal processing. Extending such platforms with a soft-GPU concept is therefore attractive: FPGA resources are already present in the payload electronics, and a GPU-like execution model can add image-processing and AI functionality without introducing a fundamentally new hardware class. This is particularly relevant for dense pixel-wise workloads such as semantic segmentation and image-local post-processing kernels [4–8].

This paper presents an FPGA-SoC-based camera payload processing unit with soft-GPU acceleration for onboard vision. The architecture is built around Engineering Minds Munich's Smart Power and Processing Module (SPPM), a remote camera front end connected via MIPI-CSI-2, and a GPU-accelerated embedded processing chain. A lightweight convolutional neural network first performs semantic segmentation of the camera image. Deterministic post-processing kernels then convert the dense output into compact geometric descriptors suitable for further interpretation. The paper focuses on the platform concept, the embedded implementation, and the use of soft-GPU acceleration for onboard image understanding. Solar-panel deployment monitoring is used as the representative application example for this paper.

2 Application Context

The proposed processing concept is intended for onboard image-recognition tasks in which local interpretation of camera data is more useful than transmitting raw imagery alone. In this paper, the selected example is visual monitoring of solar-panel deployment, where image-based assessment can provide direct geometric evidence of the observed configuration and support the generation of compact status information.

More generally, the same architecture is applicable to a broader set of payload-side vision tasks. Examples include monitoring of deployable or movable spacecraft structures, verification of antenna or panel orientation, observation of mechanism states, and Earth-observation applications in which compact image-derived outputs such as masks,

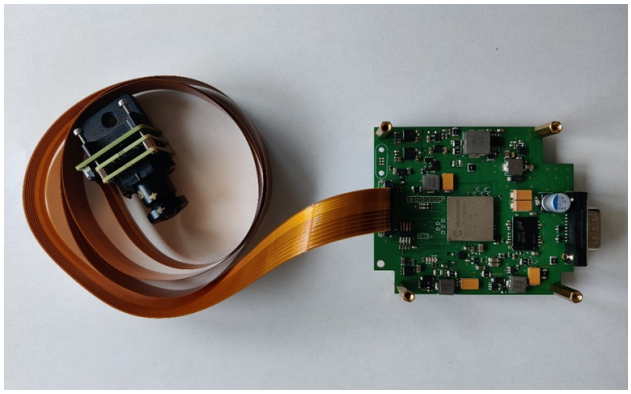

Figure 1: The CPPU (Camera payload and processing unit) includes SPPM with camera extension

regions of interest, or event flags are preferable to transmitting the full frame.

The overall concept is particularly suited to compact spacecraft platforms where power, mass, and communication resources are constrained, and where FPGA-based payload electronics are already present for control and interface functions. Under these conditions, soft-GPU acceleration offers a path to add onboard image-understanding capability with limited additional hardware complexity.

3 CPPU Concept and Hardware Architecture

The CPPU, as illustrated in Figure 1, is based on the SPPM, a modular hardware stack composed of the Smart Processing Module (SPM) and the Smart Power Supply (SPS)[6, 7]. At its core, the SPM integrates a Microchip PolarFire SoC FPGA with four RISC-V application cores and FPGA logic sharing 2 GB of LPDDR4 memory. The platform is designed for low-power, radiation-tolerant, high-data-rate processing and is therefore well suited to embedded onboard vision tasks. The hardware concept supports mission-specific application boards and combines flexible interfacing with robust onboard compute resources.

Based on this platform, the CPPU provides power and control to a remote camera via MIPI-CSI-2.0. The image sensor can be positioned up to approximately 1 m away from the main electronics by means of a flex-PCB extension, which is particularly beneficial in constrained spacecraft layouts and for self-observation geometries. The unit operates from a 28–34 V input provided by the spacecraft Electrical Power System and generates the lower voltages required for the camera. In addition, it can provide constant-current outputs up to 2 A for release-mechanism related functions. Communication with the onboard computer is implemented through UART/RS485, with telecommands and telemetry exchanged through a memory-mapped register interface.

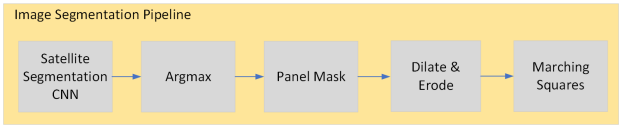

Figure 2: Image Segmentation Pipeline

The selected example in this paper is the monitoring of deployable solar-array geometry. Accordingly, the present implementation emphasizes contour extraction and geometric assessment of elongated appendage-like structures rather than generic scene understanding. In the current engineering setup, the processing chain is distributed over two hardware boards: the SPPM and a GPU@SAT devKit by IngeniArs S.r.l. The SPPM handles image capture, storage, communication, and control, while the GPU@SAT board serves as an external accelerator for image segmentation and subsequent processing [4]. Within the SPPM, functions are partitioned over the available compute resources: one processing core handles communication and platform control, another handles interaction with the accelerator and camera triggering, and a third organizes data transfer between the camera, FPGA-connected memory, and system RAM. This partitioning supports modularity and separates image acquisition, communication, and execution control of the processing chain.

3.1 Segmentation-to-contour pipeline

The core functionality of the CPPU is implemented as a segmentation-to-descriptor pipeline, see Figure 2, that converts full-frame camera imagery into compact geometric evidence for deployment assessment. The first stage uses a lightweight convolutional neural network (CNN), selected to preserve full spatial resolution between input and output while remaining compatible with the GPU@SAT framework. The network is intentionally built only from operators that are directly supported by the target execution environment. This constraint enables efficient quantization and straightforward deployment to the embedded accelerator.

Training is based on RGB images with pixel-wise labels corresponding to four semantic classes: background, solar panels, spacecraft body, and antennas. This decomposition allows the network to learn the geometry of the spacecraft scene and to isolate the solar-panel region for later analysis. Instead of directly predicting a binary deployment decision, the pipeline first reconstructs a structured semantic representation of the observed scene. This is advantageous because it separates perception from final decision logic and allows subsequent deterministic processing to operate on physically meaningful regions.

After segmentation, the output passes through a sequence of deterministic processing kernels:

1. Argmax
2. Panel mask extraction
3. Morphological dilation & erosion
4. Marching Squares contour extraction

The argmax stage converts multi-class network output into a discrete class map. The panel-mask stage isolates the solar-panel class into a binary region of interest. A dilation step fills small holes and reconnects fragmented regions caused by segmentation noise, while a subsequent erosion step suppresses isolated artifacts and refines the object shape. The combination of dilation followed by erosion forms a morphological closing operation that stabilizes the panel mask before geometric interpretation. Finally, Marching Squares converts the refined binary mask into a compact vectorized contour representation (see Figure 3). Rather than preserving every image pixel, this stage produces coordinate-based geometric information that can be rendered, tracked, or further analyzed with minimal data volume.
Deployment-state assessment is based on a minimum-area bounding rectangle computed from the extracted contour. Let C denote the contour points of the detected panel region. The enclosing rectangle is defined as

$$R^\star(C) = \arg\min_{R} \text{area}(R) \quad \text{s.t. } C \subseteq R. \tag{1}$$

In practice, $R^\star$ is obtained using a convex-hull-based edge sweep, in which each hull edge defines a candidate rectangle orientation and the enclosing rectangle with minimum area is retained. From this rectangle, the classifier derives the enclosed area

$$A = \text{area}(R^\star) \tag{2}$$

and the aspect ratio

$$\rho = \frac{\max(w, h)}{\min(w, h)}, \tag{3}$$

where w and h are the side lengths of $R^\star$. The deployment decision is then given by

$$\hat{y} = \begin{cases} 1, & A \geq A_{\text{th}} \ \wedge \ \rho \geq \rho_{\text{th}} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

with default thresholds $A_{\text{th}} = 500$ and $\rho_{\text{th}} = 1.9$.

To improve robustness under realistic acquisition conditions, the pipeline also includes a lightweight image-enhancement stage. This preprocessing applies gray-world white balance, manual channel gain adjustment, gamma correction, and unsharp masking. The purpose is to reduce color cast, improve mid-tone visibility, and enhance structural edges prior to inference. Since deployed camera imagery differs from the cleaner visual conditions typically present in training data, this deterministic enhancement stage reduces part of the domain gap without introducing excessive computational complexity.

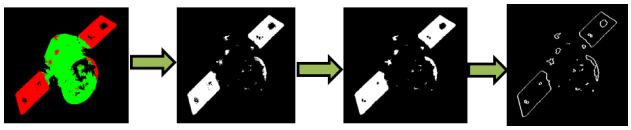

Figure 3: CNN output processing steps until Solar Panel contour extraction

4 Embedded GPU-Accelerated Implementation

The processing chain is implemented using the GPU@SAT framework by IngeniArs S.r.l., which provides a soft-GPU architecture on FPGA fabric together with software interfaces for kernel management and execution. The CNN model is quantized into fixed-point int32 parameters compatible with the accelerator arithmetic model. During deployment, each neural-network layer is mapped to a corresponding GPU@SAT kernel, and the overall execution flow is described through a configuration of memory regions, instruction blocks, and a scheduler defining the order of execution. This provides a structured path from a trained TensorFlow model to an executable embedded inference pipeline.
The custom post-processing steps were integrated into the same execution environment as dedicated compute kernels. These kernels were compiled within the framework toolchain and registered alongside the CNN operators so that mask refinement and contour extraction could follow immediately after segmentation. A key embedded-systems challenge during integration was the available 128 MB device memory limit. A straightforward implementation with separate intermediate buffers for every stage exceeded the budget. This was resolved by reusing shared buffers across sequential stages, allowing the same storage to hold argmax results, filtered masks, and later morphological outputs as long as each consumer stage completed before the buffer was repurposed. This buffer reuse is an important design aspect because it shows that the feasibility of onboard AI on small platforms depends as much on memory-aware system design as on the neural model itself.
The host-side software remains lightweight, handling data transfer, execution control, and result retrieval. Input images are converted into a flattened quantized tensor format, uploaded together with the model parameters, and executed according to the configured scheduler. After execution, the contour outputs are converted into user-oriented geometric products, such as SVG-like vector segments, and forwarded to the classification stage. In the current prototype, the GPU@SAT framework is hosted on a separate accelerator devkit rather than on the CPPU processing platform itself. Execution is accessed through the GPU@SAT software stack via a server-based interface, such that the host-side software connects remotely to the accelerator, uploads model and image data, triggers kernel execution,

and retrieves the generated outputs. This setup reflects the present validation architecture and should be distinguished from the longer-term objective of tighter integration of the soft-GPU concept into the PolarFire-based processing platform. However, the architectural intent is to enable later migration of the soft-GPU concept into the PolarFire-based processing platform itself, which would improve compactness and open the path to tighter integration and performance tuning.

5 Test Setup

5.1 Test objectives

The test campaign is designed to verify the proposed CPPU concept at three levels. First, it validates the end-to-end functional integration of the front end, including camera triggering, image acquisition, storage, RS485 communication, and transfer of image data to the embedded acceleration chain. Second, it evaluates the image-processing application under representative mission conditions by testing correct, partially deployed, and damaged solar-array configurations under controlled illumination and varying image quality. Third, it assesses the practical feasibility of descriptor-based onboard monitoring by measuring segmentation robustness, contour quality, classification consistency, workflow automation, and execution time.

To accomplish these objectives, an Electrical Ground Support Equipment (EGSE) environment was developed, which will be explained in more detail in the next subsection.

5.2 EGSE validation workflow

The proposed concept was validated in an EGSE environment designed to provide a realistic development and test setup for the camera front end and the SPPM. As illustrated in Figure 4, the setup integrates the camera, processing hardware, adjustable lighting, precision power supply, software control, and mechanical support into a single portable testbed. In addition to conventional hardware checkout, the EGSE also supports a basic hardware-in-the-loop-style digital picture feeder, enabling controlled injection of image data for accelerated machine-learning development and repeatable validation.

The software environment supports platform control, register access, image acquisition, transfer, display, housekeeping retrieval, and communication logging. It also includes robustness features for high-speed data transfer, such as check and resend routines on a full-duplex 3 Mbps RS485/RS422 interface and the handling of large raw files of approximately 16 MB. The testbench can reproduce relevant deployment-monitoring scenes under controlled illumination and can also exercise damaged and partially deployed panel cases. This makes it suitable not only for algorithm testing but also for integrated front-end verification.

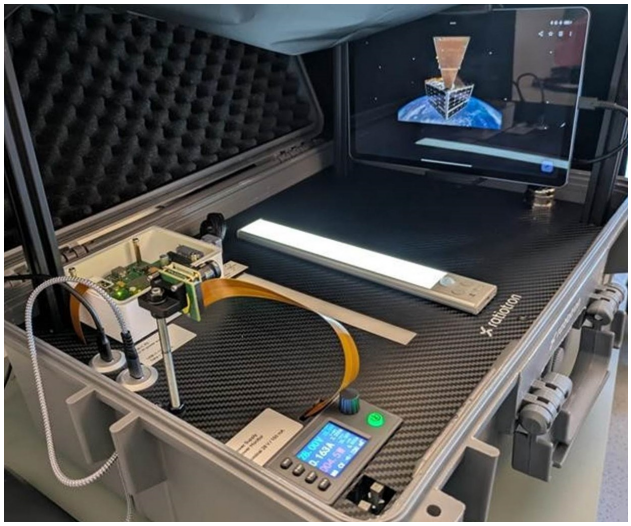

Figure 4: EGSE provided by ratiotron for CPPU integration and validation.

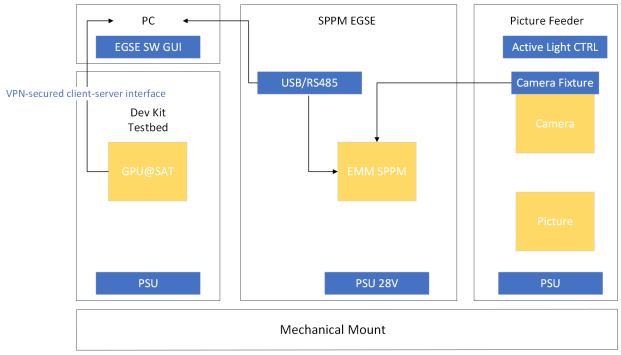

Figure 5: Functional block diagram of the EGSE and validation workflow.

The validation flow is automated end to end (see Figure 5). Images captured through the EGSE are stored in a shared directory, converted into the input representation required by the embedded accelerator, and then processed automatically by scripts that trigger the GPU execution chain. Once contours and segmentation outputs become available, additional scripts invoke the geometric classifier. This automated workflow demonstrates that the CPPU concept can be executed as a repeatable chain from image capture to deployment-state inference rather than only as an offline post-processing exercise.

6 Results

Evaluation was performed on two main validation sets: images captured through the test setup and directly fed digital images. For the camera-based test-setup images, the dominant observation is that reliable predictions de-

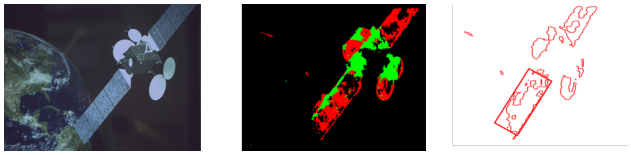

Figure 6: From left to right: (a) Enhanced test image, (b) segmentation result, and (c) minimum-area bounding rectangle of the detected contourn

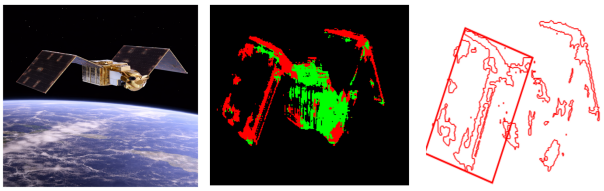

Figure 7: From left to right: (a) Directly fed Digital image, (b) segmentation result, and (c) minimum-area bounding rectangle of the detected contourn

pend strongly on careful preprocessing. Reduced image quality, lighting variation, and color tint significantly affect segmentation consistency. Without enhancement, the segmentation output becomes more fragmented and noisier, which in turn destabilizes contour extraction and the bounding-rectangle-based classification. With suitable enhancement, the panel region becomes more distinguishable, the contours become more coherent, and the resulting bounding rectangle better reflects the dominant shape and orientation of the target, as illustrated in Figure 6.

For directly fed digital images, like the one shown in Figure 7, the dominant limitation is different. Here, the network may still misclassify parts of the spacecraft under complex backgrounds and local illumination variations, producing spurious segmentation regions and unreliable contour extraction. Under these conditions, the reported performance, presented in Table 1, is an accuracy of 0.567, a precision of 0.833, a recall of 0.476, and an F1-score of 0.606, where the F1-score denotes the harmonic mean of precision and recall. These values indicate that the system can often produce correct positive detections when the extracted geometry is clean, but still misses a significant fraction of relevant positive cases and remains sensitive to scene complexity.

In addition to the deployment-state estimate, the contour-based representation significantly reduces the amount of information that must be transferred compared with the original image. In the evaluated setup, the SVG-like contour product reduces the downlinked data volume by approximately 70% on average relative to the original image, while still preserving the geometric information required for deployment assessment. This result directly supports the central objective of replacing image-as-telemetry by compact, decision-relevant outputs.

From a runtime perspective, all inputs are resized to 256×256, resulting in nearly constant compute and memory requirements across the evaluated images. The average kernel execution time reported for the current implementation is approximately 153.55 s per image, as mentioned in Table 2. In the present distributed prototype, this runtime should be interpreted together with the two-board architecture and the remote server-based execution model of the accelerator. The result therefore represents a proof of functional feasibility of the end-to-end processing chain on FPGA-based acceleration rather than an optimized flight-ready latency.

Table 1: Classification performance on directly fed digital images.

Metric	Value
Accuracy	0.567
Precision	0.833
Recall	0.476
F1-score	0.606

Table 2: Average kernel execution time for 256×256 inputs.

Parameter	Value
Input size	256×256
Average kernel execution time	153.55 s/image

Overall, the validation shows that the segmentation-to-descriptor concept is viable and that deployment-state classification can be derived from compact geometric outputs instead of full-frame telemetry. At the same time, the current results identify the main bottlenecks clearly: sensitivity to realistic illumination conditions, background-induced false detections, reflection and saturation effects, and the current execution time of the two-board implementation. Future optimization should therefore address both algorithm robustness and tighter hardware integration.

7 Discussion

The presented setup demonstrates that a camera-capable FPGA-SoC payload platform can be extended toward on-board vision by combining a remote image sensor, a semantic-segmentation pipeline, deterministic contour extraction, and a soft-GPU-based acceleration concept. In the current implementation, the distributed prototype has already shown the feasibility of running the complete chain from image acquisition to compact geometric output, while also exposing the main limitations observed during validation, namely sensitivity to illumination conditions, background complexity, and the execution time of the present two-board setup.

From a system perspective, an important aspect is that the FPGA resource is already part of the payload electronics for control, interfacing, and deterministic data handling. Soft-GPU acceleration therefore offers a path to add image-processing and AI capability on top of an existing payload-computing platform rather than introducing a

completely separate processing architecture. The demonstrated segmentation-to-descriptor chain also points to a broader class of onboard vision tasks beyond the scenario used for validation in this paper, such as monitoring of mechanisms, verification of appendage orientation, compact generation of image-derived products, and future extensions toward additional parallelizable inference workloads.

The validation results also indicate where future work is most beneficial. Improved robustness is expected from training or fine-tuning with more representative camera data, while tighter hardware integration would reduce system overhead and better exploit the compact and low-power characteristics of the FPGA-SoC platform. The current prototype therefore serves both as a functional demonstrator of the processing concept and as a basis for broader onboard vision capabilities on payload electronics.

8 Conclusion

This paper presented an FPGA-SoC-based camera payload processing unit with soft-GPU acceleration for onboard vision. The proposed architecture combines a remote camera interface, an SPM/SPPM-based payload electronics platform, a lightweight semantic-segmentation network, and deterministic post-processing to generate compact image-derived outputs under embedded constraints. The main contribution lies in demonstrating how a soft-GPU concept can extend FPGA-based payload electronics toward onboard image recognition and compact image-derived outputs.

Experimental validation in an EGSE environment showed the feasibility of the complete processing chain from image acquisition to contour-based output. The selected example in this paper was solar-panel deployment monitoring, but the underlying processing concept is applicable more broadly to onboard vision tasks in which compact, decision-relevant outputs are preferable to raw-image transfer. The results also identified the main present limitations of the prototype, including sensitivity to illumination conditions, background complexity, and the execution time of the current distributed implementation.

Future work will focus on tighter integration of the soft-GPU concept into the PolarFire-based processing platform, improved robustness through more representative training and validation data, and extension of the processing chain toward additional onboard monitoring and image-recognition tasks.

9 Acknowledgments

The authors acknowledge the collaborative framework with Markus Roner from ratiotron and IngeniArs S.r.l. for the GPU@SAT technology integration and contributions to the image segmentation development efforts. This development was carried out as contribution for ESA Contract No. 4000144652/24/I-DT - EO MAKERSPACE : AI ON EDGE APPLIED TO EO SMART SENSORS - INVEST INCUBED 2.

References

1. Labrèche, G. *et al. Artificial Intelligence for Autonomous Planning and Scheduling of Image Acquisition with the SmartCam App On-Board the OPS-SAT Spacecraft* in *AIAA SCITECH 2022 Forum* (2022), AIAA 2022–2508.
2. Chintalapati, B. *et al.* Opportunities and Challenges of On-Board AI-Based Image Recognition for Small Satellite Earth Observation Missions. *Advances in Space Research* **75,** 6734–6751 (2025).
3. Gardill, M. *et al. Towards Space Edge Computing and Onboard AI for Real-Time Teleoperations* in *2023 IEEE 22nd International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)* (2023).
4. Benelli, G. *et al.* GPU@SAT DevKit: Empowering Edge Computing Development Onboard Satellites in the Space-IoT Era. *Electronics* **13,** 3928 (2024).
5. Guo, Z. *et al.* An Overlay Accelerator of DeepLab CNN for Spacecraft Image Segmentation on FPGA. *Remote Sensing* **16,** 894 (2024).
6. Plattner, M. & Fassi, C. Payload Computer based on PolarFire SoC. *Proceedings of the 2023 European Data Handling and Data Processing Conference for Space, EDHPC 2023* (2023).
7. Plattner, M., Fassi, C. & Berner, A. *Smart Power Supply for FPGA/SoC* in *2023 13th European Space Power Conference (ESPC)* (2023), 1–4.
8. Microchip Technology Inc. *PolarFire SoC Product Overview* 2025.