



UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II



DIPARTIMENTO 2018
DI ECCELLENZA 2022



DIETI
DIPARTIMENTO
DI ECCELLENZA
2023 - 2027

Università degli Studi di Napoli Federico II
Ph.D. Program in
Information Technology and Electrical Engineering
XXXVIII Cycle

THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Edge AI for Autonomous Spacecraft: Enabling Onboard Intelligence with FPGA Acceleration

by

GIOVANNI MARIA CAPUANO

Advisor: Prof. Antonio Giuseppe Maria Strollo

Co-advisor: Prof. Nicola Petra

Co-advisor: Mariano Severi, PhD



SCUOLA POLITECNICA E DELLE SCIENZE DI BASE

DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE

Dedico questo lavoro alla mia famiglia.

*A mio padre Giuseppe, la cui passione per lo spazio
e per l'elettronica ha alimentato in me la curiosità e
l'amore per la ricerca.*

*A mio fratello Vincenzo, guida ed esempio costante
di determinazione e perseveranza.*

*A mio fratello Francesco, per il suo sostegno sincero e
per aver creduto in me con affetto e ammirazione.*

*A mia sorella Benedetta e a mia madre Aida, per
essere la mia casa, la mia sicurezza e la mia forza.*

A tutti voi, con profonda gratitudine e amore.

EDGE AI FOR AUTONOMOUS SPACECRAFT: ENABLING ONBOARD INTELLIGENCE WITH FPGA ACCELERATION

Ph.D. Thesis presented
for the fulfillment of the Degree of Doctor of Philosophy
in Information Technology and Electrical Engineering
by

GIOVANNI MARIA CAPUANO

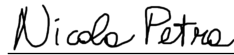
December 2025



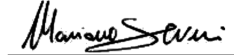
Approved as to style and content by



Prof. Antonio Giuseppe Maria Strollo, Advisor



Prof. Nicola Petra, Co-advisor



Mariano Severi, PhD, Co-advisor

Università degli Studi di Napoli Federico II

Ph.D. Program in Information Technology and Electrical Engineering

XXXVIII cycle - Chairman: Prof. Stefano Russo



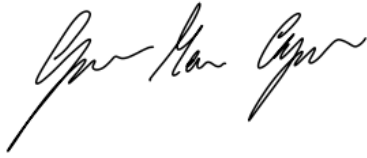
<http://itee.dieti.unina.it>

Candidate's declaration

I hereby declare that this thesis submitted to obtain the academic degree of Philosophiæ Doctor (Ph.D.) in Information Technology and Electrical Engineering is my own unaided work, that I have not used other than the sources indicated, and that all direct and indirect sources are acknowledged as references.

Parts of this dissertation have been published in international journals and/or conference articles (see list of the author's publications at the end of the thesis).

Napoli, December 10, 2025

A handwritten signature in black ink, appearing to read 'Giovanni Maria Capuano', written in a cursive style. The signature is positioned above a horizontal line.

Giovanni Maria Capuano



**Finanziato
dall'Unione europea**
NextGenerationEU



La borsa di dottorato è stata cofinanziata con risorse del Piano Nazionale di Ripresa e Resilienza, D.M. 352/2022 Missione 4 – Componente 2 – Investimento 3.3 “Dottorati innovativi” CUP: E66G22000400009

Abstract

Artificial intelligence (AI)-driven onboard data processing is a cornerstone of future space missions, enabling autonomous and time-critical decision-making in applications such as planetary exploration, in-orbit servicing, and Earth Observation (EO). Executing AI algorithms directly onboard mitigates the latency and bandwidth constraints inherent to ground-based processing. However, most commercial off-the-shelf edge AI accelerators lack the radiation tolerance required for long-duration and deep-space missions. Space-qualified Field-Programmable Gate Arrays (FPGAs) provide a robust alternative, combining fault tolerance, low power consumption, and in-flight reconfigurability. This thesis investigates FPGA-based acceleration of deep learning models for real-time target detection in optical Earth observation. The proposed *SR-YOLOv5s* network integrates a convolutional super-resolution backbone to enhance the recognition of small maritime targets, achieving a mean mAP₅₀ of 0.9272 ± 0.0029 and a 64.5% gain in very small object detection. Implemented on the Microchip PolarFire System-on-Chip with the CoreVectorBlox accelerator, the model achieves 18 fps while consuming less than 2 W of power, demonstrating the feasibility of compact, energy-efficient neural pipelines for size-, weight-, and power-constrained platforms such as CubeSats. Integration of the CoreVectorBlox within the ANHEO autonomous navigation unit demonstrated the advantages of FPGA reconfigurability through bitstream updates, yielding a $3.3\times$ performance improvement and a 50% energy reduction per frame in the satellite detection task. Altogether, the findings establish reconfigurable FPGAs as a key enabler of intelligent and resilient spacecraft, providing the capability to adapt onboard computational resources to evolving mission objectives, operational conditions, and algorithmic developments.

Keywords: Artificial Intelligence, FPGA-based Acceleration, Autonomous Spacecraft, Object Detection, Relative Navigation, Radiation Tolerance.

Sintesi in lingua italiana

L'elaborazione a bordo tramite intelligenza artificiale (IA) rappresenta un pilastro delle missioni spaziali di nuova generazione, abilitando decisioni autonome e in tempo reale che riducono la latenza e la dipendenza dalle comunicazioni a terra. Tuttavia, la maggior parte degli acceleratori commerciali non dispone della tolleranza alle radiazioni necessaria per missioni di lunga durata o nello spazio profondo. Gli FPGA qualificati per l'ambiente spaziale offrono un'alternativa affidabile, coniugando tolleranza alle radiazioni, efficienza energetica e capacità di riconfigurazione in volo. Questo lavoro esplora l'accelerazione mediante FPGA di modelli di *deep learning* per il rilevamento in tempo reale di target in immagini ottiche di osservazione terrestre. L'architettura proposta, *SR-YOLOv5s*, integra un modulo di super-risoluzione per migliorare il riconoscimento di piccole imbarcazioni nell'ambito della sorveglianza marittima, raggiungendo una mAP₅₀ media di 0.9272 ± 0.0029 e un incremento del 64.5% nella rilevazione di target molto piccoli. L'implementazione su *Microchip PolarFire System-on-Chip* con acceleratore *CoreVectorBlox* consente di ottenere 18 fps con un consumo inferiore a 2 W, dimostrando la fattibilità di pipeline neurali compatte ed efficienti per piattaforme come i CubeSat. L'integrazione del *CoreVectorBlox* nell'unità di navigazione autonoma *ANHEO* ha evidenziato i vantaggi della riconfigurabilità FPGA tramite aggiornamento del bitstream, con un miglioramento prestazionale di $3.3\times$ e una riduzione del 50% del consumo energetico per frame nell'operazione di *satellite detection*. Nel complesso, i risultati confermano il ruolo delle FPGA riconfigurabili come tecnologia abilitante per piattaforme spaziali intelligenti, capaci di adattare dinamicamente le proprie risorse computazionali all'evoluzione degli obiettivi di missione e degli algoritmi di bordo.

Parole chiave: Intelligenza Artificiale, Accelerazione FPGA, Autonomia, Object Detection, Navigazione Relativa, Tolleranza alle Radiazioni.

Acknowledgements

The research presented in this dissertation was supported by the Italian Ministry of University and Research (MUR, NextGenerationEU) under PNRR – D.M. 352/2022, Measure I.3.3 – Dottorati innovativi, with Techno System Developments S.r.l. (TSD-Space) as the industrial partner.

I am deeply grateful to *TSD-Space* for co-funding my Ph.D. scholarship and hosting me for a full year of research. Their involvement in national and international space programs allowed me to observe first-hand how scientific research is translated into operational space applications. I gratefully acknowledge their provision of essential technologies, facilities, and technical support, which were fundamental to this work.

I extend my sincere appreciation to my academic supervisors, *Prof. Antonio Strollo* and *Prof. Nicola Petra*, for their invaluable guidance and continued support throughout my doctoral studies.

I also wish to thank my colleague *Salvatore Capuozzo*, whose contribution to the experimental evaluation phase significantly strengthened the technical rigor of this dissertation.

I am thankful to the *European Space Agency* (ESA) for hosting me as a Visiting Researcher at ESTEC (Noordwijk, The Netherlands) for six months. I am particularly grateful to my supervisor, *Dr. Gianluca Furano*, for his guidance and support during my time at ESTEC, which proved invaluable to the development of this dissertation.



Contents

Abstract	i
Sintesi in lingua italiana	iii
Acknowledgements	v
List of Acronyms	xi
List of Figures	xxiii
List of Tables	xxvi
List of Symbols	xxvii
1 Introduction	1
1.1 Autonomous Earth Observation Systems	3
1.1.1 Remote Sensing Technologies	3
1.1.2 Small-Object Detection Challenges	5
1.1.3 Onboard Processing for Time-Critical Applications	7
1.2 Autonomous In-Space Operations	12
1.2.1 Autonomous Vision-Based Navigation	13
1.3 Reliable AI in Space	15
1.4 Main Contributions	17
2 Edge AI	21
2.1 Embedded GPU - NVIDIA Jetson Orin Nano	25
2.2 Intel Movidius Myriad X	28

2.3	Google Coral Edge TPU	30
2.4	FPGA-based Acceleration	31
2.4.1	Deployment Strategies	33
2.4.2	Network Optimization Techniques	36
2.4.3	Microchip CoreVectorBlox	37
2.4.4	AMD Xilinx DPU	43
3	Edge AI in Space	47
3.1	Edge AI Heritage	50
3.2	Space Environment	52
3.2.1	Radiation Effects on Electronics	55
3.3	FPGAs in Space	59
3.3.1	Fault Mitigation Strategy	62
3.3.2	Microchip RT PolarFire SoC	64
4	SR-YOLOv5s for Maritime Surveillance	67
4.1	Related Works	69
4.1.1	Super-Resolution	69
4.1.2	Object Detection	71
4.1.3	Super-Resolution for Object Detection	74
4.2	Materials and Methods	75
4.2.1	SR-Backbone	75
4.2.2	SR-YOLOv5s	76
4.2.3	Dataset and Training Details	77
4.2.4	SR-YOLOv5s Deployment	83
4.3	Results	84
4.3.1	Ship Detection	85
4.3.2	Hardware Deployment	95
5	ANHEO for Autonomous Navigation	101
5.1	ANHEO System Overview	102

5.1.1	System Architecture	103
5.1.2	Data Handling and Processing Modules	105
5.2	Vision-based Relative Navigation	106
5.2.1	Neural Network Pipeline	106
5.2.2	YOLOv5su Architecture	108
5.3	Neural Network Deployment Strategy	109
5.3.1	Initial Implementation with CoreVectorBlox v1	109
5.3.2	Bitstream Reconfiguration	110
5.3.3	ANHEO Performance Evaluation	115
6	Conclusions and Future Work	119
	Bibliography	123
	Author's publications	147

List of Acronyms

The following acronyms are used throughout the thesis.

AI	Artificial Intelligence
ANHEO	Autonomous Navigation up to High Earth Orbit
AOCS	Attitude and Orbit Control System
API	Application Programming Interface
APU	Application Processing Unit
ASIC	Application-Specific Integrated Circuit
ASI	Italian Space Agency
AXI	Advanced eXtensible Interface
B/B	BreadBoard
BLOB	Binary Large Objects
CLB	Configurable Logic Block
CMOS	Complementary Metal-Oxide-Semiconductor
CMX	Connection Matrix

CNN	Convolutional Neural Network
CoreVBX	CoreVectorBlox
COTS	Commercial-Off-The-Shelf
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DD	Displacement Damage
DDR	Double Data Rate
DHDB	Data Handling Daughter Board
DHMB	Data Handling Main Board
DHP	Data Handling and Processing
DL	Deep Learning
DLA	Deep Learning Accelerator
DMA	Direct Memory Access
DMIPS	Dhrystone Million Instructions Per Second
DNN	Deep Neural Network
DPU	Deep Learning Processing Unit
DSP	Digital Signal Processor
DRAM	Dynamic Random-Access Memory
ECSS	European Cooperation for Space Standardization
EDSR	Enhanced Deep Residual Network

EO	Earth Observation
ESA	European Space Agency
ESPCN	Efficient Sub-Pixel Convolutional Network
FLOP	Floating-Point Operation
FN	False Negative
FOPAC	FOcal Plane in-flight Active motion Control
FP	False Positive
FPA	Focal Plane Assembly
FPGA	Field Programmable Gate Array
FPS	Frame Per Second
FSRCNN	Fast SRCNN
FXP	Fixed-Point
GEO	Geostationary Earth Orbit
GFLOP	Giga Floating-Point Operation
GFLOPS	Giga Floating-Point Operations per Second
GNC	Guidance, Navigation, and Control
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPU	Graphics Processing Unit
GSD	Ground Sample Distance

GSFC	Goddard Space Flight Center
HA	Hardware Accelerator Board
HDL	Hardware Description Language
HEO	High Earth Orbit
HLS	High-Level Synthesis
HR	High-Resolution
IFOV	Instantaneous Field Of View
IMU	Inertial Measurement Unit
IOB	I/O Interface Board
IOD	In-Orbit Demonstration
IOS	In-Orbit Servicing
IoT	Internet of Things
IoU	Intersection Over Union
I/O	Input/Output
IP	Intellectual Property
IR	Intermediate Representations
ISL	Inter-Satellite Link
LE	Logic Element
LEO	Low Earth Orbit
LET	Linear Energy Transfer

LiDAR	Light Detection And Ranging
LoS	Line-of-Sight
LR	Low-Resolution
LUT	Look Up Tables
MAC	Multiply-Accumulate
MDK	Myriad Development Kit
MEKF	Multiplicative Extended Kalman Filter
MEO	Medium Earth Orbit
MIPS	Million Instructions Per Second
MISR	Multiple-Image Super-Resolution
ML	Machine Learning
MPE	Matrix of Processing Engine
MSE	Mean Square Error
MSI	Multispectral Imagers
MTF	Modulation Transfer Function
MXP	Matrix Processor
NASA	National Aeronautics and Space Administration
NCS 2	Intel Movidius Neural Compute Stick 2
NIR	Near Infrared
NMS	Non-Maximum Suppression

NN	Neural Network
NVML	NVIDIA Management Library
OBC	Onboard Computer
OD	Object Detection
ONNX	Open Neural Network Exchange
OS	Operating System
PCB	Printed Circuit Board
PCDM	Power Conditioning & Distribution Module
PE	Processing Element
PL	Programmable Logic
PS	Processing Systems
PSNR	Peak Signal-to-Noise Ratio
QNN	Quantized Neural Network
RAM	Random Access Memory
ReLU	Rectified Linear Unit
ROI	Region Of Interest
RQ	Research Question
RS	Remote Sensing
RT	Radiation Tolerant
RTL	Register-Transfer Level

SAR	Synthetic Aperture Radar
SC-LEARN	SpaceCube Low-power Edge AI Resilient Node
SDK	Software Development Kit
SEE	Single-Event Effect
SEFI	Single-Event Functional Interrupt
SEL	Single-Event Latchup
SET	Single-Event Transient
SEU	Single-Event Upset
SHAVE	Streaming Hybrid Architecture Vector Engines
SiLU	Sigmoid Linear Unit
SIMD	Single Instruction Multiple Data
SIMT	Single Instruction, Multiple Thread
SISR	Single-Image Super-Resolution
SM	Streaming Multiprocessors
SNR	Signal-to-Noise Ratio
SoC	System-on-Chip
SoM	System-on-Module
SR	Super-Resolution
SR-Backbone	Super-Resolution Backbone
SRAM	Static Random Access Memory

SRCNN	Super-Resolution Convolutional Neural Network
SSA	Space Situational Awareness
SSIM	Structural Similarity Index
STR	Star Tracker
SWaP	Size, Weight, and Power
SWIR	Short-Wave Infrared
TC	Telecommand
TFLOPS	Tera Floating-Point Operations per Second
TID	Total Ionizing Dose
TIR	Thermal Infrared
TM	Telemetry
TMR	Triple Modular Redundancy
TN	True Negative
TOPS	Tera Operations Per Second
TP	True Positive
TPU	Tensor Processing Unit
TRL	Technology Readiness Level
TSD-Space	Techno Systems Developments S.r.l.
VHDL	VHSIC Hardware Description Language
VIS	Visible

VLEO	Very Low Earth Orbit
VLIW	Very Long Instruction Word
VPU	Vision Processing Unit
YOLO	You Only Look Once

List of Figures

1.1	Along-track scanning for optical Earth observation	4
1.2	Impact of spatial resolution on target detection	6
1.3	Landsat-9 ground processing pipeline	9
1.4	Spatio-spectral scanning acquisition	10
1.5	Impact of onboard processing on latency reduction	11
1.6	Ground-centric maritime alerting workflow	12
1.7	Camera–target relative pose geometry	14
1.8	PRISMA formation-flying imagery	15
2.1	Difference between Cloud AI and Edge AI.	22
2.2	NVIDIA Jetson Orin Nano SOM.	26
2.3	Jetson Orin Nano block diagram	27
2.4	Myriad X VPU architecture	29
2.5	Intel Neural Compute Stick 2	30
2.6	Google Coral USB Accelerator	31
2.7	MAC operation and partial-sum update in a neuron	32
2.8	Dataflow architecture for on-chip FPGA deployment	34
2.9	Overlay architecture on FPGA SoC	35
2.10	Overlay architecture on pure FPGA	35
2.11	Neural network pruning illustration	37
2.12	CoreVectorBlox’s Internal Architecture [138]	38

2.13	VectorBlox SDK flow	40
2.14	Microchip PolarFire FPGA architecture [91]	42
2.15	Microchip PolarFire SoC architecture [92]	42
2.16	Microchip PolarFire SoC Video Kit	43
2.17	DPU architecture overview	44
3.1	Comparative performance of radiation-hardened processors and COTS GPUs	48
3.2	Space Radiation Sources [36]	53
3.3	Van Allen Belts Representation [87]	54
3.4	Displacement damage mechanism [160]	56
3.5	Single-event effect mechanism [195]	57
4.1	SRCNN architecture	70
4.2	FSRCNN architecture	70
4.3	Pixel-shuffle layer	71
4.4	YOLO architecture [121]	72
4.5	YOLO anchor-based prediction [121]	72
4.6	YOLO post-processing [121]	73
4.7	SR-Backbone architecture	76
4.8	SR-YOLOv5s architecture	77
4.9	ShipRSImageNet dataset	78
4.10	HR-ShipRSImage—Ship size class distribution (scatter plot)	79
4.11	HR-ShipRSImage—Ship size class distribution (pie chart) .	79
4.12	HR-ShipRSImage—Ship size class distribution (scatter plot)	80
4.13	HR-ShipRSImage—Ship size class distribution (pie chart) .	81
4.14	SR-YOLOv5s loss and accuracy during the first training step	83
4.15	CoreVectorBlox-based SoC Hardware Design	84
4.16	SR-Backbone reconstruction capability detail	85
4.17	Statistical evaluation of SR-YOLOv5s Vs. baseline YOLOv5s	87
4.18	Confusion matrices	88

4.19	Detection results under cloud occlusion	89
4.20	FP example	90
4.21	FN example	90
4.22	SR-YOLOv5s confusion matrices vs. confidence threshold	91
4.23	Confusion matrices on LR-CloudyTest	92
4.24	SR-YOLOv5s results on LR-CloudyTest	93
4.25	SR-YOLOv5s results on <i>MASATI-ship</i> subset	93
4.26	Confusion matrices on <i>MASATI-ship</i> subset	94
4.27	SR-YOLOv5s results on <i>MASATI-water</i> subset	94
4.28	Binary classification on SR-YOLOv5s detections.	95
4.29	SR-YOLOv5s power profile on PolarFire SoC	98
4.30	SR-YOLOv5s ship detections on PolarFire SoC Video Kit	100
5.1	FPGA deployment advantages: model-level flexibility and hardware-level reconfigurability.	102
5.2	ANHEO elegant B/B	103
5.3	ANHEO system architecture	104
5.4	ANHEO hardware accelerator	106
5.5	Relative pose estimation pipeline [145]	107
5.6	YOLOv5su inference workflow	110
5.7	Comparison between Sigmoid Linear Unit (SiLU) and Rectified Linear Unit (ReLU) activation functions	113
5.8	Satellite Detection	117

List of Tables

- 2.1 Embedded AI chip classification [110] 25
- 2.2 Jetson Orin module comparison [15] 26
- 2.3 CoreVectorBlox configurations [138] 38
- 2.4 Processor size configuration details 39

- 3.1 Computational and memory requirements of representative
CNN architectures 47
- 3.2 Radiation-tolerant processors commonly used in spaceborne
OBC 49
- 3.3 Performance and power consumption of edge AI accelera-
tors [43] 59
- 3.4 Radiation tolerance characteristics of AI accelerators for
space applications 59
- 3.5 Radiation Tolerance Characteristics of Space-Grade FPGAs 60

- 4.1 Super-Resolution Results 85
- 4.2 Model evaluation results on cross-validation test sets 86
- 4.3 Vessel detection accuracy 88
- 4.4 mAP50 results on LR-CloudyTest and MASATI (Ship) 92
- 4.5 Comparative benchmark of AI inference engines 96

5.1	YOLOv5s and YOLOv5su Architectures Comparison	109
5.2	CoreVectorBlox Processor Size Configuration Details	111
5.3	Performance Comparison of YOLOv5s–ReLU Across CoreVec- torBlox Versions	112
5.4	Model Performance Comparison Across CoreVectorBlox Ver- sions	115
5.5	CoreVectorBlox Performance Comparison	115
5.6	Performance Comparison Across Different HW Platforms . .	116

List of Symbols

The following symbols are used within the thesis

c Speed of light in a vacuum

h Planck constant

Chapter 1

Introduction

*The purpose of computing is insight,
not numbers.*

Richard W. Hamming

Space systems are undergoing a fundamental transformation from traditional ground-centric architectures toward autonomous frameworks capable of sensing, reasoning, and acting with minimal human intervention. This evolution is driven by critical operational constraints inherent to space missions: limited bandwidth combined with high latency in space-to-ground communication, intermittent ground station contact windows, and the imperative for rapid decision-making in dynamic scenarios spanning Earth Observation ([EO](#)), spacecraft navigation, in-orbit operations, and Space Situational Awareness ([SSA](#)) [[161](#)].

Traditional mission architectures centralize most decision-making within ground control centers. As mission complexity and constellation sizes grow, this paradigm encounters fundamental bottlenecks: downlink capacity consistently lags onboard data generation rates; communication delays imposed by orbital dynamics introduce substantial and variable latencies preventing timely ground intervention, and ground station visibility windows remain short and intermittent, fragmenting continuous operational oversight. These constraints render time-critical information products unavailable precisely when their operational value would be highest.

Onboard autonomy addresses these limitations by deploying compu-

tational capabilities directly aboard spacecraft, enabling real-time data processing *at the edge*. This paradigm shift offers three key advantages. First, executing algorithms adjacent to sensors minimizes end-to-end processing latency, enabling near-instantaneous response to detected events. Second, raw sensor measurements can be transformed onboard into compact, high-value data products—such as event notifications, extracted metadata, or prioritized image subsets—optimizing limited downlink bandwidth. Third, autonomous decision-making enables adaptive spacecraft behavior when communication delays or ground station outages make human-in-the-loop control operationally infeasible [161, 189]. Through these mechanisms, spacecraft can close the perception-to-action loop *in situ*, responding autonomously to detected phenomena without awaiting ground commands [75].

The maturation of Artificial Intelligence (AI) technologies—encompassing both Machine Learning (ML) and Deep Learning (DL)—provides the algorithmic foundation to realize this vision. Autonomous space systems are structured around functional layers mirroring the classical *sense-think-act* paradigm: a *sensing layer* for perception and information extraction, a *reasoning layer* performing state estimation and decision-making, and an *actuation layer* executing operational responses. AI techniques augment capabilities at each architectural level.

At the *sensing layer*, DL-based algorithms enable autonomous extraction of semantic information from multi-modal sensor streams through detection, classification, segmentation, and change analysis [76, 2, 81]. At the *reasoning layer*, learning-based techniques enhance state estimation, enable predictive modeling of spacecraft dynamics [78, 200] and environmental phenomena—including forecasting evolving Earth surface events [12, 203]—and support resource-constrained mission planning under uncertainty [173]. At the *actuation layer*, AI enables diverse autonomous responses: refining spacecraft Guidance, Navigation, and Control (GNC) [170]; generating time-critical alerts [211, 188]; and prioritizing data products. Collectively, these capabilities elevate operational autonomy, enabling spacecraft to respond independently to internal anomalies and external mission-relevant phenomena while substantially reducing ground-based intervention.

However, deploying autonomous systems in space requires establishing rigorous *trust and assurance frameworks* [209]. AI algorithms must demon-

strate consistent reliability, predictable behavior under diverse operational conditions, and sufficient accuracy to operate without continuous human oversight, necessitating verification and validation methodologies ensuring robustness against sensor noise, environmental disturbances, and hardware faults while maintaining safe operation throughout the mission lifecycle.

1.1 Autonomous Earth Observation Systems

EO represents a critical application domain for spacecraft autonomy. Modern missions face fundamental challenges—high data volumes, time-sensitive events, and limited downlink capacity—that autonomous systems can help address through onboard perception, reasoning, and decision-making.

Satellite Remote Sensing (RS) systematically acquires and analyzes electromagnetic radiation from Earth’s surface using spaceborne sensors operating across multiple spectral bands. This approach provides consistent, large-scale measurements with controlled spatial and temporal repeatability, producing calibrated quantitative data that support geophysical modeling and situational awareness [29].

1.1.1 Remote Sensing Technologies

Modern EO satellites employ diverse sensor technologies, each optimized for specific applications. Electro-optical sensors operating in visible and Near Infrared (NIR) wavelengths provide radiometrically calibrated surface reflectance measurements. Following atmospheric correction and geometric registration, these observations support intelligence applications—infrastructure monitoring, activity pattern detection, strategic asset assessment—and scientific applications including land-cover mapping, vegetation monitoring, agricultural assessment, and hydrological analysis [40].

These optical payloads typically employ a *pushbroom* scanning configuration, in which the spacecraft’s orbital motion combined with the sensor’s field-of-view geometry defines the spatial coverage and temporal sampling of the acquired imagery. As the instantaneous field of view (Instantaneous Field Of View (IFOV)) sweeps across the surface, the sensor line array progressively builds a continuous image swath (Figure 1.1). The combi-

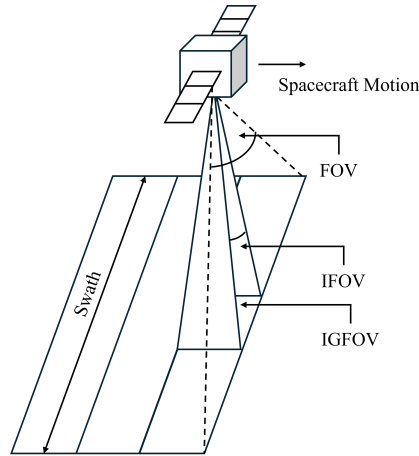


Figure 1.1. Along-track scanning for optical Earth observation

nation of orbital altitude, sensor pointing, and optical design determines the Ground Sample Distance (GSD)—the ground-projected size of a single detector element—and the swath width, while orbital dynamics and tasking strategies influence revisit frequency and overall data generation rates.

Hyperspectral sensors extend multispectral capabilities by capturing tens to hundreds of narrow, contiguous spectral bands, enabling detailed material discrimination for forest biophysical parameter retrieval, soil characterization, and geological exploration [219]. Active microwave Synthetic Aperture Radar (SAR) systems provide complementary all-weather, day-night imaging capabilities essential for persistent monitoring, surface deformation detection, and topographic mapping [184]. Thermal infrared sensors support temperature-based hazard assessment and energy balance studies.

Contemporary EO systems have advanced significantly across three dimensions. Spatially, modern satellites achieve sub-meter resolution—*WorldView-3* provides 31 cm panchromatic and 1.24 m multispectral imagery with sub-daily revisit capability [134, 76]. Spectrally, missions such as *PRISMA* offer unprecedented sampling density with 66 Visible (VIS)/NIR bands and 173 Short-Wave Infrared (SWIR) bands spanning 400–2505 nm at 30 m resolution [45]. Temporally, large constellations like *PlanetScope* approach near-daily global coverage with tasking capabilities enabling

frequent revisits to monitor rapidly evolving phenomena [65].

These advances in spatial, spectral, and temporal resolution collectively generate vast data volumes — often terabytes per satellite per day — necessitating automated intelligent processing to extract relevant information within operationally useful timeframes. Among available sensing modalities, optical imagery dominates operational applications [239], stemming from several factors: optical sensors provide high spatial resolution with intuitive visual interpretability, require less specialized processing compared to SAR, and benefit from decades of mature algorithms and operational workflows [229].

Many critical applications—disaster response, maritime surveillance, urban monitoring—require detecting specific targets within large-area imagery: individual vehicles, vessels, or infrastructure elements occupying tiny fractions of scenes yet representing mission-critical content. Automated target detection systems must identify these small, spatially sparse objects with high accuracy to enable effective operational decision-making.

1.1.2 Small-Object Detection Challenges

DL has significantly improved Object Detection (OD) performance in satellite imagery, establishing DL-based methods as standard approaches for environmental monitoring and surveillance [76]. Unlike traditional model-based approaches, DL architectures learn feature representations directly from data, enabling robust detection across diverse scenes, lighting conditions, and atmospheric variations, achieving state-of-the-art performance from land-use classification to vehicle tracking and maritime surveillance [113].

However, significant challenges remain detecting small targets—objects with limited spatial extent, low contrast against backgrounds, or presence in cluttered environments. When objects occupy few pixels, textural information becomes limited, structural cues weaken, and contextual information decreases, constraining feature extraction capabilities. Standard OD models (YOLO, SSD, RetinaNet, Faster R-CNN, EfficientDet) perform well on medium and large objects but show degraded accuracy for small targets or low-resolution imagery [226]. As deep networks downsample feature maps through successive convolutional and pooling layers, discriminative signals from small objects progressively attenuate, increasing missed detections



Figure 1.2. Impact of spatial resolution on target detection

and misclassifications [231].

Ship and vessel detection represents a particularly challenging application. The literature documents extensive methodological developments for maritime scenarios [24], with [241] surveying evolution from traditional algorithms to modern deep convolutional and transformer-based architectures, including two-stage frameworks (R-CNN family), single-stage models (YOLO, SSD, RetinaNet), and anchor-free and transformer-based approaches. Despite this diversity, fundamental challenges persist: cluttered and dynamic backgrounds, minimal contrast between targets and sea surfaces, and weak discriminative features—especially at coarse GSDs.

Figure 1.2 illustrates spatial resolution impact on target detectability through side-by-side comparison of the same maritime scene. The left panel displays high-resolution imagery at $\text{GSD}=0.5\text{ m}$, where a vessel exhibits clear shape definition, visible wake patterns, and distinct edges. The right panel shows corresponding low-resolution representation at $\text{GSD}=5\text{ m}$, obtained through $10\times$ downsampling. A $25\text{ m}\times 5\text{ m}$ patrol vessel—spanning approximately 50×10 pixels in high-resolution imagery—collapses to merely 5×1 pixels in low-resolution imagery. A red box highlights the vessel location, with an enlarged inset revealing how dramatic resolution reduction causes severe loss of shape information, edge definition, and discriminative features, rendering vessels nearly indistinguishable from background clutter and making automated detection extremely challenging.

To address these limitations, the literature explores multiple complementary strategies: multi-scale feature aggregation, high-resolution feature pyra-

mids, contextual reasoning modules, optimized anchor configurations, and learned super-resolution preprocessing [226, 231]. Super-Resolution (SR) techniques have been extensively investigated as preprocessing steps to enhance detail prior to detection, with studies reporting significant improvements for vehicles, aircraft, and maritime vessels when SR is applied before OD [226, 38, 198].

1.1.3 Onboard Processing for Time-Critical Applications

While DL techniques enable accurate automated detection of small targets in satellite imagery, their operational value ultimately depends on *when* and *where* processing occurs. In time-critical domains such as intelligence, defense, and disaster response, the timeliness of information can be as important as its accuracy. Many applications involve rapidly evolving events that demand immediate analysis: flood and wildfire monitoring rely on near-real-time imagery to support coordinated emergency response [53, 102]; urban surveillance benefits from prompt vehicle detection for traffic management and evacuation planning [8]; and maritime monitoring depends on fast ship detection to ensure safety and enforce maritime law [132]. In these contexts, low-latency, high-confidence alerts enable authorities to respond effectively, minimizing impact and potentially saving lives. Robust *vessel detection* under adverse conditions exemplifies this need: coast guard services and maritime agencies rely on timely alerts to coordinate *search-and-rescue* operations and enhance overall maritime safety.

However, generating time-critical alerts from space-based observations remains constrained when processing relies exclusively on ground infrastructure. The prevailing paradigm continues depending on ground-based image-processing facilities, introducing significant end-to-end latency between data acquisition and actionable intelligence delivery [55]. In traditional EO processing chains, raw or minimally compressed sensor data must be downlinked, archived, then reconstructed and analyzed through dedicated ground pipelines [31]. This bandwidth-intensive approach is complicated by frequent downlink of non-informative imagery. Clouds degrade large shares of optical EO imagery: pervasive cloud cover affects roughly 60–70% of acquisitions, leaving little actionable content [237], wasting scarce downlink bandwidth and consuming onboard storage without delivering usable information.

The fundamental bottleneck in ground-based EO architectures is *communication bandwidth* rather than computational capacity. Ground processing infrastructure can leverage elastic cloud computing and mature software ecosystems, whereas downlink capacity and ground station access windows remain severely constrained relative to sensor data generation rates [194]. In Low Earth Orbit (LEO), spacecraft complete orbital periods in approximately 90 minutes but maintain ground station contact for only ~ 10 minutes per pass, with typical visibility limited to four or five passes daily [220]. Modern sensors can generate several gigabytes during imaging sessions. Consequently, large data volumes accumulate onboard between transmission opportunities, introducing delays compromising operational intelligence value—particularly in time-sensitive applications.

To illustrate these constraints, consider *Landsat-9* [131], launched September 2021, acquiring multispectral optical data (VIS-NIR-SWIR 30 m spatial resolution with 15 m panchromatic) and thermal infrared observations (dual Thermal Infrared (TIR) bands at 100 m native resolution). Figure 1.3 illustrates the ground segment processing chain, encompassing a multi-stage workflow: raw telemetry (Level-0 data) undergoes X-band downlink to ground receiving stations, followed by radiometric calibration converting instrument digital numbers to at-sensor radiance and top-of-atmosphere reflectance using sensor-specific coefficients from pre-flight characterization and on-orbit calibration [197, 49]. Geometric correction processing uses ground control points and digital elevation models producing orthorectified products with sub-pixel geodetic accuracy. Finally, atmospheric correction algorithms apply radiative transfer modeling deriving surface reflectance products, yielding science-quality Level-1 and Level-2 datasets [218]. While this comprehensive pipeline ensures data quality and scientific rigor, it inherently introduces substantial temporal delays between observation and product availability—often several hours from acquisition to final delivery.

These bandwidth constraints are further exemplified by the *EarthNext* mission, a CubeSat developed within the ALCOR program—an initiative supported by Italian Space Agency (ASI) advancing innovative CubeSat missions. EarthNext is designed for In-Orbit Demonstration (IOD) of operating a CubeSat in Very Low Earth Orbit (VLEO). Operating at such low altitude offers particular interest due to VLEO’s advantages:

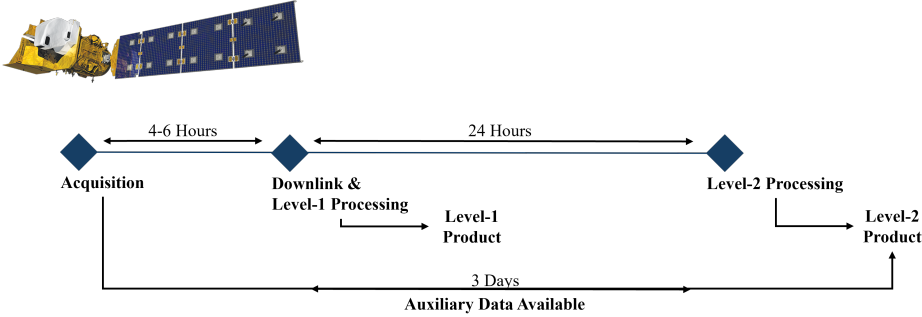


Figure 1.3. Landsat-9 ground processing pipeline

enhanced spatial resolution and radiometric performance for Earth observation, reduced revisit time, and possibility of using low-cost, volume-limited platforms [105].

Within EarthNext, Techno Systems Developments S.r.l. ([TSD-Space](#)) is responsible for camera electronics and the Focal Plane Assembly ([FPA](#)) of the multispectral payload, with *Officine Stellari* leading optics design. The [FPA](#) accommodates the CMV20000 detector, a Commercial-Off-The-Shelf ([COTS](#)) Complementary Metal-Oxide-Semiconductor ([CMOS](#)) image sensor by CMOSIS [106]. The multispectral imager implements a *spatio-spectral scanning acquisition mode*: a filter array routes selected spectral wavelengths to different regions (*sub-stripes*) of the 2D image sensor, comprising six spectral bands fixed above the detector array. In this mode, each acquired 2D frame contains a full spatial-resolution image composed of several spectral sub-stripes. Exploiting relative motion between satellite and ground, with appropriate frame acquisition rate, the imager scans the observed area producing full spatial-resolution images per spectral band, forming the *multi-spectral data cube*. Figure 1.4 depicts this technique. Each data cube amounts to approximately 166 Mbit.

At nominal altitude 320 km, the imager delivers products with [GSD](#) approximately 3.9 m and swath width about 17 km. Assuming 12 bit pixel depth, each spectral cube has volume roughly 116.5 Gbit. An uninterrupted 8 min imaging session produces about 51.8 Gbit.

[EO](#) missions typically employ X-band for payload downlink [48]. Commercial X-band transmitters offer data rates on the order of 100 Mbit s^{-1}

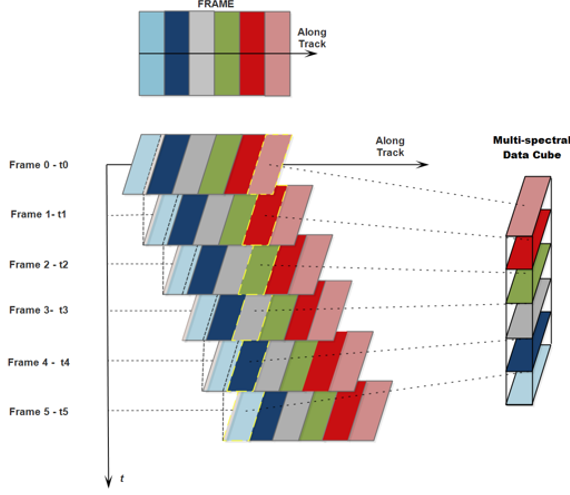


Figure 1.4. Spatio-spectral scanning acquisition

to 300 Mbit s^{-1} [191]. For small satellites (e.g., CubeSats), additional constraints arise from limited power budgets directly capping achievable downlink throughput [193].

For the *EarthNext* mission scenario, we consider X-band transmitter with nominal rate 150 Mbit s^{-1} and mean contact duration 4 min, yielding per-pass downlink volume:

$$150 \text{ Mbit s}^{-1} \times 240 \text{ s} = 36 \text{ Gbit.}$$

Consequently, even a short 8 min imaging session generating 51.8 Gbit necessitates at least two ground-station passes for downlink, delaying product availability to operational end users.

In contrast, *onboard processing* paradigms execute inference algorithms directly in orbit on raw sensor data streams. When embedded AI models identify operationally relevant events (e.g., vessel or vehicle detections), only essential mission-critical metadata—geolocation coordinates, classification labels, confidence scores—are transmitted as compact alert messages during next ground contact. This *event-driven downlink* strategy simultaneously reduces communication latency and bandwidth requirements, improving intelligence timeliness in dynamic scenarios [52], as illustrated in Figure 1.5.

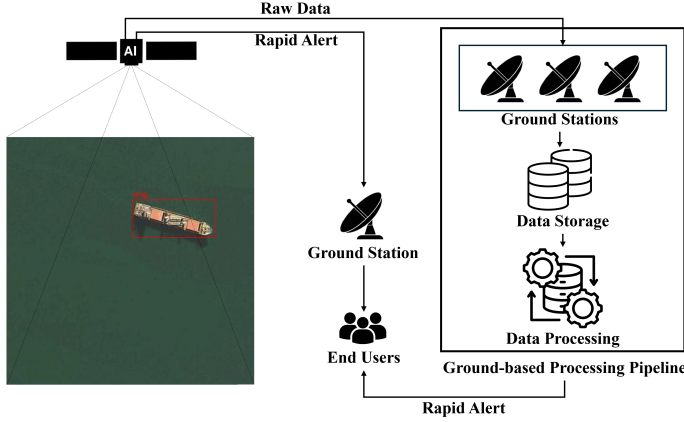


Figure 1.5. Impact of onboard processing on latency reduction

The operational impact of these architectural differences becomes particularly evident in time-sensitive applications. Consider the maritime domain awareness scenario depicted in Figure 1.6, which illustrates three phases of the conventional EO pipeline: (a) data acquisition, where the satellite images a suspicious vessel; (b) downlink and processing, where sensor data transmit across successive ground contacts for analysis; and (c) alert dissemination, where vessel position intelligence exhibits significant staleness due to cumulative end-to-end delays ($\Delta t \approx$ hours). This temporal lag reduces intercept probability and compromises operational utility. A vessel engaged in suspicious activities may have altered course or departed the area entirely by the time actionable intelligence becomes available.

While this example illustrates traditional processing limitations for time-critical operations, *ground segment* architectures remain well-suited for non-time-critical applications—scientific research, climate modeling, strategic monitoring—where analytical accuracy, cross-mission data fusion, and methodological flexibility are primary requirements. However, in time-sensitive operational domains (disaster response, defense operations), multi-hour delays compromise operational utility. User requirement analyses identify a critical *delivery threshold* of ≤ 3 hours for actionable intelligence products in such contexts [27]. This constraint underscores

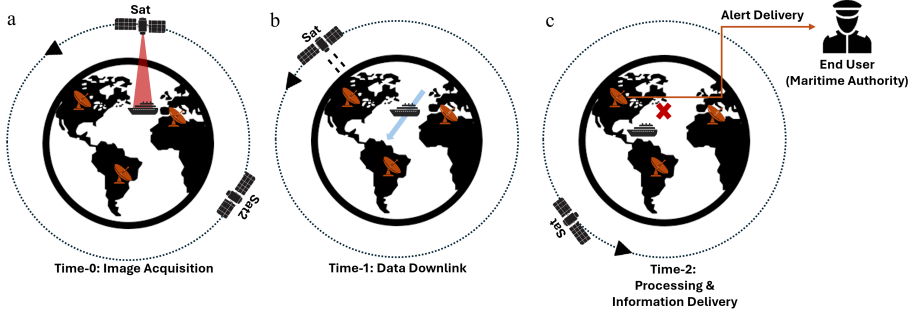


Figure 1.6. Ground-centric maritime alerting workflow

onboard processing importance, enabling near-real-time transmission of compact, high-value information products, simultaneously reducing down-link bandwidth requirements and operational response latency.

1.2 Autonomous In-Space Operations

Beyond onboard autonomy need to convert raw EO measurements into mission-ready information, modern spacecraft require *safe and reliable* autonomy for *deep-space exploration* [62] and *in-orbit operations* such as inspection and on-orbit servicing, generally involving proximity operations like *rendezvous*, *docking*, and *formation flying* [149]. In these regimes—characterized by limited ground-contact windows, variable communication latency, and stringent safety constraints—the ability to *perceive, estimate, and act onboard* (i.e., close the GNC loop at the edge) is essential for timely decision-making, operational continuity, and fault tolerance.

Traditional spacecraft operations are *sequence-driven*: command sequences are generated on ground, uplinked, and executed by onboard computers and flight software with largely linear, time-tagged control flow. Uncertainty is handled conservatively—activities are time-buffered for worst-case durations and safeguarded by basic health monitoring that may abort execution rather than adapt—so maneuvers remain planned and supervised from ground control rather than decided onboard. Control and monitoring are performed by *ground control* through Telemetry (TM)/Telecommand

(TC) links [222].

However, ground-in-the-loop sequencing can be too slow and brittle. Communication delays or interruptions can compromise mission success, particularly during proximity and terminal phases. During In-Orbit Servicing (IOS) operations, critical maneuvers such as inspection, approach, and capture demand near-instantaneous sensing-estimation-control loops. Communication delays inherent in ground-in-the-loop architectures render such control impractical, necessitating onboard autonomous decision-making [164, 209].

This imperative for autonomy becomes even more pronounced in deep-space missions, where communication latencies are orders of magnitude larger. Planetary exploration represents a compelling case: Mars rover operations face one-way communication delays up to 24 min, making real-time teleoperation infeasible. Autonomy enables onboard terrain analysis, route planning, and obstacle avoidance, allowing rovers to make time-critical decisions without waiting for Earth-based instructions [21].

1.2.1 Autonomous Vision-Based Navigation

Addressing the operational requirements for autonomous spacecraft operations demands advanced sensing and processing technologies. Among these, *vision-based navigation systems* powered by DL have emerged as key enablers, leveraging camera observations and neural network inference to estimate spacecraft position and attitude during proximity or planetary operations [205]. By complementing traditional sensors—such as Global Navigation Satellite System (GNSS), Inertial Measurement Unit (IMU), and star trackers—these systems enable autonomous guidance, navigation, and control by providing continuous state estimates without ground intervention [232].

Vision-based navigation can be categorized into *absolute* navigation—which determines spacecraft position and attitude with respect to an inertial reference frame—and *relative* navigation, which estimates these quantities with respect to another spacecraft or known target. Within the latter category, *relative pose estimation* plays a pivotal role: it retrieves the position and orientation of a target spacecraft with respect to a chaser, supporting mission-critical operations such as rendezvous, docking, and formation flying [163]. Figure 1.7 illustrates this problem, where the

objective is to estimate the target’s pose from two-dimensional image measurements.

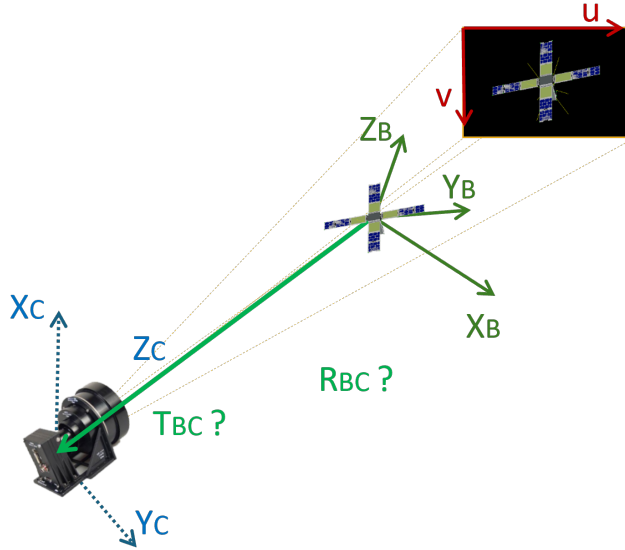


Figure 1.7. Camera–target relative pose geometry

For spacecraft constrained by mass, volume, and power budgets—such as CubeSats and small satellites—*monocular camera systems* represent an attractive solution. Compared with three-dimensional sensing modalities such as stereo vision and Light Detection And Ranging ([LiDAR](#)), monocular vision offers substantially lower power consumption while maintaining effective navigation capabilities [33]. However, monocular pose estimation becomes particularly challenging when dealing with *uncooperative targets*, which lack visual markers or communication links required by conventional navigation techniques. Recent advances in deep learning have addressed these challenges, significantly improving robustness to adverse illumination, reflections, and partial occlusions [103]. A comprehensive survey of deep-learning-based pose estimation methods can be found in [168].

The development and validation of these algorithms rely on standardized benchmarks, among which the SPEED dataset [103]—introduced through the *Satellite Pose Estimation Challenge*—has become a reference in the field.

The dataset comprises both synthetic and real imagery of the *Tango* spacecraft, acquired during the *PRISMA* formation-flying mission. *PRISMA*, launched on 15 June 2010 by the Swedish Space Corporation (SSC), was a technology demonstration mission consisting of two spacecraft—*Mango* (chaser) and *Tango* (target)—designed to validate guidance, navigation, and control strategies for autonomous rendezvous and formation flying. [TSD-Space](#) contributed to *PRISMA* by developing a high-resolution, high-frame-rate video system for acquiring sequences during in-orbit maneuvers and Earth observation [32]. Figure 1.8 shows representative formation-flying imagery captured by the onboard monitoring camera installed on the *Mango* spacecraft.

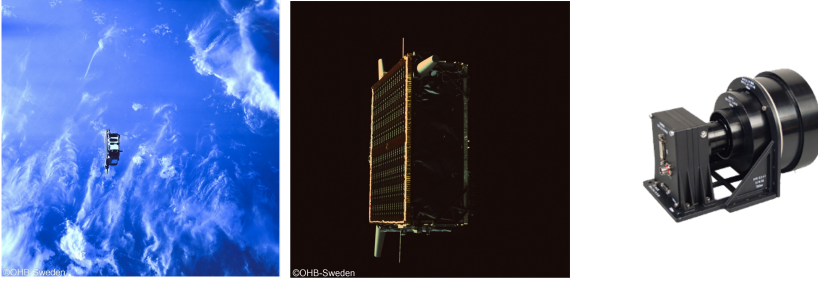


Figure 1.8. PRISMA formation-flying imagery

1.3 Reliable AI in Space

Deploying [DL](#) algorithms onboard spacecraft presents fundamental hardware challenges. Model complexity often demands significant power consumption and extensive memory, constraining their use in space systems with strict resource budgets [69]. Conventional *space-grade processors*, such as *PowerPC-FT* and *LEON* cores, typically fall short of the performance required for real-time image processing [125], particularly in [AI](#)-powered tasks, as these classical onboard processors cannot cope with the increased demands for high Input/Output ([I/O](#)) bandwidth and processing throughput.

For ground applications, advances in [AI](#) have been driven by widespread availability of high-performance Graphics Processing Units ([GPUs](#)). Unlike

traditional Central Processing Units (CPUs) with complex control logic and a small number of powerful cores optimized for sequential processing, GPUs employ a fundamentally different architectural paradigm based on massive parallelism [119]. Modern GPUs integrate thousands of simpler processing cores organized in a Single Instruction, Multiple Thread (SIMT) execution model, where each core operates on different data elements simultaneously. This architecture, combined with high memory bandwidth and hierarchical memory systems, makes GPUs exceptionally well-suited for the matrix operations and tensor computations that dominate Deep Neural Network (DNN) workloads [96]. As model complexity has grown from millions to billions of parameters, successive GPU generations have provided increasing computational throughput and memory capacity, enabling development of progressively larger and more accurate DNNs. However, the high power consumption and thermal demands of GPUs constrain their use in energy-sensitive or Size, Weight, and Power (SWaP)-limited settings, including embedded and spaceborne platforms [70].

Thermal management represents a critical constraint in such environments: for each watt dissipated by computing hardware, additional power must be allocated to active cooling systems—a requirement particularly challenging in spacecraft where radiative heat rejection is the only viable mechanism, and in small platforms where passive thermal solutions have limited capacity [235]. Consequently, low-power computing architectures reduce both primary power consumption and secondary thermal control requirements, directly addressing the stringent energy budgets of battery-powered systems and the limited heat dissipation capabilities of compact form factors.

To meet these constraints while enabling onboard AI capabilities, newer power-efficient inference engines—such as embedded GPUs and Application-Specific Integrated Circuit (ASIC)-based platforms (e.g., Intel’s Myriad 2 Vision Processing Unit (VPU) and Google’s Coral Tensor Processing Unit (TPU))—have emerged, delivering low-power inference at the edge with remarkable performance [67]. These commercial COTS devices offer significant advantages in terms of *cost-efficiency* and *ease of integration*, reducing overall mission development time and cost. Consequently, their deployment onboard satellites has gained considerable traction in recent years, particularly in LEO missions where moderate radiation levels permit the use

of general-purpose hardware for experimental AI applications. European Space Agency (ESA)’s *PhiSat-1* mission represents a successful example, deploying a Convolutional Neural Network (CNN)-based cloud-detection algorithm on the Intel Movidius Myriad 2 accelerator [71]. However, COTS devices are built with commercial silicon technology that does not meet the stringent radiation-tolerant requirements of more demanding space missions [176].

Safety-critical applications—including autonomous rendezvous and docking, as well as sensitive Earth observation data extraction—require exceptionally high reliability. These demanding requirements drive the need for alternative architectural solutions to deploy AI models on spacecraft while balancing computational performance with radiation tolerance and operational robustness.

Field Programmable Gate Arrays (FPGAs) have long flight heritage in space systems [227]. Today, space-grade devices, including radiation-hardened and Radiation Tolerant (RT) processors, are commercially available and qualified for operation in harsh environments, including long-duration missions in Geostationary Earth Orbit (GEO) and beyond. Architecturally, an FPGA integrates a reconfigurable fabric of configurable logic blocks, embedded memories, Digital Signal Processor (DSP) slices, and high-speed interconnect (e.g., multi-gigabit serial I/O), which can be tailored to implement application-specific datapaths. By customizing datapath and control as dedicated circuits, many workloads can be executed more efficiently than on general-purpose processors. In modern spacecraft, FPGAs commonly serve as the computational core of Data Handling and Processing (DHP) units, facilitating high-rate sensor acquisition, deterministic routing, on-board preprocessing, and mass-memory interfacing. This flexibility also enables the realization of custom accelerators—up to and including low-latency AI inference—while meeting stringent SWaP, reliability, and radiation-environment constraints.

1.4 Main Contributions

This thesis investigates the deployment of AI-based algorithms on resource-constrained, radiation-tolerant hardware platforms to enable autonomous spacecraft operations, addressing two critical application do-

mains: onboard processing for time-critical Earth observation, focusing on small-target detection in optical satellite imagery, and vision-based relative navigation for proximity operations with uncooperative targets. The central technical challenge is achieving real-time inference performance under stringent power, memory, and radiation-tolerance constraints typical of spaceborne systems. To address these requirements, the thesis examines [FPGA](#)-based acceleration as a viable path toward reliable onboard [AI](#), with particular emphasis on the Microchip RT PolarFire family and the CoreVectorBlox ([CoreVBX](#)) overlay accelerator.

The first major contribution presents *SR-YOLOv5s*, an object detection network integrating a lightweight, fully-convolutional Single-Image Super-Resolution ([SISR](#)) backbone designed for deployment on embedded platforms. The SR-Backbone—comprising 16,960 parameters—is permanently integrated into the detection pipeline and jointly trained with the detector to enhance fine-grained spatial detail extraction from low-resolution inputs. To evaluate the proposed architecture under operationally relevant conditions, two new datasets are introduced: *LR-ShipRSImage*, derived from ShipRSImageNet [240] and containing over 13,200 ship instances including 4,000+ very small vessels ($\leq 7 \times 7$ pixels) and 9,200+ small vessels, capturing diverse maritime conditions; and *LR-CloudyTest*, a synthetic test subset with procedurally generated cloud masks simulating degraded visibility scenarios. Experimental results demonstrate that SR-YOLOv5s achieves $\text{mAP50} = 0.927 \pm 0.003$ on low-resolution inputs, representing an 8.54% relative improvement over baseline YOLOv5s, with particularly pronounced gains for challenging object classes: +64.48% for very small targets and +16.31% for small targets. Statistical validation through 5-fold cross-validation and bootstrap resampling confirms significance ($p < 0.001$), with generalization validated on the MASATI-ship subset achieving $\text{mAP50} = 0.751$ versus 0.529 for the baseline.

To establish a practical path toward onboard deployment, SR-YOLOv5s is implemented and benchmarked across three distinct computing platforms. Deployment on the Microchip PolarFire SoC Video Kit via the VectorBlox SDK and [CoreVBX](#) overlay accelerator enables inference at 18 fps with 2.46 W average power consumption, achieving an accuracy-to-power ratio of $\text{mAP50}/\text{W} = 0.366$ —the highest among evaluated platforms—while maintaining $\text{mAP50} = 0.916$ under INT8 quantization. Evaluation on the

NVIDIA Jetson Orin Nano across multiple precision formats (FP32, FP16, INT8) and power modes reveals energy-performance trade-offs, with 15 W mode INT8 achieving 16.0 ms latency and 104.2 mJ/frame but reduced accuracy (mAP50 = 0.843). The NVIDIA A100 GPU serves as a cloud-based reference, achieving lowest inference latency (7.3 ms) while maintaining full-precision accuracy but consuming ~ 53 W and 384.2 mJ/frame. These results establish that the PolarFire SoC implementation achieves the best balance of accuracy, energy efficiency, and radiation tolerance for spaceborne applications, with the hardware design fully portable to the RT PolarFire SoC for radiation-qualified deployment.

The second major contribution demonstrates onboard AI deployment through the Autonomous Navigation up to High Earth Orbit (ANHEO) system—a compact navigation unit developed by TSD-Space and funded by ASI, currently at Technology Readiness Level (TRL)4. The system employs vision-based relative navigation for proximity operations with uncooperative targets, executing DL-based target detection and pose estimation algorithms on an RT PolarFire FPGA-based hardware accelerator.

Deployment on ANHEO demonstrates the critical advantages of FPGA-based acceleration for spaceborne AI. The initial implementation using CoreVectorBlox v1 required hardware/software partitioning due to limited layer support, achieving 791.5 ms inference latency with 0.58 W dynamic power (459 mJ/frame). Upgrading to CoreVectorBlox v2 via bitstream reconfiguration—*without any hardware modifications*—enabled fully hardware-accelerated inference, achieving $3.3\times$ speedup (240.5 ms), 50% energy reduction (230.8 mJ/frame), and maintained accuracy (mAP50 = 0.962 vs. 0.972 FP32 reference). This upgrade validates the operational value of in-flight reconfigurability, where RT PolarFire devices support up to 500 reprogramming cycles for post-deployment functional improvements.

The main original contributions of this thesis are:

1. **SR-YOLOv5s architecture:** A super-resolution-enhanced object detector integrating a lightweight SR-Backbone (16,960 parameters) optimized for FPGA deployment, achieving +64.48% improvement for very small targets.
2. **LR-ShipRSImage dataset:** A novel dataset for small vessel detection comprising 4,000+ very small ships ($\leq 7 \times 7$ pixels) and

9,200+ small ships, enriched with synthetic cloud-masked imagery (LR-CloudyTest) for robustness evaluation.

3. **Multi-platform deployment framework:** Comprehensive benchmarking across PolarFire SoC (18 fps, 2.46 W, mAP50/W=0.366), Jetson Orin Nano, and A100 GPU, establishing energy-accuracy-throughput trade-offs for spaceborne AI.
4. **ANHEO system integration:** TRL4 demonstration of vision-based relative navigation on a compact, reconfigurable platform (0.8 U, <10 W) integrating DL-based pose estimation for proximity operations.
5. **In-flight FPGA reconfigurability validation:** Demonstration of CoreVectorBlox v1→v2 upgrade via bitstream reconfiguration achieving $3.3\times$ speedup and 50% energy reduction without hardware changes, validating post-deployment adaptability for long-duration missions.

These contributions collectively establish a viable path toward deploying reliable, energy-efficient AI systems on radiation-tolerant hardware platforms for autonomous spacecraft operations.

Chapter 2

Edge AI

Early embedded intelligence architectures operated in a *cloud-first regime*, wherein edge devices streamed raw sensor data to remote servers for centralized inference. *Edge computing* decentralizes computational workloads from remote data centers to the point of data acquisition. When coupled with artificial intelligence, it enables on-device inference and autonomous decision-making directly on sensors and embedded systems, removing the need for continuous cloud interaction.

In this paradigm, *Edge AI* refers to the deployment of *perception*, *prediction*, and *control* functionalities at the network periphery, where sensory signals originate [196]. Transitioning from cloud-centric to edge-native execution addresses fundamental limitations inherent to server-based architectures: propagation latency and temporal variability, bandwidth saturation and associated transmission costs—particularly for high-resolution video streams—privacy vulnerabilities and data-sovereignty concerns, reliance on uninterrupted connectivity, and computational scalability constraints [68].

The cloud infrastructure remains essential for computationally intensive tasks—model training, fleet orchestration, Over-The-Air (OTA) updates, and latency-tolerant analytics—that benefit from unconstrained compute resources unavailable at the edge due to stringent power, thermal, and form-factor limitations [169].

As illustrated in Figure 2.1, the *cloud-first* pipeline requires transmitting large volumes of raw data to the *cloud*, incurring bandwidth pressure and multi-pass latency before analysis can begin. By contrast, *edge/onboard*

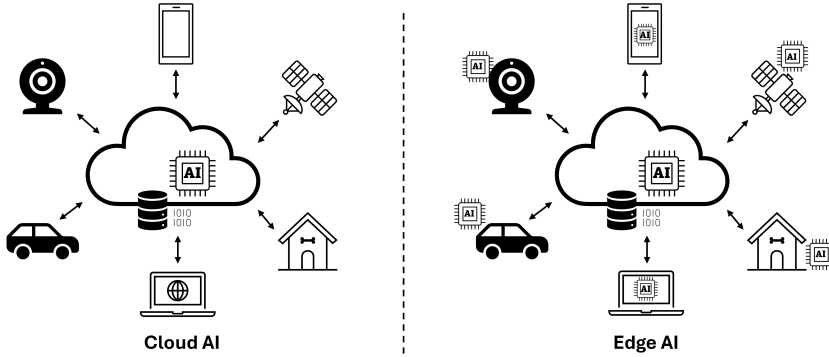


Figure 2.1. Difference between Cloud AI and Edge AI.

AI executes inference directly at the source: compact neural networks and specialized accelerators perform real-time inference locally and upload compact, event-driven metadata (e.g., geotagged detections and confidence scores), thereby reducing bandwidth, lowering time-to-alert, and improving resilience to intermittent links.

This evolution rests on three foundational pillars: (i) specialized hardware architectures featuring massive parallelism—processing element arrays, multiply-accumulate units, systolic datapaths, and hierarchical memory subsystems with optimized data movement patterns; (ii) neural network compression methodologies—structured pruning, low-precision quantization to INT8/INT4 representations, and knowledge distillation; and (iii) device-specific compiler toolchains and runtime optimizations [110].

In recent years, *edge devices* incorporating dedicated artificial intelligence accelerators have proliferated, primarily targeting the efficient execution of *neural network* inference workloads. They are typically implemented as System-on-Chip (**SoC**) or System-on-Module (**SoM**) architectures [143]. A **SoC** integrates an entire computational system—**CPU**, memory controllers, **GPU** or specialized accelerators (e.g. **ASIC**) as well as **FPGA**, input and output interfaces, and communication peripherals (Ethernet, Wi-Fi, USB)—onto a single silicon die, minimizing physical footprint, reducing unit cost, and improving power efficiency through tight integration and optimized data paths. The operating system is stored in non-volatile

memory (e.g., flash or eMMC) and loaded into RAM at boot time, rather than being permanently embedded in the silicon die [202]. Within SoC architectures, AI acceleration engines are typically implemented as dedicated *coprocessors* that offload vector and tensor computations from the host CPU, enabling efficient execution of machine learning workloads—from classical algorithms to deep neural network inference [114]. These coprocessors operate in tandem with the primary processor, which manages system-level tasks, I/O coordination, and control flow, while the AI coprocessor handles compute-intensive parallel operations on mathematical kernels.

A SoM extends this concept by packaging a SoC alongside essential support components—DRAM, flash storage, power management ICs, and clock generators—onto a compact, standardized module with well-defined electrical and mechanical interfaces. This modular approach enables rapid prototyping and system integration by abstracting low-level hardware complexity, providing a complete computing subsystem that can be directly integrated into carrier boards tailored to specific applications [202]. SoCs and SoMs are widely deployed across mobile devices, embedded systems, and Internet of Things (IoT) platforms.

Edge AI accelerators can be classified according to their underlying computational architecture:

1. **ASICs:** Custom silicon optimized for neural network operations delivers maximum computational efficiency and minimal unit cost among acceleration paradigms. However, ASICs incur substantial non-recurring engineering expenses and extended development cycles. Their architecture is fundamentally optimized for specific layer types and computational workflows, rendering them inflexible for unsupported operations. Post-fabrication immutability constrains their ability to accommodate emerging network architectures or novel layer primitives, thereby limiting adaptability to evolving model topologies and algorithmic innovations [129]. Google’s Edge TPUs or Intel’s Movidius Myriad VPU exemplify domain-specific ASIC design.
2. **FPGAs:** Provide hardware-level implementation of neural network operations through reconfigurable logic synthesis, enabling post-deployment adaptation to evolving algorithmic requirements. This architectural approach achieves a favorable power-to-performance ratio

compared to general-purpose processors such as GPUs, though at the cost of reduced programming flexibility [110]. While ASICs deliver superior energy efficiency and throughput for fixed workloads—making them optimal for energy-constrained, high-volume applications—they lack the algorithmic reconfigurability that FPGAs inherently provide. This reconfigurability allows FPGAs to accommodate network architecture updates and in-flight algorithm modifications without hardware replacement, a capability particularly valuable in space missions where post-launch adaptability is essential. FPGAs typically employ fixed-point arithmetic for neural network inference, necessitating model quantization. This constraint, however, enables efficient deployment of very deep Neural Networks (NNs) with high power efficiency and minimal accuracy degradation [54]. Representative platforms include Microchip’s PolarFire CoreVectorBlox and AMD Xilinx’s Zynq Deep Learning Processing Unit (DPU) series [10].

3. **Embedded GPUs:** Are compact, power-efficient versions of datacenter GPUs, designed for resource-constrained platforms. Their optimized GPU cores deliver high performance per watt, while established frameworks such as *CUDA*[156] and *cuDNN*[157] provide mature technology, low cost, and wide software compatibility for rapid DNN development [143]. Embedded GPUs generally achieve higher throughput than FPGAs, making them well-suited for compute-intensive inference workloads. The NVIDIA Jetson series exemplifies this category [158].

Table 2.1 summarizes the architectural trade-offs among these acceleration paradigms. ASICs offer the highest computational performance and lowest power consumption but sacrifice versatility and impose long development cycles. GPUs provide high programmability and rapid development at the cost of increased power consumption. FPGAs occupy an intermediate position, balancing reconfigurability with power efficiency, though requiring specialized design expertise. The selection of an appropriate accelerator architecture depends on application-specific constraints—mission adaptability requirements, power budgets, development timelines, and the expected evolution of deployed algorithms over the operational lifetime.

The following subsections provide high-level architectural overviews of

Table 2.1. Embedded AI chip classification [110]

AI chip	Versatility	Performance	Power consumption	Cost	Development cycle
GPU	High	Middle	Higher	Middle	Short
FPGA	Middle	High	Low	Low	Middle
ASIC	Low	Highest	Low	Lowest	Long

edge AI platforms, focusing on Intel Movidius Myriad X [VPUs](#), Google Coral [TPUs](#), NVIDIA Jetson embedded [GPUs](#), and [FPGAs/SoCs](#). For [FPGA](#)-based implementations, the discussion encompasses synthesizable hardware designs, overlay-based architectures, and dataflow acceleration approaches, covering both flash-based technologies—exemplified by Microchip’s PolarFire [FPGA](#) and [SoC](#)—and SRAM-based platforms such as AMD Xilinx Zynq UltraScale+ [MPSoC](#) .

2.1 Embedded GPU - NVIDIA Jetson Orin Nano

The NVIDIA *Jetson* family exemplifies the embedded [GPU](#) category discussed previously, offering a range of [SoM](#) platforms that balance computational throughput and power efficiency for edge [AI](#) applications. The portfolio spans from ultra-low-power modules for battery-operated devices to high-performance variants executing multiple concurrent [DNN](#) workloads in autonomous systems and industrial automation. The *Jetson Orin* series comprises three distinct subfamilies, each targeting specific performance and power envelopes: *AGX Orin* delivers maximum computational capabilities for demanding applications; *NX Orin* provides balanced performance for mid-range deployments; and *Orin Nano* prioritizes energy efficiency for power-constrained scenarios. Each module combines an ARM [CPU](#) for system management, an Ampere [GPU](#) with CUDA and Tensor cores for parallel processing, and optionally dedicated Deep Learning Accelerator ([DLA](#)) for energy-efficient inference. Notably, the Orin Nano subfamily lacks [DLA](#) cores and depends entirely on its [CPU-GPU](#) architecture for neural network execution. Table 2.2 summarizes the architectural specifications and performance characteristics across the Orin family.

This work focuses on the Jetson Orin Nano [SoM](#) (Figure 2.2), which provides a representative example of embedded GPU architecture for resource-

Table 2.2. Jetson Orin module comparison [15]

Jetson Orin module	Nano		NX		AGX	
	4 GB	8 GB	8 GB	16 GB	32 GB	64 GB
CPU cores	6	6	6	8	8	12
CUDA cores	512	1024	1024	1024	1792	2048
Tensor cores	16	32	32	32	56	64
DLA cores	—	—	1	2	2	2
TOPS	20	40	70	100	200	275
Power (W)	5–10	7–15	10–20	10–25	15–40	15–60

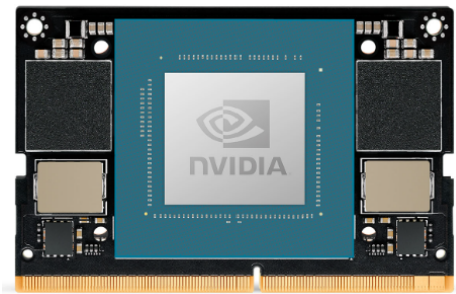


Figure 2.2. NVIDIA Jetson Orin Nano SOM.

constrained applications.

The module integrates a heterogeneous SoC combining a 6-core ARM Cortex-A78AE CPU for general-purpose computing with an NVIDIA Ampere GPU architecture optimized for AI acceleration. The GPU subsystem comprises eight Streaming Multiprocessors (SMs), each containing 128 Compute Unified Device Architecture (CUDA) cores and 4 Tensor Cores, yielding a total of 1024 CUDA cores and 32 Tensor Cores across the device. This configuration enables acceleration of computationally intensive tasks including OD, natural language processing, and robotic perception algorithms. For experimental benchmarking throughout this work, the 8 GB LPDDR5 memory variant is adopted, operating in two configurable power modes—15 W and 7 W—with peak performance reaching 40 Tera Operations Per Second (TOPS).

The GPU memory hierarchy comprises multiple levels: global memory and L2 cache are shared across all SMs, while each SM maintains a private

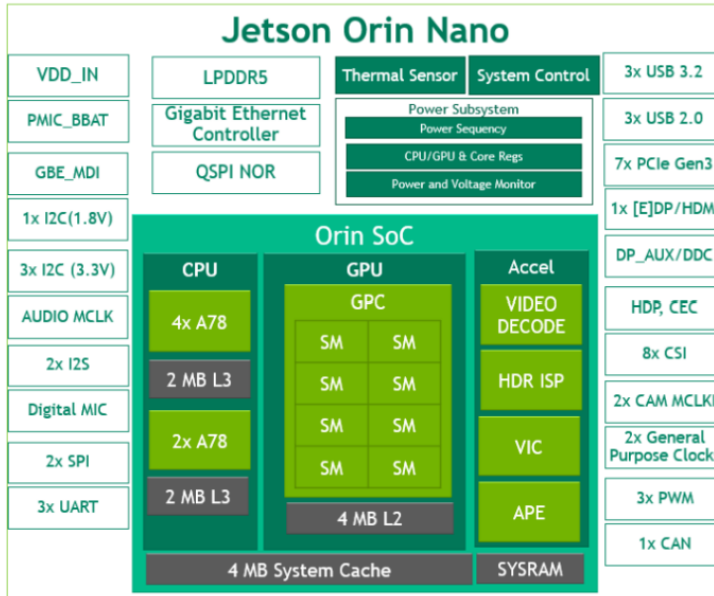


Figure 2.3. Jetson Orin Nano block diagram

L1 cache. Additionally, threads within a block communicate through shared memory, enabling efficient data exchange during parallel computations. The computational fabric integrates two complementary processing elements. **CUDA** cores function as general-purpose units executing floating-point and integer operations concurrently, supporting diverse workloads spanning graphics rendering, scientific computing, and machine learning inference. Tensor Cores operate as specialized matrix multiplication engines, accelerating the matrix-multiply-accumulate operations that dominate deep neural network training and inference workloads [1]. By executing in parallel with **CUDA** cores, Tensor Cores substantially increase throughput for **AI**-specific computations. Figure 2.3 illustrates the architectural organization.

For experimental deployment, the *Jetson Orin Nano Developer Kit* [154] is adopted as the representative edge **AI** platform. The kit integrates the Jetson Orin Nano **SoM** with a carrier board exposing PCIe, USB 3.2, HDMI, and other **I/O** interfaces, enabling straightforward integration of sensors and peripherals. The platform is supported by NVIDIA's *JetPack*

SDK [155], providing a comprehensive toolchain (CUDA, cuDNN, TensorRT) and first-class support for popular AI frameworks such as PyTorch and TensorFlow [159].

2.2 Intel Movidius Myriad X

The *Intel Movidius Myriad* family comprises multicore SoCs specifically designed for mobile, wearable, and embedded platforms requiring computational imaging and visual perception capabilities. The VPU architecture combines extensive parallelism, a specialized instruction set, and targeted microarchitectural optimizations to deliver exceptional performance-per-watt across a wide range of imaging and computer vision workloads, including latency-critical applications with millisecond-level requirements [20]. The product family encompasses *Myriad*, *Myriad 2*, and the latest generation *Myriad X*, all positioned as dedicated VPUs for efficient computer vision processing [110].

Intel provides the *OpenVINO Toolkit* [47], an open-source software platform designed to optimize machine learning models and accelerate inference across various Intel hardware platforms, including Movidius Myriad engines. OpenVINO (*Open Visual Inference and Neural Network Optimization*) converts NN models trained in cloud environments using popular frameworks such as PyTorch, TensorFlow, or ONNX into the OpenVINO Intermediate Representations (IR) format, enabling deployment at the edge. The toolkit achieves significant model size reduction through layer fusion strategies that streamline the computational graph while applying weight and activation function precision optimization techniques.

Figure 2.4 provides a high-level overview of the internal architecture. These devices are heterogeneous SoCs integrating two general-purpose LEON4 processors implementing the 32-bit RISC SPARCv8 architecture, which manage I/O peripherals and control media interfaces respectively. The primary processing cores are the Streaming Hybrid Architecture Vector Enginess (SHAVEs)—128-bit Very Long Instruction Word (VLIW) processors supporting Single Instruction Multiple Data (SIMD) instructions on multiple data streams—which operate under high-level control from the LEON processors. Low-power processing accelerators complement this computational fabric, including the *Streaming Image Processing Pipeline*

filter and the dedicated Neural Compute Engine [46].

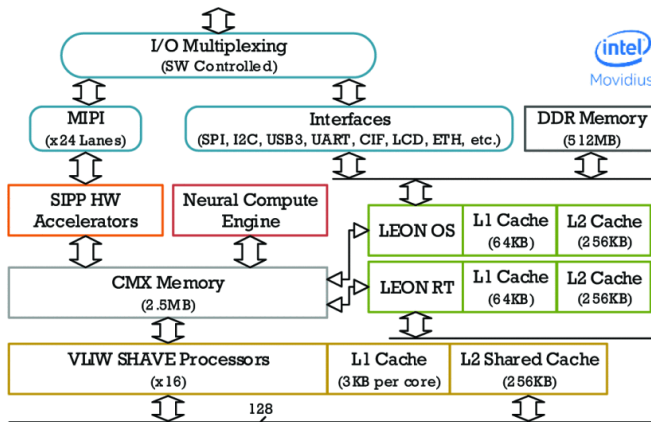


Figure 2.4. Myriad X VPU architecture

The Myriad X chip is manufactured using a 16 nm FinFET process and operates at 700 MHz [171, 110], providing 512 MB LPDDR4 DRAM and 2.5 MB Connection Matrix (CMX) memory. The CMX serves as scratchpad memory for the SHAVEs, with each core having preferential access to a dedicated memory slice. The device supports mixed-precision arithmetic with native FP16 support and fixed-point 16-bit representations, enabling developers to balance accuracy and performance requirements [112]. Overall, the platform delivers total performance exceeding 4 TOPS, with the Neural Compute Engine alone achieving over 1 TOPS for dedicated neural network inference workloads [46].

The Intel Movidius Neural Compute Stick 2 (NCS 2) (Figure 2.5) is a deep learning inference development kit based on the Myriad X, provided in an economical USB form factor that directly connects to PC or embedded device USB interfaces, augmenting host systems with neural network acceleration capabilities. The device features low power consumption and can be integrated into various embedded platforms capable of supplying USB power, including security cameras, autonomous drones, and industrial machine vision equipment. This convenient form factor facilitates rapid prototyping and deployment of advanced computer vision intelligence across diverse edge devices. The platform is programmable through the Myriad Development Kit (MDK), which provides comprehensive development tools,

frameworks, and APIs for implementing custom vision, imaging, and deep neural network workloads [46].



Figure 2.5. Intel Neural Compute Stick 2

2.3 Google Coral Edge TPU

In 2016, *Google* introduced the first generation of **TPUs** [110], dedicated processors designed for training complex deep learning algorithms in cloud environments, optimized to accelerate data processing by parallelizing operations similarly to **GPUs**. A single Cloud **TPU** V3 delivers 90 Tera Floating-Point Operations per Second (**TFLOPS**) with 32 GB of high-bandwidth memory [110]. However, the high power consumption of cloud **TPUs** limited their applicability for edge deployment.

To address this limitation, Google introduced the Edge **TPU** in 2019 [228], a low-power machine learning accelerator coprocessor specifically designed for edge devices. Built on a systolic array architecture enabling highly efficient matrix multiplication at massive scale, the device constrains all operations to 8-bit integer arithmetic, significantly increasing both performance and energy efficiency [83]. The platform performs 4 **TOPS** while consuming only 2 W, achieving an efficiency of 2 **TOPS** per watt [74].

Google offers the Edge **TPU** through various hardware form factors under the *Coral* brand [202]. The Coral Accelerator Module is a solderable component containing the Edge **TPU** chip, available with different connectors including the Coral USB Accelerator (Figure 2.6) and the Coral Mini PCIe Accelerator. For development and prototyping, Google provides the Coral Dev Board and Coral Dev Board Mini, which serve as host platforms for running and testing machine learning models.



Figure 2.6. Google Coral USB Accelerator

Unlike NVIDIA and Intel edge platforms, the Edge TPU is limited to running models based on the TensorFlow ecosystem. Executing models requires the TensorFlow Lite framework in conjunction with specialized compilation tools. The Edge TPU Compiler, a standalone tool provided separately from the TensorFlow Lite library, translates models by mapping each operator to a corresponding hardware-optimized operation on the device [116]. When an operator is incompatible with the TPU, it is automatically remapped to execute on the CPU instead. This implementation leverages the TensorFlow Lite delegate infrastructure to partition the computational graph, routing compatible operations to the TPU while offloading unsupported operations to the host processor [196].

2.4 FPGA-based Acceleration

As introduced in Chapter 1, FPGAs are reconfigurable integrated circuits built upon a programmable fabric of Configurable Logic Blocks (CLBs), Look Up Tables (LUTs), flip-flops, and specialized resources including DSPs and BRAM. Configuration bitstreams define the logical behavior and signal routing, enabling the implementation of application-specific datapaths. While this architectural flexibility has established FPGAs as reliable computational cores for spaceborne systems, their application to AI acceleration presents distinct challenges compared to GPU-based platforms.

GPUs natively support full-precision floating-point arithmetic, whereas FPGAs are inherently optimized for fixed-point and integer operations, where they achieve significantly higher throughput and energy efficiency. Floating-point computations on FPGAs, by contrast, incur substantial

resource overhead and performance penalties, making full-precision NN implementations impractical for most deployment scenarios.

Consequently, a principal optimization technique for FPGA-based deep learning deployment involves *quantization*, whereby the numerical precision of both network weights and activation functions is systematically reduced from floating-point to fixed-point or integer representations. This approach confers a dual advantage: diminished computational complexity alongside reduced on-chip memory footprint [94]. Extensive research has demonstrated that quantized models can achieve inference accuracy comparable to their full-precision counterparts without significant performance degradation [94, 54, 51, 86].

To accelerate CNN inference, FPGAs exploit the intrinsic parallelism of DNN architectures by instantiating multiple Processing Elements (PEs) within the processing fabric. Operating concurrently, these PEs increase effective throughput by distributing the computational load across the array. Each PE integrates Multiply-Accumulate (MAC) units and dedicated local accumulation Random Access Memory (RAM), expediting the core operations of convolutional layers. This parallelism enables simultaneous execution of independent operations across output channels, spatial positions, and—when resources permit—multiple input samples or data tiles. Figure 2.7 illustrates a basic neuron implementation supporting these parallel execution modes.

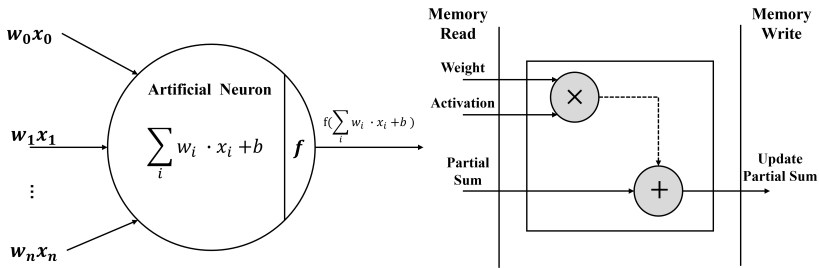


Figure 2.7. MAC operation and partial-sum update in a neuron

In convolutional layers, the dominant workload consists of repeated multiply-accumulate operations between activations and weights, followed by accumulation of partial sums. In practice, performance and energy

efficiency are often bounded not only by arithmetic throughput but also by data movement: a single **MAC** typically entails multiple memory transactions—two reads (activation and weight) plus a read-modify-write of the running partial sum (effectively one read and one write)—so memory hierarchy design and on-chip buffering are critical to sustain high utilization of the **PEs**.

2.4.1 Deployment Strategies

Deployment strategies for **NN** models on **FPGA** platforms vary considerably depending on model complexity and resource constraints. For lightweight architectures, fully on-chip dataflow implementations—wherein each network layer is mapped to a dedicated processing core with its own on-chip memory—can obviate off-chip memory transactions, thereby substantially reducing power consumption while enhancing inference throughput [58, 59]. In this paradigm, computational operations flow continuously through a spatial pipeline of layer-specific cores, with intermediate activations stored in local buffers rather than external memory, as illustrated in Figure 2.8.

However, dataflow architectures become increasingly constrained when targeting larger network topologies. Since each layer requires dedicated logic and memory resources instantiated on the **FPGA** fabric, even moderately-sized **DNN** models frequently exceed the available capacity, particularly for deeper and more sophisticated architectures [79]. The inherent limitations in logic fabric and Block RAM capacity fundamentally restrict which **CNN** architectures can be successfully realized using this implementation paradigm. Consequently, while dataflow implementations demonstrate superior performance in terms of speed and energy efficiency for models that fit on-chip, their applicability remains limited to compact network topologies.

Different automated toolchains are available today. Frameworks such as hls4ml [66] and FINN [9] map each **NN** layer to dedicated processing elements, automatically performing the translation of pre-trained networks—specified by the model’s architecture, weights, and biases—into hardware acceleration specifications that can be synthesized using High-Level Synthesis (**HLS**) tools.

For larger neural networks, overlay architectures with external memory

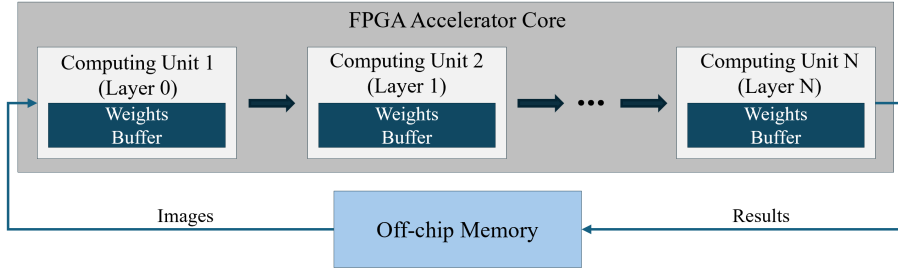


Figure 2.8. Dataflow architecture for on-chip FPGA deployment

are employed, as depicted in Figures 2.9 and 2.10. In this paradigm, designs frequently adopt external memory to store model parameters and intermediate computations. Overlay architectures leverage arrays of PEs to execute the operations required by common NN layers. During inference, model weights and biases are stored in off-chip non-volatile memory and loaded into an on-chip weight buffer as needed. Input and output data for the PE array are similarly cached in dedicated on-chip memory buffers. A main controller coordinates the activities of the PEs, managing data flow across the buffers and executing layer-specific instructions to optimize inference efficiency. By multiplexing various operations across layers, this architecture enables the deployment of deeper models on FPGA-based platforms [165].

The principal advantage of employing an overlay-based approach lies in the capability to utilize a single accelerator core for executing diverse neural network models without depth constraints, achieved through the multiplexing of inference operations. Such architectures provide substantial flexibility, enabling the execution of different models without necessitating modifications to the underlying design, thereby significantly reducing reconfiguration overhead and development time. The primary limitation pertains to the compatibility between the operations required by individual network layers and those natively supported and accelerated by the selected core.

Several FPGA vendors currently offer open-source Intellectual Property (IP) cores specifically designed for neural network acceleration, which are readily deployable and provide immediate access to optimized inference

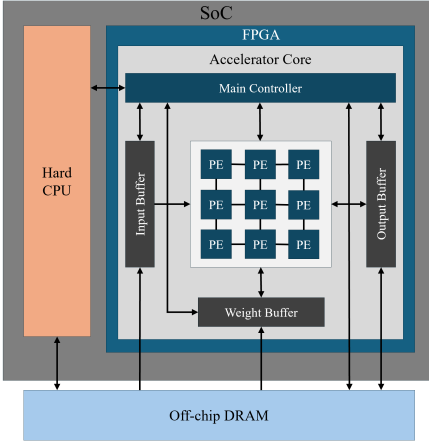


Figure 2.9. Overlay architecture on FPGA SoC

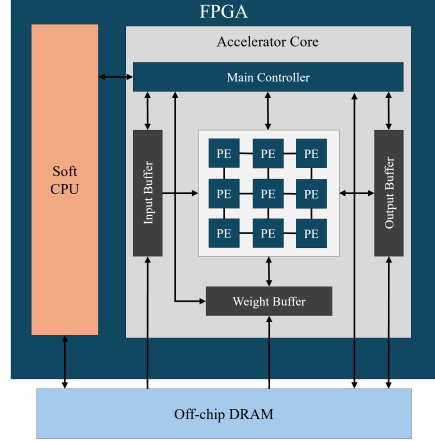


Figure 2.10. Overlay architecture on pure FPGA

capabilities. These pre-designed accelerators offer a compelling alternative to custom-built solutions, delivering both enhanced flexibility and reduced development costs while maintaining competitive performance for a broad range of CNN architectures. This combination of design reusability and vendor-supported IP cores renders overlay-based accelerators particularly attractive for applications requiring support for multiple NN models or frequent model updates in deployed systems.

Two representative examples of this approach are CoreVBX [142] and the DPU [10], provided by Microchip and AMD Xilinx, respectively. These computing units are typically not stand-alone IP cores, but rather co-processors controlled by a main processor. This can be either a soft-core processor synthesized into the programmable logic (e.g., soft RISC-V on Microchip PolarFire FPGAs) or a hard-core processor embedded within the SoC, as in the case of PolarFire SoC or AMD Xilinx Zynq UltraScale+ MPSoC.

However, the enhanced flexibility of the overlay approach, which enables general-purpose AI acceleration for diverse neural network models, is achieved at the cost of increased power consumption and higher inference latency compared to application-specific accelerators developed through custom dataflow architectures. This represents a fundamental trade-off:

overlay architectures prioritize versatility and ease of deployment, while dataflow implementations are optimized for maximum performance and energy efficiency for specific network topologies.

The FINN framework, developed by Xilinx Research Labs, facilitates the exploration of Quantized Neural Network (QNN) inference on Xilinx FPGAs through dataflow-style architectures. Beyond generating fully pipelined dataflow accelerators, FINN also supports the creation of hybrid custom accelerators capable of executing deeper neural networks through a time-multiplexed approach. These hybrid implementations leverage off-chip memory resources and dedicated hardware modules for operation multiplexing, enabling FINN to scale to larger networks while balancing performance, resource utilization, and energy efficiency. In [79], a comparative study of FINN and the DPU on Zynq UltraScale+ MPSoC reveals that the FINN-based accelerator significantly outperforms the DPU implementation despite utilizing time-multiplexing, achieving $8.4\times$ lower latency, $3.0\times$ higher throughput, and $3.3\times$ better power efficiency.

2.4.2 Network Optimization Techniques

The efficient deployment of deep neural networks on FPGA platforms demands advanced optimization strategies to address constraints in memory, computational resources, and power consumption. Among the most effective techniques are quantization and pruning, which act synergistically to reduce model complexity while preserving acceptable levels of accuracy.

Quantization converts floating-point representations into lower-precision formats, typically fixed-point integers, thereby reducing memory usage and computational cost. A comprehensive survey presented in [17] identifies two principal quantization strategies: *linear quantization*, which maps floating-point values to the nearest fixed-point representation, and *dynamic quantization* [115], which employs lookup tables and adaptive scaling to encode values more efficiently. In the dynamic approach, each layer is assigned a specific bit width that reflects the limited and heterogeneous dynamic range of its parameters [51, 54]. To mitigate the loss in accuracy introduced by reduced precision, a calibration phase is performed using a representative subset of the training data, enabling fine-tuning of the scale factors for optimal alignment with the INT8 value range [166].

In contrast, pruning systematically discards less-informative elements

of a trained network—spanning single weights, entire filters, and occasionally whole layers—thereby reducing model complexity, as illustrated in Figure 2.11 [39]. The rationale is that over-parameterized models often contain redundancies that can be eliminated without compromising predictive accuracy. Pruning strategies are typically divided into *unstructured pruning*—where isolated weights are removed—and *structured pruning*, which targets higher-level structures like channels or filters. The standard pruning workflow involves training a baseline model, ranking and eliminating low-importance components based on criteria such as weight magnitude or sensitivity, and then fine-tuning the pruned model to recover lost accuracy.

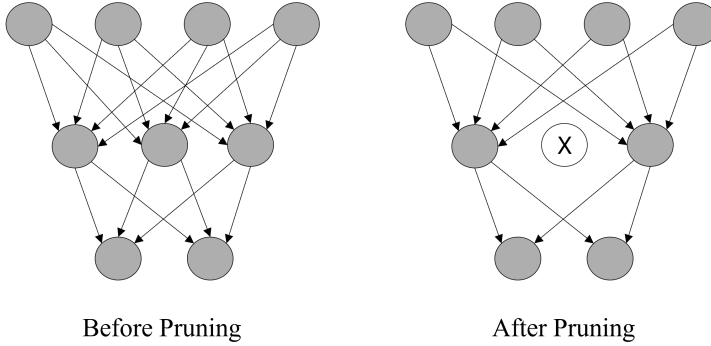


Figure 2.11. Neural network pruning illustration

Applied jointly, these methods sustain real-time inference in resource-limited settings (e.g., satellite-borne EO payloads) by compressing models with minimal accuracy impact, thereby lowering latency and energy and enabling embedded deployment.

2.4.3 Microchip CoreVectorBlox

The CoreVBX [138] is a memory-mapped core featuring an overlay architecture, designed to support diverse NNs without necessitating the reconfiguration of programmable logic resources [51]. Currently, Microchip has provided two different releases of this IP core: CoreVBX version 1 and the latest version 2. We primarily focus on the first release. Chapter 5 provides the main differences between these two releases, highlighting one

of the principal advantages of using **FPGAs**: reconfigurability.

As shown in Table 2.3, the **CoreVBX** is available in three configurations – V250, V500, and V1000 – offering progressively higher performance; the V1000 provides the greatest parallelism and fastest inference, at the cost of increased logic-resource usage and power consumption.

Table 2.3. CoreVectorBlox configurations [138]

Configuration	Resource					Utilization		Performance (MHz)	Peak GOPS
	LUT4	DFF	Math Blocks	uSRAM	LSRAM	Device	LUT4 %		
V250	27 244	27 197	90	177	47	MPF100T-1	25%	154	79
V500	46 622	48 546	176	315	84	MPF100T-1	43%	143	146
V1000	61 847	69 824	304	539	148	MPF200T-1	32%	136	279

The internal architecture of **CoreVBX**, depicted in Figure 2.12, centers on a specialized Matrix Processor (**MXP**) optimized for high-throughput matrix and vector kernels, thereby accelerating computations across diverse neural network layers. A two-dimensional array of **PEs** further parallelizes convolutional workloads, yielding performance that surpasses **MXP**-only configurations [51].

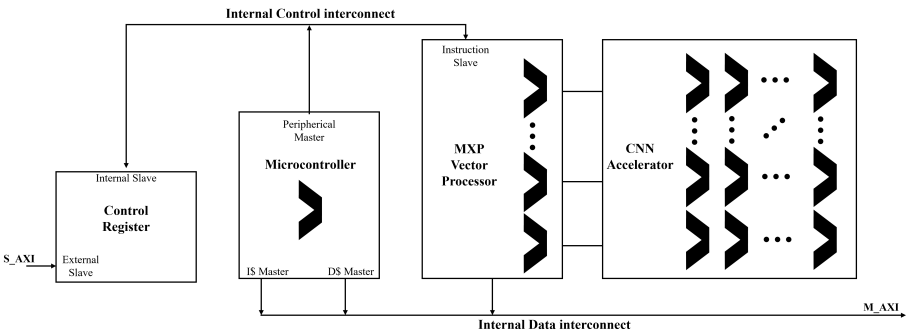


Figure 2.12. CoreVectorBlox’s Internal Architecture [138]

The **CoreVBX** operates as a co-processor capable of sequentially executing highly optimized instructions, known as *network Binary Large Objects (BLOBs)*, tailored to neural network inference tasks. They can be compiled by the user via the *Microchip CoreVectorBlox SDK*. Each instruction, representing a layer within the **AI** model, containing information

about the specific layer types in the network as well as weights and buffer space for activations. Each network has its own BLOB; multiple BLOBs can be present in memory at the same time and each time the CoreVBX is started, it can target a different BLOB. A soft RISC-V microcontroller, embedded within its architecture, manages read/write operations to control registers, decodes network-specific BLOBs, and coordinates instruction flow to both the vector processor and CNN accelerator, ensuring efficient processing of each NN layer. The core caches instructions and data to reduce traffic on the memory interface, optimizing performance. It uses an internal scratchpad as working memory and a variable-width Direct Memory Access (DMA) controller for transferring data between the scratchpad and external memory. Table 2.4 provides inference speed performance, in terms of CNN throughput and Frame Per Second (FPS), evaluating ResNet-50 inference with varying array PE sizes and scratchpad memory configurations.

Table 2.4. Processor size configuration details

Size Configuration	Vector Processor Width	Vector Scratchpad	CNN Accelerator Array Size	Peak CNN Throughput	ResNet-50 Performance
V250	128-bit	64 kB	16×16	79 GOPs	4.2 FPS
V500	256-bit	128 kB	16×32	146 GOPs	7.8 FPS
V1000	256-bit	256 kB	32×32	279 GOPs	13.6 FPS

Microchip provides a pre-compiled *firmware BLOB*, common to all core configurations, which contains the routines for *network BLOB* parsing and data memory management. During boot-up, this firmware is loaded into the working memory, from which it is executed.

The CoreVBX can be deployed within the Programmable Logic (PL) of PolarFire FPGA and SoC devices, where it operates in tandem with either a soft or hard *RISC-V CPU core*. This configuration enables the CPU to manage critical functions, such as handling interrupts generated by the CoreVBX and facilitating data transfers between the CoreVBX and peripheral devices, ensuring seamless integration within complex processing environments. The IP uses the Advanced eXtensible Interface (AXI) [23] bus to exchange data with external memory via AXI4, and state control via AXI4-Lite to external logic (e.g., *RISC-V* soft processor) or with the

Processing Systems (PS) (i.e., RISC-V hard processor). A configurable-width (64-bit to 256-bit) AXI4 memory master enables efficient data access, facilitating the transfer of data and control signals between PL and PS, thereby accelerating inference.

Microchip CoreVectorBlox SDK

The Microchip VectorBlox Software Development Kit [139] provides a robust toolset designed to convert neural networks, originally developed in frameworks such as *TensorFlow* or *PyTorch*, into a BLOB. As illustrated in Figure 2.13, the compilation flow involves several key stages to transform a model into a format compatible with the CoreVBX hardware.

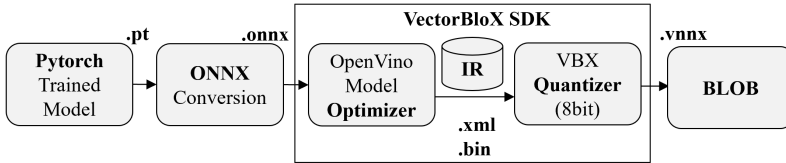


Figure 2.13. VectorBlox SDK flow

To demonstrate this process, *SR-YOLOv5s*—a novel object detector model for satellite imagery thoroughly described in Chapter 4—is compiled and deployed on the platform. The model, initially defined in PyTorch, must first be converted to the Open Neural Network Exchange (ONNX) format, as required by the toolchain. Release 1 is based on the *OpenVINO* framework. The optimizer converts the model into an IR, removing training-specific layers—such as batch normalization and dropout—to prepare the network for inference.

Subsequently, a dynamic Fixed-Point (FXP) quantization technique is applied [115], enabling a more memory-efficient representation without incurring significant degradation in model accuracy [51, 54]. To map floating-point (FP32) values to the INT8 format, a scaling factor is introduced to align the tensor’s dynamic range with the representable integer range of $[-128, 127]$. This is followed by a calibration step using a representative subset of the data, which fine-tunes the scale parameters to

preserve the model’s predictive performance under reduced precision constraints [166].

It is important to note that the toolchain does not currently support optimization techniques such as pruning within its compilation flow. Furthermore, execution on the [CoreVBX IP](#) core strictly requires INT8 quantization, as the hardware lacks support for floating-point or higher-bitwidth inference. Once quantized, the model is compiled and the corresponding [BLOB](#) is generated for deployment on the target hardware.

Microchip PolarFire SoC Video Kit

The *PolarFire FPGA* [91] is Microchip’s mid-range, non-volatile [FPGA](#) family built on a *28 nm flash process*, delivering industry-leading power efficiency—up to 50% lower total power than competing devices and approximately one-tenth the static power. The architecture integrates low-power 12.7 Gb/s SerDes, hardened PCIe, DDR physical layers, and security/crypto subsystems, targeting bandwidth- and power-constrained applications requiring secure [I/O](#) and modest transceiver rates at significantly reduced thermal budgets. The family spans densities from approximately 50k to 500k Logic Element ([LE](#)) (i.e., the basic programmable logic unit, typically a 4-input [LUT](#) (LUT4) plus a flip-flop). Figure 2.14 illustrates the main architectural blocks.

The *PolarFire SoC* [92] combines the low-power [FPGA](#) fabric with a hardened five-core 64-bit *RISC-V processor subsystem*—four Linux-capable application cores plus one monitor/real-time core—connected via coherent interconnect and deterministic L2 memory subsystem. This heterogeneous architecture enables mixed Linux and real-time workloads under tight power and thermal constraints. The platform offers mid-range logic densities (up to 460 kLE), defense-grade security with secure boot capabilities, integrated DDR/LPDDR controllers and PHYs, PCIe Gen2 endpoints/root ports, and low-power multi-gigabit SerDes. As shown in Figure 2.15, the architecture features dedicated [DSP](#) blocks and embedded SRAMs within the FPGA fabric, while Single-Event Upset ([SEU](#))-tolerant configuration enhances robustness for embedded and edge applications. As discussed in Chapter 3, flash-based technology offers greater radiation tolerance than Static Random Access Memory ([SRAM](#))-based alternatives [51], thereby improving resilience to configuration upsets.

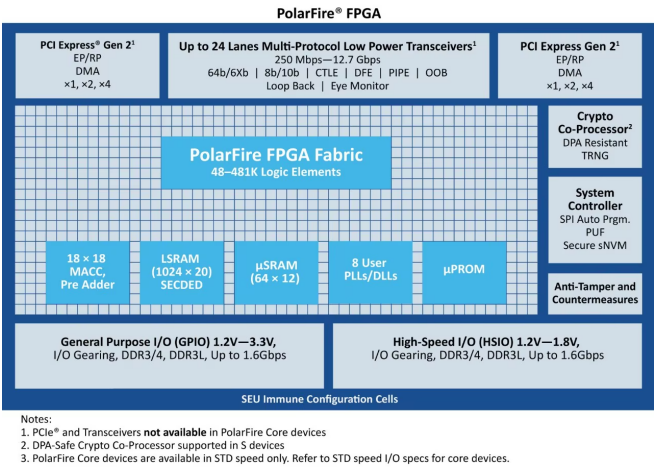


Figure 2.14. Microchip PolarFire FPGA architecture [91]

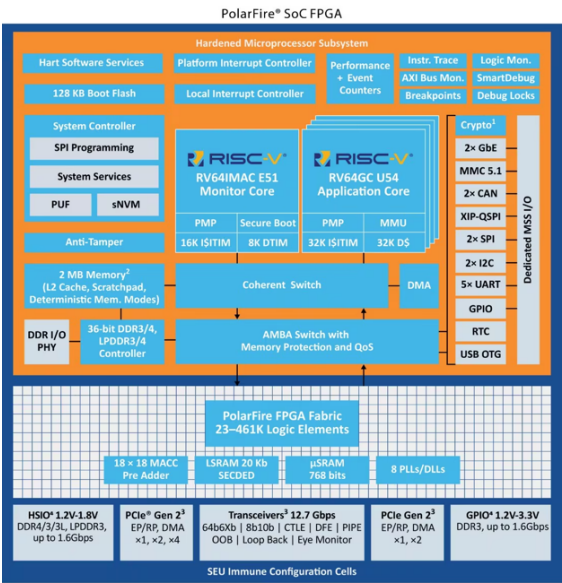


Figure 2.15. Microchip PolarFire SoC architecture [92]

In this dissertation, different CNNs are deployed on the PolarFire device to assess CoreVBX performance capabilities. For experimental evaluation, we adopt the *Microchip PolarFire SoC Video Kit* [93] (Figure 2.16), a development board designed for embedded vision applications integrating the *MPFS460T PolarFire FPGA* fabric.

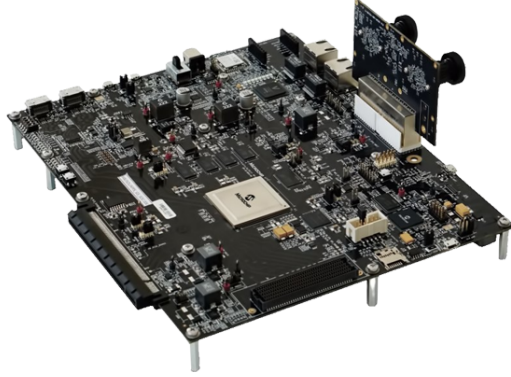


Figure 2.16. Microchip PolarFire SoC Video Kit

2.4.4 AMD Xilinx DPU

The AMD Xilinx DPUs [10] are programmable engines optimized for CNNs inference. The family encompasses several variants targeting specific platforms including *Alveo* data center cards, *Zynq UltraScale+ MPSoCs*, *Versal ACAP* devices, and *Kria System on Modules*. This thesis focuses on the *DPUCZDX8G* variant, implemented in the PL of Zynq UltraScale+ MPSoC devices with direct PS connections.

As an overlay accelerator, the DPU requires pre-compiled instructions and accessible memory locations for input, intermediate, and output data. The Application Processing Unit (APU)—a quad-core *Cortex-A53* processor running a Linux-based Operating System (OS) such as *PetaLinux*—coordinates data transfers and services DPU interrupts. The DPU fetches optimized instructions from system memory to perform inference.

As depicted in Figure 2.17, the DPU consists of multiple specialized engines. The convolutional cores comprise Matrix of Processing Engines

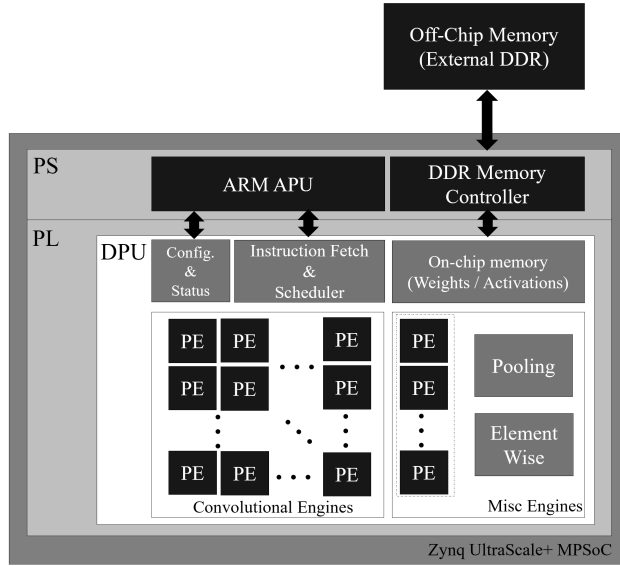


Figure 2.17. DPU architecture overview

(MPEs) constructed from DSPs, LUTs, BRAM, and optionally UltraRAM to accelerate convolutions. The MISC (miscellaneous) processor handles non-convolutional operations including element-wise functions, activation functions (e.g. ReLU variants), pooling (average, max), and classification primitives (*argmax*, *softmax*). The MISC engine operates under DPU scheduler control, forming a heterogeneous compute fabric [11, 10]. The architecture also includes an instruction fetch unit and a global memory pool module.

AMD Xilinx Vitis AI Framework

Vitis AI [11] is AMD’s end-to-end software stack that simplifies model deployment on Xilinx Zynq UltraScale+ and Versal devices [4] by abstracting hardware-specific complexities, integrating with TensorFlow and PyTorch, and enabling algorithm developers to deploy models without requiring direct knowledge of Hardware Description Languages (HDLs) or FPGA programming.

The stack provides a comprehensive toolchain spanning model preparation, optimization, and deployment. The *AI Model Zoo* offers pre-optimized reference networks for classification, detection, and segmentation, serving as deployment baselines. Model optimization is supported through two complementary tools: the *AI Optimizer*, which applies structured pruning and graph optimizations to reduce computation with minimal accuracy impact, and the *AI Quantizer*, which reduces precision from FP32 to INT8, enabling higher throughput and lower power consumption without full retraining. The *AI Compiler* translates quantized networks into [DPU](#) instructions, partitioning operators across heterogeneous engines and scheduling tensor operations within memory constraints.

Runtime execution and performance analysis are handled by the remaining components. The *AI Library (VART)* provides C/C++ APIs managing [I/O](#), tensor formatting, multi-threading, and multi-engine execution, while the *AI Profiler* measures latency, throughput, and utilization for bottleneck identification. At the hardware level, the configurable [DPU](#) acceleration [IP](#) is available in various architectures (B2304, B4096) trading off parallelism, resource utilization, and performance.

The deployment workflow begins with a trained model, optionally pruned, then quantized to INT8 and compiled into a [DPU](#) binary. VART orchestrates preprocessing, inference, and post-processing, while optimizing layer tiling, buffer management, and operator placement to maximize throughput within device constraints.

Edge AI in Space

Deploying onboard AI algorithms requires meeting the computational needs of modern DL architectures within the stringent resource constraints of spaceborne platforms. To ground these challenges, Table 3.1 summarizes the compute and memory footprints of several widely used CNN models for computer-vision tasks.

Table 3.1. Computational and memory requirements of representative CNN architectures

Model	Input Size	Params. (M)	FLOPs (G)	Mem. (MB)	Ref.
VGG-16	$224 \times 224 \times 3$	138.4	15.5	528	[201]
ResNet-50	$224 \times 224 \times 3$	25.6	4.1	98	[80]
MobileNet V1	$224 \times 224 \times 3$	4.2	0.57	16	[85]
YOLOv3	$416 \times 416 \times 3$	61.9	65.9	236	[179]
YOLOv5s	$416 \times 416 \times 3$	7.0	7.3	27	[99]

Note—Memory (Mem.) footprint assumes FP32 precision for weights and activations.

As shown, models such as VGG-16 [201] and YOLOv3 [179] demand tens of billions of Floating-Point Operations (FLOPs) per inference and require hundreds of megabytes of memory for weight storage. Even relatively efficient architectures like MobileNet V1 [85], ResNet-50 [80], and YOLOv5s [99], specifically designed for resource-constrained environments or edge deployment, still require substantial computational resources—ranging from 0.57 to 7.3 Giga Floating-Point Operations (GFLOPs) per inference and 16 to 98 MB of memory footprint.

As a result, the computational demands of these DL workloads often exceed the capabilities of conventional radiation-hardened space processors, which are built on mature semiconductor nodes to prioritize reliability in harsh environments. To achieve the required radiation tolerance, they are manufactured using relatively coarse process nodes (45–250 nm), which inherently limit parallelism and energy efficiency when compared to modern commercial processors built on sub-10 nm nodes [25, 128]. While robust against radiation effects, these processors offer limited performance and lower clock speeds compared to their commercial counterparts.

In [124], the authors present performance trends comparing radiation-hardened processors with COTS devices. As summarized in Figure 3.1, processors commonly flown on satellites lag by roughly three to four orders of magnitude. For example, *RAD5500*-class parts deliver only about 0.9 Giga Floating-Point Operations per Seconds (GFLOPs), whereas an *NVIDIA A100*—representative of modern COTS accelerators—reaches approximately 156 TFLOPs.

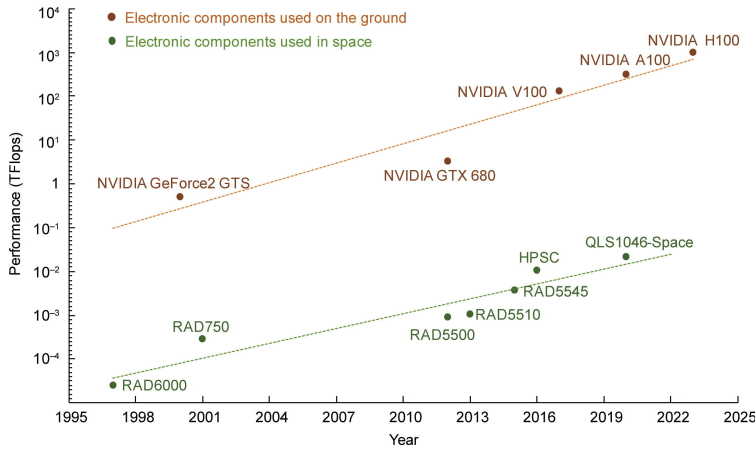


Figure 3.1. Comparative performance of radiation-hardened processors and COTS GPUs

Table 3.2 presents a comparative overview of representative space-grade general-purpose processors currently deployed in spacecraft Onboard Computers (OBCs). As shown, processors such as the RAD750, RAD5545, and GR740 operate at clock speeds ranging from 110 to 466 MHz and

deliver performance between 400 and 5,592 Dhrystone Million Instructions Per Second (DMIPS).

Table 3.2. Radiation-tolerant processors commonly used in spaceborne OBC

Processor	Power (W)	Perf. (DMIPS)	Speed (MHz)	Peak GFLOPS	Mem. BW (GB/s)	Core Type	Node (nm)	Ref.
RAD750	~10	~400	200	~0.4	~0.2	Single-core PowerPC 750	150–250	[16, 18, 127]
RAD5545	~20	5,592	up to 466	13.5	4.2	Quad-core QorIQ P5040	45	[172, 22, 128]
GR740	~2	~250	250	~0.5	~0.2	Quad-core LEON4	65	[41, 95, 128]

Note—Process nodes and memory bandwidth values are approximations based on available datasheets.

Table 3.2 further clarifies the disparity between the computational requirements of modern CNN architectures and the capabilities of radiation-hardened processors, highlighting a critical gap for autonomous onboard AI. For example, executing a single ResNet-50 inference (4.1 GFLOPs) on a RAD750 processor (peak ~0.2 GFLOPS) would take roughly 20 s, rendering real-time operation infeasible for time-critical tasks such as hazard avoidance or autonomous navigation.

In [128], the authors conduct a comprehensive comparative analysis of space-grade processors using device metrics, demonstrating that emerging multicore CPUs such as the GR740 and RAD5545 achieve substantially higher computational density and memory bandwidth compared to heritage single-core processors like the RAD750, primarily due to higher operating frequencies and modern multicore architectures. However, even considering the more capable RAD5545, executing YOLOv5s for object detection would require at least 2 seconds per frame, with an energy consumption on the order of several tens of joules per frame.

Beyond performance constraints, cost represents an additional barrier. Radiation-hardened integrated circuits use dedicated processes and screening to reach up to 1 Mrad(Si) Total Ionizing Dose (TID) [208]; however, specialized fabrication lines, stringent qualification, and low production volumes drive unit prices typically 10–100× higher than commercial parts. Consequently, true rad-hard devices are economically viable mainly for

missions with exceptional radiation requirements and solid institutional funding, while many LEO or short-duration missions instead favor COTS devices with mitigation.

Taken together, these economic pressures and performance limitations motivate the need for alternative onboard AI acceleration strategies. Historically, the adoption of COTS hardware accelerators—such as embedded GPUs, TPUs, and VPUs—has been limited in space systems due to their inherent susceptibility to radiation-induced faults and the stringent qualification protocols required for space deployment. However, the growing computational demands of onboard AI applications, combined with the increasing maturity of COTS AI hardware, have driven the exploration of heterogeneous processing architectures that incorporate such co-processors within the space domain [108].

These platforms, though originally developed for terrestrial edge and mobile applications, are demonstrating operational reliability in short-duration missions and LEO environments, accumulating initial flight heritage. In this section, we present a review of space missions that leverage edge-AI processing—spanning EO, in-orbit operations, and deep-space exploration—with the overarching aim of enhancing onboard autonomy and reducing dependence on ground control. The review also provides a comparative analysis between COTS accelerators and space-grade FPGA devices, highlighting their respective strengths and limitations in terms of computational throughput and latency, power efficiency, radiation tolerance, and reconfigurability.

3.1 Edge AI Heritage

As discussed in Chapter 2, the emergence of advanced processors tailored for embedded systems—including embedded GPUs and ASICs-based devices—has made deploying AI solutions at the edge a practical reality. This technological shift has also influenced the space sector, where there is a growing interest in exploring heterogeneous co-processing architectures that leverage COTS components originally designed for terrestrial edge applications. As a result, several space-oriented products have been introduced to meet the computational demands of modern missions.

For instance, *Aitech's S-A1760 Venus* [7], which integrates the Jetson TX2i GPU, is a rugged AI processor developed for short-duration space

missions. Following a similar approach, *Ubotica's CogniSat* offers CubeSat-compatible boards equipped with the Myriad 2 and Myriad X VPUs [212], enabling onboard execution of AI and computer vision algorithms directly on the payload.

Building on these hardware innovations, a number of recent EO missions have successfully demonstrated the feasibility of deploying DL models directly onboard satellites using COTS accelerators. These missions underscore the growing maturity of onboard AI processing and its applicability across diverse operational scenarios. Notably, *PhiSat-2*, developed by ESA as a follow-up to *PhiSat-1*, expanded onboard capabilities to include vessel detection, cloud masking, and image compression, all powered by the *Myriad 2 VPU* [135]. Likewise, the *CogniSat-6* mission by *Open Cosmos* employed the *Myriad X-based CogniSAT-XE2* platform to perform real-time flood detection and radio frequency signal classification onboard a 6U CubeSat [185]. The *MANTIS* 12U mission further validated onboard cloud detection using a CNN, achieving 90% accuracy, compared to 92% achieved in ground-based processing [144].

GPU-based inference has also been demonstrated in recent missions. The *FOREST-2* CubeSat employed an embedded *NVIDIA Jetson Xavier NX GPU* to perform onboard wildfire detection [192], while the *NewSat-27* microsatellite developed by *Satellogic* utilized a *Jetson TX2i* module for real-time ship detection and onboard image preprocessing [167]. In parallel, NASA's Goddard Space Flight Center (GSFC) has developed a CubeSat-class AI co-processor card based on a TPU, known as the SpaceCube Low-power Edge AI Resilient Node (SC-LEARN) [72]. This system has recently been launched and integrated onboard the International Space Station as part of the Space Test Program-Houston 9 (STP-H9) experiment [133].

To further support flexible AI workloads in orbit, hybrid platforms combining general-purpose processors with dedicated accelerators have also emerged. One example is the distributed computing system developed by *Unibap*, deployed on *D-Orbit's ION-MK02 Dauntless David* mission, which integrates a quad-core x86 CPU, a *Microsemi SmartFusion2 FPGA*, and an *Intel Myriad X VPU* [147].

These examples collectively underscore a growing trend toward heterogeneous computing architectures that leverage COTS co-processors—such as embedded GPUs and VPUs—to enable onboard AI capabilities. These plug-

and-play solutions offer significant advantages over traditional [FPGA](#)-based designs, including simplified development workflows, reduced integration complexity, and accelerated deployment timelines, thereby lowering the entry barrier for space applications.

Nonetheless, their operational suitability remains largely confined to short-duration missions in [LEO](#), where radiation levels are comparatively moderate and system reliability constraints are less stringent [175]. As missions extend in duration and venture into harsher orbital regimes, the space environment introduces a range of challenges—including extreme thermal conditions and elevated radiation exposure—that can severely impact the performance, reliability, and longevity of electronic components used for onboard [AI](#) processing [56]. These constraints necessitate a deeper understanding of radiation effects and mitigation strategies, which are addressed in the following sections.

3.2 Space Environment

The space environment imposes extreme conditions on electronic systems, requiring careful component selection and qualification. Two primary factors dominate the design of spaceborne electronics: thermal stress and radiation exposure [221].

Space systems experience extreme temperature variations that depend on mission profile and spacecraft configuration. Components must withstand temperatures ranging from cryogenic conditions in shadowed regions to elevated temperatures under direct solar illumination. During eclipse transitions, rapid thermal cycling accelerates aging mechanisms and can significantly reduce component lifetime and reliability [210]. While thermal management is critical, the radiation environment presents more fundamental challenges for electronic reliability in space missions.

The term radiation refers to the transfer of energy through space in the form of waves or particles. It can be broadly classified into two categories: *electromagnetic radiation* and *particulate radiation* [63].

High-energy electromagnetic radiation, including X-rays from astrophysical sources (galaxy clusters, black holes, stellar processes) and gamma-rays from radioactive decay and thermonuclear fusion, poses significant threats to space electronics. These energetic photons penetrate semiconductor devices

and generate electron-hole pairs through ionization [180]. The resulting radiation-induced currents can alter timing circuit states, corrupt memory contents, trigger spurious logic signals, or cause permanent device damage. The deep penetration capability of gamma rays makes them particularly hazardous, as they induce charge accumulation throughout microelectronic structures, leading to progressive degradation and failure [162].

Particle radiation, as opposed to electromagnetic radiation composed of massless photons, consists of fast-moving subatomic particles with intrinsic mass, such as *protons*, *electrons*, and *nuclei of atoms*, including alpha particles. Figure 3.2 provides a schematic view of Earth’s magnetosphere showing how *Solar Energetic Particles* and *Galactic Cosmic Rays* interact with the geomagnetic field.

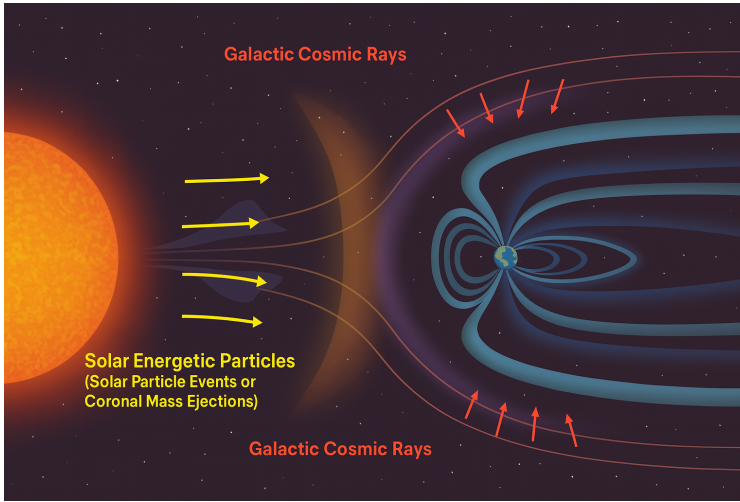


Figure 3.2. Space Radiation Sources [36]

In [160], the impact of space radiation on satellite systems is discussed in depth, providing a comprehensive review of the main radiation sources in the space environment.

Galactic Cosmic Rays are high-energy particles—predominantly protons ($\sim 85\%$) and alpha particles ($\sim 14\%$)—originating from outside the Solar System. With energies ranging from hundreds of MeV to several GeV, and some particles exceeding 10^{20} eV, their extreme penetrating power makes

effective shielding particularly challenging [64].

Solar Particle Events produce sporadic, intense fluxes of energetic particles—primarily protons and electrons—ejected during solar flares and coronal mass ejections. These events are characterized by rapid onset, short duration (hours to days), and highly variable intensity, posing significant risks to spacecraft electronics during solar maximum periods [190].

Trapped Radiation Belts, known as the Van Allen Belts (Figure 3.3), consist of energetic electrons and protons confined by Earth’s magnetic field in toroidal regions around the planet. The inner belt contains a relatively stable population of high-energy protons exceeding 10 MeV, while the outer belt contains primarily energetic electrons up to 10 MeV.

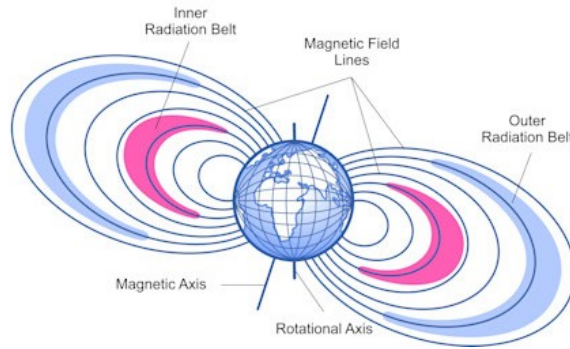


Figure 3.3. Van Allen Belts Representation [87]

The radiation dose, measured in rad(Si)—where 1 rad(Si) corresponds to 0.01 J/kg—quantifies the amount of energy absorbed by silicon due to radiation exposure. The severity of radiation effects varies dramatically with orbital parameters. Satellites in **LEO** benefit from substantial protection provided by Earth’s magnetic field, with typical dose rates of 100–1000 rad(Si)/year for low-inclination orbits ($<28^\circ$) and 1000–10,000 rad(Si)/year for high-inclination orbits [146]. In contrast, spacecraft in Medium Earth Orbit (**MEO**) and **GEO** experience doses approximately two orders of magnitude higher [224]. For reference, the International Space Station experiences doses below 3 Gy/year (300 rad(Si)/year) under worst-case conditions [77].

Beyond Earth’s protective magnetic field, the radiation environment

becomes significantly more severe, presenting major challenges for deep-space missions [204]. The radiation environment is also strongly influenced by transient solar weather phenomena, with major solar events capable of increasing instantaneous radiation flux by several orders of magnitude [98].

A critical feature of the trapped radiation environment is the *South Atlantic Anomaly (SAA)*, a region where the inner Van Allen belt dips to approximately 200 km altitude over the southern Atlantic Ocean off the coast of Brazil. Despite the generally benign radiation environment in LEO, the SAA accounts for the largest fraction of radiation exposure during LEO missions, with high-inclination orbits traversing this region up to six or seven times per day [160]. Within the orbit, radiation dose varies significantly by position: the SAA exhibits elevated concentrations of trapped protons that can produce upset rates exceeding those in other orbital regions by an order of magnitude [227, 160].

3.2.1 Radiation Effects on Electronics

Radiation effects on electronic devices originate from the interaction between energetic particles and matter. Two primary damage mechanisms can be distinguished: *ionization* and *displacement damage*.

Ionization occurs when radiation—such as X-rays, gamma rays, or charged particles—deposits sufficient energy to remove electrons from atoms, thereby generating electron–hole pairs within semiconductor materials. This ionization process generates cumulative effects (TID), which arise from the gradual accumulation of ionizing dose over time, causing gradual degradation of device parameters, such as threshold voltage shifts, increased leakage currents, and deterioration of insulating layers [42].

In contrast, predominantly non-ionizing energetic particles—such as neutrons, protons, and heavy ions—can transfer momentum to the atomic lattice, displacing atoms from their original sites and producing crystal defects, as illustrated in Figure 3.4. This process, known as *displacement damage* or *Non-Ionizing Energy Loss (NIEL)*, shortens the minority-carrier lifetime and degrades the performance of bipolar devices, ultimately leading to gain reduction and increased leakage currents in bipolar transistors [206, 160].

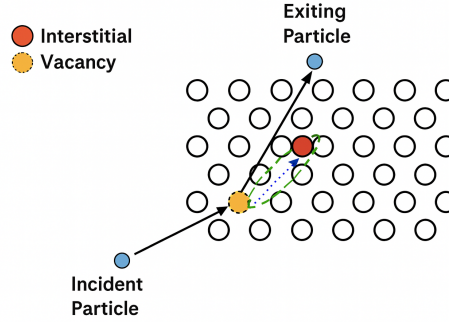


Figure 3.4. Displacement damage mechanism [160]

Total Ionizing Dose

Unlike transient effects, cumulative damage mechanisms develop progressively throughout the operational lifetime of electronic systems operating in radiation environments. Components subject to **TID** or *displacement damage* degrade gradually until reaching their tolerance threshold, at which point functional failure occurs.

A key advantage of cumulative effects is their predictability: for well-characterized components with known dose tolerance limits, the time to failure can be estimated. The operational lifetime t (in years) before **TID**-induced failure can be expressed as:

$$t \text{ (yrs)} = \frac{\text{TID}_{\text{threshold}}}{\text{Dose/yr}}$$

where the numerator represents the maximum tolerable dose of the component, and the denominator corresponds to the annual dose rate in its operational environment.

As energetic particles traverse the satellite structure, they continuously deposit energy in the material through ionization and scattering processes. Although shielding materials attenuate the incident radiation via stopping power mechanisms, a fraction of the particle energy still penetrates and contributes to the dose accumulated by internal components. Over extended mission durations, this accumulated dose can exceed the component's

damage threshold, ultimately leading to permanent degradation or complete functional failure.

Single Event Effects

Single-Event Effects (SEEs) are highly localized phenomena induced by individual energetic particles, in contrast to the gradual cumulative damage caused by TID and Displacement Damage (DD). When a single particle traverses the active volume of an electronic device, it deposits energy in the semiconductor material, generating an ionization track of electron-hole pairs along its trajectory, as illustrated in Figure 3.5.

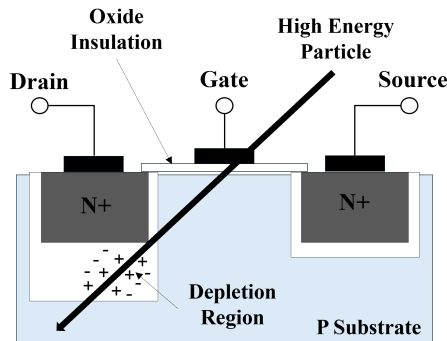


Figure 3.5. Single-event effect mechanism [195]

The energy deposited per unit path length by an incident particle is characterized by its Linear Energy Transfer (LET), typically expressed in $\text{MeV cm}^2 \text{mg}^{-1}$. This parameter provides a fundamental measure of a device's susceptibility to single-event effects, since the ionization density along the particle track directly determines the magnitude of the induced charge transient [230].

Within the neutral regions of a semiconductor, the generated electron-hole pairs typically recombine without observable effects. However, near a p-n junction, the built-in electric field can separate these carriers before recombination occurs. The subsequent collection of the separated charges induces a transient current pulse that may disturb normal device operation. Depending on the device architecture and the characteristics of the incident particle, SEEs can manifest as either destructive phenomena,

such as Single-Event Latchup (SEL), or non-destructive events, including SEU, Single-Event Functional Interrupt (SEFI), and Single-Event Transient (SET).

A SEU occurs when the ionization generated by a single particle alters the logic state of a digital device. In memory cells, the stored digital value transitions (from logic 1 to 0 or vice versa) if the collected charge exceeds a device-specific *critical charge* threshold. Importantly, SEU events are non-destructive and reversible, causing no permanent damage to the affected component.

SEFI represents a more severe non-destructive condition wherein the device ceases normal functionality and typically requires a complete power cycle to restore nominal operation. In analog integrated circuits, single-particle strikes can generate momentary voltage or current spikes on device outputs, known as SET.

The most critical single-event effect is latch-up (SEL), which occurs when the energy deposited by radiation triggers a parasitic path inside the integrated circuit, creating a short connection between the power and ground rails. If the resulting current stays within safe limits, the device may simply lock up temporarily and recover normal operation after a power cycle. However, if the current exceeds the device's thermal tolerance, the excessive heat can permanently damage the silicon junctions, leading to irreversible failure and, in severe cases, damage to other connected components [195].

According to the European Cooperation for Space Standardization (ECSS), devices exhibiting latch-up susceptibility below a LET threshold of $68.2 \text{ MeV cm}^2 \text{ mg}^{-1}$ cannot be classified as latch-up immune [64].

3.3 FPGAs in Space

As discussed in Section 3.1, several current space missions illustrate a growing trend toward the adoption of heterogeneous computing paradigms based on COTS co-processing architectures for onboard artificial intelligence, particularly in LEO and short-duration contexts where environmental radiation levels and mission risk tolerances are more favorable. Table 3.3 illustrates the high power efficiency and computational throughput achieved by these platforms.

Table 3.3. Performance and power consumption of edge AI accelerators [43]

Accelerator/Platform	Manufacturer	Technology	TOPS	Power (W)
Movidius Myriad 2	Intel	VPU	2	2
Movidius Myriad X	Intel	VPU	4	2
Google Coral	Google	TPU	4	2
NVIDIA Jetson Xavier NX	NVIDIA	GPU	21	10

Despite offering attractive computational performance and cost-efficiency, COTS processors remain inherently vulnerable to ionizing radiation, which poses significant challenges for their deployment in more demanding space scenarios. Table 3.4 presents the radiation tolerance characteristics of selected COTS devices, specifically their TID endurance and SEL immunity.

Table 3.4. Radiation tolerance characteristics of AI accelerators for space applications

Device	TID (Krad)	SEL Immunity (LET)	Ref.
Myriad 2 VPU	49	8.8 MeV·cm ² /mg	[176]
RT PolarFire SoC	100	68.2 MeV·cm ² /mg	[183]
RT PolarFire FPGA	100	80 MeV·cm ² /mg	[182]
Myriad X VPU	20	49.3 MeV·cm ² /mg	[35]
Google Coral TPU	30	57.3 MeV·cm ² /mg	[35]
NVIDIA Jetson Xavier	20	40 MeV·cm ² /mg	[187, 30]

FPGAs have long played a central role in space avionics architectures, particularly for accelerating high-performance digital signal processing

workloads. Available in space-grade variants for several decades, they have been widely adopted across numerous space missions. Their inherent reconfigurability, combined with highly parallel processing capabilities, makes them especially suitable for computationally intensive applications. In this context, **FPGAs** are frequently employed for real-time data transformations—such as compression and decompression—essential to managing the large data volumes generated by onboard sensors and instruments. They also support the implementation of space-qualified communication interfaces, including SpaceWire and SpaceFibre, which enable high-speed and fault-tolerant data exchange between spacecraft subsystems [108, 123]. Table 3.5 summarizes a selection of the main space-grade **FPGAs** currently available on the market, highlighting their radiation-tolerance characteristics.

Table 3.5. Radiation Tolerance Characteristics of Space-Grade FPGAs

Device	TID (Krad)	SEL Immunity (LET)	Ref.
Kintex UltraScale	100	80 MeV·cm ² /mg	[3]
Virtex-5QV	1000	>100 MeV·cm ² /mg	[233]
RT PolarFire	100	80 MeV·cm ² /mg	[140]

The Xilinx Virtex-5QV (XQR5XVFX130) [233], introduced in 2011, was the first high-performance radiation-hardened reconfigurable **FPGA** designed for processing-intensive space applications and has since been adopted on hundreds of spacecraft. It provides exceptional radiation hardness, withstanding up to 1 Mrad of **TID**, and—thanks to its dedicated rad-hard design—exhibits remarkable resilience to single-event effects (SEU rate in configuration memory at GEO $\sim 3.8 \times 10^{-10}$ upsets device⁻¹ day⁻¹ [34]), despite relying on **RAM**-based configuration storage.

Driven by the increasing demand for higher-density and higher-performance **FPGAs** in advanced space missions, Xilinx has recently undertaken major efforts to qualify a new generation of devices for space use, notably the 20 nm Kintex UltraScale family [3]. The UltraScale XQRKU060 significantly exceeds the Virtex-5QV in terms of logic density, speed, and overall processing capability. However, with regard to radiation performance, the Kintex UltraScale can be classified as radiation-tolerant rather than

radiation-hardened, offering robust but comparatively lower resistance to ionizing and single-event effects.

The Microchip RT PolarFire, when compared to the Xilinx Kintex UltraScale, provides slightly fewer resources but is inherently immune—by design—to configuration-bitstream SEUs. Moreover, RT PolarFire reduces power consumption to roughly half that of equivalent SRAM-based FPGAs offering similar density and performance, representing a major advantage for space applications where power management is critical [137].

In recent years, the use of FPGAs for deploying DNNs has expanded beyond the EO domain, supporting autonomy at the edge in planetary exploration and proximity operations. NASA’s *Mars 2020* mission, which deployed the *Perseverance* rover, incorporated a *Vision Compute Element (VCE)* that couples a radiation-hardened RAD750 CPU with a radiation-tolerant *Xilinx Virtex-5QV FPGA*. This architecture enabled the acceleration of perception and autonomy workloads that would otherwise exceed the capabilities of the primary flight processor [223]. Onboard *Perseverance*, AI augments both in-situ science and mobility. The PIXL instrument—a spectrometer developed by NASA’s Jet Propulsion Laboratory in Southern California—performs adaptive sampling with real-time inference to achieve precise target registration, identify diagnostic mineralogical signatures, and prioritize scientifically valuable regions without ground-in-the-loop intervention. Complementarily, the *AutoNav* system executes onboard obstacle detection and path optimization, extending access to challenging terrains and increasing science time-on-target. Furthermore, onboard AI filters and ranks observations under stringent downlink constraints, progressively refines mineral classification through machine learning, and informs sample-acquisition decisions for the Mars Sample Return campaign, with selected specimens sealed in titanium tubes for subsequent analysis on Earth [148].

ESA’s *Hera* mission marks another milestone in autonomous space operations. As part of the AIDA (Asteroid Impact and Deflection Assessment) collaboration with NASA, *Hera* will conduct a detailed post-impact survey of the Didymos binary asteroid system following NASA’s DART (Double Asteroid Redirection Test) kinetic impact on Dimorphos [5]. Currently in flight, *Hera* relies on onboard computer-vision and AI algorithms that fuse data from multiple sensors to navigate and operate efficiently far from Earth—reducing reliance on ground control and enabling timely guidance

updates. These workloads are offloaded from the OBC to a dedicated image processing unit, which integrates two radiation-tolerant FPGAs: a NanoXplore NG-Medium handling avionics interfaces (e.g., SpaceWire), command/telemetry services, and in-flight reprogramming, and a Xilinx Virtex-5QV dedicated to high-throughput image processing for vision-based navigation [19].

While COTS-based accelerators provide an attractive plug-and-play option for onboard AI, simplifying development and integration compared to FPGA-based designs, their limited radiation tolerance restricts their use to LEO and short-duration missions. Conversely, fully radiation-hardened FPGAs such as the Xilinx Virtex-5QV offer outstanding robustness but at the expense of power efficiency and architectural flexibility.

Bridging this gap, the radiation-tolerant PolarFire FPGA emerges as a compelling compromise, combining high reliability with low power consumption and in-flight reconfigurability. Its flash-based technology provides intrinsic immunity to configuration SEUs, while maintaining competitive logic density and performance. Furthermore, the platform integrates the CoreVBX dataflow accelerator for CNNs, available as a ready-to-use IP core that can be instantiated directly within custom spacecraft subsystems, enabling reliable and efficient AI acceleration in space [176, 51, 64].

Recent demonstrations on commercial PolarFire devices—running quantized CNNs trained on EuroSAT with TensorFlow and PyTorch—show substantial improvements in throughput and energy efficiency, reinforcing the suitability of this architecture for edge AI in space. Crucially, the same design flow and accelerator map to the radiation-tolerant family, providing a straightforward path to space-qualified on-board intelligence [51].

3.3.1 Fault Mitigation Strategy

The intrinsic radiation resilience of RT PolarFire devices arises from their flash-based configuration architecture. Unlike conventional SRAM-based FPGAs, which require continuous configuration monitoring and periodic scrubbing to correct radiation-induced upsets, RT PolarFire FPGAs employ non-volatile memory cells that are inherently immune to such effects. This robustness results from the use of floating-gate Metal–Oxide–Semiconductor Field-Effect Transistors (FG-MOSFETs), which reliably retain their programmed state even under ionizing radiation exposure [217].

In contrast, SRAM-based configuration cells are vulnerable to single-event upsets (SEUs) caused by high-energy particles. Since these cells define the functionality of configurable logic blocks, routing resources, and I/O structures, upsets in configuration memory can directly alter circuit behavior—potentially disrupting routing paths or modifying logic operations [126]. To mitigate these effects, designers often adopt fault-tolerant strategies such as Triple Modular Redundancy (TMR), which replicates critical logic and employs majority voting to mask single faults. Configuration memory integrity can also be restored through scrubbing—a process that continuously reads and rewrites configuration data—but this approach increases power consumption. By eliminating the need for scrubbing, RT PolarFire devices achieve greater power efficiency without compromising functional reliability [227].

Nevertheless, while the configuration memory is resilient to SEUs, the user logic and embedded memories remain susceptible to transient radiation effects, including SEUs and SETs, which may be induced by cosmic rays, charged particles, or electromagnetic disturbances [37].

To ensure system reliability, a comprehensive suite of mitigation techniques has been developed for flash-based FPGAs, addressing specific vulnerabilities in both sequential and combinational logic. Among these, TMR remains the foundational method for SEU mitigation. TMR can be implemented in various forms—sequential, distributed, or global—based on logic replication and majority voting. As demonstrated in [165], applying local TMR to D-type flip-flops in RT PolarFire FPGAs can significantly reduce susceptibility to SEU, lowering the estimated error rate from 1×10^{-7} to 1×10^{-11} errors/bit-day under GEO SolarMin conditions. Building on this result, the use of synthesized local TMR was explored for the CoreVBX neural network IP. While this approach is expected to improve reliability, it introduces considerable implementation overheads: logic utilization increases from 13% to approximately 33% of the available resources, and the maximum clock frequency drops by about 33%, leading to a 35%–45% reduction in inference throughput. However, it is important to note that no radiation testing has yet been performed on the CoreVBX—either with or without TMR—and thus the actual fault tolerance benefits for this specific IP remain unverified.

For SET mitigation, more refined approaches have emerged to address

the limitations of full redundancy techniques. In particular, [207] presents an advanced design flow that analytically identifies the most critical flip-flops and logic paths based on dynamic propagation models of SET pulses. This enables the selective insertion of guard gates only where necessary, drastically reducing unnecessary overhead. The approach, validated on several benchmark circuits including a RISC processor, achieved a two-order-of-magnitude improvement in SET resilience while reducing area overhead by more than 83% compared to traditional TMR combined with full guarding.

Another mitigation technique for user memory involves the use of Error Detection and Correction (EDAC) schemes, combined with orbit-specific scrubbing intervals adapted to the radiation flux and mission profile [37]. At the design level, additional hardening strategies further enhance reliability, including safe finite-state machine (FSM) coding with explicitly defined recovery paths and the use of Hamming distance-3 encodings. These techniques prevent radiation-induced state corruption from driving the system into undefined or hazardous behavior [150].

Overall, the combination of flash-based immunity to configuration faults and the integration of selective, resource-efficient fault mitigation strategies enables RT PolarFire FPGAs to operate reliably in harsh space radiation environments while preserving the performance and reconfigurability advantages that define modern programmable logic devices [217].

3.3.2 Microchip RT PolarFire SoC

Among recent advances, *Microchip* has introduced the RT PolarFire SoC [141], which marries a real-time, Linux-capable RISC-V microprocessor with the rugged, radiation-tolerant PolarFire FPGA fabric. This blend of deterministic control and reconfigurable acceleration opens a clear path to sophisticated, on-board intelligence for space systems.

Radiation test results in [183] highlight the device's robustness: no configuration upsets or SELs were observed up to a LET of $68.2 \text{ MeV cm}^2 \text{ mg}^{-1}$, supporting its classification as SEL-immune under ECSS guidance [64]. The SoC also maintained correct operation through 100,krad TID, underscoring suitability for safety-critical, radiation-intense missions.

With a hard RISC-V processor capable of running Linux, developers inherit a mature and flexible environment that simplifies task schedul-

ing, cleanly partitions functionality between software and hardware, and accelerates the integration of high-level AI frameworks alongside custom accelerators. In sum, the architecture is a compelling foundation for next-generation autonomous spacecraft subsystems.

Building on these strengths, this thesis adopts the RT PolarFire SoC family as the primary compute platform for reliable on-board AI. The next chapter presents a maritime case study featuring a tailored deep-learning model for reliable small-target detection (e.g., vessels). The model is implemented on the Microchip PolarFire SoC Video Kit, which serves as an evaluation platform for the space-grade counterpart, thereby demonstrating near-real-time detection and time-critical alerting directly on board.

SR-YOLOv5s for Maritime Surveillance

As introduced in Chapter 1, detecting small targets in optical satellite imagery plays a crucial role in various monitoring and surveillance applications, supporting emergency response, situational awareness, and the identification of anomalous or illegal activities. However, many targets manifest as small objects with limited spatial footprint, low contrast against backgrounds, and visual occlusion from atmospheric haze or cloud cover, making accurate detection extremely challenging.

This thesis investigates the use of DL-based **SISR** to enhance small-object detection in maritime domain and introduces **SR-YOLOv5s**—an improved variant of YOLOv5s [99]—integrating a custom lightweight **SISR** backbone. The approach permanently integrates the super-resolution module into the pipeline, enabling joint training to enhance fine-grained spatial details and improve feature extraction. Unlike conventional end-to-end architectures, the **SR** module is fully convolutional and compact, designed for efficient deployment on embedded devices under computational and power constraints.

To assess performance, we construct the **LR-ShipRSImage** dataset, derived from ShipRSImageNet [240], specifically designed for detecting small and very small vessels in satellite imagery. The dataset introduces a novel *very small ship* category, defined as instances with bounding box area $\leq 7 \times 7$ pixels, and includes diverse images capturing vessels under varying

maritime conditions—calm and rough seas, haze, varying illumination, and partial cloud coverage. The primary objective is to assess whether super-resolution can improve accuracy when targets occupy only a few pixels. To increase task complexity, we augment the dataset with **LR-CloudyTest**, a synthetic test subset created by overlaying procedurally generated cloud masks onto test images, simulating degraded visibility scenarios.

To evaluate **SR** benefits under consistent conditions, *LR-ShipRSImage* is paired with a high-resolution counterpart where identical objects occupy more pixels. This enables assessing whether SR-YOLOv5s can reconstruct sufficient detail to match baseline models operating on native high-resolution inputs. For generalization capability, we test the model on two MASATI dataset [13] subsets: one containing isolated small vessels in open sea, another with only background scenes. This external validation confirms robustness across diverse operational scenarios and applicability to heterogeneous maritime imagery.

Beyond model design and dataset construction, we investigate deploying SR-YOLOv5s on FPGA using the *VectorBlox SDK* and the overlay-based *CoreVBX* accelerator. Rather than synthesizing the network directly into programmable logic, the inference pipeline is mapped onto the overlay architecture. We evaluate both performance and development workflow, demonstrating feasibility of rapid prototyping and efficient edge-AI deployment on resource-constrained platforms.

The hardware design is implemented on Microchip’s *PolarFire SoC Video Kit* and evaluated for real-time performance. To assess portability and efficiency, we benchmark the full pipeline on two additional platforms—NVIDIA Jetson Orin Nano and NVIDIA A100—highlighting trade-offs in accuracy, throughput, and power consumption. While COTS AI accelerators are currently used in space missions, their limited radiation tolerance restricts long-term or high-radiation applications. Our design is fully portable to the *RT PolarFire SoC* [165], providing a robust foundation for radiation-tolerant onboard AI inference systems.

The main contributions of this section are as follows:

- **SR-YOLOv5s**: An end-to-end model integrating a lightweight super-resolution backbone optimized for FPGA deployment, enhancing small vessel detection in satellite images.

- **LR-ShipRSImage dataset:** Featuring over 4,000 very small ships ($\leq 7 \times 7$ pixels) and 9,200 small ships ($\in [7 \times 7, 32 \times 32]$ pixels), covering diverse maritime conditions and enriched with synthetic cloud-masked imagery.
- Evaluation of Microchip’s **CoreVectorBlox** for deploying SR-YOLOv5s, confirming effectiveness for rapid, low-power AI deployment.
- Benchmarking on three platforms—**NVIDIA Jetson Orin Nano**, **NVIDIA A100**, and **PolarFire SoC Video Kit**—comparing inference speed, accuracy, and power consumption.

The remainder is structured as follows. Section 4.1 reviews relevant literature on [SISR](#) and [OD](#), highlighting super-resolution techniques for improving small object detection in satellite imagery. Section 4.2 introduces the SR-YOLOv5s architecture, training pipeline, experimental setup including the custom dataset, and hardware design with the [CoreVBX IP](#) core on PolarFire [SoC](#). Section 4.3 reports and analyzes results.

4.1 Related Works

4.1.1 Super-Resolution

Super-Resolution encompasses techniques aimed at enhancing spatial resolution by reconstructing finer details from low-resolution inputs. The primary goal is to overcome imaging system limitations and improve detail levels, making images more suitable for visualization, analysis, and other computer vision tasks [234].

Recent advances in [DL](#) have transformed the field, with modern algorithms achieving state-of-the-art performance and significantly surpassing traditional fixed-kernel methods [109]. These models learn intricate patterns and structures from data, enabling reconstruction of finer details and delivery of more accurate, high-resolution outputs compared to conventional approaches like bicubic interpolation.

[SR](#) methods are categorized into [SISR](#) and Multiple-Image Super-Resolution ([MISR](#)). [SISR](#) recovers high-resolution images from single low-resolution observations by learning spatial priors or statistical mappings, making it well-suited for real-time or onboard applications where

only one frame is available [114]. Conversely, **MISR** leverages information from multiple low-resolution images—typically acquired with sub-pixel shifts—to reconstruct higher-quality outputs by exploiting complementary details across observations. While **MISR** often achieves superior reconstruction, it requires accurate image registration and is computationally demanding, limiting applicability in low-latency or resource-constrained environments [101].

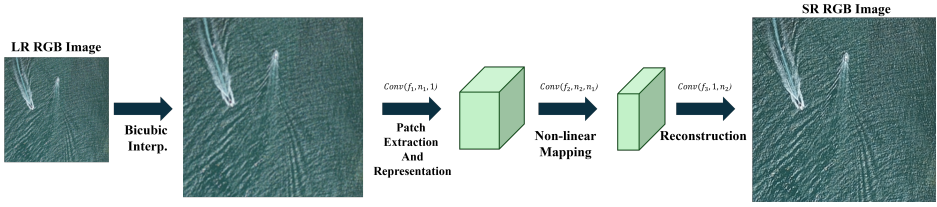


Figure 4.1. SRCNN architecture

Super-Resolution Convolutional Neural Network (**SRCNN**) [60] is a pioneering deep learning model that enhances low-resolution images using a fully convolutional network to reconstruct high-resolution details from bicubic-interpolated inputs.

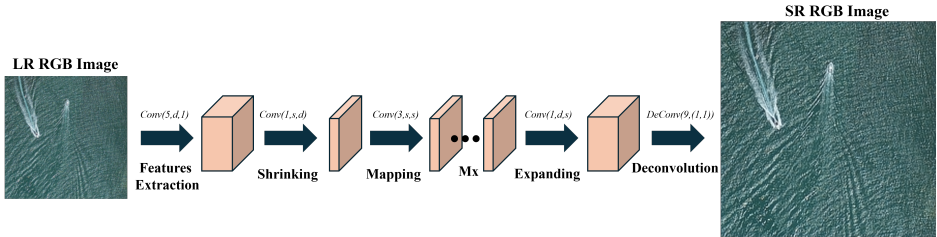


Figure 4.2. FSRCNN architecture

Fast SRCNN (**FSRCNN**) [61] is a highly efficient model designed for real-time tasks, developed as an advanced alternative to SRCNN for faster inference and improved performance, particularly for single-image and video applications. Unlike SRCNN, which operates primarily in High-Resolution (**HR**) space, FSRCNN conducts most processing in Low-Resolution (**LR**) space, accepting low-resolution input and performing upscaling only at the final stage using a deconvolutional layer [236], significantly reducing

computational complexity. Architectures are illustrated in Figures 4.1 and 4.2.

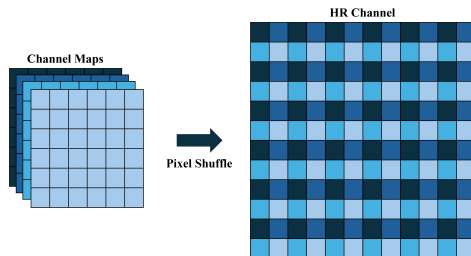


Figure 4.3. Pixel-shuffle layer

Efficient Sub-Pixel Convolutional Network (ESPCN) [199] represents another innovative approach to real-time single-image and video super-resolution, utilizing a novel sub-pixel convolutional layer that transforms LR feature maps into HR images directly at the network’s output. The sub-pixel layer employs a pixel-shuffle operation where, given c input channels and upscale factor r , it rearranges pixels from $r^2 \times c$ feature maps along the channel dimension. Figure 4.3 illustrates this layer, optimizing upscaling with minimal computational overhead.

4.1.2 Object Detection

OD enables identification and localization of objects within images by recognizing them from various classes and determining boundaries with bounding boxes [26]. Since its 2015 introduction, You Only Look Once (YOLO) [178] versions have rapidly evolved, with YOLOv5 being one of the most widely adopted [88].

The YOLOv5 architecture [214], developed by Ultralytics, is optimized for fast and accurate detection. As a single-stage detector, it predicts bounding boxes and class probabilities directly from input images in a single forward pass, enabling real-time applications. The architecture comprises: an input layer; CSPDarknet53 backbone for feature extraction; Path Aggregation Network (PANet) neck merging feature maps across scales; and YOLO detection layer with multiple heads at three scales for detecting small, medium, and large objects. Post-processing employs Non-

Maximum Suppression (NMS) [82] and confidence thresholding to refine the final detections. Figure 4.4 illustrates the overall architecture.

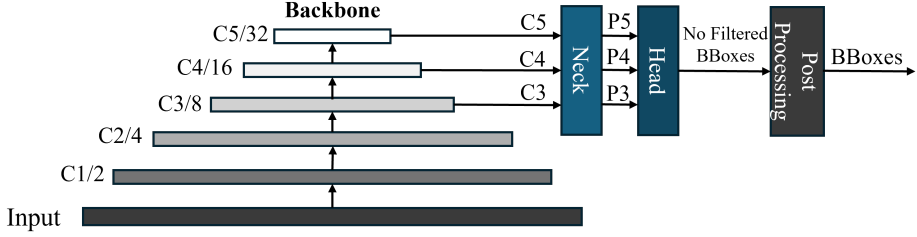


Figure 4.4. YOLO architecture [121]

Input images first pass through the backbone, which extracts hierarchical feature maps capturing both low-level textures and high-level semantics across multiple spatial resolutions. These multi-scale features are then processed by the neck, a feature pyramid and path aggregation module that fuses information across layers to enhance object localization and scale invariance. The resulting feature maps—denoted as $P3$, $P4$, and $P5$ —have resolutions of 52×52 , 26×26 , and 13×13 respectively for a 416×416 input, and are specialized for detecting small, medium, and large objects.

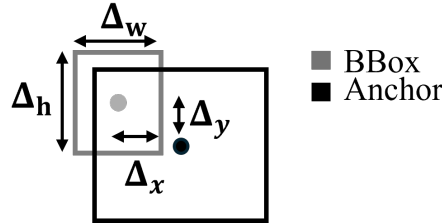


Figure 4.5. YOLO anchor-based prediction [121]

YOLOv5 adopts an anchor-based detection strategy, in which each feature map cell is associated with several predefined anchor boxes (w, h) that represent typical object aspect ratios. During training, each ground-truth bounding box is assigned to one or more anchors whose Intersection Over Union (IoU) [181] exceeds a predefined threshold. Unlike earlier YOLO versions, YOLOv5 extends the assignment to neighboring cells,

enabling multiple positive anchors to contribute to the same target and improving detection of objects near grid boundaries.

For each assigned anchor, the detection head predicts bounding-box offsets ($\Delta x, \Delta y, \Delta w, \Delta h$) relative to the anchor center and size, together with an objectness score and class probabilities (Figure 4.5).

Anchor dimensions are obtained by applying k -means clustering [6] on the training dataset, using an IoU-based distance metric to capture the empirical distribution of target shapes.

At inference time, YOLOv5 adjusts each anchor’s position and scale using the predicted offsets, generating candidate bounding boxes. These are filtered by confidence thresholds and refined through Non-Maximum Suppression (NMS) [82] to eliminate redundant detections, yielding the final prediction set (Figure 4.6).

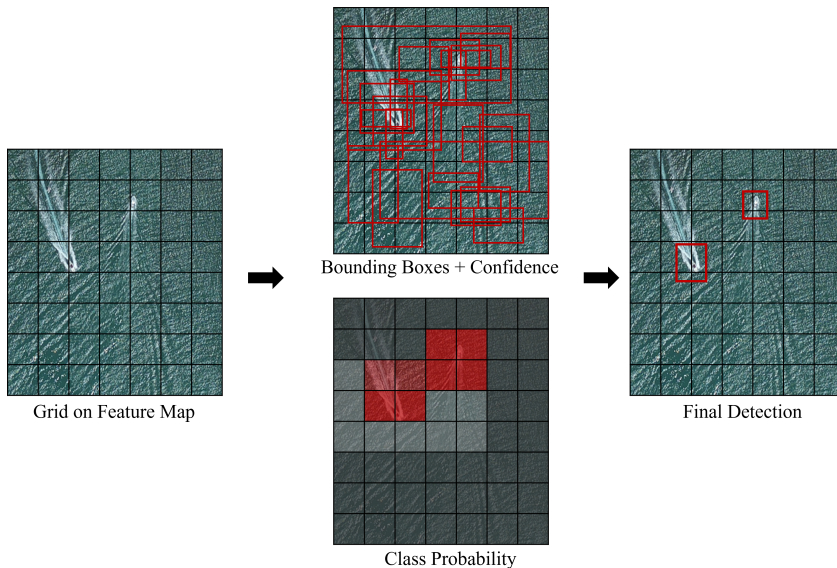


Figure 4.6. YOLO post-processing [121]

YOLOv5 is released in four model variants—YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), and YOLOv5x (extra-large)—which differ primarily in backbone and neck depth (number of layers) and width (channel scaling). This modular design allows users to balance inference

speed and detection accuracy according to mission requirements. The smallest variant, YOLOv5s, with only 7.5 million parameters, achieves an outstanding trade-off between efficiency and precision [97].

4.1.3 Super-Resolution for Object Detection

Recent studies have explored integrating SR techniques to enhance image quality prior to or during detection [38]. One of the earliest works [198] evaluated SISR impact on performance using the *xView* dataset. Very Deep Super-Resolution (VDSR) and a Random Forest-based algorithm simulated imagery at multiple GSD levels (from 30 cm to 4.8 m), assessing performance across upsampling scales ($2\times$, $4\times$, $8\times$) using detectors like Single Shot Detector (SSD). Results showed significant mAP gains: enhancing 30 cm imagery to 15 cm improved mAP by 13%, while upscaling 60 cm to 15 cm yielded 20% increase.

Building on these findings, subsequent approaches proposed end-to-end models jointly optimizing SISR and OD components. For example, [174] introduces an edge-aware GAN-based architecture (EESRGAN) coupling an SR generator with a detector (SSD or Faster R-CNN), trained with composite loss including adversarial, perceptual, and detection terms. This yielded notable gains in vehicle detection on the COWC dataset, improving mAP from 61.9% to 88.1% (SSD), and from 64% to 93.1% (Faster R-CNN). In [120], ESRTMDet integrates a lightweight SR module directly into the RTMDet framework (based on YOLOX). Experimental results demonstrate that ESRTMDet achieves accuracy comparable to models trained and tested on high-resolution inputs, consistently outperforming methods applying SR as separate preprocessing.

Recent studies have also explored using super-resolution modules exclusively during training. These auxiliary branches guide the backbone toward learning HR feature representations from low-resolution inputs. Crucially, SR components are discarded at inference, allowing models to retain training benefits without additional computational overhead during deployment. Specifically, [238] proposes *SuperYOLO*, a detector for multimodal remote sensing images (RGB + infrared), augmented with an encoder-decoder SR module active only during training. This auxiliary branch encourages the backbone to learn HR features while being removed at inference to preserve speed. *SuperYOLO* achieves 18.3% improvement over YOLOv5s

baseline on the VEDAI dataset [177]. Similarly, [122] presents SOD-Net, integrating a Super-Resolution Perception Branch (SRPB) based on ESPCN. Also active only during training, this auxiliary module enhances feature extraction capabilities without impacting runtime performance. *SOD-Net* achieves state-of-the-art results on VEDAI, reaching 82.6% *mAP50* and outperforming *SuperYOLO* in both accuracy and computational efficiency.

While these works confirm super-resolution effectiveness in boosting small-object detection, they often rely on GPU-based inference or omit hardware deployment considerations and embedded platform applicability. Consequently, their practical use in real-time remote sensing scenarios—where bandwidth, latency, and autonomy are critical—remains limited. This gap underscores the importance of onboard processing strategies capable of delivering rapid, in-situ analytics within satellite system constraints.

4.2 Materials and Methods

This section presents the SR-YOLOv5s architecture, highlighting the convolutional SR-Backbone that improves YOLOv5s performance in detecting small vessels in optical satellite imagery. Then, details related to the FPGA design on the *Microchip PolarFire SoC* are provided. On PolarFire, SR-YOLOv5s is deployed via the *CoreVBX* integrated with the VectorBlox SDK, enabling power-efficient, real-time onboard processing.

4.2.1 SR-Backbone

In this subsection, the Super-Resolution Backbone (*SR-Backbone*) is presented — a novel CNN architecture specifically developed for real-time *SISR* applications. The model is optimized as an integral component of object detection networks, featuring a lightweight design that minimizes memory footprint and computational overhead. The architecture of the *SR-Backbone* effectively prevents significant increases in the computational demands of the primary network, thereby ensuring rapid inference speeds while maintaining high capability for reconstructing HR images. It integrates elements from the *FSRCNN* and *ESPCN* models, utilizing the *ReLU* for activation.

The network is composed of several layers, each serving a specific

function. The first hidden layer, inspired by [FSRCNN](#), handles feature extraction through 56 convolutional filters of size 5×5 applied directly to the [LR](#) input. A subsequent shrinking layer employs 12 convolutional filters of size 1×1 to decrease feature dimensionality, reducing computational complexity. Four additional convolutional layers extract fine spatial details within the feature maps for efficient nonlinear mapping from [LR](#) to [HR](#) space. Then, an expansion layer restores the number of feature maps to match the first hidden layer, using 56 convolutional filters of size 1×1 . Following the [ESPCN](#) model, [HR](#) image reconstruction is achieved through the pixel-shuffle operation. The final convolutional layer performs RGB channel regression, after which the model combines the feature maps along the channel dimension to produce a final output of size $r^2 \times c$. This approach enables efficient upscaling while ensuring high-quality image reconstruction. The Mean Square Error ([MSE](#)) is used as the loss function during training. Figure 4.7 illustrates the [SR-Backbone](#) architecture.

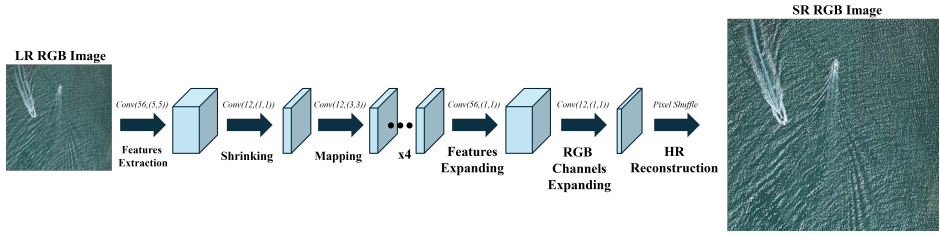


Figure 4.7. SR-Backbone architecture

4.2.2 SR-YOLOv5s

This thesis proposes SR-YOLOv5s, an enhanced variant of the YOLOv5s object detector that integrates the [SR-Backbone](#) within its architecture. The overall structure of SR-YOLOv5s is shown in Figure 4.8. The backbone's purpose is to map small objects from low-resolution to high-resolution space, thereby increasing YOLOv5s's detection accuracy for these targets. It reconstructs fine spatial details of the input image, improving detection performance without significantly impacting inference time. The overall YOLOv5 architecture remains unchanged, except for replacing the original [SiLU](#) activation function [14] with the [ReLU](#) activation function. This mod-

ification ensures compatibility with the FPGA-based CoreVBX accelerator, as the version 1 (v1) release employed in this study does not support the SiLU activation function.

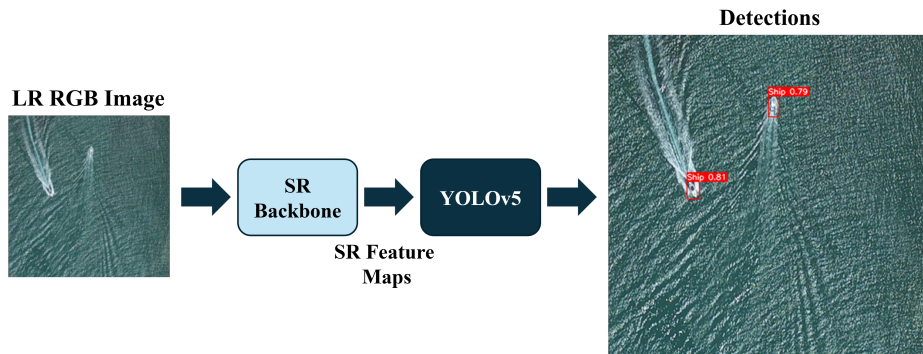


Figure 4.8. SR-YOLOv5s architecture

4.2.3 Dataset and Training Details

A comprehensive dataset is developed from ShipRSImageNet [240], specifically tailored for ship detection in high-resolution optical remote sensing imagery. It consists of 3,435 RGB images, each with a resolution of approximately 930×930 pixels, acquired from a variety of sensors, satellite platforms, geographic regions, and seasonal conditions. The dataset includes vessels exhibiting a wide range of scales, orientations, and aspect ratios.

Figure 4.9 illustrates sample images from the reference dataset with ground-truth bounding boxes. The imagery encompasses a diverse set of maritime scenarios, including open ocean, rough sea states, port infrastructures, coastal proximity, and a variety of adverse atmospheric conditions such as cloud cover, fog, haze, and low solar elevation angles. This heterogeneity introduces substantial background complexity, which is critical for systematically evaluating the robustness of object detectors to false positives. Moreover, it enables the development and validation of operationally reliable onboard detection systems capable of maintaining accuracy across a wide range of real-world environmental conditions.

Following the COCO [118] object size classification, annotated bounding boxes are categorized into *small*, *medium*, and *large* based on their pixel

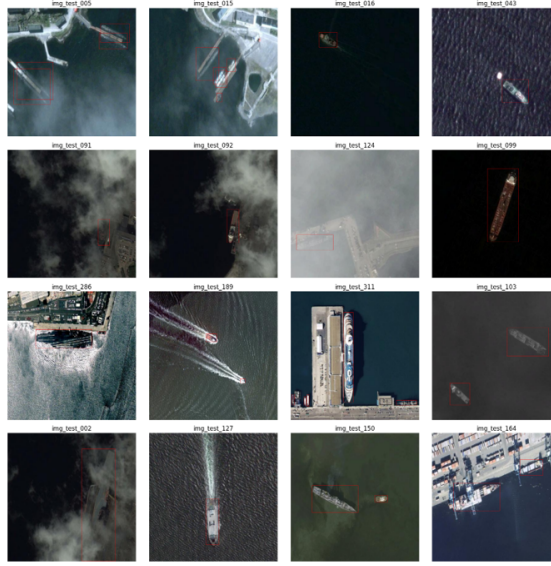


Figure 4.9. ShipRSImageNet dataset

area. Additionally, an extra class, *very small*, is introduced to account for particularly challenging instances with an area less than or equal to 49 pixels (e.g., 7×7 or smaller). Stratified sampling is then applied across these categories to allocate images with an 80%–15%–5% split for training, validation, and testing, respectively, ensuring balanced representation of object sizes across all data partitions.

To enable a rigorous evaluation of object detection enhanced by super-resolution, two complementary versions of the dataset were constructed through systematic image processing steps. The first variant, referred to as *HR-ShipRSImage*, was generated by resizing the original images to 416×416 pixels while preserving their aspect ratios. Bounding box annotations were scaled accordingly to maintain spatial accuracy within the resized images. The second variant, referred to as *LR-ShipRSImage*, was obtained by applying a Gaussian filter [199] to the high-resolution images, followed by a downsampling step that reduced each image to a resolution of 208×208 pixels. Figure 4.10 and Figure 4.11 illustrate the distribution of bounding box sizes in the *HR-ShipRSImage* dataset. Specifically, Figure 4.10 displays scatter plots showing the range of object

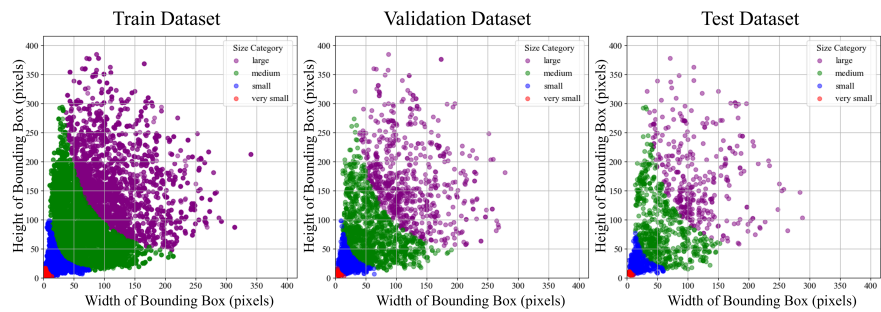


Figure 4.10. HR-ShipRSImage—Ship size class distribution (scatter plot)

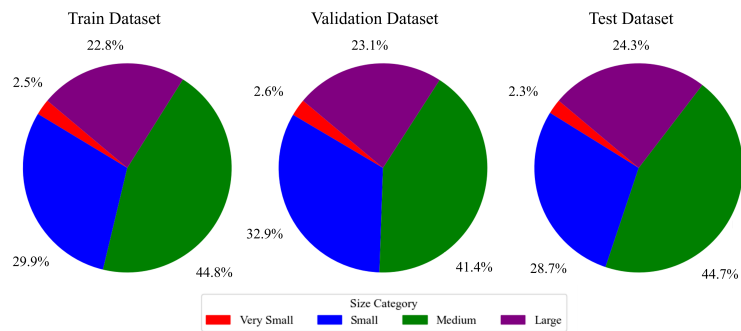


Figure 4.11. HR-ShipRSImage—Ship size class distribution (pie chart)

sizes across the training, validation, and test sets, while Figure 4.11 presents a pie chart summarizing the proportion of each size category.

The downsampling process reduces spatial resolution in a controlled manner, increasing the relative frequency of small and very small object instances. This shift in object scale distribution creates a more challenging detection setting, ideal for assessing the ability of super-resolution models to recover fine spatial details essential for accurate small-object detection. As shown in Figure 4.10 and Figure 4.11 (scatter plot distribution and pie chart), the proportion of *very small* and *small* instances increases significantly in the *LR-ShipRSImage* dataset compared to the high-resolution version, making the detection task more challenging. Specifically, the low-resolution test set contains a total of 143 *very small* and 664 *small* ship instances, whereas the corresponding high-resolution test set includes only 32 *very small* and 402 *small* instances.

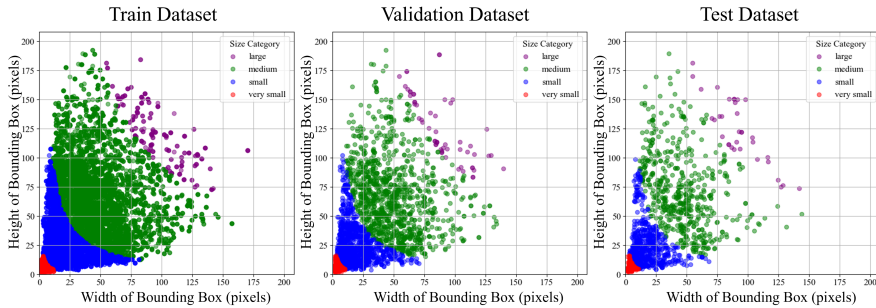


Figure 4.12. HR-ShipRSImage—Ship size class distribution (scatter plot)

Within this framework, the *LR-ShipRSImage* dataset is employed as input to both the baseline YOLOv5s and the enhanced SR-YOLOv5s architectures, enabling a direct comparison of their performance under identical low-resolution conditions. The high-resolution variant (*HR-ShipRSImage*) is used to train the standard YOLOv5s model and serves as a reference benchmark, where objects are natively represented with higher pixel density. This experimental setup enables a controlled evaluation of whether super-resolution techniques can recover sufficient spatial detail to achieve detection accuracy comparable to that obtained with high-resolution inputs.

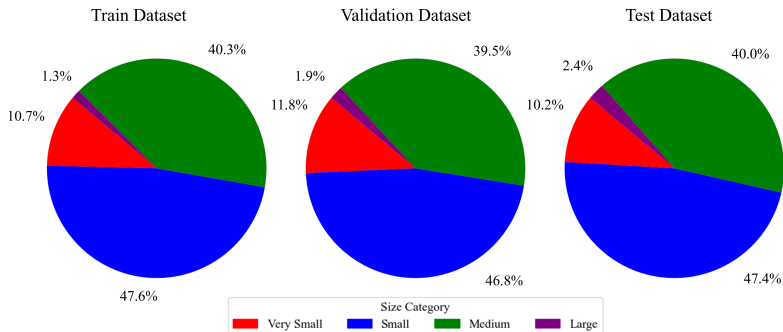


Figure 4.13. HR-ShipRSImage—Ship size class distribution (pie chart)

To simulate adverse weather conditions and atmospheric occlusions, an additional synthetic test set, referred to as *LR-CloudyTest*, was generated by augmenting the *LR-ShipRSImage* test images with procedurally generated cloud layers using the open-source *SatelliteCloudGenerator* tool [50]. This tool applies Perlin noise-based textures to overlay semi-transparent clouds with controllable parameters such as coverage (20%–80%), transparency (min: 0.0–0.3, max: 0.5–0.85), and edge softness. To ensure target visibility and avoid extreme occlusions, all images in which more than 30% of any bounding box was covered by clouds were filtered out. The final *LR-CloudyTest* set comprises 999 test images with realistic and diverse atmospheric occlusions, allowing for a thorough evaluation of model robustness under degraded optical conditions—one of the primary failure modes of vision-based detection systems.

To assess generalization performance beyond the constructed dataset, both SR-YOLOv5s and the baseline YOLOv5s models were evaluated on two additional test sets derived from the MASATI dataset [13]. The first set (*MASATI-ship*) comprises images containing isolated *small* vessels (587 instances) and *very small* vessels (438 instances) in open-sea environments, captured under varying illumination and meteorological conditions, and is used to test detection performance on external, previously unseen data. The second set consists of the *MASATI-water* subset (1,022 images), which includes various illumination and sea conditions but no ship instances, and is employed to evaluate model robustness against false positives. This setup

also allows investigation of whether the use of super-resolution leads to an increased false-positive rate compared to the baseline model.

Training, validation, and testing were conducted on an NVIDIA A100 GPU via the Google Colab cloud computing platform [73]. Initially, the SR-Backbone weights were trained for 150 epochs using the *Adam* optimizer [57], until no further improvements were observed in the MSE loss function, aiming to maximize similarity between the super-resolved output and the corresponding HR ground truth.

To evaluate reconstruction performance, several baseline models were included alongside the SR-Backbone. The SRCNN model was trained on the same dataset as a benchmark, given its status as the pioneering convolutional neural network for super-resolution tasks. Additionally, the FSRCNN and ESPCN models—both designed for real-time SR applications—were trained, together with the Enhanced Deep Residual Network (EDSR) [117], a state-of-the-art deep convolutional architecture for SR. Each model was trained for 150 epochs, following the optimal hyperparameter configurations recommended by their respective authors.

To enhance the robustness and statistical reliability of the evaluation conducted for SR-YOLOv5s, a *5-fold stratified cross-validation* strategy was employed over the combined training and validation set. This procedure allows the model to be trained and assessed across five distinct data partitions while preserving the original distribution of object-size classes within each fold. By mitigating the risk of overfitting to a specific split, this approach provides a more generalizable and reliable estimate of detection performance.

The training of SR-YOLOv5s followed a structured, multi-phase protocol. First, the baseline YOLOv5s model was trained on the *HR-ShipRSImage* dataset for 100 epochs using the original configuration provided by the YOLOv5 authors.

Figure 4.14 shows the loss and accuracy profiles during the initial training stage. Subsequently, using a staged transfer learning approach, SR-YOLOv5s was trained on the *LR-ShipRSImage* dataset through a three-phase procedure, applied independently for each fold. This strategy was designed to progressively adapt both the detection head and the super-resolution backbone to low-resolution inputs while maintaining model stability and optimizing performance. In the first phase (50 epochs), the

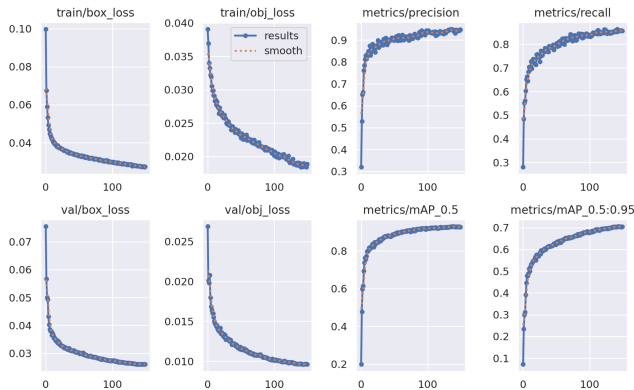


Figure 4.14. SR-YOLOv5s loss and accuracy during the first training step

SR-Backbone weights were frozen, allowing the YOLOv5s detection head to adapt to the high-resolution features generated by the pre-trained SR module. In the second phase (50 epochs), the detection head was frozen while the super-resolution backbone was fine-tuned to produce features more aligned with the detection objective. In the final phase (50 epochs), the entire architecture was trained jointly in an end-to-end manner, enabling full network adaptation and tight integration between the super-resolution and detection components.

To benchmark detection capability, the original YOLOv5s was also trained to recognize the target class on both the *LR-ShipRSImage* and *HR-ShipRSImage* datasets. Additionally, the performance of YOLOv5s with **ReLU** activation was compared against the standard **SiLU**-based model. All models were trained according to the specifications recommended by Ultralytics.

4.2.4 SR-YOLOv5s Deployment

SR-YOLOv5 is deployed on the Microchip PolarFire **SoC**, hosted within the Microchip PolarFire SoC Video Kit. The complete design, incorporating the *CoreVBX V1000* instance, utilizes approximately 72% of the available **LEs**, with nearly 70% allocated to **CoreVBX** functionality. Figure 4.15 presents the hardware design schematic.

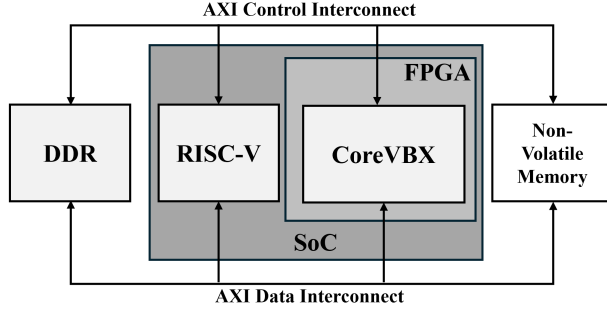


Figure 4.15. CoreVectorBlox-based SoC Hardware Design

Network and firmware **BLOBs** are stored in non-volatile memory accessible to the **CoreVBX**, which operates as a memory-mapped master unit. During boot, both elements are loaded into Double Data Rate (**DDR**) memory. The hard-core RISC-V processor transfers input images from external memory to **DDR**, performs initial preprocessing if required, and commands the **CoreVBX** to execute inference. The accelerator reads weights and layer configurations from the network BLOB to perform computations, writing outputs back to **DDR** for processor access. The processor supervises the overall workflow, running a C/C++ application under *Yocto Linux* to manage data flow and control the **AI** engine.

For comparative evaluation, SR-YOLOv5s was also deployed on the Jetson Orin Nano Developer Kit [154] to benchmark inference performance against execution on the PolarFire **SoC**. This assessment focuses on inference speed and power consumption, as well as on the capability of the VectorBlox SDK to optimize neural networks for efficient onboard deployment on the PolarFire **SoC**.

4.3 Results

This section presents the experimental evaluation of the proposed system. The analysis begins by quantifying the detection accuracy of SR-YOLOv5 using standard performance metrics, in order to isolate and assess the specific contribution of the integrated **SR** module. Subsequently, the study evaluates end-to-end throughput, latency, and power consumption

on a Microchip PolarFire SoC implementation, adopted as a low-power computing platform for on-board, near-real-time inference. The overarching goal is to demonstrate the feasibility of on-orbit extraction of actionable information—such as vessel localization and alert generation for maritime surveillance—within the broader EO context.

4.3.1 Ship Detection

The super-resolution capabilities of the SR-Backbone are evaluated on the test set using two standard SR metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [225]. Although it includes approximately 98% fewer parameters than the EDSR, the SR-Backbone achieves only slightly lower performance compared to reference models such as SRCNN, FSRCNN, and EDSR, the latter representing the state of the art for SISR tasks. Designed for compactness and computational efficiency, the SR-Backbone introduces minimal overhead to the overall computational load of the SR-YOLOv5s model.

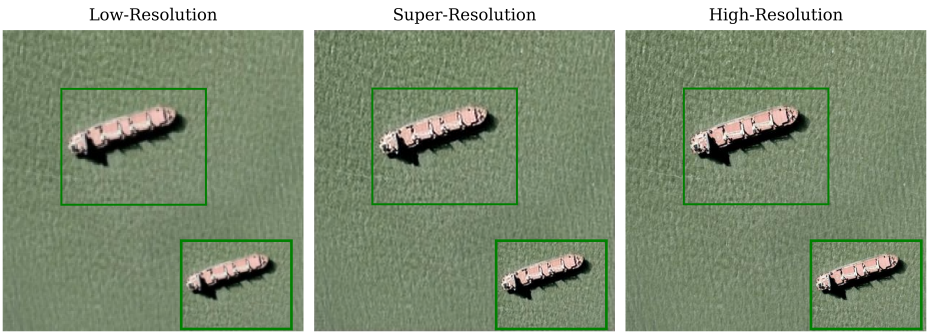


Figure 4.16. SR-Backbone reconstruction capability detail

Table 4.1. Super-Resolution Results

Model	Parameters	PSNR (dB)	SSIM
SRCNN	20,099	31.668	0.880
FSRCNN	24,683	31.978	0.889
EDSR	1,395,788	32.790	0.898
SR-Backbone	16,960	30.640	0.884

Table 4.1 details the performance comparison between the models. Figure 4.16 presents a visual comparison between the low-resolution input, the corresponding high-resolution ground-truth, and the super-resolved output produced by SR-Backbone. The reconstruction demonstrates the network’s ability to recover high-frequency spatial details, indicating the effectiveness of the super-resolution module in restoring fine-grained structures that are critical for accurate object detection.

Table 4.2. Model evaluation results on cross-validation test sets

Model	mAP50 All	mAP50:95 All	mAP50 Small	mAP50 Very Small
YOLOv5s	0.8544 ± 0.0054	0.6610 ± 0.0046	0.7972 ± 0.0129	0.2832 ± 0.0615
SR-YOLOv5s	0.9272 ± 0.0029	0.7604 ± 0.0072	0.9272 ± 0.0079	0.4658 ± 0.0221

Table 4.2 reports the average results obtained through *5-fold cross-validation* on the *LR-ShipRSImage* test set to evaluate the detection performance of SR-YOLOv5s compared to the baseline YOLOv5s. Specifically, we report mean values for both *mAP50* and *mAP50:95*, together with a focused analysis on the *very small* and *small* object categories. SR-YOLOv5s achieves a mean *mAP50* of **0.9272 ± 0.0029** , corresponding to a relative improvement of **8.54%** over the baseline. Performance gains are also evident when analyzing size-specific classes: SR-YOLOv5s achieves relative improvements of **64.48%** for *very small* targets and **16.31%** for *small* targets in terms of *mAP50*. These results support the effectiveness of the proposed end-to-end super-resolution strategy, demonstrating that SR-YOLOv5s can significantly enhance detection performance—particularly for low-resolution and small-scale objects.

To statistically validate the effectiveness of SR-YOLOv5s compared to the baseline YOLOv5s, a rigorous evaluation was performed using a bootstrap-based resampling methodology. A total of 100 independent re-sampled test sets were generated from the original *LR-ShipRSImage* test split following a non-parametric bootstrap with replacement. Considering the relatively low standard deviation of *mAP50* across the five folds of cross-validation, a representative pair of models—SR-YOLOv5s and baseline YOLOv5s—trained on the same fold was selected to ensure a controlled statistical comparison, minimizing potential confounding effects due to data partitioning. Based on the 100 bootstrap samples, a paired *t-test* was carried out to compare detection accuracy between the two mod-

els. The analysis confirmed that SR-YOLOv5s significantly outperforms the baseline, achieving a mean $mAP50$ of $93.27\% \pm 0.80\%$ compared to $84.86\% \pm 1.14\%$ ($mean\ difference = +8.40\% \pm 1.32\%$, $p < 0.001$).

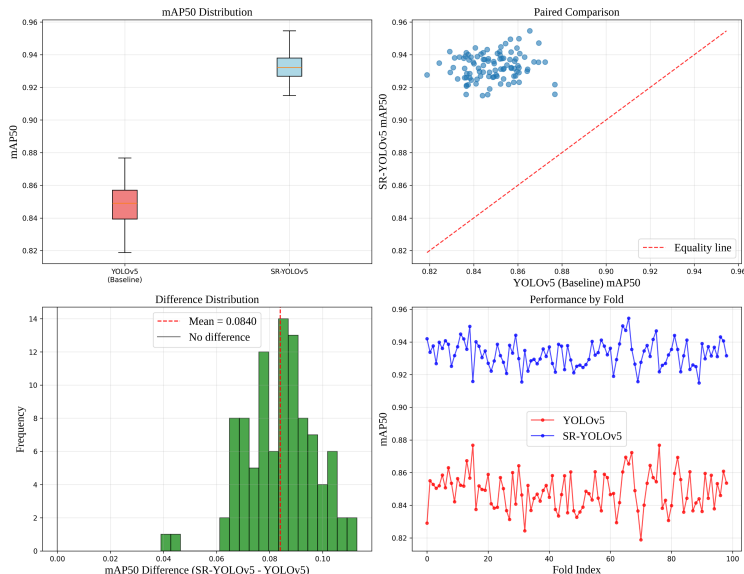


Figure 4.17. Statistical evaluation of SR-YOLOv5s Vs. baseline YOLOv5s

Figure 4.17 summarizes the statistical comparison between SR-YOLOv5s and the YOLOv5s baseline over 100 bootstrap resamples of the test set. The boxplot (top-left) indicates a higher median and reduced dispersion for SR-YOLOv5s; the paired scatter (top-right) lies almost entirely above the equality line, evidencing a systematic advantage on a per-resample basis. The distribution of paired differences (bottom-left) is shifted to the right of zero, with an average increase of approximately 0.084. The per-iteration traces (bottom-right) further confirm the persistence of this gap across resamples. Taken together, these results demonstrate a substantial and robust improvement of SR-YOLOv5s over the baseline.

Table 4.3 reports the detection performance of YOLOv5s on both *HR-ShipRSImage* and *LR-ShipRSImage*, along with results for a *ReLU*-based variant and the proposed *SR-YOLOv5s* model. While the original YOLOv5s achieves the highest accuracy on high-resolution images, per-

formance degrades markedly on low-resolution data, particularly due to the increased prevalence of *very small* and *small* objects. SR-YOLOv5s effectively mitigates this degradation, not only closing the performance gap but also outperforming the ReLU-based YOLOv5s variant on the HR dataset, despite operating on LR inputs. This highlights the ability of the super-resolution backbone to recover critical spatial details.

Table 4.3. Vessel detection accuracy

Model	Input size	Parameters	mAP50	mAP50:95
YOLOv5s (Original)	208	7,012,822	0.854	0.661
YOLOv5s (Original)	416	7,012,822	0.935	0.792
YOLOv5s (ReLU)	416	7,012,822	0.920	0.761
SR-YOLOv5s	208	7,029,782	0.927	0.764

Figure 4.18 presents the confusion matrices obtained on the *HR-ShipRSImage* and *LR-ShipRSImage* test sets, each comprising 1,402 annotated instances of the class *ship*, evaluated using the baseline YOLOv5s and the proposed SR-YOLOv5s models. All evaluations were conducted at an *IoU* threshold of 0.6 and a confidence threshold of 0.4. The results indicate a marked degradation in the number of True Positives (TPs) detected by YOLOv5s when applied to the LR dataset, primarily due to the increased presence of *small* and *very small* targets. This degradation is accompanied by a rise in both False Positives (FPs) – background regions erroneously classified as ships – and False Negatives (FNs), where actual ships fail to be detected.

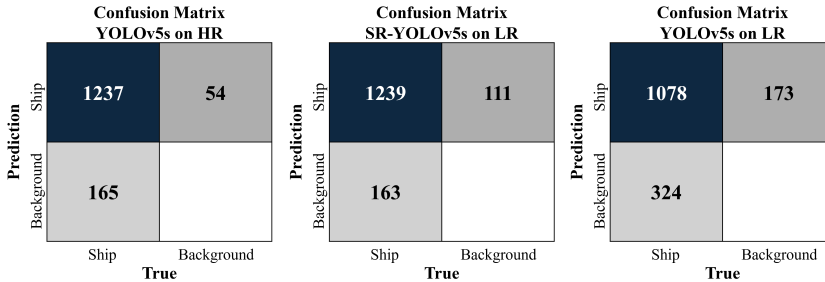


Figure 4.18. Confusion matrices

By incorporating SR-Backbone, SR-YOLOv5s substantially improves detection performance, achieving a higher TP count on the LR dataset than the baseline model achieves on the HR dataset. Additionally, SR-YOLOv5s demonstrates a significant reduction in FP compared to the baseline on the same LR test set, underscoring the effectiveness of the learned super-resolution stage in recovering discriminative spatial details.

Figure 4.19 illustrates a representative case of a FN in the baseline YOLOv5s due to partial cloud occlusion and the corresponding improvement with SR-YOLOv5s. From left to right, the baseline YOLOv5s fails to detect the vessel (false negative), the ground-truth bounding box is shown for reference, and SR-YOLOv5s correctly detects the vessel with high confidence under the same conditions.

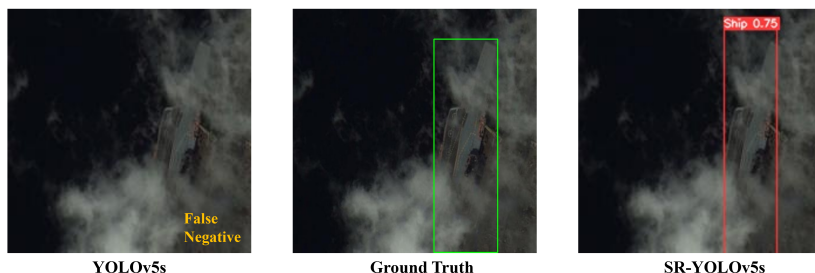


Figure 4.19. Detection results under cloud occlusion

In the right panels of Figure 4.20, the baseline YOLOv5s produces spurious detections where the water–shore interface and man–made structures form high–contrast, repetitive patterns. These textures mimic ship–like cues at low scale, triggering false positives. By contrast, SR-YOLOv5s (left) suppresses such responses: super-resolution reconstructs sharper contours and finer appearance details, allowing the detector to discriminate true targets from background clutter and to reduce the number of false alarms.

In Figure 4.21, several vessels shrink to only a few pixels and become heavily pixelated; the baseline model (right) frequently fails to localize them (false negatives) or assigns very low confidence. Applying super-resolution (left) effectively maps these very small instances from the low-resolution space to a higher-resolution representation, restoring discriminative cues and yielding consistent, higher-confidence detections.



Figure 4.20. FP example

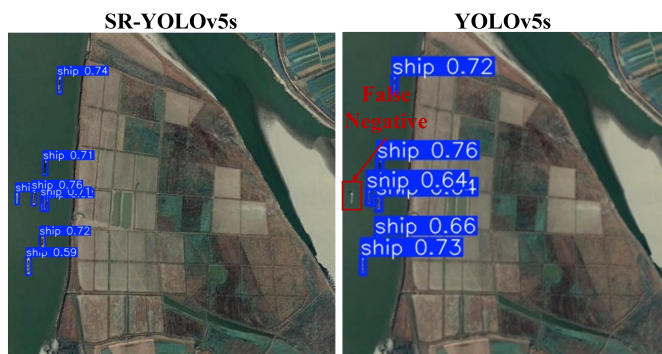


Figure 4.21. FN example

The total number of FPs is inherently sensitive to the selected confidence threshold. As shown in Figure 4.22, reducing the confidence threshold to 0.3 increases the $mAP50$ for SR-YOLOv5s up to 0.936 by allowing more detections to be counted as TP. However, this improvement comes at the cost of a considerable increase in FPs, which may compromise the model’s reliability in alert-triggering applications. Conversely, adopting a more stringent confidence threshold results in a lower number of FPs but leads to an increase in FNs and a consequent reduction in $mAP50$ to approximately 0.89.

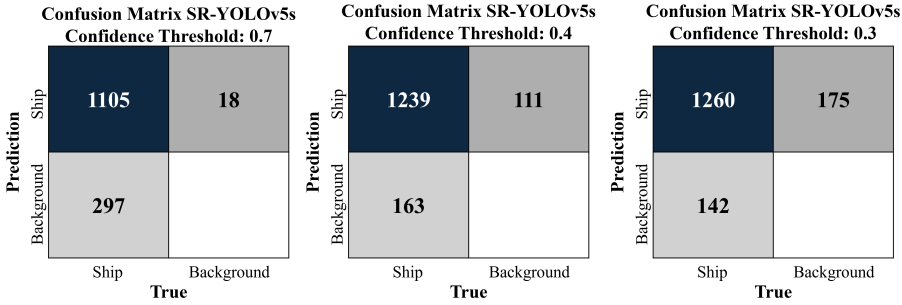


Figure 4.22. SR-YOLOv5s confusion matrices vs. confidence threshold

To further suppress FPs, a lightweight post-classification module based on *ResNet-18* [80] was trained to discriminate between true and false detections produced by SR-YOLOv5s. The model was optimized using *binary cross-entropy loss* with the *Adam optimizer* over 100 epochs. Training samples were extracted from SR-YOLOv5s predictions using a low confidence threshold, and cropped regions were resized to 224×224 while preserving the aspect ratio. The classifier was applied to predictions with confidence scores below 0.5, resulting in a modest improvement in overall detection accuracy to **0.928** ($mAP50$). While the performance gain was limited—potentially due to class imbalance in the training data—these findings suggest that post-hoc binary classification can enhance precision and merits further exploration.

Table 4.4 reports the $mAP50$ results obtained with SR-YOLOv5s and the baseline YOLOv5s on the *LR-CloudyTest* dataset, as well as on the *ship* subset derived from the MASATI dataset. The results demonstrate the ability of SR-YOLOv5s to achieve comparable or slightly better performance

Table 4.4. mAP50 results on LR-CloudyTest and MASATI (Ship)

Model	LR-CloudyTest	MASATI-ship
YOLOv5s	0.610	0.529
SR-YOLOv5s	0.660	0.751

than the baseline on the synthetic cloudy dataset, suggesting that the super-resolution backbone does not introduce semantic artifacts during upsampling. This is likely due to the end-to-end training process, which enables the model to preserve detection-relevant features.

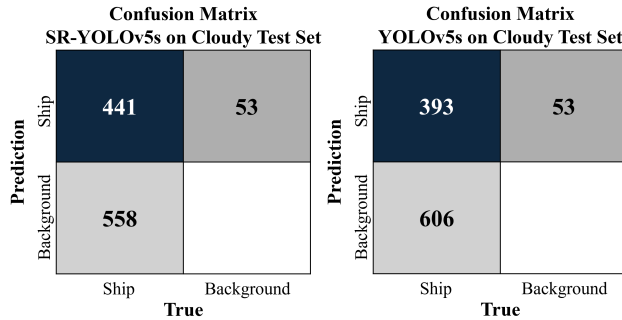
**Figure 4.23.** Confusion matrices on LR-CloudyTest

Figure 4.23 reports, for the *LR-CloudyTest* dataset, the confusion matrices of SR-YOLOv5s and the baseline YOLOv5s, summarizing true/false positives and negatives and indicating the overall reduction of false alarms and missed detections achieved by the super-resolution model. Complementary, Figure 4.24 provides qualitative examples: correct detections (top), a representative false positive (top-right), and challenging false negatives (bottom-right) under heavy cloud occlusion. Taken together, the two figures highlight both the difficulty posed by severe atmospheric degradation and the improved target localization delivered by SR-YOLOv5s.

SR-YOLOv5s outperforms the baseline YOLOv5s also in terms of detection accuracy on the *MASATI-ship* subset, demonstrating the model's ability to generalize to new and heterogeneous operational domains, distinct from the original training distribution. As illustrated in Figure 4.25, the model accurately detects ships under a wide range of visual conditions,

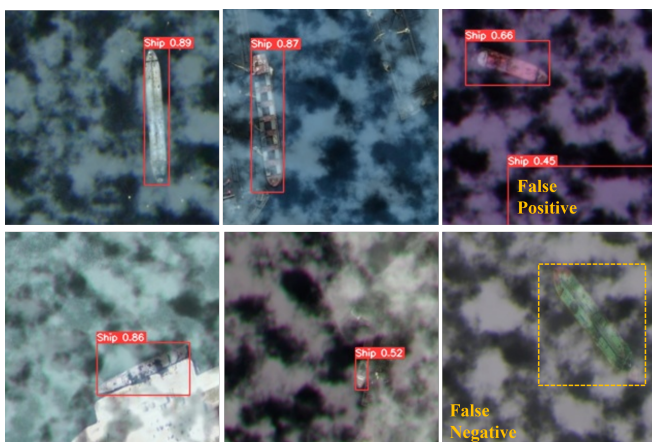


Figure 4.24. SR-YOLOv5s results on LR-CloudyTest

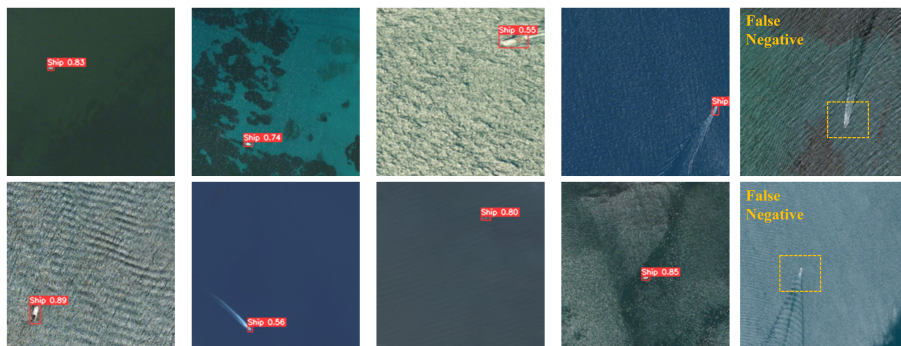


Figure 4.25. SR-YOLOv5s results on *MASATI-ship* subset

including variations in water texture, color, background clutter, and wake patterns, while also minimizing false positives and reducing the number of false negatives compared to the baseline. Specifically, red boxes indicate SR-YOLOv5s predictions (with confidence scores); yellow dashed boxes highlight ground-truth ship instances missed by the detector (false negatives).

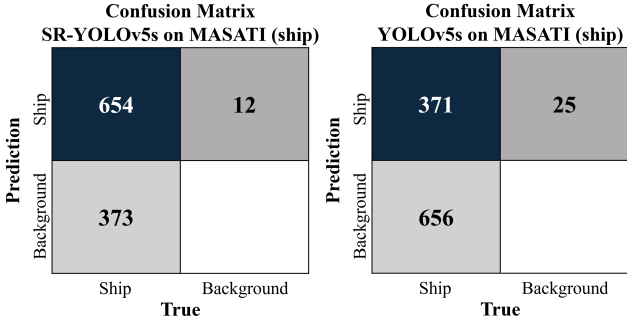


Figure 4.26. Confusion matrices on *MASATI-ship* subset

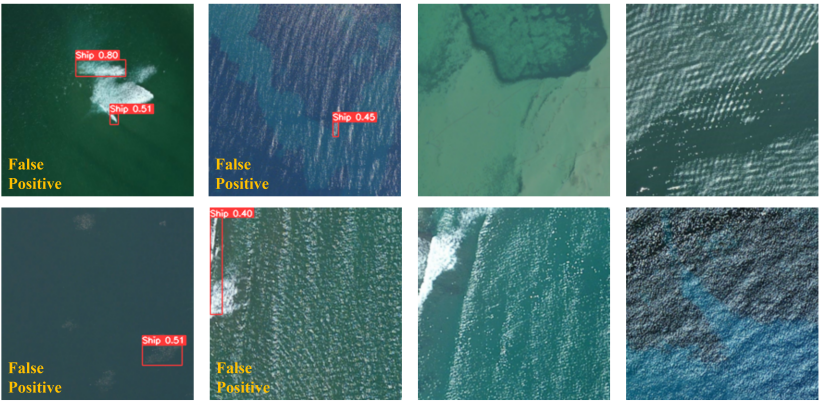


Figure 4.27. SR-YOLOv5s results on *MASATI-water* subset

This finding is further supported by the confusion matrices shown in Figure 4.26, where SR-YOLOv5s exhibits a higher true positive rate and a lower number of false positives compared to the baseline model. (Left:

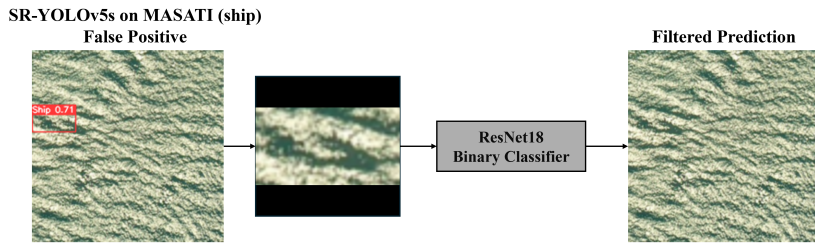


Figure 4.28. Binary classification on SR-YOLOv5s detections.

YOLOv5s baseline; Right: SR-YOLOv5s.)

Finally, SR-YOLOv5s was evaluated on the *MASATI-water* subset, which contains no ship instances, to assess the impact of FPs relative to the baseline. Out of 1,022 images, SR-YOLOv5s produced 32 false detections, improving upon the 42 FPs generated by the baseline YOLOv5s. When the binary classifier was applied to low-confidence predictions, the number of FPs was further reduced to 20. Figure 4.27 shows representative examples of FPs and challenging background textures that caused incorrect detections.

Figure 4.28 illustrates the binary post-classification applied to an FP. A false positive generated by SR-YOLOv5s on *MASATI-ship* is cropped, resized to 224×224 , and passed to a *ResNet-18* binary classifier. The classifier correctly rejects the candidate (*no-ship*), thereby filtering spurious detections induced by strongly textured sea backgrounds.

4.3.2 Hardware Deployment

Table 4.5 compares the performance of the SR-YOLOv5s model deployed on the *NVIDIA A100 GPU*, the *embedded NVIDIA Jetson Orin Nano GPU*, and the *PolarFire SoC Video Kit*. The evaluation focuses on key performance metrics, including accuracy ($mAP50$), inference speed, and estimated power and energy consumption. In addition, we introduce the ratio of detection accuracy to power consumption as a comparative metric. This provides further insight into the efficiency of each hardware implementation, emphasizing not only absolute detection performance but also how effectively computational resources and available energy are utilized.

The assessment of the average and maximum power demand of the SR-YOLOv5s algorithm was conducted by executing the inference process 200 times on a single input image. This approach allows the accurate determination of the mean values for both metrics, focusing exclusively on the inference phase.

Despite the quantization process, the detection performance of the SR-YOLOv5s on the PolarFire SoC closely aligns with that obtained at full precision, exhibiting only a marginal reduction in accuracy. However, the quantization performed by the TensorRT library introduces a slightly more pronounced decrease in accuracy compared to that produced by the VectorBlox Quantizer. This indicates a superior capability of the quantization tool provided by Microchip for optimizing the model relative to its TensorRT counterpart.

Table 4.5. Comparative benchmark of AI inference engines

Device	Prec.	mAP50	Inference speed	Avg. power consumption	Avg. energy per frame	mAP50/W
A100	FP32	0.927	7.30 ms	53.04 W	384.2 mJ/frame	0.017
Jetson Orin Nano (15 W)	FP32	0.927	20.80 ms	10.50 W	218.4 mJ/frame	0.087
Jetson Orin Nano (15 W)	FP16	0.916	18.07 ms	10.50 W	189.7 mJ/frame	0.087
Jetson Orin Nano (15 W)	INT8	0.843	16.00 ms	6.51 W	104.2 mJ/frame	0.127
Jetson Orin Nano (7 W)	INT8	0.843	23.54 ms	6.30 W	148.3 mJ/frame	0.132
PolarFire SoC	INT8	0.916	55.00 ms	2.46 W	135.3 mJ/frame	0.366

Although model pruning was considered in this work, only an *unstructured pruning* operation was applied, with the objective of evaluating its impact on the model's accuracy rather than on computational performance. In unstructured pruning, individual weights are set to zero without altering the original dimensionality of the tensors. As a result, when the model is

deployed on hardware accelerators such as the [CoreVBX](#), the convolution operations are still computed over the full parameter matrices, including zero-valued weights. This implies that the computational workload remains effectively unchanged, since the execution units do not inherently skip over pruned weights unless explicitly designed to exploit sparsity. Consequently, the applied pruning strategy does not yield a significant measurable speedup on the target architecture.

Nevertheless, experimental results on the *LR-ShipRSImage* test set show that unstructured pruning of 30% of the convolutional filters leads to a limited drop in accuracy, with *mAP50* decreasing to 0.885. When the pruning ratio is increased to 50%, performance degrades significantly, with *mAP50* falling to 0.511. However, retraining the pruned model (30%) for 100 epochs enables recovery of the original performance, with *mAP50* reaching 0.923. This result highlights the potential of pruning as an effective compression strategy, provided that a fine-tuning phase is included after pruning. It also motivates the adoption of structured pruning techniques that can effectively translate into reduced computational complexity and improved runtime efficiency when deployed on hardware accelerators. Future work will therefore explore the integration of structured pruning and subsequent model compilation using the VectorBlox SDK, in order to fully exploit the computational advantages of the [CoreVBX](#) architecture.

The NVIDIA A100 was evaluated exclusively as a cloud-hosted resource on Google Colab, without physical access for direct instrumentation. Consequently, power consumption was estimated using the NVIDIA Management Library ([NVML](#)) [100, 152] library through the *pynvml* Python interface. [NVML](#) provides telemetry from on-board sensors, allowing access to power registers derived from current and voltage measurements to generate power traces. The obtained estimates indicate an average power consumption of approximately 53 W, with observed peaks reaching 55.6 W. Since NVIDIA does not disclose details regarding sensor sampling rates or filtering, the reported measurements may not be fully synchronized with kernel execution, potentially leading to a slight underestimation of both average and peak power values.

For the PolarFire [SoC](#), a hybrid power evaluation methodology was employed, combining Microchip’s *SmartPower Analyze* [136] with direct

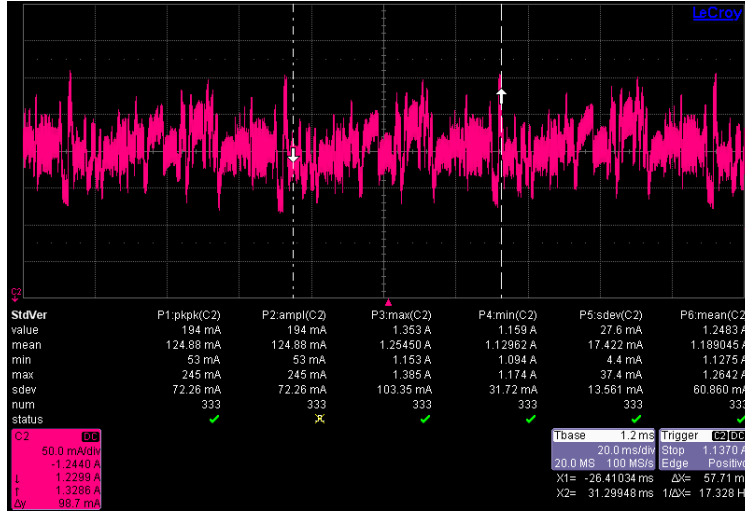


Figure 4.29. SR-YOLOv5s power profile on PolarFire SoC

dynamic power measurements. SmartPower, integrated within the *Libero Design Suite* [90], provided a static power estimate of approximately 1.3 W for the target design (excluding DDR idle power). Dynamic power during inference was measured on the *PolarFire SoC Video Kit* by executing the SR-YOLOv5s application while monitoring board current using a *LeCroy WavePro 7100 digital oscilloscope* [107] and a current probe. The test application loads a JPEG image from non-volatile memory, configures the *CoreVBX* accelerator, and performs 200 consecutive inference iterations on the same image, subsequently printing detected bounding boxes to the serial terminal.

Figure 4.29 depicts the measured power profile of the PolarFire SoC Video Kit during testing. The dynamic power during inference—i.e., the increment over idle—averages 1.16 W, with instantaneous peaks up to 2.6 W. The total average power reported in Table 4.5 is obtained by adding the SmartPower-estimated static component (1.3 W) to the measured dynamic component, yielding 2.46 W.

The Jetson Orin Nano’s power consumption was evaluated by executing SR-YOLOv5s under three numerical precision modes—FP32, FP16, and INT8—within the 15 W power configuration, and additionally assessing

INT8 performance in the 7 W mode. INT8 quantization was carried out using NVIDIA TensorRT [151] through a post-training (dynamic) quantization workflow. The validation set was employed as the calibration dataset, consistent with the dataset used for VectorBlox SDK calibration, to ensure methodological alignment across toolchains.

Static (idle) and dynamic (inference) power were characterized using the on-board three-channel INA3221 power monitor [153]. Dynamic power was estimated by repeatedly executing inference and measuring the incremental current draw relative to the idle state, following the same protocol used for the PolarFire SoC Video Kit.

In FP32, the Orin Nano drew an average power of approximately 10.5 W, with peaks around 22.8 W. Switching to FP16 reduced latency while maintaining a similar average power level, thus improving efficiency with only a minor accuracy reduction. Under INT8, average power decreased to about 6.5 W in the 15 W mode and 6.3 W in the 7 W mode, with corresponding peak values of roughly 12.6 W and 12.0 W, respectively. These results highlight the expected latency–accuracy–energy trade-offs across precision representations and demonstrate the efficiency gains achievable with TensorRT-based post-training quantization on the Jetson Orin Nano.

CoreVBX executes inference at approximately **18 FPS**, with a notably lower average power consumption than both the NVIDIA A100 GPUs and the embedded Jetson Orin Nano GPU for floating-point and INT8 inference. GPUs typically operate at very high clock frequencies (1–2 GHz or higher) and sustain billions of operations per second thanks to massive parallelism and high-bandwidth memory interfaces; however, this performance comes at the cost of increased power consumption. By contrast, FPGAs operate at lower clock frequencies—200 MHz in our design. Although quantization reduces model complexity and the programmable logic accelerates neural-network computations, the lower clock rate inherently limits operations per unit time, leading to longer inference latency. Consequently, CoreVBX exhibits slightly higher energy per frame than the Jetson Orin Nano operating in 15 W mode with INT8 inference. Conversely, when the Jetson Orin Nano is set to 7 W mode, reduced tensor-core availability and lower GPU frequency decrease computational throughput and extend inference time, yielding inferior performance to the PolarFire SoC in both power and energy consumption. Overall, the PolarFire SoC achieves superior

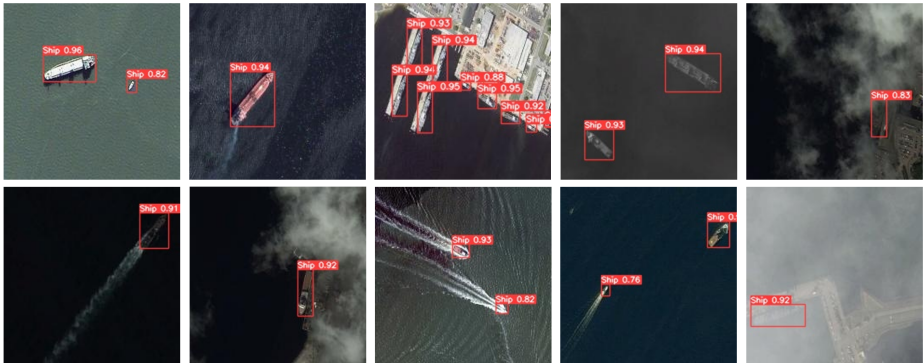


Figure 4.30. SR-YOLOv5s ship detections on PolarFire SoC Video Kit

efficiency in terms of accuracy per unit power ($mAP50/power = 0.366$).

Figure 4.30 presents representative detection outputs produced by SR-YOLOv5s deployed on the PolarFire SoC, spanning diverse operational scenarios including open sea, harbors, partial cloud occlusions, and varying illumination conditions. The model yields high-confidence predictions with accurate localization, even under degraded visual conditions, underscoring its robustness and generalization capability.

ANHEO for Autonomous Navigation

In Chapter 4, the PolarFire [FPGA](#) is proposed as a computational platform for deploying deep-learning models directly onboard spacecraft. Specifically, a novel model named *SR-YOLOv5s*, designed for small vessel detection, is implemented to demonstrate inference speed and power efficiency suitable for near-real-time information extraction from satellite imagery under strict power constraints. The availability of the radiation-tolerant PolarFire [FPGA](#) further enables the synthesis of the design into hardware suitable for more demanding space missions. This chapter also highlights the additional benefits of employing [FPGAs](#) for onboard [AI](#) acceleration. Accelerators based on overlay architectures—such as [CoreVBX](#)—implement a general-purpose processing engine within the [FPGA](#) fabric, allowing multiple neural networks to operate on the same hardware through software-level updates without the need for bitstream reloading. As discussed in Chapter 3, the capability to reconfigure programmable logic without any physical hardware modification enables post-deployment functional upgrades, including the addition of new [NN](#) operators or even the implementation of entirely new accelerator architectures.

From a lifecycle perspective, this significantly reduces mission costs by allowing firmware-based adaptation to evolving computational requirements—particularly valuable for long-duration missions where objectives and algorithms may change over time. As reported in [165], [RT](#) PolarFire

devices support up to 500 in-orbit reprogramming cycles, ensuring sufficient endurance for mission-critical updates throughout a satellite’s operational lifetime.

In the context of space systems, this dual-level flexibility enables both model updates—deploying refined networks trained on newly acquired data—and strategic hardware evolution through in-flight bitstream replacement. Figure 5.1 summarizes these advantages, illustrating the model-level flexibility provided by overlay architectures and the hardware-level reconfigurability achieved through bitstream updates, which together enable continuous adaptation of onboard AI capabilities during the mission.

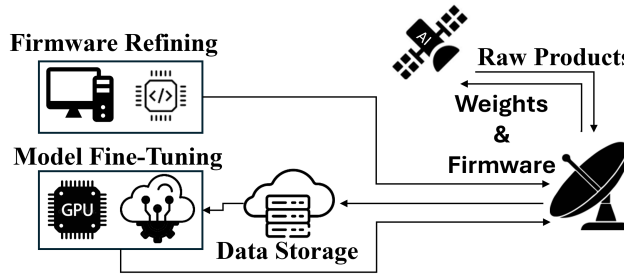


Figure 5.1. FPGA deployment advantages: model-level flexibility and hardware-level reconfigurability.

A representative implementation of this paradigm is the [ANHEO](#) unit, which leverages reconfiguration capabilities to instantiate custom accelerators on demand. The following sections describe the [ANHEO](#) system architecture, the vision-based relative navigation pipeline implemented, and a case study demonstrating the practical benefits of bitstream reconfigurability through the upgrade from [CoreVBX](#) version 1 to version 2 for optimized satellite pose-estimation performance.

5.1 ANHEO System Overview

The [ANHEO](#) system is a highly integrated navigation unit designed for autonomous *absolute* and *relative* navigation of nano- and microsatellites across orbits extending from [LEO](#) to lunar altitudes. Developed under the lead of [TSD-Space](#), in collaboration with the University of Naples

Federico II and the Italian Center for Aerospace Research (CIRA), and funded by the [ASI](#), the system has reached [TRL 4](#) through an elegant BreadBoard ([B/B](#)) demonstration unit.

Designed for integration within 6U–12U CubeSat platforms, [ANHEO](#) features an extremely compact 0.8 U form factor ($95 \times 95 \times 80 \text{ mm}^3$) and a total mass below 0.8 kg. Power consumption is limited to approximately 10 W. Figure 5.2 shows the elegant [B/B](#) demonstration unit of the [ANHEO](#) system.



Figure 5.2. ANHEO elegant B/B

5.1.1 System Architecture

The [ANHEO](#) architecture comprises five interconnected modules, as illustrated in Figure 5.3: the Data Handling Main Board ([DHMB](#)), the Data Handling Daughter Board ([DHDB](#)), the Hardware Accelerator Board ([HA](#)), the Power Conditioning & Distribution Module ([PCDM](#)), and the I/O Interface Board ([IOB](#)).

All subsystems are physically arranged and interconnected to optimize volume utilization and maximize component density, allowing the entire system to fit within a compact form factor. The [DHMB](#) and [IOB](#) are connected through a flexible [PCB](#), which makes it possible to orient the two rigid sections and maintain a precise spacing between them — a configuration that would not be achievable using conventional inter-board connectors. The remaining boards are interconnected via high-density,

high-data-rate micro-miniature board-to-board connectors, ensuring both electrical reliability and robust mechanical integration.

This compact integration allows the entire system to fit within the 0.8 U form factor (1U: $10 \times 10 \times 10$ cm).

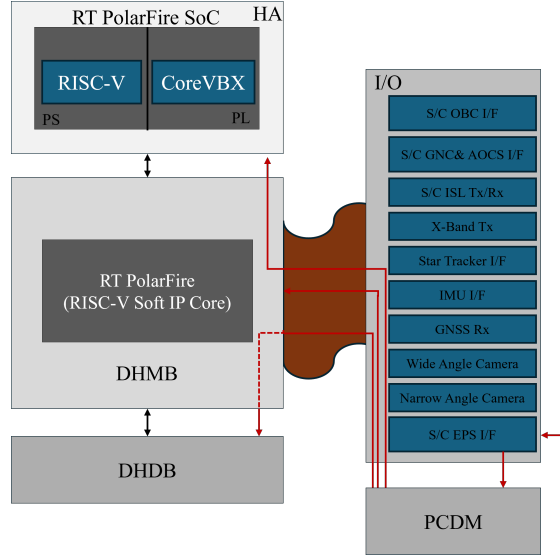


Figure 5.3. ANHEO system architecture

The system provides comprehensive interfaces to the spacecraft subsystems: the **OBC**, the Attitude and Orbit Control System (**AOCS**)/**GNC** subsystem (sensors and actuators), an external **IMU**, a **GNSS** receiver, a Star Tracker (**STR**), up to two cameras, an inter-satellite-link (Inter-Satellite Link (**ISL**)) transceiver for cooperative navigation data exchange, and the payload data transmitter for image downlink.

Functionally, **ANHEO** integrates data from multiple onboard sensors to determine both the satellite's orientation and its position in space with high precision. It combines measurements from a **GNSS** receiver, an **IMU**, and a star tracker to continuously estimate the satellite's attitude and orbit.

For operations involving another spacecraft, the system performs relative navigation by processing visual data acquired by one or two monocular cameras. A wide-field camera provides situational awareness and long-range detection of the target, while a narrow-field camera enables high-precision

tracking and pose estimation at closer distances.

When the target is cooperative—meaning it also carries GNSS and inter-satellite link (ISL) capabilities—the system fuses visual data with dual-receiver GNSS measurements to improve robustness in degraded conditions (such as weak signals, temporary occlusions, or multipath) and to accelerate carrier-phase integer ambiguity resolution, thereby enhancing ranging accuracy.

In non-cooperative cases, where the target lacks communication or navigation aids, AI-based computer-vision algorithms are employed to estimate its relative pose directly from camera images. The final outputs, including absolute and relative position, velocity, and orientation, are delivered together with uncertainty metrics and health flags for integration within the spacecraft’s AOCS and GNC systems.

5.1.2 Data Handling and Processing Modules

The DHMB constitutes the core processing subsystem, hosting a radiation-tolerant PolarFire FPGA. This non-volatile, flash-based 28 nm device provides 481 k logic elements, 33 Mbit embedded RAM, 1480 DSP blocks, and 24 multi-gigabit transceivers supporting line rates up to 10 Gbit s^{-1} . As analyzed in Chapter 3, compared to SRAM-based alternatives, the RT PolarFire family exhibits approximately 50% lower power consumption and near-immunity to configuration SEU [217].

Within the programmable logic, a soft-IP RISC-V CPU [130] executes the application software and coordinates dedicated VHDL modules for high-throughput I/O and control. The DHMB receives commands from the OBC, performs pre-processing of measurements from GNSS, IMU, STR, and ISL sources, manages video streams from wide-angle and narrow-angle cameras (including pixel equalization and compression), and orchestrates activation of the HA for computationally intensive AI-based tasks.

The HA leverages the RT PolarFire SoC, as shown in Figure 5.4. The subsystem features a compact PCB design integrating the SoC with 2 GB DDR4 memory for the RISC-V hard-core microprocessor and an additional 4 GB DDR4 memory dedicated to the FPGA fabric, ensuring sufficient memory capacity to support execution of high computational workloads. This architecture enables execution of complex algorithms for autonomous navigation, including vision-based navigation powered by the CoreVBX

neural network accelerator.

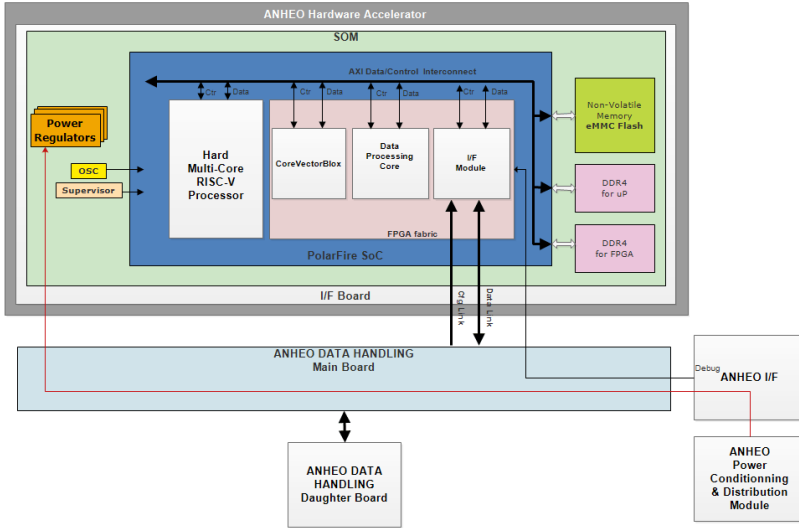


Figure 5.4. ANHEO hardware accelerator

5.2 Vision-based Relative Navigation

The ANHEO architecture incorporates monocular camera sensing to enable autonomous relative navigation during proximity operations with uncooperative but geometrically known target satellites. The objective is to provide real-time estimates of the target's relative translational and rotational state parameters with respect to the approaching satellite. The proposed method leverages CNNs to extract pose parameters—relative position and attitude—from monocular imagery through a multi-stage processing pipeline.

5.2.1 Neural Network Pipeline

The processing pipeline, illustrated in Figure 5.5, employs two state-of-the-art CNNs performing complementary tasks. The vision-based relative pose estimation algorithm was developed by the Department of Aerospace

Engineering at the University of Naples Federico II as part of the ANHEO project, and is described in detail in [145]. Images of the target satellite are acquired by the sensor and processed sequentially through detection and keypoint localization stages before pose estimation and state filtering.

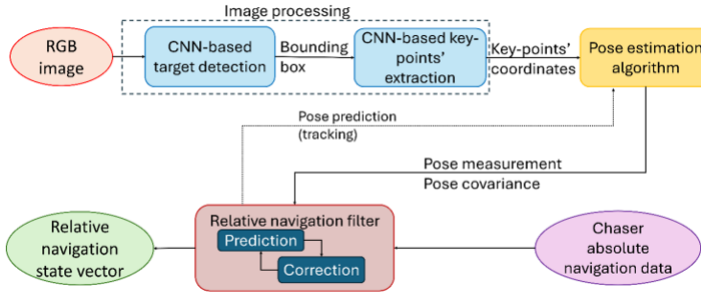


Figure 5.5. Relative pose estimation pipeline [145]

First, a modified *YOLOv5su* network [213] trained on synthetic data performs target detection, outputting the bounding box enclosing the satellite along with an associated confidence score. The cropped image region is subsequently processed by a *YOLOv8s-pose* network [216], also trained on synthetic data, to detect and localize 33 natural keypoints on the ENVISAT satellite surface (assumed as the representative target). These keypoints include corners of the main body, solar panels, antenna structures, and geometric shape centers. The network outputs 2D coordinates for each keypoint along with visibility flags and confidence scores, enabling geometric reconstruction even for partially occluded features.

Pose parameters are estimated by mapping filtered 2D keypoint coordinates (selected via confidence thresholding) to their corresponding known 3D positions in the target body-fixed reference frame. The estimation employs the *Efficient Perspective-n-Point* (EPnP) algorithm for initialization and *Levenberg-Marquardt* optimization when a pose prior is available. Finally, the resulting pose measurements are integrated into a loosely-coupled *Multiplicative Extended Kalman Filter* (MEKF) to estimate relative position, velocity, attitude, and angular velocity between the two satellites.

5.2.2 YOLOv5su Architecture

YOLOv5u represents an enhanced evolution of the YOLOv5 architecture (outlined in Section 4.1.2), characterized by a *unified*, anchor-free detection strategy. The “u” suffix denotes the removal of predefined anchor boxes while preserving comparable detection accuracy. Although the network supports both object detection and instance segmentation, within the proposed pipeline it is configured solely for object detection, providing bounding boxes and confidence scores for target identification.

Structurally, the model retains the core components of YOLOv5: a CSPDarknet53 backbone for feature extraction, a multi-scale feature fusion neck combining PANet and FPN elements, and non-maximum suppression (NMS) for post-processing. Its main innovation lies in the anchor-free, objectness-free split head—a design paradigm first introduced in YOLOv8—which refines the trade-off between accuracy and speed.

Traditional anchor-based detectors depend on predefined bounding box templates to guide predictions, requiring dataset-specific anchor tuning through clustering algorithms (e.g., K-Means) and extensive hyperparameter optimization. While effective, increasing the number of anchors raises computational costs and limits generalization across datasets [104]. YOLOv5u overcomes these constraints by directly regressing bounding box coordinates and learning object presence probabilities end-to-end. This approach simplifies the training process, reduces hyperparameter sensitivity, and improves generalization to targets with diverse geometries. Empirical evidence shows that anchor-free detectors achieve comparable or superior accuracy while offering greater architectural flexibility [104].

Table 5.1 summarizes the comparative performance of YOLOv5s and YOLOv5su on the COCO dataset. YOLOv5su achieves a marked improvement in accuracy (mAP₅₀₋₉₅ increasing from 37.4 to 43.0, a +14.9% relative gain) with moderate added complexity (+26.4% parameters, +45.5% FLOPs). Inference latency rises by approximately 23% on CPU and shows a similar increase on GPU (from 0.9 ms to 1.27 ms on an A100 with TensorRT). Despite the higher computational demand, the resulting performance boost is particularly valuable for satellite proximity operations, where robust and accurate localization directly affects mission safety and navigation precision. Moreover, the anchor-free design enhances adaptability across varying target aspect ratios and observation distances, eliminating

the need for dataset-specific anchor tuning.

Considering these factors, YOLOv5su was selected as the detection backbone, offering a favorable balance between accuracy, robustness, and computational efficiency—well suited for hardware acceleration on the RT PolarFire FPGA platform.

Table 5.1. YOLOv5s and YOLOv5su Architectures Comparison

Model	Input Size (pixels)	mAP ^{val} 50-95	mAP ^{val} 50	Inference Speed (ms)		Params (M)	FLOPs (B)	Ref.
				CPU	A100			
YOLOv5s	640	37.4	56.8	98	0.9 [†]	7.2	16.5	[214]
YOLOv5su	640	43.0	—	120.7 [‡]	1.27 [§]	9.1	24.0	[215]

[†]V100 GPU, batch size 32. [‡]ONNX Runtime on CPU. [§]TensorRT on A100.

5.3 Neural Network Deployment Strategy

Implementing the pose estimation pipeline on ANHEO required hardware acceleration of both the YOLOv5su and YOLOv8s-pose models. This section focuses on the deployment of YOLOv5su, describing the transition from an initial hardware/software co-design to a fully hardware-accelerated implementation enabled through FPGA bitstream reconfiguration.

5.3.1 Initial Implementation with CoreVectorBlox v1

The first B/B prototype was developed using CoreVBX version 1, which did not support several neural network layers and activation functions required by the original YOLOv5su architecture. In particular, the SiLU activation function introduced by Ultralytics was unsupported, and had to be replaced with ReLU (as detailed in Section 4.2.2). Moreover, the concatenation operation within the prediction head—responsible for merging feature maps across three scales along axis 2—was incompatible with the available CoreVBX hardware primitives.

To overcome these limitations, a *hardware/software co-design* approach was adopted. The model was first converted from PyTorch to ONNX format, with only the backbone and neck layers mapped for hardware execution. The CoreVBX accelerator was configured to perform feature extraction (backbone) and multi-scale feature fusion (neck), producing

three feature maps—P3, P4, and P5—with spatial dimensions of 80×80 , 40×40 , and 20×20 , respectively. These maps were optimized for detecting objects at different scales.

The prediction head and post-processing stages were instead executed in software on the RISC-V multiprocessor, handling confidence scoring, bounding-box regression, thresholding, and non-maximum suppression (NMS) to generate final detections. Figure 5.6 illustrates this partitioned workflow, highlighting the division between the hardware-accelerated layers (backbone and neck on CoreVBX) and the software-executed components (prediction head and NMS on RISC-V). Although this hybrid design enabled full functionality despite hardware constraints, it incurred significant performance penalties due to software processing overhead.

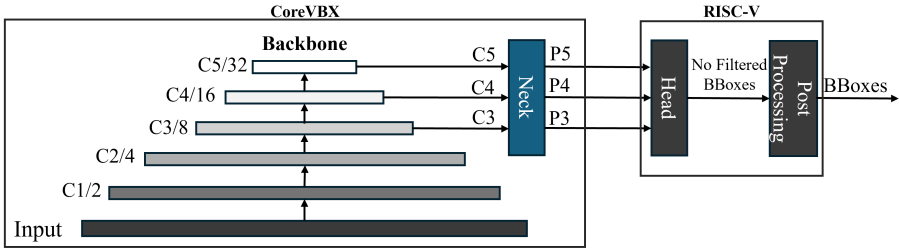


Figure 5.6. YOLOv5su inference workflow

5.3.2 Bitstream Reconfiguration

Microchip’s subsequent release of CoreVBX version 2 introduced extended support for additional layer types and activation functions, enabling full hardware acceleration of modern CNN architectures. To exploit these enhancements, a revised ANHEO implementation was developed, integrating the latest CoreVBX [89] to accelerate the entire YOLOv5su pipeline without any software fallback. This upgrade delivers substantial improvements in inference throughput—critical for real-time relative navigation—while significantly reducing per-frame energy consumption, thereby meeting CubeSat power budget constraints.

The upgrade to version 2 was done solely via FPGA bitstream reconfiguration, with no changes to the acceleration board. This underscores

the flexibility and field-upgradability of [FPGA](#) platforms—crucial in space, where hardware cannot be replaced and in-flight updates are needed to adapt or fix anomalies.

CoreVectorBlox v2 Architecture

Both [CoreVBX](#) releases share the same overlay concept: a single hardware instantiation executes multiple neural networks through a soft [MXP](#) processor and a dedicated [CNN](#) accelerator, orchestrated by an internal RISC-V *Mi-V microcontroller* that parses the network [BLOB](#). While the three standard configurations (V250, V500, and V1000) remain available, version 2 introduces several architectural refinements. As summarized in [Table 5.2](#), vector processor widths were reduced (from 128-bit to 64-bit for V250, and from 256-bit to 128-bit for V500), and the [CNN](#) accelerator array dimensions were scaled down (16×16 to 16×8 for V250, 16×32 to 16×16 for V500, and 32×32 to 32×16 for V1000).

Table 5.2. CoreVectorBlox Processor Size Configuration Details

Config.	Vector Proc. Width	Vector Scratch.	CNN Accel. Array Size
V250	64-bit	64 kB	16×8
V500	128-bit	128 kB	16×16
V1000	256-bit	256 kB	32×16

Collectively, these architectural refinements reflect Microchip’s design philosophy of extending hardware compatibility to a broader range of modern network operations while maintaining a constrained logic footprint. To accommodate new layer types and activation functions, the convolution pipeline was moderately simplified, resulting in slightly lower peak efficiency but significantly higher deployment flexibility across heterogeneous [CNN](#) architectures.

The upgrade also represents a major step forward in the modernization of the software toolchain. In SDK 1.x, OpenVINO’s infrastructure was employed to convert networks from various frameworks (TensorFlow, Caffe, PyTorch, and [ONNX](#)) into the OpenVINO Intermediate Representation, subsequently processed through `generate_vnnx` to produce [BLOBs](#). This workflow required managing both firmware and network-specific [BLOBs](#),

with mandatory firmware reloading after each accelerator reset.

In contrast, SDK 2.x introduces a streamlined conversion pipeline based on TensorFlow Lite, replacing the dual-BLOB architecture with a single unified network BLOB. This simplification greatly improves deployment efficiency and runtime flexibility: firmware reloads are no longer needed, memory management is simplified, and model switching can be performed dynamically. Such improvements considerably enhance maintainability and make the framework particularly well-suited for multi-model applications that benefit from in-field reconfiguration.

CoreVectorBlox Comparison

Table 5.3 compares the two CoreVBX versions in terms of inference speed and power consumption when accelerating the same model. The reduced 2D array of PEs and the narrower vector-processing width in the newer CoreVBX architecture lead to a modest decrease in peak throughput (cf. Release 1 vs. Release 2). Specifically, for *YOLOv5s-ReLU*, the end-to-end inference time on CoreVBX 2 is only *slightly lower* than on CoreVBX 1, despite the more recent design.

This behaviour can be attributed to the additional hardware support for new functionalities and operators, which introduces extra computational complexity and scheduling overhead. Accordingly, the measured dynamic power increases slightly (from 1.16 W to 1.32 W), highlighting a trade-off between energy efficiency and architectural flexibility. In this sense, CoreVBX 2 is less specialised for certain operations than its predecessor, but compensates by offering native acceleration for a broader range of neural-network layers and models, thereby expanding the set of topologies that can be efficiently deployed and optimised.

Table 5.3. Performance Comparison of YOLOv5s-ReLU Across CoreVectorBlox Versions

Model	Accel. Core	CNN Speed	Power Consumption
YOLOv5s – ReLU	CoreVBX 1	~55.00 ms	1.16 W
YOLOv5s – ReLU	CoreVBX 2	~59.56 ms	1.32 W

Additionally, Table 5.4 reports the benchmarking results for the YOLOv5 nano model and its enhanced variant (YOLOv5nu), comparing

the performance of [CoreVBX](#) versions 1 and 2 to evaluate their computational efficiency, while also highlighting the latency introduced by the post-processing stage.

Both models were pre-trained on the COCO dataset containing 80 object classes. CoreVectorBlox v2 introduces support for advanced architectural features, including the [SiLU](#) activation function, which was not natively supported in the instruction set of the first release. This added flexibility enables the deployment of modern model architectures optimised for higher accuracy, but also introduces non-negligible performance implications when executing networks not specifically tuned for the new hardware constraints.

The performance degradation observed in the post-processing of YOLOv5s with the [SiLU](#) activation function—compared to its [ReLU](#)-based counterpart—stems from fundamental differences between the two activations. As shown in [Figure 5.7](#), [ReLU](#) features a sharp discontinuity at zero, completely suppressing negative inputs and yielding sparse activations.

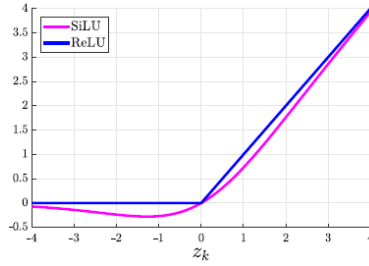


Figure 5.7. Comparison between [SiLU](#) and [ReLU](#) activation functions

Conversely, [SiLU](#) is a smooth, continuously differentiable activation function characterised by a gentle, non-monotonic curve that allows small negative outputs and gradual activation around the origin. Unlike [ReLU](#), which abruptly zeros out all negative inputs, [SiLU](#) preserves a low-level response even for slightly negative values. This smoother behaviour reduces sparsity in intermediate feature maps, leading to denser activation patterns across the network. Consequently, [SiLU](#)-based models tend to produce a larger number of detection candidates whose confidence scores exceed the predefined threshold during inference.

Since the computational cost of the [NMS](#) algorithm scales approximately with the square of the number of input boxes ($O(N^2)$) in its naive form,

or $O(N \log N)$ in optimised batched variants), an increase in the number of candidates N has a super-linear impact on processing time [44, 28]. In practice, even a twofold increase in candidate detections can lead to a post-processing latency greater than $2\times$ due to the pairwise overlap computations required between bounding boxes.

The INT8 quantisation used in the VectorBlox toolflow further amplifies this effect: quantising the smooth SiLU response curve, together with the associated calibration shifts in activation distributions, often results in overconfident objectness and class predictions. Consequently, additional borderline detections surpass the confidence threshold, further increasing the post-processing workload. Implementing Quantisation-Aware Training [94]—in which quantisation effects are explicitly modelled during training through simulated quantisation in the forward pass—could mitigate this issue by making weight distributions more robust to quantisation noise. However, the current CoreVectorBlox SDK does not yet support this feature.

The YOLOv5nu variant, fully accelerated on CoreVectorBlox v2 with native SiLU support, exhibits higher CNN inference latency (~ 126.78 ms for YOLOv5nu-SiLU and ~ 51.6 ms for YOLOv5n-SiLU versus ~ 20.9 ms for YOLOv5n-ReLU). This difference arises primarily from two factors: (i) YOLOv5nu incorporates additional trainable parameters, increasing the multiply-accumulate workload; and (ii) CoreVBX v2 supports new operations, such as SiLU activations and concatenations along axis 2, that were previously executed in software. The computational complexity of SiLU ($x \cdot \sigma(x)$) is inherently higher than that of ReLU ($\max(0, x)$), particularly in INT8 implementations that rely on lookup tables or polynomial approximations for the sigmoid component [84].

Despite the higher computational cost of YOLOv5nu, the architectural evolution yields substantial improvements in post-processing efficiency. The anchor-free detection head fundamentally redefines the computational paradigm: while the anchor-based YOLOv5n generates roughly three candidate boxes per grid cell (corresponding to three anchor templates per feature-pyramid scale), the anchor-free “u” head produces only one candidate per cell, reducing the decoding workload by about $3\times$.

This reduction cascades through the entire post-processing pipeline, producing cumulative benefits. With fewer outputs to decode, redundant

operations—such as sigmoid evaluation, grid offset, and anchor transformations—along with associated INT8→FP32 conversions, are avoided. Consequently, the NMS stage processes a much smaller candidate set, leading to faster overall execution. The anchor-free architecture simplifies the decoding process itself by removing complex anchor-to-grid transformations and reducing temporary memory usage. Finally, the updated loss functions and detection head design in Ultralytics’ “u” variant yield better-calibrated confidence scores, which help suppress false positives near the decision threshold and further reduce the NMS computational load.

Notably, while SiLU activations tend to increase candidate counts in conventional anchor-based architectures, the anchor-free head design dominates this effect, resulting in an observed post-processing latency of 629.6 ms—approximately a 3× improvement compared to the 1908.6 ms measured for YOLOv5n–SiLU, despite both using the same activation function.

Table 5.4. Model Performance Comparison Across CoreVectorBlox Versions

Model	Accel. Core	CNN Speed	Post-Proc. Type	RISC-V Latency
Yolov5n – ReLU	CoreVBX 1	~20.9 ms	Anchor-based	802.7 ms
Yolov5n – SiLU	CoreVBX 2	~51.6 ms	Anchor-free	1908.6 ms
Yolov5nu – SiLU	CoreVBX 2	~126.78 ms	Anchor-free	629.6 ms

5.3.3 ANHEO Performance Evaluation

Table 5.5 presents YOLOv5su performance on the satellite detection task across CoreVBX versions. Metrics were obtained through direct power measurements during 1000 inference cycles with batch size 1. Version 1 requires software execution of final head layers due to limited hardware support, while version 2 enables full hardware acceleration, significantly improving inference speed and reducing per-frame energy consumption.

Table 5.5. CoreVectorBlox Performance Comparison

Device	Inference Speed	Avg. Dynamic Power	Avg. Dynamic Energy
CoreVectorBlox v1 + RISC-V	~791.5 ms	~0.58 W	~459 mJ/fr.
CoreVectorBlox v2	~240.5 ms	~0.96 W	~230.8 mJ/fr.

Table 5.6 provides additional context by comparing ANHEO performance with high-performance commercial hardware platforms. The comparison reveals the substantial energy efficiency advantages of the FPGA-based solution: while commercial CPU and GPU platforms achieve lower inference latency, they consume orders of magnitude more power. The Intel Core i9 CPU requires ~ 7.0 J/frame, and the NVIDIA A100 GPU consumes ~ 3.0 J/frame, compared to ~ 0.6 J/frame for ANHEO with CoreVectorBlox v2. Despite INT8 quantization causing only minimal accuracy degradation (mAP50: 0.972 for FP32 vs. 0.962 for INT8), the substantial energy savings make the FPGA solution particularly well suited for power-constrained satellite platforms.

Table 5.6. Performance Comparison Across Different HW Platforms

Device	Prec.	mAP50	Inference Speed	Avg. Power	Avg. Energy
Intel Core i9	FP32	0.972	~ 56 ms	~ 125 W	~ 7.0 J/fr.
A100 GPU	FP32	0.972	~ 10 ms	~ 300 W	~ 3.0 J/fr.
ANHEO (CoreVBX v1 + RISC-V)	INT-8	0.962	~ 791.5 ms	~ 2.3 W	~ 1.8 J/fr.
ANHEO (CoreVBX v2)	INT-8	0.962	~ 240.5 ms	~ 2.7 W	~ 0.6 J/fr.

The results highlight the substantial performance gains achieved with the CoreVBX v2 upgrade. Although the average dynamic power increases moderately (from ~ 0.58 W to ~ 0.96 W), version 2 delivers an impressive $3.3\times$ boost in inference speed (from ~ 791.5 ms to ~ 240.5 ms per frame) thanks to full hardware acceleration. This enhanced throughput translates into a notable 50% reduction in energy per frame (from ~ 459 mJ to ~ 230.8 mJ), underscoring the effectiveness of removing software execution from the detection head.

The combined reduction in energy consumption and latency validates the architectural upgrade as a decisive step forward. These improvements are particularly advantageous for embedded space missions such as ANHEO, where both energy efficiency and real-time responsiveness are critical. Achieving sub-second inference while maintaining energy consumption below 231 mJ per frame positions CoreVBX v2 as a highly effective solution for satellite-based object detection under continuous, power-constrained operation.

Figure 5.8 illustrates representative detections performed by the YOLOv5su model deployed onboard the ANHEO unit.



Figure 5.8. Satellite Detection

Conclusions and Future Work

The research presented in this thesis explores the integration of artificial intelligence into reconfigurable computing platforms for space applications, addressing the core challenges of performance, energy efficiency, and adaptability in resource-constrained environments. The work encompasses the design of lightweight neural architectures, the optimization of FPGA-based acceleration frameworks, and their practical deployment within Earth observation and navigation systems, focusing on small satellites constrained by limited power, mass, and volume. Accurate and timely target detection through optical remote sensing is a cornerstone of modern monitoring and surveillance, spanning both civilian and defense applications. In the maritime domain, these include vessel tracking, detection of illegal fishing and unauthorized migration, and support for search-and-rescue operations. Traditional AI-based remote sensing approaches, which rely on ground-based image processing, often introduce significant latency between data acquisition and actionable insights. This delay can hinder real-time decision-making in critical maritime scenarios such as hazards and contingency events.

To overcome this limitation, the work proposed a novel hardware architecture based on the Microchip PolarFire System-on-Chip, enabling low-power, real-time vessel detection directly onboard spacecraft. The first major contribution introduced the *SR-YOLOv5s* model, a lightweight object-detection network integrating a fully convolutional super-resolution backbone to enhance small-object recognition in satellite imagery acquired

under diverse maritime conditions, including varying sea states, cloud coverage, haze, and complex coastal backgrounds. The model achieved a mean mAP50 of 0.9272 ± 0.0029 , representing an 8.5% improvement over the baseline YOLOv5s, with significant gains in detecting very small objects (area $\leq 7 \times 7$ pixels), where mAP50 increased from 0.2832 to 0.4658—a 64.5% improvement in detection capability. External validation on the MASATI dataset confirmed the model’s robustness and generalization across diverse operational conditions.

Beyond algorithmic innovation, this thesis demonstrated the benefits of performing AI inference directly on FPGA hardware. Using the PolarFire SoC and the CoreVBX accelerator, the *SR-YOLOv5s* model achieved real-time inference at approximately 18 fps with a dynamic power consumption of only 1.16 W, yielding the highest accuracy-per-watt ratio among all tested platforms, including the NVIDIA Jetson Orin Nano and the cloud-based A100 GPU. These results confirm the practical viability of compact, energy-efficient AI pipelines suitable for space missions using space-grade hardware. The complete design, fully deployable on the radiation-tolerant RT PolarFire SoC, provides a robust and reliable platform for executing AI workloads in harsh orbital environments, including long-duration or GEO missions. The hardened quad-core RISC-V subsystem, capable of running a full Linux environment, enables rapid software development, advanced task management, and efficient orchestration of complex AI pipelines through flexible hardware/software partitioning. The flash-based, non-volatile configuration technology of the PolarFire family inherently protects against configuration SEUs, ensuring stable operation even under high-radiation exposure. Moreover, the device exhibits excellent tolerance to TID effects—maintaining nominal performance well beyond standard mission doses up to 100 krad—and strong SEL immunity with a LET threshold of $68.2 \text{ MeV cm}^2 \text{ mg}^{-1}$.

A second major contribution concerned the design and integration of the ANHEO system—an autonomous spacecraft navigation unit developed by TSD-Space. The ANHEO elegant B/B integrates a PolarFire SoC with a CoreVBX AI engine, enabling onboard deep-learning inference for visual navigation within a compact 0.8 U form factor and a power envelope below 10 W. The demonstrated bitstream reconfigurability highlights the intrinsic advantage of FPGA technology: the ability to evolve mission capabilities

over time without hardware redesign. The transition from [CoreVBX v1](#) to v2, achieved entirely through bitstream reconfiguration, resulted in a $3.3\times$ performance increase and a 50% reduction in per-frame energy consumption—underscoring the scalability and sustainability of the proposed approach. Overall, these results demonstrate that [FPGAs](#) are not only efficient and radiation-resilient but also fundamental in enabling upgradeable, and autonomous [AI](#) systems in orbit. The synergy between hardware flexibility and algorithmic adaptability paves the way for a new class of intelligent, reconfigurable spacecraft capable of responding dynamically to evolving mission requirements.

Building on these results, several promising research directions emerge. A first step involves embedding real-time detection and alert generation directly within the payload camera electronics, enabling autonomous identification of relevant events such as ship movements, wildfire outbreaks, or geological anomalies. These intelligent payloads could trigger selective data transmission, drastically reducing downlink bandwidth while enhancing mission responsiveness. The adoption of *space relay* links [111] for ultra-light alert downlink—where information is condensed into just a few bytes of metadata, such as a vessel’s coordinates—offers an efficient strategy to ensure low-latency situational awareness in remote or bandwidth-limited environments.

Future developments should also address data-transfer bottlenecks between high-throughput sensors and AI accelerators. Executing neural inference directly at the [FPGA](#) level of the camera electronics can minimize interfacing overhead with external COTS accelerators, thereby supporting true real-time responsiveness. Expanding on this concept, a modular onboard processing pipeline may include adaptive pre-processing, super-resolution, and detection stages (*e.g.*, SR-YOLOv5 on [CoreVBX](#)), followed by lightweight binary classifiers for false-positive filtering and contextual alert logic based on environmental conditions and temporal consistency.

Onboard vessel detection could be further enhanced through data fusion with the *Automatic Identification System* (AIS)—a VHF-based transponder network broadcasting a vessel’s identity, GNSS-derived position, course, speed, and status [186]. Correlating onboard detections with AIS signals would allow identification of vessels appearing in imagery but absent from AIS, *i.e.*, *dark vessels*, highlighting potentially non-cooperative or anomalous

behavior. At constellation scale, distributing this function across multiple CubeSats enables persistent, wide-area coverage with short revisit times. By performing inference and data triage on board—transmitting only event summaries or alerts—the system can achieve high surveillance effectiveness with markedly improved energy and downlink efficiency.

From a technological perspective, future efforts should focus on refining the AI-hardware co-design workflow. Optimizing quantization through *quantization-aware training* will enable mission-specific trade-offs between accuracy, latency, and power. Furthermore, integrating *structural pruning* techniques—removing entire filters or convolutional blocks rather than individual weights—offers a promising means of reducing computational load and memory footprint while preserving accuracy. Structural pruning directly benefits [FPGA](#) architectures by simplifying scheduling and maximizing parallel utilization, thereby improving performance-per-watt efficiency.

Finally, this work outlines a roadmap for ultra-low-power, intelligent vision systems in space. Achieving this vision will require coordinated co-design across hardware-aware neural networks, reconfigurable accelerators, and low-power optimization toolchains. As these technologies converge, they will enable compact, autonomous, and upgradeable spacecraft capable of operating reliably for extended durations in harsh environments—ultimately redefining space exploration and Earth observation through onboard intelligence.

Bibliography

- [1] Hamdy Abdelkhalik, Yehia Arafa, Nandakishore Santhi, and Abdel-Hameed A Badawy. Demystifying the nvidia ampere architecture through microbenchmarking and instruction-level analysis. In *2022 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–8. Ieee, 2022.
- [2] Adekanmi Adeyinka Adegun, Serestina Viriri, and Jules-Raymond Tapamo. Review of deep learning methods for remote sensing satellite images classification: experimental survey and comparative analysis. *Journal of Big Data*, 10(1):93, 2023.
- [3] Advanced Micro Devices, Inc. Xqr space-grade kintex ultrascale and space heritage: Space solution brief. <https://www.xilinx.com/publications/solution-briefs/xilinx-space-solution-brief.pdf>, 2023. Product Brief, PID2044250. Accessed on October 21, 2025.
- [4] Advanced Micro Devices, Inc. Versal adaptive socs and fpgas. <https://www.amd.com/en/products/adaptive-socs-and-fpgas/versal.html>, 2025. Accessed on October 22, 2025.
- [5] European Space Agency. Artificial intelligence in space. https://www.esa.int/Enabling_Support/Preparing_for_the_Future/Discovery_and_Preparation/Artificial_intelligence_in_space, 2025. Accessed October 11, 2025.
- [6] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020.
- [7] Aitech Systems. S-a1760 venus: Radiation-characterized gpgpu for ai in space. <https://aitechsystems.com/product/s-a1760-space-gpgpu/>, 2025. Accessed on October 20, 2025.

-
- [8] Mohammed AA Al-qaness, Aaqif Afzaal Abbasi, Hong Fan, Rehab Ali Ibrahim, Saeed H Alsamhi, and Ammar Hawbani. An improved yolo-based road traffic monitoring system. *Computing*, 103(2):211–230, 2021.
 - [9] AMD Xilinx. Finn documentation. <https://xilinx.github.io/finn/>. Accessed on July 20, 2025.
 - [10] AMD Xilinx. Vitis AI 3.0 — workflow & system integration. <https://xilinx.github.io/Vitis-AI/3.0/html/docs/workflow-system-integration.html>, 2024. Accessed on October 15, 2025.
 - [11] AMD Xilinx. Vitis ai 3.0 — documentation. <https://xilinx.github.io/Vitis-AI/3.0/html/index.html>, 2025. Accessed on October 22, 2025.
 - [12] Henintsoa S Andrianarivony and Moulay A Akhloufi. Machine learning and deep learning for wildfire spread prediction: A review. *Fire*, 7(12):482, 2024.
 - [13] Antonio Pertusa Antonio-Javier Gallego and Pablo Gil. Automatic ship classification from optical aerial images with convolutional neural networks. *Remote Sensing*, 10(4), 2018.
 - [14] Andrea Apicella, Francesco Donnarumma, Francesco Isgrò, and Roberto Prevete. A survey on modern trainable activation functions. *Neural Networks*, 138:14–32, 2021.
 - [15] Agathe Archet, Nicolas Gac, François Orieux, and Nicolas Ventroux. Embedded ai performances of nvidia’s jetson orin soc series. In *17ème Colloque National du GDR SOC2*, 2023.
 - [16] Rahul Kumar Arram, Aashman Gupta, and Kilic Cagri. Space exploration challenges a novel investigation and study of unmanned space vehicles. In *2025 Regional Student Conferences*, page 99722, 2025.
 - [17] Riadh Ayachi, Yahia Said, and Abdessalem Ben Abdelali. Optimizing neural networks for efficient fpga implementation: A survey. *Archives of Computational Methods in Engineering*, 28(7):4537–4547, 2021.
 - [18] BAE Systems. Rad750® component — datasheet. <https://www.baesystems.com/en-us/dam/jcr:c297f318-c618-4015-9399-adf76ac5cf55/16-f78-RAD750-Component-datasheet-2025-web.pdf>, 2025. Accessed on September 10, 2025.
 - [19] P. Bajanaru, R. Domingo Torrijos, D. Gonzalez-Arjona, F. A. Stancu, C. Onofrei, M. Marugan Borelli, R. Chamoso Rojo, and C. G. Miha-lache. Reconfigurable co-processor for spacecraft autonomous navigation.
-

- In *OBDP2021 – 2nd European Workshop on On-Board Data Processing*. Zenodo, June 2021. Conference paper, CC BY 4.0.
- [20] Brendan Barry, Cormac Brick, Fergal Connor, David Donohoe, David Moloney, Richard Richmond, Martin O’Riordan, and Vasile Toma. Always-on vision processing unit for mobile applications. *IEEE Micro*, 35(2):56–66, 2015.
 - [21] Svetla Ben-Itzhak. The use of artificial intelligence in outer space capabilities. *SAIS Review of International Affairs*, 44(2):5–30, 2024.
 - [22] Richard Berger, Steve Chadwick, Ernesto Chan, Richard Ferguson, Patrick Fleming, Jane Gilliam, Michael Graziano, Mary Hanley, Andrew Kelly, Marla Lassa, et al. Quad-core radiation-hardened system-on-chip power architecture processor. In *2015 IEEE Aerospace Conference*, pages 1–12. IEEE, 2015.
 - [23] Ramesh Bhaktavatchalu, B Sasi Rekha, G Ananta Divya, and V Usha Sai Jyothi. Design of axi bus interface modules on fpga. In *2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pages 141–146. IEEE, 2016.
 - [24] LI Bo, XIE Xiaoyang, WEI Xingxing, and TANG Wenting. Ship detection and classification from optical remote sensing images: A survey. *Chinese Journal of Aeronautics*, 34(3):145–163, 2021.
 - [25] Marc Solé Bonet and Leonidas Kosmidis. Sparrow: A low-cost hardware/-software co-designed simd microarchitecture for ai operations in space processors. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1139–1142. IEEE, 2022.
 - [26] Ali Borji, Ming-Ming Cheng, Qibin Hou, Huaizu Jiang, and Jia Li. Salient object detection: A survey. *Computational visual media*, 5:117–150, 2019.
 - [27] Molly E Brown, Mark L Carroll, and Vanessa M Escobar. User needs and assessing the impact of low latency nasa earth observation data availability on societal benefit. *Space Policy*, 30(3):135–137, 2014.
 - [28] Lile Cai, Bin Zhao, Zhe Wang, Jie Lin, Chuan Sheng Foo, Mohamed Sabry Aly, and Vijay Chandrasekhar. Maxpoolnms: getting rid of nms bottlenecks in two-stage object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9356–9364, 2019.
 - [29] James B Campbell and Randolph H Wynne. *Introduction to remote sensing*. Guilford press, 2011.
-

-
- [30] Matthew Cannon, William Evans, David Lee, Ian Troxel, Matthew Gruber, and Daniel Sabogal. Proton and heavy ion see data on nvidia and amd gpus. In *2023 IEEE Radiation Effects Data Workshop (REDW)(in conjunction with 2023 NSREC)*, pages 1–8. IEEE, 2023.
 - [31] Michele Caon, Paolo Motto Ros, Maurizio Martina, Tiziano Bianchi, Enrico Magli, Francisco Membibre, Alexis Ramos, Antonio Latorre, Murray Kerr, Stefan Wiehle, et al. Very low latency architecture for earth observation satellite onboard data handling, compression, and encryption. In *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pages 7791–7794. IEEE, 2021.
 - [32] G Capuano, M Severi, F Cacace, R Lirato, P Longobardi, G Pollio, M DeNino, M Ippolito, et al. Video system for prisma formation flying mission. In *Proceedings of the IAA symposium on small satellite systems and services (4S), Rhodes, Greece*, pages 26–30, 2008.
 - [33] Vincenzo Capuano, Alexei Harvard, and Soon-Jo Chung. On-board cooperative spacecraft relative navigation fusing gnss with vision. *Progress in Aerospace Sciences*, 128:100761, 2022.
 - [34] Vincenzo Capuano, Roberto Opromolla, Giovanni Cuciniello, Vincenzo Pesce, Salvatore Sarno, Giuseppe Capuano, Michelle Lavagna, Michele Grassi, Federico Corrado, Paolo Tabacco, et al. A highly integrated navigation unit for on-orbit servicing missions. In *INTERNATIONAL ASTRO-NAUTICAL CONGRESS: IAC PROCEEDINGS*, pages 1–13. International Astronautical Federation, IAF, 2018.
 - [35] Megan Casey, Ed Wyrwas, and Rebekah Austin. Recent radiation test results on cots ai edge processing asics. In *NEPP Electronics Technology Workshop (ETW)*, 2022.
 - [36] Jeffery C Chancellor, Graham BI Scott, and Jeffrey P Sutton. Space radiation: the number one risk to astronaut health beyond low earth orbit. *Life*, 4(3):491–510, 2014.
 - [37] Mengfu Chen, Chenguang Guo, Lei Chen, Wenjie Li, Fan Zhang, Xiaoxiang Hu, and Jiancheng Xu. Research on edac schemes for memory in space applications. *Electronics*, 10(5):533, 2021.
 - [38] Gong Cheng, Xiang Yuan, Xiwen Yao, Kebin Yan, Qinghua Zeng, Xingxing Xie, and Junwei Han. Towards large-scale small object detection: Survey and benchmarks. *IEEE transactions on pattern analysis and machine intelligence*, 45(11):13467–13488, 2023.
-

-
- [39] Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
 - [40] Ming-Fang Cheng, Arvind Mukundan, Riya Karmakar, Muhamed Adil Edavana Valappil, Jumana Jouhar, and Hsiang-Chen Wang. Modern trends and recent applications of hyperspectral imaging: A review. *Technologies*, 13(5):170, 2025.
 - [41] Remy Chevallier, Russel Forsyth, Roland Weigand, and Luca Fossati. Gr740: Rad-hard quad-core leon4ft system-on-chip.
 - [42] Yaqing Chi, Chang Cai, and Li Cai. Radiation effects of advanced electronic devices and circuits. *Electronics*, 13(6):1073, 2024.
 - [43] Bharadwaj Chintalapati, Arthur Precht, Sougata Hanra, Rene Laufer, Marcus Liwicki, and Jens Eickhoff. Opportunities and challenges of on-board ai-based image recognition for small satellite earth observation missions. *Advances in Space Research*, 2024.
 - [44] Marcian Cirstea, Khaled Benkrid, Andrei Dinu, Romeo Ghiriti, and Dorin Petreus. Digital electronic system-on-chip design: Methodologies, tools, evolution, and trends. *Micromachines*, 15(2):247, 2024.
 - [45] Sergio Cogliati, F Sarti, Leandro Chiarantini, M Cosi, R Lorusso, Ettore Lopinto, Franco Miglietta, Lorenzo Genesio, Luis Guanter, Alexander Damm, et al. The prisma imaging spectroscopy mission: overview and first performance analysis. *Remote sensing of environment*, 262:112499, 2021.
 - [46] Intel Corporation. Intel movidius myriad x vpu — product brief. <https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/myriad-x-product-brief.pdf>, 2025. Accessed on October 16, 2025.
 - [47] Intel Corporation. Openvino toolkit — overview. <https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html>, 2025. Accessed on October 16, 2025.
 - [48] Felix Cuervo, Johannes Ebert, Michael Schmidt, and Pantelis-Daniel Arapoglou. Q-band leo earth observation data downlink: Radiowave propagation and system performance. *IEEE Access*, 9:165611–165617, 2021.
 - [49] Jeffrey S Czapla-Myers, Kurtis J Thome, Nikolaus J Anderson, Larry M Leigh, Cibele Teixeira Pinto, and Brian N Wenny. The ground-based absolute radiometric calibration of the landsat 9 operational land imager. *Remote Sensing*, 16(6):1101, 2024.
-

-
- [50] Mikolaj Czerkawski, Robert Atkinson, Craig Michie, and Christos Tachtatzis. Satellitcloudgenerator: Controllable cloud and shadow synthesis for multi-spectral optical satellite images. *Remote Sensing*, 15(17), 2023.
 - [51] Matteo Dadà, Luca Zulberti, Pietro Nannipieri, Luca Fanucci, and Silvia Moranti. Inference and evaluation of deep convolutional neural networks on microchip’s hardware accelerator vectorblox. In *2023 European Data Handling & Data Processing Conference (EDHPC)*, pages 1–8. IEEE, 2023.
 - [52] Roberto Del Prete, Gabriele Meoni, Manuel Salvoldi, Domenico Barretta, Maria Daniela Graziano, Nicolas Longép  , and Alfredo Renga. Enhanced maritime monitoring via onboard processing of raw multi-spectral imagery by deep learning. In *IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium*, pages 1713–1717. IEEE, 2024.
 - [53] Gil Denis, H  l  ne de Boissezon, Steven Hosford, Xavier Pasco, Bruno Montfort, and Franck Ranera. The evolution of earth observation satellites in europe and its impact on the performance of emergency response services. *Acta Astronautica*, 127:619–633, 2016.
 - [54] Tim Dettmers. 8-bit approximations for parallelism in deep learning. *arXiv preprint arXiv:1511.04561*, 2015.
 - [55] Pietro Di Stasio, Alessandro Sebastianelli, Gabriele Meoni, and Silvia Liberata Ullo. Early detection of volcanic eruption through artificial intelligence on board. In *2022 IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRINE)*, pages 714–718. IEEE, 2022.
 - [56] Lorenzo Diana and Pierpaolo Dini. Review on hardware devices and software techniques enabling neural network inference onboard satellites. *Remote Sensing*, 16(21):3957, 2024.
 - [57] P Kingma Diederik. Adam: A method for stochastic optimization. (*No Title*), 2014.
 - [58] Gianmarco Dinelli, Gabriele Meoni, Emilio Rapuano, Gionata Benelli, and Luca Fanucci. An fpga-based hardware accelerator for cnns using on-chip memories only: Design and benchmarking with intel movidius neural compute stick. *International Journal of Reconfigurable Computing*, 2019(1):7218758, 2019.
 - [59] Gianmarco Dinelli, Gabriele Meoni, Emilio Rapuano, and Luca Fanucci. Advantages and limitations of fully on-chip cnn fpga-based hardware accelerator. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2020.
-

-
- [60] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.
- [61] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part II 14*, pages 391–407. Springer, 2016.
- [62] Richard J Doyle. Autonomy needs and trends in deep space exploration. 2003.
- [63] JE Drennan and DJ Hamman. Radiation effects design handbook. *NASA Contractor Report, Section, 4*, 1971.
- [64] European Cooperation for Space Standardization (ECSS). ECSS-Q-ST-60-15C Rev. 1: Radiation Hardness Assurance. <https://ecss.nl/standard/ecss-q-st-60-15c-rev-1-radiation-hardness-assurance-20-march-2025/>, March 2025. Standard, published 20 March 2025.
- [65] European Space Agency. Planetscope — earth online (eo gateway). <https://earth.esa.int/eogateway/missions/planetscope>, n.d. Accessed on October 3, 2025.
- [66] Fast Machine Learning Lab. hls4ml. <https://fastmachinelearning.org/hls4ml/>. Accessed on July 20, 2025.
- [67] Gianluca Furano, Gabriele Meoni, Aubrey Dunne, David Moloney, Veronique Ferlet-Cavrois, Antonis Tavoularis, Jonathan Byrne, Léonie Buckley, Mihalís Psarakis, Kay-Obbe Voss, et al. Towards the use of artificial intelligence on the edge in space systems: Challenges and opportunities. *IEEE Aerospace and Electronic Systems Magazine*, 35(12):44–56, 2020.
- [68] Asier Garcia-Perez, Raúl Miñón, Ana I Torre-Bastida, and Ekaitz Zulueta-Guerrero. Analysing edge computing devices for the deployment of embedded ai. *Sensors*, 23(23):9495, 2023.
- [69] Max Ghiglione, Amir Raoofy, Gabriel Dax, Gianluca Furano, Richard Wiest, Carsten Trinitis, Martin Werner, Martin Schulz, and Martin Langer. Machine learning application benchmark for in-orbit on-board data processing. In *European Workshop on On-Board Data Processing*, 2021.
- [70] Gianluca Giuffrida, Lorenzo Diana, Francesco De Gioia, Gionata Benelli, Gabriele Meoni, Massimiliano Donati, and Luca Fanucci. Cloudscout: A deep neural network for on-board cloud detection on hyperspectral images. *Remote Sensing*, 12(14):2205, 2020.
-

-
- [71] Gianluca Giuffrida, Luca Fanucci, Gabriele Meoni, Matej Batič, Léonie Buckley, Aubrey Dunne, Chris Van Dijk, Marco Esposito, John Hefele, Nathan Vercruyssen, et al. The ϕ -sat-1 mission: The first on-board deep neural network demonstrator for satellite earth observation. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–14, 2021.
- [72] Justin Goodwill, Gary Crum, James MacKinnon, Cody Brewer, Michael Monaghan, Travis Wise, and Christopher Wilson. Nasa spacecube edge tpu smallsat card for autonomous operations and onboard science-data analysis. In *Proceedings of the Small Satellite Conference*, number SSC21-VII-08. AIAA, 2021.
- [73] Google. Google colab. <https://colab.google/>, 2025. Accessee on February 4, 2025.
- [74] Google Coral. Edge tpu benchmarks. <https://gweb-coral-full.uc.r.appspot.com/docs/edgetpu/benchmarks/>, 2025. Official Coral documentation. Accessed on October 16, 2025.
- [75] Giorgia Guerrisi, Fabio Del Frate, and Giovanni Schiavon. Artificial intelligence based on-board image compression for the φ -sat-2 mission. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 16:8063–8075, 2023.
- [76] Shengxi Gui, Shuang Song, Rongjun Qin, and Yang Tang. Remote sensing object detection in the deep learning era—a review. *Remote Sensing*, 16(2):327, 2024.
- [77] Oscar Gutiérrez, Manuel Prieto, Alvaro Perales-Eceiza, Ali Ravanbakhsh, Mario Basile, and David Guzmán. Toward the use of electronic commercial off-the-shelf devices in space: Assessment of the true radiation environment in low earth orbit (leo). *Electronics*, 12(19):4058, 2023.
- [78] Tamer Mekky Ahmed Habib. Artificial intelligence for spacecraft guidance, navigation, and control: a state-of-the-art. *Aerospace Systems*, 5(4):503–521, 2022.
- [79] Fumio Hamanaka, Takashi Odan, Kenji Kise, and Thiem Van Chu. An exploration of state-of-the-art automation frameworks for fpga-based dnn acceleration. *IEEE Access*, 11:5701–5713, 2023.
- [80] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
-

-
- [81] Thorsten Hoeser and Claudia Kuenzer. Object detection and image segmentation with deep learning on earth observation data: A review-part i: Evolution and recent trends. *Remote Sensing*, 12(10):1667, 2020.
 - [82] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning non-maximum suppression. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4507–4515, 2017.
 - [83] Seyedehfaezeh Hosseininoorbin, Siamak Layeghy, Brano Kusy, Raja Jurdak, and Marius Portmann. Exploring edge tpu for deep feed-forward neural networks. *Internet of Things*, 22:100749, 2023.
 - [84] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.
 - [85] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
 - [86] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *journal of machine learning research*, 18(187):1–30, 2018.
 - [87] Avery Hurt. Van allen belts are dangerous radiation rings in space — here’s how astronauts get past them, 2025. Accessed on October 20, 2025.
 - [88] Muhammad Hussain. Yolo-v1 to yolo-v8, the rise of yolo and its complementary nature toward digital manufacturing and industrial defect detection. *Machines*, 11(7):677, 2023.
 - [89] Microchip Technology Inc. Vectorblox sdk. <https://github.com/Microchip-Vectorblox/VectorBlox-SDK>. GitHub repository, release 2.0.3.
 - [90] Microchip Technology Inc. Libero® soc design suite – versions 12.x and later. <https://www.microchip.com/en-us/products/fpgas-and-plds/fpga-and-soc-design-tools/fpga/libero-software-later-versions>, 2025. Accessed on November 10, 2025.
 - [91] Microchip Technology Inc. Polarfire® fpgas. <https://www.microchip.com/en-us/products/fpgas-and-plds/fpgas/polarfire-fpgas>, 2025. Accessed on October 19, 2025.
-

-
- [92] Microchip Technology Inc. Polarfire® soc fpgas. <https://www.microchip.com/en-us/products/fpgas-and-plds/system-on-chip-fpgas/polarfire-soc-fpgas>, 2025. Accessed on October 19, 2025.
- [93] Microchip Technology Inc. Polarfire® soc video kit (mpfs250–video-kit). <https://www.microchip.com/en-us/development-tool/mpfs250-video-kit>, 2025. Accessed on October 22, 2025.
- [94] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.
- [95] Javier Jalle. Dsp benchmark results of the gr740 rad-hard quad-core leon4ft. Presentation at ESA DSP Day 2016, AMICSA & DSP 2016, held in Gothenburg, Sweden. Available at https://indico.esa.int/event/102/contributions/62/attachments/125/148/5_1_P_DSP_Benchmark_Results_of_the_GR740_Rad-Hard_Quad-Core_LEON4FT_20160614.pdf, June 2016. Session: DSP software, tools and libraries. Accessed on October 20, 2025.
- [96] Won Jeon, Gun Ko, Jiwon Lee, Hyunwuk Lee, Dongho Ha, and Won Woo Ro. Deep learning with gpus. In *Advances in computers*, volume 122, pages 167–215. Elsevier, 2021.
- [97] Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma. A review of yolo algorithm developments. *Procedia computer science*, 199:1066–1073, 2022.
- [98] Piers Jiggins, Marc-Andre Chavy-Macdonald, Giovanni Santin, Alessandra Menicucci, Hugh Evans, and Alain Hilgers. The magnitude and effects of extreme solar particle events. *Journal of Space Weather and Space Climate*, 4:A20, 2014.
- [99] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, et al. Ultralytics YOLOv5. *GitHub repository*, 2020. Version 7.0.
- [100] Kiran Kasichayanula, Dan Terpstra, Piotr Luszczek, Stan Tomov, Shirley Moore, and Gregory D Peterson. Power aware computing on gpus. In *2012 Symposium on Application Accelerators in High Performance Computing*, pages 64–73. IEEE, 2012.
- [101] Michal Kawulok, Pawel Benecki, Szymon Piechaczek, Krzysztof Hryneczenko, Daniel Kostrzewa, and Jakub Nalepa. Deep learning for multiple-image
-

- super-resolution. *IEEE Geoscience and Remote Sensing Letters*, 17(6):1062–1066, 2019.
- [102] Amina Khan, Sumeet Gupta, and Sachin Kumar Gupta. Multi-hazard disaster studies: Monitoring, detection, recovery, and management, based on emerging technologies and optimal techniques. *International journal of disaster risk reduction*, 47:101642, 2020.
- [103] Mate Kisantal, Sumant Sharma, Tae Ha Park, Dario Izzo, Marcus Märtens, and Simone D’Amico. Satellite pose estimation challenge: Dataset, competition design, and results. *IEEE Transactions on Aerospace and Electronic Systems*, 56(5):4083–4098, 2020.
- [104] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, Lei Li, and Jianbo Shi. Foveabox: Beyond anchor-based object detection. *IEEE Transactions on Image Processing*, 29:7389–7398, 2020.
- [105] Tobia Armando La Marca, Maria Daniela Graziano, Michele Grassi, Lorenzo Iannascoli, Leonardo Cavicchiolo, Annalisa Capanni, Matteo Duzzi, Giovanni Maria Capuano, Vito Fortunato, Christian Cardenio, et al. Mission and system design for the earthnext cubesat vle0 mission. *CEAS Space Journal*, pages 1–17, 2025.
- [106] G Leccese, S Natalucci, L Iannascoli, M Melozzi, A Turella, E Piersanti, M Duzzi, F Trezzolani, GM Capuano, TA La Marca, et al. Preliminary design and perspectives of the earthnext cubesat mission for earth observation from very-low-earth orbit. In *Small Satellites Systems and Services Symposium (4S 2024)*, volume 13546, pages 1264–1279. SPIE, 2025.
- [107] LeCroy. Wavepro 7000 series. <https://www.testequimenthq.com/datasheets/LECROY-WAVEPRO%207100-Datasheet.pdf>, 2004. Accessed on July 7, 2024.
- [108] Vasileios Leon, George Lentaris, Dimitrios Soudris, Simon Vellas, and Mathieu Bernou. Towards employing fpga and asip acceleration to enable onboard ai/ml in space applications. In *2022 IFIP/IEEE 30th international conference on very large scale integration (VLSI-SoC)*, pages 1–4. IEEE, 2022.
- [109] Dawa Chyophel Lepcha, Bhawna Goyal, Ayush Dogra, and Vishal Goyal. Image super-resolution: A comprehensive review, recent trends, challenges and applications. *Information Fusion*, 91:230–260, 2023.
- [110] Bin Li. Embedded artificial intelligence. In *Embedded Artificial Intelligence: Principles, Platforms and Practices*, pages 3–25. Springer, 2024.
-

-
- [111] Fan Li, Kun Yu, Chao Yuan, Yichen Tian, Guang Yang, Kai Yin, and Youguang Li. Dark ship detection via optical and sar collaboration: An improved multi-feature association method between remote sensing images and ais data. *Remote Sensing*, 17(13):2201, 2025.
 - [112] Ke Li, Gang Wan, Gong Cheng, Liqui Meng, and Junwei Han. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS journal of photogrammetry and remote sensing*, 159:296–307, 2020.
 - [113] Liangzhi Li, Ling Han, Yuanxin Ye, Yuming Xiang, and Tingyu Zhang. Deep learning in remote sensing image matching: A survey. *ISPRS Journal of Photogrammetry and Remote Sensing*, 225:88–112, 2025.
 - [114] Wenbin Li and Matthieu Liewig. A survey of ai accelerators for edge environment. In *World Conference on Information Systems and Technologies*, pages 35–44. Springer, 2020.
 - [115] Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, 461:370–403, 2021.
 - [116] Leandro Ariel Libutti, Francisco D Igual, Luis Pinuel, Laura De Giusti, and Marcelo Naiouf. Benchmarking performance and power of usb accelerators for inference with mlperf. In *Proc. 2nd Workshop Accelerated Mach. Learn.(AccML)*, pages 1–15, 2020.
 - [117] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.
 - [118] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.
 - [119] Erik Lindholm, John Nickolls, Stuart Oberman, and John Montrym. Nvidia tesla: A unified graphics and computing architecture. *IEEE Micro*, 28(2):39–55, 2008.
 - [120] Fei Liu, Renwen Chen, Junyi Zhang, Shanshan Ding, Hao Liu, Shaofei Ma, and Kailing Xing. Esrtmdet: An end-to-end super-resolution enhanced real-time rotated object detector for degraded aerial images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 16:4983–4998, 2023.
-

-
- [121] Haiying Liu, Fengqian Sun, Jason Gu, and Lixia Deng. Sf-yolov5: A lightweight small object detection algorithm based on improved feature fusion mode. *Sensors*, 22(15):5817, 2022.
 - [122] Jiahang Liu, Jinlong Zhang, Yue Ni, Weijian Chi, and Zitong Qi. Small-object detection in remote sensing images with super resolution perception. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2024.
 - [123] Weiwei Liu, Bowen Cheng, Yalong Pang, and Yuehua Niu. Building an onboard high-performance computing platform based on spacewire and spacefibre. In *2022 4th International Conference on Intelligent Information Processing (IIP)*, pages 287–292. IEEE, 2022.
 - [124] Yaoqi Liu, Yinhe Han, Hongxin Li, Shuhao Gu, Jibing Qiu, and Ting Li. Computing over space: Status, challenges, and opportunities. *Engineering*, 2025.
 - [125] Yunjie Liu, Anne Bettens, and Xiaofeng Wu. Hardware/software co-design of a feature-based satellite pose estimation system. *IEEE Journal on Miniaturization for Air and Space Systems*, 5(1):16–26, 2023.
 - [126] Zhe Liu, Zukun Lu, Long Huang, Zhiwei Yao, Zhaojun Lu, and Jiliang Zhang. Recent advances on reliability of fpgas in a radiation environment. *Microelectronics Journal*, 148:106176, 2024.
 - [127] Tyler M Lovelly, Donavon Bryan, Kevin Cheng, Rachel Kreynin, Alan D George, Ann Gordon-Ross, and Gabriel Mounce. A framework to analyze processor architectures for next-generation on-board space computing. In *2014 IEEE Aerospace Conference*, pages 1–10. IEEE, 2014.
 - [128] Tyler M Lovelly and Alan D George. Comparative analysis of present and future space-grade processors with device metrics. *Journal of aerospace information systems*, 14(3):184–197, 2017.
 - [129] Raju Machupalli, Masum Hossain, and Mrinal Mandal. Review of asic accelerators for deep neural network. *Microprocessors and Microsystems*, 89:104441, 2022.
 - [130] Steven Malone, Patrick Saenz, and Patrick Phelan. Risc-v processors for spaceflight embedded platforms. In *2023 IEEE Aerospace Conference*, pages 1–11. IEEE, 2023.
 - [131] Jeffrey G Masek, Michael A Wulder, Brian Markham, Joel McCorkel, Christopher J Crawford, James Storey, and Del T Jenstrom. Landsat 9: Empowering open science and applications through continuity. *Remote Sensing of Environment*, 248:111968, 2020.
-

-
- [132] Giona Matasci, Jonathan Plante, Kevin Kasa, Payam Mousavi, Andrew Stewart, Andrew Macdonald, Anne Webster, and Jennifer Busler. Deep learning for vessel detection and identification from spaceborne optical imagery. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 3:303–310, 2021.
- [133] Gonzalo Mateo-Garcia, Josh Veitch-Michaelis, Cormac Purcell, Nicolas Longepe, Simon Reid, Alice Anlind, Fredrik Bruhn, James Parr, and Pierre Philippe Mathieu. In-orbit demonstration of a re-trainable machine learning payload for processing optical imagery. *Scientific Reports*, 13(1):10391, 2023.
- [134] Maxar Technologies. Worldview-3: Data sheet. <https://resources.maxar.com/data-sheets/worldview-3>, 2025. Accessed on October 03, 2025.
- [135] Nicola Melega, Valentina Marchese, Agnė Paškevičiūtė-Kidron, Bernardo Carnicero Dominguez, Nicolas Longepe, Oriol Aragon Casaled, Irina Babkina, Alessandro Marin, Hano Steyn, Léonie Buckley, et al. The esa ϕ sat-2 mission: An ai enhanced multispectral cubesat for earth observation. 2023.
- [136] Microchip. Smartpower user guide. <https://onlinedocs.microchip.com/pr/GUID-CA4693AD-4A50-4A30-B0B1-B2C7F4ADB3A6-en-US-6/index.html>, 2024. Accessed on July 22, 2024.
- [137] Microchip Technology Inc. Power comparison: Polarfire® fpgas and socs vs. competitors. https://ww1.microchip.com/downloads/aemDocuments/documents/FPGA/campaign/Results_Power_Comparison.pdf, 2023. Accessed on October 21, 2025.
- [138] Microchip Technology Inc. Corevectorblox ip handbook (vectorblox sdk v1.4.4.1). https://github.com/Microchip-Vectorblox/VectorBlox-SDK/blob/master/docs/CoreVectorBlox_IP_Handbook.pdf, 2024. Accessed on October 22, 2025.
- [139] Microchip Technology Inc. Vectorblox sdk (version 1.4.4.1). <https://github.com/Microchip-Vectorblox/VectorBlox-SDK>, 2024. Accessed on October 22, 2025.
- [140] Microchip Technology Inc. Rt polarfire® fpgas. <https://www.microchip.com/en-us/products/fpgas-and-plds/radiation-tolerant-fpgas/rt-polarfire-fpgas>, 2025. Accessed on October 21, 2025.
- [141] Microchip Technology Inc. Rt polarfire® soc. <https://www.microchip.com/en-us/products/fpgas-and-plds/radiation-tolerant-fpgas/rt-polarfire-soc>, 2025. Accessed on October 22, 2025.
-

-
- [142] Microchip Technology Inc. Vectorblox accelerator sdk for polarfire fpgas. <https://www.microchip.com/en-us/products/fpgas-and-plds/fpga-and-soc-design-tools/vectorblox>, 2025. Accessed on October 22, 2025.
- [143] Sparsh Mittal. A survey on optimized implementation of deep learning models on the nvidia jetson platform. *Journal of Systems Architecture*, 97:428–442, 2019.
- [144] Marina Mercè Cañete Muñoz, Florian Deconinck, Jordi Barrera Ars, Nicola Melega, Camille Pirat, Borja Jimeno Soto, Aubrey Dunne, Léonie Buckley, Juan Romero Cañas, Gionata Benelli, et al. Mantis, a 12u smallsat mission taking advantage of super-resolution and artificial intelligence for high-resolution imagery. 2024.
- [145] Giuseppe Napolano, Mario Pastore, Alessia Nocerino, Giancarmine Fasano, Michele Grassi, Roberto Opromolla, et al. A cnn-based relative navigation architecture for proximity operations in active debris removal missions. In *Proceedings of the 75th International Astronautical Congress (IAC)*, pages 541–559, 2024.
- [146] NASA. Space radiation effects on electronic components in low-earth orbit. Technical Report Lesson 824, NASA Lessons Learned Information System.
- [147] NASA. Towards learning-based visual perception with gavin, 2024. NASA TechPort, Project ID 146767.
- [148] NASA. Here’s how ai is changing nasa’s mars rover science, n.d. Accessed on 2025-10-09.
- [149] Issa AD Nesnas, Lorraine M Fesq, and Richard A Volpe. Autonomy for space robots: Past, present, and future. *Current Robotics Reports*, 2(3):251–263, 2021.
- [150] TS Nidhin, Anindya Bhattacharyya, RP Behera, T Jayanthi, and K Velusamy. Verification of fault tolerant techniques in finite state machines using simulation based fault injection targeted at fpgas for seu mitigation. In *2017 4th International Conference on Electronics and Communication Systems (ICECS)*, pages 153–157. IEEE, 2017.
- [151] NVIDIA. Nvidia tensorrt. <https://developer.nvidia.com/tensorrt>, 2024. Accessed on November 15, 2024.
- [152] NVIDIA. Nvml api reference. <https://docs.nvidia.com/deploy/nvml-api/nvml-api-reference.html#nvml-api-reference>, 2024. Accessed on July 22, 2024.
-

-
- [153] NVIDIA. Software-based power consumption modeling — jetson orin nx series and jetson orin nano series. <https://docs.nvidia.com/jetson/archives/r35.4.1/DeveloperGuide/text/SD/PlatformPowerAndPerformance/JetsonOrinNanoSeriesJetsonOrinNxSeriesAndJetsonAgxOrinSeries.html?#jetson-orin-nx-series-and-jetson-orin-nano-series>, 2024. Accessed on July 23, 2024.
- [154] NVIDIA. Jetson orin nano developer kit. <https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/nano-super-developer-kit/>, 2025. Accessed on October 15, 2025.
- [155] NVIDIA. Nvidia jetpack sdk. <https://developer.nvidia.com/embedded/jetpack>, 2025. Accessed on October 15, 2025.
- [156] NVIDIA Corporation. Nvidia cuda toolkit — developer zone. <https://developer.nvidia.com/cuda-toolkit>, 2025. Accessed on October 15, 2025.
- [157] NVIDIA Corporation. Nvidia cudnn — cuda deep neural network library. <https://developer.nvidia.com/cudnn>, 2025. Accessed on October 15, 2025.
- [158] NVIDIA Corporation. Nvidia jetson modules — embedded AI computing platform. <https://developer.nvidia.com/embedded/jetson-modules>, 2025. Accessed on October 15, 2025.
- [159] NVIDIA Developer Blog. Solving entry-level edge ai challenges with nvidia jetson orin nano, 2025. Accessed on October 15, 2025.
- [160] Victor UJ Nwankwo, Nnamdi N Jibiri, and Michael T Kio. The impact of space radiation environment on satellites operation in near-earth space. In *Satellites Missions and Technologies for Geosciences*. IntechOpen, 2020.
- [161] Paul A Oche, Gideon A Ewa, and Nwanneka Ibekwe. Applications and challenges of artificial intelligence in space missions. *IEEE Access*, 12:44481–44509, 2021.
- [162] Timothy R Oldham and FB McLean. Total ionizing dose effects in mos oxides and devices. *IEEE transactions on nuclear science*, 50(3):483–499, 2003.
- [163] Roberto Opromolla, Giancarmine Fasano, Giancarlo Rufino, and Michele Grassi. A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations. *Progress in Aerospace Sciences*, 93:53–72, 2017.
-

-
- [164] Roberto Opromolla, Dmitriy Grishko, John Auburn, Riccardo Bevilacqua, Luisa Buinhas, Joseph Cassady, Markus Jäger, Marko Jankovic, Javier Rodriguez, Maria Antonietta Perino, et al. Future in-orbit servicing operations in the space traffic management context. *Acta Astronautica*, 220:469–477, 2024.
- [165] K O'Neill, A Severance, and D Nandi. Using the vectorblox software development kit to create programmable ai/ml applications in radiation-tolerant rt polarfire fpgas. In *Proc. Eur. Workshop Board Data Process.(OBDP)*, pages 1–8, 2021.
- [166] Tommaso Pacini, Emilio Rapuano, and Luca Fanucci. Fpg-ai: a technology-independent framework for the automation of cnn deployment on fpgas. *IEEE Access*, 11:32759–32775, 2023.
- [167] Palantir. Edge ai in space: Palantir and satellogic, 2022. Palantir Blog.
- [168] Leo Pauly, Wassim Rharbaoui, Carl Shneider, Arunkumar Rathinam, Vincent Gaudilliere, and Djamila Aouada. A survey on deep learning-based monocular spacecraft pose estimation: Current state, limitations and prospects. *Acta Astronautica*, 212:339–360, 2023.
- [169] Santhosh Kumar Pendyala. Edge-cloud continuum for ai-driven remote patient monitoring: A scalable framework. *J. Data Sci. Inf. Technol*, 2:66–74, 2025.
- [170] Vincenzo Pesce, Andrea Colagrossi, and Stefano Silvestrini. *Modern spacecraft guidance, navigation, and control: from system modeling to AI and innovative applications*. Elsevier, 2022.
- [171] Evangelos Petrongonas, Vasileios Leon, George Lentaris, and Dimitrios Soudris. Paralos: A scheduling & memory management framework for heterogeneous vpus. In *2021 24th Euromicro Conference on Digital System Design (DSD)*, pages 221–228. IEEE, 2021.
- [172] Wesley Powell. Nasa high performance spaceflight computing (hpsc) overview. In *2025 NASA core Flight System (cFS) Symposium*, 2025.
- [173] Yuheng Qi, Defeng Gu, Yuan Liu, Jubo Zhu, Jian Wang, and Daoping Liu. A mission planning method for deep space detectors using deep reinforcement learning. *Aerospace Science and Technology*, 153:109417, 2024.
- [174] Jakaria Rabbi, Nilanjan Ray, Matthias Schubert, Subir Chowdhury, and Dennis Chao. Small-object detection in remote sensing images with end-to-end edge-enhanced gan and object detector network. *Remote Sensing*, 12(9):1432, 2020.
-

-
- [175] Devin P Ramaswami, David M Hiemstra, Zhi Wu Yang, Shuting Shi, and Li Chen. Single event upset characterization of the intel movidius myriad x vpu and google edge tpu accelerators using proton irradiation. In *2022 IEEE Radiation Effects Data Workshop (REDW)(in conjunction with 2022 NSREC)*, pages 1–3. IEEE, 2022.
 - [176] Emilio Rapuano, Gabriele Meoni, Tommaso Pacini, Gianmarco Dinelli, Gianluca Furano, Gianluca Giuffrida, and Luca Fanucci. An fpga-based hardware accelerator for cnns inference on board satellites: Benchmarking with myriad 2-based solution for the cloudscout case study. *Remote Sensing*, 13(8):1518, 2021.
 - [177] Sebastien Razakarivony and Frederic Jurie. Vehicle detection in aerial imagery: A small target detection benchmark. *Journal of Visual Communication and Image Representation*, 34:187–203, 2016.
 - [178] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
 - [179] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.
 - [180] Yanru Ren, Min Zhu, Dongyu Xu, Minghui Liu, Xuehui Dai, Shengao Wang, and Longxian Li. Overview on radiation damage effects and protection techniques in microelectronic devices. *Science and Technology of Nuclear Installations*, 2024(1):3616902, 2024.
 - [181] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019.
 - [182] N Rezzak, JJ Wang, R Chipana, C Lao, G Bakker, F Hawley, and E Hamdy. Heavy ion and proton induced single event effects on microchip rt polarfire fpga. In *2020 20th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, pages 1–4. IEEE, 2020.
 - [183] Nadia Rezzak, R Chipana, Alex Cai, G Bakker, F Hawley, and E Hamdy. Tid and see characterization of microchip polarfire® soc fpga. In *2022 22nd European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, pages 1–5. IEEE, 2022.
 - [184] John Alan Richards et al. *Remote sensing with imaging radar*, volume 1. Springer, 2009.
-

-
- [185] David Rijlaarsdam, Tom Hendrix, Pablo T Toledano González, Alberto Velasco-Mata, Léonie Buckley, Juan Puig Miquel, Oriol Aragon Casaled, and Aubrey Dunne. The next era for earth observation spacecraft: An overview of cognisat-6. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2024.
- [186] Maximilian Rodger and Raffaella Guida. Classification-aided sar and ais data fusion for space-based maritime surveillance. *Remote Sensing*, 13(1):104, 2020.
- [187] Ivan Rodriguez-Ferrandez, Maris Tali, Leonidas Kosmidis, Alessandra Costantino, and David Steenari. Exploring total ionizing dose radiation effects across generations of nvidia jetson devices: a comparative analysis. In *2024 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pages 1–6. IEEE, 2024.
- [188] Ning Ruan, Kun Li, Qiyang Zhang, Lauri Lovén, Praveen Kumar Donta, Yi Jia, and Schahram Dustdar. Edge ai for earth observation. *IEEE Internet Computing*, 29(3):31–40, 2025.
- [189] Antonia Russo and Gianluca Lax. Using artificial intelligence for space challenges: A survey. *Applied Sciences*, 12(10):5106, 2022.
- [190] James M Ryan, John A Lockwood, and Hermann Debrunner. Solar energetic particles. *Space Science Reviews*, 93(1):35–53, 2000.
- [191] satsearch. An overview of satellite x-band transmitters available on the global marketplace. <https://blog.satsearch.co/2020-02-27-an-overview-of-satellite-x-band-transmitters-available-on-the-global-marketplace>, 2020.
- [192] Fabian Schöttl, Andrea Spichtinger, Julian Franquinet, and Martin Langer. Real-time on-orbit fire detection on forest-2. In *IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium*, pages 2360–2364, 2024.
- [193] Daniel Selva and David Krejci. A survey and assessment of the capabilities of cubesats for earth observation. *Acta Astronautica*, 74:50–68, 2012.
- [194] Chahira Serief, Youcef Ghelamallah, and Youcef Bentoutou. Deep-learning-based system for change detection onboard earth observation small satellites. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 16:8115–8124, 2023.
- [195] Fred W Sexton. Destructive single-event effects in semiconductor devices and ics. *IEEE Transactions on Nuclear Science*, 50(3):603–621, 2003.
-

-
- [196] Vasuki Shankar. Edge ai: a comprehensive survey of technologies, applications, and challenges. In *2024 1st International Conference on Advanced Computing and Emerging Technologies (ACET)*, pages 1–6. IEEE, 2024.
 - [197] Jerad Shaw, Cody Anderson, Mike Choate, and Esad Micijevic. Landsat geometric and radiometric calibration and characterization. Technical report, US Geological Survey, 2024.
 - [198] Jacob Shermeyer and Adam Van Etten. The effects of super-resolution on object detection performance in satellite imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
 - [199] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.
 - [200] Stefano Silvestrini and Michèle Lavagna. Deep learning and artificial neural networks for spacecraft dynamics, navigation and control. *Drones*, 6(10):270, 2022.
 - [201] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2015.
 - [202] Tuomo Sipola, Janne Alatalo, Tero Kokkonen, and Mika Rantonen. Artificial intelligence in the iot era: A review of edge ai hardware and software. In *2022 31st Conference of Open Innovations Association (FRUCT)*, pages 320–331. IEEE, 2022.
 - [203] Zuxiang Situ, Qisheng Zhong, Jianliang Zhang, Shuai Teng, Xiaoguang Ge, Qianqian Zhou, and Zhiwei Zhao. Attention-based deep learning framework for urban flood damage and risk assessment with improved flood prediction and land use segmentation. *International Journal of Disaster Risk Reduction*, 116:105165, 2025.
 - [204] Tony C Slaba, Shirin Rahmanian, Stuart George, Diego Laramore, John W Norbury, Charles M Werneth, and Cary Zeitlin. Validated space radiation exposure predictions from earth to mars during artemis-i. *npj Microgravity*, 11(1):6, 2025.
 - [205] Jianing Song, Duarte Rondao, and Nabil Aouf. Deep learning-based spacecraft relative navigation methods: A survey. *Acta Astronautica*, 191:22–40, 2022.
-

-
- [206] Joseph R Srour and James M McGarrity. Radiation effects on microelectronics in space. *Proceedings of the IEEE*, 76(11):1443–1469, 2002.
- [207] Luca Sterpone and Boyang Du. Analysis and mitigation of single event effects on flash-based fpgas. In *2014 19th IEEE European Test Symposium (ETS)*, pages 1–6. IEEE, 2014.
- [208] STMicroelectronics. Rad-hard products for space applications. Brochure, available at <https://www.st.com/resource/en/brochure/rad-hard-space-applications.pdf>, May 2024. Order code: BR2405RHSA. Accessed on October 20, 2025.
- [209] Kathiravan Thangavel, Roberto Sabatini, Alessandro Gardi, Kavindu Ranasinghe, Samuel Hilton, Pablo Servidia, and Dario Spiller. Artificial intelligence for trusted autonomous satellite operations. *Progress in Aerospace Sciences*, 144:100960, 2024.
- [210] Alan C Tribble. *The space environment: implications for spacecraft design-revised and expanded edition*. Princeton University Press, 2003.
- [211] Devis Tuia, Konrad Schindler, Begüm Demir, Gustau Camps-Valls, Xiao Xiang Zhu, Mrinalini Kochupillai, Sašo Džeroski, Jan N van Rijn, Holger H Hoos, Fabio Del Frate, et al. Artificial intelligence to advance earth observation: a perspective. *arXiv preprint arXiv:2305.08413*, 1(3), 2023.
- [212] Ubotica Technologies. Products – ubotica technologies: Edge computing platforms. <https://ubotica.com/ubotica-products/>, 2025. Accessed on October 20, 2025.
- [213] Ultralytics. Ultralytics docs — yolov5. <https://docs.ultralytics.com/models/yolov5/>, 2025. Accessed on October 14, 2025.
- [214] Ultralytics. Yolov5. <https://github.com/ultralytics/yolov5>, 2025. Accessed on October 21, 2025.
- [215] Ultralytics. Yolov5 — modelli: Documentazione ultralytics (it). <https://docs.ultralytics.com/it/models/yolov5/>, October 2025. Documentation page.
- [216] Ultralytics. Yolov8 — modelli: Documentazione ultralytics (it). <https://docs.ultralytics.com/it/models/yolov8/>, October 2025. Documentation page.
- [217] C Urbina-Ortega, G Furano, G Magistrati, K Marinis, A Menicucci, and D Merodio-Codinachs. Flash-based fpgas in space, design guidelines and trade-off for critical applications. In *2013 14th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, pages 1–8. IEEE, 2013.
-

-
- [218] U.S. Geological Survey. Landsat collection 2 generation timeline. <https://www.usgs.gov/media/images/landsat-collection-2-generation-timeline>, 2025. Accessed on October 5, 2025.
- [219] Freek D Van der Meer, Harald MA Van der Werff, Frank JA Van Ruitenbeek, Chris A Hecker, Wim H Bakker, Marleen F Noomen, Mark Van Der Meijde, E John M Carranza, J Boudewijn De Smeth, and Tsehaie Woldai. Multi-and hyperspectral geologic remote sensing: A review. *International journal of applied Earth observation and geoinformation*, 14(1):112–128, 2012.
- [220] Deepak Vasisht, Jayanth Shenoy, and Ranveer Chandra. L2d2: Low latency distributed downlink for leo satellites. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, pages 151–164, 2021.
- [221] Raoul Velazco, Pascal Fouillat, and Ricardo Reis. *Radiation effects on embedded systems*. Springer Science & Business Media, 2007.
- [222] Vandí Verma, Ari Jónsson, Reid Simmons, Tara Estlin, and Rich Levinson. Survey of command execution systems for nasa spacecraft and robots. In *International Conference on Automated Planning and Scheduling 2005*, 2005.
- [223] Vandí Verma, Mark W Maimone, Daniel M Gaines, Raymond Francis, Tara A Estlin, Stephen R Kuhn, Gregg R Rabideau, Steve A Chien, Michael M McHenry, Evan J Graser, et al. Autonomous robotics is driving perseverance rover’s progress on mars. *Science Robotics*, 8(80):eadi3099, 2023.
- [224] Juyu Wang, Shenyi Zhang, Guohong Shen, Ying Sun, Binqun Zhang, Zheng Chang, Chunqin Wang, Donghui Hou, and Zhe Yang. Detection and analysis of radiation doses in multiple orbital space during solar minimum. *Aerospace*, 10(11):944, 2023.
- [225] Peijuan Wang, Bulent Bayram, and Elif Sertel. A comprehensive review on deep learning based remote sensing image super-resolution methods. *Earth-Science Reviews*, 232:104110, 2022.
- [226] Yi Wang, Syed Muhammad Arsalan Bashir, Mahrukh Khan, Qudrat Ullah, Rui Wang, Yilin Song, Zhe Guo, and Yilong Niu. Remote sensing image super-resolution and object detection: Benchmark and state of the art. *Expert Systems with Applications*, 197:116793, 2022.
- [227] Michael Wirthlin. High-reliability fpga-based systems: Space, high-energy physics, and beyond. *Proceedings of the IEEE*, 103(3):379–389, 2015.
-

-
- [228] Cristian Wisultschew, Andrés Otero, Jorge Portilla, and Eduardo de la Torre. Artificial vision on edge iot devices: a practical case for 3d data classification. In *2019 XXXIV Conference on Design of Circuits and Integrated Systems (DCIS)*, pages 1–7. IEEE, 2019.
- [229] Man Sing Wong, Xiaolin Zhu, Sawaid Abbas, Coco Yin Tung Kwok, and Meilian Wang. *Optical Remote Sensing*, pages 315–344. Springer Singapore, Singapore, 2021.
- [230] Michael A Xapsos. Applicability of let to single events in microelectronic structures. *IEEE transactions on nuclear science*, 39(6):1613–1621, 1992.
- [231] Xiaozhu Xie, Linhao Li, Zhe An, Gang Lu, and Zhiqiang Zhou. Small ship detection based on hybrid anchor structure and feature super-resolution. *Remote Sensing*, 14(15):3530, 2022.
- [232] Yongchun Xie, Yongjun Lei, Jianxin Guo, and Bin Meng. Autonomous guidance, navigation, and control of spacecraft. In *Spacecraft Dynamics and Control*, pages 371–422. Springer, 2021.
- [233] Xilinx, Inc. Radiation-hardened, space-grade virtex-5qv family data sheet: Overview (ds192). https://docs.amd.com/v/u/en-US/ds192_V5QV_Device_Overview, January 2018. DS192 v1.6, Product Specification. Accessed on October 21, 2025.
- [234] Wenming Yang, Xuechen Zhang, Yapeng Tian, Wei Wang, Jing-Hao Xue, and Qingmin Liao. Deep learning for single image super-resolution: A brief review. *IEEE Transactions on Multimedia*, 21(12):3106–3121, 2019.
- [235] Boris Yendler, Ashton Meginnis, and Adam Reif. Thermal management for high power cubesats. 2020.
- [236] Matthew D Zeiler, Graham W Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *2011 international conference on computer vision*, pages 2018–2025. IEEE, 2011.
- [237] Cong Zhang, Li Yuan, Mingyu Xie, Sihang Zhang, and Jiaxing Li. Autonomous mission planning of earth observation satellite based on onboard cloud detection. *Advances in Space Research*, 70(8):2178–2194, 2022.
- [238] Jiaqing Zhang, Jie Lei, Weiying Xie, Zhenman Fang, Yunsong Li, and Qian Du. Superyolo: Super resolution assisted object detection in multimodal remote sensing imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 61:1–15, 2023.
-

- [239] Xiangrong Zhang, Tianyang Zhang, Guanchun Wang, Peng Zhu, Xu Tang, Xiuping Jia, and Licheng Jiao. Remote sensing object detection meets deep learning: A metareview of challenges and advances. *IEEE Geoscience and Remote Sensing Magazine*, 11(4):8–44, 2023.
 - [240] Zhengning Zhang, Lin Zhang, Yue Wang, Pengming Feng, and Ran He. Shipsimagenet: A large-scale fine-grained dataset for ship detection in high-resolution optical remote sensing images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:8458–8472, 2021.
 - [241] Tianqi Zhao, Yongcheng Wang, Zheng Li, Yunxiao Gao, Chi Chen, Hao Feng, and Zhikang Zhao. Ship detection with deep learning in optical remote-sensing images: A survey of challenges and advances. *Remote Sensing*, 16(7):1145, 2024.
-

Author's publications

1. G.M. Capuano, S. Capuozzo, A.G.M. Strollo, N. Petra, *Super-Resolution YOLO Object Detection for Maritime Surveillance with Real-Time FPGA Processing Onboard Spacecraft*, *Acta Astronautica*, Elsevier, vol. 238, Part A, pp. 1291–1310, 2026. DOI: 10.1016/j.actaastro.2025.09.018.
2. T.A. La Marca, M.D. Graziano, M. Grassi, L. Iannascoli, L. Cavicchiolo, A. Capanni, M. Duzzi, G.M. Capuano, V. Fortunato, C. Cardenio, G. Lecce, M. Rinaldi, M. Di Clemente, R. Luciani, R. Natalucci, V. Pulcino, *Mission and System Design for the EarthNext CubeSat VLEO Mission*, *CEAS Space Journal*, Springer, vol. 238, pp. 1–17, 2025. DOI: 10.1007/s12567-025-00663-2.
3. G.M. Capuano, A.G.M. Strollo, N. Petra, *Super Resolution CNN for a Quincunx Sampling-Based Panchromatic Earth Observation Imager for Nanosatellites*, *International Astronautical Congress*, Baku, Azerbaijan, October 2023. In *Proceedings of the International Astronautical Congress*, IAC-23,E2,4,5,x79398.
4. G.M. Capuano, S. Capuozzo, A.G.M. Strollo, N. Petra, *FPGA-Based Hardware Acceleration for Real-Time Maritime Surveillance and Monitoring Onboard Spacecraft*, *SPAICE 2024: Conference on AI in and for Space*, European Centre for Space Applications and Telecommunications (ECSAT), Didcot, United Kingdom, October 2024. In *Proceedings of SPAICE2024: The First Joint ESA/IAA Conference on AI in and for Space*, pp. 156–161.
5. G.M. Capuano, S. Capuozzo, A.G.M. Strollo, N. Petra, *Super-Resolution-Based Small Object Detection for Real-Time Surveillance and Monitoring: An Onboard Satellite FPGA Implementation*, *International Astronautical Congress, Earth Observation Symposium*, Milan, Italy, October 2024. In *Proceedings of the International Astronautical Congress*, IAC-24-B1.IP.92.X89480.

6. G.M. Capuano, V. Capuano, G. Napolano, R. Opromolla, M. Severi, A.G.M. Strollo, N. Petra, G. Cuciniello, E. Zaccagnino, G. Varacalli, *FPGA Hardware Acceleration for Deep Learning-Based Satellite Relative Pose Estimation*, *AIAA SciTech Forum on Aerospace Research, Development, and Technology*, Orlando, USA, January 2025. In *Proceedings of the AIAA SciTech 2025 Forum*, p. 2715.
 7. G.M. Capuano, F. Saggiomo, A.G.M. Strollo, N. Petra, B. Confuorto, E. Zaccagnino, M. Di Clemente, *Enhancing Satellite Imagery Through Advanced Focal Plane In-Flight Active Motion Control: The FOPAC System Implementation and Results*, *International Astronautical Congress, Earth Observation Symposium*, Sydney, Australia, October 2025. In *Proceedings of the International Astronautical Congress*, IAC-25-B1,IPB,17,x96978.
 8. G. Leccese, S. Natalucci, L. Iannascoli, M. Melozzi, A. Turella, E. Piersanti, M. Duzzi, F. Trezzolani, G.M. Capuano, T.A. La Marca, M.D. Graziano, M. Grassi, V. Fortunato, P. De Marchi, C. Cardenio, *Preliminary Design and Perspectives of the EarthNext CubeSat Mission for Earth Observation from Very-Low-Earth Orbit*, *Small Satellites Systems and Services Symposium (4S 2024)*, Palma de Mallorca, Spain, May 2024. Vol. 13546, pp. 1264–1279, SPIE.
 9. G.M. Capuano, S. Capuozzo, A.G.M. Strollo, N. Petra, *Benefits of FPGAs for Deep Learning Acceleration*, *PRIME Conference, 20th International Conference on PhD Research in Microelectronics and Electronics*, Taormina, Italy, September 2025.
-