# High Performance Embedded Computing in Space: Evaluation of Platforms for Vision-based Navigation

George Lentaris[a], Konstantinos Maragos[a], Ioannis Stratakos[a],

Lazaros Papadopoulos[a], Odysseas Papanikolaou[a] and Dimitrios Soudris[b]
*National Technical University of Athens, 15780 Athens, Greece*

Manolis Lourakis[c] and Xenophon Zabulis[c]
*Foundation for Research and Technology - Hellas, POB 1385, 71110 Heraklion, Greece*

David Gonzalez-Arjona[d]
*GMV Aerospace & Defence SAU, Tres Cantos, 28760 Madrid, Spain*

Gianluca Furano[e]
*European Space Agency, European Space Technology Centre,*

*Keplerlaan 1 2201AZ Noordwijk, The Netherlands*

**Vision-based navigation has become increasingly important in a variety of space applications for enhancing autonomy and dependability. Future missions, such as active debris removal for remediating the low Earth orbit environment, will rely on novel high-performance avionics to support advanced image processing algorithms with substantial workloads. However, when designing new avionics architectures, constraints relating to the use of electronics in space present great challenges, further exacerbated by the need for significantly faster processing compared to conventional space-grade central processing units. With the long-term goal of designing high-performance embedded computers for space, in this paper, an extended study and trade-off analysis of a diverse set of computing platforms and architectures (i.e., central processing units,**

[a] Research Associate, School of Electrical and Computer Engineering, NTUA, Athens. glentaris@microlab.ntua.gr
[b] Associate Professor, School of Electrical and Computer Engineering, NTUA, Athens. dsoudris@microlab.ntua.gr
[c] Principal Researcher, Institute of Computer Science, FORTH, Heraklion. lourakis@ics.forth.gr
[d] Avionics and On-Board Software Engineer, GMV, Madrid.
[e] On-Board Computer Engineer, Microelectronics & Data Systems Division, ESA ESTEC, Noordwijk.

**multicore digital signal processors, graphics processing units, and field-programmable gate arrays) are performed with radiation-hardened and commercial off-the-shelf technology. Overall, the study involves more than 30 devices and 10 benchmarks, which are selected after exploring the algorithms and specifications required for vision-based navigation. The present analysis combines literature survey and in-house development/testing to derive a sizable consistent picture of all possible solutions. Among others, the results show that certain 28nm system-on-chip devices perform faster than space-grade and embedded CPUs by 1−3 orders of magnitude, while consuming less than 10 Watts. FPGA platforms provide the highest performance per Watt ratio.**

## I. Introduction

Vision-based navigation (VBN) endows robotic systems with the ability to guide themselves autonomously in an unknown environment by using active or passive image sensors to perceive their surroundings. Vision applications in space vary from rover exploration on Mars with simultaneous localization and mapping, to Moon landing with landmark tracking and hazard avoidance, and to spacecraft proximity operations with pose estimation of cooperative or uncooperative orbiting targets. However, the ever increasing computational demands of these enabling technologies are now challenging the classical on-board computing, both in terms of architecture and processing power. An example where this becomes especially apparent concerns active debris removal (ADR) scenarios, whose aim is to actively remove heavy debris objects by capturing them and then either disposing them by destructive re-entry in the atmosphere or moving them to graveyard orbits [1]. In future ADR missions, chaser spacecraft must rely, among others, on high-definition cameras and complex computer vision algorithms performing image feature extraction, matching, 3D reconstruction, tracking, etc., at high frame rates. Given the requirements for fast and accurate autonomous relative navigation, the gap between algorithmic complexity and conventional space-grade processors is quickly widening and calls for novel high-performance embedded computing in space.

The Conventional Avionics Architecture

Classical on-board data handling [2] is based on a central on board computer (OBC) connected to various platform subsystems, such as the radio communication, attitude & orbit control (AOCS), payloads, etc. (Fig. 1). Most often, the internal communication network involves a serial bus of high data integrity, i.e., the military standard 1553 or the controller area network (CAN) bus at around 1 Mbps. The OBC has two primary functions, namely to decode telecommands and to serve as the central terminal of the spacecraft. In few cases, the OBC also handles payload data, e.g., for a remote sensing application which would otherwise require a separate processor. The peripheral terminals are stand-alone units or embedded within the instruments of the spacecraft. The more complicated of these instruments include their own embedded processor to support intelligent control units (ICUs), sophisticated remote bus interfaces, and specialized science data processing, which normally demands higher performance than that of the OBC [2]. Occasionally, the instrument data is collected by an independent "payload data-handling system" supporting relatively high data rates.

Routine spacecraft tasks such as attitude control, housekeeping reports, command decoding and systems management, require very little processing power, which can be provided even by simple 8-bit microcontrollers [2]. More demanding tasks, such as payload software, intensive digital signal processing (e.g., time/frequency domain filtering, compression, feature extraction and matching), data formatting, error control, or encryption, are executed on more advanced microprocessors. Such tasks within payloads/experiments may have a dedicated controller interfacing with the platform, under real-time operation, to output data in source packet format. Their controller will typically be a standard microprocessor, such as the SPARC-based LEON, a digital signal processor (DSP) device, such as the TSC21020F, or, in more extreme cases, even an application-specific integrated circuit (ASIC). Peripheral supporting devices, like universal asynchronous receiver and transmitters (UARTs), dynamic random-access memory (DRAM) controllers, I/O controllers, and communications devices, have custom implementations on field programmable technology [2]. Until now, the aforementioned computers have predominantly been of space-grade technology and hence significantly slower compared to their contemporary terrestrial counterparts, lagging behind them by $1-2$ orders of magnitude in performance [2–4].

Trends and Hazards for Future Avionics

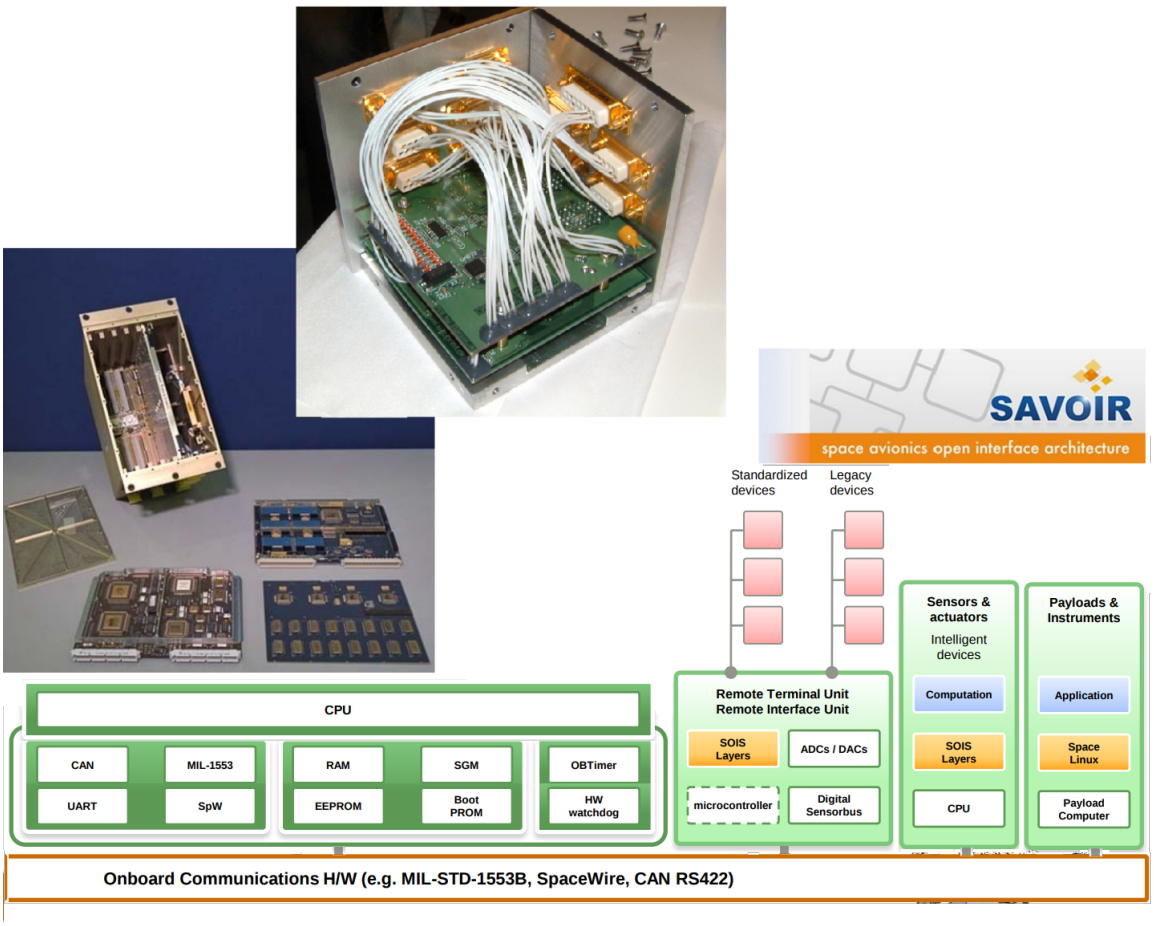


**Fig. 1 Examples of OBC (for SSETI Express, top, DMS-R for ISS, left) and avionics architecture (SAVOIR, bottom). Images © ESA.**

Recent trends in the space industry push the on-board processing power and storage towards terrestrial industrial practices [2]. The goal is to support more complex data organization, autonomous decision-making, intensive signal processing and multitasking, and the coordination of large distributed development teams. Commercial-off-the-shelf (COTS) devices are competing with conventional rad-hard components by offering the advantages of higher performance, faster/cheaper development, ease of maintenance, off-the-shelf development systems and processing/communications libraries, built-in file handling, multitasking and inter-task communication, and software reliability [2]. Lately, the use of COTS in high-reliability space applications is on the rise and ranges from memory chips in flagship planetary explorers to various parts in pico-nano sats [4]. Operational constellations of the latter currently provide useful services based on COTS that have been selected

and tested to meet certain mission requirements [5]. ESA roadmaps [6] present an emerging trend towards COTS adoption, whilst the US is moving aggressively to accept COTS chips in space. This is also evident in institutional missions e.g. by ESA and CNES who have multiple activities with commercial Xilinx Zynq devices for processing.

On the downside, when deployed in space, COTS devices can be severely harmed by the harsh environmental conditions of high-vacuum, extreme temperatures and high levels of ionizing radiation, or even by the vibrations during launch. Space users consider applying mitigation methods, e.g., redundancy, to handle the damage and increase the availability/reliability of these components in-flight [3, 6]. More specifically [2], although with very low probability, plastic materials may outgas under vacuum or cause electrostatic discharge. Thermal-wise, many COTS devices can only operate in a limited temperature range, e.g., $-30^o$ to $+70^o$ C, which must be moderated within a spacecraft. Radiation-wise, a non rad-hard COTS device will suffer from total dose effects (TDEs) and single event effects (SEEs) owing to the trapped particles in the Van Allen belts surrounding Earth, the galactic cosmic-rays originating from deep space and the solar-flare particles [7]. In general [2], avionics should be tolerant to $10-50$ kRad total ionizing dose (TID) for low earth orbit (LEO) missions and $100-300$ kRad for MEO/GEO missions. However, specifically for short duration missions in LEO orbit, e.g., as in an ADR scenario, the aforementioned radiation effects could be tolerable even for COTS due to the protection offered by the Earth's magnetic field and the limited exposure time.

Challenges in the Advent of VBN and High-Performance Avionics

COTS processing units will facilitate high-performance embedded computing in space, with VBN being one of the first subsystems to reap the increased benefits of new technologies. In turn, advanced image processing systems and complex guidance, navigation & control (GNC) will become the enabling building blocks in ADR, as well as in similar future space proximity operations such as inspection, in-orbit servicing, cargo and satellite delivery, space tug, in-orbit structure assembly, etc. [8–11]. Novel image processing will be combined with data fusion from a number of sensors providing information on the Sun, Earth, stars, spacecraft inertia, etc., in order to feed the control algorithms and provide dependable autonomous navigation. However, adjectives such as “advanced", “com-

plex", and "autonomous", conceal increased processing workloads and decreased time budgets. Put into perspective, conventional space-grade processors, such as LEON3 and RAD750, can provide only a very limited level of performance (e.g., 50−400 DMIPS or Dhrystone Million Instructions per Second), whereas the COTS desktop central processing units (CPUs) used by algorithm designers during the early development stages of VBN techniques provide 10−100x more speed when processing images. Even with the latest space-grade devices, such as LEON4 and RAD5545 that achieve 10x more speed compared to their predecessors, or even with embedded COTS processors, the current on-board CPU performance seems to be one order of magnitude less than that necessary to execute computationally expensive vision algorithms at adequate frame rates. Therefore, to support reliable autonomous VBN, the community should consider advancing the space avionics architectures by including hardware accelerators, such as field-programmable gate arrays (FPGAs), graphics processing units, (GPUs), or multi-core very long instruction word (VLIW) DSP processors. That is, in short, use VLIWs or GPUs consisting of a number of cores able to execute bigger or smaller instructions and facilitate data parallelization. In contrast, the FPGAs consist of a network of much smaller processing/storage elements (such as flip-flops), which can be combined altogether to create bigger components with very deep pipelining and parallelization at multiple levels.

Designing new high-performance avionics architectures for embedded computing becomes particularly challenging when one is also required to meet stringent constraints related to space. The spacecraft resources available to electronic systems are restricted [2], with the three most critical limiting factors being mass, power and volume. Power availability in space is low due to solar powering and power dissipation capabilities, whereas volume and mass have enormous costs per unit. Furthermore, the engineers must factor in reconfigurability, application-design complexity, fault/radiation tolerance, vibration tolerance, component pricing, etc. Last but not least, one of the most crucial factors is the achieved processing power. All of the aforementioned parameters are competing against each other in a mission-specific trade-off manner, which becomes even more complex when the types of the aforementioned hardware accelerators are included in the architecture.

Paper Contribution and Outline

In this paper, we perform a trade-off analysis for an extended number of processing units,

both COTS and rad-hard. We consider image processing and computer vision algorithms for VBN, i.e., computationally intensive benchmarks, while we pay particular attention to the performance and power parameters. We define a set of representative benchmarks by studying the application scenario of VBN and collecting generic specifications and commonly used algorithms. Subsequently, we survey, test in-house, and compare a wide range of microchips falling into seven main categories: space-grade CPUs, embedded CPUs, desktop CPUs, mobile GPUs, desktop GPUs, single- and multi-core DSPs, and FPGAs. Notice that, besides selecting a specific device, it is of utmost importance to determine which of these architectural categories is the most promising for the future mission scenarios and thus should be further supported/exploited. Put into context, our preliminary work in ESA has shown that although multi-core DSPs are gaining the attention of the space industry with $2-3$ platforms currently being developed/considered, they still have insufficient processing power for the high-performance applications. For example, ESA considers internally the space-oriented scalable sensor data processor (SSDP) platform based on a LEON3-FT together with Xentium DSPs [12], but this does not meet the high-performance requirements of VBN. Also, ESA considers the latest space-grade GR740 CPU, which is also not sufficient for VBN, as parallelism cannot be exploited and the projected clock speed is still too low. In the past, the most frequently considered type of hardware for intensive applications were FPGAs, i.e., Xilinx Virtex-5QV and Microsemi RTG4, or the very recent European NanoXplore NX-eFPGA (dubbed BRAVE or NG-MEDIUM) [13]. Today, besides space-grade devices, cutting-edge technology includes system-on-chip (SoC) off-the-shelf devices, which constitute a very attractive solution for certain space missions, like, for example, ESA's e.Deorbit ADR [14].

The survey and analysis performed in the current paper will be useful for a variety of missions. It provides an in-depth exploration of a vast solution space, which can then be given as input to a focused trade-off for the selection of avionics based on the custom-weighted criteria of a particular mission. For simplicity and in order to allow the consideration of many devices, we assume here generic mission specifications with demanding VBN, like in the ADR scenario: low-earth orbit, short-term life and increased VBN autonomy. Such specifications call for increased accuracy/flexibility VBN, high-resolution images and high frame-rates, all of which contribute to

increased computational workload. The assumed high-level architecture is preliminary, based on previous implementations/experience, state-of-the-art image processing and available technologies, as well as the future road-maps.

The paper is organized as follows: Section II outlines the computer vision algorithms commonly employed for VBN with an emphasis on ADR, Section III surveys the computing platforms available for use in space, Section IV performs the benchmarking and comparison of the devices, and Section V highlights the most important results and concludes the paper.

## II. Algorithms for Vision-based Navigation in Orbit

Keeping track of the position of an orbiting spacecraft with respect to its surroundings is one of the key ingredients of reliable spacecraft navigation. Accurate localization information is necessary to ensure that the spacecraft follows its intended path and operates in proximity with other objects while avoiding collisions [15]. The present section reviews computer vision techniques which estimate the relative pose between a chaser and a target space resident spacecraft in the context of autonomous rendezvous, inspection or docking scenarios involving orbiting spacecraft [16]. Despite that the scope of this review focuses on the ADR scenario, it is pointed out that other applications of vision-based navigation in space rely on analogous principles and employ similar visual primitives. Such applications, for example, include obstacle avoidance, mapping and visual odometry for planetary rovers [17, 18] or approach velocity estimation and localization for precision landing [18–20]. As a result, the choice of a processing platform suitable for ADR is also highly relevant to visual navigation applications for planetary exploration rovers and landers. Fig. 2 depicts some indicative applications of VBN for rovers (top-left) and ADR (bottom-right): an image acquired by a Mars rover (top-left) is processed by 3D reconstruction (shown to its right, "mapping" via stereo matching) and visual odometry estimation (shown below, "localization" via feature detection and matching between successive images, annotated with circles and lines, respectively) [17], whereas edges detected on a satellite and matched against its geometric model (bottom-right, red outline) serve as features for estimating the satellite's pose relative to the camera [21].

Relative pose refers to the 6 DoF position and attitude parameters defining the geometrical relationship between the two spacecrafts. The pose estimates can be used in a closed control loop

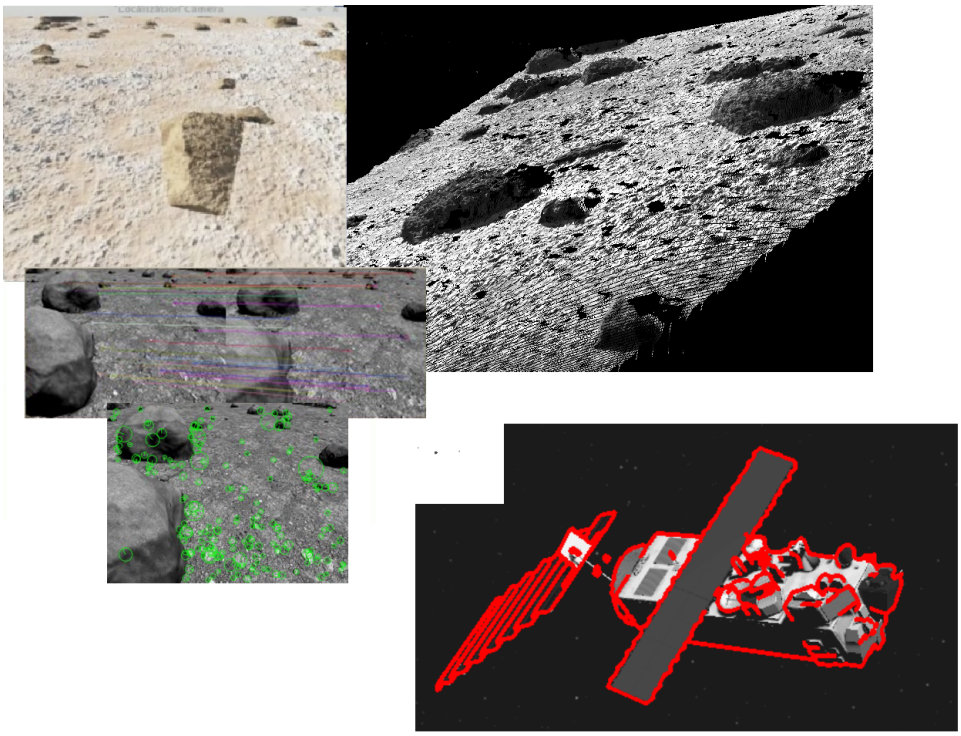

**Fig. 2 Example outputs of computer vision algorithms employed in space: navigation for planetary rovers (top-left) and pose for ADR targeting the ENVISAT satellite (bottom-right).**

to guide the motion of the chaser spacecraft in order to safely approach the target or collect it as debris. Relevant approaches can be characterized by the type of sensor they employ, their operating range and by whether the target spacecraft is "cooperative", i.e. carries markers or reflectors that are installed on it with the aim of facilitating pose estimation by the chaser spacecraft. The desired operating distance relates to the choice of sensor, sensory processing method and the type of information which can be extracted from the input [22]. The established consensus is that a single sensor will not perform adequately across the entire operating range, therefore autonomous rendezvous and docking requires the deployment of a suite of multiple sensor systems, each with a different range and potentially principle of operation [1]. The most commonly used types of sensors are passive cameras and laser range finders.

In this review, we limit ourselves to passive camera sensors applied to pose estimation at distances below 100 m. The choice of cameras is motivated by the fact that they are attractive for use in ADR due to their compact size, low cost, simple hardware, low power consumption and

potential for dual use as navigation and visual inspection instruments. Furthermore, we focus on non-cooperative targets since ADR will almost certainly concern not specially prepared or partially unknown targets.

Estimating the pose of a target in these settings is quite challenging, because no strong prior information regarding its attitude and position is available. The situation is aggravated due to the varying illumination conditions, largely varying chaser-target distances that result in dramatic aspect changes, self-occlusions, background clutter, unstabilized target motion and measurement noise. Such challenges call for elaborate and, therefore, computationally demanding processing of images. A variety of algorithmic approaches to visual pose estimation with the purpose of docking have been proposed. We note that some of these rely on prior knowledge for pose initialization, i.e. cannot handle lost-in-space configurations. These approaches as well as the majority of working systems account for the temporal continuity of a target's motion by using tracking [23]. In this respect, by integrating past estimates, tracking can deliver a better estimate compared to that directly determined with single frame pose estimation methods. At the same time, it can provide a prediction for the next frame which typically accelerates pose estimation since it bounds considerably the pose search space. Vision-based pose estimation approaches can be broadly classified into two categories, namely monocular and stereo-based, depending on whether they are based on images from a single or several cameras. A comprehensive discussion is offered by Petit [24] and a brief overview follows next.

### A. Monocular approaches

Monocular approaches to pose estimation employ a single camera and typically rely on the assumption that a 3D model of the target is known beforehand. Depending on the approach used, the 3D model can be in the form of an unordered set of points (i.e., point cloud) and their image descriptors, a wireframe, or a polygon mesh. Use of a properly scaled model helps to resolve the well-known depth/scale ambiguity [25]. Monocular approaches infer pose by matching features detected in an image with the model. Before providing more details, a short introduction to image features is given first.

Specific structures in an image give rise to features. For example, sparse distinctive points,

also called “interest points”, “keypoints” or “corners” in the vision literature, refer to salient image locations with sufficient local variation [26]. Keypoints can be accurately localized in an image and their local descriptors can be robustly matched from moderately different viewpoints. Edges are another type of feature occurring when intensity changes sharply. Edges correspond to maxima of the gradient magnitude in the gradient direction [27] and are moderately robust against noise, illumination, and viewpoint changes. In general, point and edge detection is computed by using 2D convolutions with derivative filters. Popular algorithms for the detection of keypoint and edge features are those by Harris [28] and Canny [27], respectively.

Feature-based approaches to monocular pose estimation operate by matching the detected image features with certain locations from the target's known 3D model. Pose is then estimated so that it accounts for the observed arrangement of the detected features in an acquired image. Proposed approaches differ according to whether the image-to-model correspondences are explicitly or implicitly established. Explicit methods strive to establish (usually one-to-one) correspondences of keypoint locations, relying on their feature descriptors or their spatial arrangement [29–32]. Implicit methods attempt the establishment of correspondences through the matching of less localized features, such as edges, contours, or line segments [21, 33–36]. Aiming to further increase robustness, approaches such as [37–39] which combine edge and point features have also been proposed. We note that despite the impressive results obtained in various application domains with state of the art keypoint detectors and descriptors, their use in space is hindered by the fact that the target objects of interest lack strong texture and are often subject to illumination changes and specular reflections. Therefore, approaches based on edges are more frequently employed for ADR.

A few approaches such as [40–42] track features over time and employ simultaneous localization and mapping (SLAM) or structure from motion (SfM) techniques to estimate the 3D structure and dynamics of the target object. Pose is again computed by matching observed features to the estimated structure. Such approaches do not require any prior information about the appearance of the target, therefore are most suitable when the state of the latter is uncertain or unknown. However, their increased flexibility comes at the price of higher complexity and risk of failure.

### B. Stereo approaches

Stereo vision employs triangulation to provide estimates of the distances from the camera of the target's visible surfaces. The primary challenge in order to achieve this concerns the determination of correspondences among the pixels appearing in a binocular image pair [43]. Correspondence establishment involves local searches for identifying the minimum of a matching cost function that measures the affinity of pixels in the two images. For efficiency, these searches are restricted in one dimension along image scanlines, after the input images have been rectified. Commonly used stereo matching costs are the sum of squared differences (SSD), the sum of absolute differences (SAD) or the normalized cross correlation (NCC) [43]. Knowledge of the baseline, i.e. the distance between the cameras, via calibration, permits scale to be determined.

To estimate pose from stereo data, either the actual surface or a set of 3D features extracted from it are matched with a 3D model of the target (see [44] and references therein). Stereo sensing is sensitive to the lack of texture and possible illumination artifacts, such as shadows and specular reflections, which are common in space environments. Furthermore, the camera resolution limits the depth resolution, i.e. the smallest change in the depth of a surface that is discernible by the stereo system. Depth resolution is also proportional to the square of the depth and is inversely proportional to the baseline. For these reasons, stereo measurements are more accurate at fairly short distances from a target. This is in contrast to monocular approaches presented in Section II A, which are also usable at the far end of the distance range.

Indicative stereoscopic approaches to pose estimation are briefly discussed next. Stereo images are used in [45] to reconstruct 3D data which are then matched with an object model using the iterative closest point (ICP) algorithm to estimate pose. Sparse 3D point clouds recovered with stereo have been employed to estimate pose, for example in [46] where the 3D points originate from natural keypoints and [47] which determines image points as the intersections of detected line segments. To achieve more dense reconstruction, [48] suggests the combination of conventional feature-based stereo with shape-from-shading. It is noted that all stereo approaches still need to solve the problem of establishing correspondences between the 3D data and the object model. Approaches which estimate pose by using stereo to detect particular structures on the target satellite

have also been proposed, e.g. [49] extracts line segments to identify a triangular solar panel support whereas [50] relies on locating a circular feature. Clearly, such approaches do not generalize to arbitrarily shaped targets.

## III. Survey of Processing Platforms for Space Applications

Hitherto, the majority of space missions rely on rad-tolerant and rad-hard IBM PowerPC and Gaisler LEON CPUs. Especially in European missions, LEON is primarily used for implementing critical functions, e.g., GNC, whereas PowerPC is preferred as payload in less critical situations, e.g., on board rovers. In terms of power consumption, the LEON-based solutions are less demanding than PowerPC, e.g., $1-2$ Watts versus $10-20$ Watts, respectively. In terms of radiation resilience, one can find reliable implementations in both families, e.g., withstanding $100-1000$ Krad (Si) TID and suffering only $10^{-6}$ single event upset (SEU) per bit-day. In terms of performance, space-grade CPUs always lack $1-2$ orders of magnitude from their commercial counterparts; the commercial PowerPC-750 road-map showed a performance improvement of $\sim$2400x from the 90's to 2005 [51], while the rad-hard version was improved by only $\sim$300x. Similarly, the relatively new space-qualified dual-core Leon3-FT GR712RC provides only $140-200$ DMIPS, whereas the older Intel Pentium III provides 2K DMIPS. The latest 64-bit quad-core RAD5545 and SPARC V8 E698PM devices can deliver up to $2-5$K DMIPS performance, i.e., they provide a tenfold increase compared to their predecessors. Table 1 summarizes space-grade processors from the two families (powerPC, LEON).

LEON evolved from ERC32 and TSC695FL SPARC V7 processor (20 DMIPS at 25 MHz) to the more advanced AT697F 32-bit LEON2-FT SPARC V8 from Atmel ($43-70$ DMIPS at $50-80$ MHz). Developed by Saab Aerospace, the COLE SoC bases on LEON2-FT (Fault-Tolerant) and includes space interfaces such as MIL-1553, CAN, and SpaceWire. Airbus [52][53] uses the LEON4 High Reliable Processor Board (HRP-board) providing up to 800 MIPS, as well as the LEON3-based SCOC3 SoC with considerably worse performance. RamonChip [54] implemented one of the most advanced dual-core LEON3-FT devices, namely the GR712RC, which is fabricated on 180nm CMOS and provides up to 200 DMIPS/MFLOPS at 100 MHz, dissipates less than 2 Watts and withstands up to 300 Krad TID with SEU tolerance and SEL above 118 Mev-cm2/mg. Today, the top LEON-based processor is LEON4-FT, e.g., the GR740 at 65nm, which includes a quad-core operating at

**Table 1 CPU-based processors for space with rad-hard (RH) and rad-tolerant (RT) devices.**

| | Device | Process Node | CPU Freq. | DMIPS | Power Diss. | Radiation Tolerance |
|---|---|---|---|---|---|---|
| PowerPC | RT RAD750 (BAE Systems) | 0.25$\mu m$ radiation tolerant | 110-130 MHz | 260 @133MHz (80 MFLOPS) | 14W @133MHz | TID: 200 Krad (Si) SEU: $<1.6 \cdot 10^{-10}$ err/bitday Latch-up immune |
| | RH RAD750 (BAE Systems) | 0.15$\mu m$ radiation hardened | 200 MHz | 400 @200MHz | 14W @133MHz | TID: 1 Mrad (Si) SEU: $<1.6 \cdot 10^{-10}$ err/bitday Latch-up immune |
| | SCS750 SBC (3 PPC750FX) (Maxwell) | 0.13$\mu m$ SOI (TRP) | 400-800 MHz | 1800 @800MHz (200@400MHz) | 7-25W | TID: 100 Krad (Si) SEL: $>80 MeV\text{-}cm^2/mg$ (or $50 MeV$ for SDRAM) |
| | RAD5545 64-bit 4-core (BAE Systems) | 45 nm SOI | TBD | 5200 MIPS (3700 MFLOPS) | 17.7W | TID: 1Mrad (Si) SEU: $<2 \cdot 10^{-9}$ err/bitday Latch-up immune |
| hardcore LEON | AT697E SPARC V8 (Atmel) | 0.18$\mu m$ CMOS | 100 MHz | 86 @100MHz | 1W @100MHz | TID: 60 Krad (Si) SEU: $<10^{-5}$ err/dev/day SEL: $>70\ MeV.cm^2/mg$ |
| | TSC695F SPARC v7 (Atmel) | 0.50 $\mu m$ SOI | 10-25 MHz | 20 @25MHz | 1W @25MHz | TID: 300 Krad (Si) SEU: $<3 \cdot 10^{-8}$ err/dev/day SEL: $>70\ MeV.cm^2/mg$ |
| | GR712RC Dual-Core SoC (RamonChip) | 180 $nm$ CMOS | 100 MHz | 140-200 DMIPS (200 MFLOPS) | 1.5Wx2 | TID: 300 Krad (Si) SEL: $>118\ MeV\text{-}cm^2/mg$ SEU tolerant |
| softcore LEON | GR-PCI-XC5V SPARC V8 | 65$nm$ | 50 MHz | 70 DMIPS (50 MFLOPS) | TBD | RH by design (Virtex5QV tolerance) |
| | LEON3FT-RT3PE on RT ProASIC3 (Gaisler, Actel) | 130 $nm$ | 20-25 MHz | 20 DMIPS | 0.2W | TID: $15 - 25$ Krad (Si) SEU: $>6\ MeV\text{-}cm^2/mg$ SEL: $>96\ MeV\text{-}cm^2/mg$ |
| higher-performance & multicores | P4080 Board | | 1.5 GHz | 27.6 DGIPS | 45W | Latch-up immune Virtex-5 Voting System |
| | Intel Atom TMR Board | | 1.6 GHz | 3.3 GIPS | 25W | Radiation Tolerant |
| | High Reliable Processor Board (Airbus, SPARC V8 LEON4, Virtex-5 I/O, PikeOS) | | 150 MHz | 800 MIPS | TBD | TBD |
| | GR-CPCI-GR740 (4-core LEON4FT+2 FPU) | | 250 MHz | 425 DMIPS | 1.8W | Latch-up immune (RH-bd, on ST's 65nm) |
| | OCE E698PM (RH, 4-core 32-bit SPARC v8, MAC/UMAC DSP 64-bit double precision FPU) | | 600 MHz | 2100 MIPS (900 MFLOPS) | 2.8W | TID: >100 Krad (Si), SEL: $>120\ MeV.cm^2/mg$ SEU: $>37\ MeV.cm^2/mg$ |

250 MHz and delivering 425 DMIPS per core; this multicore LEON4 targets up to 4,300 DMIPS at 800 MHz and supports both asymmetric and symmetric multiprocessing configurations (AMP or SMP). Regarding PowerPCs developed by BAE, the RAD6000 single board computer (SBC) and the RAD750 are the most widely used. RAD750 is implemented on 150nm rad-hard bulk CMOS and clocked at 200 MHz with 400 DMIPS and 14 Watts.

The aforementioned CPUs can be implemented either as softcores, e.g., in an FPGA or as hard IPs on ASIC. However, it becomes clear that softcores provide considerably lower processing speed, which is around 20−70 MIPS. For example, LEON3FT-RTAX Aeroflex Gaisler on Microsemi RTAX2000s delivers 20 DMIPS, operating at 25MHz and consuming only 0.5 Watts. Hence, softcores can only serve as a secondary solution for hardware/software (HW/SW) co-design of VBN. On the contrary, ARM-based ASIC CPUs integrated in new SoC FPGAs can provide more than 1K MIPS even with single-core execution. Processors from ARM company are now being considered in multiple European space projects, e.g., ARM4Space (H2020), ASCOT (CNES), as well as for the on-board computer of the Ariane 6 launch vehicle. ARM's Cortex-A9 processor is suitable for low-power applications and has been used in multiple mobile phones, as well as in CubeSats. NASA [55] has exposed the Snapdragon APQ8064 IFC6410 board to Ar @ LET = 7 MeV-cm2/mg. However, up to now, only the ARM Cortex-M0 has been implemented as radiation tolerant by triplication, to provide no more than 50 DMIPS. In more powerful CPU-based scenarios, the P4080 bases on octo-core PowerPC plus Xilinx Virtex-5 FPGA for voting to provide up to 27.6 DGIPS, however, at the cost of more than 25 Watts power dissipation. At similar power budget, the Intel Atom TMR board provides up to 3.3 GIPS with radiation tolerance being provided through triplication. Finally, we mention that ESA's GAIA mission uses a video processing unit with a SCS750 board (commercial PPC750FX) providing 1,800 DMIPS at 800 MHz, but consuming 33 Watts and requiring mitigation via triple redundancy of the PPC plus voting with Actel FPGA to withstand radiation.

Beyond the CPU-based approach, the most common general-purpose accelerators are FPGAs, VLIW DSP multi-core processors, as well as GPUs, which, however, are not widely considered for space applications due to their increased power dissipation. The VLIW DSP solution is being examined through European projects such as MACSPACE (with 64 X1643 DSP cores from CEVA)

**Table 2 DSP-based processors (space-grade and COTS).**

| Device | Process Node | CPU Freq. | Power Diss. | DMIPS | Radiation Tolerance |
|---|---|---|---|---|---|
| MPPB/SSDP (TAS-I & Recore, LEON2 + 2/4 Xentium DSPs) | 90$nm$ | 80 MHz | 1.5W (min.) | TBD | TBD |
| TI SMV320C6727B (C67x+ single-core VLIW DSP) | 130$nm$ | 250 MHz | TBC | 1.5 GFLOPS | TID 100 Krad (Si) & SEL immune up to LET=117$MeVcm^2/mg$ |
| TI 66AK2H12 SoC (4-core ARM Cortex-A15 + 8-core C66x DSP) | 28$nm$ | 1.4 GHz | 10W | 160 GFLOPS | Not space-qualified |
| MACSPACE RC64 Manycore (GR712RC Dual-core LEON3 + 64x CEVA X1643 DSP cores) | 65$nm$ | 300 MHz | 10W | 38 GFLOPS | Targeting space (300 Krad TID & SEL immune) |

and MPPB/SSDP [56] (with Xentium VLIW DSP cores from Recore), or devices such as the 8-core C66x DSP Texas Instruments 66AK2H12, as summarized in Table 2. The conventional single-core VLIW space-grade DSP devices (e.g., TI SMV320C6727B) deliver inadequate processing power to be considered for high-performance applications. Instead, TI 66AK2H12 offers two orders of magnitude more processing power, however, without space-qualification yet. Similarly, at only 10 Watts, the European MACSPACE 64-core DSP will also provide one order of magnitude higher performance than single-DSP solutions.

Regarding space-grade FPGAs (Table 3), given the increased requirements of image processing in terms of logic and memory, the most suitable devices for VBN seem to be the Xilinx Virtex-5QV and the Microsemi RTG4. Between these two, the most preferable is Virtex-5QV due to its double on-chip memory, which is a critical resource for image processing. The space-grade European market offers today the ATMEL devices, which are considered quite small for high-performance tasks. However, the recently introduced European NanoXplore NX-eFPGA should be able to support high-performance tasks in the near future. Also in the near future, it is expected

that the Zynq Ultrascale+ SoC FPGA will qualify as rad-tolerant and provide more FPGA resources than any other space-grade device, e.g., 10x more logic and 7x more memory compared to Virtex-5QV. Currently, multiple research groups consider avionics architectures utilizing FPGAs, even with COTS devices. More specifically, when combining trends such as high-performance, SoC, and COTS, the most reasonable solution studied is the Xilinx Zynq FPGA [57, 58]. The JPL proposes for NASA the APEX [57], an FPGA-based SoC, which is being prototyped on Zynq-7Z100. Next to the Zynq chip, APEX integrates unvolatile SSD and DDR memories (Solid State and Double Data Rate), all on ZedBoard mini-ITX, plus an FMC connected card providing ADC functionality. Additionally, the authors of [57] propose to combine certain functionality of the COTS Zynq such as internal watchdogs with mitigation techniques on the FPGA side in order to increase the resilience of the system to space radiation. Overall, the volume of this avionics module is $17 \times 17 \times 5$ cm$^3$, whereas its power dissipation is around 5 Watts. The CHREC space processor [58] is also based on Xilinx Zynq7000. The developed board includes rad-hard flash memory and has a form factor of 1U. This means that it fits in a $10 \times 10 \times 11.35$ cm$^3$ cubesat, weighing less than 0.1Kg and consuming around 2 Watts of electrical power. NASA's Goddard centre proposes using SpaceCube [59, 60], which is an architecture based on space-grade FPGAs (e.g., Virtex-5QV). They develop a hybrid computing platform that consists of a CPU, DSP processors and FPGA logic, with various versions already tested in a space environment. "SpaceCube mini" is the smallest module, with a form-factor 1U consisting of three U-folded boards, one of which includes the Xilinx Virtex-5QV as the main processing unit, weighing less than 1.4 Kg (with a 2.5mm thick housing), and consuming around 5 Watts, whereas its estimated TDI is $50-700$ Krad(Si). "Spacecube 2.0" is larger, including 4 boards mounted in a rack of 3U form factor and having a volume of $12.7 \times 17.8 \times 22.9$ cm$^3$, a mass of 5.8Kg and consumption of 20 Watts. Finally, EREMS has developed the ELISE main on-board computer based on Zynq-7030, which dissipates 5 Watts, is SET/SEU/SEL protected and withstands $\sim$5 Krad.

Following the preceding survey, we summarize a number of high-level choices for designing a high-performance architecture with: 1) many-core CPU, 2) CPU plus multi-core DSP, 3) CPU plus FPGA, 4) multi-FPGA including soft-core CPU, and 5) System-on-Chip. Choices $2-4$ assume dis-

**Table 3 FPGA-based processors (space-grade and COTS).**

| FPGA | Logic | Memory | DSPs | Power | Techn. | Radiation Tolerance |
|---|---|---|---|---|---|---|
| Xilinx Virtex-5QV | 82K LUT6 | 596 RAMB16 | 320 | 5-10 W | 65nm SRAM | SEE immune up to LET 100MeV/mg/cm2 & 1 Mrad (Si) TID |
| Microsemi RTG4 | 151K LE | 5.3 Mbit | 462 | 1-4 W | 65nm Flash | SEE immune up to LET 110Mev/mg/cm2 & TID >100Krad |
| Microsemi ProASIC3 | 35K LE | 0.5 Mbit | - | ∼1 W | 130nm Flash | 40 Krad TID |
| Microsemi RTAX | 4M gates (∼37K LUT) | 0.5 Mbit | 120 | TBD | 150nm Antifuse | SEL immune up to 117MeV/mg/cm2 & 300 Krad TID |
| ATMEL ATFEE560 | 560K gates (26K LUT4) | 0.23 Mbit | - | TBD | 180nm SRAM | SEL immune up to 95MeV/mg/cm2 & 60 Krad TID |
| NanoXplore NG-MEDIUM | 34K LUT4 | 56 RAMB48 | 112 | TBD | 65nm SRAM | SEL immune at 60MeV/mg/cm2 & 300 Krad TID |
| Xilinx Zynq7000 SoC | 277K LUT6 | 1510 RAMB18 | 2020 | 3-6 W | 28nm SRAM | Not space-qualified |
| Microsemi SmartFusion2 SoC | 146K LE | 4.5 Mbit | 240 | ∼2 W | 65nm Flash | Not space-qualified |
| Xilinx Zynq Ultrascale+ SoC | ∼800K LUT6 | ∼4K RAMB16 | ∼2K | TBD | 16nm SRAM | Rad tolerant (by 2018) |

tinct devices interconnected with, e.g., a high-speed network like SpaceWire at 100 Mbps. The drawback of such an approach would be the increased size, mass and power dissipation due to the multiple chips involved. Moreover, co-processing bottlenecks could arise in cases 2−4 due to congestion or slow interconnects among the devices, which would result in limited communication

bandwidth between intermediate steps of an algorithm. When considering multicores running SW, delivering a hard real-time system implies functional isolation together with time analysis against single-core chips. This is difficult due to state interruption between processes/applications, constraints of high-criticality over low-criticality applications as well as safety and worst-case execution time requirements of SW, as is the common practice in the space industry. Instead, FPGAs can avoid such complicated SW analysis, although they require increased development time. Softcore CPUs (e.g., LEON synthesized in an FPGA) provide one order of magnitude lower performance than hardcore IPs integrated in a SoC device (e.g., ARM Cortex A9 in Zynq), which can execute more demanding functions in a HW/SW co-design scenario. By taking into account the above factors, we identify the SoC approach as the most promising for high-performance embedded computing. SoC devices offer a wide range of advantages owing to the coupling of acceleration resources with general purpose processors and peripherals. They can lead to architectures with single-device, low-power, small size/weight and rapid intra-chip communication able to support efficient co-processing.

## IV. Comparison and Evaluation of Processing Platforms

This section compares candidate platforms, primarily, in terms of throughput, power, performance per Watt, and secondarily, in terms of connectivity, size/mass, and error mitigation opportunities. We evaluate their capabilities in relation to image processing and computer vision tasks, with the intention of selecting the most suitable platform for the future space missions under consideration. Towards a fair/accurate comparison, we combine various literature results and in-house development/testing, while we utilize a number of benchmarks common to all platforms. Table 4 summarizes the devices and algorithms used in our study. Specifically, we examine 30 representatives from all four processor categories (namely CPU, FPGA, GPU and DSP) by utilizing more than 10 well-known benchmarks applied to images of size 512x384 and 1024x1024 pixels. These benchmarks were selected as a representative set of the operations required by algorithms surveyed in section II.

The comparison of such diverse platforms becomes very challenging due to the dependence of the relative performance of each benchmark on every device on such factors as: a) the distinctiveness/peculiarities of each computational & programming model, b) the programming effort, coder's

**Table 4 Summary of platforms and benchmarks considered in our comparative study**

| | | In-house development & testing | Literature survey |
|---|---|---|---|
| **platforms** | *FPGA* | Xillinx Virtex6 VLX240T-2, Zynq7000 (Z7020, Z7045) | Altera Stratix III E260, Xilinx Virtex 5QV Virtex4 VLX100, Virtex6 VSX475T, Zynq7000 |
| | *CPU* | desktop Intel i5-4590, laptop i3-4010U<br>embedded: ARM CortexA9, Intel X1000<br>space: LEON3, OCE E698PM (LEON4) | desktop: Intel i7-3820, AMD FX-8120<br>embedded ARM CortexA15<br>space-grade BAE RAD5545 |
| | *GPU* | Nvidia GeForce GTX 670, GTX 680, GTX 960 | Nvidia GTX 980Ti / 650Ti / 295, Tesla C2050, mobile: Nvidia Tegra K1/X1, ARM Mali-T760 |
| | *DSP* | space-grade Xentium MPPB<br>embedded multi-core Myriad2 | TI multi-core TMS320C6678 and 66AK2H14, 1-core C674x, 64-core RH RC64 (MACSPACE) |
| **benchmarks** | | 2D convolutions (5x5 to 11x11), Harris-corner & Canny-edge detectors, Stereo Matching, Hyperspectral search, Pose Estimation (incl. feature detection, description, matching), Super-Resolution | 2D convolutions and SAD (up to 25x25), Harris-corner & Canny-edge detectors, Stereo Matching, Image Denoising and Block Matching, Hyperspectral imaging, *etc.* (plus nominal DMIPS and MFLOPS figures) |

experience and code optimization level of each benchmark, c) the application/algorithmic domain targeted by each architecture, and d) the technology node and size of each chip. To tackle these issues, along with the plethora of chip−benchmark combinations that emerge, we follow a two-stage pruning method tailored to our goals. First, we perform a general "CPU vs. GPU vs. DSP vs. FPGA" evaluation, and then we proceed to a more in-depth comparison of the $4-5$ most promising chips. To address d) in the second stage, we focus on $28nm$ technology node and, in particular, SoC devices. To address c) in both stages, we focus on image processing. To address b), we select published works that devote equal effort to their examined platforms, avoiding comparisons of optimized CUDA vs. generic High-Level-Synthesis implementations, and we assign distinct, experienced, developers to each platform tested in-house. To address a), we rely on the multitude of results and their convergence to consistent conclusions by experienced developers. Finally, we note that our primary goal is to derive the relative performance of the devices rather than absolute numbers

for specific algorithms/datasets. Thus, we avoid running the exact same test on all devices, which would become impractical for 10 benchmarks, 2 image sizes and 30 chips. Instead, we proceed by extracting multiple individual performance ratios for key pairs of devices and then, as explained in section IV C, we combine all ratios/comparisons in a unified table.

For clarity of presentation, sections IV A and IV B report separately the most important results derived from the literature and the measurements performed in-house for both stages of the aforementioned comparison strategy. These sections focus on performance and power consumption, while combining absolute numbers and extrapolations/estimations to facilitate straightforward comparisons between the devices. Subsequently, section IV C merges all results to provide a unified perspective. Additionally, section IV C considers factors like size/mass of the COTS board, development time, COTS cost, and discusses suitability to space missions.

### A. Literature results

The literature includes a multitude of implementations on various devices, which we use here to construct our first set of performance comparisons. Primarily, we collect results regarding platforms not available in-house and, secondarily, we include papers to enrich and verify our own tests.

Regarding mobile GPUs, the authors in [61] perform a wide platform comparison which, among others, includes Nvidia Tegra and ARM Mali devices. They employ a set of nine computer graphics benchmarks (L-Bench) with various workloads. On average, their results indicate that Tegra K1 (365 GFLOPS) is performing 1.89x faster than ARM Mali-T760 MP6 (143 GFLOPS). However, the Mali-T760 MP6 consumes ∼6 Watts, whereas K1 consumes ∼10 Watts, and hence, these mobile GPUs provide similar performance per Watt. We note here that the Mali GPU is integrated in the latest MPSoC FPGAs of Xilinx; the MPSoC uitlize Mali-400 providing only 21 GFLOPS and performing one order of magnitude slower than K1. The latest Tegra X1 provides 1.4x higher throughput compared to Tegra K1 (1−1.8x according to various benchmarks [62]), and hence, we can infer that these high-end mobile GPUs span a performance range of $1 - 3$x. The authors in [63] compare the mobile Tegra K1, to desktop GPU Nvidia GTX 680, to the many-core DSP TI 6638K2K, and to Intel Core i7-3770K CPU, by employing a set of sparse matrix-vector multiplication benchmarks. On average, they show that K1 is 9x slower than GTX 680, however, it achieves almost similar

performance with the single-threaded desktop CPU; they use Intel's MKL library for parallelization on CPU, but we use here single-thread extrapolations to facilitate our own comparisons. The authors in [64] implement a compression method for deep neural networks and evaluate the performance of their pruned network on Intel i7-5930K CPU, Nvidia GeForce GTX Titan X, and Tegra K1, also employing optimized libraries and MKL, from which we make rough estimations for single-threaded execution. They show that Tegra K1 is ∼3x faster than the single-threaded CPU and one order of magnitude slower than Titan X, however, with nominal 250 Watts. Similarly, for telecom benchmarks including data transfer, the authors in [65] show that mobile GPUs are up to 10x slower than desktop GPUs (Tegra K1 vs. Nvidia GTX 680/780ti/970).

Regarding embedded CPUs, the authors in [66] consider the performance gain obtained via single instruction, multiple data (SIMD) processing and the comparison of conventional CPUs to embedded ARM processors, such as Cortex-A9 (integrated in SoC devices, e.g., OMAP 4460, Exynos 4412, Tegra T30). Their results indicate that, for the single-threaded benchmark of simple binary thresholding on images, the Core i5-3360M CPU achieves 20−24x faster execution compared to Cortex-A9 (reduced at 667MHz), while for a more complex benchmark such as edge detection, the i5 CPU is 21−32x faster than the embedded Cortex-A9. The authors in [67] use N-body simulation as a benchmark and show that, for single-threaded execution, the Intel i5-2467M is 12.8x faster than the embedded Cortex-A9 (at 667MHz). Furthermore, they employ optimized CUDA coding (Compute Unified Device Architecture) on GPUs to show that the mobile Tegra K1 is 61.8x faster than Cortex-A9, whereas high-end desktop GPUs achieve a huge gain over A9, in the area of 1000x (e.g., 2032x vs. Nvidia Tesla K80 with nominal 300 Watts). We note that, in a broader sense [67, 68], the throughput of embedded CPUs could increase even by 10x when we consider higher clock rates, e.g., 2.3 GHz, and/or bigger micro-architectures, e.g., ARM Cortex A-57. However, for the sake of categorization, we focus here on more conservative scenarios, such as Cortex A9/A15 operating below 1 GHz.

Regarding high-end many-core DSPs, such as TI's TMS320C6678 and 66AK2H which are both utilizing the C66x core, the authors in [69] and [70] compare the 8-core C66x to ARM Cortex A15 and get ∼9x faster execution for various benchmarks (4−10x). By reducing MIPS and MHz, as well

as by considering the results of [67], we derive that the A15 processor at 1.2GHz is 3−5x faster than the Cortex A9 of Zynq (at 667MHz), which we use hereafter as reference. Hence, we deduce that 66AK2H14 is ∼36x faster than A9. For hyperspectral unmixing, when comparing the 8-core C6678 at 1 GHz to ARM cortex A9 at 1 GHz, the authors in [71] derive slightly smaller factors than the above averaged ∼36x. If we adjust their ARM A9 to 667 MHz and their 8-core C66x to 1.2 GHz as in 66AK2H, we conclude that the DSP is up to ∼25x faster than the 1-core ARM CPU. The authors in [72] use a stereo matching benchmark, which in terms of complexity is roughly similar to our own "Disparity" that is detailed in [73], on a 1-core C66x to achieve 14 fps for small images. They also show in [74] that the speedup to 8-cores is ∼4x, and hence, they would achieve ∼56 fps on 66AK2H14. By making extrapolations from our own Virtex6 "Disparity" to emulate their complexity (we assume 434x380 resolution images with 18 candidate disparities per pixel and by compute only one map per image, thus, we decrease our workload by 11x) and by employing 4-engine parallelization on Zynq (to consume 1/4 cycles at 350 MHz instead of 283 MHz clock, due to the technology scaling), we estimate a throughput of ∼337 fps on FPGA. Thus, we get a coarse comparison showing that Zynq is ∼6x faster than 66AK2H14 for "Disparity". For the Harris corner detection benchmark, considering *cvCornerHarris* on the C674x core [75], we can roughly estimate that 66AK2H14 will complete a 1 Megapixel image in ∼50msec, and hence, it is ∼4x slower than our Zynq implementation (see section IV B 2). The authors in [76] test the 8-core TMS320C6678 and show that, for frequency-domain FIR, it achieves 1/3 throughput vs. Nvidia Tesla C2050, whereas for 25x25 convolution kernel, the TMS320C6678 achieves similar throughput with Nvidia GTX 295, or 1/3 throughput compared to FPGA Stratix III E260, or 1/9 compared to our own Zynq-7000 implementations reported in sec. IV B 2 of the current paper (90msec vs. 10msec for 1080p images). It is also shown in [76] that code optimization is important, as it boosts the DSP performance by 2−3x. Finally, based on the results of [63] for matrix multiplication benchmarks, we conclude that 66AK2H14 achieves similar throughput with Tegra K1 GPU and single-threaded Intel Core i7-3770K. Regarding the 64-core DSP computer RC64 "MACSPACE" [77], in lieu of relative benchmarking, we consider the nominal values: 10 Watts for 20 single-precision GFLOPS (75GMACS/150GOPS) when operating at 300MHz. This FLOPS value is half than that of 66AK2H14 (GMACS are similar) when also

operating at 300MHz. Hence, for mixed fixed/floating point arithmetic benchmarks, we expect that RC64 will achieve at most similar throughput with 66AK2H14.

Regarding direct comparisons of GPU and FPGAs, the authors in [78] perform an extensive literature survey for such platforms for stereo matching benchmarks. They examine dozens of algorithms and implementations, but their results concern devices which are now 5−10 years old, e.g., Intel Core2duo, Nvidia GTX 280, and TI C64x+. When looking at the cloud of results collected in their study, as a general conclusion one can remark that GPUs and FPGAs are difficult to distinguish with respect to throughput: using the paper's metrics for the fastest 50% of the implementations in each category, GPUs achieve 0.1−7 Gde/sec, whereas FPGAs achieve 1−6 Gde/sec. This picture is consistent with our own findings. Similarly, for older devices and various image processing benchmarks such as stereo matching, face detection, object detection and beamforming, some authors like [79] show ±10% throughput differences when comparing specific devices, whereas others show differences around 2−4x, either favoring GPUs [80] or FPGAs [81]. The survey in [78] also includes DSP processors, which however are 1−2 orders of magnitude slower than GPUs and FPGAs. This gap seems to be narrowing due to the latest high-end many-core DSPs considered in the current paper (see section IV C). Considering vision algorithms for space [82], the desktop GPU achieved one order of magnitude higher performance/Watt than the desktop CPU, i.e., up to 12x with respect to GFLOPS and 49x with respect to GFLOPS/Watt. Finally, we note that for GPU-GPU comparisons, e.g., to combine results among papers and derive indirect comparisons, or to extend/support our own conclusions, we occasionally consider the variety of GPU benchmarks available online [62].

### B. Development and In-house Testing

#### *1. CPU*

Our in-house development and testing involves a number of CPUs ranging from small space-grade and embedded devices to big desktop/laptop platforms. The C/C++ benchmarks used are Harris corner detection (employing five 2D convolutions [17, 28]), hyperspectral searching (of pre-stored spectral signatures in a pixel-by-pixel fashion [83] by using metrics such as $\chi^2$, $L_1$, $L_2$, cf. [84]),

Canny edge detection [27] and binocular visual odometry [17] (including distinct algorithms for feature detection, feature description, temporal and spatial matching, pose estimation [85], outlier removal, etc.). The code used is fairly optimized and common among devices, with a reasonable level of compiler optimization (e.g., gcc version 4.6.3 with the -O3 flag), and the execution is single-threaded in all cases, so that more straightforward comparisons can be made.

Harris was implemented on embedded Intel Quark X1000 using a Galileo board [86], embedded ARM Cortex A9 on Zynq7020 at 667 MHz [87], on desktop Intel Core i5-4590 and i3-4010U and i7-3820, as well as on space-grade LEON3 implemented as soft-core on FPGA at 50 MHz and LEON4 on chip OCE E798PMPM at 200 MHz. The execution time for a 512x384 image was $1054-2177$ms on X1000, depending on the use of yocto linux 3.8.7 OS or bare-metal, 456ms on ARM A9 with petalinux and 10-20ms on the various Intel Core devices with ubuntu linux. On the space-grade devices with RTEMS OS, we obtained 5sec on LEON3 and 0.55sec on LEON4. This common benchmark provides an accurate relative performance evaluation of the platforms showing that space-grade CPUs are slower than embedded ones, which in turn are $25-50$x slower than desktop CPUs. This 25x factor is also derived, on average, when using our hyperspectral benchmark to compare ARM A9 to the small desktop CPUs (i.e., Intel i3-4010U and i3-3110M on laptops): depending on the benchmark configuration, the execution time for 1 Mega-pixel image ranges from $0.4-62$sec on i3 to $3.5-1702$sec on A9. When testing Canny on 1 Mega-pixel greyscale images, the relative performance of desktop CPUs becomes smaller than above, i.e., only 10x versus ARM and 40x versus X1000 (51ms on i5, 502ms on A9, 2120ms on X1000).

For the space-grade CPUs, to evaluate the performance gain of the latest LEON generation, Table 5 analyzes the execution time of our visual odometry benchmark [17] processing stereo pairs of 512x384 pixels. We remark that the first three columns/functions are executed twice and the entire pipeline is completed in 0.25sec on Intel i5-4590. On average, with a limited deviation among functions, the LEON4 achieves a 9.2x gain due to its 4x higher clock rate and improved architecture. If we assume further clock increase to 600MHz and very effective parallelism to 4 cores, then we could anticipate future gains of even up to 100x. Even so, we note that such impressive performance improvement will not be sufficient for high-definition image processing at high frame-rates. For

**Table 5 Profiling of a complete visual odometry (pose estimation) pipeline on space-grade LEON3 and latest LEON4 (E698PM) CPUs; pipeline stage descriptions are provided in [17].**

| | Harris | ANMS | SIFT | Match-T | Match-S | other | Total |
|---|---|---|---|---|---|---|---|
| LEON3 (1-core, 50MHz) | 5 sec | 15.3 sec | 29.7 sec | 20 sec | 4.8 sec | $< 5$ sec | 129 sec |
| LEON4 (1-core, 200MHz) | 0.55 sec | 1.7 sec | 2.7 sec | 2.3 sec | 0.32 sec | $< 1$ sec | 14 sec |
| gain | 9x | 9x | 11x | 8.7x | 15x | – | 9.2x |

example, with the given algorithm, we would process only $\sim$1 fps of size 512x384, which is one order of magnitude less than, e.g., 1 Mega-pixel at 5 fps. That is, the space-grade CPUs are still $1-2$ orders of magnitude slower than necessary for computationally demanding vision-based navigation, and hence, some form of acceleration is essential.

### *2. FPGA*

To perform in-house evaluation of high-performance FPGA platforms, we accelerate multiple benchmarks on Xilinx Virtex6 and Zynq-7000 devices with hand-coded hardware description language (VHDL). The majority of these accelerators were originally developed and carefully optimized within completed ESA projects targeting vision-based navigation for planetary rovers [88]. First, we focus on the Harris benchmark, which includes diverse operations, such as 2D convolutions, evaluation of floating point formulas, and repetitive comparisons. Our design bases on sliding windows, deep pipelining at pixel-basis, and high parallelism of arithmetic operations, such that we can process all intermediate values/pixels with a rate of one value per cycle. The VHDL design is an optimization/reorganization of the architecture detailed in [17]. Here, we consider 1024x1024 8-bit images and we deploy 4 parallel Harris engines, one per image quadrant. Moreover, to account for the usual HW/SW co-design and I/O scenarios of SoC devices, we assume that the accelerators in the Programmable Logic (PL) are fed by the Processing System (PS) and we develop a PS-PL communication scheme based on AXI and Xillibus [89]. Depending on burst size, the achieved bandwidth is $1.3-2.6$ Gbps, and hence, a 1 Mega-pixel image is transferred in around 3.2msec. The total resources utilized by Harris on Zynq-7045 PL are 30612 LUTs (12%), 43100 DFFs (10%), 186

RAMB36 (34%) and 216 DSPs (24%). By operating at 200MHz, the processing time is 8.8msec and the power is 4 Watts, distributed as 0.6W per Harris engine and 1.6W for the ARM processor. Therefore, Zynq-7045 is 233x faster than Cortex A9 (2.8sec per image) and $5-9$x faster than desktop CPUs.

Furthermore, we also test stereo matching, which is a very intensive benchmark involving highly repetitive calculations. Specifically, our stereo matcher concerns bi-directional disparity map generation with 7x7 Gauss-weighted cost aggregation and its architecture is detailed in [73]. For one pair of 1120x1120 8-bit pixel images with 200 disparity levels, a 4-engine parallelization on Virtex6 VLX240T requires 0.54sec. Therefore, the FPGA is 29x faster than Intel i5 (see [90] for detailed comparison) and approximately 7700x faster than space-grade LEON3; this factor was derived after extrapolation from smaller images due to memory limitations on LEON3. Such huge factors can be also derived for Harris detection and SIFT description. For the former, comparing the aforementioned 12msec total time of Zynq-7045 to the almost 27sec of LEON3, which is an extrapolation based on image size and Table 5, leads to a factor of $\sim$2250x. Even when we estimate the performance on a space-grade FPGA like Virtex-5QV (52% LUT and 55% RAM utilization, with 14.7ms processing time due to 120MHz clock), the total speedup is still $\sim$1500x. For the latter, we optimized our SIFT descriptor architecture which is presented in [17], to use fixed-point arithmetic and smaller description window (33x33), and hence, we improved the execution time by 30% on Virtex6 to consume 28msec when processing 1200 features. When porting to Zynq-7045 (200MHz) and assuming 4-engine parallelization, even if we add the 3.2msec overhead of PS-PL communication, then we obtain less than 10ms total execution time, i.e., an estimated $\sim$3000x speedup versus LEON3 (vs. SIFT in Table 5).

To test more streaming applications relying on continuous pixel I/O, we assume a faster PS-PL communication scheme than the one implemented above using Xillibus [89]. We assume the 4 AXI VDMA scheme developed by Xillinx [91], which achieves 12 Gbps transmission rate and another 12 Gbps reception. In our hyperspectral benchmark that involves matching of incoming images to known spectral signatures, we develop in VHDL a large systolic network of processing elements to parallelize the pixel subtraction and metric accumulation (e.g., for $L_1$, the $\sum_i |x_i - y_i|$), as well as

the comparison to distinct pixel-signatures. When we assume ordinary application scenarios and up to 12 Gbps PS-PL communication, the Zynq-7045 PL operates up to 428MHz and accelerates its own ARM Cortex A9 by 340−630x. For more complex scenarios, e.g., more elaborate metrics like $\chi^2$, or hundreds of examined signatures, or 100 Gbps bandwidth, the FPGA acceleration would approach $10^4$x, however, in the current paper we focus on the more general case for fair comparisons.

### *3. GPU*

Based on the same C/C++ benchmark executed on the CPUs, we implemented Harris in CUDA and tested it on our desktop GPU Nvidia GTX 680 (Kepler microarchitecture at 28nm, 1536 cores at 1GHz). Our implementation uploads a 512x384 image to the GPU and sequentially executes the steps of Harris by employing parallelism on a pixel basis, i.e., one pixel per thread. Harris involves separable convolutions for image derivatives ×2, squared derivatives computation ×3, Gauss smoothing ×2, and cornerness calculation. For all kernels, we use the GPU cache for reading the pixels and the constant memory of the GPU to store the convolution kernels. The GPU achieves a throughput of around 1000 fps (i.e., 1ms per image), with 1/3 of the time spent on processing (e.g., 0.052 msec for a single 2D convolution with a 7x7 separable kernel, or 0.033 msec for a 5x5 kernel) and the rest 2/3 dedicated to communication. Our 4-engine FPGA Harris with 200MHz clock would spend for the same image 1.6ms on PS and 0.13ms on PS-PL communication (assuming 12 Gbps bandwidth), therefore, the GPU is faster than Zynq by a factor of 1.8x. In terms of performance per Watt, the Zynq FPGA outperforms the GPU by a factor of more than 10x. The GTX680 is 10−20x faster than desktop CPUs and up to 500x faster than ARM A9. Compared to high-end mobile GPUs, we expect/estimate the GTX680 to be 6−10x faster than Tegra X1 due to 7x bigger memory bandwidth and 6x more CUDA cores.

Besides Harris, we also use image super resolution (SR) and stereo matching ("Disparity" [73]) to compare desktop GPUs to FPGAs. We tested a variety of devices and performed in-depth optimization of the implemented benchmarks on both platforms; a detailed analysis has been published separately [90]. Here, we consider Nvidia GTX 670/960, whereas for the FPGA (Zynq7000 Artix) we consider both processing and communication time, in contrast to the study in [90] that focuses

on processing. The assumed 12 Gbps PS-PL bandwidth on Zynq can support continuous operation of only up to 3 SR processing engines at 120MHz or 2 engines operating at 180 MHz. Hence, a 1920x1080 image can be upsampled by Zynq in 5.7ms. Hypothetically, the FPGA performance could increase by up to 10x if we assume a larger chip and a more efficient means of communication feeding the fabric with the very high I/O of SR. The GPUs can process the same image in 2.9−9ms with 67−76 Watts measured power, i.e., they are up to 2x faster than FPGA. However, for smaller images, due to limited occupancy, GPUs can become 10x slower. For the stereo matching benchmark, as mentioned in section IV B 2, our FPGA requires 0.54sec and our GPU implementations on GTX 670/960 require 0.5−0.36sec. Notice that in this benchmark, the I/O is not a bottleneck and processing could be supported even with 1 Gbps communication. Moreover, on Zynq-7045, due to the technology scaling (vs. Virtex6) we can expect/estimate the execution time to decrease to 0.4−0.5sec and provide almost equal throughput with the high-end GPU.

#### *4. DSP*

To perform in-house evaluation of high-end many-core DSP-oriented platforms, we implemented Harris corner detection on the 12-core VLIW Movidius Myriad2 [92]. By applying a number of source code modifications to exploit the underlying architecture, we efficiently utilized the limited local memory and exploited the long instructions of Myriad2. In particular, we first identified the most frequently accessed data structures and placed them in the local memory. Second, we identified and removed storage redundancies. Third, we partitioned the image in 12 slices and assigned them to distinct cores. Fourth, we applied platform-oriented modifications, such as employing one 2D convolution kernel instead of two 1D separable filters, aiming to improve HW utilization and decrease memory accesses. Steps 1 and 3 yielded the highest performance improvements, i.e., 35x gain with 25x less energy with respect to the initial porting. The final result shows that, for 1 Mega-pixel image, execution on 12 VLIWs at 600MHz requires 110ms and 1 Watt. Therefore, compared to the aforementioned 4-engine Zynq7045 implementation, Myriad2 is 9x slower, but only 2x worse in terms of performance per Watt due to its very low power consumption. We note that, when compared to Zynq-7020 processing 512x384 images with only 1 Harris engine, i.e., when compared

against smaller devices with limited room for parallelization and smaller input, Myriad2 achieves similar performance per Watt and is only 2.5x slower than the FPGA (execution times are 8ms/2W on Zynq-7020 vs. 20ms on Myriad2, which approaches the Intel i5 performance).

Besides accelerating Harris on 12 VLIWs, we evaluated Myriad2's integrated ASIC accelerator. We note that the computational complexity in this version of Harris is lower than our own Harris benchmark, which also includes convolutions for Gaussian blurring. This Myriad2 Harris consumes 116 msec and 0.25 Watts for the 1 Mega-pixel image, i.e., we still get one order of magnitude slower execution than Zynq. In addition to Harris, we also tested on 12 VLIWs a native/example SW function for Canny edge detection (also not the same with our own Canny benchmark) to derive 25ms and 1.2W for a 1 Mega-pixel image. Therefore, compared to ARM Cortex A9 at 667 MHz, which needed 2.8sec and 0.5sec for Harris and Canny respectively on a 1 Mpix image, we deduce that Myriad2 is 20−25x faster than such an embedded CPU. This factor increases to the area of 30x for a single 11x11 convolution, when Myriad2's time becomes similar even to that of Zynq, which suffers from its PS-PL communication overhead when sending small workloads to PS. Finally, we note that the clock rate on Myriad2 is programmable, e.g. 200−600MHz, with the overall performance decreasing proportionally and the power consumption decreasing down to 0.5W.

In addition to high-end DSP, we also test the space-oriented massive parallel processor board (MPPB) [56], which integrates 2 VLIW Xentium DSPs running at 50 MHz. We ran simulations using the Xentium SDK environment to develop/profile 4 optimized code versions (specifically fixed-/floating-point, C and assembly) for 2D convolutions. Notice that these simulations serve only as a best-case scenario, because they discard communication delays, as well as LEON, AMBA and NoC latency. The final results show that MPPB requires 96−223 msec for one convolution with separable kernels of size 5x5 or 7x7 on a 1024x1024 8-bit image. Such performance is one order of magnitude lower than FPGAs, which are in the area of 10 msec or less. Furthermore, given that our Harris benchmark requires two 5x5 convolutions and three 7x7 convolutions, we estimate that MPPB requires almost 1 sec for 1 Mega-pixel image. Hence, currently, it outperforms LEON4 by only 3x, whereas if we make use of multi-threading on LEON4, their performance will probably be similar. We note here that MPPB is expected to improve in future versions, e.g., get a 100 MHz

clock. However, this is not sufficient to bridge the gap with Myriad2 or FPGAs.

### C. Combination and Summary of Results

Table 6 provides a rough comparison of all platforms in terms of performance (normalized throughput), power, and performance per Watt, by grouping the platforms in seven categories/columns. Each table cell provides a range of values to capture all the results derived for the designated category (they vary per benchmark and device). In the first row, we normalize the relative performance of all devices such that values around 1 correspond to rad-hard CPUs and values in the area of 1000 correspond to the fastest accelerators. More specifically, we begin with our pair of LEON processors showing ∼9x performance gap –on average– and we assign them reference values 0.2 and 1.7, respectively. Subsequently, we consider that Zynq is ∼2250x faster than LEON3 for Harris and we assign a sample value of 450 ($= 0.2 \cdot 2250$). The procedure continues in this fashion until all individual relative performances for all benchmarks have been incorporated in Table 6 (with rounding/simplifications where necessary). Therefore, all results are unified in a linear scale such that the ratio of any two values in Table 6 equals the performance ratio of the selected platforms. The second row reports nominal/measured power and the third row reports performance per Watt. This is the ratio of the first two rows, however, it is calculated separately for each device, because the range limits in each category do not necessarily correspond to the same device, e.g., in rad-hard CPUs, throughput= 1.7 represents E698PM, but power= 18 represents RAD5545. Notice that the purpose of Table 6 is to provide succinctly a coarse overview of the vast solution space by focusing, mainly, on the high-performance image processing capabilities of each category. It is intended neither as an exhaustive evaluation of all industrial chips nor as a rigorous comparison between categories. Such a more accurate comparison is provided in Table 7, for five selected devices, as a result of the second stage of our evaluation methodology.

As shown in Table 6, for single-threaded execution, the space-grade CPUs achieve performance that is comparable to that of the embedded CPUs, albeit consuming more power. Altogether, space-grade and embedded CPUs are $1-2$ orders of magnitude slower than desktop CPUs, which however suffer from size limitations and high power consumption and become inappropriate for space applications. Overall, the CPUs achieve the lowest, by $1-3$ orders of magnitude, performance per

**Table 6 Coarse comparison of performance (normalized throughput) & power for seven categories of processing platforms (devices summarized in Table 4, e.g., E698PM@200MHz, A9@667MHz, etc., results vary with benchmark and device).**

| | rad-hard CPU (1-core) | embedded CPU (1-core) | desktop CPU (1-core) | mobile GPU | high-end DSP | FPGA | desktop GPU |
|---|---|---|---|---|---|---|---|
| Throughput | 0.2−1.7 | 0.5−2 | 20−100 | 50−150 | 50−240 | 300−1460 | 200−2000 |
| Power (W) | 1−18 | 1−2 | 20−90 | 6−10 | 1−10 | 2−10 | 70−195 |
| Perf/Watt | 0.1−0.6 | 0.25−2 | 0.5−1 | 8−15 | 12−50 | 60−250 | 5−25 |

Watt ratio, consequently they are not deemed as suitable candidates for the application scenario of the current paper. The high-end mobile GPUs overcome the aforementioned power limitations and provide similar performance with desktop CPUs by consuming only ∼10 Watts. However, within this low power budget, the high-end DSPs can improve even further the performance per Watt of the desktop-CPU and mobile-GPUs, hence they should be preferred. The desktop GPUs have great potential for speed, but require excessive amounts of power. Compared to mobile GPUs, they trade one order of magnitude of power for speed. Hence, overall, either due to relatively low perf/Watt or increased power and size, the GPUs are not considered as a good candidate for the application scenario we are concerned with. The FPGAs achieve the highest, by at least one order of magnitude, performance per Watt of all platforms, and even if we include the I/O overhead of the FPGA (e.g., PS-PL communication), they achieve almost the highest throughput that is close to, or in some cases even better than, that of desktop GPUs. Therefore, given the overview of Table 6, we now focus on FPGAs and high-end multi-core DSP processors as being the most suitable choices for the space avionics architecture.

Table 7 focuses on one representative FPGA (Xilinx Zynq7045) and two multi-core DSP processors (namely 12-core Movidius Myriad2 and 8-core TI 66AK2H14), which for fairness are all SoC devices at $28nm$ technology node. In addition, for reference, Table 7 includes one of the latest

**Table 7 Comparison of Performance (normalized throughput) & Power for selected SoC at 28nm technology (excl. rad-hard LEON4). Based on multiple benchmarks for images of 1024x1024 8-bit pixels (in-house and literature results).**

| | 4-core LEON4 (E698PM) 600 MHz | 12-core VLIW (Myriad2) 600 MHz | 8-core DSP (66AK2H14) 1200 MHz | FPGA (Zynq7045) 200−300 MHz | desktop GPU (GTX 670/680/960) $\geq 1$ GHz |
|---|---|---|---|---|---|
| Performance | $< 20$ | 50 | 70−240 | 430−1460 | 600−1800 |
| Power (W) | $\simeq 2.8$ | 1 | $\simeq 10$ | 4−6 | $> 70$ |
| Perf/Watt | $< 7$ | 50 | $7 - 24$ | 110−240 | $< 25$ |

rad-hard CPUs (OCE E698PM, also SoC) and desktop GPUs (Nvidia GTX 670/680/960, also at $28nm$). The results base on multiple benchmarks with a fair level of optimization on each platform. Moreover, we highlight that the performance includes all communication overheads (PS-PL for Zynq, memories for Myriad2, etc), i.e., we evaluate each SoC device as a whole HW/SW system. We note that the throughput of E698PM is hypothetical, i.e., it is extrapolated from our measurements by assuming very efficient 4-core parallelization and 600 MHz clock. This represents a best case scenario for the performance of near-future space-grade CPU, which in practice could prove even 10x lower. For Zynq, the wide throughput range is due to the varying nature of our benchmarks, which result in various acceleration factors vs. space-grade CPUs. Specifically, the small factors correspond to feature extraction and large factors to stereo matching. For GPUs, the range is also due to distinct chips, whereas for 66AK2H14, it accounts for diverse results from the literature. According to Table 7, the FPGA achieves 10x better performance per Watt than all platforms except Myriad2, which is however almost 10x slower than FPGA when considering high-definition images and increased FPGA parallelization/resources. If we assume a power budget of 10 Watts, then the FPGA would be the fastest choice, with 66AK2H14 being 4−6x slower. If we assume an even more limited power budget, e.g., 1−3 Watts, then Myriad2 improves the CPU throughput by ∼3x (or even by one order of magnitude considering the results of Table 5 instead of the aforementioned extrapolation). Nevertheless, the FPGA could further improve this throughput

by 10x by paying only a small penalty in power dissipation. That is, the FPGA can achieve $1-2$ orders of magnitude faster execution than any space-grade CPU with comparable power consumption. On the other hand, even with throughput=20, the CPU will doubtfully meet the increased demands of future vision-based navigation. Altogether, today's multi-core DSP architectures approach the FPGA figures in terms of performance and performance/Watt, however, not simultaneously for the same device. For instance, we must choose either Myriad2 or 66AK2H14 depending on whether power or speed is the preferred criterion. Moreover, we note that the DSP values of Table 7 are derived with maximum clock frequency, i.e., 1.2 GHz for 66AK2H14. When/if this rate is decreased during space flight, e.g., for reliability reasons, then the speed gap between DSPs and FPGA will widen proportionally; we observe here that the relatively low clock rate is an advantage of FPGAs. Therefore, when both power and performance are important, the FPGA is the most effective architecture/solution currently available.

Beyond performance and power, we consider factors such as the connectivity of the COTS board, radiation tolerance, size/mass, re-programmability, development effort, and cost. The majority of available COTS boards provide multiple interfaces like Eth, USB, PCIe, etc. However, the FPGA boards offer the advantage of extra GPIO and FMC pins allowing us to communicate with custom connectors and daughter-boards. A second advantage of FPGAs relates to the various mitigation techniques that can be applied for error correction due to radiation [93]; these mitigation techniques can be applied almost at gate-level within the same FPGA, and hence, more efficiently than in the case of CPU/core-based processors applying, e.g., triple core lock-step. Considering programmability, the dynamic reconfiguration of SRAM FPGAs renders them almost as useful as the remaining many-core platforms, even for remote updating of their firmware. Considering size/mass, the FPGAs are among the most competitive COTS boards (e.g., $5.72 \times 10.16\ cm^2$ for the Zynq MMP Zedboard, or $7.8 \times 4.3 \times 1.9\ cm^3$ for the Zynq Xiphos Q7, with a mass of only $65-24$ g). On the downside, as already mentioned, the development effort is increased for FPGAs by a factor of $\sim$4x compared to SW platforms. Still, efficient programming of many-core chips with multiple levels of parallelization, also requires an increased amount of effort compared to conventional, serial SW coding.

## V. Conclusion

The current paper performed a detailed survey and trade-off analysis involving an extended number of diverse processing units competing as platforms for space avionics. The set of candidate platforms included both rad-hard and COTS devices, of older and recent technology, such as CPUs, GPUs, multi-core DSPs, and FPGAs. Gradually, our analysis focused on high-performance embedded platforms, which were also compared to more conventional devices for the sake of perspective. The application scenario considered was visual based navigation, for which we performed a distinct exploration to collect generic specifications and widely used algorithms, i.e., to define a set of representative benchmarks. Overall, the profiling of each device depended both on the benchmark complexity and the underlying HW architecture, and hence, the relative performance of the devices varied greatly among groups and created overlapping clouds of results (with size expanding even by 10x per group). The challenge was tackled in our comparative study by combining numerous literature results with in-house development/testing, which ultimately led to a big, consistent picture.

The results show that new generation space-grade CPUs are 10x faster than their predecessors, however, they are still one order of magnitude slower than what will be needed for reliable autonomous VBN. Therefore, we must design high-performance avionics architectures utilizing HW accelerators. In particular, instead of utilizing multiple chips, it is preferable to utilize SoC devices, which integrate general purpose processors and HW accelerators towards size/mass minimization, power reduction, fast intra-communication, re-programmability, and increased connectivity. Separated by orders of magnitude, the FPGA accelerators provide the highest performance per Watt across all platforms, whereas the CPUs provide the lowest, irrespective of their type. In terms of speed alone, high-end desktop GPUs and FPGAs are difficult to distinguish (as groups, they provide similar clouds of results). Likewise, high-end mobile-GPUs and many-core DSPs are difficult to distinguish, although the latter have the potential for slightly better performance and power. In terms of speed alone, desktop GPUs and FPGAs are clearly better than mobile-GPUs and many-core DSPs. In terms of power, desktop CPUs and GPUs seem prohibitive for space avionics, however, a 10 Watt budget is enough to allow many-core DSPs or mobile-GPUs or FPGAs to accelerate a

conventional space-grade CPU by 1−3 orders of magnitude. In such a high-performance embedded computing scenario, with relaxed constraints on radiation tolerance due to mission specifications, it would be preferable to utilize COTS 28nm SoC FPGAs, which provide 2−29x faster execution than 28nm DSP processors.

## Acknowledgments

This work has received support from the European Space Agency (ESA) via the project "HIP-NOS": High Performance Avionics Solution for Advanced and Complex GNC Systems (ESTEC ref. 4000117700/16/NL/LF).

## References

[1] Bonnal, C., Ruault, J.-M., and Desjean, M.-C., "Active Debris Removal: Recent Progress and Current Trends," *Acta Astronautica*, Vol. 85, 2013, pp. 51–60.

[2] Fortescue, P., Swinerd, G., and Stark, J., *Spacecraft Systems Engineering*, John Wiley & Sons, 2011.

[3] Pignol, M., "DMT and DT2: two fault-tolerant architectures developed by CNES for COTS-based spacecraft supercomputers," *International On-Line Testing Symposium (IOLTS'06)*, 2006.

[4] Pignol, M., "COTS-based Applications in Space Avionics," *Proceedings of the Conference on Design, Automation and Test in Europe (DATE'10)*, 2010, pp. 1213–1219.

[5] Selva, D. and Krejci, D., "A Survey and Assessment of the Capabilities of Cubesats for Earth Observation," *Acta Astronautica*, Vol. 74, No. C, 2012, pp. 50–68.

[6] Furano, G. and Menicucci, A., "Roadmap for On-Board Processing and Data Handling Systems in Space," *Dependable Multicore Architectures at Nanoscale*, edited by M. Ottavi, D. Gizopoulos, and S. Pontarelli, Springer, 2018, pp. 253–281.

[7] Maurer, R. H., Fraeman, M. E., Martin, M. N., and Roth, D. R., "Harsh Environments: Space Radiation Environment, Effects, and Mitigation," *Johns Hopkins APL Technical Digest*, Vol. 28, No. 1, 2008, pp. 17–29.

[8] Geller, D., "Orbital Rendezvous: When Is Autonomy Required?" *Journal of Guidance Control and Dynamics*, Vol. 30, 07 2007, pp. 974–981.

[9] TEC-T, "GSTP Element 1 "Develop" - Compendium of Potential Activities 2017: Clean Space," ESA-GSTP-TECT-PL-005452, May 2017.

[10] Liou, J.-C., "Engineering and Technology Challenges for Active Debris Removal," *EUCASS Proceedings Series*, Vol. 4, March 2013, pp. 735–748.

[11] Flores-Abad, A., Ma, O., Pham, K., and Ulrich, S., "A Review of Space Robotics Technologies for On-orbit Servicing," *Progress in Aerospace Sciences*, Vol. 68, Jul. 2014, pp. 1–26.

[12] R.Pinto et al., "Scalable Sensor Data Processor: Architecture and Development Status," `https://indico.esa.int/indico/event/102/session/22/contribution/35/material/slides/0.pdf`, DSP Day, ESA's AMICSA 2016, Gothenburg.

[13] Lepape, E., "NanoXplore NXT-32000 FPGA (NG-MEDIUM) Presentation," `https://indico.esa.int/indico/event/130/session/11/contribution/59/material/slides/0.pdf`, 3rd SEFUW workshop, ESTEC, ESA, 2016.

[14] Biesbroek, R., Innocenti, L., Wolahan, A., and Serrano, S. M., "e.Deorbit - ESA's Active Debris Removal Mission," *European Conference on Space Debris*, 2017.

[15] Tweddle, B. E., *Computer Vision-Based Localization and Mapping of an Unknown, Uncooperative and Spinning Target for Spacecraft Proximity Operations*, Ph.D. thesis, Dept. of Aeronautics and Astronautics, MIT, Cambridge, Mass, USA, 2013.

[16] Woffinden, D. and Geller, D., "Navigating the Road to Autonomous Orbital Rendezvous," *Journal of Spacecraft and Rockets*, Vol. 44, 06 2007, pp. 898–909.

[17] Lentaris, G., Stamoulias, I., Soudris, D., and Lourakis, M., "HW/SW Codesign and FPGA Acceleration of Visual Odometry Algorithms for Rover Navigation on Mars," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 26, No. 8, Aug 2016, pp. 1563–1577.

[18] Matthies, L., Maimone, M., Johnson, A., Cheng, Y., Willson, R., Villalpando, C., Goldberg, S., Huertas, A., Stein, A., and Angelova, A., "Computer Vision on Mars," *Int. J. of Comp. Vision*, Vol. 75, No. 1, 2007, pp. 67–92.

[19] Van Pham, B., Lacroix, S., and Devy, M., "Vision-based Absolute Navigation for Descent and Landing," *J. Field Robot.*, Vol. 29, No. 4, July 2012, pp. 627–647.

[20] Delaune, J., Besnerais, G. L., Voirin, T., Farges, J., and Bourdarias, C., "Visual-inertial Navigation for Pinpoint Planetary Landing Using Scale-based Landmark Matching," *Robot. Auton. Syst.*, Vol. 78, No. C, April 2016, pp. 63–82.

[21] Lourakis, M. and Zabulis, X., "Model-based Visual Tracking of Orbiting Satellites Using Edges," *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2017, pp. 3791–3796.

[22] English, C., Okouneva, G., Saint-Cyr, P., Choudhuri, A., and Luu, T., "Real-Time Dynamic Pose Estimation Systems in Space: Lessons Learned for System Design and Performance Evaluation," *Intl.*

*Journal of Intelligent Control and Systems*, Vol. 16, No. 2, 2011, pp. 79–96.

[23] Lepetit, V. and Fua, P., "Monocular Model-Based 3D Tracking of Rigid Objects: A Survey," *Foundations and Trends in Computer Graphics and Vision*, Vol. 1, No. 1, 2005.

[24] Petit, A., *Robust Visual Detection and Tracking of Complex Objects: Applications to Space Autonomous Rendez-vous and Proximity Operations*, Ph.D. thesis, Universite Rennes 1, 2013.

[25] Lourakis, M. and Zabulis, X., "Accurate Scale Factor Estimation in 3D Reconstruction," *Intl. Conf. on Computer Analysis of Images and Patterns (CAIP)*, Springer Berlin Heidelberg, 2013, pp. 498–506.

[26] Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Gool, L. V., "A Comparison of Affine Region Detectors," *Int. J. Comput. Vision*, Vol. 65, No. 1-2, Nov. 2005, pp. 43–72.

[27] Canny, J., "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 8, No. 6, Nov. 1986, pp. 679–698.

[28] Harris, C. and Stephens, M., "A Combined Corner and Edge Detector," *Proceedings of the 4th Alvey Vision Conference*, 1988, pp. 147–151.

[29] Vacchetti, L., Lepetit, V., and Fua, P., "Stable Real-Time 3D Tracking Using Online and Offline Information," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 26, No. 10, 2004, pp. 1385–1391.

[30] Collet Romea, A. and Srinivasa, S., "Efficient Multi-View Object Recognition and Full Pose Estimation," *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2010, pp. 2050–2055.

[31] Augenstein, S., *Monocular Pose and Shape Estimation of Moving Targets for Autonomous Rendezvous and Docking*, Ph.D. thesis, Dept. of Aeronautics and Astronautics, Stanford University, Palo Alto, CA, 2011.

[32] Lourakis, M. and Zabulis, X., "Model-Based Pose Estimation for Rigid Objects," *Intl. Conf. on Computer Vision Systems (ICVS)*, Springer-Verlag, 2013, pp. 83–92.

[33] Drummond, T. and Cipolla, R., "Real-Time Visual Tracking of Complex Structures," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 24, No. 7, 2002, pp. 932–946.

[34] Comport, A. I., Marchand, É., Pressigout, M., and Chaumette, F., "Real-Time Markerless Tracking for Augmented Reality: The Virtual Visual Servoing Framework," *IEEE Trans. Vis. Comput. Graph.*, Vol. 12, No. 4, 2006, pp. 615–628.

[35] Petit, A., Marchand, E., and Kanani, K., "Tracking Complex Targets for Space Rendezvous and Debris Removal Applications," *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2012, pp. 4483–4488.

[36] D'Amico, S., Benn, M., and Jørgensen, J., "Pose Estimation of an Uncooperative Spacecraft from Actual Space Imagery," *International Journal of Space Science and Engineering*, Vol. 2, No. 2, 2014, pp. 171–189.

[37] Vacchetti, L., Lepetit, V., and Fua, P., "Combining Edge and Texture Information for Real-Time Accurate 3D Camera Tracking," *Proc. Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, 2004, pp. 48–57.

[38] Rosten, E. and Drummond, T., "Fusing Points and Lines for High Performance Tracking," *Proc. IEEE Intl. Conf. on Computer Vision (ICCV)*, Vol. 2, IEEE Computer Society, 2005, pp. 1508–1515.

[39] Choi, C. and Christensen, H. I., "Real-time 3D Model-based Tracking Using Edge and Keypoint Features for Robotic Manipulation," *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2010, pp. 4048–4055.

[40] Augenstein, S. and Rock, S. M., "Improved Frame-to-Frame Pose Tracking During Vision-Only SLAM/SFM with a Tumbling Target," *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011, pp. 3131–3138.

[41] Tweddle, B. E., Saenz-Otero, A., Leonard, J. J., and Miller, D. W., "Factor Graph Modeling of Rigid-body Dynamics for Localization, Mapping, and Parameter Estimation of a Spinning Object in Space," *Journal of Field Robotics*, Vol. 32, No. 6, 2015, pp. 897–933.

[42] Post, M. A., Li, J., and Clark, C., "Visual Pose Estimation System for Autonomous Rendezvous of Spacecraft," *Symp. on Advanced Space Technologies in Automation and Robotics (ASTRA)*, 2015.

[43] Scharstein, D. and Szeliski, R., "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *Int. J. Comput. Vision*, Vol. 47, No. 1-3, 2002, pp. 7–42.

[44] Zabulis, X., Lourakis, M., and Koutlemanis, P., "Correspondence-free Pose Estimation for 3D Objects From Noisy Depth Data," *The Visual Computer*, Vol. 34, No. 2, 2018, pp. 193–211.

[45] Jasiobedzki, P., Greenspan, M., and Roth, G., "Pose Determination and Tracking for Autonomous Satellite Capture," *Int. Symp. on Artificial Intelligence and Robotics & Automation in Space (i-SAIRAS)*, 2001.

[46] Yu, F., He, Z., Qiao, B., and Yu, X., "Stereo-Vision-Based Relative Pose Estimation for the Rendezvous and Docking of Noncooperative Satellites," *Mathematical Problems in Engineering*, Vol. 2014, No. 11, 2014, pp. 1–12.

[47] Li, R., Zhou, Y., Chen, F., and Chen, Y., "Parallel Vision-based Pose Estimation for Non-cooperative Spacecraft," *Advances in Mechanical Engineering*, Vol. 7, No. 7, 2015, pp. 1–9.

[48] Carlo, S. D., Prinetto, P., Rolfo, D., Sansonne, N., and Trotta, P., "A Novel Algorithm and Hardware Architecture for Fast Video-based Shape Reconstruction of Space Debris," *EURASIP Journal on Advances in Signal Processing*, Vol. 2014, No. 1, 2014, pp. 147.

[49] Xu, W., Liang, B., Li, C., and Xu, Y., "Autonomous Rendezvous and Robotic Capturing of Non-cooperative Target in Space," *Robotica*, Vol. 28, No. 5, 2010, pp. 705–718.

[50] Xu, W., Xue, Q., Liu, H., Du, X., and Liang, B., "A Pose Measurement Method of a Non-cooperative GEO Spacecraft Based on Stereo Vision," *International Conference on Control Automation Robotics Vision (ICARCV)*, 2012, pp. 966–971.

[51] Hillman, R., Swift, G., Layton, P., Conrad, M., Thibodeau, C., and Irom, F., "Space Processor Radiation Mitigation and Validation Techniques for an 1,800 MIPS Processor Board," *7th European Conference on Radiation and Its Effects on Components and Systems, (RADECS)*, 2003, pp. 347–352.

[52] Notebaert, O., "Research & Technology activities in on-board data processing domain," `https://indico.esa.int/indico/event/85/session/12/contribution/96/material/0/0.pdf`, Avionics Technology Trends Workshop, ESTEC, ESA, 2012.

[53] Airbus, "High Reliability Processing Node (HRPN-LEON4)," `http://microelectronics.esa.int/gr740/Airbus-ST-NGMP-Board-Flyer-HRPN.pdf`.

[54] Sturesson, F., Gaisler, J., Ginosar, R., and Liran, T., "Radiation characterization of a dual core LEON3-FT processor," *Radiation and Its Effects on Components and Systems (RADECS), 2011 12th European Conference on*, IEEE, 2011, pp. 938–944.

[55] Guertin, S. M., "CubeSat and Mobile Processors," *NASA Electronics Technology Workshop*, 2015, pp. 23–26.

[56] Pinto, R., Sanchez, P., Berrojo, L., Sunesen, K., Potman, J., and Rauwerda, G., "On-Board Compression Using a Next-Generation Multicore DSP Architecture," *5th International Workshop on On-Board Payload Data Compression OBPDC 2016, ESA ESRIN, Frascati (Rome - Italy)*, 2016.

[57] Iturbe, X., Keymeulen, D., Ozer, E., Yiu, P., Berisford, D., Hand, K., and Carlson, R., "An Integrated SoC for Science Data Processing in Next-generation Space Flight Instruments Avionics," *Very Large Scale Integration (VLSI-SoC), 2015 IFIP/IEEE International Conference on*, IEEE, 2015, pp. 134–141.

[58] Rudolph, D., Wilson, C., Stewart, J., Gauvin, P., George, A., Lam, H., Crum, G. A., Wirthlin, M., Wilson, A., and Stoddard, A., "CSP: A Multifaceted Hybrid Architecture for Space Computing," *SSC14-III-3, 28th Annual AIAA/USU Conference on Small Satellites*, 2014.

[59] Flatley, T. P., "SpaceCube: A Family of Reconfigurable Hybrid On-board Science Data Processors," *Keynote 2, 2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14)*,

2014.

[60] Petrick, D., Gill, N., Hassouneh, M., Stone, R., Winternitz, L., Thomas, L., Davis, M., Sparacino, P., and Flatley, T., "Adapting the SpaceCube v2.0 Data Processing System for Mission-unique Application Requirements," *Adaptive Hardware and Systems (AHS), 2015 NASA/ESA Conference on*, IEEE, 2015, pp. 1–8.

[61] Nah, J.-H., Suh, Y., and Lim, Y., "L-Bench: An Android Benchmark Set for Low-power Mobile GPUs," *Computers & Graphics*, Vol. 61, 2016, pp. 40–49.

[62] Online Performance Comparisons of GPU Devices Based on Benchmarks and Specifications: `http://www.videocardbenchmark.net/`, `http://gpuboss.com/`, `http://www.notebookcheck.net/`, `https://compubench.com/` (image processing benchmarks, i.e., Gaussian blurring, Histogram, Face detection), 2016.

[63] Gao, Y., Zhang, F., and Bakos, J. D., "Sparse Matrix-vector Multiply on the Keystone II Digital Signal Processor," *High Performance Extreme Computing Conference (HPEC), 2014 IEEE*, IEEE, 2014, pp. 1–6.

[64] Han, S., Mao, H., and Dally, W. J., "Deep Compression: Compressing Deep Neural Network With Pruning, Trained Quantization and Huffman Coding," *CoRR, abs/1510.00149*, Vol. 2, 2015.

[65] Kim, S. C. and Bhattacharyya, S. S., "An Efficient GPU Implementation of a Multirate Resampler for Multi-carrier Systems," *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2015, pp. 751–755.

[66] Mitra, G., Johnston, B., Rendell, A. P., McCreath, E., and Zhou, J., "Use of SIMD Vector Operations to Accelerate Application Code Performance on Low-powered ARM and Intel Platforms," *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2013 IEEE 27th International*, IEEE, 2013, pp. 1107–1116.

[67] Peng, B., Wang, T., Jin, X., and Wang, C., "An Accelerating Solution for N-Body MOND Simulation with FPGA-SoC," *International Journal of Reconfigurable Computing*, Vol. 2016, 2016.

[68] Cloutier, M. F., Paradis, C., and Weaver, V. M., "A Raspberry Pi Cluster Instrumented for Fine-Grained Power Measurement," *Electronics*, Vol. 5, No. 4, 2016, pp. 61.

[69] Sioutas, T., Corporaal, H., van den Braak, G.-J., and Larisis, N., *Benchmarking Novel Multi-SoC DSP Architecture*, Bachelor's thesis, Eindhoven University of Technology, Nov. 2015.

[70] Kumar, V., Sbîrlea, A., Jayaraj, A., Budimlić, Z., Majeti, D., and Sarkar, V., "Heterogeneous Work-stealing Across CPU and DSP Cores," *High Performance Extreme Computing Conference (HPEC), 2015 IEEE*, IEEE, 2015, pp. 1–6.

[71] Castillo, M. I., Fernández, J. C., Igual, F. D., Plaza, A., Quintana-Ortí, E. S., and Remón, A., "Hyperspectral Unmixing on Multicore DSPs: Trading off Performance for Energy," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, Vol. 7, No. 6, 2014, pp. 2297–2304.

[72] Menant, J., Pressigout, M., Morin, L., and Nezan, J.-F., "Optimized Fixed Point Implementation of a Local Stereo Matching Algorithm onto C66x DSP," *Conference on Design and Architectures for Signal and Image Processing (DASIP)*, 2014, pp. 1–6.

[73] Lentaris, G., Diamantopoulos, D., Siozios, K., Soudris, D., and Rodrigálvarez, M. A., "Hardware Implementation of Stereo Correspondence Algorithm for the Exomars Mission," *22nd International Conference on Field Programmable Logic and Applications (FPL)*, IEEE, 2012, pp. 667–670.

[74] Pelcat, M., Desnos, K., Heulot, J., Guy, C., Nezan, J.-F., and Aridhi, S., "Preesm: A Dataflow-Based Rapid Prototyping Framework for Simplifying Multicore DSP Programming," *European Embedded Design in Education and Research Conference (EDERC)*, 2014, pp. 36–40.

[75] Coombs, J. and Prabhu, R., "OpenCV on TI's DSP+ ARM Platforms: Mitigating the Challenges of Porting OpenCV to Embedded Platforms," *White Paper, Texas Instruments*, 2011.

[76] Ramesh, B., Bhardwaj, A., Richardson, J., George, A. D., and Lam, H., "Optimization and Evaluation of Image-and Signal-Processing Kernels on the TI C6678 Multi-core DSP," *High Performance Extreme Computing Conference (HPEC)*, 2014, pp. 1–6.

[77] Ginosar, R., Aviely, P., Gellis, H., Liran, T., Israeli, T., Nesher, R., Lange, F., Dobkin, R., Meirov, H., and Reznik, D., "RC64, a Rad-Hard Many-Core High-Performance DSP for Space Applications," *ESA Special Publication*, Vol. 732, 2015, p. 5.

[78] Tippetts, B., Lee, D. J., Lillywhite, K., and Archibald, J., "Review of Stereo Vision Algorithms and Their Suitability for Resource-limited Systems," *Journal of Real-Time Image Processing*, Vol. 11, No. 1, 2016, pp. 5–25.

[79] Chen, J., Alfred, C., and So, H. K.-H., "Design Considerations of Real-time Adaptive Beamformer for Medical Ultrasound Research Using FPGA and GPU," *International Conference on Field-Programmable Technology (FPT)*, 2012, pp. 198–205.

[80] Pauwels, K., Tomasi, M., Alonso, J. D., Ros, E., and Van Hulle, M. M., "A Comparison of FPGA and GPU for Real-time Phase-based Optical Flow, Stereo, and Local Image Features," *IEEE Transactions on Computers*, Vol. 61, No. 7, 2012, pp. 999–1012.

[81] Xu, Q., Varadarajan, S., Chakrabarti, C., and Karam, L. J., "A Distributed Canny Edge Detector: Algorithm and FPGA Implementation," *IEEE Transactions on Image Processing*, Vol. 23, No. 7, 2014, pp. 2944–2960.

[82] Tweddle, B. E., *Computer Vision Based Navigation for Spacecraft Proximity Operations*, Master's thesis, Dept. of Aeronautics and Astronautics, MIT, Cambridge, Mass, USA, 2010.

[83] Vellas, S., Lentaris, G., Maragos, K., Soudris, D., Kandylakis, Z., and Karantzalos, K., "FPGA acceleration of hyperspectral image processing for high-speed detection applications," *Circuits and Systems (ISCAS), 2017 IEEE International Symposium on*, IEEE, 2017, pp. 1–4.

[84] Rubner, Y., Puzicha, J., Tomasi, C., and Buhmann, J., "Empirical Evaluation of Dissimilarity Measures for Color and Texture," *Comput. Vis. Image Und.*, Vol. 84, No. 1, 2001, pp. 25–43.

[85] Horn, B., "Closed-Form Solution of Absolute Orientation Using Unit Quaternions," *J. Optical Soc. Am. A*, Vol. 4, No. 4, Apr. 1987, pp. 629–642.

[86] Intel, "Intel Quark SoC X1000 Core Hardware Reference Manual," 329678-001US, Oct. 2013.

[87] Crockett, L. H., Elliot, R. A., Enderwitz, M. A., and Stewart, R. W., *The Zynq Book: Embedded Processing with the Arm Cortex-A9 on the Xilinx Zynq-7000 All Programmable SoC*, Strathclyde Academic Media, 2014.

[88] Lentaris, G., Stamoulias, I., Diamantopoulos, D., Maragos, K., Siozios, K., Soudris, D., Rodrigálvarez, M. A., Lourakis, M., Zabulis, X., Kostavelis, I., Nalpantidis, L., Boukas, E., and Gasteratos, A., "SPARTAN/SEXTANT/COMPASS: Advancing Space Rover Vision via Reconfigurable Platforms," *International Symposium on Applied Reconfigurable Computing (ARC)*, 2015, pp. 475–486.

[89] Xillybus Ltd., "Xillybus FPGA Designer's Guide, Version 2.0," `http://xillybus.com/downloads/doc/xillybus_fpga_api.pdf`, 2016.

[90] Georgis, G., Lentaris, G., and Reisis, D., "Acceleration Techniques and Evaluation on Multi-core CPU, GPU and FPGA for Image Processing and Super-resolution," *Journal of Real-Time Image Processing*, 2016, pp. 1–28.

[91] Lucero, J. and Slous, B., "Designing High-Performance Video Systems with the Zynq-7000 All Programmable SoC Using IP Integrator," *Xilinx Application Note XAPP1205 (v1.0)*, March 28, 2014.

[92] Barry, B., Brick, C., Connor, F., Donohoe, D., Moloney, D., Richmond, R., O'Riordan, M., and Toma, V., "Always-on Vision Processing Unit for Mobile Applications," *IEEE Micro*, Vol. 35, No. 2, 2015, pp. 56–66.

[93] Siegle, F., Vladimirova, T., Ilstad, J., and Emam, O., "Mitigation of Radiation Effects in SRAM-Based FPGAs for Space Applications," *ACM Comput. Surv.*, Vol. 47, No. 2, Jan. 2015, pp. 37:1–37:34.