Orbital Edge Computing: Machine Inference in Space

Bradley Denby
bdenby@andrew.cmu.edu
Brandon Lucia
blucia@andrew.cmu.edu
Carnegie Mellon University

Abstract—Edge computing is an emerging paradigm aiding responsiveness, reliability, and scalability of terrestrial computing and sensing networks like cellular and IoT. However, edge computing is largely unexplored in high-datarate nanosatellite constellations. Cubesats are small, energy-limited sensors separated from the cloud by hundreds of kilometers of atmosphere and space. As they proliferate, centralized architectures impede advanced applications. In this work, we define and characterize Orbital Edge Computing. We describe power and software optimizations for the orbital edge, and we use formation flying to parallelize computation in space.

Index Terms—cubesat, edge computing, remote sensing, computer vision.

1 Introduction

Space system architects are eschewing large, costly (e.g. \$650,000,000 [1]), "exquisite" [2] monolithic satellites for constellations of small, cheap (e.g. \$65,000 ea.) "CubeSats" [3]. Commercial efforts [4], [5] use this 10,000× lower cost to deploy camera-equipped constellations to low Earth orbit (LEO), yielding high temporal resolution for Earth observation. Such constellations support precision agriculture, weather monitoring, and disaster relief. Existing efforts use a "bent pipe" architecture: ground stations issue commands for nanosatellites to downlink unprocessed data [6]. Bent pipes do not scale as data volume grows with constellation size and sensor quality.

Edge computing places processing hardware near data sources, unlike cloud architectures that centralize data analysis. Clouds accelerate computing when they are available [7], but they depend on a backhaul network. Trends toward ubiquitous, high-datarate sensors across large geographic areas — e.g. cameras throughout cities — dramatically increase data volume. Bandwidth from sensor to datacenter has not increased proportionally, limiting cloud-based analysis [8], [9].

Nanosatellite constellations with high-datarate cameras are restricted by centralized, terrestrial processing. Ground station location and orbit parameters limit link availability, impeding effective datarate scalability. Intermittent downlinks add latency between data collection and processing, requiring orbital data buffers. Downlinks are often unreliable. Some nanosatellites have a packet loss rate of 88% [10], and commercial ventures have complex downlink architectures [6]. Shared "last mile" infrastructures [11], [12] aid availability but do not address the terrestrial centralization bottleneck.

We explore *Orbital Edge Computing* (OEC) as an alternative to bent pipes. We propose colocating sophisticated processing hardware with sensors in small, low-cost satellites. Like work on warehouse scale computing [13] did for datacenters, we aim to raise awareness of system-level research questions for computational nanosatellites with high-datarate cameras.

We build intellectually on recent work. The $500\,\mathrm{kg}$ Earth Observing-1 (EO-1) satellite [14] has onboard cloud filtering and image novelty detection software [15]. The Intelligent Payload Experiment (IPEX) 1U cubesat performs support vector machine (SVM) image classification on $3\,\mathrm{Mpx}$ images using an Atmel AT91SAM9 $210\,\mathrm{MHz}$ ARM microcontroller [16].

We focus on challenges of large constellations with high-fidelity (e.g. 4K) cameras. We characterize the physically-constrained design space, considering volume, mass, energy storage, power, cost, and computing performance. We show quantitatively that a streaming downlink architecture is infeasible as constellation size grows. We establish the viability of OEC as an alternative to current architectures, supporting sophisticated image processing with convolutional neural networks (CNNs). We leverage formation flying [17], [18] to parallelize computing work across a *cubesat pipeline*. We show that existing optimizations such as image tiling, early discard [19], and hardware acceleration remain effective.

In summary, this work makes the following contributions.

- We study the design of OEC systems for visual inference.
- We describe a collection of OEC system optimizations.
- We propose OEC pipelining, flying devices in a line and parallelizing computing across the formation.
- We evaluate an OEC system for several inference tasks, showing viability and efficiency.

2 Nanosatellite System Constraints

Many nanosatellites use the "CubeSat" standard [3], [20], enabling use of low-cost, commercial off-the-shelf (COTS) components. Cubesats are composed of $10\,\mathrm{cm}\times 10\,\mathrm{cm}\times 10\,\mathrm{cm}$ ("1U") volumes. Each 1U volume is restricted to $1.33\,\mathrm{kg}$. Small solar panels provide tens of watts of power or less. Existing onboard computers are simple, low-performance parts used mainly for pointing sensors, buffering data, and managing radios.

2.1 Physically-Constrained OEC Design Space

Physical constraints limit OEC system design and data quality. Figure 1, bottom, shows how COTS cubesat components contribute to system volume, mass, power, and cost based on component datasheets. We assume a solar array covering the cubesat exterior, producing a peak of $7.1\,\mathrm{W}$. The data reveal guiding design constraints: OEC systems are volume and power limited but are neither mass nor cost limited.

Physical constraints limit sensor data. Ground sample distance (GSD) is the geographic distance between adjacent pixel-centers and a key figure of merit for visual data. Monolithic systems have GSDs around $0.3\,\mathrm{m/px}$ [21], and cubesats have GSDs around $3.0\,\mathrm{m/px}$ [22]. Three parameters govern GSD: orbit altitude, camera focal length, and pixel sensor size. GSD merit is proportional to focal length and inversely proportional to altitude and sensor size. Orbit altitude depends on the mission.

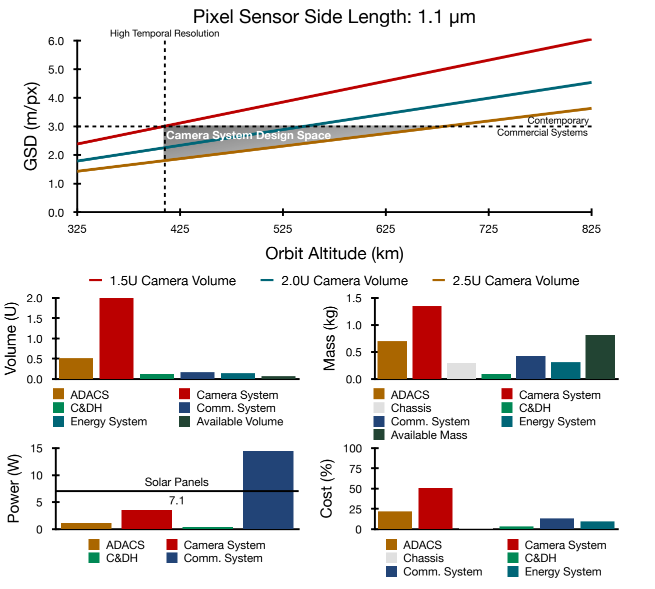

Fig. 1. Top: Image data quality is physically limited. Bottom: The OEC design space: volume, mass, power, and cost of LEO 3U OEC devices. OEC is limited in volume and energy, but not mass or cost.

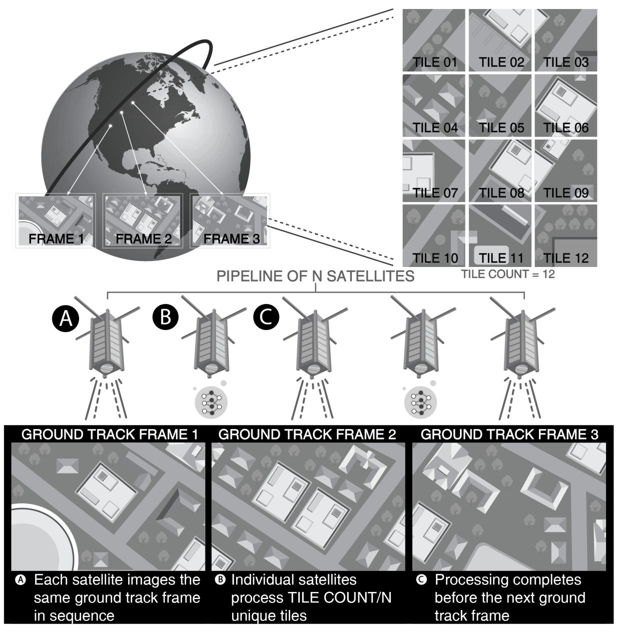

Fig. 2. The ground track frame rate, tiling, and cubesat pipelining.

A cubesat typically orbits in LEO between 325 km and 825 km. A higher altitude increases mission duration but degrades GSD. Below 400 km, a cubesat reenters the atmosphere in weeks; one above 400 km can operate for years. Camera focal length is limited by cubesat volume consumed by other systems, such as the attitude determination and control system (ADACS) and batteries. Sensor technology is mature; we assume a COTS pixel sensor side length of 1.1 μm with at least 4096×3072 pixels [23].

Figure 1, top, shows the OEC design space assuming a 3U volume (like commercial systems [6]), and computes GSDs using a pinhole camera model [24]. Each curve is a different camera focal length. The data show that a viable 2U camera system has a GSD of 2.26 m/px: 7.5× worse than monolithic satellite data, but comparable to existing cubesat constellations.

System Architecture. We propose a simple, yet versatile OEC architecture. Based on volume and GSD constraints, we allocate 2U to camera equipment for a GSD ≤ 3.0 m/px. The remaining 1U contains the ADACS, radio, and computing hardware. We propose using a fast commodity mobile GPU (e.g. GP10B Tegra GPU) optimized for deep neural network (DNN) inference, a key OEC workload. A commodity X-Band radio and a mobile GPU each consume over 7.5 W: slightly too high to run continuously and far too high to run simultaneously with 7.1 W from a solar array. We propose a power system without a battery to avoid related volume, mass, and management hardware. We opt for small, dense supercapacitors. Supercapacitors buffer energy, permitting high-power bursts of operation once charged. An OEC system computer sleeps while solar panels charge supercapacitors, and it runs in bursts until energy is depleted. Charge time and burst time are dictated by panel power and supercapacitor size, similar to intermittent computing [25].

2.2 Downlinking Does Not Scale

Downlinks limit OEC constellation sensor datarate. Each cubesat must store data until near a ground station. Existing systems downlink all images, incurring a 5.5 h delay before data reach customers [6]. Large constellations amplify scalability challenges [4] because devices share a link and downlinking all data is infeasible as constellations grow. Earth-observing satellites use polar orbits to maximize geographic coverage and sun-synchronous orbits for constant pass times [26]. A satellite at 410 km maximizes spatial resolution with a GSD under 3.0 m/px, has a temporal recurrence cycle of 2 d, and has an orbit period of 92.9 min. Fixed ground stations have downlink windows of 10 min per device [6]. Existing systems downlink at 200 Mbit/s, receiving up to 15 GB of data per pass. A ground station optimally positioned to observe every pass (e.g. a polar station for a polar orbit) supports up to 9 satellites per revolution. Similarly supporting a 1000-satellite constellation requires 112 ground stations. Provisioning many ground stations is wasteful if downlinked data are unused.

OEC reduces ground station count by computing on orbit and discarding uninteresting data early. If early discard reduces 15 GB of raw data to 0.75 GB of useful data, all data downlink in 30 s at 200 Mbit/s. Each station services 185 satellites per 92.9 min revolution. Equipped to compute, a constellation of 1000 OEC satellites is adequately supported by 6 ground stations — increasing downlink capacity by 20×.

3 Optimizing Orbital Edge Computing

We develop support for processing high-rate data in OEC constellations. Tiled image processing can be tuned empirically to maximize accuracy. Intelligent early discard avoids processing uninteresting sensor data. OEC pipelining parallelizes work across a constellation, mitigating latency.

3.1 Geographic Tiling to Maximize Accuracy

OEC systems split camera frames into tiles of fixed geographic area and scale each tile to the input size of an inference kernel, like terrestrial satellite image processing [27]. Tile size determines feature (e.g. building) size after scaling; smaller tiles have more pixels per feature after resizing, and larger tiles have fewer. Tile size determines tile count per frame, which varies latency. Section 4 shows that tiling dictates accuracy and latency, and some cases have an empirically optimal tile size.

3.2 Intelligent Early Discard

An OEC system avoids redundant tiles by capturing images at no greater rate than the *ground track frame rate* (GTFR). The GTFR is the rate at which entirely new geographic scenes appear in the camera view, and the *ground track frame period* (GTFP) is its inverse. Figure 2 illustrates tiling and the GTFR.

An OEC system processes images to filter uninteresting data, akin to recent drone-based systems [7], [19]. Earth observation missions may have very specific targets (e.g. building footprint detection [27]) or broad goals (e.g. Doves imaging Earth's landmass daily [4]). In most cases, some images are not useful. For example, Doves avoid imaging large water bodies based on location. However, Doves downlink images above land regardless of content. Processing data before downlink as in OEC allows content-based filtering. With OEC, images obscured by clouds can be discarded as in [15]. An OEC disaster relief mission can find and downlink building coordinates instead of raw images (Section 4.3 evaluates this goal).

3.3 Computational Nanosatellite Pipelines

An OEC system can leverage ample existing work on reliable formation flying [17], [18] to keep nanosatellites in a line and parallelize image processing across a constellation. Parallelism reduces effective frame latency. Each satellite captures an identical frame at the GTFR, processing a *subset* of frame tiles statically determined by satellite position in the linear formation. A frame's tiles should be distributed across nanosatellites in proportion to their computing capability; identical satellites process equal tile shares. A key advantage of OEC pipelining is that satellite GPS location determines when to capture a frame and static pipeline position determines the tile subset, requiring no satellite-to-satellite cross-linking to process data in parallel. *Subframe latency*, the latency for one satellite to process its assigned tiles, is given by single tile processing latency, times the number of tiles per frame, divided by pipeline depth. The pipeline completely processes an image before the next frame when subframe latency is less than the GTFP. An OEC pipeline should have enough satellites to meet this requirement. Figure 2 shows an OEC pipeline in which each device processes only a subset of the frame tiles.

A computational nanosatellite pipeline requires propulsion and positioning. Recent surveys of deployed and proposed nanosatellite propulsion systems make formation flying feasible [17], [28]. Contemporary, low-power COTS navigation constellation receivers provide positioning for image capture triggers once unlocked for high velocity, high altitude use [29].

4 Evaluation

We evaluate an OEC system designed for image classification [30], object detection [31], and pixel segmentation [32], running on the Jetson TX2 mobile GPU module, which prior work shows is robust to space radiation [33]. We use satellite data from SpaceNet [27] to train the networks. We use SpaceNet evaluation metrics and ground truth for evaluation on separate SpaceNet test data.

4.1 OEC Maintains Accuracy

Despite physical limitations on achievable nanosatellite GSD, inference accuracy remains comparable to that on higher quality monolithic satellite data. Figure 3, left, shows inference precision, recall, and F1 score [34] as GSD varies. In each case, F1 score remains steady as GSD degrades from $0.3\,\mathrm{m/px}$ to $6.0\,\mathrm{m/px}$, with only a slight downward trend. Any modest decrease in F1 score due to lower GSD is likely acceptable given the 10,000$\times$ decrease in OEC device cost and higher temporal resolution compared to monolithic systems.

Figure 3, right, shows how to optimize an OEC system tiling scheme for F1 score and latency. Tile size determines latency. Across GSD values, there is a "goldilocks" optimal tile size maximizing F1 score for detection and segmentation workloads. Smaller tiles increase latency without increasing F1 score, and larger tiles decrease latency at the expense of F1 score. Given computing performance parameters, orbit parameters, and representative data, a mission engineer can optimize tile size empirically before launch.

4.2 OEC Pipelines Mitigate Latency

OEC pipelines parallelize processing and effectively amortize frame latency. Figure 4, top, plots frame processing latency versus pipeline depth. We evaluate a pipeline in a $400\,\mathrm{km}$ altitude polar orbit with a $7.23\,\mathrm{km/s}$ ground track velocity [26], a GTFP of $1.70\,\mathrm{s}$, and a GTFR of 0.59 fps. We model full system behavior, including our supercapacitor-based power system (using capacitor charging equations), and require compute components to sleep until capacitors charge. We use directly measured Jetson module power values to study systems with one or three Jetsons (1J/3J), $7.1\,\mathrm{W}$ or $21.3\,\mathrm{W}$ solar arrays (1P/3P), and 5 Farad or 15 Farad energy storage capacitors (5F/15F).

The data show that building detection is feasible at the GTFR, but that segmentation latency is infeasible at the GTFR without increasing pipeline depth beyond existing systems. With a single GPU, low energy storage, and a single solar panel, continuous object detection at the GTFR is feasible with a 245 satellite pipeline. With low energy storage and a single panel, the workload consumes energy quickly, requiring more sleep periods to recharge and increasing frame latency. With larger, folding, deployable panels (3P), detection at the GTFR is feasible with 78 OEC devices — even allowing continuous operation while not in eclipse. Classification is feasible at the GTFR in all configurations, with as few as 19 devices.

4.3 Performance Optimizations at the Orbital Edge

Intelligent early discard decreases network traffic and total energy. We run classification to find images with buildings as

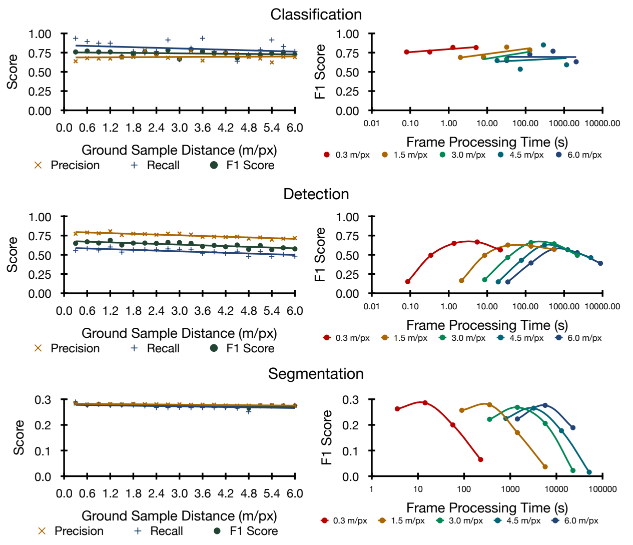

Fig. 3. Left: Accuracy is stable as GSD degrades. Right: Tile area dictates tile count, determining latency.

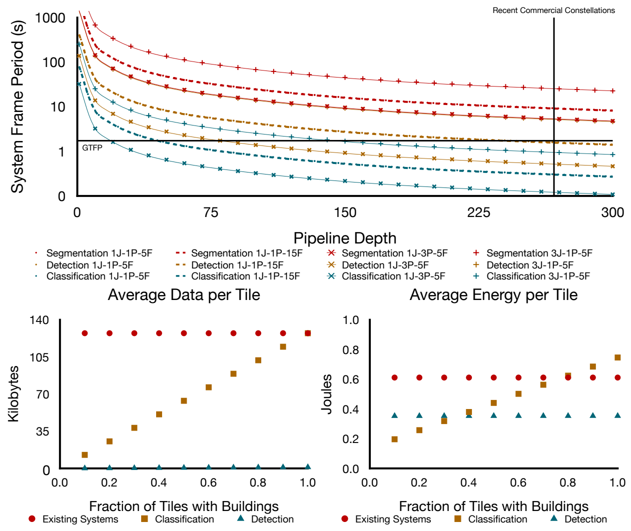

Fig. 4. Top: Pipeline depth determines frame processing latency. For each workload (segmentation, detection, classification), the number of Jetsons (1J/3J), input power (1P/3P), and energy storage capacity (5F/15F) vary. Configuration 1P corresponds to $7.1\,\mathrm{W}$, and configuration 3P corresponds to $21.3\,\mathrm{W}$. Energy storage capacities are in farads. Bottom: The amount of data transmitted depends on the filtering technique. Different filtering techniques lead to different energy tradeoffs.

a filter for building detection, and compare to sending all data, sending images with buildings, and sending building coordinates only. Figure 4, bottom, plots the network traffic and total energy of an existing and OEC system. With a varied fraction of interesting tiles (i.e. with buildings), sending coordinates only greatly reduces network traffic. With classification, network traffic depends on the fraction of interesting tiles. Energy also depends on the fraction of interesting tiles: below 30%, filtering avoids expensive detection. Above 30%, classification filtering increases total energy cost when detection is also performed.

Architectural accelerators promise to optimize inference computations. Table 1 collects energy and latency data from prior work to estimate the performance impact of acceleration in an OEC system. Acceleration greatly decreases energy cost, avoiding sleep states and decreasing pipeline depth.

TABLE 1
Accelerators enable shorter pipelines.

Accelerator	Energy (mJ)	Latency (ms)	Pipeline Depth
Jetson TX2 [35]	643.3	42.89	78
ShiDianNao [36]	0.457	1.430	3
Origami [37]	0.722	1.415	3
EIE [38]	1.980	3.299	6
Eyeriss [39]	0.928	3.338	7

5 Conclusion

In this work, we develop orbital edge computing. The low cost of nanosatellites compared to monolithic satellites makes large constellations feasible for the first time. Applications of this emerging technology are impeded by traditional, centralized architectures. Orbital edge computing provides responsiveness, reliability, and scalability benefits. Future work should study energy collection and storage for orbital edge computing and radiation hardened machine learning accelerators. We anticipate that cubesat pipelining may motivate further research in nanosatellite control and orbital cross link communication.

Acknowledgments

We thank the reviewers for the helpful feedback. This work was generously funded by the Kavcic-Moura Endowment Fund and National Science Foundation CAREER Award #1751029.

References

[1] W. Ferster, "Digitalglobe adding infrared capability to worldview-3 satellite," *Space News, https://spacenews.com/digitalglobe-adding-infrared-capability-worldview-3-satellite/*, 2012.
[2] Tactical Technology Office, "Broad agency announcement: Blackjack pit boss, hr001119s0012," DARPA, Tech. Rep., 2018.
[3] A. Mehrparvar, D. Pignatelli, J. Carnahan, R. Munakat, W. Lan, A. Toorian, A. Hutputanasin, and S. Lee, "Cubesat design specification rev. 13," California Polytechnic State University, San Luis Obispo, Tech. Rep., 2014.
[4] L. Leung, V. Beukelaers, S. Chesi, H. Yoon, D. Walker, and J. Egbert, "Adcs at scale: Calibrating and monitoring the dove constellation," *AIAA/USU Conference on Small Satellites*, 2018.
[5] J. Cappaert, "Building, deploying and operating a cubesat constellation-exploring the less obvious reasons space is hard," *AIAA/USU Conference on Small Satellites*, 2018.
[6] K. Devaraj, R. Kingsbury, M. Ligon, J. Breu, V. Vittaldev, B. Klofas, P. Yeon, and K. Colton, "Dove high speed downlink system," *AIAA/USU Conference on Small Satellites*, 2017.
[7] B. Boroujerdian, H. Genc, S. Krishnan, W. Cui, A. Faust, and V. Reddi, "Mavbench: Micro aerial vehicle benchmarking," in *MICRO*. IEEE, 2018.
[8] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE IoT*, 2016.
[9] K. Hsieh, G. Ananthanarayanan, P. Bodik, S. Venkataraman, P. Bahl, M. Philipose, P. B. Gibbons, and O. Mutlu, "Focus: Querying large video datasets with low latency and low cost," in *USENIX OSDI*, 2018.
[10] C. Nogales, B. Grim, M. Kamstra, B. Campbell, A. Ewing, R. Hance, J. Griffin, and S. Parke, "Makersat-0: 3d-printed polymer degradation first data from orbit," *AIAA/USU Conference on Small Satellites*, 2018.
[11] D. White, C. Shields, P. Papadeas, A. Zisimatos, M. Surligas, M. Papamatthaiou, D. Papadeas, E. Kosmas, V. Tsiligiannis, A. Csete *et al.*, "Overview of the satellite networked open ground stations (satnogs) project," *AIAA/USU Conference on Small Satellites*, 2018.
[12] Amazon Web Services, Inc., "Amazon Ground Station," https://aws.amazon.com/ground-station/, 2018.
[13] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, 2007.
[14] S. Chien, R. Sherwood, D. Tran, B. Cichy, G. Rabideau, R. Castano, A. Davies, R. Lee, D. Mandl, S. Frye *et al.*, "The eo-1 autonomous science agent," in *Autonomous Agents and Multiagent Systems*. IEEE Computer Society, 2004.
[15] K. L. Wagstaff, A. Altinok, S. A. Chien, U. Rebbapragada, S. R. Schaffer, D. R. Thompson, and D. Q. Tran, "Cloud filtering and novelty detection using onboard machine learning for the eo-1 spacecraft," in *IJCAI Workshop on AI in the Oceans and Space*, 2017.
[16] D. R. Thompson, A. Altinok, B. Bornstein, S. A. Chien, J. Doubleday, J. Bellardo, and K. L. Wagstaff, "Onboard machine learning classification of images by a cubesat in earth orbit," *AI Matters*, 2015.
[17] S. Bandyopadhyay, R. Foust, G. P. Subramanian, S.-J. Chung, and F. Y. Hadaegh, "Review of formation flying and constellation missions using nanosatellites," *Journal of Spacecraft and Rockets*, 2016.
[18] K. Luu, M. Martin, A. Das, A. Peffer, H. Schlossberg, J. Mitola, D. Weidow, R. Blomquist, M. Campbell, and C. Hall, "Microsatellite and formation flying technologies on university nanosatellites," in *Space Technology Conference and Exposition*, 1999.
[19] J. Wang, Z. Feng, Z. Chen, S. George, M. Bala, P. Pillai, S.-W. Yang, and M. Satyanarayanan, "Bandwidth-efficient live video analytics for drones via edge computing," in *SEC*. IEEE, 2018.
[20] J. Puig-Suari, C. Turner, and W. Ahlgren, "Development of the standard cubesat deployer and a cubesat class picosatellite," in *Aerospace Conference, IEEE Proceedings*, 2001.
[21] DigitalGlobe, "Worldview-3 data sheet," DigitalGlobe, Tech. Rep., 2017.
[22] Planet Labs, "Planet imagery product specifications," Planet Labs, Tech. Rep., 2018.
[23] ON Semiconductor, "Ar1335 cmos image sensor datasheet," ON Semiconductor, Tech. Rep., 2018.
[24] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.
[25] A. Colin, E. Ruppel, and B. Lucia, "A reconfigurable energy storage architecture for energy-harvesting devices," in *ASPLOS*. ACM, 2018.
[26] M. Capderou, *Satellites: Orbits and missions*. Springer, 2006.
[27] A. Van Etten, D. Lindenbaum, and T. M. Bacastow, "Spacenet: A remote sensing dataset and challenge series," *arXiv preprint arXiv:1807.01232*, 2018.
[28] A. Tummala and A. Dutta, "An overview of cube-satellite propulsion technologies and trends," *Aerospace*, 2017.
[29] B. Yost, "State of the art of small spacecraft technology," https://sst-soa.arc.nasa.gov/, 2016.

[30] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Computer vision and pattern recognition*. IEEE, 2015.

[31] A. Tao, J. Barker, and S. Sarathy, "Detectnet: Deep neural network for object detection in digits," *Parallel Forall*, vol. 4, 2016.

[32] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[33] H. Wang, Q. Chen, L. Chen, D. M. Hiemstra, and V. Kirischian, "Single event upset characterization of the tegra k1 mobile processor using proton irradiation," in *Radiation Effects Data Workshop*. IEEE, 2017.

[34] G. James *et al.*, *An introduction to statistical learning*. Springer, 2013.

[35] NVIDIA, "Nvidia jetson tx2/tx2i system-on-module data sheet," NVIDIA, Tech. Rep., 2018.

[36] Z. Du, R. Fasthuber, T. Chen, P. Ienne, L. Li, T. Luo, X. Feng, Y. Chen, and O. Temam, "Shidiannao: Shifting vision processing closer to the sensor," in *ACM SIGARCH*, 2015.

[37] L. Cavigelli and L. Benini, "Origami: A 803-gop/s/w convolutional network accelerator," *IEEE TCSVT*, 2017.

[38] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "Eie: Efficient inference engine on compressed deep neural network," in *ISCA*. IEEE, 2016.

[39] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *Solid-State Circuits*, 2017.