Jet Propulsion Laboratory
California Institute of Technology

Benchmarking Space Mission Applications On The Snapdragon Processor On-board The ISS

Jason Swope[1], Faiz Mirza[1], Emily Dunkel[1], Zaid Towfic[1], Steve Chien[1], Damon Russell[1], Joe Sauvageau[1], Doug Sheldon[1], Mark Fernandez[2], Carrie Knox[2]

[1]Jet Propulsion Laboratory, California Institute of Technology, USA

[2]Hewlett Packard Enterprise

June 29, 2023

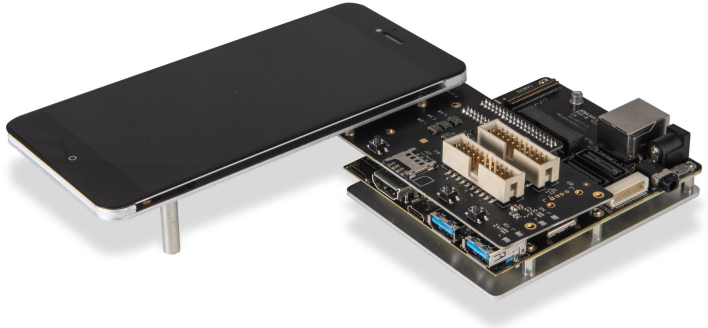
IEEE LEO SatS 2023 Workshop

This research was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Jet Propulsion Laboratory
California Institute of Technology

Background

Qualcomm Snapdragon 855 HDK



All Snapdragon images courtesy Qualcomm

- 8 core ARM system
  - 4 “silver” high efficiency cores ~ 1.80 GHz
  - 3 “gold” high performance cores ~ 2.42 GHz
  - 1 “gold prime” very high performance core ~ 2.84 GHz
- Adreno 640 Graphics Processing Unit (GPU)
- Qualcomm Hexagon 690 Digital Signal Processor (DSP)
- Neural Processing Engine
  - Directly supports Convolutional Neural Networks (CNNs) in hardware
- Running Android OS

Also Snapdragon Automotive Development Processor Units running linux (for automotive).

Rad 750

Current computing for MSL, M2020

PowerPC Heritage

Image re: MSL from CNET

JPL’s Sabertooth

- LEON 4 Based CPU
- Target 8-10x improvement in evolution from Sphinx

Image credit citation below.

W. Whittaker
Sabertooth: Integrated Avionics for Small Spacecraft Missions
2019 Space Computing Conference
https://trs.jpl.nasa.gov/bitstream/handle/2014/51550/CL%2319-4553.pdf?sequence=1&isAllowed=y

Embedded Processors

Processor	Snapdragon 855	Rad750	Sabertooth
Power	5W	10+ W 5 W?	3W https://trs.jpl.nasa.gov/bitstream/handle/2014/51550/CL%2319-4553.pdf?sequence=1&isAllowed=y
MIPS	https://www.notebookcheck.net/Qualcomm-Snapdragon-855-SoC-Benchmarks-and-Specs.375436.0.html	typical 266 up to 400	1200
Cores, Clocks	8 @ 1.7-2.8 GHz	1@110-200 MHz	4@ ?
RAM, NVM	16 GB	256 MB 2GB	192 MB 8 GB
Coprocessors	GPU, DSP*, AIP*		Motor controllers

*quantized models: 8 bit fixed point

**half-precision floating point

jpl.nasa.gov

International Space Station Experiment
Hewlett Packard Enterprise Spaceborne Computing-2

Delivered to ISS turnover: Fall 2020

Delivered to the ISS onboard Cygnus NG-15: February 20, 2021.

Powered on: May 12th 2021.

Hewlett Packard Enterprise Spaceborne Computing-2 package:

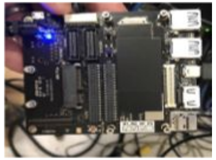
- COTS Linux workstations from HPE
- Intel Xeon 5215 Processor (10 cores)
- 4 NVIDIA Tesla GPUs
- 2 Machines aboard the ISS

2x Snapdragon 855 HDKs running Android

- Radios disabled

2x Myriad X Processors

+



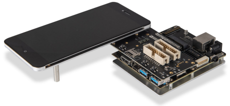
+

Uplinks possible periodically to load new SW

All SBC-2 images credit HPE.

Experiment Setup

1. JPL
Develops and Tests



HPE:
Chippewa Falls, WI

HPE
Ground
Testbed

2. JPL Tests

HPE Flight
Testbed
(Ground)

3. HPE Tests

4. HPE Tests

SBC-02 on ISS

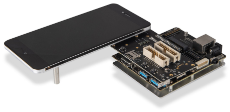
5. Results → JPL

Test Harness Setup

Test Harness Computer

or

Code, Data



Results

This setup is not completely unlike an “instrument coprocessor” setup

Test Harness runs on laptop or SBC-02 linux host
Test Harness:

- iterates through experiments:
  - delivers experiment code and data to embedded processor,
  - runs experiments on embedded processor
  - cleans up after execution, retrieving test results,
  - reboots if needed (timeouts)

Jet Propulsion Laboratory
California Institute of Technology

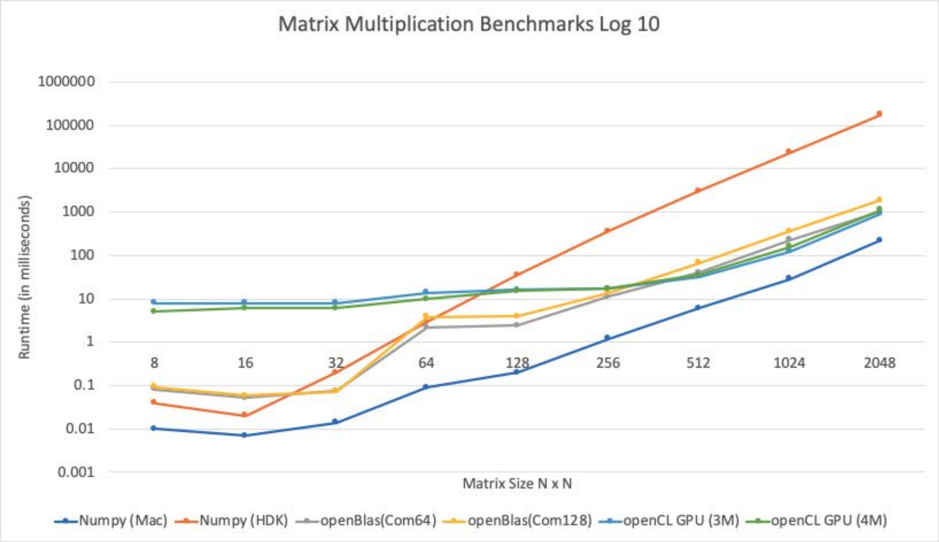
General Algorithms

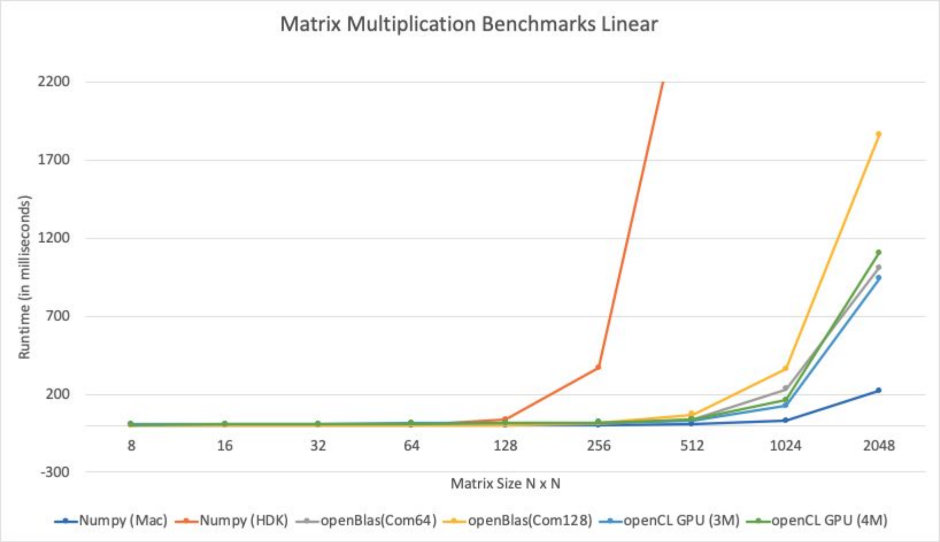
Fast Fourier Transform (FFT)

2X 5X ≈ 1.75X 2.75X

Duration Power Current	**Float64 ARM**	**Float32 ARM**	**Float32 GPU**	**UINT8 ARM (FASTCV)**	**UINT8 DSP (QHL)**	**UINT8 DSP (QCOM)**
1D FFT (1x2048)	**0.032ms** 2.5W 600mA	**0.025 ms** 2.5W 600mA	**0.025 ms** 2.1 W 525mA	**0.025 ms** 2.5W 600mA	**0.774 ms (1 thread)** 1.50W 420mA	
2D FFT (2048x2048)	**928 ms** 2.4W 550mA	**492 ms** 2.5W 600mA	**105 ms** 2.3 W 575mA	106 ms 2.5W 600mA	125.07 ms (1 thread) 2.00W 500mA 79.58 ms (2 thread) 2.47W 600mA **60.69 ms (4 thread)** 2.76W 670mA	
2D FFT (1024x1024)	**78.11 ms** 3.06W 770mA	**63.18 ms** 3.00W 766mA	**23.436 ms** 2.35W 590mA 20% GPU Utilization	**23.56 ms** 3.00W 766mA	26.33 ms (1 thread) 2.0W 500mA 17.00 ms (2 thread) 2.32W 600mA **14.14 ms (4 thread)** 2.60W 640mA	**5.128 ms (4 thread)** 2.40W 612mA

Matrix Multiplication





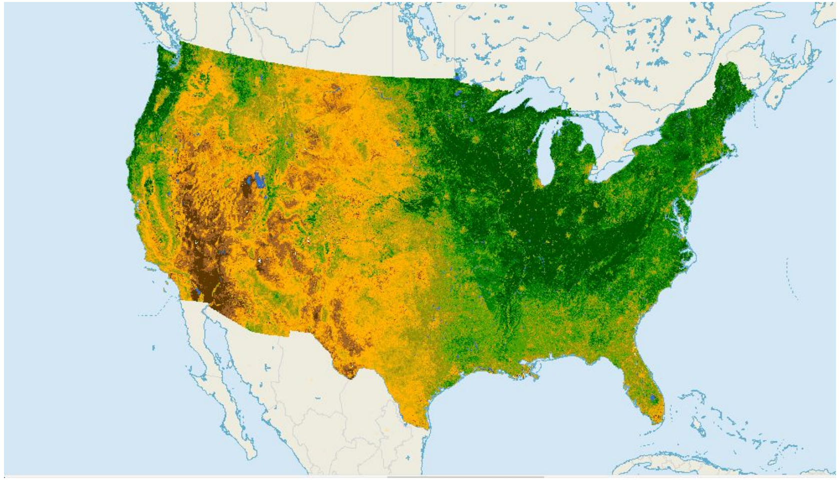
Generic benchmarks: more extensive parameter changes relating to different libraries and input data sizes.

Jet Propulsion Laboratory
California Institute of Technology

Instrument Processing Applications

Normalized Difference Index (NDI)

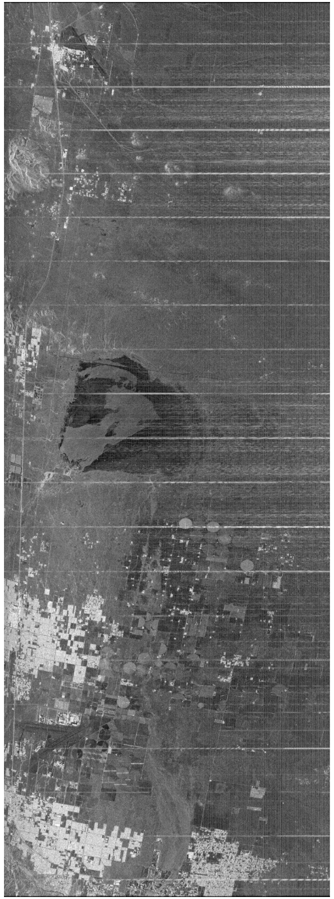
- Many target science products are normalized difference indices (e.g. normalized difference vegetation index, normalized difference snow)
- Future proposals would are expected to downlink higher resolution multispectral data (~30m)
- This extension necessitates selectivity to intelligently reduce data volumes to enable the use of economical ground stations
- Used to perform mineral detection on 8 minerals on AVIRIS-NG data from Cuprite Hills, Nevada
- Implemented on the Snapdragon ARM CPU
- Dataset: 20,000 pixels with 425 bands each
- Runtime: 56s $\rightarrow$ 0.0028s per pixel



Normalized Difference Vegetation Index over the US

Synthetic Aperture Radar (SAR) Image Formation

- Image Processing pipeline from Uninhabited Aerial Vehicle SAR (UAVSAR) instrument
- A pipeline of 3 ARM applications, 2 GPU applications
  - Mainly a row-wise and column-wise 2D FFT with filters applied
- Goal of <240 Seconds (~ real time)
- Image Size: 27916x26880
- Currently takes 217 seconds
  - Could possibly be further improved, as GPU usage is only at about 60%

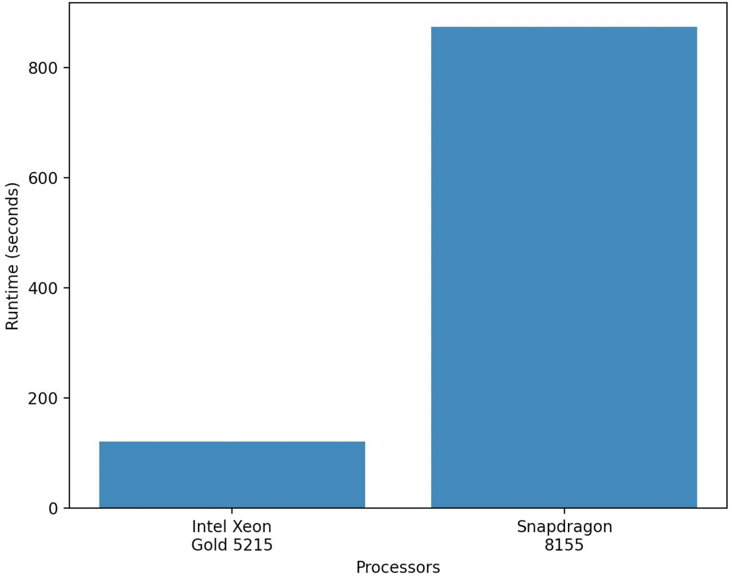


The Rosamond Calibration Array (RCRA), located near the south beach of Rosamond Dry Lake Bed in California.

Match Filters

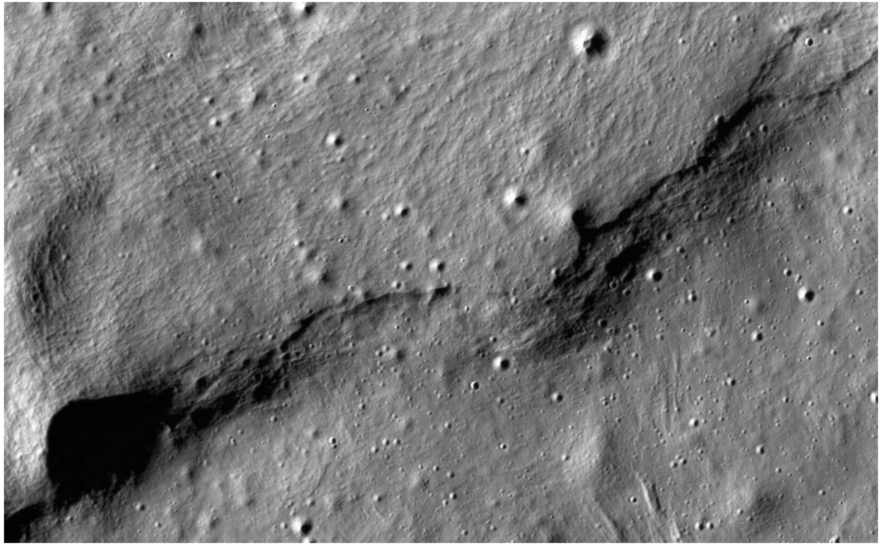
- Running on images from the Airborne Visible / Infrared Imaging Spectrometer (AVIRIS)
  - Images of Cuprite site in 2014
- Runs Kaolinite, Calcite and Alunite detection
- Runs on ARM only, single threaded
- Much of the runtime is I/O, so multithreading has modest effect
- Image Size: 670x2512x425
- Runtime through 8 images on 1 mineral on the graph
  - ~850s on Snapdragon ARM

https://www.jpl.nasa.gov/missions/airborne-visible-infrared-imaging-spectrometer-aviris



Lunar Match Filters

- Running on a dataset of modified M3 imaging spectrometer images of the Karpinsky feature of the moon
- Aims to demonstrate the detection of volatile water molecules on the lunar surface
  - Currently running on different forms of water (OH, molecular H_2O, and H_2O ice)
- Runs on ARM only, single threaded
- Performance mirrors Match Filters
- Image size 304x1000x301
- Runtime 108.4s



https://photojournal.jpl.nasa.gov/catalog/PIA23236

Hyperspectral Compression

- Benchmarked on test images for Earth Surface Mineral Dust Source Investigation (EMIT)
- 64 lines, 640 samples per line, 481 spectral bands
- MSamples/sec = lines * samples per line * bands / runtime
- EMIT Target is 23.1 MSamples/sec (near real time)

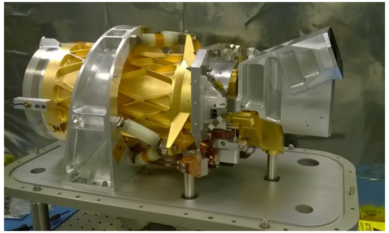
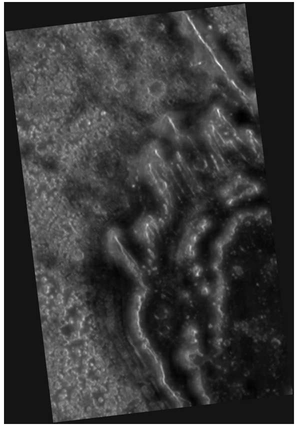


Image from https://earth.jpl.nasa.gov/emit/instrument/overview/

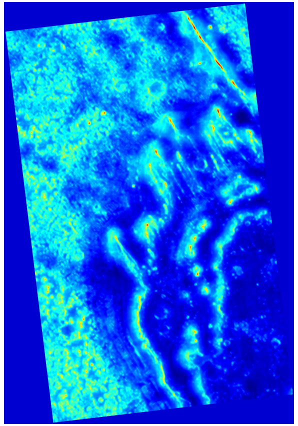
	CPU	GPU	DSP	Virtex-5	GTX 580
CCSDS Standard	123.0.B-2	123.0.B-1	123.0.B-2	123.0.B-1	123.0.B-1
Compression	Lossless				
Runtime (ns)	14.12	6.5	184.5	25	16.18
Sample Rate (MSamples/sec)	70.82	153.85	5.42	40.00	61.80
Power (W)	6.1	3.5	1.9	2	>100

Salience Detector: Computer Vision for Landmark Detection

- What: Landmark / regions of interest detection in imagery.
- Applications: autonomous data collection, targeted downloads, commanding space assets
- Details:
  - Generates salience maps of large image swaths to enable reduction in data volume
  - Uses computer vision techniques [Mars1]
- Ported to Snapdragon CPU
  - Processes single grayscale image 2048x4032 pixels in 23 seconds
    - Compared w/ 12 seconds on my mac



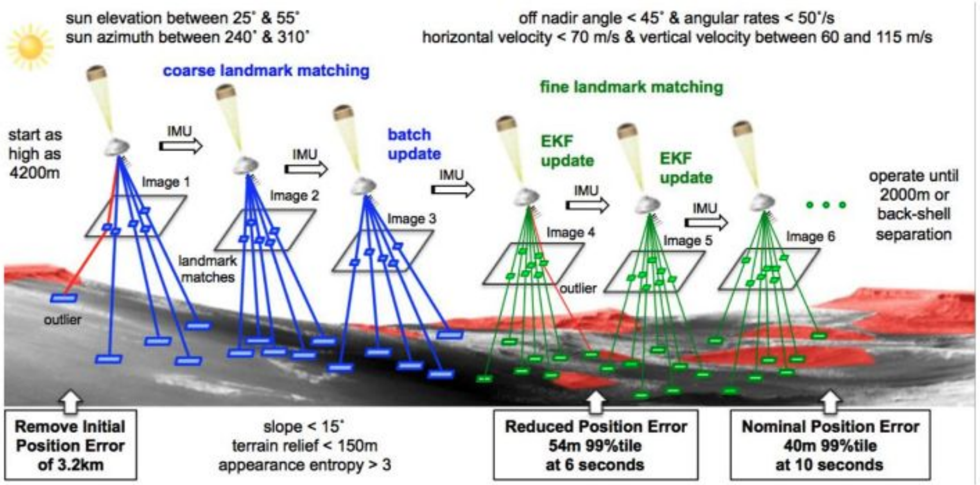
Mars HiRISE Image



Salience map

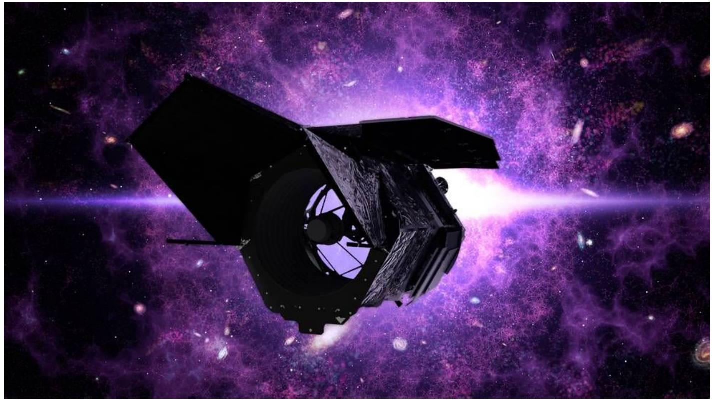
Landing Vision System/Astrotipping

- Landing Vision System currently implemented as a hybrid FPGA + Processor solution
- Problem divided into COARSE and FINE modes
- COARSE
  - Run on 1024x1024 image
  - Image Warping
  - Match to template (FFT)
  - Runtime: 2.46s
- FINE
  - Run on 1024x1024 image
  - Normalized Cross Correlation
  - Runtime: 2s
- Implemented on the Snapdragon ARM



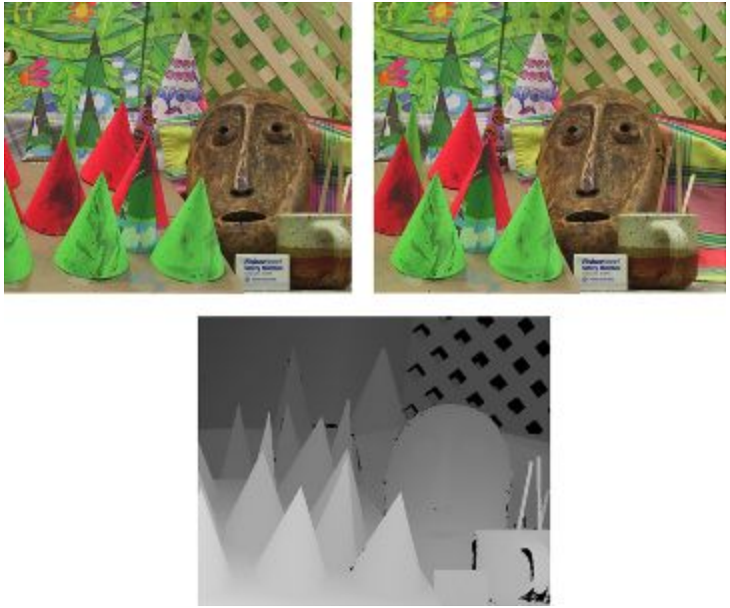
High Order WaveFront Sensing (HOWFS)

- Python implementation of HOWFS for Roman Space Telescope Coronagraph Instrument
  - Onboard processing would facilitate the mission
  - Faster processor is necessary for onboard processing
- Single threaded non-optimized port - Currently a little slow
  - Original double precision code took 2.2 hours
  - Moving to single precision took 1.8 hours
- Further Work
  - GPU
  - Multi-thread
  - Move from Python (allow easier use of specialized hardware)



Europa Lander Stereo Vision

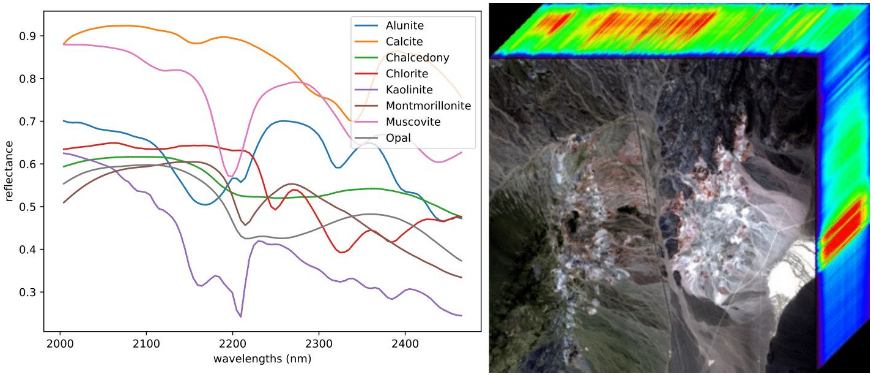
- Application determines the relative range of objects in a particular image is using a set of images from a stereo vision camera
- Success scored on the measurement of the sum of absolute differences (SAD) between five patches in the two images
- Implemented on the Snapdragon ARM
- Dataset: 24 image pairs
  - Each image is natively at 5120x3840
- Images processed at three different resolutions:
  - Level 0: 5120x3840: 15.6 minutes
  - Level 1: 2560x1920: 2.1 minutes
  - Level 2: 1280x960: 19 seconds



Stereo Vision Depth Map Example

Spectral Algorithms

- Benchmark the match filter (MF) and spectral angle mapper (SAM) algorithms for spectral analysis
  - Both algorithms use a spectral library containing objects of interest to be searched
  - SAM is a distance function between a spectrum and an object of interest
  - MF is a linear detector that requires background statistics: the mean covariance matrix
- Used to perform mineral detection on 8 minerals on AVIRIS-NG data from Cuprite Hills, Nevada
- Implemented on the Snapdragon ARM CPU
- Dataset: 20,000 pixels with 425 bands each



Spectral library of 8 minerals (left)
AVIRIS-NG hyperspectral image of Curpite Hills, Nevada (right)

Timing in ms per pixel	SBC2	Snapdragon	Sabertooth
SAM	0.04	0.0425	0.16
MF	0.04	0.06	0.15

SAM:
3.8x speedup from the Sabertooth to the Snapdragon

MF:
2.5x speedup from the Sabertooth to the Snapdragon

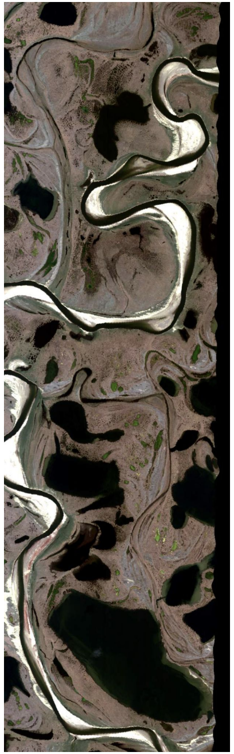
Decision Trees - Thermal Anomaly Detection

- Decision tree logic to detect high thermal emissions (lava, wildfires) using radiance band ratios
  - (Human expert constructed trees Davies et al. 2006 RSE)
- Classifies into hot and extremely hot pixels utilizing multiple radiance bands
- Run over Aviris-NG data
  - Image size 4500x390x425
- Implemented on the ARM and GPU of the Snapdragon

Skysat image of the Fagradalsfjall volcano in Iceland

Decision Trees Cryosphere Detection

- Decision tree based on cryosphere detection
  (human expert constructed tree Doggett et al. 2006 RSE)
- Classifies clouds, water, snow, ice, and land within an image
- Run using AVIRIS-NG data over Alaska
  - Image size 4500x390x425
- Implemented on the Snapdragon ARM CPU and Snapdragon Adreno GPU
  - CPU runtime: 21s (cryosphere and lava)
  - GPU runtime: 13s (cryosphere and lava)



AVIRIS-NG image of Alaska

	SBC2	Snapdragon CPU	Snapdragon GPU	Sabertooth
ms per pixel	0.00068	0.012	0.0074	.31

25.8x speedup from the Sabertooth to the Snapdragon CPU

41.9x speedup from the Sabertooth to the Snapdragon GPU

Random Decision Forest - Thermal Anomaly

- Random Forest to detect thermal anomalies in images using radiance band ratios
- Classifies into the binary classes of thermal anomaly and not a thermal anomaly
- Run over Planet Skysat data (4 bands: Red, Green, Blue, NIR)
- Image labels were generated from radiance band ratios established by Ashley Davies
- Training Dataset: 13 images
  - Image size: ~ 4x10,000x12,500, pixel size = 50cm
  - 11 of Fagradalsfjall
  - 2 of Kilauea
- Test Dataset:
  - 4,000,000 pixels from a Fagradalsfjall image
- RDF: number of trees: 1,300 trees (100 trees trained on each training image), max depth: 10, weights equalized by class
- Runs single threaded on the CPU

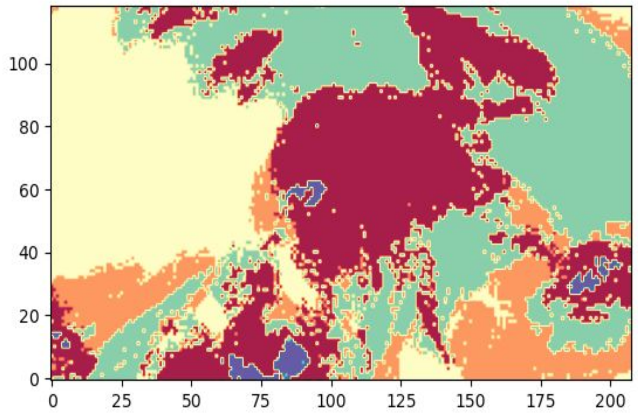
Skysat image of the Kilauea volcano in Hawaii

	SBC2	Snapdragon	Sabertooth	RAD750
ms per pixel	0.034	0.057	0.725	1.075

12.7x speedup from the Sabertooth to the Snapdragon
18.9x speedup from the RAD750 to the Snapdragon

SMICES: Machine Learning for Storm Classification

- What: Deep Convective Storm Classification using Machine Learning
- Applications: Storm targeting, Cloud avoidance
- Details:
  - Uses simulated radiometer data from Global Weather Research and Forecasting (GWRF) simulations
  - 5 classes:
    - Clear sky, thin cirrus, cirrus, rainy anvil, convective core
  - Data has 8 channels
- Single pixel classification models have been ported as ARM CPU python app on the Snapdragon and ported in C for the Sabertooth and RAD750
  - Models from scikit learn: Random Forest, MLP, Linear SVM, Gaussian Naive Bayes
  - Runtime over 198,016 pixels
  - Model accuracies same on ARM CPU as a laptop



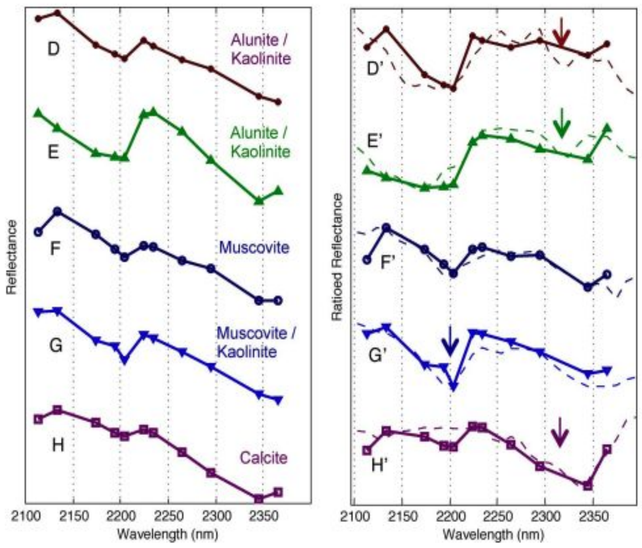
Pixel-wise pixel classification of cloud type

Time in ms per pixel	Random Forest	MLP	Linear SVM	Naive Bayes
SBC2	0.0017	0.003	0.00016	0.00025
Snapdragon	0.0025	0.0027	0.00032	0.0014
Sabertooth	0.013	0.108	n/a	0.039
RAD750	0.006	0.14	n/a	0.03
Snapdragon Speedup over RAD750	**2.4x**	**51.9x**	**n/a**	**21.4x**

SMACC

- Sequential Maximum Angle Convex Cone (SMACC) spectral endmember extraction
- Snapdragon implementation run with AVIRIS-NG Data
- Extracts the top 5 endmembers from the data
- Can be used to extract radiance values that match with minerals and other materials
- Previously flown on EO-1 with 6h runtime (incl. superpixel segmentation) on Mongoose V (smaller image cube)
- Image Size: 638x679x425
- Based on D. Thompson et al. 2012 TGARS

	SBC2	Snapdragon	Sabertooth	RAD750
ms per pixel	0.027	0.039	1	2.3



Example endmember reflectance values compared to mineral reflectance base values

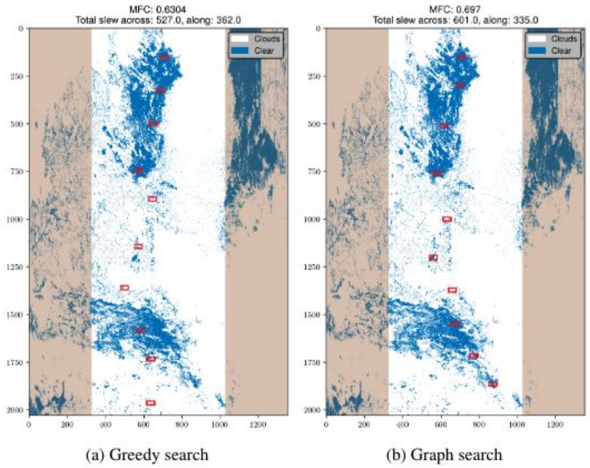
25.6x speedup from the Sabertooth to the Snapdragon
59x speedup from the RAD750 to the Snapdragon

Jet Propulsion Laboratory
California Institute of Technology

Targeting Remote Sensing Instruments

Cloud Avoidance

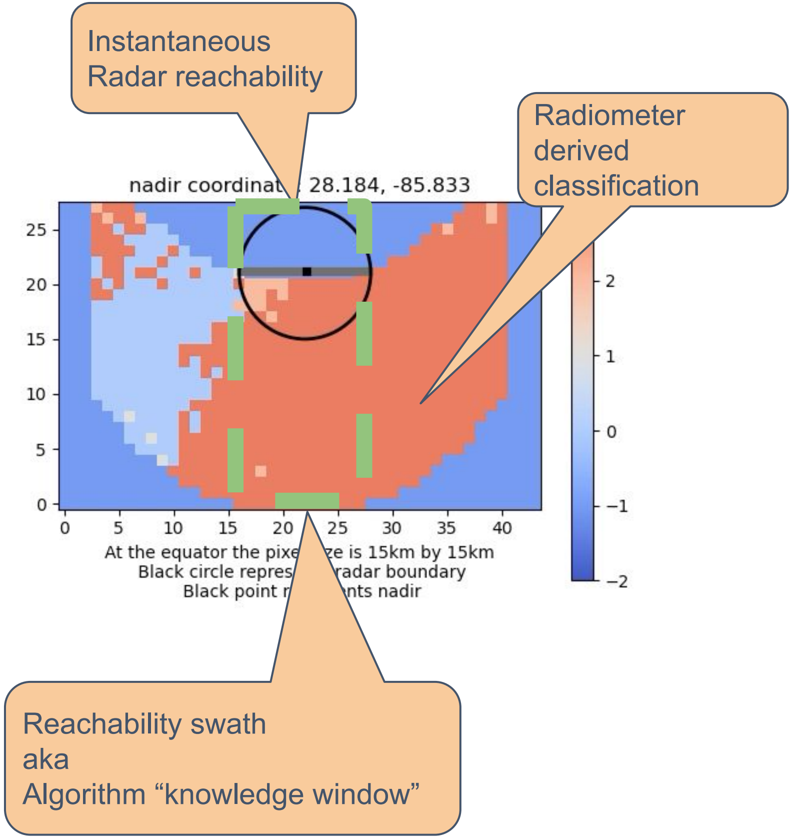
- Algorithms developed to avoid imaging clouds while in orbit given a lookahead and instrument view
- Run over 50 images with dimensions 1354x2030
- Four algorithms developed
    - Greedy search
        - Median Runtime Per Image: 49.7 ms
    - Graph search
        - Three Implementations
            - Adaptive Grids
                - Median runtime per image: 199.2 ms
            - Mixed Grids
                - Median Runtime Per image: 13.7 ms
            - Fixed Grids
                - Median runtime per image: 9.7 ms
- Implemented in Rust for the Snapdragon ARM



Visualizations of the two cloud avoidance algorithms

SMICES Pointing Planning

- SMICES Point Planning is the Smart Ice Cloud Sensing (SMICES) storm targeting planner
- Running on the Caribbean region of a Global Weather Research and Forecasting (GWRF) model's simulated data
- Uses a radiometer to identify cloud types in the orbit path and schedules a radar for further imaging
- Scientists define the value of each cloud type
- Generates an observation list based on the constraints of the radar and available clouds
- Run over 15,232 timesteps (1 timestep = 2 seconds, ~8.5 hours simulation time)
- Runtime of 53.6 seconds on Snapdragon ARM

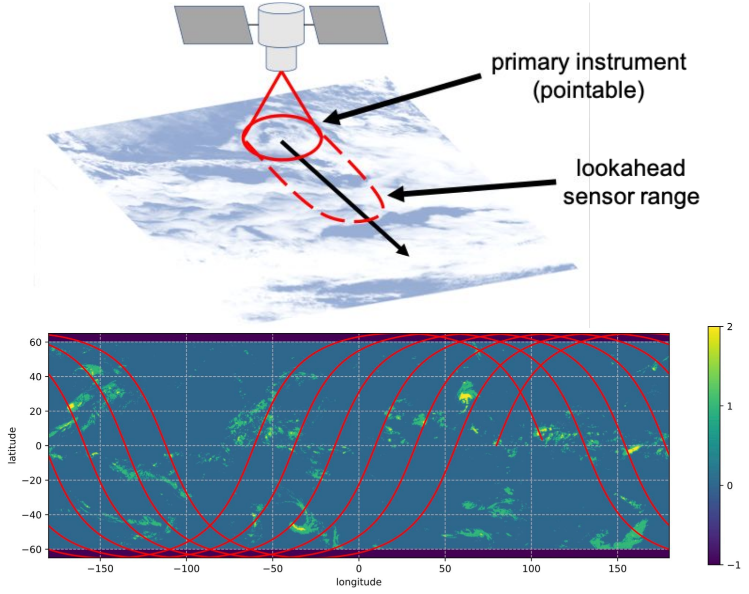


Dynamic Targeting Algorithms

- Dynamic targeting (DT) uses information from a lookahead sensor to identify targets for the primary, pointable sensor
- Improves science yield given energy constraints
- Scenario: maximize observations of storms clouds
- Simulations using global storm dataset GPM IMERG (Candela et al., 2022)
- We benchmark and compare 5 DT algorithms:
    - Random sampling at nadir (baseline)
    - Smart sampling only at nadir
    - Smart sampling along cross-path direction
    - Smart sampling within the primary instrument’s FOV
    - Smart sampling within the primary instrument’s FOV while leveraging lookahead data
- Dataset: 18,000 timesteps at 2 seconds per timestep (10 hours of simulation time, 6.3 orbits)

	SBC2	Snapdragon	Sabertooth	RAD750
ms per timestep	0.39	20	1,500	2,800

75x speedup from Sabertooth to Snapdragon ARM CPU
140x speedup from the RAD750 to the Snapdragon ARM CPU



Graphic of lookahead sensor with spacecraft targeting primary sensor based on lookahead sensor data (top)
Simulated orbit path over GPM IMERG dataset (bottom)

Jet Propulsion Laboratory
California Institute of Technology

Mission Planning

MEXEC

- Separately threaded Planner and Executive
- Takes “Task Network” as input
  - Set of state timelines, task templates, and tasks
- Generates conflict free plans and monitors task execution, responding to deviations or exogenous events

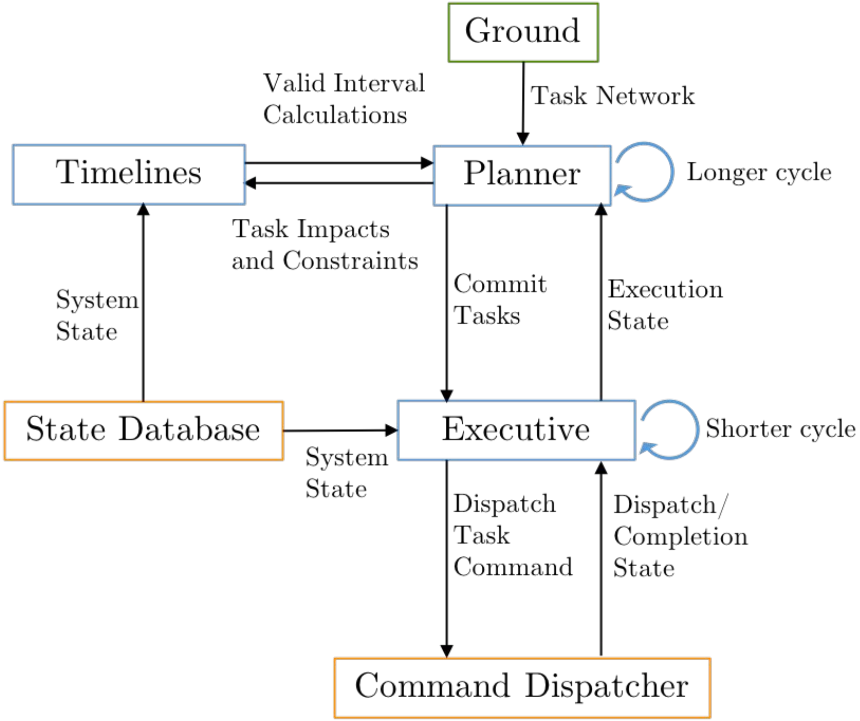
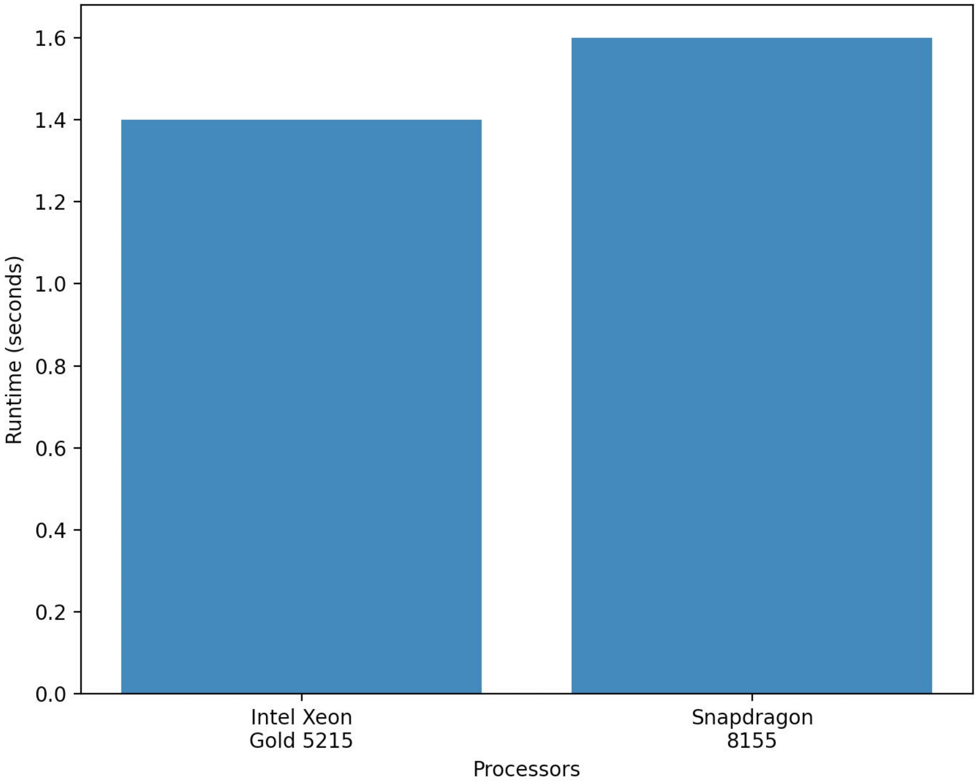


Image from https://ai.jpl.nasa.gov/public/projects/mexec/

MEXEC Benchmark Scenario

- MEXEC consists of multiple components, but the most computational demanding is the planner, so that is used for benchmarking purposes.
- MEXEC also runs continuously on a cycle, for benchmark purposes we only time the first plan generation.
- As a benchmark, we use the Europa Lander Prototype test scenario *(Wang et al. 2020)*
- Multi-day schedule, exercises hierarchical planning, valid interval search, constraint satisfaction, etc.
- Running as a single threaded ARM application

MEXEC Results

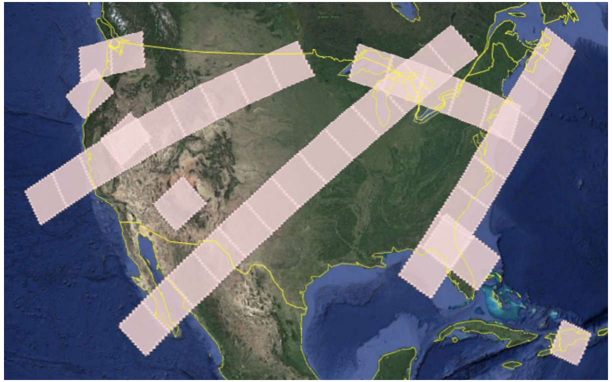


Sabertooth (scaled to final clock speed)	Snapdragon 8155 ARM
92s	**1.6s**

57.5x speedup from Sabertooth to Snapdragon ARM CPU

CLASP

- CLASP is the Compressed Large-scale Activity Scheduler and Planner
- CLASP has been used for NISAR, ECOSTRESS, EMIT, OCO3, and other missions.
- Spacecraft, instrument, and orbiting body models define the scenario
- Science Campaigns define the scientific goals
- CLASP generates an observation schedule based on the scenario constraints

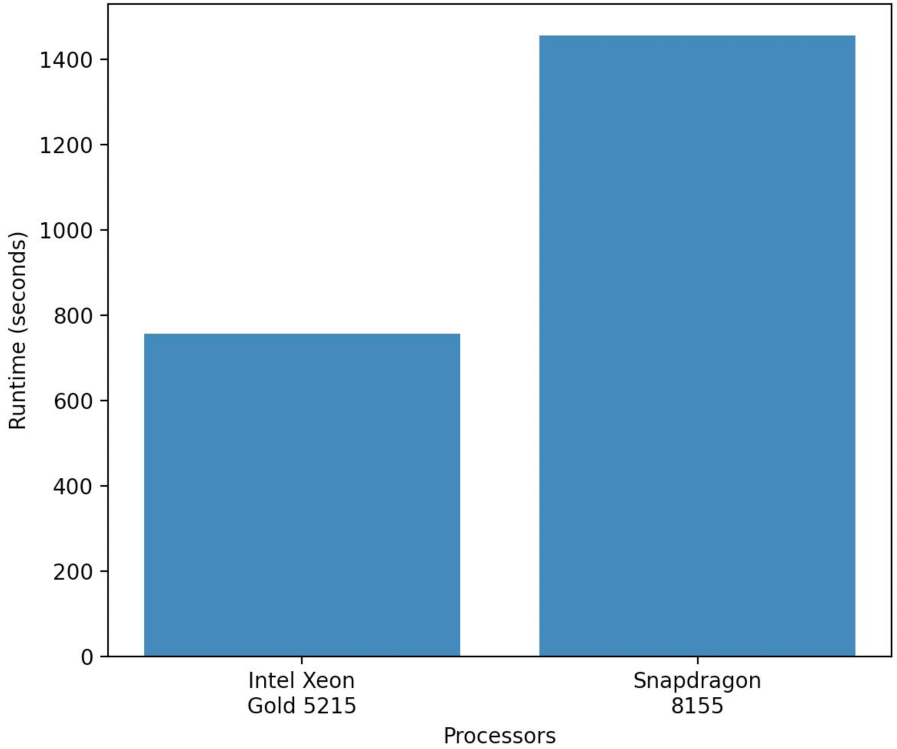


ECOSTRESS schedule portion of the Contiguous United States

CLASP Benchmark Scenario

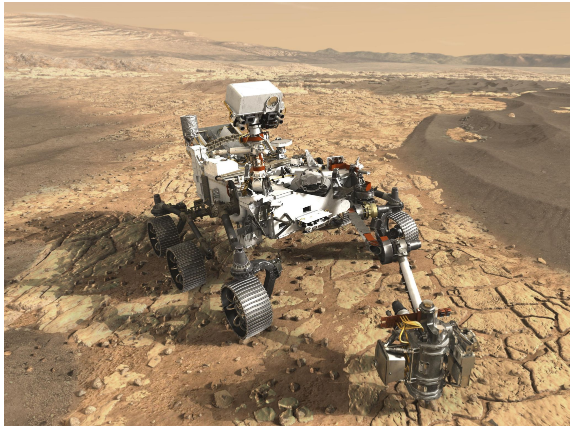
- For our benchmark, we generate 2 years of 2 week schedules using ECOSTRESS data from 2018-2020
- We generate a single 2 week schedule to collect our timing metric faster
- Currently CLASP is single threaded on the Snapdragon ARM
    - CLASP GPU port did not show significant speedup over the ARM version

CLASP Results



Copilot

- M2020 ground scheduler; currently in use for M2020 operations for scheduling wake/sleep and preheats
- Uses the same scheduling algorithms as the M2020 onboard scheduler
- Challenges include wake/sleep constraints, preheat constraints, variability in execution, and complex operations handover handling.



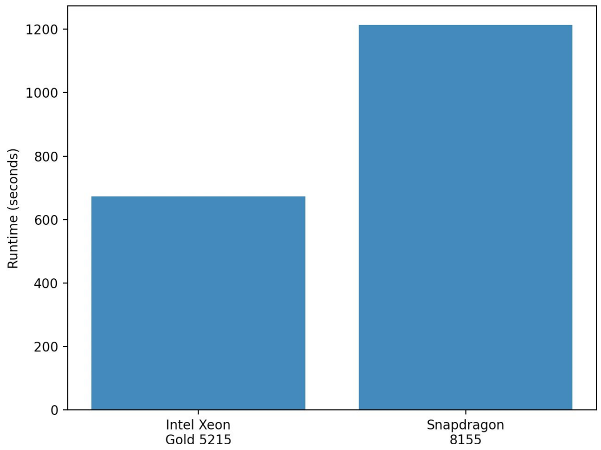
Plan for sol type “medium drive with post drive imaging”

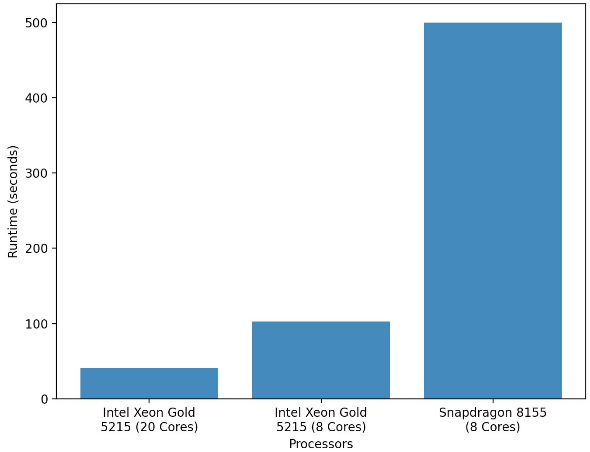
Image from https://ai.jpl.nasa.gov/public/projects/m2020-scheduler/

Copilot Benchmark Scenario

- For this benchmark, we are running with ~800 x 1 martian day (or sol) planning problems that are generated by random variation of 7 base plans or “sol types”
    - Vary execution durations, incoming/outgoing energy state, and alternative action options
- Copilot has a single threaded and multithreaded version
    - Large problem already split into 800+ small problems, so easy to parallelize
- Benchmarked against SBC2 using 1 core, 8 cores (to match 855) and 20 cores

Copilot Results





- Runtime of processors on the Copilot benchmark problem
  - Top: Serial
  - Bottom: Parallelized
- On the Intel, all 10 cores 2.5GHz
  On the Snapdragon, 8 cores range from 1.8-2.8 GHz
- Generates 800 variants, so Snapdragon takes <1s per generated plan in parallelized results

Porting to Android

- Most applications involve cross-compiling C/C++ code for ARMv8-A architecture
- CPU only ports are fairly straightforward
    - Some single threaded, some multithreaded
- Applications are ported to GPU, DSP and/or NPU with reasonable effort
- Some applications in python, ported using the python-for-android library provided by Kivy
- Long term flight usage will likely be under ADP/Linux

Conclusion

- Ported machine learning, instrument processing, scheduling, mission planning, and benchmark applications on the Qualcomm Snapdragon on ISS and compared its performance with conventional flight hardware
- Benchmarked applications on
  - **Qualcomm Snapdragon 855**
  - Intel Linux (Mac Pro) and SBC-2
  - Sabertooth
  - Rad 750
- Work intended to facilitate future flight of these capabilities to enable future single and networked autonomous spacecraft missions.

Team

- Jet Propulsion Laboratory, California Institute of Technology
  - Faiz Mirza
  - Jason Swope
  - Dr. Emily Dunkel
  - Dr. Alberto Candela Garza
  - Evan Davis
  - Dr. Zaid Towfic
  - Dr. Damon Russell
  - Dr. Joseph Sauvageau
  - Dr. Douglas Sheldon
  - Dr. Steve Chien
  - Dr. Dennis Ogbe
  - Lauren West
- Ubotica
  - Juan Romero-Cañas
  - Dr. José Luis Espinosa-Aranda
  - Léonie Buckley
  - Elena Hervas-Martin
  - Fintan Buckley
- Hewlett Packard Enterprise
  - Dr. Mark Fernandez
  - Carrie Knox

- We would also like to thank many of the applications providers who also supported the testing of their applications, including:
  - Dr. Kiri Wagstaff
  - Christopher Wells
  - Dr. Ashley Davies
  - Steven Lu
  - Michael Denbina
  - Deegan Atha
  - Michael Swan
  - Dr. Hiro Ono

Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov