A Machine Learning Toolset for PNT Threat Detection and Response

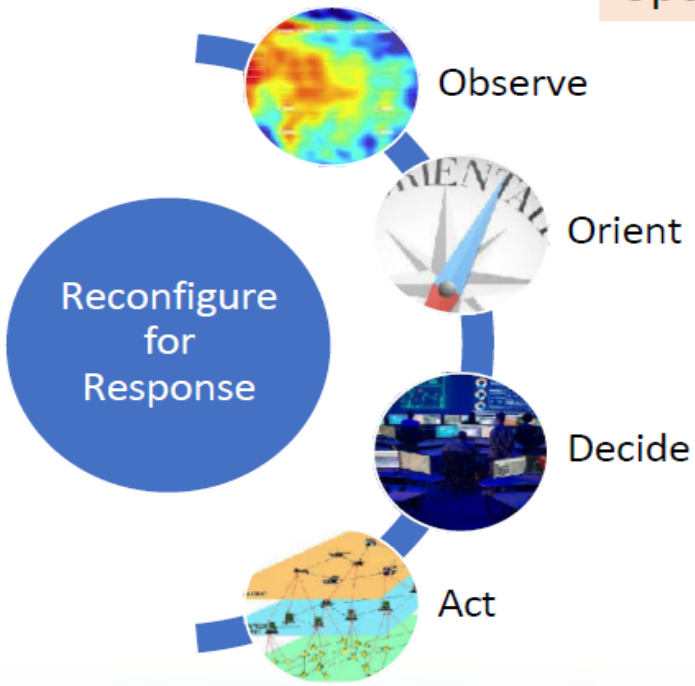
February 2023

Prepared by:
David Choi, Brian Zufelt, Renee Yazdi, Kevin Slimak

Acronyms defined in appendix

AFRL

Can We Close the OODA Loop, on the Scale of the Joint Warfighter?



Spectrum Sensing

Sensor & Shooter & Threat
- Characterization
- Known Placement in time, and trust
- Connected (Regionally and Centrally)

=> Authorities, **Tools (ML for speed),** Positioning, Interfaces/comm/connectivity, Equipment, Training

Options:
- Increase Regional Power (spot beam)
- Change Waveform or Frequency
- Initiate Avoidance TTPs
- Utilize Alternative PNT
- Excise Threat

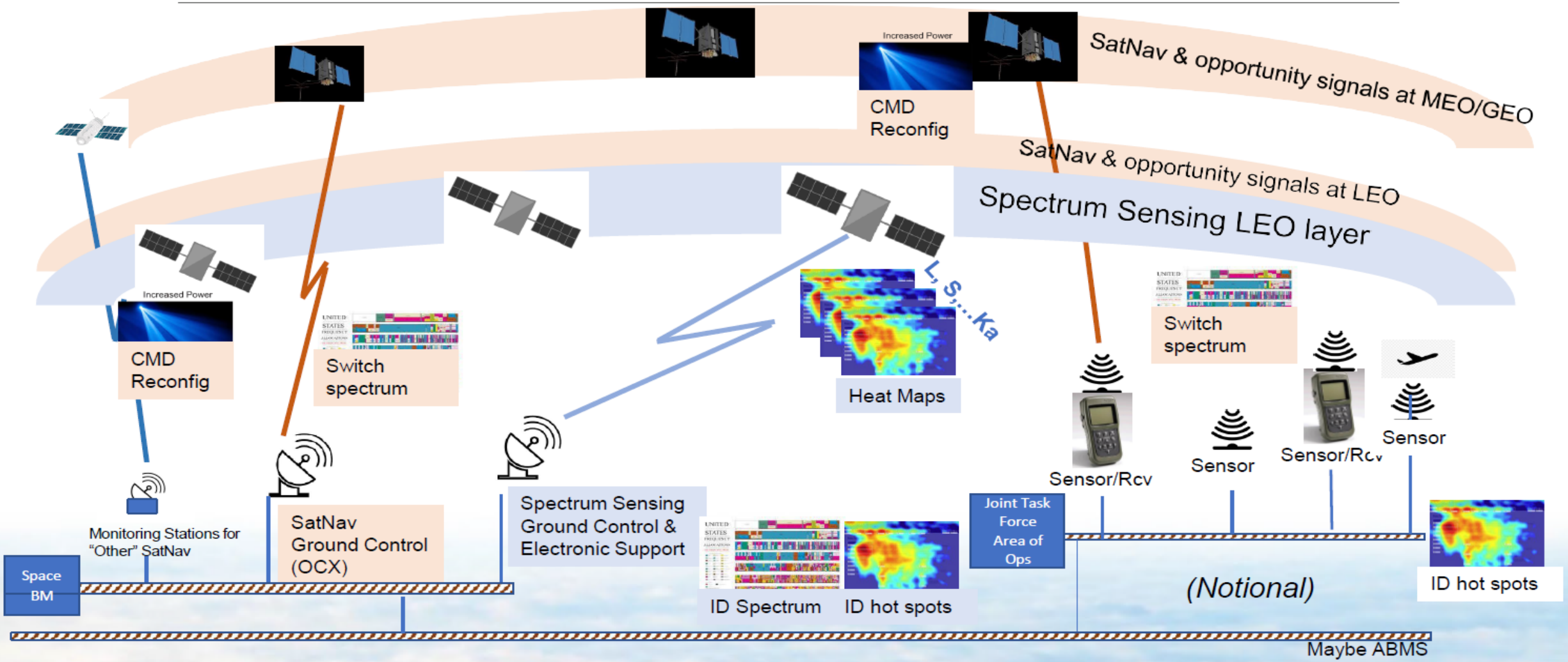
=> Authorities, **Tools (ML for speed),** Policies, Training

Orchestrated Response => Authorities, Policies, Positioning, Interfaces/comm/connectivity, Equipment, Training

Machine Learning Enables Fast OODA Detection, Characterization, and Response

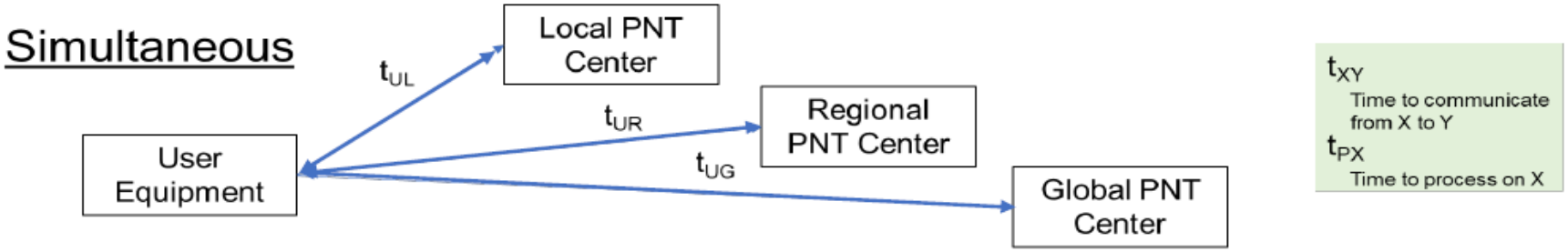
A Holistic Look is Needed –

This picture is notional superset of possibilities



AFRL Staying Aware of the Communications Cost

Simultaneous



t_{XY}
Time to communicate from X to Y
t_{PX}
Time to process on X

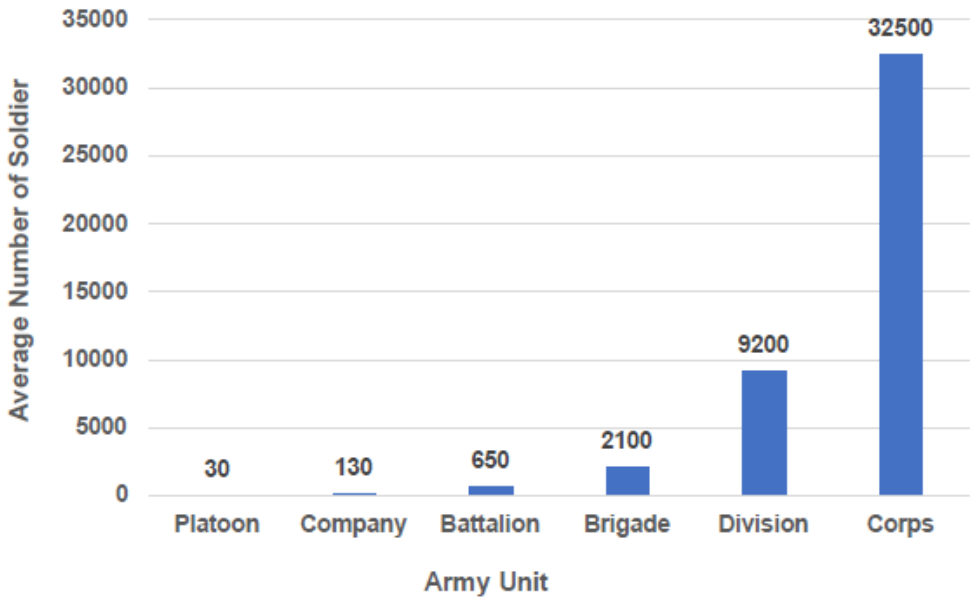
Sequential

One way transfer time relationships:

$t_{UL} < t_{UR} < t_{UG}$ $t_{UL} < t_{LR} < t_{RG}$ $t_{UR} < t_{UL} + t_{LR}$ $t_{UG} < t_{UL} + t_{LR} + t_{RG}$

- Adding in processing times increases the communication times in the sequential case
  - Even if $t_{PU} \sim t_{PL} \sim t_{PR} \sim t_{PG}$
- Sequential attempts to select a counter-counter-measure also increase the effective response time

More Levels = More Time



- Shown here is the average number of soldiers in Army units ranging from platoon to corps
- The number of pieces of UE in each unit would exceed this because of
  - UE in vehicles and equipment (may be multiple units in some)
  - Each soldier may have more than 1 (e.g., separate GPS receiver and radio with GPS receiver)
- Navy & Air Force units would further increase these numbers

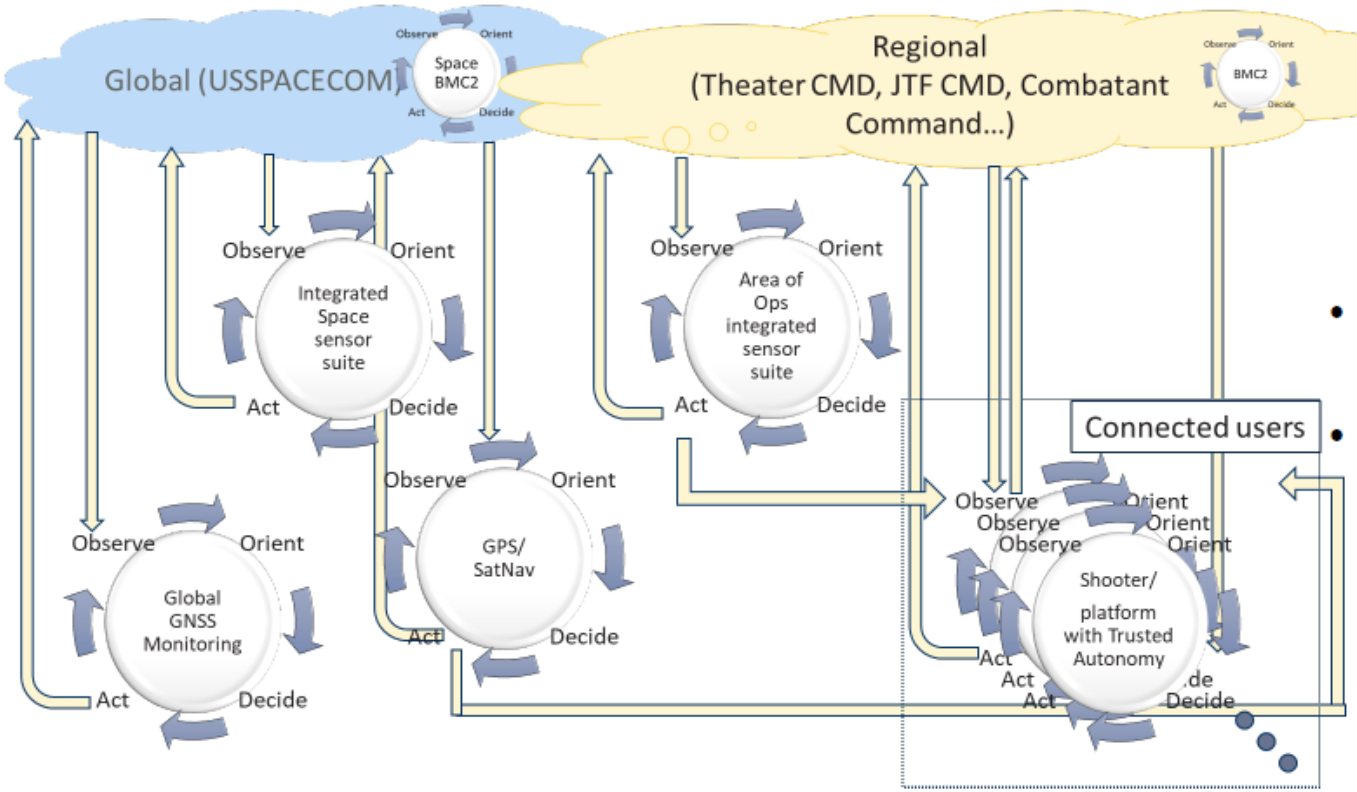
Using <u>just</u> the Army numbers (as indicative values) results in these loads to PNT Centers

Level	UE units Reporting*
Local	650-2,100
Regional	32,500
Global	1,000,000s-1,000,000,000s

* Could be lowered if UE not being jammed/spoofed does not report

Interface loading + UE SWaP considerations likely makes having all UE report RF environment impractical. So, What is the Right Answer? – Need to find the right balance

Architecture Alternative: PNT System Level– Global Solutions



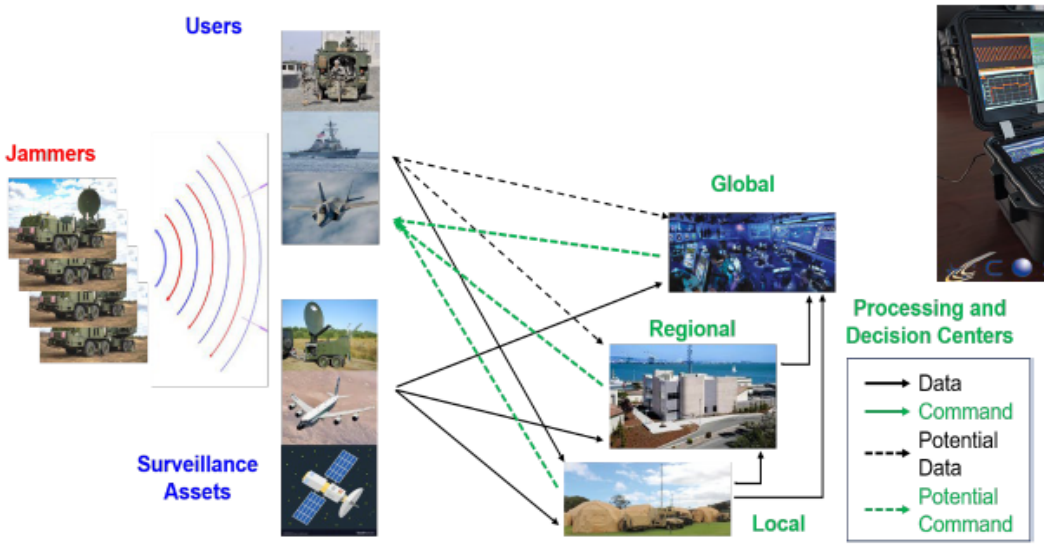
PNT System level response

- Necessary in cases where the UE, even with assistance, is unable to effectively counter the Jamming or Spoofing to which it is being subjected
- Approach could also be used in other cases

- Pros: Likely the most capable, flexible, effective option
- Cons:
  - Implementation time may be significantly greater than other options
  - Particularly if satellites need to be re-tasked
  - May not be able to support tactical level in a timely manner
  - May need to have pre-planned options at operational level
  - May have challenging comm requirements

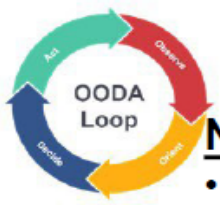
What is the real answer?- A Balance of Local, Regional and Global Solutions

A Big Problem – How do we step into it?



Warfighter Benefit Targetted

- Enables US & Allied forces to successfully counter adversary jamming & spoofing by enabling friendly forces to close their OODA Loop faster than our adversaries
  - Adapts to changing threats
  - Tightens US OODA loop by bringing the processing & information to the edge, if needed, and to a processing facility
- Increases PNT battlespace awareness
  - Provides a template into which diverse prototype efforts can provide battlespace awareness data and be fused with other efforts to achieve a consolidated threat picture



Objectives of Initial First Steps – Set foundation

- Reduce time to ID & counter new PNT threats from years days/hours, by implementing ML to
  - Identify and type adversary jamming and spoofing
  - Recommend OR develop new countermeasures
- Develop tools/techniques supporting government development and use
  1. PIGEON Sensor – senses the battlefield
  2. Data Set Library – standardized algorithm learning
  3. MLTos: Machine Learning Toolset Operating System – standard development platform
  4. Model Zoo: Reference ML Implementations
  5. Machine Learning Testbed – standard evaluation platform

Next Steps – Expand applicability

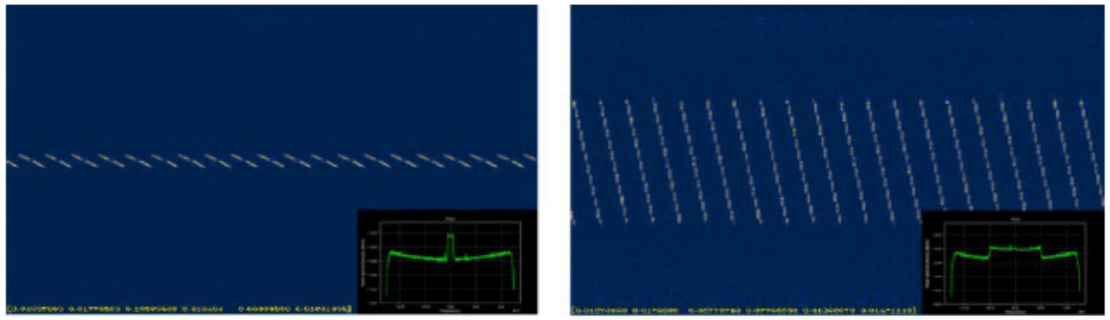
- Battlespace detection/classification: Jamming and spoofing multi-detection and classification, SV RF fingerprinting, threat localization
- Intel/Order of Battle: RF fingerprinting of threat, threat tracking
- Response: Q-Code, adaptive/ML waveforms
- Platforms: space and ground

Work to bridge the gap: fusion and dissemination

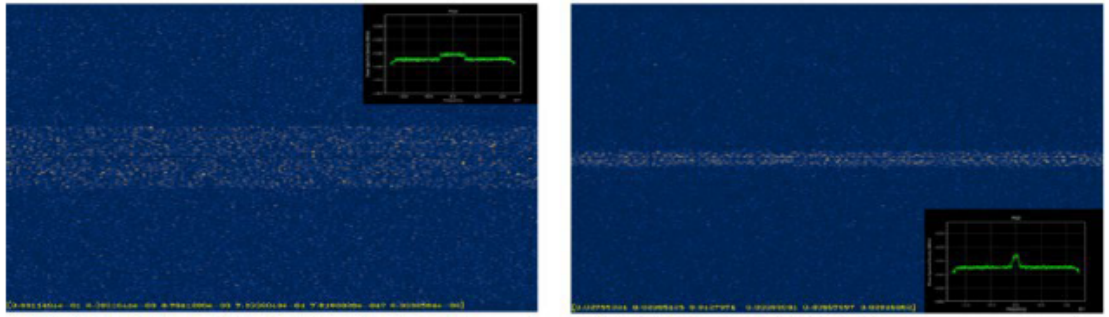
Machine Learning Toolset: Datasets

Datasets

- Real field data, **not** modelling results
  - Jamming and spoofing data sets
- Each is gathered in the field in live GNSS contested environments
  - Parameter (power, sweep, bandwidth, etc,) are varied within and among sets
- Data is available in multiple formats ranging from raw to filtered, annotated, categorized, post processed and formatted
- Unclassified data sets include CHIRP, BPSK, AWGN, BOC, PULSE, and CW jamming
- Live event dataset collection: field test
- Data supports supervised and unsupervised model training
- Library always expanding
- Current collections available for sharing
- Always looking for more data (Hawkeye 360?)

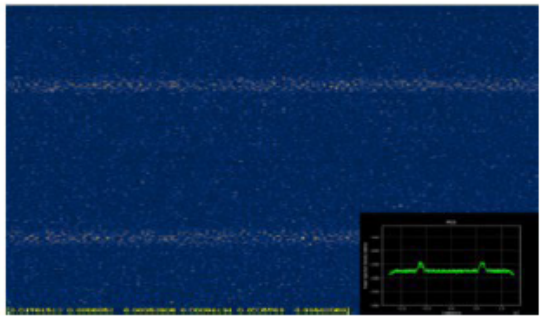


Signals with different sweep, power and bandwidth



Datasets includes a diverse collection of signals with varying parameters (power, sweep, bandwidth, etc).

Live signal collection is accomplished using different receivers and environments

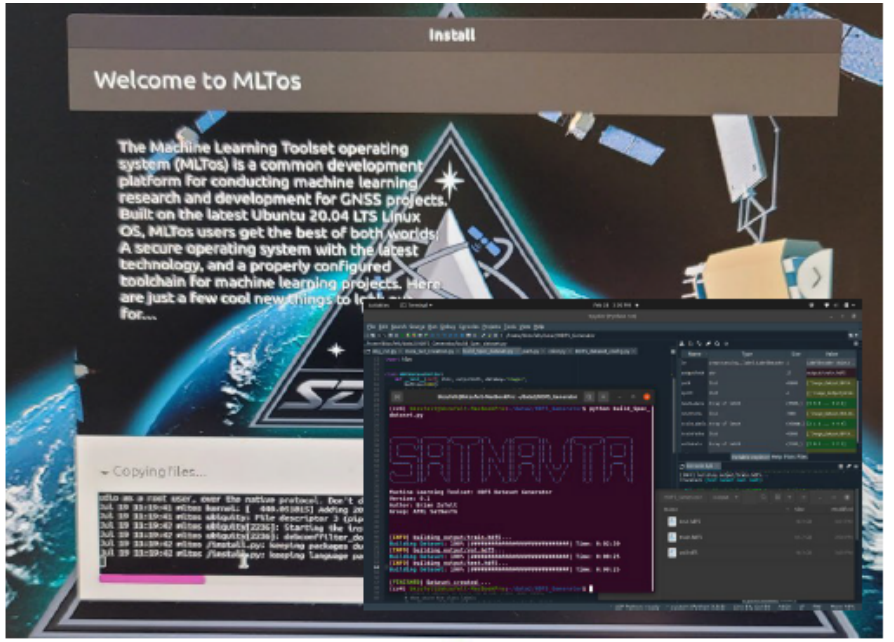


Machine Learning Toolset: Development Platform

Common Development Environment

- Provides developers with a single platform path from data set creation to model development, training, optimization, and deployment
- Provides a platform to allow multiple development teams to interact using compatible products, in a compatible framework
- Tools tested for multi-device compatibility and interoperability
- Network or stand-alone possible
- Alpha release available now
- Beta testing in process
- Custom Tools and Utilities included
- Projected availability for transition JNC 2023

MLTOS *v0.1a*

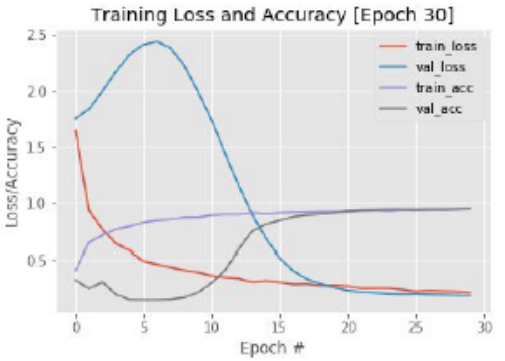


- **Amd64 (CUDA/ CPU)**
  - Nvidia (Maxwell, Pascal, Volta, Turing, and Ampere Architectures): CUDA ; CuDNN , TensorRT
  - ML Acceleration support: EdgeTPU Runtime, OpenCV 4, OpenVINO, Tensorflow Lite
  - ML Frameworks: Scikit Learn, Tensorflow 2, PyTorch
  - Software Defined Radio Support: BladeRF, LimeSDR, USRP, GNU Radio 3, GNSS-SDR; GPS-SDR-SIM
- **ARMv8:**
  - Nvidia Tegra (Maxwell, Pascal, and Volta Architectures): CUDA ; CuDNN, TensorRT
  - ML Acceleration support: EdgeTPU Runtime, OpenCV 4, Tensorflow Lite
  - Tensorflow 2, PyTorch
  - Software Defined Radio Support: (BladeRF, LimeSDR, USRP), GNU Radio 3, GNSS-SDR; GPS-SDR-SIM
  - Raspberry Pi 4 (*in development*) and Xilinx Kria (*in development*) support in development

Machine Learning Toolset: Model Zoo

- Trained, non-proprietary AI models
  - Example/demo of ML response automation to GNSS threats
- Models for user, control, and space segment
- JDCNN Classifier
  - A lite weight CNN for low-power embedded systems
  - Consumes < 2W, w/200 inferences per sec
  - Tested and validated at a field test
  - Detects and classifies 6 jammer profiles and variations (BB noise, CW BOC, BPSK, AWGN, Chirp)
- JVAE in development for data reduction
- CNN User Segment Models for detection and classification of jamming are complete
- Spoofing models are in development
- Projected availability for transition for jamming models: **NOW**, with active tech support

GPS Segment	ML Application	Application	Suggested Models
User	Detection	Detect Jamming	Jam/NoJam CNN
		Detect Spoofing	JDCNN, JDVGG CNN
	Classification	Define interference profiles	JDVGG CNN
			Autoencoder w/Latnet Space Classifier
	Transmitter Identification	Identify transmitter information (type, Make, configuration)	JVAE w/Latnet Space Classifier
	Environment descriptor	Compress environment to relay environment state	JVAE Variational Encoder
	Mitigation Control Agent	standby or decide to respond to environment	QDNN
Control	Environment Analysis	Used to Reconstruct the environment from encoded data	JVAE Variation Decoder
	Mitigation Control Agent	standby or decide to respond to environment	QDNN
Space	Mitigation Control Agent	standby or decide to respond to environment	QDNN



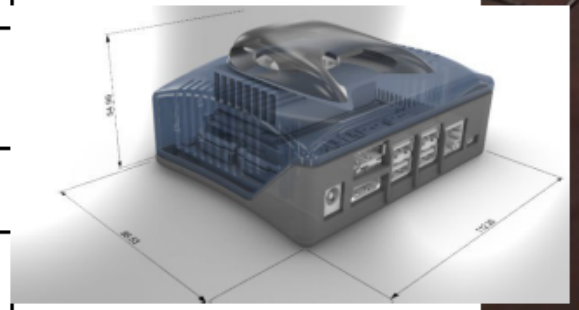

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 256, 446, 3)]     0
_________________________________________________________________
conv2d (Conv2D)              (None, 128, 223, 16)      6928
_________________________________________________________________
leaky_re_lu (LeakyReLU)      (None, 128, 223, 16)      0
_________________________________________________________________
batch_normalization (BatchNo (None, 128, 223, 16)      64
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 127, 222, 16)      0
_________________________________________________________________
dropout (Dropout)            (None, 127, 222, 16)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 127, 222, 16)      9232
_________________________________________________________________
leaky_re_lu_1 (LeakyReLU)    (None, 127, 222, 16)      0
_________________________________________________________________
batch_normalization_1 (Batch (None, 127, 222, 16)      64
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 126, 221, 16)      0
_________________________________________________________________
dropout_1 (Dropout)          (None, 126, 221, 16)      0
_________________________________________________________________
flatten (Flatten)            (None, 445536)            0
_________________________________________________________________
dense (Dense)                (None, 32)                14257184
_________________________________________________________________
leaky_re_lu_2 (LeakyReLU)    (None, 32)                0
_________________________________________________________________
batch_normalization_2 (Batch (None, 32)                128
_________________________________________________________________
dropout_2 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 198
_________________________________________________________________
activation (Activation)      (None, 6)                 0
=================================================================
Total params: 14,273,798
Trainable params: 14,273,670
Non-trainable params: 128
_________________________________________________________________
```

Machine Learning Toolset: Hardware Development Reference Design

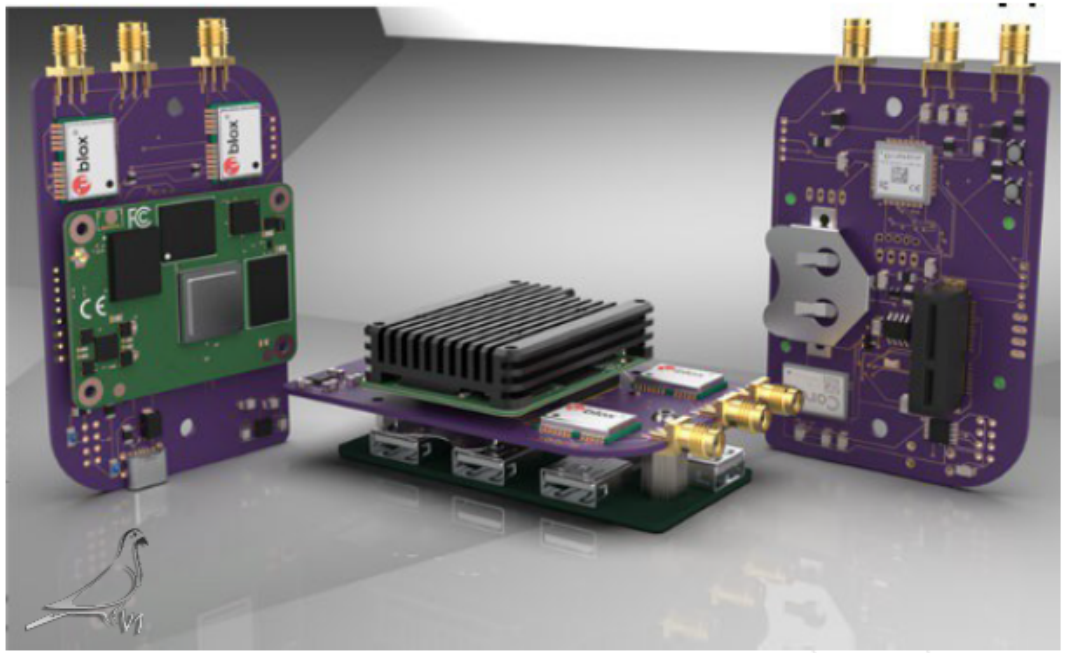
Machine Learning Testbed & Reference Unit

- Full stand-alone system to allow quick access to model testing and evaluation in the field
  - Usable for remote data collection
  - Ruggedized for field experimentation but not operational deployment
- Intended for testing and validating model performance for a wide variety of ML solutions that can later be ported to their target hardware platform for deployment

Size	30 x 24.9 x 19.6 cm
Mass	~3.5kg (Depending on configuration)
Power	<40W (Max Load)
Cost	$900 - $4,200 depending on hardware configuration and parts availability



Machine Learning Toolset: Hardware Development PIGEON



Size	96mm,58mm,24mm
Mass	~150g
Power	<15W (Max Load with USB Power)
Comm BW/Capacity	SDR Configuration Dependent
Cost	BoM < $250

Portable Intelligence Gathering Experimental Observation Node (PIGEON) Collects and encodes received spectrum (25Mhz BW, 25 Msps centered at L1)

- Situational awareness data
- Potential to develop sensor for other bands as well (not currently planned)
- Capable of controlling multiple Software Defined Radio platforms (HackRF, PlutoSDR, BladeRF, USRP)

- Various device storage configurations (32GB – 2TB)
- 4 USB3.0 ports
- On-board Tensor Processor Accelerator for ML neural network acceleration
- Data Transmit range ~8km
- Size of a deck of cards Currently in test
  - Field testing complete in March 2023
- Can be configured for Store-Forward or Store-retrieve
- V1 PIGEON available for transition Q1 2023

- Thanks for Listening…

Acronyms

- ABMS: Advanced Battle Management System
- BDS: BeiDou navigation Satellite system
- BM: Battle Management
- C2: Command and Control
- ConOps: Concept of Operations
- CMD: Command
- GEO: Geosynchronous Earth Orbit
- GNSS: Global Navigation Satellite Systems
- ID: Identify
- INS: Inertial Navigation Systems Model
- IoT: Internet of Things
- JNC: Joint Navigation Conference
- JTF: Joint Task Force
- LEO: Low Earth Orbit
- MEO: Medium Earth Orbit
- ML: Machine Learning
- MNAV: Military Navigation
- M&S: Modeling and Simulation
- NavWar: Navigation Warfare
- NTS 3: Navigation Test Satellite 3
- OCX: Next generational Operational Control System
- OODA: Observe Orient Detect Actl
- PNT: Position, Navigation, Timing
- PVT: Position, Velocity, Timing
- Q-CODE: Quick Cognitive Operationally-responsive, Dispersive, and Enhanced signals (Q-CODE)
- RCV: Receive
- RF: Radio Frequency
- RMP: Regional Military Power
- SAM: Surface to Air Missile
- SATNAV: Satellite Navigation
- SV: Space Vehicle
- SOF: Special Operations Forces
- SOW: Stand-off Weapon
- SWAP: Size Weight And Power
- TTP: Tactics, Techniques, and Procedures
- TRANSEC: Transmission Security
- TTFF: Time to First Fix
- UAV: Unmanned Aerial Vehicle
- UCAV: Unmanned Combat Aerial Vehicle
- UE: User Equipment
- UGS: Unattended Ground Sensors
- URE: User Range Error