Near Real-Time Airborne Total Propagated Uncertainty Framework

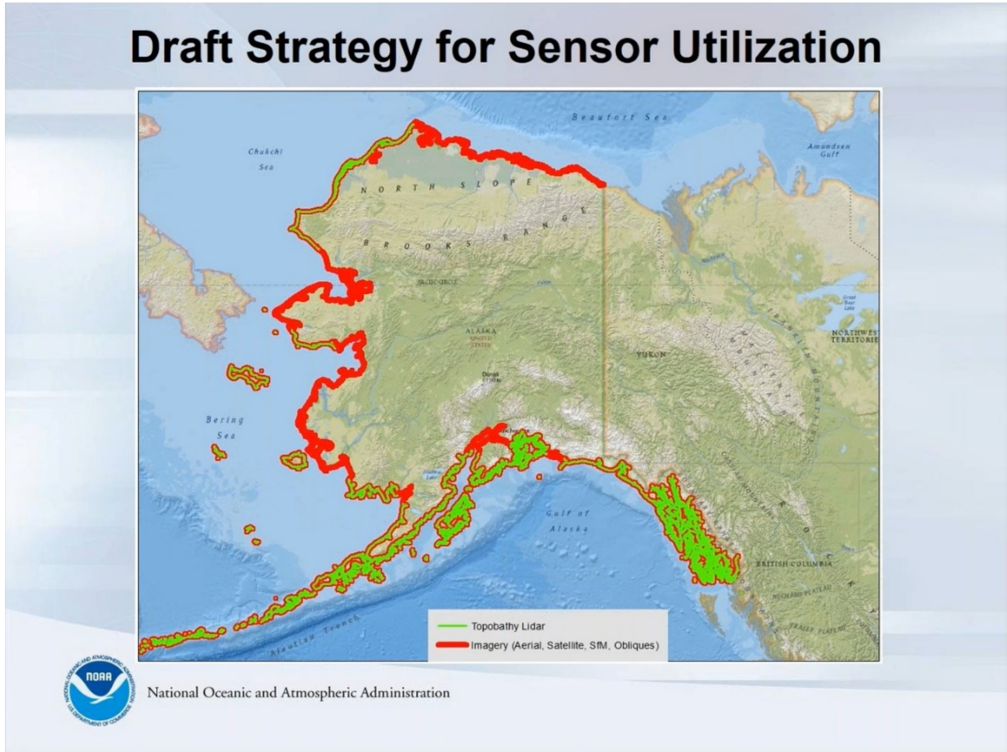
Bathy Lidar

Presented by:
Nathan Hopper Ph.D.
Woolpert Maritime Research

WOOLPERT
ARCHITECTURE | ENGINEERING | GEOSPATIAL

Outline

- Why Total Propagate Uncertainty?
- Some History
- Calculation Framework
- Modeling and Simulation

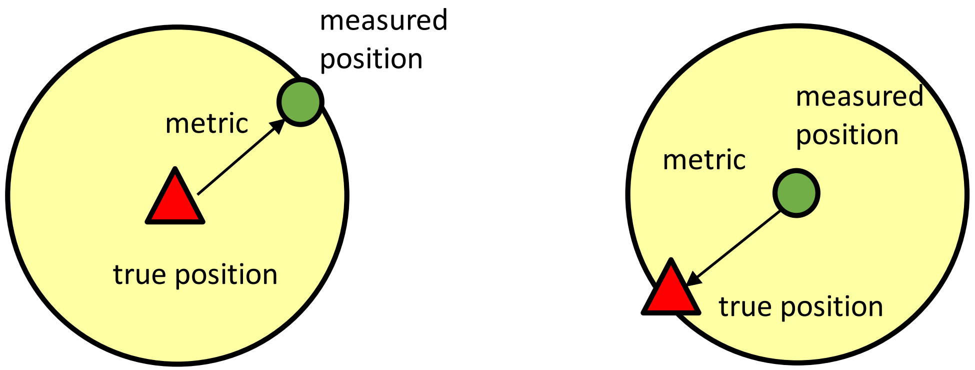


NOAA NGS November 12, 2020

Why Uncertainty?

- Any **dataset** is only a representation of reality, therefore contains **uncertainty** about the nature of the real world being represented.
  (De, Goodchild and Longley, 2007)

- Bathy Lidar Sources of Uncertainty
  - Trajectory, boresight, sensor measurements, water characteristics and ranging values with variances.
  - Total Propagated Uncertainty (TPU)



- How much is acceptable?
  - Surveying / Mapping
  - Specification / Standard

Recent TPU History

2013

JALBTCX Workshop, Mobile, AL
- G. Tuell, "Toward real-time Total Propagated Error in bathymetric lidar" (GTRI)
- **11** Parameter TPU Model

2014

JALBTCX Workshop, Mobile, AL
- G. Tuell, "More thoughts on TPU for Bathy Lidar"
- **17** Parameter TPU Model

2016

PhD Dissertation, Georgia Tech
- D. Carr, "A shift-variant restoration technique leveraging **high-resolution** interface modeling in lidar-based seafloor imaging"
- **18** Parameter TPU Model (SST)

2016

GTRI's BRDL System
- Real-time TPU
- **17** Parameter TPU Model

2017

JALBTCX Workshop, Savannah, GA
- First TPU "session"

2018

JALBTCX Workshop, Providence, RI
- Second TPU "session"
- **NOAA/UNH** TPU Model

2018

SEAHAWK System
- Real-time TPU
- **17** Parameter TPU Model

2019

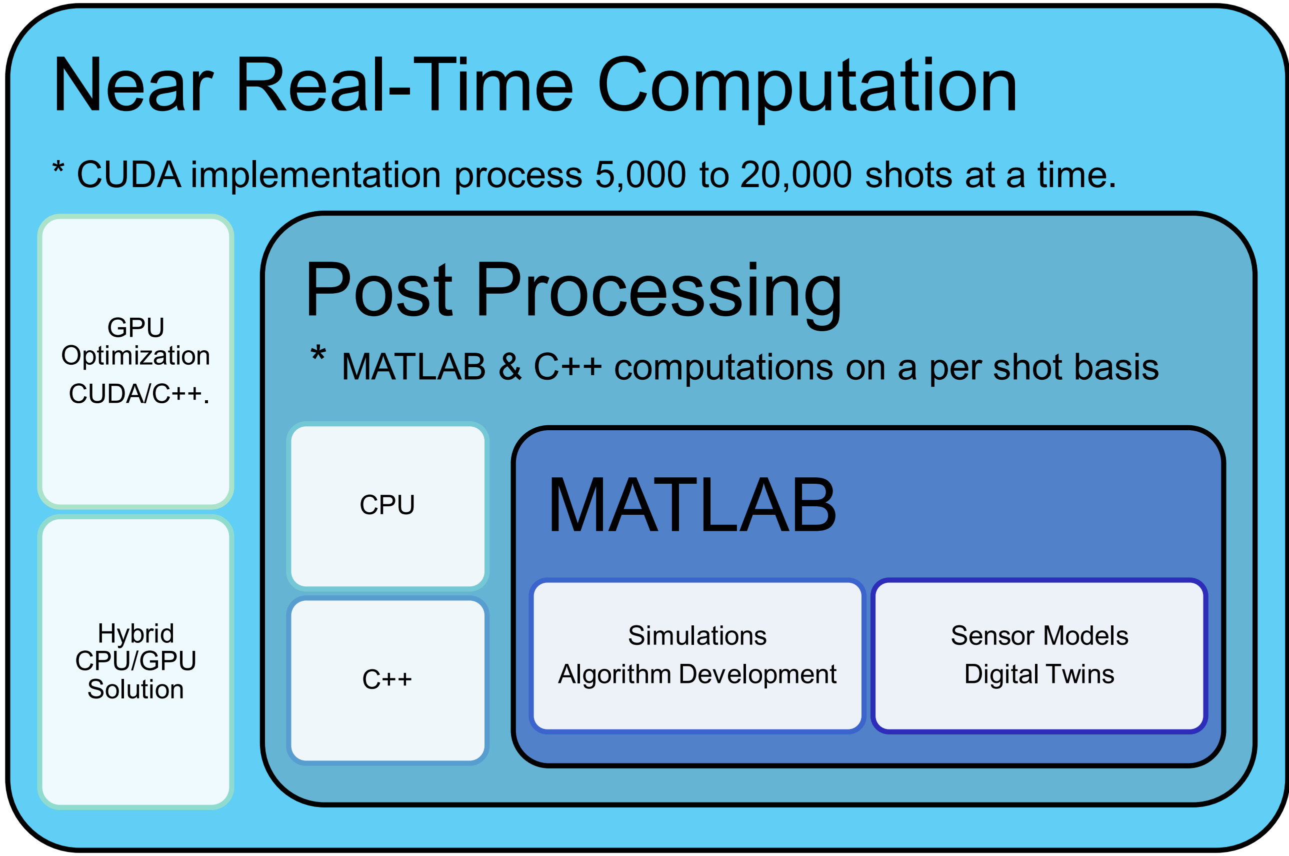
USGS 3DEP
- TPU Model

$$\begin{bmatrix} N_P \\ E_P \\ D_P \end{bmatrix} = R_Z(\kappa)R_Y(\varphi)R_X(\omega)R_Z(\theta)\left[R_Y(\phi_A)\begin{bmatrix} 0 \\ 0 \\ l_A \end{bmatrix} + R_Y(\phi_W)\begin{bmatrix} 0 \\ 0 \\ l_W \end{bmatrix}\right] + \begin{bmatrix} N_T \\ E_T \\ D_T \end{bmatrix}$$

$$\begin{bmatrix} N_P \\ E_P \\ D_P \end{bmatrix} = R_z(k)R_Y(\varphi)R_x(\omega)\left(R_z(\Delta k)R_Y(\Delta\varphi)R_x(\Delta\omega)R_z(\theta)\left(R_Y(\phi_a)\begin{bmatrix} 0 \\ 0 \\ l_A \end{bmatrix} + R_Y(\phi_w)\begin{bmatrix} 0 \\ 0 \\ l_w \end{bmatrix}\right) + \begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix}\right) + \begin{bmatrix} N_T \\ E_T \\ D_T \end{bmatrix}$$

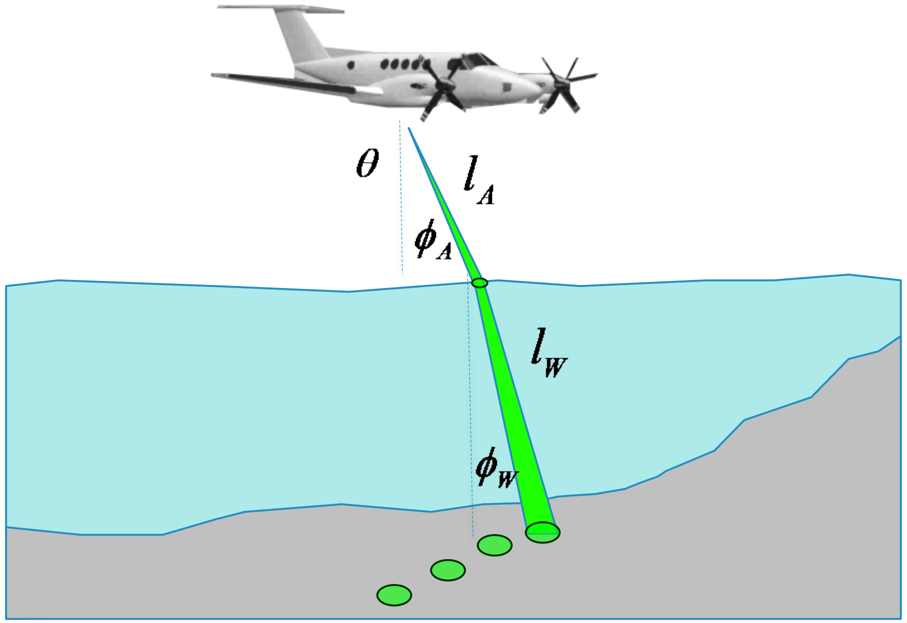
TPU Computational Framework

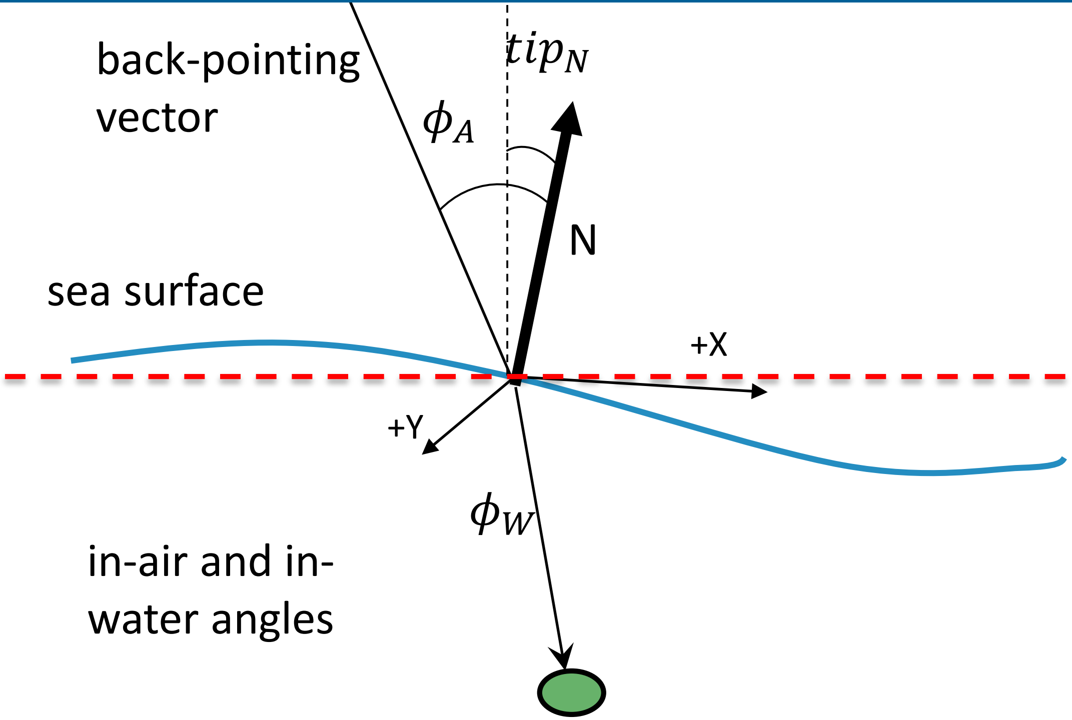
- Nested Approach
  – MATLAB
  – Implementation in C++ /CPU
  – GPU optimization in CUDA/C++
    – **Compute Unified Device Architecture (CUDA)** is a new approach to solve complex problems by transforming the GPU into a massive parallel processor
    – **140,000** TPU values per second
    – **Hybrid** CPU/GPU solution create a live point cloud with TPU in real time



Traditionally ….TPU is computed using 17-parameter model

$$\begin{bmatrix} X & Y & Z & \omega & \varphi & \kappa \end{bmatrix}$$





surface often assumed to be horizontally flat

$$\begin{bmatrix} N_P \\ E_P \\ D_P \end{bmatrix} = R_z(k)R_Y(\varphi)R_x(\omega)\left(R_z(\Delta k)R_Y(\Delta\varphi)R_x(\Delta\omega)R_z(\theta)\left(R_Y(\phi_a)\begin{bmatrix} 0 \\ 0 \\ l_A \end{bmatrix} + R_Y(\phi_w)\begin{bmatrix} 0 \\ 0 \\ l_w \end{bmatrix}\right) + \begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix}\right) + \begin{bmatrix} N_T \\ E_T \\ D_T \end{bmatrix}$$

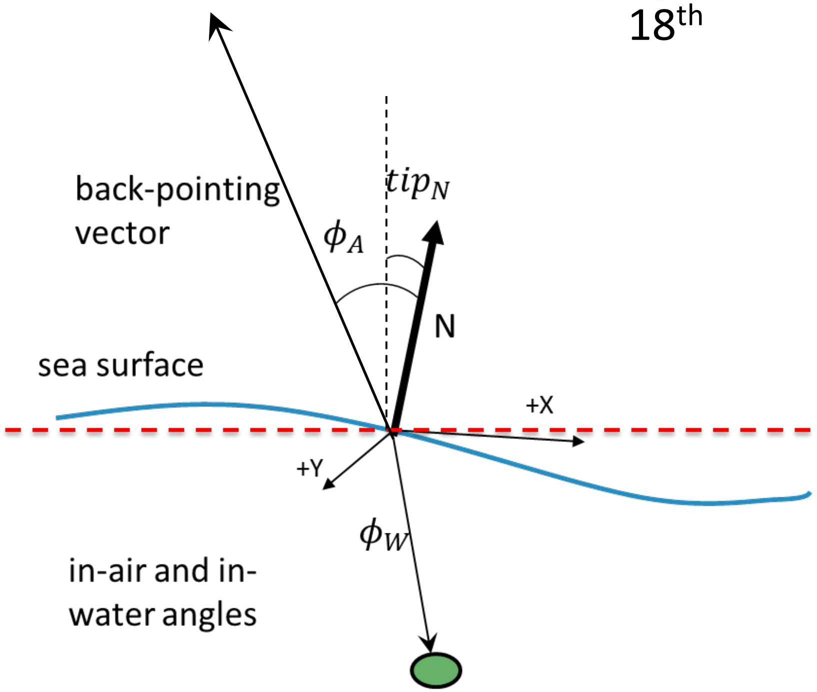
18 Parameter TPU Model

- For every seafloor coordinate

$$\begin{bmatrix} E \\ N \\ U \end{bmatrix}_M = R_Z(-h)R_x(p)R_y(r)\left[R_Z(-\Delta h)R_x(\Delta p)R_y(\Delta r)R_Z(-\theta-\Delta\theta)\left(R_x(\phi_A)\begin{bmatrix} 0 \\ 0 \\ -l_A \end{bmatrix} + R_x(\phi_W)R_y(\omega_W)\begin{bmatrix} 0 \\ 0 \\ -l_W \end{bmatrix}\right) + \begin{bmatrix} \Delta Y \\ \Delta X \\ -\Delta Z \end{bmatrix}\right] + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

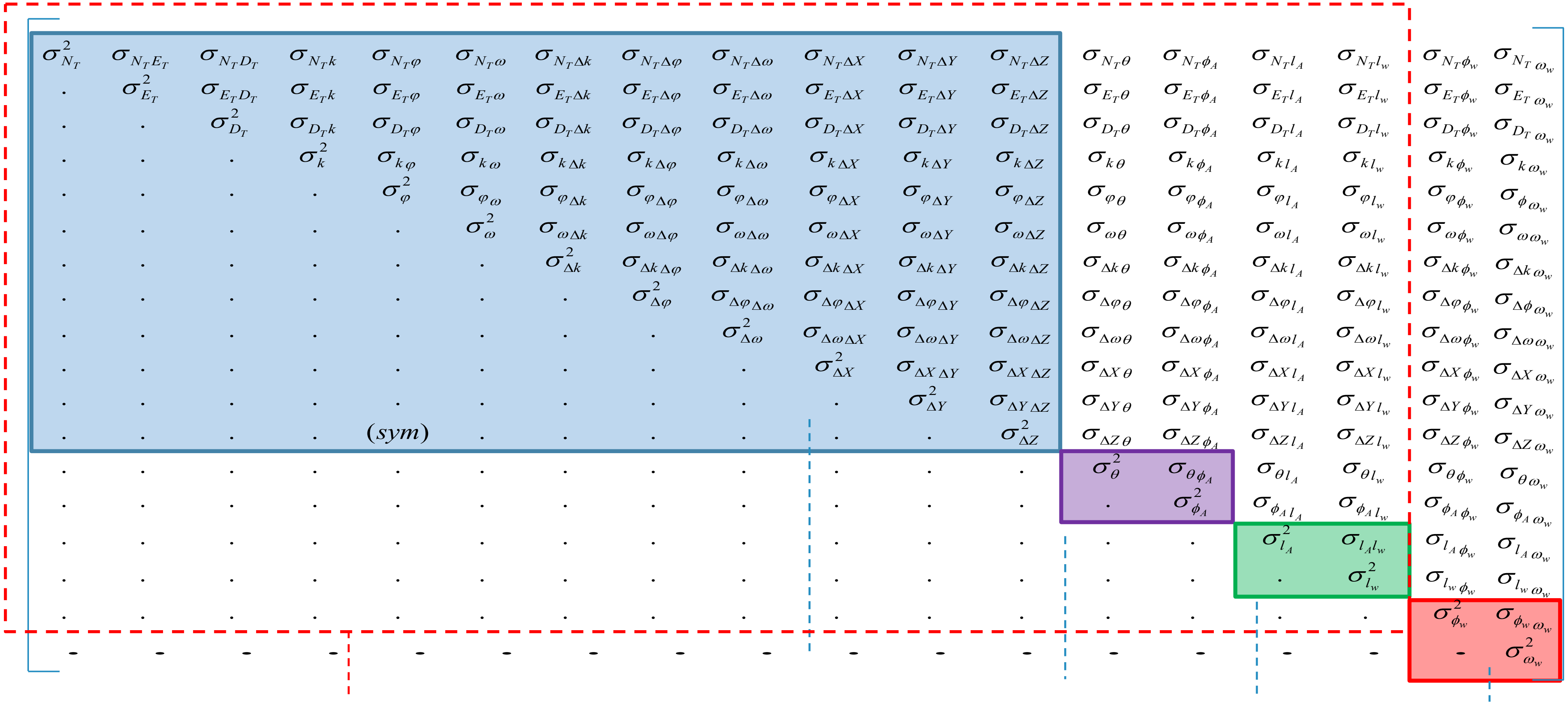
18^{th}

- 18 Parameters
  - Trajectory
  - Boresight
  - Sensor measurements
  - Water characteristics
  - Ranging values with variances
  - **Sea Surface Topography**



TPU Computation

- A series of matrix operations compute the partial and inner derivatives



$$\tilde{\Sigma}_M(1:16;1:16) = \Sigma_M(1:16;1:16)$$

From Tuell (2014)

TPU Computation

- 3x3 matrix of variance – covariance

$$\Sigma_V = \begin{bmatrix} \sigma_E^2 & \sigma_{EN} & \sigma_{EU} \\ \sigma_{NE} & \sigma_N^2 & \sigma_{NU} \\ \sigma_{UE} & \sigma_{UN} & \sigma_U^2 \end{bmatrix} = J\Sigma_M J^T$$

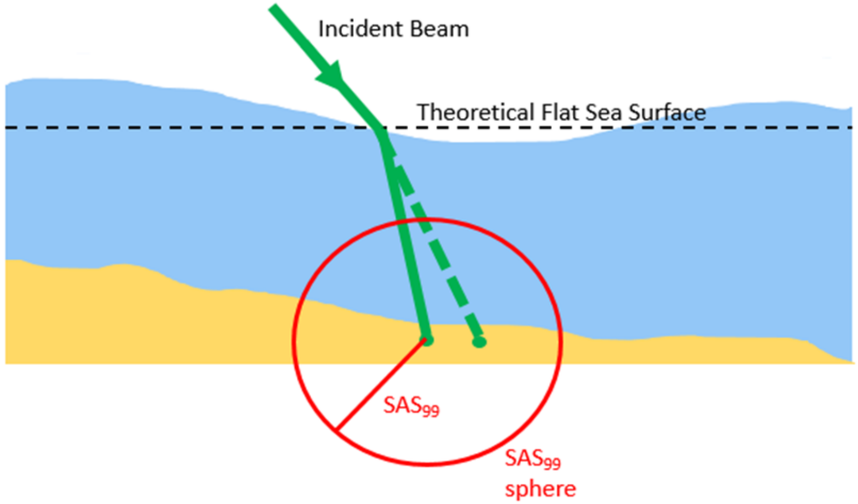
3 variances representing ENU
6 representing off-axis components

- Single value representing accuracy

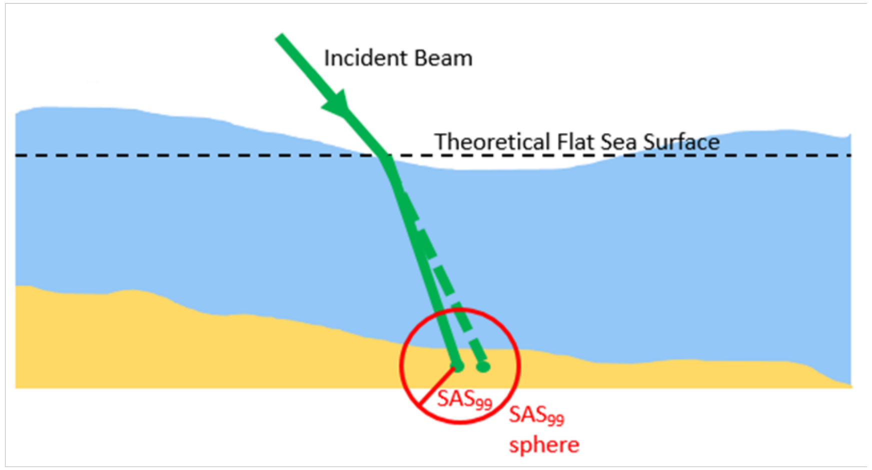
$$SAS_{99} = 1.122(\sigma_N + \sigma_E + \sigma_D)$$

Spherical Accuracy Standard 99%

Flat Surface



SST Measurements



Geo location error and uncertainty are amplified when a flat surface is assumed.

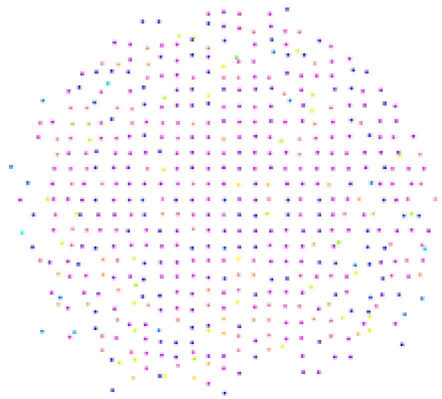
High Resolution SST

- Segmented detectors on Co-located Green and IR.

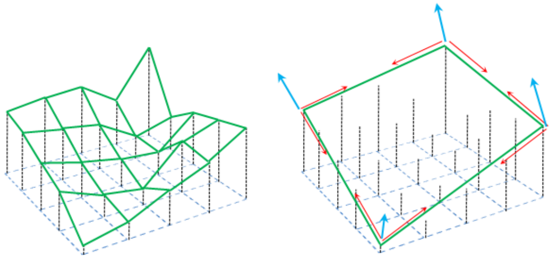
Guida (2015) Method

2	0	85	84	0
0	83	88	87	88
0	0	84	85	0
0	0	83	85	0
0	0	0	83	0

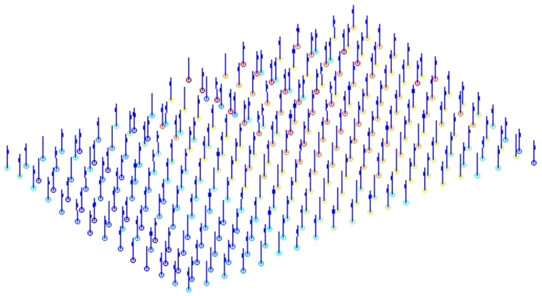
Capture Geiger-mode time-of-flight measurements



Compute water surface coordinates



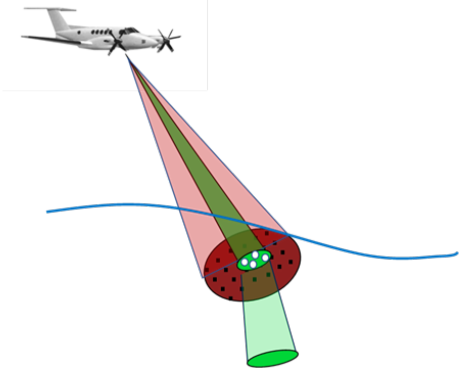
Create surface facets



Compute surface normals

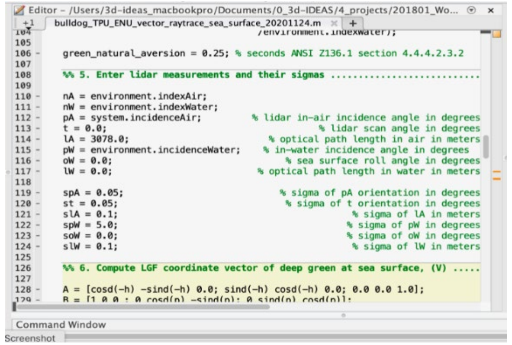
From Carr (2016)

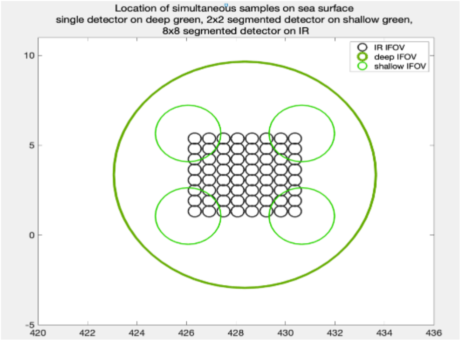
Development Design and Testing

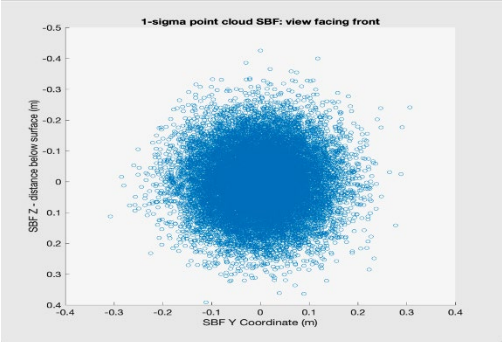


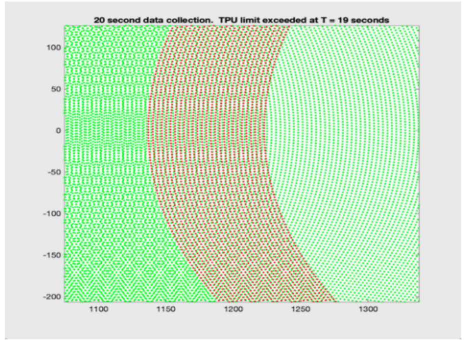
MATLAB + Simulink

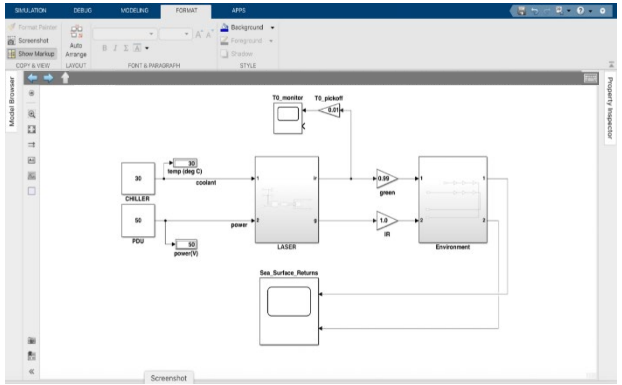
Simulations Algorithm Development	Sensor Models Digital Twins

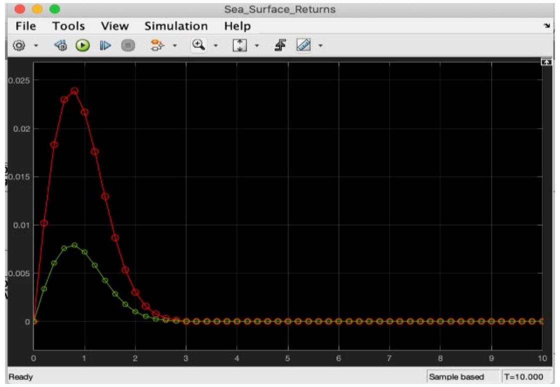


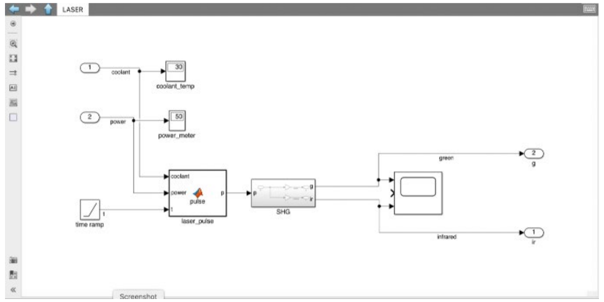


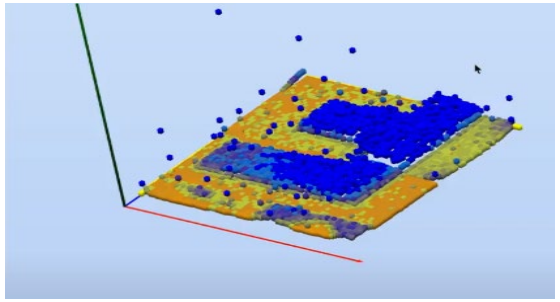












SUMMARY

- A GPU accelerated TPU calculation from multiple channels, each creating thousands of waveforms per second is achievable, reliable, and useful.
- Near real-time TPU will be especially beneficial for assessing data quality when performing airborne surveying operations in remote locations.

Acknowledgments

- Dr. Grady Tuell
- Dr. Joong Yong Park
- Dr. Joshua Gluckman
- Mark Millman
- Eric Cahoon

Speaker Contact Information

Nathan Hopper, Ph.D.
Woolpert
nathan.hopper@woolpert.com
601.916.4121