Background
The Palouse Robosub Club participates in an annual competition for designing an autonomous submarine to complete challenges in a US naval sonar testing pool. To navigate the pool, the submarine needs to estimate its location in the pool using environmental data. To meet this need, Curiosity Senior Design team deployed a system called PeRLS: the Palouse Robosub Localization System. PeRLS consumes RoboSub sensor data and, using a particle filter and Kalman filter, provides a real-time estimate of the AUV's position in the pool.

Goals
- AUV capable of self-localization accurate to 1 meter
- Consume sensor data from hydrophones, IMU, and depth sensor
- Localization system robustness
  - Gracefully recover from one of three sensor inputs failing
  - Test/verify system functionality in simulation prior to production use

Hardware Resources
- Intel NUC: i5 Skylake Processor, 8GB RAM
- 4 BlueRobotics Har30 300m High-Res depth sensors
- Bosch BNO055 (IMU) for linear acceleration
- Tri-M Technologies PNI TRAX (IMU) for orientation
- 4 hydrophones with Zynq 7010 FPGA interface
- Power Distribution/Shielding Materials as needed

Software Block Diagram

Point-Cloud/Particle Filter Algorithm
Particles randomly distributed as “cloud” around pre-determined point in simulated 3D space. Each point represents a hypothesized location.

Pinger/hydrophones provide relative bearing towards pinger. Point-cloud probability distribution resembles hollow, "vectored" cone.

IMU provides world-framed orientation data for AUV, restricting possible “vector”. Cone of probability is reduced to a ray.*

*Run concurrently with previous step for optimization purposes

Depth sensor provides world-framed elevation data for AUV. Ray of possible locations reduced to cross-section - position estimated at weighted mean of point-cloud.

Point cloud particles shifted (with noise) based on velocity estimated via Kalman filter (see right). Particle Filter re-starts at Step 2.

Kalman Filter Algorithm
Kalman filters are a class of algorithms designed to unify one or more noisy and/or intermittent signals into an estimated output signal based on the prior behavior of the input signals. This "smooth" output estimate is often more accurate than the noisy input values, and has powerful applications in time-series data analysis scenarios - including signal processing and navigation.

PeRLS leverages a Kalman filter to provide an accurate estimate of linear velocity based on position and linear acceleration, which is used to shift the point cloud by the appropriate amount (see left) between iterations.

Future Expansions
- Supplement velocity estimate with optical flow from bottom camera
- Localization without initial position parameter
  - Current algorithm requires (parameterized) initial position of AUV relative to pinger
  - "Natural" (Angle-based) pinger-bearing hypothesis validity calculation
- Integration of additional positional-data sources (e.g. vision system)

Glossary
- AUV: Autonomous Underwater Vehicle
- KF: Kalman Filter (noisy signal estimator - see above)
- IMU: Inertial Measurement Unit (Compass, Gyroscope, Accelerometer)
- NUC: Next Unit of Computing; Small form-factor x64 PC
- PF: Particle Filter (discrete solution approximation method - see left)
- ROS: The Robot Operating System (FOSS middleware for robotics)

Acknowledgements
- NAVSEA – sponsorship; weekly consults with engineering mentors
- Dr. Aaron Crandal – Mentoring and direction
- Palouse Robosub Club – Hardware installation, software assistance, club leadership, pre-existing designs/specifications
- Dominic at Travis-CI – Assistance in repairing CI workflow