**Pose Estimation**

The six degree of freedom pose of the center of mass of the UAV in the world frame is (x, y, z, Φ, θ, ψ), where Φ is roll angle, θ is pitch angle, and ψ is yaw angle.

Body frame is the center of mass of the UAV. xb is the preferred forward direction and zb is perpendicular to the plane of the rotors pointing vertically up during perfect hover.

Use ZXY Euler angles to model rotation in the world frame. To translate from body frame to world frame:

1. rotate about the zw axis by the yaw angle, ψ
2. rotate about the intermediate y axis by the pitch angle, θ
3. rotate about the xb axis by the roll angle, Φ

Rotation matrix to transform from body frame to world frame is

R = R(zw, ψ)R (y, θ) R(xb, Φ), where the Rs are elementary rotation matrices with respect to the x, y, and z (see [3] pages 4-9 for additional details).

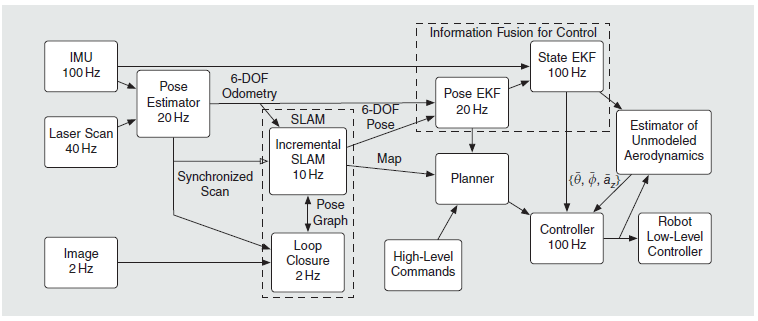


Figure 1. The architecture diagram with update rates from [1].

The Iterative Closest Point (ICP) Algorithm (see [2] for an overview of the concept) finds the transformation between two sets of data points and provides an estimate of (x, y, ψ). This algorithm resides in the *Pose Estimator* process in Fig. 1. It will be necessary to improve on the speed of the basic algorithm.

The basic *ICP algorithm*:

*Input: A reference point cloud and the new data point cloud.*

*Output: An estimate of (x, y, ψ)*

*Step 1. Preprocessing cleans the data.*

*Step 2. Matches the associated points from the reference to the data using the neighbor search. It can use features to identify associated points.*

*Step 3. Weighting changes the importance of some pairs.*

*Step 4. Reject some of the pairs*

*Step 5. Compute the error for each pair and where they should be located.*

*Step 6. Find the best transformation (minimization)*

*Step 7. Loop back to step 2, unless there is convergence.*

Step 2 is very computationally intensive. Penn uses a grid-based search [4] to reduce the computation requirements of the closest point search. Another resource reduces the computational complexity [5], but I have not studied it enough to determine if it will reduce the computation enough (but it does go through the steps in more detail).

Chetverikov ‘s paper [4] effectively breaks the point clouds down into regions (boxes). The paper uses squares, but the regions can be rectangles or arbitrary shapes. We can probably use squares or rectangles. This decomposition occurs prior to the neighborhood search.

SLAM TEAM ADD Chetverikov’s algorithms here… You need both the Boxing and search algorithms.

The output of the *Pose Estimator* and the *Incremental SLAM* processor are the inputs to the *Pose EKF* process. This process combines (fuses) the IMU data with the corrected laser range finder scans and provides altitude estimates. Penn uses a variant of work done at the University of Freiburg [6]. Note that Freiburg also uses mirrors with their LRF.

The input to the variant of the Kalman Filter (KF) [6], which actually uses two KFs. Frieburg’s paper does an excellent job of describing the algorithm in Fig 2, and the team needs to read this paper. Note also that [6] has a better picture of the mirrors they use for the LRF.

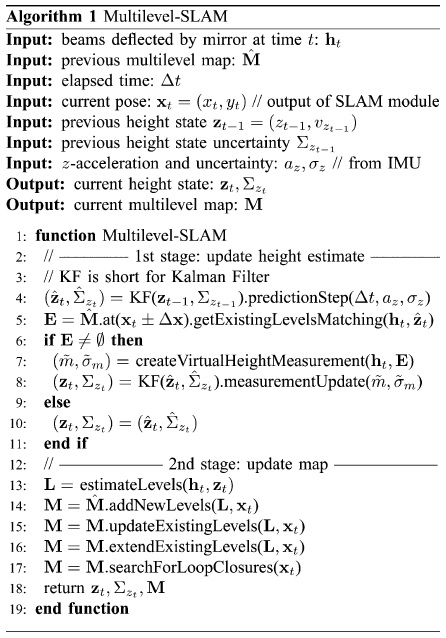


Figure 2. The Multilevel-SLAM algorithm from Univ. of Frieburg [6] that Penn uses a variant of the Kalman Filter, lines 3-11.

Penn estimates (Φ, θ) based solely on the IMU data.

The *Incremental SLAM* process completes portions of Freiburg’s algorithm, specifically, the matching (lines 5-10) and step 5 of the ICP algorithm. STILL NEED TO FINISH!

**References**

1. Shen, S., Michael, N., and Kumar, V. (2013) Obtaining liftoff Indoors: Autonomous Navigation in Confined Indoor Environments. *IEEE Robotics and Automation Magazine*, 20(4):40-48.
2. Iterative Closest Point Algorithm – overview. <http://en.wikipedia.org/wiki/Iterative_closest_point>
3. Kumar, V. (2014) Phase 2 MEAM 620 Project notes slides : <https://alliance.seas.upenn.edu/~meam620/wiki/index.php?n=Main.Projects?action=download&upname=2014_proj1_phase2_slides.pdf>
4. Cherverikov, D. (1991) Fast neighborhood search in planar point sets. *Pattern Recognition Letters,* 12(7):409-412.
5. Rusinkiewicz, S. and Levoy, M. (2001) Efficient Variants of the ICP Algorithm, *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling.* <http://www.cs.princeton.edu/~smr/papers/fasticp/>
6. Grzona, S., Grisetti, G. and Burgard, W. (2012) A fully autonomous indoor quadrotor. *IEEE Transactions on Robotics*, 28(1): 90-100. <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6004839&tag=1>