**Algorithm Input**: 4-by-x matrix contains 4 sets of variables; frame, x, y, z

1. Sorts the frames and makes sure the frames are continuous
2. Merges frames based on *mergedF*
3. Filtering begins
   1. Builds a new frame and sorts the frame
   2. Produces the matrix containing the x, y, and z value corresponding to the sorted frame
   3. Counts the frames produced
4. Set up current point
5. Build up first and second frames
   1. KN filter used for the first frame model
6. Calls ***icp2***, which performs the iterative closest point algorithm on the 3-dimensional point cloud
7. Loop over the matrix to calculate the displacement and the trajectory in 3D space
8. Loop over each position and do the optimisation
   1. Calculate the average distance between each point which is used to calibrate the trajectory
      1. Check if the *difference* between the **current** and **next** point on the calculated trajectory is larger than the defined **threshold**
         1. If the difference exceeds the threshold, the **previous** value of the average distance is assigned
   2. Re-calculate the position information based on the optimised translation and rotation matrix from the step above
9. Work out the 2D projection and displacement information from the projection matrix
10. Calculate the 2D measurement given the 3D environment data