Subject: yet another 2d matching question
Posted by Gray on Fri, 30 Jul 2010 14:01:22 GMT
View Forum Message <> Reply to Message

Hi all,

For quite a while I've been using JD Smith's match_2d routine to match xy coords between lists. However, this and all the other matching codes I've seen out there suffer from a variation of the uniqueness of matches problem.

Codes like SRCOR in the NASA IDL library let you specify a one-to-one match, i.e. enforcing that each element in list 2 only be matched to one element in list 1; using match_2d's match_distance keyword one could implement the same effect oneself. However, while that excludes multiple matches to the same element, it's all done after the fact, after the original match was determined.

What I'm looking for is an algorithm that matches 2 lists, identifies multiple-matches, and then looks for additional matches within the search radius for elements which would become unmatched after enforcing a one-to-one relationship. What I mean is, say element 0 in list 2 is matched to both element 3 and element 5 in list 1, and that the distance between 2_0 and 1_3 is smaller than the distance between 2_0 and 1_5. In that case, 1_5 would become unmatched; but what if there is element 2_1 which is also within the search radius of 1_5? Then, 1_5 should be re-matched with 2_1.

My best idea thus far is to run match_2d once, identify multiple-matches, keep the matches with minimum distance using match_distance, then iterate with the remaining elements until match_2d returns no matches. Can anyone come up with a better solution?

--Gray