Subject: Re: x-y offsets
Posted by Craig Markwardt on Thu, 20 May 2010 01:45:18 GMT
View Forum Message <> Reply to Message

On May 18, 4:29 pm, Gray <graylikethec...@gmail.com> wrote:
> Hi all,
>
> This is a variation on the 2D matching problem that I'm having trouble
> algorithm-ing (to coin an incredibly awkward word).
>
> I have two sets of XY coordinates of unequal length (i.e., x1/y1/n1,
> x2/y2/n2, n1 ne n2). I want to find offsets in both X and Y that
> match the two sets as closely as possible (there will obviously be
> some unmatched coordinates in the larger set). I'm just looking for
> constant offsets, so basically (for n1 < n2) x1 + Cx -> x2, y1 + Cy ->
> y2, with some elements of x2 and y2 being unmatched. How do I go
> about doing this? I don't think I can use JD's MATCH_2D because I
> don't know a priori what my matching radius is.
>
> Any suggestions? Thanks, as always!

This seems like a fragile problem. It seems like the key thing is to try to select the matching pairs first, and then you can try to refine the offset determination. If you match incorrectly, then it's possible for a spoiler pair to corrupt the offset refinement. If you had an a priori guess for the offset, that would help immensely.

As far as selecting pairs...

Compute the distances between all pairs of points. For your sample, that would be N1*N2 distance calculations. The result would be a list of N1*N2 distances. Presumably the "correct" offset would appear most frequently. One way to check this is make a histogram of the distances and look for a peak. Here, having an a priori guess is helpful to choose the bin size and histogram range. Otherwise you might have to try those iteratively.

Once you have a preferred offset distance, then you should be able to go through and match pairs reasonably efficiently. Also, you can enforce the constraint that a pair can only match once.

From that point, it should be easy to compute the mean offset, or if you have error bars, then the weighted mean offset. But you will probably have to check for outliers, which correspond with poorly matched pairs, and then iteratively recompute offset.

As I said, "fragile."