Subject: Re: checking for connectedness of a given set of pixels
Posted by James[2] on Mon, 08 Nov 2010 19:22:29 GMT
View Forum Message <> Reply to Message

LABEL_REGION's speed relies on a fast implementation of the union and find operations on disjoint set data structures (http://en.wikipedia.org/wiki/Disjoint-set_data_structure). I'm guessing that the IDL programmers have implemented this in C for LABEL_REGION, probably with a higher speed than you'd be able to get with anything written in IDL. If they did it "right" (with path compression) then it's an optimally fast algorithm for this problem (see Wikipedia article).

One problem is that LABEL_REGION runs over a whole array even if it is mostly empty (sparse). This is why LABEL_REGION might be slow for your task - it could be doing a lot of unnecessary extra work. Since your data is already in 2D coordinates, I'd take the max/min values and use a tightly-fitted array. Supposing your coordinates are in a 2-by-N array POINTS:

```
function ISCONNECTED(points, _extra=ex)
  mins = min(points, dimension=2, max=maxs)
  shift = mins # replicate(1, n_elements(points)/2)
  arr = bytarr(maxs-mins+1)
  arr[points - shift] = 1
  return, max(label_region(arr, _extra=ex)) le 1
end
```

the _extra keywords allow you to pass ALL_NEIGHBORS along to choose between 4- and 8-connectivity. Of course, this function adds a lot of extra baggage, so it's only going to be faster if your regions are significantly smaller than the whole array.