Subject: Re: checking for connectedness of a given set of pixels
Posted by David Fanning on Tue, 02 Nov 2010 20:37:55 GMT
View Forum Message <> Reply to Message

Guillermo writes:

> Hi there, I would need to create a function that tells me whether a
> set of pixels (given by their 2D coordinates) forms a single blob or
> consists of two or more disjoint groups, in either a 4- or 8-
> neighborhood. Can anyone suggest a computationally cheap method for
> doing that (i.e., one that doesn't rely on label_region or search_2d)?

Why are you hoping to re-write these two useful routines?

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: checking for connectedness of a given set of pixels
Posted by guillermo.castilla.ca on Tue, 02 Nov 2010 22:06:03 GMT
View Forum Message <> Reply to Message

On Nov 2, 5:37 pm, David Fanning <n...@dfanning.com> wrote:
> Why are you hoping to re-write these two useful routines?

Libreme Dios! Both functions are great, but I just need to know
whether or not the pixels are connected. That is, as soon as I find
one of the little guys who doesn't go hand in hand with the rest,
that's good enough for me. So I thought there should be a way of
getting this info without having to call the big shots ...

Cheers

Guillermo

Subject: Re: checking for connectedness of a given set of pixels
Posted by David Fanning on Tue, 02 Nov 2010 22:14:42 GMT
View Forum Message <> Reply to Message

Guillermo writes:

> Libreme Dios! Both functions are great, but I just need to know
> whether or not the pixels are connected. That is, as soon as I find
> one of the little guys who doesn't go hand in hand with the rest,
> that's good enough for me. So I thought there should be a way of
> getting this info without having to call the big shots ...

Bueno! You might try some variation of the chain-code algorithm
then. If you get to the end of a chain and there are no roads
leading anywhere but back the way you came, you are probably
at the end of the road.

This is not the easiest of algorithms to code, but you
can find one version (written for my purposes) in
find_boundary.pro. I think I found the algorithm in the
Russ image processing book.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

---

Subject: Re: checking for connectedness of a given set of pixels
Posted by guillermo.castilla.ca on Wed, 03 Nov 2010 13:32:22 GMT
View Forum Message <> Reply to Message

> You might try some variation of the chain-code algorithm then

I see what you mean David. So you pick from the input set the pixel
with lowest 1D index, which necessarily lies in the boundary of (one
of) the region(s), find the boundary of that region with
Find_Boundary, use that boundary as input to POLYFILLV as to get the
addresses of the pixels within it, and if there are some pixels in the
input set that do not belong to boundary+interior, then you know the
input set is not connected. However, if no pixel is outside but the
boundary+interior contains more pixels than the input set, then you
have to make additional checks to make sure that the holes are not
such that some group from the input set is disconnected from the rest.
At the end of the day, it might make more sense to stick to
Label_Region :( Thanks anyways for the advice.

Cheers

Guillermo

---

Subject: Re: checking for connectedness of a given set of pixels
Posted by David Fanning on Wed, 03 Nov 2010 13:35:49 GMT
View Forum Message <> Reply to Message

Guillermo writes:

> At the end of the day, it might make more sense to stick to
> Label_Region :(

That would be my conclusion, too. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

---

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.

---

## Subject: Re: checking for connectedness of a given set of pixels
## Posted by guillermo.castilla.ca on Sun, 14 Nov 2010 21:56:47 GMT

View Forum Message <> Reply to Message

On Nov 8, 3:22 pm, James <donje...@gmail.com> wrote:

>  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.

Thanks a lot James, very insightful comments, and very nice piece of
code :). After giving it some more thought, I found out that there is
an IDL function called REGI0N_GROW that may be faster than
LABEL_REGION for this purpose when the minimum bounding box for the
input set of pixels is large. I have modified your function to include
the former as an alternative method (below). I'll do some tests and
report back (hopefully within this year :).

```
function ISCONNECTED, points, method=method, _extra=ex
  mins = min(points, max=maxs, dimension=2)-1
  dims=maxs-mins+3
  npts= n_elements(points)/2
  indices= points - mins # replicate(1, npts)
  indices= indices[0,*] + dims[0]*indices[1,*]
  arr = bytarr(dims)
  arr[indices] = 1B
  if ~keyword_set(method) then                    $
  return, max(label_region(arr, _extra=ex)) le 1   $
  else return, n_elements(region_grow(             $
  arr, indices[0], thresh=[1B,1B], _extra=ex)) eq npts
end
```

Subject: Re: checking for connectedness of a given set of pixels
Posted by guillermo.castilla.ca on Mon, 03 Jan 2011 21:50:10 GMT
View Forum Message <> Reply to Message

On Nov 14 2010, 5:56 pm, Guillermo
<guillermo.castilla.castell...@gmail.com> wrote:

> ...REGI0N_GROW that may be faster than
>  LABEL_REGION for this purpose when the minimum bounding box for the
>  input set of pixels is large. I have modified your function to include
>  the former as an alternative method (below). I'll do some tests and
>  report back (hopefully within this year :).

Just for the record, today I made a test with a 50 megapixel labeled
image derived from the rasterization of a polygon vector layer
containing  185 elongated features (rivers), of which 65 ended up
consisting of disconnected groups of pixels. Checking for
connectedness of the pixels within each feature took almost 3 times
more using the REGI0N_GROW method than using LABEL_REGION (the latter
taking in average 30 millisecs per feature). I quickly looked into the
REGI0N_GROW code and noticed that it is based on LABEL_REGION itself,
so no wonder. Repeated the test with SEARCH2D instead of REGI0N_GROW
and although it was two times faster than the latter, it was still
slower than LABEL_REGION. So it looks like LABEL_REGION, even if it
does a lot of unnecessary extra work for this task, is the method of
choice for checking for connectedness of a set of pixels.

Happy new year!

Guillermo