Subject: Re: Area of a Blob
Posted by Karsten Rodenacker on Thu, 12 Dec 2002 09:11:01 GMT
View Forum Message <> Reply to Message

This reminds me on a thread
 http://groups.google.com/groups?hl=en&lr=&ie=UTF-8&a
mp;oe=UTF-8&threadm=3BA05C3A.78C09388%40Rodenacker.de&am
p;rnum=1&prev=/groups%3Fq%3Dgroup:comp.lang.idl-pvwave%2
BROI%2BRodenacker%26hl%3Den%26lr%3D%26ie%3DUTF-8%26oe%3DUTF-
8%26selm%3D3BA05C3A.78C09388%2540Rodenacker.de%26rnum%3D1
moving around just this topic!

Are you changing from pragmatics to progressives, David?

At least we have learned in the meantime that either the question is
ill-posed or some answers do not hit the question.

What is the area of a blob on an IMAGE?
   If IMAGE is considerd as a digital image the area can only be an
integer, the number of pixels of the blob region, multiplied by the area
of one pixel. -> simple count

   If the blob is defined by the contour coordinates located at the
CENTER of the pixels AND area is calculated based on integration formula
as done in POLY_AREA (Russ method), the result will be at least for
convex blobs smaller than the simple count. Area is defined by some sort
of rubberband (geodesy) around the pixel CENTERS.

   If the blob is defined by the coordinates at the outsides of the
pixels (more than one coordinate pair possible and necessary for one
pixel) the POLY_AREA joines simple count. However it is relatively
complicated to calculate these coordinates. Area is defined by some sort
of rubberband (geodesy) around the pixel EDGES.

The definition of coordinate locations of course tackles the problem of
defining the coordinate system of the data (image). The pixel (0,0) has
coordinates (0.5,0.5) or (0.0,0.0) or (0.0,1.0) or (1.0,0.0)?

   Area by simple count applied on POLYFILLV result, which returns
usually not the the generating blob pixel region, will again differ from
all.

   AND as already stated in the mentioned thread the object graphics
methods differ again.

My wish is that at least in IDL the methods would be consistent, e.g.

POLYFILLV and IDLgrROI Mask Method as well as POLY_AREA and IDLgrROI computeGeometry.

What to resume?

Discrete pixelized data input can only result in multiples of pixel areas.

Mixing data representations are dangerous. My remedy is as recommeded by David in the mentioned thread to stay in one methodical system, e.g.:

    coordinates from CONTOUR,...,/PATH_DATA_COOR,/CLOSED
    indices from POLYFILLV

It is necessary to define beforhand what is meant with data input (float coordinates, pixels) and area. Everything said is also true for perimeter.

There is not only one truth.

Thank you David to come up with this topic.

Best regards,
Karsten

David Fanning schrieb:
> Folks,
>
> Here is a question for you:
>
>   How much money did you make this year?
>
> Oh, wait, sorry. That has the same answer, but
> it's the wrong question. Here it is:
>
>   What is the area of a blob on an image?
>
> The answer, of course, is that it depends on who
> is asking.
>
> Ben Tupper and I were musing about this question this
> week, because it turns out you can get several answers,
> depending upon how you calculate it.
>
> Here are the results I got for a typical "blob" on
> an image I am analyzing:
>
> Area by .....
>   Simple Count: 7390.00

```
>                 Russ Method:  7236.50
>             PolyfillV Method:  7313.00
>       IDLgrROI computeGeometry: 7236.50
>          IDLgrROI Mask Method: 7391.00
>
```

> The Simple Count method just finds the unique indices in
> the ROI. The Russ method and the PolyFillV method involve
> calculating the chain code boundary of the ROI and using
> that to count the area of the pixels inside the boundary.
> The PolyFillV method misses most of the boundary pixels
> on the upper-right of the ROI. The Russ algorithm is this:
>

```
>    area = sum( (x(I) + x(i-1)) * (y(I) - y(i-1)) ) / 2.
```

>
> Where X and Y are the boundary points that close back on
> themselves. (We use my FIND_BOUNDARY program to find the
> boundary.)
>
> The Compute Geometry and ROI Mask method are used in
> IDL IDLgrROI object.
>
> What do you make of this? Does anyone have any insight?
> Does it matter how you computer area as long as you are
> consistent? Or is one method more accurate than others?
> What is the *real* answer?
>
> Appreciate your thoughts. :-)
>
> Cheers,
>
> David
>

--
Karsten Rodenacker (LapTop)
-- -------------:-)
GSF - Forschungszentrum Institute of Biomathematics and Biometry
D-85758 Oberschleissheim Postfach 11 29
Tel: +49 89 31873401 | FAX: ...3369 | rodena@gsf.de |
Karsten@Rodenacker.de
http://www.gsf.de/ibb/homepages/rodenacker

Subject: Re: Area of a Blob
Posted by Stein Vidar Hagfors H[2] on Thu, 12 Dec 2002 17:07:22 GMT
View Forum Message <> Reply to Message

David Fanning <david@dfanning.com> writes:

[...]
>   What is the area of a blob on an image?
>
> The answer, of course, is that it depends on who
> is asking.
[...]
```
>  Area by .....
>                 Simple Count:  7390.00
>                  Russ Method:  7236.50
>              PolyfillV Method:  7313.00
>       IDLgrROI computeGeometry: 7236.50
>           IDLgrROI Mask Method: 7391.00
```

Now, without knowing the details of the different methods, I'd say that it's not at all surprising to get different answers using different methods, because the question isn't well defined as stated, without a qualification of "what do you *mean* by the area of a blob (and how do you specify a blob, anyway)".

If you cound a blob as those pixels that are picked by a series of ROI indices, then just count the pixels. If you mean the geometrical area inside a polygon in units of square pixels, then you get a different answer (pixels may be bisected by the border polygon). If you mean "the number of pixels fully enclosed by the polygon" you get a third answer, if you mean "the number of pixels partially *or* fully enclosed by the polygon" you get a fourth, and so on (ROI methods may be returning one of the above).

> What do you make of this? Does anyone have any insight?
> Does it matter how you computer area as long as you are
> consistent? Or is one method more accurate than others?
> What is the *real* answer?
>
> Appreciate your thoughts. :-)

Not sure if the above helps you, but as long as you appreciate it, I'm happy ;-)

--
 -- --------------
Stein Vidar Hagfors Haugan
ESA SOHO SOC/European Space Agency Science Operations Coordinator for SOHO

NASA Goddard Space Flight Center,   Tel.: 1-301-286-9028
Mail Code 682.3, Bld. 26, Room G-1,   Cell: 1-240-354-6066
Greenbelt, Maryland 20771, USA.   Fax: 1-301-286-0264

--- --------------

Subject: Re: Area of a Blob
Posted by thompson on Thu, 12 Dec 2002 19:19:04 GMT
View Forum Message <> Reply to Message

Without knowing much about the details of the individual methods, I would guess that each uses a different interpolation technique to convert the discrete measurements into continuous space. Obviously, that would give you different answers. There are mathematical texts which discuss the assumptions and errors associated with different interpolation techniques. I seem to recall a discussion of this in Numerical Recipes, but I don't have a copy to hand to check.

William Thompson

David Fanning <david@dfanning.com> writes:

> Folks,

> Here is a question for you:

>   How much money did you make this year?

> Oh, wait, sorry. That has the same answer, but
> it's the wrong question. Here it is:

>   What is the area of a blob on an image?

> The answer, of course, is that it depends on who
> is asking.

> Ben Tupper and I were musing about this question this
> week, because it turns out you can get several answers,
> depending upon how you calculate it.

> Here are the results I got for a typical "blob" on
> an image I am analyzing:

```
> Area by .....
>               Simple Count:  7390.00
>               Russ Method:  7236.50
>           PolyfillV Method:  7313.00
>     IDLgrROI computeGeometry: 7236.50
>         IDLgrROI Mask Method: 7391.00
```

> The Simple Count method just finds the unique indices in
> the ROI. The Russ method and the PolyFillV method involve
> calculating the chain code boundary of the ROI and using
> that to count the area of the pixels inside the boundary.
> The PolyFillV method misses most of the boundary pixels
> on the upper-right of the ROI. The Russ algorithm is this:

>   area = sum((x(I) + x(i-1)) * (y(I) - y(i-1))) / 2.

> Where X and Y are the boundary points that close back on
> themselves. (We use my FIND_BOUNDARY program to find the
> boundary.)

> The Compute Geometry and ROI Mask method are used in
> IDL IDLgrROI object.

> What do you make of this? Does anyone have any insight?
> Does it matter how you computer area as long as you are
> consistent? Or is one method more accurate than others?
> What is the *real* answer?

> Appreciate your thoughts. :-)

> Cheers,

> David

> --
> David W. Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Phone: 970-221-0438, E-mail: david@dfanning.com
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Toll-Free IDL Book Orders: 1-888-461-0155