Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by David Fanning on Thu, 12 Oct 2006 02:40:24 GMT
View Forum Message <> Reply to Message

Gongqin Shen writes:

> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k
> = [1, 1], according to the help provided by IDL, the result of running
> the code:
> result = DILATE(a, k)
> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
> ERODE performs in a similar way. Does that mean the help is actually
> broken?

I think it means DILATE and ERODE are broken, but you
are sure to hear from Karsten Rodenacker about this. He
has been sending pleas to the folks at ITTVIS for years,
all without effect. Maybe the two of you could gang up
on them. :-)

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: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Karsten Rodenacker on Thu, 12 Oct 2006 07:41:29 GMT
View Forum Message <> Reply to Message

Don't use IDL's dilate and erode without embedding your data into a
sufficiently large array. Border handling is not coherently implemented.
That is a large disadvantage, not to say an error, for the application of
math. morph. operations in sequences. Ask for improvement, possibly ITTVIS
can be convinced!
Regards
Karsten

Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
<gqshen2008@gmail.com>:

> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k

> = [1, 1], according to the help provided by IDL, the result of running
> the code:
> result = DILATE(a, k)
> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
> ERODE performs in a similar way. Does that mean the help is actually
> broken?
>

--
Erstellt mit Operas revolutionï¿œrem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Haje Korth on Thu, 12 Oct 2006 12:22:26 GMT
View Forum Message <> Reply to Message

Karsten,
do you have suggestions for an alternative library (C, Fortran, IDL)?

Haje

Karsten Rodenacker wrote:
> Don't use IDL's dilate and erode without embedding your data into a
> sufficiently large array. Border handling is not coherently implemented.
> That is a large disadvantage, not to say an error, for the application of
> math. morph. operations in sequences. Ask for improvement, possibly ITTVIS
> can be convinced!
> Regards
> Karsten
>
> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
> <gqshen2008@gmail.com>:
>
>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k
>> = [1, 1], according to the help provided by IDL, the result of running
>> the code:
>> result = DILATE(a, k)
>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>> ERODE performs in a similar way. Does that mean the help is actually
>> broken?
>>
>
>
> --
> Erstellt mit Operas revolutionärem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Jo Klein on Thu, 12 Oct 2006 13:27:31 GMT
View Forum Message <> Reply to Message

The same applies to LABEL_REGION, which has caused me quite some
headache. IDL help doesn't even mention that restriction for
label_region, and it's not available in source code to figure out. Try:
a=shift(dist(20),10)
b=a lt 7
tvimage,bytscl(b),/noint
Then:
c=label_region(b)
tvimage,bytscl(c),/noint
The voxels on the border are mistakenly incorporated into the background
component, which is *really* dangerous when you use this to extract
features (e.g. a histogram each) from different objects in an image.
I would go as far as to say this is an error in IDL.
Cheers,
Jo

Karsten Rodenacker wrote:
> Don't use IDL's dilate and erode without embedding your data into a
> sufficiently large array. Border handling is not coherently
> implemented. That is a large disadvantage, not to say an error, for the
> application of math. morph. operations in sequences. Ask for
> improvement, possibly ITTVIS can be convinced!
> Regards
> Karsten
>
> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
> <gqshen2008@gmail.com>:
>
>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k
>> = [1, 1], according to the help provided by IDL, the result of running
>> the code:
>> result = DILATE(a, k)
>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>> ERODE performs in a similar way. Does that mean the help is actually
>> broken?
>>
>
>

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Karl Schultz on Thu, 12 Oct 2006 14:56:26 GMT

View Forum Message <> Reply to Message

On Thu, 12 Oct 2006 09:41:29 +0200, Karsten Rodenacker wrote:

> Don't use IDL's dilate and erode without embedding your data into a
> sufficiently large array. Border handling is not coherently implemented.
> That is a large disadvantage, not to say an error, for the application of
> math. morph. operations in sequences. Ask for improvement, possibly ITTVIS
> can be convinced!
> Regards
> Karsten
>
> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
> <gqshen2008@gmail.com>:
>
>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k
>> = [1, 1], according to the help provided by IDL, the result of running
>> the code:
>> result = DILATE(a, k)
>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>> ERODE performs in a similar way. Does that mean the help is actually
>> broken?
>>

All I can say is that we know about this problem and fixing it is "on the list". Karsten has already sent me some more detail. If anyone else would like to submit additional input, besides what is already in this thread, email it to kschultz at ittvis dot com. Thanks!

Karl

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by gqshen2008@gmail.com on Thu, 12 Oct 2006 15:32:12 GMT
View Forum Message <> Reply to Message

If possible, can I know when the fix will appear in IDL? Thx.

Karl Schultz wrote:
> On Thu, 12 Oct 2006 09:41:29 +0200, Karsten Rodenacker wrote:
>
>> Don't use IDL's dilate and erode without embedding your data into a
>> sufficiently large array. Border handling is not coherently implemented.
>> That is a large disadvantage, not to say an error, for the application of
>> math. morph. operations in sequences. Ask for improvement, possibly ITTVIS
>> can be convinced!
>> Regards

>> Karsten
>>
>> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
>> <gqshen2008@gmail.com>:
>>
>>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k
>>> = [1, 1], according to the help provided by IDL, the result of running
>>> the code:
>>> result = DILATE(a, k)
>>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>>> ERODE performs in a similar way. Does that mean the help is actually
>>> broken?
>>>
>
> All I can say is that we know about this problem and fixing it is "on the
> list". Karsten has already sent me some more detail. If anyone else
> would like to submit additional input, besides what is already in this
> thread, email it to kschultz at ittvis dot com. Thanks!
>
> Karl

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Karsten Rodenacker on Thu, 12 Oct 2006 17:06:37 GMT
View Forum Message <> Reply to Message

In fact I sent mail only to Karl. Possibly the things are of interest for
others, although there were not so many math. morph adepts:

Mathematical Morphology (MM) in IDL

I consider the actual state of the routines derived from math. morph. in
IDL as more or less useless. MM is in fact a methodology which could be
VERY helpful for image processing, provided the implementation does not it
break out, what is the case in IDL. My remark comprise the actual routines
ERODE
DILATE
MORPH_DISTANCE
THIN
LABEL_REGION
WATERSHED
as well as the derivatives
MORPH_CLOSE
MORPH_GRADIENT
MORPH_HITORMISS
MORPH_OPEN
MORPH_THIN

MORPH_TOPHAT
which suffer all from the unsufficient implementation.

Main point which should be improved is the processing of the array border points. I had some time ago already exchange about that but seemingly the rsi counterpart did nothing know about MM. However, I am interested in MM so I try it again!

ERODE/DILATE:
Basically a result of ERODE/DILATE is the logical AND/OR rspw. min/max of all pixels of the structuring element (st.e.). A border point is a point where the st.e. does contain points outside the data array. What should be done with these points? The worst is to say don't use these points, embed the data in a sufficient large data array. This is the way IDL uses. This approach is worse since the successive application of these operators is not possible or to slow. Additionally erosion and dilation are dual, with other words both should behave in the same way:
ERODE(x,st.e.) = NOT DILATE(NOT x, st.e)
What to do: There should be a switch, a system variable !MM.edge saying:
outside the data matrix everything is 1b/true (binary) or max(data) (gray) OR
outside the data matrix everything is 0b/false (binary) or min(data) (gray)
Now all border points have well defined values and duality of erosion and dilation can be preserved, no embedding is necessary, successive operations are possible and the whole MM apparatus can be implemented. See e.g. the book "Morphological Image Analysis", P. Soille, Springer

MORPH_DISTANCE
This routines suffers from the same problem. If an object is touching the border in the actual implementation border points are considered as object border points. Using the above mentioned !MM.edge morph_distance could deliver border independent results. and could be applied to the complement of the image without poblem, which is often necessary!
This routine is even usable for quick erosions and dilations by thresholding the distance map as I did earlier. The neighbor_sampling keyword selects the corresponding st.e..

THIN
is quick but in MM terms completely useless, since not connectivity preserving. It should be replaced by some skeleton routines similar to WATERSHED. This comprises a skeleton, an exoskeleton, a skeleton by zones of influence (SKIZ) etc. in fact the implementation of MORPH_THIN/THICK in a loop. For large images a mask parameter restricting on the mask pixels could be helpful (also for watershed).

LABEL_REGION
Although not directly MM related it is one of the jewels of IDL (like

where and histogram). The disturbing thing is (again) the border point behavior. It is deleting a one bit border. This is understandable at times where memory restrictions were severe but today?!?. Why not replacing the border points set by the newly calculated label values? Using LABEL_REGION e.g. for filling up objects (labeling the complement, deleting the background object, (res gt 0) OR with original) would become an easy task!

WATERSHED
First: why not implementing a 3d version of it. Only the choice of the neighborhood (as pointer array in the Vincent algorithm) should be changed, very easy and very helpful.
Second: WATERSHED is a labelling algorithm, it should correspond with LABEL_REGION! I am using watershed on binary images where the border behavior of LABEL_REGION is disturbing. Connectivity keyword corresponds to st.e.!

Am Thu, 12 Oct 2006 16:56:26 +0200 schrieb Karl Schultz <k_remove_schultz@ittvis.com>:

> On Thu, 12 Oct 2006 09:41:29 +0200, Karsten Rodenacker wrote:
>
>> Don't use IDL's dilate and erode without embedding your data into a
>> sufficiently large array. Border handling is not coherently implemented.
>> That is a large disadvantage, not to say an error, for the application
>> of
>> math. morph. operations in sequences. Ask for improvement, possibly
>> ITTVIS
>> can be convinced!
>> Regards
>> Karsten
>>
>> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
>> <gqshen2008@gmail.com>:
>>
>>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k
>>> = [1, 1], according to the help provided by IDL, the result of running
>>> the code:
>>> result = DILATE(a, k)
>>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>>> ERODE performs in a similar way. Does that mean the help is actually
>>> broken?
>>>
>
> All I can say is that we know about this problem and fixing it is "on the

> list". Karsten has already sent me some more detail. If anyone else
> would like to submit additional input, besides what is already in this
> thread, email it to kschultz at ittvis dot com. Thanks!
>
> Karl

--
Erstellt mit Operas revolutionï¿œrem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Karsten Rodenacker on Thu, 12 Oct 2006 17:22:27 GMT
View Forum Message <> Reply to Message

Not really, except the free implementations in Java for ImageJ. Good to
bridge by a Java freak into IDL! Look into the plugins of ImageJ.

I have written some things in C. But the state is not tranferrable.
I have lots of MM routines but not good enough documented. Usually I am
using 3x3 structuring elements to construct larger ones and for that I
have a border-consistet implementation of erosion and dilation in C (see
above), in fact a 3x3 hitormiss transformation.
I have a version of microMorph from Fontainebleau, an Windows program with
documentation. It is used to train the students there and for
'prototyping'. That is a very small interpreter but rather efficient in
math. morph. From the software included I learn. They have a certain poor
language for implementation of most of the more elevated routines.

Regards
Karsten

Am Thu, 12 Oct 2006 14:22:26 +0200 schrieb Haje Korth
<haje.korth@jhuapl.edu>:

> Karsten,
> do you have suggestions for an alternative library (C, Fortran, IDL)?
>
> Haje
>
> Karsten Rodenacker wrote:
>> Don't use IDL's dilate and erode without embedding your data into a
>> sufficiently large array. Border handling is not coherently implemented.
>> That is a large disadvantage, not to say an error, for the application
>> of
>> math. morph. operations in sequences. Ask for improvement, possibly

>> ITTVIS
>> can be convinced!
>> Regards
>> Karsten
>>
>> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
>> <gqshen2008@gmail.com>:
>>
>>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as
>> k
>>> = [1, 1], according to the help provided by IDL, the result of running
>>> the code:
>>> result = DILATE(a, k)
>>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>>> ERODE performs in a similar way. Does that mean the help is actually
>>> broken?
>>>
>>
>>
>> --
>> Erstellt mit Operas revolutionï¿œrem E-Mail-Modul:
>> http://www.opera.com/m2/
>

--
Erstellt mit Operas revolutionï¿œrem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by btt on Thu, 12 Oct 2006 18:41:15 GMT
View Forum Message <> Reply to Message

Karsten Rodenacker wrote:
> In fact I sent mail only to Karl. Possibly the things are of interest
> for others, although there were not so many math. morph adepts:
>
> Mathematical Morphology (MM) in IDL
>
> I consider the actual state of the routines derived from math. morph. in
> IDL as more or less useless. MM is in fact a methodology which could be
> VERY helpful for image processing, provided the implementation does not
> it break out, what is the case in IDL. My remark comprise the actual
> routines
> ERODE
> DILATE

> MORPH_DISTANCE
> THIN
> LABEL_REGION
> WATERSHED
> as well as the derivatives
> MORPH_CLOSE
> MORPH_GRADIENT
> MORPH_HITORMISS
> MORPH_OPEN
> MORPH_THIN
> MORPH_TOPHAT
> which suffer all from the unsufficient implementation.
>
> Main point which should be improved is the processing of the array
> border points. I had some time ago already exchange about that but
> seemingly the rsi counterpart did nothing know about MM. However, I am
> interested in MM so I try it again!
>
> ERODE/DILATE:
> Basically a result of ERODE/DILATE is the logical AND/OR rspw. min/max
> of all pixels of the structuring element (st.e.). A border point is a
> point where the st.e. does contain points outside the data array. What
> should be done with these points? The worst is to say don't use these
> points, embed the data in a sufficient large data array. This is the way
> IDL uses. This approach is worse since the successive application of
> these operators is not possible or to slow. Additionally erosion and
> dilation are dual, with other words both should behave in the same way:
> ERODE(x,st.e.) = NOT DILATE(NOT x, st.e)
> What to do: There should be a switch, a system variable !MM.edge saying:
> outside the data matrix everything is 1b/true (binary) or max(data)
> (gray) OR
> outside the data matrix everything is 0b/false (binary) or min(data)
> (gray)

Hi Karsten,

I should mention that LabView's image processing does something like the
!MM.edge you describe with added variations including:

- user specification of the width of the pseudo-edge pixels (the padding)
- repeating the values of the edge pixels into the psueudo-edge
- wrapping the values of the edge pixels into the pseudo-edge
- extrapolating the values of the edge-pixels into the pseudo-egde
- user defined gray value for the pseudo-edge pixels.
- and some other things I can't recall just now.

Ben

> Now all border points have well defined values and duality of erosion
> and dilation can be preserved, no embedding is necessary, successive
> operations are possible and the whole MM apparatus can be implemented.
> See e.g. the book "Morphological Image Analysis", P. Soille, Springer
>
> MORPH_DISTANCE
> This routines suffers from the same problem. If an object is touching
> the border in the actual implementation border points are considered as
> object border points. Using the above mentioned !MM.edge morph_distance
> could deliver border independent results. and could be applied to the
> complement of the image without poblem, which is often necessary!
> This routine is even usable for quick erosions and dilations by
> thresholding the distance map as I did earlier. The neighbor_sampling
> keyword selects the corresponding st.e..
>
> THIN
> is quick but in MM terms completely useless, since not connectivity
> preserving. It should be replaced by some skeleton routines similar to
> WATERSHED. This comprises a skeleton, an exoskeleton, a skeleton by
> zones of influence (SKIZ) etc. in fact the implementation of
> MORPH_THIN/THICK in a loop. For large images a mask parameter
> restricting on the mask pixels could be helpful (also for watershed).
>
> LABEL_REGION
> Although not directly MM related it is one of the jewels of IDL (like
> where and histogram). The disturbing thing is (again) the border point
> behavior. It is deleting a one bit border. This is understandable at
> times where memory restrictions were severe but today?!?. Why not
> replacing the border points set by the newly calculated label values?
> Using LABEL_REGION e.g. for filling up objects (labeling the complement,
> deleting the background object, (res gt 0) OR with original) would
> become an easy task!
>
> WATERSHED
> First: why not implementing a 3d version of it. Only the choice of the
> neighborhood (as pointer array in the Vincent algorithm) should be
> changed, very easy and very helpful.
> Second: WATERSHED is a labelling algorithm, it should correspond with
> LABEL_REGION! I am using watershed on binary images where the border
> behavior of LABEL_REGION is disturbing. Connectivity keyword corresponds
> to st.e.!
>
>
>
>

>
> Am Thu, 12 Oct 2006 16:56:26 +0200 schrieb Karl Schultz
> <k_remove_schultz@ittvis.com>:
>
>> On Thu, 12 Oct 2006 09:41:29 +0200, Karsten Rodenacker wrote:
>>
>>> Don't use IDL's dilate and erode without embedding your data into a
>>> sufficiently large array. Border handling is not coherently implemented.
>>> That is a large disadvantage, not to say an error, for the
>>> application of
>>> math. morph. operations in sequences. Ask for improvement, possibly
>>> ITTVIS
>>> can be convinced!
>>> Regards
>>> Karsten
>>>
>>> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
>>> <gqshen2008@gmail.com>:
>>>
>>>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as k
>>>> = [1, 1], according to the help provided by IDL, the result of running
>>>> the code:
>>>> result = DILATE(a, k)
>>>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>>>> ERODE performs in a similar way. Does that mean the help is actually
>>>> broken?
>>>>
>>
>> All I can say is that we know about this problem and fixing it is "on the
>> list". Karsten has already sent me some more detail. If anyone else
>> would like to submit additional input, besides what is already in this
>> thread, email it to kschultz at ittvis dot com. Thanks!
>>
>> Karl
>
>
>
> --Erstellt mit Operas revolutionï¿œrem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Jo Klein on Fri, 13 Oct 2006 08:46:31 GMT
View Forum Message <> Reply to Message

> LABEL_REGION
> Although not directly MM related it is one of the jewels of IDL (like
> where and histogram). The disturbing thing is (again) the border point

> behavior. It is deleting a one bit border.
Hi Karsten,
At the risk of repeating myself - it's even worse than that. It doesn't
delete the border, but puts it into the first component. What if you're
actually interested in that one, say, to calculate background features?
It will be contaminated with portions of the other components - not nice.
Cheers,
Jo

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Haje Korth on Fri, 13 Oct 2006 11:53:10 GMT
View Forum Message <> Reply to Message

Karsten,
thanks for the info. At this point I am not desperate about a solution,
but I like to keep my eyes open. Fortunately for you ITTVIS is drifting
more and more toward image processing. So unlike my "I want nice fonts
in direct graphics" issue you have a good chance of getting that fixed
eventually. :-)

Cheers,
Haje

Karsten Rodenacker wrote:
> Not really, except the free implementations in Java for ImageJ. Good to
> bridge by a Java freak into IDL! Look into the plugins of ImageJ.
>
> I have written some things in C. But the state is not tranferrable.
> I have lots of MM routines but not good enough documented. Usually I am
> using 3x3 structuring elements to construct larger ones and for that I
> have a border-consistet implementation of erosion and dilation in C (see
> above), in fact a 3x3 hitormiss transformation.
> I have a version of microMorph from Fontainebleau, an Windows program with
> documentation. It is used to train the students there and for
> 'prototyping'. That is a very small interpreter but rather efficient in
> math. morph. From the software included I learn. They have a certain poor
> language for implementation of most of the more elevated routines.
>
> Regards
> Karsten
>
> Am Thu, 12 Oct 2006 14:22:26 +0200 schrieb Haje Korth
> <haje.korth@jhuapl.edu>:
>

>> Karsten,
>> do you have suggestions for an alternative library (C, Fortran, IDL)?
>>
>> Haje
>>
>> Karsten Rodenacker wrote:
>>> Don't use IDL's dilate and erode without embedding your data into a
>>> sufficiently large array. Border handling is not coherently implemented.
>>> That is a large disadvantage, not to say an error, for the application
>>> of
>>> math. morph. operations in sequences. Ask for improvement, possibly
>>> ITTVIS
>>> can be convinced!
>>> Regards
>>> Karsten
>>>
>>> Am Thu, 12 Oct 2006 04:33:59 +0200 schrieb Gongqin Shen
>>> <gqshen2008@gmail.com>:
>>>
>>>> For example, if you have the data as a = [0, 1, 1, 0] and kernel as
>>> k
>>>> = [1, 1], according to the help provided by IDL, the result of running
>>>> the code:
>>>> result = DILATE(a, k)
>>>> will be [0, 1, 1, 0], however, IDL's output is [1, 1, 1, 0].
>>>> ERODE performs in a similar way. Does that mean the help is actually
>>>> broken?
>>>>
>>>
>>>
>>> --
>>> Erstellt mit Operas revolutionärem E-Mail-Modul:
>>> http://www.opera.com/m2/
>>
>
>
>
> --
> Erstellt mit Operas revolutionärem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Karsten Rodenacker on Fri, 13 Oct 2006 15:05:59 GMT
View Forum Message <> Reply to Message

Jo, I meant with deleting border points 'set to integer 0' which is infact
the background.

To define connected components there has background to be defined in
advance,
hence the background is NOT in the result a labelled component, even not
necessarily connected!
Connected compenents start with index 1! This differs with the results of
contour path where each connected border gets an entry in the info list
with mark inside or outside.

However, the deletion of the border in LABEL_REGION is nasty and
superfluos today!
Regards
KR

Am Fri, 13 Oct 2006 10:46:31 +0200 schrieb Jo Klein <jo_kln@yahoo.co.uk>:

>> LABEL_REGION
>> Although not directly MM related it is one of the jewels of IDL (like
>> where and histogram). The disturbing thing is (again) the border point
>> behavior. It is deleting a one bit border.
> Hi Karsten,
> At the risk of repeating myself - it's even worse than that. It doesn't
> delete the border, but puts it into the first component. What if you're
> actually interested in that one, say, to calculate background features?
> It will be contaminated with portions of the other components - not nice.
> Cheers,
> Jo

--
Erstellt mit Operas revolutionï¿œrem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by Jean H. on Fri, 13 Oct 2006 16:37:46 GMT
View Forum Message <> Reply to Message

Could one simply add a border around its image, process the
"label_region" and delete the border?

Jean

Karsten Rodenacker wrote:
> Jo, I meant with deleting border points 'set to integer 0' which is
> infact the background.
> To define connected components there has background to be defined in
> advance,

> hence the background is NOT in the result a labelled component, even
> not necessarily connected!
> Connected compenents start with index 1! This differs with the results
> of contour path where each connected border gets an entry in the info
> list with mark inside or outside.
>
> However, the deletion of the border in LABEL_REGION is nasty and
> superfluos today!
> Regards
> KR
>
> Am Fri, 13 Oct 2006 10:46:31 +0200 schrieb Jo Klein <jo_kln@yahoo.co.uk>:
>
>>> LABEL_REGION
>>> Although not directly MM related it is one of the jewels of IDL
>>> (like where and histogram). The disturbing thing is (again) the
>>> border point behavior. It is deleting a one bit border.
>>
>> Hi Karsten,
>> At the risk of repeating myself - it's even worse than that. It
>> doesn't delete the border, but puts it into the first component. What
>> if you're actually interested in that one, say, to calculate
>> background features? It will be contaminated with portions of the
>> other components - not nice.
>> Cheers,
>> Jo
>
>
>
>

Subject: Re: IDL's built-in function DILATE and ERODE doesn't work as described in help
Posted by gqshen2008@gmail.com on Mon, 16 Oct 2006 20:21:28 GMT
View Forum Message <> Reply to Message

Technically speaking, we can do that, but why we use IDL at all if we
have to put extra effort like that? I am sure many people here can
write most of IDL's routines by themselves. What IDL should do is
actually lessen the programming burden, not increase that.

Jean H. wrote:
> Could one simply add a border around its image, process the
> "label_region" and delete the border?
>
> Jean
>

> Karsten Rodenacker wrote:
>> Jo, I meant with deleting border points 'set to integer 0' which is
>> infact the background.
>> To define connected components there has background to be defined in
>> advance,
>> hence the background is NOT in the result a labelled component, even
>> not necessarily connected!
>> Connected compenents start with index 1! This differs with the results
>> of contour path where each connected border gets an entry in the info
>> list with mark inside or outside.
>>
>> However, the deletion of the border in LABEL_REGION is nasty and
>> superfluos today!
>> Regards
>> KR
>>
>> Am Fri, 13 Oct 2006 10:46:31 +0200 schrieb Jo Klein <jo_kln@yahoo.co.uk>:
>>
>>>> LABEL_REGION
>>>> Although not directly MM related it is one of the jewels of IDL
>>>> (like where and histogram). The disturbing thing is (again) the
>>>> border point behavior. It is deleting a one bit border.
>>>
>>> Hi Karsten,
>>> At the risk of repeating myself - it's even worse than that. It
>>> doesn't delete the border, but puts it into the first component. What
>>> if you're actually interested in that one, say, to calculate
>>> background features? It will be contaminated with portions of the
>>> other components - not nice.
>>> Cheers,
>>> Jo
>>
>>
>>
>>