Subject: Re: Need Some Good Ideas
Posted by Craig Markwardt on Thu, 21 Feb 2002 07:02:37 GMT
View Forum Message <> Reply to Message

Hey David--

Interesting problem! The first thing that comes to mind is to try to smooth the data somewhat. For example, by fitting a spline through it (SPL_INIT), and then interpolating the spline onto a much finer grid (SPL_INTERP).

Now, that doesn't totally solve your problem, because the interpolants will still be irregularly spaced, but they will be much more finely sampled. You could then sum them to get an arc-length over each segment, and then resample after adjusting for the arc length.

Do you have some example data?

Craig

David Fanning <david@dfanning.com> writes:
> Folks,
>
> Do you have your thinking caps on? I'm looking for
> a few good ideas.
>
> I have a bunch of blobs. (Think spots on the
> Gateway cow.) I would like to analyze the curvature
> and bends in the perimeter of the blobs. I have
> the indices of the points that make up the blob, and
> I have obtained the "perimeter" points by contouring
> the blob. Unfortunately, these perimeter points are
> not evenly distributed. (Think of a blob that has a
> long, straight side. The contour command will put a
> point at either end of the straight bit, so the points
> on that side of the blob are sparse, while the points
> along a tight bend on the other side of the blob
> are dense.)
>
> I say "unfortunately" because we have a method that
> uses the derivative of the perimeter at each point
> and the FFT transform of the derivative distribution,
> but it seems to be giving funny results because of this
> point distribution problem.
>
> Has anyone heard of this kind of curvature analysis
> before? Any pointers to literature? I've heard that

> IDL can be used to solve these kinds of problems. :-)
>
> Thanks,
>
> David
> --
> David W. Fanning, Ph.D.
> Fanning Software Consulting
> 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

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: Need Some Good Ideas
Posted by Craig Markwardt on Thu, 21 Feb 2002 07:46:35 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@cow.physics.wisc.edu> writes:

> Hey David--
>
> Interesting problem! The first thing that comes to mind is to try to
> smooth the data somewhat. For example, by fitting a spline through it
> (SPL_INIT), and then interpolating the spline onto a much finer grid
> (SPL_INTERP).

I caught the bug. Here is my submission, in the form of a procedure called BLOBTERP which follows. You pass it your array of X and Y contour points, and the number of output points you want. It produces a set of contour points which are smoothly interpolated and regularly sampled along the length of the arc.

Here is a test data set and the way that BLOBTERP is called.

```
x = randomn(seed, 20) & y = randomn(seed, 20)  ;; Random points
th = atan(y,x)        & ii = sort(th)          ;; Sort in a circle
x = x(ii)  & x = [x, x(0)]   ;; Complete the arc
y = y(ii)  & y = [y, y(0)]

blobterp, x, y, 50, xx, yy
plot, xx, yy
```

The only thing that might not be "perfect" is the seam where the outline joins itself, where there will be a derivative discontinuity. Unfortunately the IDL spline routine doesn't do cyclic constraints, so it would be hard to perfect this. You might consider dropping points near (x(0),y(0)).

Time for resting,
Craig

```
;; X & Y = outline of blob
;; N = number of desired interpolants
;; XX & YY = regularly sampled interpolants
pro blobterp, x, y, n, xx, yy

  npt = n_elements(x)
  nc = npt*100
  t = dindgen(npt)

  ;; Interpolate very finely
  t1 = dindgen(nc+1)/100
  x1 = spl_interp(t, x, spl_init(t, x), t1)
  y1 = spl_interp(t, y, spl_init(t, y), t1)

  ;; Compute cumulative path length
  ds = sqrt((x1(1:*)-x1)^2 + (y1(1:*)-y1)^2)
  ss = [0d, total(ds, /cum)]

  ;; Invert this curve, solve for TX, which should be evenly sampled in
  ;; the arc length space
  sx = dindgen(n+1)*max(ss)/n
  tx = spl_interp(ss, t1, spl_init(ss, t1), sx)

  ;; Reinterpolate the original points using the new values of TX
  xx = spl_interp(t, x, spl_init(t, x), tx)
  yy = spl_interp(t, y, spl_init(t, y), tx)
  return

end
```

--

 -- --------------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 -- --------------

Subject: Re: Need Some Good Ideas
Posted by ronn on Thu, 21 Feb 2002 12:46:17 GMT
View Forum Message <> Reply to Message

Hi David,

Craig's method is much more elegant than this one, but I find it useful.

After you draw the contours around your blobs then create a blank pixmap of the same size as the image. (I am guessing that you are using device coordinates) Then draw the contour to the pixmap in a single color like 255. Do an im= TVRD() to get a new image and then do an indices = where(im eq 255). Turn these back to x,y coords and you have a very nearly evenly spaced series of points.

-Ronn

--
Ronn Kling
KRS, inc.
email: ronn@rlkling.com
"Application Development with IDL"ï¿½ programming book updated for IDL5.5!
"Calling C from IDL, Using DLM's to extend your IDL code"!
http://www.rlkling.com/

Subject: Re: Need Some Good Ideas
Posted by dw on Thu, 21 Feb 2002 15:22:58 GMT
View Forum Message <> Reply to Message

Hi David,

This is almost exactly the same problem I'm working on (except not very skillfully and only part-time....). I found some help in a couple of articles on imaging of snow particles:
Computation of 3D curvatures on a wet snow sample, by Brzoska et al., Eur. Physical Journal, Applied Physics, 7, 45-57, 1999
Three-dimensional snow images by x-ray microtomography by Coleou et al., Annals of Glaciology, 32, 2001, 75-81.

The main idea of their model is to use the medial axes (or alternatively chamfer distances) of the object to locate the normal vectors of the points on the curve and then compute the local curvature in a point as C=ï¿½[(1/R1)+(1/R2)] where R1 and R2 are the radii of curvature of the two planes orthogonal to each other and containing the normal vector to the surface.

So now it just has to coded....
Don't know if this is of help to you? Are you working only in 2D?
Cheers,
Dorthe

David Fanning <david@dfanning.com> wrote in message
news:<MPG.16de0d88b1cdeb82989813@news.frii.com>...
> Folks,
>
> Do you have your thinking caps on? I'm looking for
> a few good ideas.
>
> I have a bunch of blobs. (Think spots on the
> Gateway cow.) I would like to analyze the curvature
> and bends in the perimeter of the blobs. I have
> the indices of the points that make up the blob, and
> I have obtained the "perimeter" points by contouring
> the blob. Unfortunately, these perimeter points are
> not evenly distributed. (Think of a blob that has a
> long, straight side. The contour command will put a
> point at either end of the straight bit, so the points
> on that side of the blob are sparse, while the points
> along a tight bend on the other side of the blob
> are dense.)
>
> I say "unfortunately" because we have a method that
> uses the derivative of the perimeter at each point
> and the FFT transform of the derivative distribution,
> but it seems to be giving funny results because of this
> point distribution problem.
>
> Has anyone heard of this kind of curvature analysis
> before? Any pointers to literature? I've heard that
> IDL can be used to solve these kinds of problems. :-)
>
> Thanks,
>
> David

Subject: Re: Need Some Good Ideas
Posted by David Fanning on Thu, 21 Feb 2002 15:50:03 GMT
View Forum Message <> Reply to Message

Craig Markwardt (craigmnet@cow.physics.wisc.edu) writes:

> I caught the bug. Here is my submission, in the form of a procedure
> called BLOBTERP which follows.

Very nice, we are going to try using this today
and see if it works. :-)

Thanks to everyone (both here and via private e-mail)
for their ideas. We have a number of new leads this morning
to track down. I'll let you know what we come up with.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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

Subject: Re: Need Some Good Ideas
Posted by David Fanning on Thu, 21 Feb 2002 16:09:48 GMT
View Forum Message <> Reply to Message

Ronn Kling (ronn@rlkling.com) writes:

> After you draw the contours around your blobs then create a blank pixmap of
> the same size as the image. (I am guessing that you are using device
> coordinates) Then draw the contour to the pixmap in a single color like 255.
> Do an im= TVRD() to get a new image and then do an indices = where(im eq
> 255). Turn these back to x,y coords and you have a very nearly evenly
> spaced series of points.

I've used this method before, and it might actually
work in this instance too, although it seems such
a hack I'm always embarrassed to have it in my
"professional" code. :-)

One of the problems with it, incidentally, is
that the points returned by the WHERE function
are not in the order you expect them to be in.
That is to say, the WHERE function doesn't find
one end of the line and proceed to the other end.
It can jump around, depending (I suppose) on how
the line is oriented on the image. We learned this

to our dismay when we were using this method to
calculate the length of line segments. We had
to add a routine to sort the WHERE indices into
a real line!

Cheers,

David
--
David W. Fanning, Ph.D.
Fanning Software Consulting
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

Subject: Re: Need Some Good Ideas
Posted by btupper on Thu, 21 Feb 2002 18:43:21 GMT
View Forum Message <> Reply to Message

> I have a bunch of blobs. (Think spots on the
> Gateway cow.) I would like to analyze the curvature
> and bends in the perimeter of the blobs. I have
> the indices of the points that make up the blob, and
> I have obtained the "perimeter" points by contouring
> the blob. Unfortunately, these perimeter points are
> not evenly distributed. (Think of a blob that has a
> long, straight side. The contour command will put a
> point at either end of the straight bit, so the points
> on that side of the blob are sparse, while the points
> along a tight bend on the other side of the blob
> are dense.)

Hi,

I have stumbled over this ground before. I use the FillIn_Contour
function pasted in below. It's pretty typical of my brute force
technique for doing geometery with IDL.

While we are on the subject of contouring, I recall that a call to
CONTOUR with the Path_XY keyword set to retrieve the vertex pairs will
change the direct graphics plot scaling. A call to CONTOUR with the
OverPlot keyword set does not change the plot scaling. I *always*
forget that Contour does this when someone signing my paycheck is
looking over my shoulder. I show an example of this in the
FunnyContour procedure below.

Ben

```
;*******START EXAMPLE HERE

PRO FunnyContour

TVLCT, R, G, B, /get

Img = BytArr(100,100)
Img[25:75,25:75] = 1
Img = Rot(Img, 30)
Img[0:50, *] = 0

LoadCT, 0, bottom = 32
Tek_Color
ImDisp, Img, /no_Scale, /Axis, /Erase
Print, 'XS after plot ',  !X.S

 ;save these for later
XS = !X.S
YS = !Y.S


;contour directly to the display - shown in red
Contour, Img, levels = [1], c_color = 2, /over
Print, 'XS after contour to display ',  !X.S

;contour directly to the output keywords show in green
Contour, Img, levels = [1], Path_XY = xy, /Path_Data_Coord
Print,   'XS after contour to path_xy ',  !X.S
oPlot, XY[0,*], XY[1,*], color = 3, psym = 1

;restore the plotting parameters
;and show the results in blue
!X.S = XS
!Y.S = YS
oPlot, XY[0,*], XY[1,*], color = 4, psym = 2


 ;pad in the missing pixel locations
 ;show these in orange
newXY = FillIn_Contour(XY)
oPlot, newXY[0,*], newXY[1,*], color = 8, psym = 5

TVLCT, R, G, B
```

```
END

;*******END EXAMPLE HERE


;**********ANOTHER START HERE
;+
; NAME:
; FillIn_CONTOUR
;
; PURPOSE:
; Given the XY data path coordinates returned
; from a call to CONTOUR using
; an image, this routine returns ALL of the pixels
; coordinates that lie along the contour.
;
; CATEGORY:
; Image analysis.
;
; ARGUMENTS:
; XY  A 2xn element array of contour vertices returned
;  by the CONTOUR routine keyword PATH_XY.
; DELTA  A one or two element vector of the 'pixel'
; size in the x and y directions. By default [1,1] is used.
;
; RETURNED:
; A 2xm element array of the coordinates of all
; the pixels/cells that lie along the path.
; The first point and last are the not the same.
;
; EXAMPLE:
; ;make an image
; img = BytArr(250,250)
; Img[50:200, 50:200] = 1
; Tek_color
; TV, Image
;  CONTOUR, img, Path_XY = XY,
;  /Path_Data_Coord, Levels = [1]
;  ;only 4 vertices (since the 'island' is a rectangle
; plots, XY[0,*], XY[1,*], psym = 6, color = 2, /dev
;  ;get all the pixels
;  newxy = filin_contour(xy)
;  ;show the new points
; plots,newXY[0,*], newXY[1,*], psym = 3, /dev, color = 3
;
; MODIFICATION HISTORY
; 18 JUNE 2001 Written by Ben Tupper.
;-
```

```
FUNCTION FillIn_Contour, XY, delta

 ;establish the step size
Case n_elements(Delta) of
 0: d = [1.0,1.0]
 1: d = [float(delta),delta]
 Else: d = Float(Delta[0:1])
EndCase

d = ABS(d)
neg_d = 0.0 - d

;convert original array into 'wrapped' vectors
oX = [Reform(XY[0,*]), XY[0,0]]
oY = [Reform(XY[1,*]), XY[1,0]]

 ;seed the new xy paring
X = Ptr_new(oX[0])
Y = Ptr_New(oY[0])

 ;step through the original verticies, padding the
 ;'new' vertices as needed

i = 1L & newCount = 1L
While i lt n_elements(oX) do begin

  ;get the x and y displacements
 dx = oX[i] - (*X)[newCount-1]
 dy = oY[i] - (*Y)[newCount-1]

 Case 1 of

   ;in this case the next vertex in origianl set
is more than
   ;one pixel away
  ABS(dx) GT d[0] OR ABS(dy) GT d[1]: Begin

  *X = [*X,(*X)[newCount-1] + (neg_d[0] > dx < d[0])]
  *Y = [*Y,(*Y)[newCount-1] + (neg_d[1] > dy < d[1])]
    ;the newCount will be incremented, but
not the
    ;counter for the old set
   newCount = NewCount+1

  End ;big gap to be filled in

  ;in this case we have arrived at a vertex... so simply
```

```
  ;move on to the next vertex in the original data set
 dx EQ 0 AND dy EQ 0: i = i+1

  ;in this case the next vertex is the one form the
original data set
 Else:  Begin

  *X = [*X, oX[i]]
  *Y = [*Y, oY[i]]
   ;move on to the next vertex
  i = i + 1
  NewCount = NewCount + 1
  End

 EndCase
EndWhile


NewXY = $
 transpose( [ [ (*X)[0: newCount-2] ],[ (*Y)[0: newCount-2] ]
])
ptr_free, X,Y

Return, NewXY
END
;*********ANOTHER END HERE
```

Subject: Re: Need Some Good Ideas
Posted by David Fanning on Thu, 21 Feb 2002 19:22:35 GMT
View Forum Message <> Reply to Message

Ben Tupper (btupper@bigelow.org) writes:

> While we are on the subject of contouring, I recall that a call to
> CONTOUR with the Path_XY keyword set to retrieve the vertex pairs will
> change the direct graphics plot scaling. A call to CONTOUR with the
> OverPlot keyword set does not change the plot scaling.

Isn't that because the Path_XY values are
always in Normalized coordinate space? This
seems reasonable to me, especially after I've
been bit by it 8 or 9 times. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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

Subject: Re: Need Some Good Ideas
Posted by Martin Downing on Thu, 21 Feb 2002 22:35:30 GMT
View Forum Message <> Reply to Message

Hi David,

refering to your gateway blobs:
if thats what you have then they are easy to segment, you then identify each individual blob using region labeling

now for each blob:
1. identify the boundary (this should be as a [x,y] array of every boundary pixel (no long sides)- as either an internal or external boundary
2. transform the array to [s,theta], where s[i] is the cumulative distance along the boundary to P[i] and theta[i] is the angle of the link of P[i-1] from P[i] relative to (say) the X axis
3. [s,theta] will be periodic,(i.e. you could carry on going round and round), so you can now run a FFT or calculate the first fourier descriptors of it - see reference

good luck - I spend rather too much time playing with this kind of stuff!

Martin

Reference List
1. Lin, Chellappa R. Classification Of Partial 2-D Shapes Using Fourier Descriptors. IEEE Transactions On Pattern Analysis And Machine Intelligence 1987; 9:686-690

--

Martin Downing,
Clinical Research Physicist,
Grampian Orthopaedic RSA Research Centre,
Woodend Hospital, Aberdeen, AB15 6LS.
m.downingATabdn.ac.uk

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.16de0d88b1cdeb82989813@news.frii.com...

> Folks,
>
> Do you have your thinking caps on? I'm looking for
> a few good ideas.
>
> I have a bunch of blobs. (Think spots on the
> Gateway cow.) I would like to analyze the curvature
> and bends in the perimeter of the blobs. I have
> the indices of the points that make up the blob, and
> I have obtained the "perimeter" points by contouring
> the blob. Unfortunately, these perimeter points are
> not evenly distributed. (Think of a blob that has a
> long, straight side. The contour command will put a
> point at either end of the straight bit, so the points
> on that side of the blob are sparse, while the points
> along a tight bend on the other side of the blob
> are dense.)
>
> I say "unfortunately" because we have a method that
> uses the derivative of the perimeter at each point
> and the FFT transform of the derivative distribution,
> but it seems to be giving funny results because of this
> point distribution problem.
>
> Has anyone heard of this kind of curvature analysis
> before? Any pointers to literature? I've heard that
> IDL can be used to solve these kinds of problems. :-)
>
> Thanks,
>
> David
> --
> David W. Fanning, Ph.D.
> Fanning Software Consulting
> 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

Subject: Re: Need Some Good Ideas
Posted by btupper on Thu, 21 Feb 2002 22:35:46 GMT
View Forum Message <> Reply to Message

On Thu, 21 Feb 2002 12:22:35 -0700, David Fanning <david@dfanning.com>
wrote:

>

> Isn't that because the Path_XY values are
> always in Normalized coordinate space? This
> seems reasonable to me, especially after I've
> been bit by it 8 or 9 times. :-)
>

Hi,

Not if the PATH_DATA_COORD keyword is set.

Ben

Subject: Re: Need Some Good Ideas
Posted by Stein Vidar Hagfors H[1] on Fri, 22 Feb 2002 17:05:22 GMT
View Forum Message <> Reply to Message

dw@er.dtu.dk (Dorthe Wildenschild) writes:

> Hi David,
>
[references to 3D problem by Dorthe Wildenschild omitted]

> David Fanning <david@dfanning.com> wrote in message
news:<MPG.16de0d88b1cdeb82989813@news.frii.com>...
>> Folks,
>>
>> Do you have your thinking caps on? I'm looking for
>> a few good ideas.
>>
>> I have a bunch of blobs. (Think spots on the
>> Gateway cow.) I would like to analyze the curvature
>> and bends in the perimeter of the blobs. I have
>> the indices of the points that make up the blob, and
>> I have obtained the "perimeter" points by contouring
>> the blob. Unfortunately, these perimeter points are
>> not evenly distributed. (Think of a blob that has a
>> long, straight side. The contour command will put a
>> point at either end of the straight bit, so the points
>> on that side of the blob are sparse, while the points
>> along a tight bend on the other side of the blob
>> are dense.)
>>
>> I say "unfortunately" because we have a method that
>> uses the derivative of the perimeter at each point
>> and the FFT transform of the derivative distribution,
>> but it seems to be giving funny results because of this
>> point distribution problem.

>>
>> Has anyone heard of this kind of curvature analysis
>> before? Any pointers to literature? I've heard that
>> IDL can be used to solve these kinds of problems. :-)

I haven't *really* heard of that kind of curvature analysis before, but with my thinking cap on, I think I can guess!

The thing is that the power spectrum of a function can tell you something about the degree of continuity of the function. Oh yes, I have a literature pointer: The Fourier Transform and Its Application (R. N. Bracewell, second edition) - a must for anyone using Fourier transform, IMHO.

However, the short (and only handwavingly accurate) story is: The power spectrum of a continuous function goes to zero for high frequencies (i.e., "no" infinitely high frequency components necessary to describe discontinuities in the function). (Note - noise, even just numerical, will often introduce a nonzero flat level, becase measured values will jump slightly in a discontinuous fashion from one point to the next)

When you take the derivative of a function, it corresponds to multiplying it's Fourier transform with something proportional (in magnitude) to the frequency (f) - thus multiplying the power spectrum with something propto f^2. So, if a function's *derivative* is continuous, the power spectrum of the function drops *faster* than 1/f^2. The nth derivative is continuous if the power spectrum drops *faster* than 1/f^(2n).

The function in your case is the "derivative of the perimeter at each point" - whatever that means! I presume you mean something like, for each point, the change in orientation between the preceeding and following curve segments...

Anyhow, a perimeter which has only continuous changes in its direction (ie power gong to zero at high frequencies), will not have any "sharp" points, if I've got everything right so far. Also, a blob whose perimeter is a pure circle would only have power in the second lowest Fourier component (i.e. 1 period per cycle around the perimeter). More [frequent or abrupt] twists and turns will add higher frequencies, etc..

But back to the problem of varying sampling frequency: Use the accumulated length of the segments as you go around the perimeter to parameterize your derivative, then do a linear (or higher order, as desired) interpolation onto a fixed-spacing array before doing the FFT's..

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

NASA Goddard Space Flight Center, Email: shaugan@esa.nascom.nasa.gov
Mail Code 682.3, Bld. 26, Room G-1, Tel.: 1-301-286-9028/240-354-6066
Greenbelt, Maryland 20771, USA. Fax: 1-301-286-0264
