Subject: Re: Is there a quick way to find the intersection of two lines?
Posted by ben.bighair on Tue, 05 Feb 2008 00:33:39 GMT
View Forum Message <> Reply to Message

On Feb 4, 7:08 pm, eyuc...@gmail.com wrote:
> Hi there,
>
> I have two sets of x-y data:
> x1=[1,2,3,4,5] y1=[3.2,7.4,8.2,9.3,7.9];
> x2=[1.2,1.4,2.3,2.8,3.3,3.9,4.1,4.5,5.2]
> y2=[3.1,5.2,6.2,7.3,7.5,8.6,9.6,8.7,7.4];
>
> By running:
> plot, x1, y1
> oplot, x2, y2
> we can clearly see that there are four intersections, but it is not
> clear what are the x,y coordinates of these points.
>
> Is there an easy way to do it? Thank you very much.
>

Hi,

You might want to check out Paul Bourke's great online tidbits about geometry.

http://local.wasp.uwa.edu.au/~pbourke/geometry/

I have been chipping away at coding some of the algorithms he describes into IDL, but have been easily sidetracked. You're welcome to use what I have (mostly documented) as a starting point. My implementations come with zero warranty...

www.tidewater.net/~pemaquid/pb.zip

Cheers,
Ben

Subject: Re: Is there a quick way to find the intersection of two lines?
Posted by eyuchen on Tue, 05 Feb 2008 08:24:33 GMT
View Forum Message <> Reply to Message

On Feb 4, 4:33 pm, "ben.bighair" <ben.bigh...@gmail.com> wrote:
> On Feb 4, 7:08 pm, eyuc...@gmail.com wrote:
>
>> Hi there,
>

>> I have two sets of x-y data:
>> x1=[1,2,3,4,5] y1=[3.2,7.4,8.2,9.3,7.9];
>> x2=[1.2,1.4,2.3,2.8,3.3,3.9,4.1,4.5,5.2]
>> y2=[3.1,5.2,6.2,7.3,7.5,8.6,9.6,8.7,7.4];
>
>> By running:
>> plot, x1, y1
>> oplot, x2, y2
>> we can clearly see that there are four intersections, but it is not
>> clear what are the x,y coordinates of these points.
>
>> Is there an easy way to do it? Thank you very much.
>
> Hi,
>
Thank you very much. I think I do understand how to find the
intersection of two lines in principle, but actually doing it requires
some details such as narrowing down the interval that the two lines
intersect, counting intersection points, allocating memory to store
the points and finally solving them.

I just wonder if there are pre-made subroutines, since this is really
a job we often do. If there are none, I'll try to make one...

> You might want to check out Paul Bourke's great online tidbits about
> geometry.
>
> http://local.wasp.uwa.edu.au/~pbourke/geometry/
>
> I have been chipping away at coding some of the algorithms he
> describes into IDL, but have been easily sidetracked. You're welcome
> to use what I have (mostly documented) as a starting point. My
> implementations come with zero warranty...
>
> www.tidewater.net/~pemaquid/pb.zip
>
> Cheers,
> Ben

Subject: Re: Is there a quick way to find the intersection of two lines?
Posted by Wox on Tue, 05 Feb 2008 10:45:45 GMT
View Forum Message <> Reply to Message

On Mon, 4 Feb 2008 16:08:16 -0800 (PST), eyuchen@gmail.com wrote:

> Hi there,
>

> I have two sets of x-y data:
> x1=[1,2,3,4,5] y1=[3.2,7.4,8.2,9.3,7.9];
> x2=[1.2,1.4,2.3,2.8,3.3,3.9,4.1,4.5,5.2]
> y2=[3.1,5.2,6.2,7.3,7.5,8.6,9.6,8.7,7.4];
>
> By running:
> plot, x1, y1
> oplot, x2, y2
> we can clearly see that there are four intersections, but it is not
> clear what are the x,y coordinates of these points.
>
> Is there an easy way to do it? Thank you very much.
>
> Eugene

I got carried away and made something with Ben's suggestion. Maybe
there's a better way, but it's a start.

```
function segmentintersect,L1x,L1y,L2x,L2y,xy=xy

; code:
; 0: not intersecting
; 1: intersect in 1 point
; 2: parallel
; 3: coincident

 denom=float(L2y[1]-L2y[0])*(L1x[1]-L1x[0])-(L2x[1]-L2x[0])*( L1y[1]-L1y[0])
 numa=(L2x[1]-L2x[0])*(L1y[0]-L2y[0])-(L2y[1]-L2y[0])*(L1x[0] -L2x[0])
 numb=(L1x[1]-L1x[0])*(L1y[0]-L2y[0])-(L1y[1]-L1y[0])*(L1x[0] -L2x[0])

if denom eq 0 then code= (numa eq 0 and numb eq 0)+2 $
else begin
 ua = numa / denom
 ub = numb / denom

 code= ua ge 0 and ua le 1 and ub ge 0 and ub le 1
 if code then
xy=[L1x[0]+ua*(L1x[1]-L1x[0]),L1y[0]+ua*(L1y[1]-L1y[0])]
endelse

return,code
end;function segmentintersect
```

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%% %%%%%%%%%%

pro segtest

x1=[1,2,3,4,5]
y1=[3.2,7.4,8.2,9.3,7.9];
x2=[1.2,1.4,2.3,2.8,3.3,3.9,4.1,4.5,5.2]
y2=[3.1,5.2,6.2,7.3,7.5,8.6,9.6,8.7,7.4]
window
plot,x1,y1,psym=-2
oplot,x2,y2,psym=-2

n=n_elements(x2)
y2_1=interpol(y1,x1,x2)
b=y2_1 gt y2
interval=where(b[0:n-2]-b[1:*],ct)
if ct ne 0 then begin
 xy=fltarr(2,ct)
 for i=0,ct-1 do begin
  j=interval[i]
  L2x=x2[j:j+1]
  L2y=y2[j:j+1]
  j=value_locate(x1,L2x)
  k=0
  repeat begin
   L1x=x1[j[k]:j[k]+1]
   L1y=y1[j[k]:j[k]+1]
   code=segmentintersect(L1x,L1y,L2x,L2y,xy=tmp)
   b=code eq 1
   if b then begin
    xy[*,i]=tmp
    plots,[tmp[0],tmp[0]],[0,tmp[1]],/data
   endif
   k++
  endrep until b or (k eq 2)
 endfor
endif

end;pro segtest

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%% %%%%%%%%%%
```

Subject: Re: Is there a quick way to find the intersection of two lines?
Posted by ben.bighair on Tue, 05 Feb 2008 14:00:31 GMT

On Feb 5, 3:24 am, eyuc...@gmail.com wrote:
> On Feb 4, 4:33 pm, "ben.bighair" <ben.bigh...@gmail.com> wrote:
>
>> On Feb 4, 7:08 pm, eyuc...@gmail.com wrote:
>
>>> Hi there,
>
>>> I have two sets of x-y data:
>>> x1=[1,2,3,4,5] y1=[3.2,7.4,8.2,9.3,7.9];
>>> x2=[1.2,1.4,2.3,2.8,3.3,3.9,4.1,4.5,5.2]
>>> y2=[3.1,5.2,6.2,7.3,7.5,8.6,9.6,8.7,7.4];
>
>>> By running:
>>> plot, x1, y1
>>> oplot, x2, y2
>>> we can clearly see that there are four intersections, but it is not
>>> clear what are the x,y coordinates of these points.
>
>>> Is there an easy way to do it? Thank you very much.
>
>> Hi,
>
> Thank you very much. I think I do understand how to find the
> intersection of two lines in principle, but actually doing it requires
> some details such as narrowing down the interval that the two lines
> intersect, counting intersection points, allocating memory to store
> the points and finally solving them.
>
> I just wonder if there are pre-made subroutines, since this is really
> a job we often do. If there are none, I'll try to make one...
>
>> You might want to check out Paul Bourke's great online tidbits about
>> geometry.
>
>> http://local.wasp.uwa.edu.au/~pbourke/geometry/
>
>> I have been chipping away at coding some of the algorithms he
>> describes into IDL, but have been easily sidetracked. You're welcome
>> to use what I have (mostly documented) as a starting point. My
>> implementations come with zero warranty...
>
>> www.tidewater.net/~pemaquid/pb.zip
>

Hi,

I think the routines you are looking for might be found in the pb.zip

collection I shared. You might need to work them in a sequence testing each segment of polyline A against polyline B, but the functions are all there.

Cheers,
Ben