Subject: Re: Newbie to IDL needs help :)
Posted by Jeremy Bailin on Thu, 09 May 2013 23:09:21 GMT
View Forum Message <> Reply to Message

On 5/9/13 5:25 PM, alchemymetalworks@gmail.com wrote:
> Ok, this is probably simple and I'm just missing something obvious. I'm basically trying to self-teach myself some IDL to complete a project at work, and since I've spent the last 2 days fighting the urge to throw my keyboard through my monitor I decided to put aside my pride and beg for help from the masses/experts... so here goes...
>
> I essentially have 4 maps that were created in ENVI (remote sensing software, if anyone's familiar with it). The maps are the exact same size/pixel count. Each pixel in each map is assigned a value from 1 - 15 based on the land cover that is (assumed)present. I want to create a composite map from these 4, using a set of rules:
>
> -if the value is the same in all the maps, keep that value in the final output map
> -if the value is 1, 2, 4-9, or 13 use the pixels with that value from map 1
> -if the value is 3, 8, 12, 14, or 15 use the pixels with that value from map 2
> -if the value is 10 use the pixels with that value from map 3
> -if the value is 11 use the pixels with that value from map 4
> -if a pixel meets several of these criteria (for example, pixel P is value 1 in map 1 and value 10 in map 3)I want to designate which map should take priority.
>
> The goal is to ultimately have a single output with all pixels classified (again) as 1-15 based on the preferred map(s), if that makes sense. And I'm stumped... I know what I want to do, but I can't figure out how to tell the computer to do it... Any help/guidance would be greatly appreciated.
>

I agree with David that I'm kind of getting lost, but I think this might
do what you're looking for. There's probably a more elegant way of
dealing with the masks, but this should work.

I am assuming that your maps are called map1...map4.

```
; create masks of which pixels you want from each map
mask_for_map1 = (map1 eq 1) or (map1 eq 2) or (map1 eq 4) or $
   (map1 eq 5) or (map1 eq 6) or (map1 eq 7) or (map1 eq 8) or $
   (map1 eq 9) or (map1 eq 13)
mask_for_map2 = (map2 eq 3) or (map2 eq 8) or (map2 eq 12) or $
   (map2 eq 14) or (map2 eq 15)
mask_for_map3 = (map3 eq 10)
mask_for_map4 = (map4 eq 11)

; clobber lower priority ones in the case where a pixel is part
; of the mask of a higher priority map. In this case, I am arbitrarily
; assuming map3 takes precedence, followed by maps 1, 4, 2 to show
; you the structure of the code.
mask_for_map1 *= (1-mask_for_map3)
```

```
mask_for_map4 *= (1-mask_for_map3)*(1-mask_for_map1)
mask_for_map2 *= (1-mask_for_map3)*(1-mask_for_map1)*(1-mask_for_map4)

combined_map = map1*mask_for_map1 + map2*mask_for_map2 + $
   map3*mask_for_map3 + map4*mask_for_map4
```

-Jeremy.
