
Subject: How do I reduce active memory used by program?

Posted by [eite2335](#) on Sun, 19 Mar 2006 20:57:04 GMT

[View Forum Message](#) <> [Reply to Message](#)

Dear IDL community,

I wrote a little program which calculates x, y, and z ground control points (gcp) for each pixel of an image. To do this, I am placing pitch, roll, and yaw parameters into a formula given by Hongjian and Shukai (2004)

The problem I have is that my program works fine for small images e.g. if I do have only 100 scanlines. However, if I try to calculate the gcps for larger images (e.g. > than 5000 scanlines) the computer crashes because it uses too much active memory.

I am suspecting that too much active memory is necessary for large images due to the therefore resulting large loops and arrays in my program.

Since I am new to programming I do not really know how to reduce the active memory necessary to run the program for large images.

It would be great if somebody has a suggestion, how to reduce the amount of active memory necessary to run this program so that the program can also be used to calculate gcp's for large images.

I really appreciate your time and help with this

Best regards,

Jan

```
.....  
; Programmer: Jan Eitel  
; Last update: 3-17-2006  
; Program to calculate x, y, and z ground control points (gcp's) by  
; using pitch, roll, and yaw variables provided by  
; the inertial navigation system.
```

```

pro easy_shift
;# of columns in the image
Columns = 752
;# of rows in the image
Rows = 100
;# of loops
rep = (Rows-1)
;; Type in average flying height (fh)
fh = 4000
;; Type in average elevation
elev = 2583
;; Convert flying height and elevation to meters
fh = fh*0.3048
elev = elev*0.3048
;; Calculate flying height above sea level
fhasl = fh - elev
;; Field of view (FOV) per scanline
fov = 15.2483314 * (3.14159265/180)/376
;; Create array. Array will be used to place the calculated internal
geometry values into

```

```
xyz_array = make_array(3, 752, /double, value = 0)
```

```
;; calculate the internal geometry of the sensor:
```

```
for i = 1, 376 do begin
```

```
  fovsl = fov * i
```

```
  x = tan(fovsl)*fhasl
```

```
  y = x - xyz_array[0, i + 374]
```

```
  z = fhasl/cos(fovsl)
```

```
  b = 375 + i
```

```
  xyz_array[0,b] = x
```

```
  xyz_array[1,b] = y
```

```
  xyz_array[2,b] = z
```

```
endfor
```

```
for i = 1, 376 do begin
```

```
  b = 376 - i
```

```
  c = 375 + i
```

```
  xyz_array[0,b] = xyz_array[0, c] * (-1)
```

```
  xyz_array[1,b] = xyz_array[1, c] * (-1)
```

```
  xyz_array[2,b] = xyz_array[2, c]
```

```
endfor
```

```
;; read in roll, pitch, yaw, xgps, ygps, zgps
```

```
openr, 1, 'C:/HC/input1.txt'
```

```

ins = dblarr(5, Rows)

readf, 1, ins

close, 1

;; i is the scanline number (starting with scanline #1)

i = 0

o = ins[0, i]

w = ins[1, i]

k = ins[2, i]

;; populate matrix given in Hongjian and Shukai (2004) and calculate
gcp (ground control points)

a1 = cos(k)*cos(o)

a2 = -cos(k)*sin(o)*sin(w) - sin(k)*cos(w)

a3 = sin(k)*cos(o)

b1 = sin(k)*cos(o)

b2 = -sin(k)*sin(o)*sin(w)+cos(k)*cos(w)

b3 = -sin(k)*sin(o)*cos(w)-cos(k)*sin(w)

c1 = sin(o)

c2 = cos(o)*sin(w)

c3 = cos(o)*cos(w)

matrix1 = make_array(3, 3, /double, value = 0)

matrix1[0,0] = a1

matrix1[1,0] = a2

```

```
matrix1[2,0] = a3
```

```
matrix1[0,1] = b1
```

```
matrix1[1,1] = b2
```

```
matrix1[2,1] = b3
```

```
matrix1[0,2] = c1
```

```
matrix1[1,2] = c2
```

```
matrix1[2,2] = c3
```

```
xgps = ins[3, i]
```

```
ygps = ins[4, i]
```

```
zgps = fhasl
```

```
matrix3 = make_array(1, 3, /double, value = 0)
```

```
matrix3[0,0] = xgps
```

```
matrix3[0,1] = ygps
```

```
matrix3[0,2] = zgps
```

```
matrix2 = xyz_array
```

```
matrix2 = matrix2[0:2,0]
```

```
matrix2 = transpose(matrix2)
```

```
cf = (matrix1 ## matrix2) + matrix3
```

```
cf_x = cf[0,0]
```

```
cf_y = cf[0,1]
```

```
cf_z = cf[0,2]
```

```
out_x = (cf_x)
```

```
out_y = (cf_y)
```

```
out_z = (cf_z)
```

```
for g = 1, 751 do begin
```

```
matrix2 = xyz_array
```

```
matrix2 = matrix2[0:2,g]
```

```
matrix2 = transpose(matrix2)
```

```
cf = (matrix1 ## matrix2) + matrix3
```

```
cf_x = cf[0,0]
```

```
cf_y = cf[0,1]
```

```
cf_z = cf[0,2]
```

```
out_x = [[out_x], [cf_x]]
```

```
out_y = [[out_y], [cf_y]]
```

```
out_z = [[out_z], [cf_z]]
```

```
endfor
```

```
for i = 1, rep do begin
```

```
o = ins[0, i]
```

```
w = ins[1, i]
```

```
k = ins[2, i]
```

```
;;populate matrix given in Hongjian and Shukai (2004) and calculate gcp  
(ground control points)
```

```
a1 = cos(k)*cos(o)
```

```
a2 = -cos(k)*sin(o)*sin(w) - sin(k)*cos(w)
```

```
a3 = sin(k)*cos(o)
```

```
b1 = sin(k)*cos(o)
```

```
b2 = -sin(k)*sin(o)*sin(w)+cos(k)*cos(w)
```

```
b3 = -sin(k)*sin(o)*cos(w)-cos(k)*sin(w)
```

```
c1 = sin(o)
```

```
c2 = cos(o)*sin(w)
```

```
c3 = cos(o)*cos(w)
```

```
matrix1 = make_array(3, 3, /double, value = 0)
```

```
matrix1[0,0] = a1
```

```
matrix1[1,0] = a2
```

```
matrix1[2,0] = a3
```

```
matrix1[0,1] = b1
```

```
matrix1[1,1] = b2
```

```
matrix1[2,1] = b3
```

```
matrix1[0,2] = c1
```

```
matrix1[1,2] = c2
```

```
matrix1[2,2] = c3
```

```
xgps = ins[3, i]
```

```
ygps = ins[4, i]
```

```
zgps = fhasl
```

```
matrix3 = make_array(1, 3, /double, value = 0)
```

```
matrix3[0,0] = xgps
```

```
matrix3[0,1] = ygps
```

```
matrix3[0,2] = zgps
```

```
matrix2 = xyz_array
```

```
matrix2 = matrix2[0:2,0]
```

```
matrix2 = transpose(matrix2)
```

```
cf = (matrix1 ## matrix2) + matrix3
```

```
cf_x = cf[0,0]
```

```
cf_y = cf[0,1]
```

```
cf_z = cf[0,2]
```

```
out_xx = (cf_x)
```

```
out_yy = (cf_y)
```

```
out_zz = (cf_z)
```

```
for g = 1, 751 do begin
```

```
matrix2 = xyz_array
```

```
matrix2 = matrix2[0:2,g]
```

```
matrix2 = transpose(matrix2)
```

```
cf = (matrix1 ## matrix2) + matrix3
```

```
cf_x = cf[0,0]
```

```
cf_y = cf[0,1]
```

```
cf_z = cf[0,2]
```

```
out_xx = ([[out_xx], [cf_x]])
```

```
out_yy = ([[out_yy], [cf_y]])
```

```
out_zz = ([[out_zz], [cf_z]])
```

```
endfor
```

```
out_x = ([[out_x], [out_xx]])
```

```
out_y = ([[out_y], [out_yy]])
```

```
out_z = ([[out_z], [out_zz]])
```

endfor

;Create arrays to place x, y, z gcp into

output_x = make_array(752, rows, /string, value = 0)

output_y = make_array(752, rows, /string, value = 0)

output_z = make_array(752, rows, /string, value = 0)

counter = lindgen(752, rows)

for i = 0, rep do begin

for j = 0, 751 do begin

counter1 = counter[j, i]

output_x [j, i] = out_x [0, counter1]

output_y [j, i] = out_y [0, counter1]

output_z [j, i] = out_z [0, counter1]

endfor

endfor

;Write x gcp to file

data = output_x

ColumnHeaders=columnHeaders

Filename= 'C:/HC/x_gcp.txt'

width = 10000

;define delimiter

Delimiter = String(9B)

```
OpenW, lun, filename, /Get_Lun, Width=width
```

```
sData = StrTrim(data,1)
```

```
for i = 0, rep do begin
```

```
sData[751, i] = sData[751, i]
```

```
endfor
```

```
PrintF, lun, sData
```

```
; Close the file.
```

```
Free_Lun, lun
```

```
; Write y gcp to file
```

```
data1 = output_y
```

```
ColumnHeaders=columnHeaders
```

```
Filename= 'C:/HC/y_gcp.txt'
```

```
width = 10000
```

```
Delimiter = String(9B)
```

```
OpenW, lun, filename, /Get_Lun, Width=width
```

```
sData = StrTrim(data1,1)
```

```
for i = 0, rep do begin
```

```
sData[751, i] = sData[751, i]
```

```
endfor
```

```
PrintF, lun, sData
```

```
Free_Lun, lun
```

```
;Write y gcp to file

data2 = output_z

ColumnHeaders=columnHeaders

Filename= 'C:/HC/z_gcp.txt'

width = 10000

Delimiter = String(9B)

OpenW, lun, filename, /Get_Lun, Width=width

sData = StrTrim(data2,1)

for i = 0, rep do begin

sData[751, i] = sData[751, i]

endfor

PrintF, lun, sData

Free_Lun, lun

END
```
