Revision: **1.3**- 24 March, 2004

Image Position Elements

A number of redundant private elements have also been eliminated in the Excite II release. Images with this change will have a software version number of 11 or greater, contained in the first value of DICOM element (0018,1020).

The private elements containing Series Plane (0019,1017) and Plane Type (0027,1035) will be maintained for local query and parameter computation purposes.

EXCITE II has dropped the following private elements which contain redundant spatial information (used for reverse genesis translation):

- loc_ras (0027,1040)
- loc (0027,1041) this is already contained in (0020,1041)
- ctr_X (0027,1042-1044)
- norm_X (0027,1045-1047)
- trhc_X (0027,1048-104A
- brhc_X (0027,104B-104D)
- obplane (0027,1036)
- series start/end loc (0019,1018-101B)

Note: The series start/end loc elements will be dropped only for Mx, 3-plane, and RTIA scans.

The algorithms described in the equations section can be used to recover the private spatial position elements from the remaining DICOM elements. The following DICOM elements are required for this computation.

- Image Position (0020,0032) multi-valued (3 values) order L, P, S
- Image Orientation (0020,0037) (direction cosine), multi-valued (6 values) order: row L,P,S column L,P,S
- pixel size (0028,0030) multi-valued (2 values) order pixel_size_Y, pixel_size_X
- imatrix_X (0028,0011)
- imatrix_Y (0028,0010)
- % phase fov (0018,0094)
- xres, yres (0018,1310) multi-valued (4 values) order is either xres,0,0,yres or 0,xres,yres,0
- scan options (0018,0022) multi-valued, need to check if SQPIX_GEMS is present
- plane type (0027,1035)

Note: GE MR GENESIS images defined the image position element to be the upper left hand corner of the image. The DICOM standard specifies that the image position field contains the x,y,z coordinate of the center of the upper left hand corner pixel of the image, in mm. The calculation for the GENESIS corner points takes into account this ½ pixel shift.

Nomenclature:

For the following equations the something_C is an abbreviation for a three dimensional vector, for example:

tlhc_C means [tlhc_R,tlhc_A,tlhc_S]
row_C means [row_L,row_P,row_S]

also:

tlhc means top left hand corner
trhc means top right hand corner
brhc means bottom right hand corner

Using the DICOM element (0020,0037)

row_C be the (LPS) coordinates of the direction cosine in the row direction
col_C be the (LPS) coordinates of the direction cosine in the column direction

Using the DICOM elements (0028,0010) (0028,0011) and (0028,0030)

norm_row = pixel_size_X * imatrix_X
norm_col = pixel_size_Y * imatrix_Y

ip_X = (LPS) coordinates of the image position (0028,0032)

Mask values:

The following table shows the mask values that are used in the plane and obplane variables to designate the image plane orientation. Used in equations 7 and 8.

Plane Designator	Value
AXIAL_PLANE	2
SAGITTAL_PLANE	4
CORONAL_PLANE	8
OBLIQUE_PLANE	16
OBLIQUE_AXIAL_PLANE	18
OBLIQUE_SAGITTAL_PLANE	20
OBLIQUE_CORONAL_PLANE	24

Equations:

1. loc use standard element (0020,1041)

2. tlhc_C (was (0021,1054), need to offset from image position (0020,0032) due to ½ pixel shift in DICOM coordinate definition)

```
tlhc_R = -ip_L + (row_L * pixel_size_X/2) + (col_L * pixel_size_Y/2)
tlhc_A = -ip_P + (row_P * pixel_size_X/2) + (col_P * pixel_size_Y/2)
tlhc_S =  ip_S – (row_S * pixel_size_X/2) – (col_S * pixel_size_Y/2)
```

3. trhc_C

```
trhc_R = tlhc_R – (row_L * norm_row)
trhc_A = tlhc_A – (row_P * norm_row)
trhc_S = tlhc_S + (row_S * norm_row)
```

4. brhc_C

```
brhc_R = trhc_R – (col_L * norm_col)
brhc_A = trhc_A – (col_P * norm_col)
brhc_S = trhc_S + (col_S * norm_col)
```

5. ctr_C

```
ctr_R = (tlhc_R + brhc_R) / 2
ctr_A = (tlhc_A + brhc_A) / 2
ctr_S = (tlhc_S + brhc_S) / 2
```

6. norm_C (this is the normalized cross product of the Row and Column vectors relative to the top left corner) Note, you can also use the row and column direction cosines instead of the corner points in this calculation since we really want unit vectors.

   using corner points:

```
row_R = trhc_R - tlhc_R                 col_R = brhc_R – trhc_R
row_A = trhc_A – tlhc_A                 col_A = brhc_A – trhc_A
row_S = trhc_S – tlhc_S                 col_S = brhc_S – trhc_S
```

   or using direction cosines:

```
row_R = -row_L                          col_R = -col_L
row_A = -row_P                          col_A = -col_P
row_S =  row_S                          col_S =  col_S

R = row_A * col_S - row_S *col_A
A = row_S * col_R - row_R *col_S
S = row_R * col_A - row_A *col_R

M = sqrt( R*R + A*A +S*S)
norm_R = R / M
norm_A = A / M
norm_S = S / M
```

7. obplane (uses plane (0027,1035) and results from norm_C above, eps = 0.00001)

```
fnr = | norm_R |
fna = | norm_A |
fns = | norm_S |
obplane = plane
if (plane & OBLIQUE_PLANE)
       if ( | fnr – fna | < eps ) fnr = fna
       if ( | fnr – fns | < eps ) fnr = fns
       if ( | fna – fns | < eps ) fna = fns
       obplane = OBLIQUE_AXIAL_PLANE
       if ( (fna > fnr) && (fna > fns)) obplane = OBLIQUE_CORONAL_PLANE
       if (!(fna > fnr) && (fnr > fns)) obplane = OBLIQUE_SAGITTAL_PLANE
endif
```

8. loc_ras (uses results from ctr_C above and mask bits for plane orientation)

```
if (obplane & AXIAL_PLANE)          loc_ras = (ctr_S < 0) ? 'I' : 'S'
else if (obplane & SAGITTAL_PLANE)  loc_ras = (ctr_R < 0) ? 'L' : 'R'
else                                loc_ras = (ctr_A < 0) ? 'P' : 'A'
```

9. Field of view (note: dfov_rect is not relevant for propeller or spiral images)

```
dfov = imatrix_X * pixel_size_X

if (SQPIX_GEMS) then
       dfov_rect = dfov * (yres / xres)
else
       dfov_rect = dfov * (%phase fov / 100.0)
```