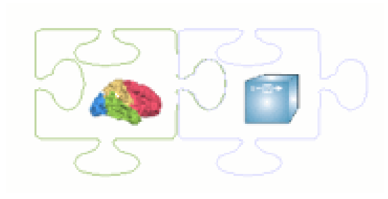
addendum for nifti converter plugins



Hester Breman and Rainer Goebel

Maastricht, 2013

1 Spatial transformations in BrainVoyager

Rainer Goebel

1.1 Introduction

This technical document aims to provide detailed knowledge about spatial transformations in general and how they are implemented in BrainVoyager. The document focuses on the axes systems used in BrainVoyager and the assumed order of axes rotations. In addition, it is described how rotation, translation and scaling transformations are properly combined to create a $4x4$ affine transformation matrix as well as how such a matrix is properly decomposed into elementary transformations. The presented information is aimed towards advanced users who want to a) simply understand these issues better or b) want to use transformation results from other software in BrainVoyager or c) want to use transformation results produced by BrainVoyager for other (custom) software.

It is a necessity that successive rotations about coordinate axes are treated consistently in all volume- and surface-level coordinate transformation routines of BrainVoyager. This is particularly important since successive axis rotations (in contrast to successive translations) do not commute, in that the composed transformation depends on the order in which individual rotations are applied. BrainVoyager saves spatial transformations in a .TRF file, which contains, among other parameters, three values for rotations around the three coordinate axes. The order of applying these angles must be consistent across different modules of the program. It is, for example, possible to load a TRF file within the surface module in order to apply the same transformation on a surface which had been previously applied to a 3D VMR data set, or vice versa. Besides ensuring a consistent explicit specification of rotation angles across modules, all automatic rigid body coregistration routines (3D motion correction, 3D-3D coregistration etc.) also have to result in rotation angles, which are consistent with the implied order of axes rotations.

1.2 BrainVoyager's axes conventions

BrainVoyager uses several different coordinate systems: the internal axes, the standard Dicom and Talairach axes and the OpenGL axes. To the user, normally only the Dicom/Talairach axes system is presented. The internal axes system of BrainVoyager was defined initially for sagittal 3D volumes. The dimensions of the sagittal images defined the X and Y axes with values ranging from 0 to 255 (X: anterior to posterior, Y: superior to inferior) and the dimension across the slices defined the Z axis (right-to-left) with values from 0 to 255 or less. This original decision (which was at the end of the year 1995) is still the basis of the internal axes system, which is depicted in figure **??**.

To the user, the axes are presented according to the Talairach/Dicom naming standard, i.e. the X axis in Talairach space corresponds to the Z axis in BVs internal definition, the Y axis in Talairach space corresponds to the X axis in BVs internal definition, and the Z axis in Talairach space corresponds to the Y axis in BVs interal definition. Note that so far, this only changes the labeling of the axes, the values are still from 0 to 255 along these relabeled axes in the original direction. Since these relabeled axes are still internal definitions, they are shown to the user as "system coordinates", for example in the System coords field of the 3D Volume Tools dialog (see below). In the following these relabeled axes will be called X_{SYS}, Y_{SYS} and Z_{SYS} (SYS for "System"). The original (internal) axes are referred to as X_{BV}, Y_{BV} and Z_{BV} (BV for "BrainVoyager internal").

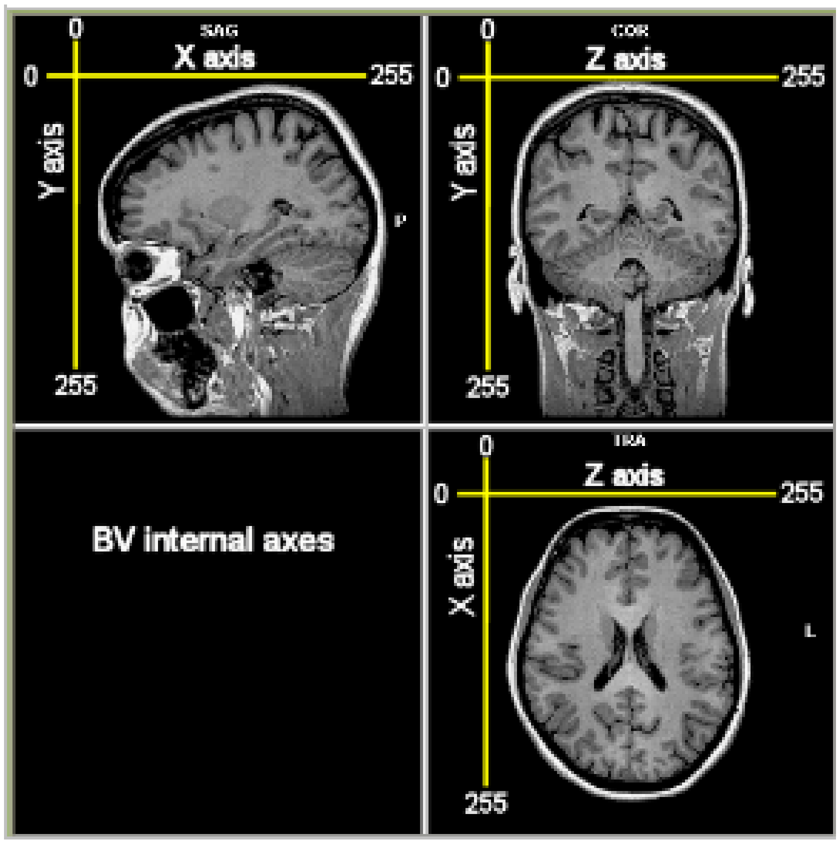


Figure 1: The BrainVoyager internal coordinate system

Besides the internal (BV) and system (SYS) coordinates, BrainVoyager also supports "real" Talairach coordinates, if appropriate. In the Talairach coordinate system, the origin and axes values are defined with respect to landmarks of the brain. Most importantly, the origin of the coordinate system is specified to be the anterior commissure (AC) of the brain. Together with the posterior commissure (PC) and additional landmarks specifying the border of the brain, the values along the X, Y, and Z axis are defined. These Talairach coordinates are shown in the Talairach coords field (see figure above). To enable Talairach coordinates, the Use Tal ref points option has to be checked in the Talairach tab of the 3D Volume Tools dialog. The figure below shows that the directions of the Talairach axes are oppositely defined as compared to the internal/system axes (compare 0 to 255 with $-$ to $+$ directions). The Talairach axes will be referred to as X_{TAL}, Y_{TAL} and Z_{TAL}. In BrainVoyager, a brain is transformed into Talairach space in two steps, 1) ACPC transformation and 2) Talairach scaling based on the proportional grid system. The first step is a normal rigid body transformation (represented with a standard TRF file) while the latter requires a special step based on a "TAL" file. A TAL file contains x,y,z specifications of the AC and PC points and the cerebrum borders defined on the ACPC brain. The landmarks are used for Talairach piecewise scaling of the ACPC brain according to the proportional grid (Talairach & Tournaux, 1988) resulting in a normalized brain.

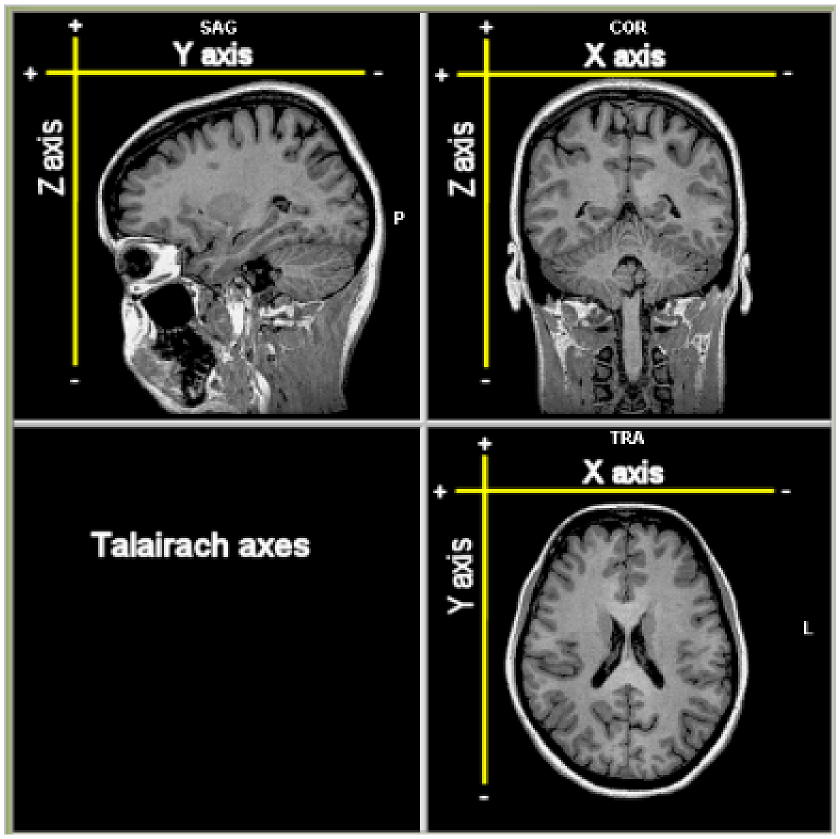
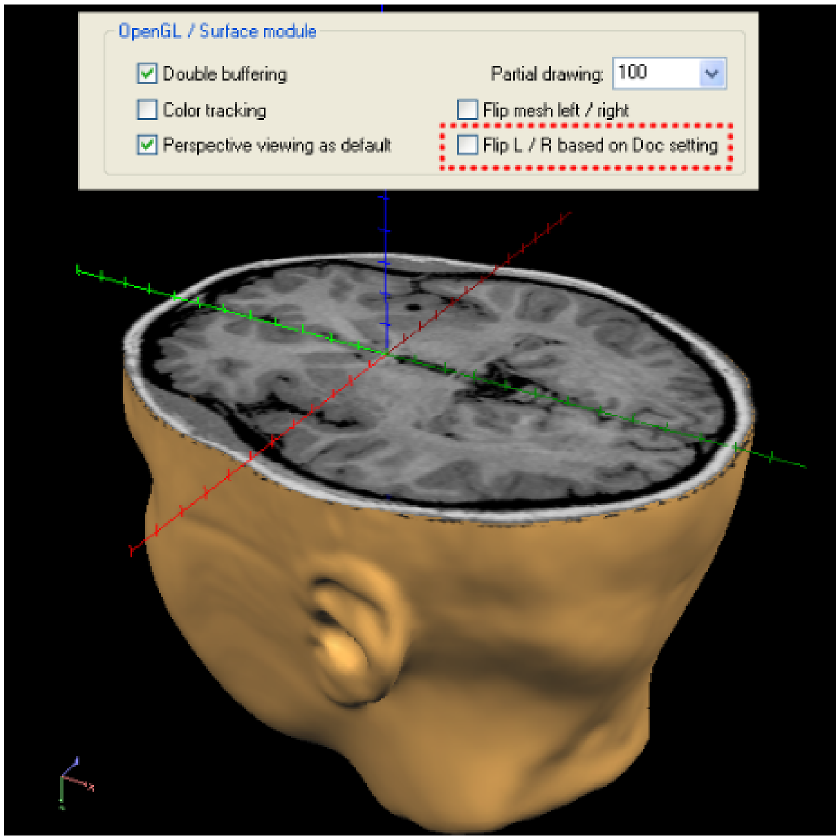


Figure 2: The Talairach coordinate system

Note that the definitions of the system coordinates assume that the brain is in BrainVoyagers "standard" orientation, i.e. that 3D data is represented as a series of sagittal planes. If this is not the case for a raw data set, the program provides the "To SAG" function to exchange the axes accordingly. BrainVoyager QX tries to perform this step automatically based on header information and an analysis of the symmetry properties of the data set. BrainVoyagers standard orientation also assumes that the data set is in radiological convention ("Left-Is-Right"). This is normally the case when reading native scanner data (manufacturers DICOM or proprietary file formats such as Siemens IMA (Numaris versions prior to 4) or GE "I" or GE "MR" files). If you are sure that your data is not in radiological but in neurological convention (Left-Is-Left), you have to specify this in the *Transform to Standard SAG* dialog. Data in neurological convention may be encountered if you read data not directly from the scanner but from files exported by another program.

The surface module visualizes reconstructed meshes and optionally displays two coordinate frames, the OpenGL and the Talairach coordinate system. The OpenGL coordinate axes (see figure below) are shown in the lower left corner and correspond directly to the system coordinates ($X_{OGL} = X_{BV}$ etc.). The OpenGL axes are identified with letters as well as with a color code denoting the X axis with red, the Y axis with green and the Z axis with blue. In addition to the OpenGL axes, the Talairach coordinate system is also shown (see figure below). The axes can be identified by color, i.e. the X_{TAL} axis is drawn in red, the Y_{TAL} axis in green and the Z_{TAL} axis in blue. If enabled, the Talairach axes are shown always in the same way even if a displayed mesh is not normalized into Talairach space. The mesh shown in figure **??** is drawn 1-to-1 from voxel coordinates of the corresponding 3D (VMR) data set. Since the original VMR data sets are normally in radiological convention, the mesh is shown in a left-right mirrored way.

As default, the surface module does not display the meshes as shown above but attempts always to display them in its natural space: The program tries to assure that the seen left side of a mesh always corresponds to the true left side of the data set independent of neurological or radiological convention. To accomplish this, a general flag `Flip L / R based on Doc` setting is defined, which flips the values of the left-right coordinate axis in case that the data set is in radiological convention. This flag can be found in the Global Options dialog, which can be invoked by clicking the File $\rightarrow$ Global options menu item (see figures above and below). This flag is checked as default and it should be always turned on to ensure correct display of a mesh as shown in the figure below.



OpenGL / Surface module

[✓] Double buffering

[ ] Color tracking

[✓] Perspective viewing as default

Partial drawing: 100

[ ] Flip mesh left / right

[ ] Flip L / R based on Doc setting

Figure 3: The OpenGL coordinate system

1.3 3D affine transformation matrices

Any combination of translation, rotations, scalings/reflections and shears can be combined in a single 4 by 4 affine transformation matrix:

$$M = \begin{pmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

The 4 by 4 matrix M corresponds to a affine transformation T() that transforms point v to point u. In other words, the transformation of point u is found by multiplying v by M:

$$u = Mv \tag{2}$$

The 4 by 4 transformation matrix uses homogeneous coordinates, which allow to distinguish between points and vectors. Vectors have a direction and magnitude whereas points are at certain coordinates with respect to the origin and the three base vectors i, j and k. Points and vectors are both represented as mathematical column vectors (column-matrix representation scheme, see note below) in homogeneous coordinates with the difference that points have a 1 in the fourth position whereas vectors have a zero at this position. The transformation of the point v to point u is thus written as:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 0 \end{pmatrix} = M \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix} \tag{3}$$

We now consider the nature of elementary 3D transformations individually and then compose them into general affine transformations. Note that for an affine transformation matrix, the final row of the matrix is always 0001 and we have to understand the role of the elements in the upper 3 by 4 matrix.

1.3.1 Translation

For a pure translation, the matrix M has the simple form:

$$\begin{pmatrix} 1 & 0 & 0 & M_{14} \\ 0 & 1 & 0 & M_{24} \\ 0 & 0 & 1 & M_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}$$

Applying this matrix to point v reveals that $u = Mv$ is simply a shift in v by the vector $t = (t_x = m_{14}, t_y = m_{24}, t_z = m_{34})$.

$$M = \begin{pmatrix} 1 & 0 & 0 & x + t_x \\ 0 & 1 & 0 & y + t_y \\ 0 & 0 & 1 & z + t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

1.3.2 Scaling

A scaling operation along the three axes is represented by the following matrix:

$$\begin{pmatrix} m_{11} & 0 & 0 & 0 \\ 0 & m_{22} & 0 & 0 \\ 0 & 0 & m_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (6)$$

Applying this matrix to point v results in (with $sx = m_{11}$, $sy = m_{22}$, $sz = m_{33}$):

$$M = \begin{pmatrix} x' \\ y' \\ z' \\ 0 \end{pmatrix} = \begin{pmatrix} m_{11} & 0 & 0 & 0 \\ 0 & m_{22} & 0 & 0 \\ 0 & 0 & m_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} x & v_x \\ y & v_y \\ z & v_z \\ 0 & 0 \end{pmatrix} \quad (7)$$

1.3.3 Shearing

Shearing operations belong to affine transformations and are achieved by non-zero off-diagonal elements in the upper 3 by 3 submatrix. Shears are, however, not used in BrainVoyagers standard spatial transformation, which corresponds to pure rigid body transformations (rotations and translations) plus scaling for matching different voxel sizes between data sets.

1.3.4 Rotations

Rotations represent the last elementary 3D transformation, which are the most important ones in the present context. It is common to specify arbitrary rotations with a sequence of simpler ones each along one of three coordinate axes. In each case, the rotation is through an angle, about the given axis. The following three matrices R_x, R_y and R_z and represent transformations that rotate points through the angle b in radians about the coordinate origin:

$$R_x(b) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(b) & -sin(b) & 0 \\ 0 & sin(b) & cos(b) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (8)$$

$$R_y(b) = \begin{pmatrix} cos(b) & 0 & sin(b) & 0 \\ 0 & 1 & 0 & 0 \\ sin(b) & 0 & cos(b) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (9)$$

$$R_z(b) \begin{pmatrix} cos(b) & -sin(b) & & 0 \\ sin(b) & cos(b) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (10)$$

It must be further defined whether positive angles perform a clockwise (CW) or counterclockwise (CCW) rotation around an axis with respect to a specification of the orientation of the axis. In BrainVoyager QX, positive rotation angles cause a counterclockwise rotation about an axis as one looks inward from a point on the positive axis toward the origin. This is commonly the case for right-handed coordinate systems as used in BrainVoyager.

1.4 Composing 3D affine transformations

An important property of affine transformations is that they can be composed, and the result is another 3D affine transformation. A single matrix an be set up for any sequence of transformations as a composite transformation matrix. Forming products of transformation matrices is often referred to as a concatenation, or composition of matrices. For column-matrix representation of coordinate positions, we form composite transformations by multiplying matrices in order from right to left. That is, each successive transformation matrix premultiplies the product of the preceding transformation matrices. The matrix that represents the overall transformation is the product of the individual matrices M_1 and M_2 that perform the two transformations, with M_2 premultiplying M_1:

$$M = M_2 M_1 \tag{11}$$

Any number of affine transformations can be composed in this way, and a single matrix results that represents the overall transformation. This composite matrix can then be applied to any point (column vector) as usual, i.e. $u = Mv$.

NOTE 1 Matrix multiplication is associative. For any three matrices, A, B, and C, the matrix product ABC can be performed by first multiplying A and B or by first multiplying B and $C : ABC = (AB)C = A(BC)$. Therefore we can evaluate matrix products using either a left-to-right or a right-to-left associative grouping. The important point is that matrix multiplication is not commutative in general: The matrix product AB ("A premultiplies B") is generally not the equal to BA ("B premultiplies A"). This means that if a sequence of translations, rotations and scalings is applied, the order in which the elementary transformation matrices appear is critical to determine the overall transformation. Only for some special cases, such as a sequence of transformations all of the same kind (i.e. two translations or two rotations around the same axis), the multiplications of transformation matrices is commutative.

NOTE 2 The "right-to-left" order of transformation matrices holds for column-matrix representations as used in this text. In this representation, points such as u and v are represented as column vectors. Another convention being used in the literature is row-matrix representation in which points are represented as row vectors. A conversion between these conventions is easy by exploiting a property of matrix transposition: The transposition of a matrix product is equivalent to the product of the transposition of each matrix, with the order of multiplication reversed: $(AB)^T = B^T A^T$. Thus, the transformation of vector **v** in columnar-matrix representation $u = M_2 M_1 v$ equals $u = v^T M_1^T M_2^T$ in row-matrix representation.

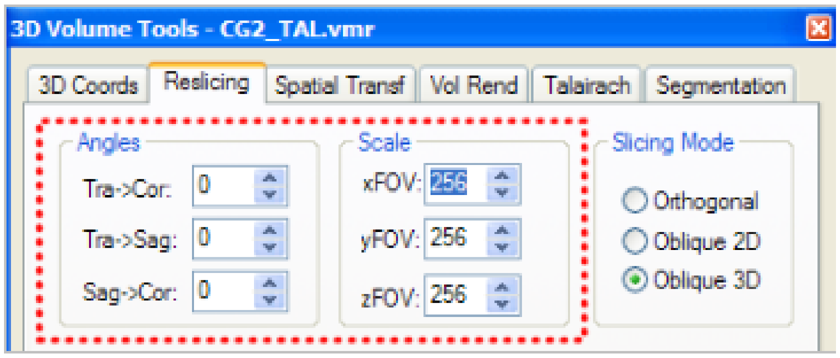
1.5 The order of rotations in BrainVoyager

Since translations commute, the order of applying displacements along the three axes does not matter. The order of rotations about the three coordinate axes, however, is critical since rotations are not commutative. The default order of rotations in BrainVoyager is:

1. Rotation around Y_{SYS} axis (X_{BV} axis)
2. Rotation around Z_{SYS} axis (Y_{BV} axis)
3. Rotation around X_{SYS} axis (Z_{BV} axis)

If three non-zero angles are supplied, BrainVoyager performs first the rotation about the Y_{SYS} axis (X_{BV} axis), then about the Z_{SYS} axis (Y_{BV}) and finally about the X_{SYS} axis (Z_{BV}). This order was defined in a "natural order" (X_{BV}, Y_{BV},

Z_{BV}) with respect to the internal axes definition, but appears arbitrary with respect to the system coordinates. In BrainVoyager QX and BrainVoyager 5.x, the order of axes rotation can now be specified in the new TRF file format (see below). Because BrainVoyager was developed initially in the context of data from Siemens scanners, the rotation about the coordinate axes does also appear in Siemens terminology in the user interface, especially in the Angles field of the Reslicing tab of the 3D Volume Tools dialog (see red rectangle in the figure below).



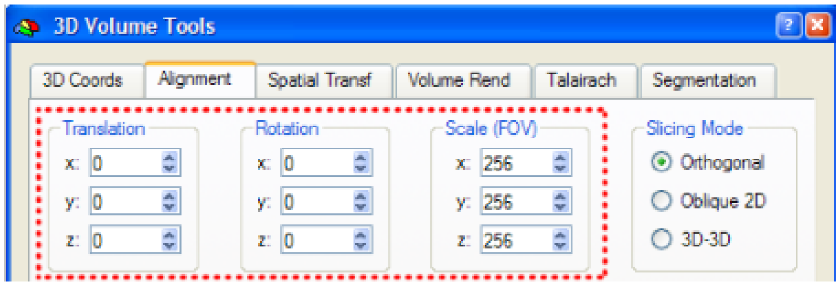
Angles
Tra->Cor: 0
Tra->Sag: 0
Sag->Cor: 0

Scale
xFOV: 256
yFOV: 256
zFOV: 256

Slicing Mode
Orthogonal
Oblique 2D
[✓ Oblique 3D]

Figure 4: Rotations in BrainVoyager 2000

The $Tra \rightarrow Cor$ angle corresponds to rotation about the X axis, the $Tra \rightarrow Sag$ angle corresponds to rotation about the Y axis, and the $Sag \rightarrow Cor'$ angle corresponds to rotation about the Z axis. In BrainVoyager QX, this Siemens terminology is no longer used and is replaced by standard transformation labels (see figure below). The rotation axes are now denoted as "x", "y", "z", corresponding to the X_{SYS} (Z_{BV}), Y_{SYS} (X_{BV}) and Z_{SYS} (Y_{BV}) axes:



Translation
x: 0
y: 0
z: 0

Rotation
x: 0
y: 0
z: 0

Scale (FOV)
x: 256
y: 256
z: 256

Slicing Mode
[✓ Orthogonal]
Oblique 2D
3D-3D

Figure 5: Rotations in BrainVoyager QX

The scaling parameter can be specified now either as Field-Of-View units (millimeter) or as standard scaling parameters. A FOV value of 256 corresponds to a scale value of "1.0".
For a complete specification of a rotation, we must specify a rotation angle and the position of the *rotation point* (or *pivot point*) about which the data set is to be rotated. The default coordinates for the rotation point is the center of the 3D data set, i.e. D/2 with D equal to the number of voxels in the respective dimension. In a 256 by 256 by 256 data set, the rotation point would be thus 128, 128, 128. If translation parameters are specified, the rotation point changes accordingly because the translation is performed prior to the rotation. In BrainVoyager, the translation/rotation point is defined as the coordinates of the current position of the red cross. In BrainVoyager QX, the position of the red cross and the x, y, z translation

values are separated (see Translation fields in figure **??**). This separation of the translation parameters from the position of the red cross in BrainVoyager QX has the advantage that a spatial transformation can be specified while it is still possible to "browse" the data set.

Note. The default rotation point is not the exact center of the data set, which would be for the x axis: $XC = (DX - 1.0)/2.0$. With a dimension of 256 voxels, the center would be $XC = 127.5$. Since this would, however, put the rotation point at a non-integral (non-visible, intermediate) point, the $D/2$ definition is used for the default rotation point. For scaling operations, however, the default fixed point (the point which remains unchanged) is the true center of the data set, $(D - 1.0)/2.0$. Scaling is used to match the voxel resolution of different data sets, i.e. during FMR-VMR coregistration.

1.5.1 Decomposition of a rotation matrix into Euler angles

As described above, a complex affine transformation can be constructed by composing a number of elementary ones. We can also ask the opposite question and ask, what elementary operations "reside in" a given affine transformation matrix? Unfortunately, this problem has not a unique solution since a matrix M may be factored into a product of elementary matrices in various ways. There are, for example, many ways to combine basic rotations to achieve the same composite rotation. In the following, we assume that we have a matrix representing only translation, rotation and scaling transformations.

The three translation values are easy to extract, they are simply the three upper values of the fourth column

$Tx = m_{14}$
$Ty = m_{24}$
$Tz = m_{34}$

The scaling factors are then extracted as:

Finally the rotations are extracted as follows:

```
$Ry = asin((-row[0].z)$
**if** $(cos(y)! = 0)$ **then**
    $Rx = atan2(row[1].z, row[2].z)$
    $Rz = atan2(row[0].y, row[0].x)$
**else**
    $Rx = atan2((row[1].x, row[1].y)$
    $Rz = 0$
**end if**
```

1.6 The TRF file format for spatial transformations

Spatial transformations are saved in "TRF" files in BrainVoyager. These plain text files do not contain a 4x4 matrix but save translation, rotation and scale values separately for each axis. This choice has been made solely to allow for easy readability. If a TRF file is applied, a respective 4x4 matrix is internally constructed from the individual values. Transformation matrices from multiple TRF files are also internally multiplied as detailed above. This happens, for example, during VTC creation combining two TRF files, one for FMR-VMR and one for VMR-VMR (ACPC) transformation. BrainVoyager QX will also support the explicit combination of multiple TRF files as well as the composition and decomposition of homogeneous $4x4$ matrices.
Version 3 is the latest version of this file format introduced with BrainVoyager QX

and also supported in BrainVoyager 5.x. The new format allows to explicitly specify the order of rotation while the old format supported only the implicit order: Y_{SYS}, Z_{SYS}, X_{SYS}. A typical TRF file used to look like this:

```
FileVersion:        3

xTranslation:       0
yTranslation:       8
zTranslation:       14

xRotation:          -14
yRotation:          1
zRotation:          -1

xScaleAsFoV:        256
yScaleAsFoV:        256
zScaleAsFoV:        256

OrderOfRotations:   XYZ

TransformationType: 2
CoordinateSystem:   1
```

while in the newer BrainVoyager QX versions the parameters are provided in the form of a transformation matrix in the TRF file:

```
FileVersion:        5

DataFormat:         Matrix

  0.0000010660081671   0.9786220788955688  -0.2056666463613510   4.3583703041076660
 -0.0019511014688760   0.2056662589311600   0.9786202311515808  -9.4430999755859375
  0.9999980926513672   0.0004002332862001   0.0019096103496850   1.4527800083160400
  0.0000000000000000   0.0000000000000000   0.0000000000000000   1.0000000000000000

TransformationType: 1
CoordinateSystem:   1

NSlicesFMRVMR:      20
SlThickFMRVMR:      3.5
SlGapFMRVMR:        0
CreateFMR3DMethod:  3
AlignmentStep:      1

ExtraVMRTransf:     0

SourceFile:         "C:/Data//fmr/series-0005.fmr"
TargetFile:         "C:/Data/vmr/series-0003.vmr"
```

The file shown below is an example of an initial alignment transformation file (`*_IA.trf`), that registers a functional file (*.fmr) to an anatomical file (*.vmr).

1.7 Summary of coordinate systems

Summary of axes systems in BrainVoyager QX:

1. Internal coordinates. Origin at voxel (0, 0, 0).
   X_{BV}: anterior $\rightarrow$ posterior
   Y_{BV}: superior $\rightarrow$ inferior
   Z_{BV}: right $\rightarrow$ left

2. System coordinates. Origin, directions/values are defined the same as the internal coordinate system but axes names follow Talairach standard:
   X_{SYS}: right $\rightarrow$ left
   Y_{SYS}: anterior $\rightarrow$ posterior
   Z_{SYS}: superior $\rightarrow$ left

3. Talairach coordinates. Axes names like in system coordinates but opposite directions, origin in AC (128,128,128), values defined according to 8 landmarks (AC, PC, LP, RP, SP, IP, AP, PP).
   X_{TAL}: left $\rightarrow$ right
   Y_{TAL}: posterior $\rightarrow$ anterior
   Z_{TAL}: superior $\rightarrow$ left

4. OpenGL coordinates. Like internal (but also shown as system coordinates to the user, except small axes cross in left lower corner of OpenGL (surface) window.

Rotations CCW when looking along positive (OpenGL) axis to origin IN OPENGL. With respect to real Tal axes, the opposite holds. Rot X and Z change sign in VMR.

Fiber coordinates are supported as “BV” or “TAL”.

2 On positioning terminology

This section has been added for people with an interest to learn more about the positioning terminology in image processing and how positioning information is represented in the different coordinate systems of BrainVoyager, DICOM and NIfTI-1. It is not necessary to read this section in order to be able to use the converter.
To indicate the position of an image in a coordinate system, one can use orientation and position vectors. The *orientation vectors* indicate how much an image is rotated with respect to its coordinate system, and the *position vector* indicates the location of the image with respect to the origin.
The information provided in this section has been summarized via image **??**.

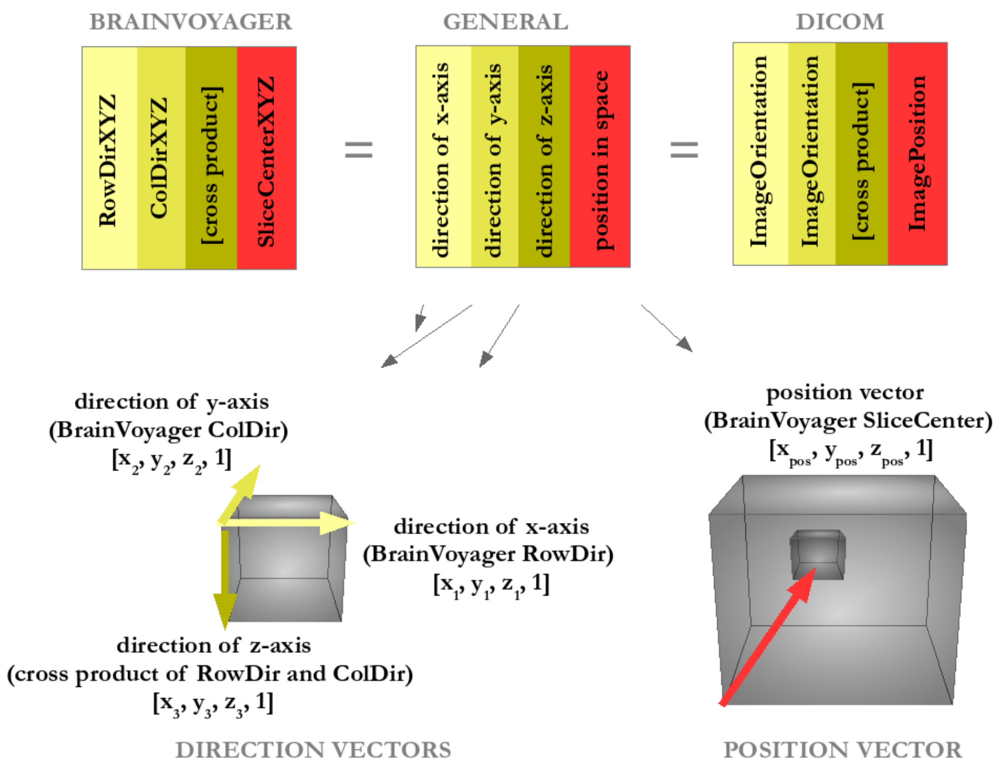
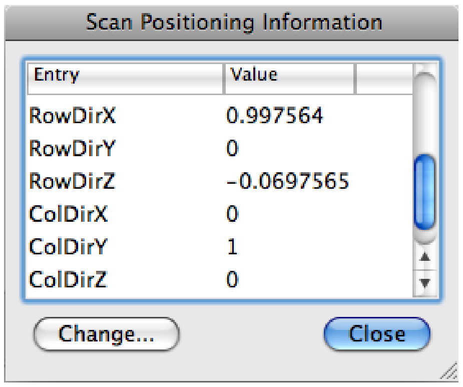


Figure 6: Positioning terminology and depiction of the orientation and position vectors

2.1 Orientation of the image

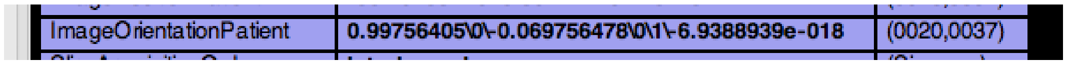
There are three orientation vectors (see left side of figure **??**). The rotation of the x-axis of the image with respect to the coordinate system is one vector. The second vector is the rotation of the y-axis of the image with respect to the y-axis of the coordinate system. The third vector is the rotation of the z-axis of the image with respect to its coordinate system. The rotations are indicated in radians. These vectors that indicate the orientation of the image can also be called *direction vectors*. In BrainVoyager, these direction vectors can be found under the name "RowDir" for the x-axis vector and "ColDir" for the vector that indicates the orientation of the y-axis (see figure **??**). The vector for the z-axis can be computed by taking the cross product of the first two vectors. This results in a normal vector.



Entry	Value
RowDirX	0.997564
RowDirY	0
RowDirZ	-0.0697565
ColDirX	0
ColDirY	1
ColDirZ	0

Figure 7: Orientation vectors in the FMR header

In DICOM, these orientation vectors can be found in the tag ImageOrientation-Patient (0020, 0037). The first three values represent the vector for the x-axis; in figure **??** these are the values [0.99756405, 0, -0.069756478]. The fourth, fifth and sixth values indicate the vector for the orientation of the y-axis; in figure **??** these values are [0, 1, -6.9388939e-018], values which can be rounded to [0, 1, 0].



ImageOrientationPatient	0.99756405\0\-0.069756478\0\1\-6.9388939e-018	(0020,0037)

Figure 8: Orientation vectors in the DICOM header tag (0020, 0037)

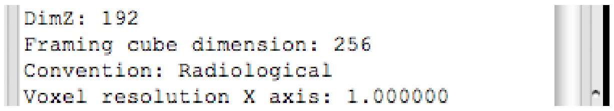
When taking the cross product, the vector for the z-axis can be computed. The cross product of [0.99756405, 0, -0.069756478] and [0,1,0] turns out to be [0.0698, 0, 0.9976]. This is the vector that indicates the orientation of the z-axis of the image and can be placed in the third column of the positioning matrix (see upper part of figure **??**).

2.2 Orientation: left- or righthandedness, improper rotations or radiological vs. neurological convention

If the determinant of the transformation matrix is -1 (instead of 1) then the orientation of the image will be reversed. This means that on some axis, the image is reflected in a plane.
With Matlab it is easy to calculate the determinant (with the `det()` function?), for

BrainVoyager we will provide a JavaScript (*.js) to test this. However, in BrainVoyager itself one can check the orientation according to the descriptions below. In the header of a NIfTI image, this is also indicated (see description below).

The "handedness" of an image can be checked by running the "Obtain image information" function, see chapter **??**.
In the BrainVoyager anatomical data format, a header field `RadiologicalConvention` is available, which indicates that the image is in right-handed position (see figure **??**).



```
DimZ: 192
Framing cube dimension: 256
Convention: Radiological
Voxel resolution X axis: 1.000000
```

Figure 9: The handedness of a *.vmr file printed to the BrainVoyager QX Log tab

If necessary, change the handedness by flipping the X-axis. This can be performed via the BrainVoyager QX function "Flip X-axis" on the "VMR Properties" dialog (see figure **??**).

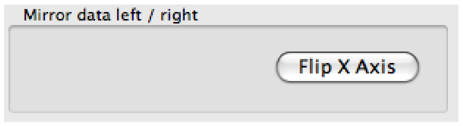
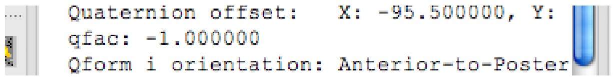


Figure 10: Change the handedness of a *.vmr file by flipping the X-axis

In NIfTI-1 images, the information is stored in the "qfac" parameter, in `pixdim[0]`. If this parameter is -1, this indicates that an improper rotation will be applied, so that the orientation will be reversed, whatever it originally was (see figure **??**).



```
Quaternion offset:   X: -95.500000, Y:
qfac: -1.000000
Qform i orientation: Anterior-to-Poster
```

Figure 11: The handedness of a *.nii/*.hdr file printed to the BrainVoyager QX Log tab

Applying reflections To change the orientation of an image, one could apply a reflection to the volume (for 4D images, this transformation will be applied to each single volume).

Reflect volume in a plane, x-axis:

$$\begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Sometimes this matrix can be applied to flip the left and right of a VMR image.

Reflect volume in a plane, y-axis:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Reflect volume in a plane, z-axis:

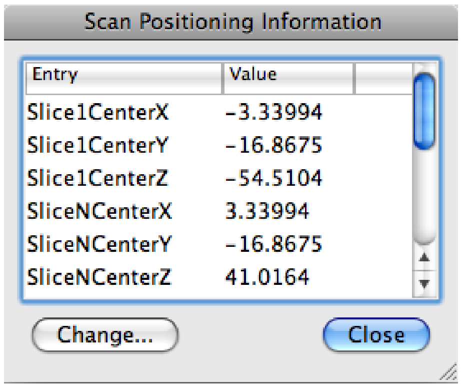
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

This is the matrix to change left-right for NIfTI images in NIfTI coordinate system.
(See http://nifti.nimh.nih.gov/nifti-1/documentation/nifti1fields/nifti1fields$_{p}ages/qsform.html$.)

Any of the reflections can be combined with any of the rotations, or reflections with reflections, by matrix multiplication.

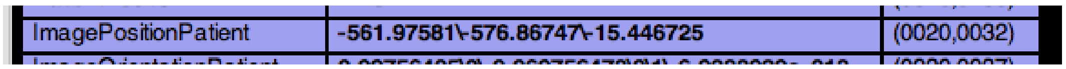
2.3 Position of the image

There is one position vector (see right side of figure **??**). This vector contains the vector from the origin (0,0,0) to the first voxel of the image. In BrainVoyager, this vector can be found with the name SliceCenter1X/Y/Z and SliceCenterNX/Y/Z (see figure **??**). In the DICOM header, this can be found in tag (0020, 0032). When creating a project in BrainVoyager, this information is also depicted in the "Info" tab (see figure **??**).



Entry	Value
Slice1CenterX	-3.33994
Slice1CenterY	-16.8675
Slice1CenterZ	-54.5104
SliceNCenterX	3.33994
SliceNCenterY	-16.8675
SliceNCenterZ	41.0164

Figure 12: Position vector in the FMR header



ImagePositionPatient	-561.97581\-576.86747\-15.446725	(0020,0032)

Figure 13: Position vector in the DICOM header (tag 0020, 0032) of functional data. In the Siemens version of the DICOM format, all images of one volume are often placed in one file (mosaic). The positioning information is now computed using concatenated image matrices, as if the images of the volume were lying in one plane. The plane is usually square, so an image volume of 36 slices would be placed in a plane with $6x6$ images.

2.4 Orientation and positioning information in a matrix

The positioning information in BrainVoyager anatomical (*vmr) and functional (*.fmr) files and the positioning file (*.pos) can be placed in a positioning matrix in the following way:

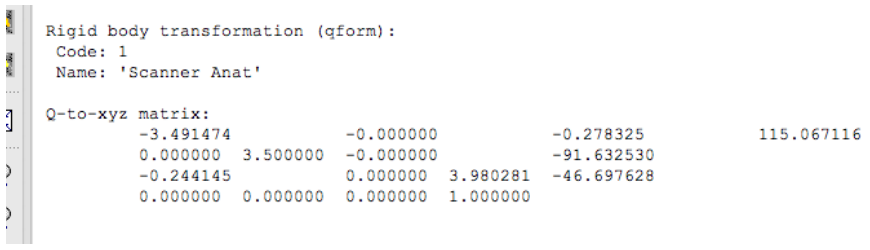
$$world2voxel = \begin{pmatrix} RowDir_x & ColDir_x & normal_x & SliceCenter_x \\ RowDir_y & ColDir_y & normal_y & SliceCenter_y \\ RowDir_z & ColDir_z & normal_z & SliceCenter_z \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (12)$$

This matrix represents the BrainVoyager world-to-voxel transformation, which converts positions of voxels in the DICOM coordinate system (x, y, z) to BrainVoyager voxel indices (i, j, k). To compute the coordinates of a voxel, multiply the inverse of this "world-to-voxel" matrix with the voxel indices (i, j, k) of a point in the image:

$$voxelposition = world2voxel^{-1} * [i, j, k, 1]^T \quad (13)$$

This computation transforms the set of voxels back to a metric space (coordinate system).

In NIfTI, the positioning matrix can be created from the *qform* and *sform* header fields (see figure **??**). Also in this case the first three columns indicate the orientation of the image, while the fourth column indicates the position with respect to the origin (0,0,0).



```
Rigid body transformation (qform):
 Code: 1
 Name: 'Scanner Anat'

Q-to-xyz matrix:
        -3.491474           -0.000000           -0.278325           115.067116
        0.000000  3.500000  -0.000000           -91.632530
        -0.244145           0.000000  3.980281  -46.697628
        0.000000  0.000000  0.000000  1.000000
```

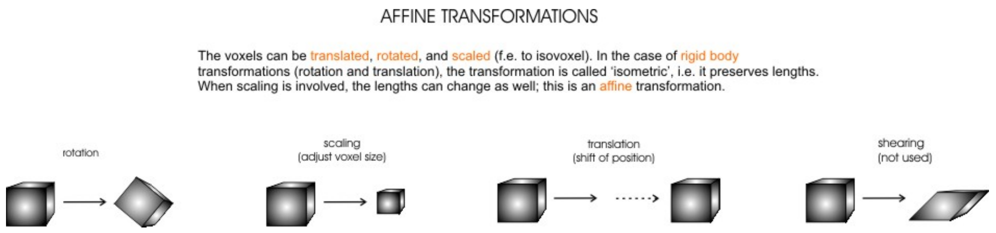
Figure 14: Orientation and position vectors in the NIfTI-1 header

3 Notes on transformations

3.1 Introduction

This section has been added for people with an interest to learn more about the transformations terminology in image processing and how transformation information is represented in the different coordinate systems of BrainVoyager, DICOM and NIfTI-1. It is not necessary to read this section in order to be able to use the converter.

Coordinate changes for a 3-dimensional object in vector space can be classified as *global* or *local*. When the coordinates of an object are changed, this can be represented in a matrix. In global transformations, the same matrix can be applied to each voxel.
For a 3-dimensional brain image, a 4×4 matrix can be used for a global coordinate change. This involves a coordinate change along all axes or some of the axes. Changes can be translations (shifts to a certain direction), rotations, scaling (change of size, sometimes referred to as zooms), shears (skewness towards one direction) and perspective transformations (vanishing horizon in a single point) (see figure **??**).


AFFINE TRANSFORMATIONS

The voxels can be translated, rotated, and scaled (f.e. to isovoxel). In the case of rigid body transformations (rotation and translation), the transformation is called 'isometric', i.e. it preserves lengths. When scaling is involved, the lengths can change as well; this is an affine transformation.

Figure 15: There are four types of actions in affine transformations

In local, or non-linear, coordinate changes, different transformations are applied to certain voxels. This is the case for the Talairach transformation in BrainVoyager, where different matrices are applied for different sub-volumes. This is sometimes called 'piecewise affine', since the matrix is still a 4×4 matrix.
An example of an affine transformation is the initial alignment in BrainVoyager. In this transformation, the position of the functional image (*.fmr) is transformed to the position of the anatomical image (*.vmr). To achieve this, the inverse of the positioning matrix of the anatomical image is multiplied with the positioning matrix of the functional image (see figure **??**).

The resulting matrix is stored in a transformation file (`*_IA.trf`). In figure **??** is shown that the shift or translation of the image can be found in the fourth column of the transformation matrix. The other parameters are shown in appendix **??**.

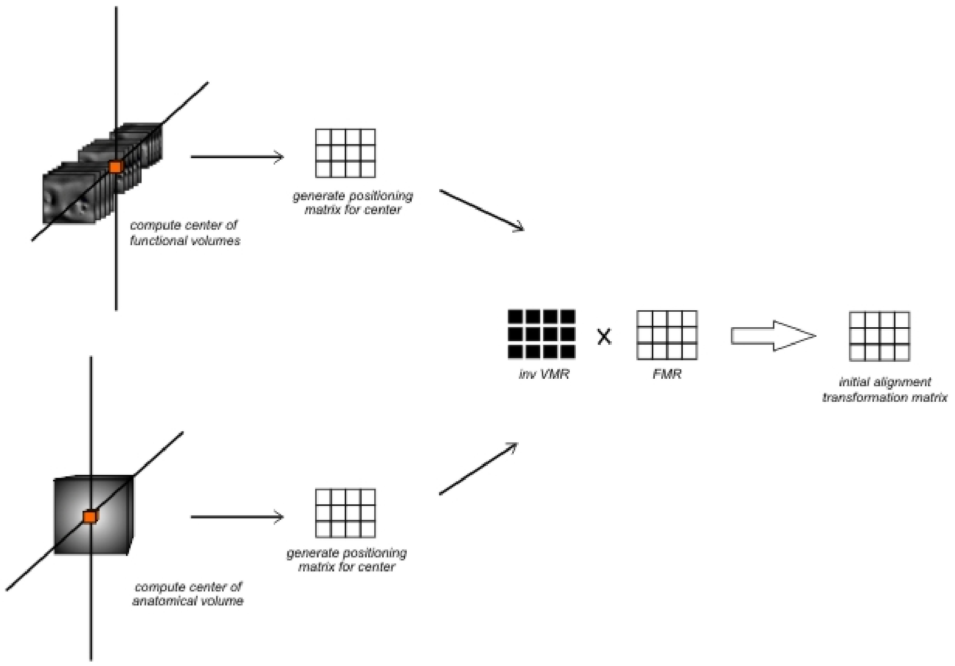
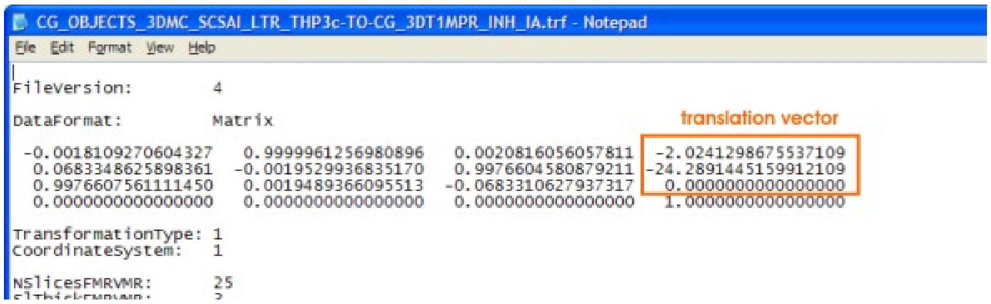

compute center of functional volumes
generate positioning matrix for center
inv VMR
FMR
initial alignment transformation matrix
compute center of anatomical volume
generate positioning matrix for center

Figure 16: Transformation from FMR to VMR (please note that the drawing of the center on the FMR volume should be on the first volume only)


CG_OBJECTS_3DMC_SCSAI_LTR_THP3c-TO-CG_3DT1MPR_INH_IA.trf - Notepad
File Edit Format View Help
translation vector

```
FileVersion:        4

DataFormat:         Matrix

 -0.0018109270604327   0.9999961256980896   0.0020816056057811  -2.0241298675537109
  0.0683348625898361  -0.0019529936835170   0.9976604580879211 -24.2891445159912109
  0.9976607561111450   0.0019489366095513  -0.0683310627937317   0.0000000000000000
  0.0000000000000000   0.0000000000000000   0.0000000000000000   1.0000000000000000

TransformationType: 1
CoordinateSystem:   1

NSlicesFMRVMR:      25
```

Figure 17: The translation vector can be found in the fourth column of a transformation matrix

3.2 Storage of transformation information

While the original position of the image is stored in the image in the cases of BrainVoyager image data (anatomical (*.vmr), functional (*.fmr), diffusion weighed (*.dmr)), each transformation being applied to the image is also stored in some way.
The storage of transformations for volumetric data in BrainVoyager is summarized in table **??**. Please note that for the piecewise affine transformation to Talairach space only the landmarks are saved.

type	internal storage	external storage
position (scanner)	VMR Properties	*.pos
rigid body transformation	VMR Properties	*.trf
affine transformation (IA, FA)	VMR Properties	*.trf
Talairach landmarks	VMR Properties	*.tal

Table 1: Storage of position and transformation information in BrainVoyager files

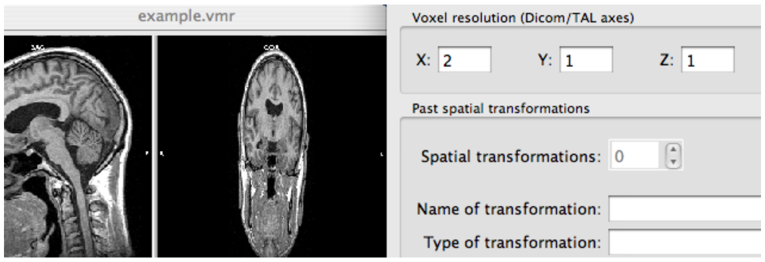
The storage for volumetric data in NIfTI is summarized in table **??**.

type	internal storage	external storage
position	qform	n/a
rigid body transformation	qform	n/a
affine transformation	sform	n/a
Talairach transformation	no	n/a

Table 2: Storage of position and transformation information in NIfTI files

3.3 Differences in storage of transformations

There is a difference between how the transformations are stored in NIfTI-1 images and how the transformations are stored in BrainVoyager image data.
In BrainVoyager image files, the original position of the image in a BrainVoyager flavour of DICOM coordinates is preserved (see the figures **??** and **??**). All subsequent transformations are also preserved (see figure **??**). The current position of the image has to be calculated by multiplying these transformations. The fact that the original position in the scanner and the history of transformations is preserved in BrainVoyager image data, provides the user with a high degree of flexibility and a nice logging functionality.



Voxel resolution (Dicom/TAL axes)

X: 2 Y: 1 Z: 1

Past spatial transformations

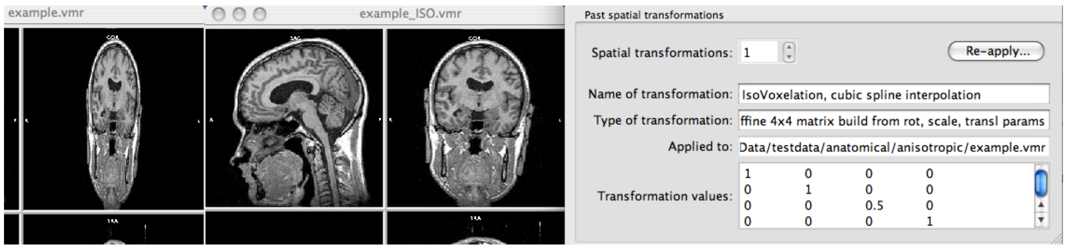
Spatial transformations: 0

Name of transformation:

Type of transformation:

Figure 18: Before the transformation: anisotropic voxel sizes in the VMR Properties

The image properties, for example voxel sizes, can be inspected via the 'VMR Properties' of the BrainVoyager 'File' menu (see figure **??**). As in shown in figure **??**, transformations are also saved in the 'VMR Properties'.



Past spatial transformations

Spatial transformations: 1

Name of transformation: IsoVoxelation, cubic spline interpolation

Type of transformation: ffine 4x4 matrix build from rot, scale, transl params

Applied to: Data/testdata/anatomical/anisotropic/example.vmr

Transformation values:

```
1    0    0    0
0    1    0    0
0    0    0.5  0
0    0    0    1
```

Figure 19: After the transformation: storage in the VMR Properties

The matrix in figure **??** makes clear that the transformation is applied to BrainVoyager internal coordinates, because the anisotropic voxel size being reported in the VMR Properties is X, while in the matrix the size for Z is set to 0.5. This indicates that the x-axis in BrainVoyager system coordinates is z-axis in the BrainVoyager internal coordinate system (see appendix **??**).
In NIfTI images, the positioning information is always updated to the newest transformation. Since there is only place for storing two transformations, in the *qform* and *sform* fields, this might not make it possible to retrieve the native position as the image was acquired.

Resulting transformations for VMR

In the paragraphs below, the effects of rotating over axes of a standard coordinate system (Talairach, NIfTI) on the images in BrainVoyager are shown.

Rotating over x-axis of an image

In a VMR, the x-axis is the anterior-posterior (sagittal) axis. So the effect of rotating over the x-axis on TAL-VMR is that the head rotates leftward (counter-clockwise as seen from the back of the head) over the anterior-posterior (sagittal) axis (see COR window in figure **??**).

rotation over x-axis	0π: $0°$	0.5π: $90°$	π: $180°$	1.5π: $270°$
$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & -0 \\ 0 & 0 & -1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix}$

Table 3: Rotation matrices for rotation over x-axis

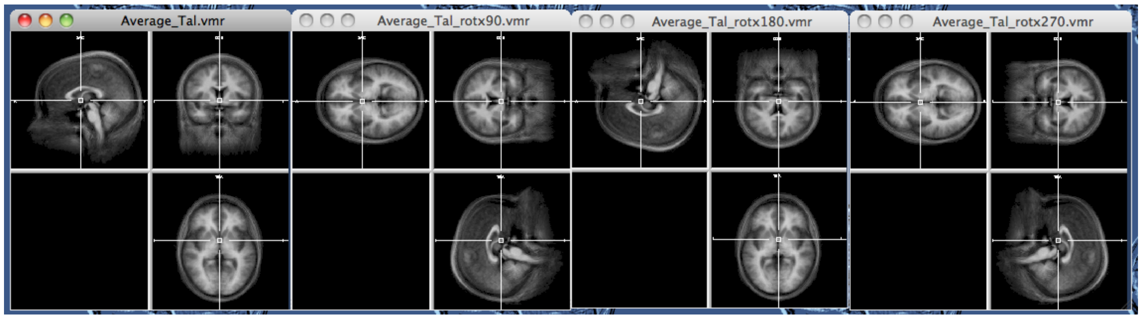


Figure 20: Effect of rotation matrices on x-axis of coordinate system on VMR image

Rotating over y-axis of an image

In a VMR, the y-axis is the inferior-superior (axial) axis. So the effect of rotating over the y-axis on TAL-VMR is that the head rotates leftward (counter-clockwise as seen from the top of the head) over superior-inferior (axial) axis (see TRA window in figure **??**).

$$\begin{array}{ccccc}
\text{rotation over y-axis} & 0\pi\text{: } 0° & 0.5\pi\text{: } 90° & \pi\text{: } 180° & 1.5\pi\text{: } 270° \\
\begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix} &
\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} &
\begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} &
\begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} &
\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}
\end{array}$$

Table 4: Rotation matrices for rotation over y-axis

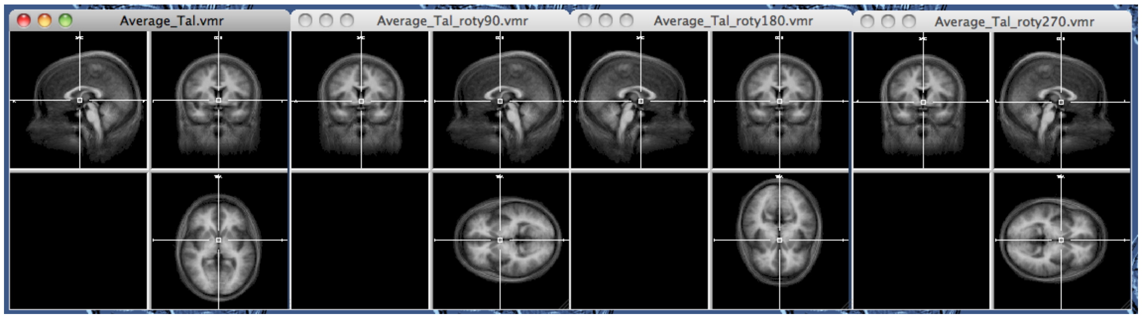


Figure 21: Effect of rotation matrices on y-axis of coordinate system on VMR image

Note: some use the following matrix for rotations over the y-axis [**?**]: $\begin{matrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{matrix}$

Rotating over z-axis of an image

In a VMR, the z-axis is the left-right (coronal) axis. So the effect of rotating over the z-axis on TAL-VMR is that the rotates backward (clockwise as seen from the left side of the head) over left-right axis (see SAG window in figure **??**).

rotation over z-axis	0π: 0°	0.5π: 90°	π: 180°	1.5π: 270°
$\begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Table 5: Rotation matrices for rotation over z-axis

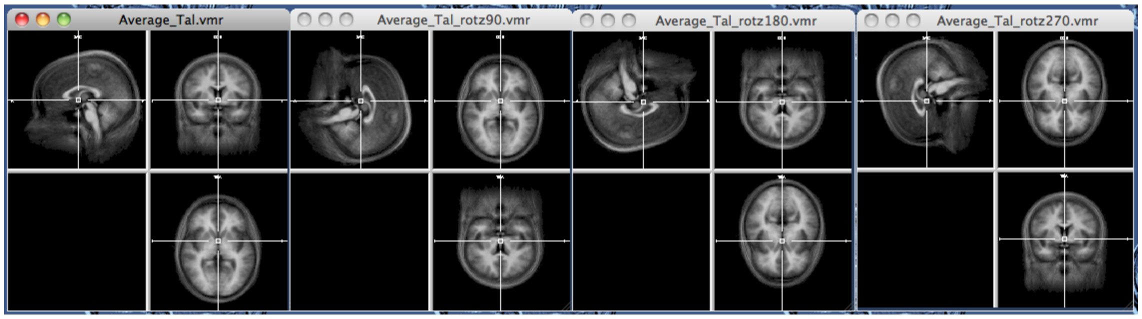


Figure 22: Effect of rotation matrices on z-axis of coordinate system on VMR image