Volumetric Image Visualization

Alexandre Xavier Falcão

LIDS - Institute of Computing - UNICAMP

afalcao@ic.unicamp.br

- A **geometric transformation** is a function applied to a set of points (e.g., spels or any subset of $\Re^3$) or a vector.

Geometric Transformations

- A **geometric transformation** is a function applied to a set of points (e.g., spels or any subset of $\Re^3$) or a vector.
- Such a set of points may contain elements of a plane or a line segment, and we are often interested in estimating the image intensities at them, whenever they fall in some image domain.

Geometric Transformations

- A **geometric transformation** is a function applied to a set of points (e.g., spels or any subset of $\Re^3$) or a vector.

- Such a set of points may contain elements of a plane or a line segment, and we are often interested in estimating the image intensities at them, whenever they fall in some image domain.

- We are interested in affine transformations in 3D and projections from 3D to 2D:
  - Translation and scaling;
  - Rotation around one of the axis x, y, or z, with respect to the origin of a coordinate system;
  - Rotation around an arbitrary axis with respect to an abitrary point; and
  - Orthogonal projection.

Translation

Let $\mathbf{T}(t_x, t_y, t_z)$ be a translation that moves a point from $p = (x_p, y_p, z_p)$ to $q = (x_q, y_q, z_q)$ by using a displacement vector $\mathbf{t} = (t_x, t_y, t_z)$.

$$\begin{bmatrix} x_q \\ y_q \\ z_q \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} + \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix}$$

Translation

Let $\mathbf{T}(t_x, t_y, t_z)$ be a translation that moves a point from $p = (x_p, y_p, z_p)$ to $q = (x_q, y_q, z_q)$ by using a displacement vector $\mathbf{t} = (t_x, t_y, t_z)$.

$$\begin{bmatrix} x_q \\ y_q \\ z_q \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} + \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix}$$

Such an additive form of describing a translation is inconvenient, but the use of **homogeneous coordinates** can fix that and allow to describe and combine multiple geometric transformations by matrix multiplications.

Translation

In **homogeneous coordinates**, the points p and q may be written as $(x_p, y_p, z_p, 1)$ and $(x_q, y_q, z_q, 1)$, respectively, and $\mathbf{T}(t_x, t_y, t_z)$ becomes a multiplicative translation matrix.

$$\begin{bmatrix} x_q \\ y_q \\ z_q \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

Translation

In **homogeneous coordinates**, the points p and q may be written as $(x_p, y_p, z_p, 1)$ and $(x_q, y_q, z_q, 1)$, respectively, and $\mathbf{T}(t_x, t_y, t_z)$ becomes a multiplicative translation matrix.

$$\begin{bmatrix} x_q \\ y_q \\ z_q \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

The inverse simply requires the translation matrix $\mathbf{T}(-t_x, -t_y, -t_z)$ in homogeneous coordinates.

Scaling

One can increase/decrease the size of a geometric structure (e.g., a cube) by scaling the coordinates of its vertices. Let $\mathbf{S}(s_x, s_y, s_z)$ be the scaling matrix, such a transformation is described by

$$\begin{bmatrix} x_q \\ y_q \\ z_q \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

Scaling

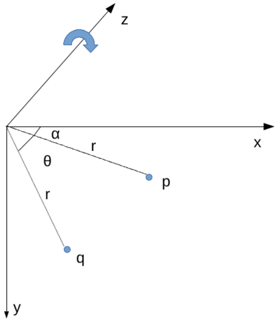
One can increase/decrease the size of a geometric structure (e.g., a cube) by scaling the coordinates of its vertices. Let $\mathbf{S}(s_x, s_y, s_z)$ be the scaling matrix, such a transformation is described by

$$\begin{bmatrix} x_q \\ y_q \\ z_q \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

- When applied to spels of an image domain, it changes the image size, but intensities must be estimated at the new spels.
- Reduction in size requires factors in $(0, 1)$, factors greater than 1 increase the size, and negative factors reflect it along the corresponding axis with respect to the origin.
- The inverse is $\mathbf{S}(\frac{1}{s_x}, \frac{1}{s_y}, \frac{1}{s_z})$.

Rotation around axis z with respect to the origin

Let $\mathbf{v}_z = (0, 0, 1, 0)$ be a unit vector along z and $(0, 0, 0, 1)$ be the origin, a rotation θ around axis z with respect to $(0, 0, 0, 1)$



changes only the x and y coordinates when mapping a point $p = (x_p, y_p, z_p, 1)$ into a new point $q = (x_q, y_q, z_q, 1)$ by following the right-hand rule — right hand with the thumb out-stretched along z, the index finger along x, and the middle finger along y, movement from x to y of an angle θ.

Rotation around axis z with respect to the origin

This rotation is represented by a matrix $\mathbf{R}_z(\theta)$ derived as follows.

$$
\begin{aligned}
x_p &= r\cos(\alpha) \\
y_p &= r\sin(\alpha) \\
x_q &= r\cos(\theta + \alpha) \\
x_q &= r\cos(\alpha)\cos(\theta) - r\sin(\alpha)\sin(\theta) \\
x_q &= x_p\cos(\theta) - y_p\sin(\theta) \\
y_q &= r\sin(\theta + \alpha) \\
y_q &= r\cos(\alpha)\sin(\theta) + r\sin(\alpha)\cos(\theta) \\
y_q &= x_p\sin(\theta) + y_p\cos(\theta) \\
z_q &= z_p
\end{aligned}
$$

Rotation around axis z with respect to the origin

$$\begin{bmatrix} x_q \\ y_q \\ z_q \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

Rotations around x and y with respect to the origin

Similarly, the rotation matrices around x and y with respect to the origin are given by

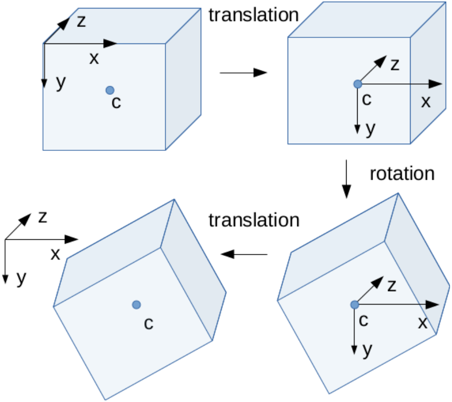
$$\mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Note only that the signs of the sines are interchanged in $\mathbf{R}_y(\theta)$, because this rotation is contrary to the right-hand rule.

Rotations around the main axes with respect to an arbitrary point

In order to rotate around the main axes with respect to an arbitrary point c (e.g., the center of the image domain), the origin must be translated to c, the rotation applied, and then the origin translated back to a position that avoids clipping.



Rotations around the main axes with respect to an arbitrary point

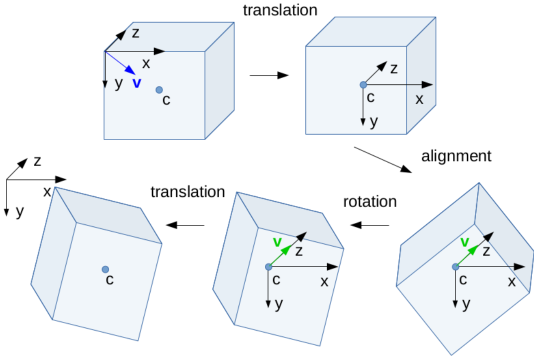
For instance, a rotation $\mathbf{R}_x(\theta)$ (tilt) followed by $\mathbf{R}_y(\alpha)$ (spin) with respect to the center $c = (x_c, y_c, z_c, 1)$ of an image domain D_I is

$$\begin{bmatrix} x_q \\ y_q \\ z_q \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & d/2 \\ 0 & 1 & 0 & d/2 \\ 0 & 0 & 1 & d/2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\alpha) & 0 & \sin(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_c \\ 0 & 1 & 0 & -y_c \\ 0 & 0 & 1 & -z_c \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

where d is the diagonal of D_I.

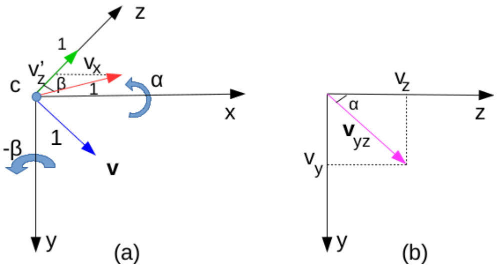
Rotation around an arbitrary axis with respect to an arbitrary point

In order to rotate θ around an arbitrary axis $\mathbf{v} = (v_x, v_y, v_z, 0)$, with respect to an arbitrary point $c = (x_c, y_c, z_c, 1)$, we apply a translation $\mathbf{T}(-x_c, -y_c, -z_c)$, alignment between $\mathbf{v}$ and z, rotation θ, and translation $\mathbf{T}(\frac{d}{2}, \frac{d}{2}, \frac{d}{2})$.



Rotation around an arbitrary axis with respect to an arbitrary point

Given that z is the chosen axis, the alignment of $\mathbf{v}$ with the vector $\mathbf{v}_z = (0, 0, 1, 0)$ requires a rotation α around x followed by a rotation $-\beta$ around y, with respect to the origin c.



(a) The rotation α of $\mathbf{v}$ (blue) stops at the plane xz (red) and the rotation $-\beta$ aligns $\mathbf{v}$ (red) with the axis z (green). (b) The projection $\mathbf{v}_{yz}$ (magenta) of $\mathbf{v}$ (blue) over the plane yz forms the same angle α with the axis z.

Rotation around an arbitrary axis with respect to an arbitrary point

The desired transformation is represented by the following sequence of matrices.

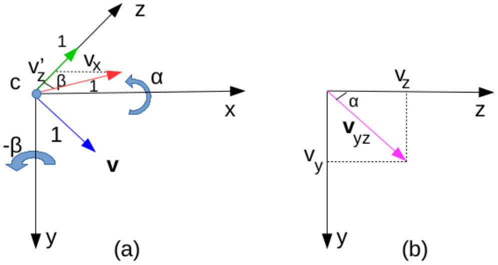
$$\mathbf{T}(\frac{d}{2},\frac{d}{2},\frac{d}{2})\mathbf{R}_x^{-1}(\alpha)\mathbf{R}_y^{-1}(-\beta)\mathbf{R}_z(\theta)\mathbf{R}_y(-\beta)\mathbf{R}_x(\alpha)\mathbf{T}(-x_c,-y_c,-z_c).$$

Given that the inverse of a rotation $\mathbf{R}^{-1}(\theta)=\mathbf{R}^t(\theta)=\mathbf{R}(-\theta)$ and $(\mathbf{R}_1(\alpha)\mathbf{R}_2(\beta))^{-1}=\mathbf{R}_2^{-1}(\beta)\mathbf{R}_1^{-1}(\alpha)=\mathbf{R}_2(-\beta)\mathbf{R}_1(-\alpha)$. We may rewrite the above equation as

$$\mathbf{T}(\frac{d}{2},\frac{d}{2},\frac{d}{2})\mathbf{R}_x(-\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(\theta)\mathbf{R}_y(-\beta)\mathbf{R}_x(\alpha)\mathbf{T}(-x_c,-y_c,-z_c)$$

Rotation around an arbitrary axis with respect to an arbitrary point

The main issue is how to find α and β based on the figure below?



From (b) and (a), we have respectively that

$$\alpha = \tan^{-1}\left(\frac{v_y}{v_z}\right)$$

$$\beta = \tan^{-1}\left(\frac{v_x}{v'_z}\right).$$

Rotation around an arbitrary axis with respect to an arbitrary point

This implies that

$$
\begin{aligned}
\tan\alpha &= \frac{v_y}{v_z},\\
\tan\beta &= \frac{v_x}{v'_z},\\
v_x^2 + v_y^2 + v_z^2 &= 1,\\
v_x^2 + {v'}_z^2 &= 1,
\end{aligned}
$$

Rotation around an arbitrary axis with respect to an arbitrary point

then

$$\begin{aligned} v_x^2 + v_y^2 + v_z^2 &= v_x^2 + {v'}_z^2, \\ v_z^2 \tan^2\alpha + v_z^2 &= {v'}_z^2, \\ v_z^2(1+\tan^2\alpha) &= {v'}_z^2, \\ \frac{v_z^2}{\cos^2\alpha} &= {v'}_z^2, \\ v'_z &= \frac{v_z}{\cos\alpha}. \end{aligned}$$

This is then valid for $v_z > 0$. For $v_z = 0$ and $v_z < 0$, we will have the following rules, left as exercise.

Rotation around an arbitrary axis with respect to an arbitrary point

- If $v_z < 0$, then α and β must be subtracted by 180, whenever they are different of zero.

Rotation around an arbitrary axis with respect to an arbitrary point

- If $v_z < 0$, then α and β must be subtracted by 180, whenever they are different of zero.
- If $v_z = 0$, the above equations need special treatment:
  - if $v_x = 0$ and $v_y \neq 0$, then $\beta = 0$ and $\alpha = 90 sign(v_y)$;
  - if $v_x \neq 0$ and $v_y = 0$, then $\beta = 90 sign(v_x)$ and $\alpha = 0$; and
  - if $v_y \neq 0$ and $v_x \neq 0$, then it is better to apply $\mathbf{R}_z(\gamma)$ with $\gamma = -sign(v_y) \cos^{-1} v_x$ followed by $\mathbf{R}_y(-90)$ to align $\mathbf{v}$ first with the axis x and then next with the axis z.

Rotation around an arbitrary axis with respect to an arbitrary point

- If $v_z < 0$, then α and β must be subtracted by 180, whenever they are different of zero.
- If $v_z = 0$, the above equations need special treatment:
  - if $v_x = 0$ and $v_y \neq 0$, then $\beta = 0$ and $\alpha = 90 sign(v_y)$;
  - if $v_x \neq 0$ and $v_y = 0$, then $\beta = 90 sign(v_x)$ and $\alpha = 0$; and
  - if $v_y \neq 0$ and $v_x \neq 0$, then it is better to apply $\mathbf{R}_z(\gamma)$ with $\gamma = -sign(v_y) \cos^{-1} v_x$ followed by $\mathbf{R}_y(-90)$ to align $\mathbf{v}$ first with the axis x and then next with the axis z.
- Among several applications, an interesting one is the reslicing of a 3D image along an arbitrary axis $\mathbf{v}$.