Closed-form solution to 4x4 inverse transform

Start with a standard cartesian coordinate frame A. Now suppose a second cartesian coordinate frame B is obtained by rotating and then translating frame A, where the rotation is defined by **R** (3x3 rotation matrix) and the translation is defined by **d** (3x1 vector).

Given that a vector ***p*** defines a point in space with respect to coordinate frame B, this same point with respect to coordinate frame A is defined by the vector ***q*** as follows:

$$\boldsymbol{q} = \boldsymbol{Rp} + \boldsymbol{d}$$

This equation for vector q can also obtained by the matrix multiplication

$$\begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{1} \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{1} \end{bmatrix}$$

where T is the 4x4 homogenous transform formed as

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R_{3x3}} & \boldsymbol{d_{3x1}} \\ \boldsymbol{0_{3x3}} & \boldsymbol{1} \end{bmatrix}$$

Using the 4x4 transform is also convenient because if you have ***q*** and want to solve for ***p***, you can use the inverse of the 4x4 transform.

$$\boldsymbol{T^{-1}} \begin{bmatrix} \boldsymbol{q} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix}$$

Inverting ***T*** is often not an issue, but it does require numerical solvers.

As an alternative, you can “invert” ***T*** by realizing that the inverse of the rotation matrix is its transpose, and avoid numerical solvers altogether.

$$\boldsymbol{R^T} = \boldsymbol{R^{-1}}$$

Go back to the first equation and follow these steps.

$$\boldsymbol{q} - \boldsymbol{d} = \boldsymbol{Rp}$$

$$\boldsymbol{R^T}[\boldsymbol{q} - \boldsymbol{d}] = \boldsymbol{R^T R p} = \boldsymbol{p}$$

$$\boldsymbol{R^T q} - \boldsymbol{R^T d} = \boldsymbol{p}$$

And the solution for ***p*** given ***q*** is

$$\begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R^T_{3x3}} & -\boldsymbol{R^T_{3x3} d_{3x1}} \\ \boldsymbol{0_{3x3}} & \boldsymbol{1} \end{bmatrix} \begin{bmatrix} \boldsymbol{q} \\ 1 \end{bmatrix}$$

This is closed-form and avoids all possible numerical errors and is faster than solvers for matrix inverses.

The Matlab function in this repo *Tinv.m* does this for you.