Real Orthogonal Transformations of 3 - space via Mathematica

Barry H Dayton https : // www .barryhdayton . space

The word "transformations" in the title implies that I am looking at orthogonal matrices geometrically. Loosely people refer to these linear transformations as rotations and reflections. An orthogonal trans - formation is even (direct) or odd (opposite) according to whether the matrix of the transformation has determinant 1 or -1. In this note I show by means of a Mathematica procedure that even orthogonal transformations are rotations and give the angle and axis of the rotation. Odd orthogonal transforma - tions are rotary reflections, that is the composition of a reflection and a rotation in an axis perpendicu - lar to the mirror of the reflection. Again I give a procedure finding the axis and angle of rotation. As usual for me, I work numerically.

For even orthogonal transformations I have the following procedure with argument the even orthogo - nal matrix and returning the angle and axis.

```
In[ ]:= EvenOrth2Rotation[U_] := Module[{eigs, v1, v2, b, c, M, R, w},
  If[Det[U] ≠ 1, Echo["Not even, use OddOrth2RR"]; Abort[]];
  eigs = Eigensystem[U];
  If[eigs[[1, 1]] == 1., v1 = Chop[eigs[[2, 1]],
   If[eigs[[1, 2]] == 1., v1 = Chop[eigs[[2, 2]],
    If[eigs[[1, 3]] == 1., v1 = Chop[eigs[[2, 3]], Return["Fail"]]]];
  b = Normalize[RandomReal[{-1, 1}, 3]];
  v2 = Normalize[b - (v1.b) v1];
  w = U.v2;
  c = ArcCos[v2.w];
  R = Chop[RotationMatrix[c, v1]];
  If[Norm[R - U] > .01, c = -c];
  {c, v1}]
```

As a random example let

```
In[ ]:= M1 = {{0.7737717265350375` , -0.6296380711474817` , 0.06952132461817584` },
    {-0.10395959774273329` , -0.01795896138416882` , 0.9944193671400267` },
    {-0.624875761453552` , -0.7766809995536654` , -0.07935305715700698` }};
```

```
In[ ]:= M1 // MatrixForm
```

```
Out[ ]//MatrixForm=
```

$$\begin{pmatrix} 0.773772 & -0.629638 & 0.0695213 \\ -0.10396 & -0.017959 & 0.994419 \\ -0.624876 & -0.776681 & -0.0793531 \end{pmatrix}$$

Note that M is orthogonal of determinant 1.

```
In[ ]:= M1.Transpose[M1]
        Det[M1]
Out[ ]= {{1., 2.08167 × 10^-16, -1.56125 × 10^-17},
         {2.08167 × 10^-16, 1., 9.71445 × 10^-17}, {-1.56125 × 10^-17, 9.71445 × 10^-17, 1.}}

Out[ ]= 1.

In[ ]:= M1data = EvenOrth2Rotation [M1]
Out[ ]= {-1.73328 , {0.89737 , -0.351833 , -0.266347 }}
```

To check, I recover my matrix M1 by

```
In[ ]:= rot1 = RotationMatrix [M1data[[1]], M1data[[2]]]
Out[ ]= {{0.773772 , -0.629638 , 0.0695213 },
         {-0.10396 , -0.017959 , 0.994419 }, {-0.624876 , -0.776681 , -0.0793531 }}

In[ ]:= Norm[M1 - rot1]
Out[ ]= 9.68783 × 10^-16
```

For odd orthogonal transformations one exception is the inversion

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

which is a rotary reflection but can be represented by the composition of any reflection and a half turn in its perpendicular. So the next procedure simply returns “inflection” in this case. Otherwise it returns the angle and axis of the rotation which will match the perpendicular of the reflection.

```
In[ ]:= OddOrth2RR[U_] := Module[{eigs, v1, v2, b, w, ρ, σ},
          If[Det[U] ≠ -1, Echo["Not odd, use EvenOrth2Rotation "]; Abort[]];
          eigs = Eigensystem[U];
          If[eigs[[1, 1]] == -1. && eigs[[1, 2]] == -1. && eigs[[1, 3]] == -1., Return["Inversion "]];
          If[eigs[[1, 1]] == -1.`, v1 = Chop[eigs[[2, 1]]], If[eigs[[1, 2]] == -1.`,
            v1 = Chop[eigs[[2, 2]]], If[eigs[[1, 3]] == -1.`, v1 = Chop[eigs[[2, 3]], Return["Fail"]]]];
          b = Normalize[RandomReal[{-1, 1}, 3]];
          v2 = Normalize[b - (v1.b) v1];
          w = U.v2;
          If[Norm[w - v2] < .001, Return[{0, v1}]];
          ρ = ReflectionMatrix [v1];
          c = ArcCos[v2.w];
          σ = RotationMatrix[c, v1];
          If[Norm[σ.ρ - U] > 0.01`, c = -c];
          σ = RotationTransform [c, v1];
          {c, v1}]
```

Here an example is

```
In[ ]:= M2 = {{0.670820393249937` , -0.16245984811645334` , -0.7236067977499789` },
            {-0.6881909602355871` , -0.5000000000000001` , -0.5257311121191336` },
            {0.27639320225002084` , -0.85065080835204` , 0.4472135954999581` }};
        M2 // MatrixForm
```

Out[ ]//MatrixForm=

$$\begin{pmatrix} 0.67082 & -0.16246 & -0.723607 \\ -0.688191 & -0.5 & -0.525731 \\ 0.276393 & -0.850651 & 0.447214 \end{pmatrix}$$

```
In[ ]:= M2data = OddOrth2RR[M2]
Out[ ]= {-0.628319 , {0.276393 , 0.850651 , 0.447214}}

In[ ]:= rot2 = RotationMatrix[M2data[[1]], M2data[[2]]]
Out[ ]= {{0.823607 , 0.307768 , -0.476393},
         {-0.217963 , 0.947214 , 0.235114}, {0.523607 , -0.0898056 , 0.847214}}

In[ ]:= ref2 = ReflectionMatrix[M2data[[2]]]
Out[ ]= {{0.847214 , -0.470228 , -0.247214},
         {-0.470228 , -0.447214 , -0.760845}, {-0.247214 , -0.760845 , 0.6}}

In[ ]:= rr = rot2.ref2;
        rr // MatrixForm
```

Out[ ]//MatrixForm=

$$\begin{pmatrix} 0.67082 & -0.16246 & -0.723607 \\ -0.688191 & -0.5 & -0.525731 \\ 0.276393 & -0.850651 & 0.447214 \end{pmatrix}$$

```
In[ ]:= Norm[M2 - rr]
Out[ ]= 4.87461 × 10^-16
```

If we apply the algorithm to the reflection above

```
In[ ]:= OddOrth2RR[ref2]
Out[ ]= {0, {-0.276393 , -0.850651 , -0.447214}}
```

we get a rotation of angle 0, that is the identity. Reflections are here just a special case of rotary reflections with angle 0.

Here is an interesting phenomena, if a rotation matrix has finite even order n then the rotary reflection with same axis also has order n, but if the order is odd then the rotary reflection has order $2n$. Here are some examples.

```
In[ ]:= rot3 = RotationMatrix[2 Pi / 3,
            {0.30353099910334314` , 0.9341723589627159` , -0.18759247408507984` }];
```

```
In[ ]:= rot3.rot3.rot3
Out[ ]= {{1., -6.86156 × 10^-17, 5.55112 × 10^-17},
  {-1.05162 × 10^-16, 1., 3.06858 × 10^-17}, {-5.55112 × 10^-17, 4.89589 × 10^-17, 1.}}
```

So the rotation matrix has order 3. The reflection with the same axis is

```
In[ ]:= ref3 = ReflectionMatrix [
    {0.30353099910334314` , 0.9341723589627159` , -0.18759247408507984` }]
Out[ ]= {{0.815738 , -0.567101 , 0.11388},
  {-0.567101 , -0.745356 , 0.350487}, {0.11388 , 0.350487 , 0.929618}}
```

We can construct the rotary reflection

```
In[ ]:= rr3 = rot3.ref3
Out[ ]= {{-0.546066 , 0.0206847 , 0.837487},
  {-0.304235 , -0.936339 , -0.175244}, {-0.780547 , 0.350487 , -0.517595}}

In[ ]:= rr3cube = rr3.rr3.rr3
Out[ ]= {{0.815738 , -0.567101 , 0.11388},
  {-0.567101 , -0.745356 , 0.350487}, {0.11388 , 0.350487 , 0.929618}}
```

Thus this does not have order 3, but

```
In[ ]:= rr3cube.rr3cube
Out[ ]= {{1., -1.48858 × 10^-16, -1.46519 × 10^-17},
  {-1.00709 × 10^-16, 1., 7.40833 × 10^-17}, {2.29288 × 10^-16, -2.52181 × 10^-17, 1.}}
```

The rotary reflection rr3 has order 6. However if

```
In[ ]:= rot4 = RotationMatrix [Pi / 2,
    {0.20910145332310015` , -0.609341288974087` , 0.7648397059316191` }]
Out[ ]= {{0.0437234 , -0.892254 , -0.449412},
  {0.637426 , 0.371297 , -0.67515}, {0.76927 , -0.256947 , 0.58498}}

In[ ]:= rot4.rot4.rot4.rot4
Out[ ]= {{1., -2.77556 × 10^-17, 2.22045 × 10^-16},
  {-1.11022 × 10^-16, 1., -2.77556 × 10^-16}, {1.11022 × 10^-16, -2.77556 × 10^-16, 1.}}
```

This has order 4. The rotary reflection associated with this is

```
In[ ]:= ref4 =
   ReflectionMatrix [{0.20910145332310015` , -0.609341288974087` , 0.7648397059316191` }]
Out[ ]= {{0.912553 , 0.254828 , -0.319858},
  {0.254828 , 0.257406 , 0.932097}, {-0.319858 , 0.932097 , -0.16996}}
```

```
In[ ]:= rr4 = rot4.ref4
```

```
Out[ ]= {{-0.0437234, -0.637426, -0.76927},
  {0.892254, -0.371297, 0.256947}, {0.449412, 0.67515, -0.58498}}
```

```
In[ ]:= rr4.rr4.rr4.rr4
```

```
Out[ ]= {{1., -5.78916 × 10^-18, 8.89495 × 10^-17},
  {-8.49495 × 10^-17, 1., -2.6744 × 10^-16}, {1.17604 × 10^-16, -2.77451 × 10^-16, 1.}}
```

So the rotary reflection still has order 4. In the odd case the problem is that a point gets caught on the wrong side of the mirror after one full rotation.