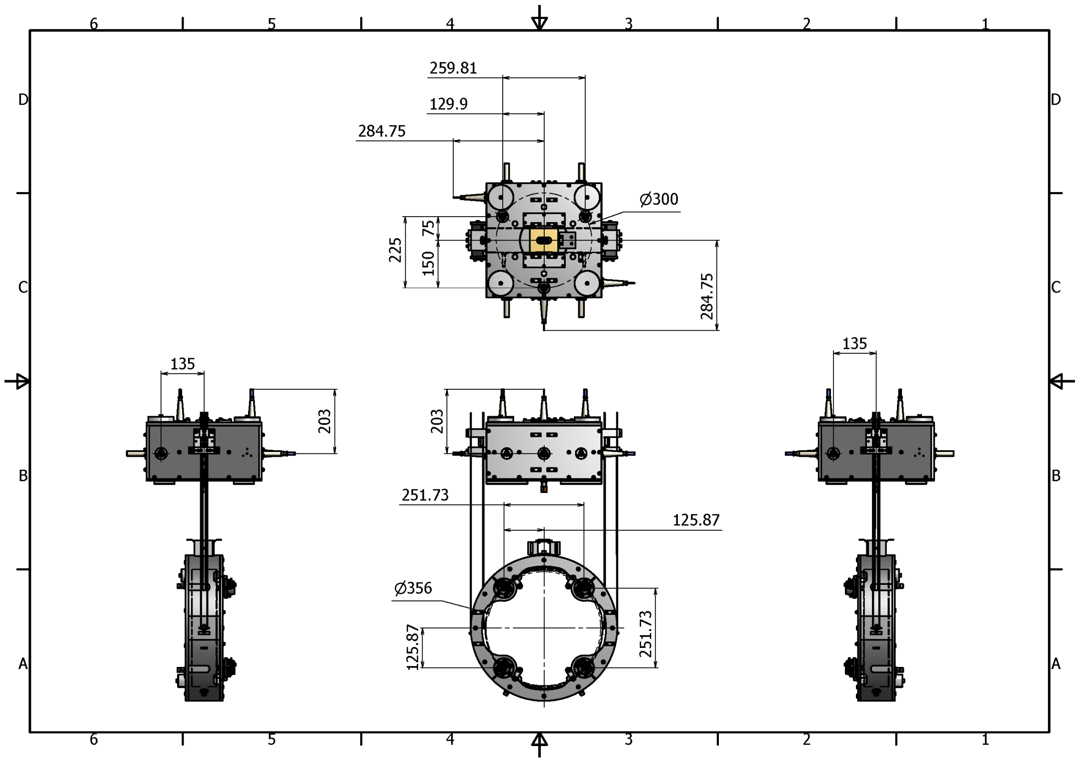
T1707205-v4 - PI Matrix Calc - BS Final Hang

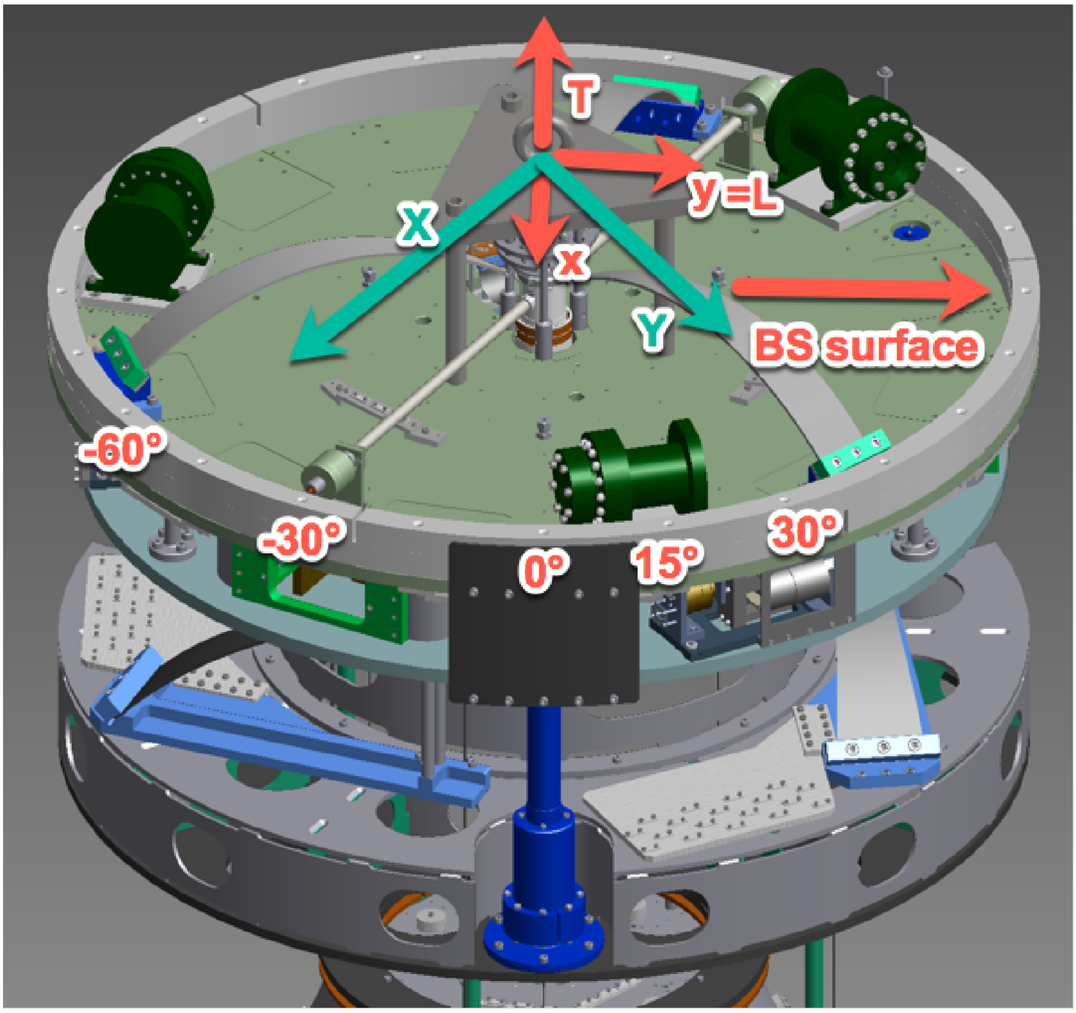
Data

Payload Data



The calculation is for the BS Final Hang arrangement in the tank.
X and Y are the standard interferometer global coordinates.
The BS surface faces midway between +Y and -X, and defines L (longitudinal).
T (transverse) is 90° anticlockwise from L (so as to make right handed LTV with vertical up).
For convenience, so that standard rotation matrices can be used, the calculation is done first in x/y coordinates where y=L and x=-T.
Leg #0 is in the +x direction and angles are measured anticlockwise from it.
FR #0, Geophone #0 and LVDT #0 are offset by -30°, 15° and 30° from Leg #0.
Legs #1 and #2 are at 120° and 240°.

PI Data



Data Summary

```
vals = {
   (* IP stuff *)
   R -> 715. / 1000, (* PI radius *)
   Fradius -> 1268.8 / 1000 / 2, (* from center to FR stepper; 3D CAD *)
   Gradius -> 1183. / 1000 / 2, (* from center to geophone;3D CAD *)
   Lradius -> 1188. / 1000 / 2, (* from center to LVDT;
   3D CAD; CHECK ME! *)
   Eradius → 650. / 1000, (* from center to EQ stop hole; guesstimate *)
   Iradius → 650. / 1000, (* from center to leg; guesstimate *)
   Eradius2 → 80. / 1000 / 2, (* radius of EQ stop hole; guesstimate *)
   Iradius2 → 80. / 1000 / 2, (* radius of leg; guesstimate *)
   l -> 200. / 1000, (* arrow length, for diagram *)
   origin -> 0 °, (* 3 o'clock *)
   (* IM stuff *)
   IMV1L → 0.15,
   IMV2L → 0.15 * Sin[Pi / 6],
   IMV3L → 0.15 * Sin[Pi / 6],
   IMV1T → 0,
   IMV2T → 0.15 * Cos[Pi / 6],
   IMV3T → 0.15 * Cos[Pi / 6],
   IMH1L → 0,
   IMH2L → 0.135,
   IMH3L → 0.135,
   (* TM stuff *)
   TMH1V → 0.25173 / 2,
   TMH2V → 0.25173 / 2,
   TMH3V → 0.25173 / 2,
   TMH4V → 0.25173 / 2,
   TMH1T → 0.25173 / 2,
   TMH2T → 0.25173 / 2,
   TMH3T → 0.25173 / 2,
   TMH4T → 0.25173 / 2
  };
```

TM Geometry

- **The test matrix, which gives the change at each sensor for excursions in L, P and Y**

```
(TMtest = Transpose[{
      {-1, -1, -1, -1},
      {-TMH1V, TMH2V, -TMH3V, TMH4V},
      {TMH1T, TMH2T, -TMH3T, -TMH4T}
     }] /. vals
 ) // TableForm
```

-1	-0.125865	0.125865
-1	0.125865	0.125865
-1	-0.125865	-0.125865
-1	0.125865	-0.125865

- **The diagonalization matrix, which attempts to undo the test matrix. This matrix should be typed as-is into the OSEM2EUL screen and transposed into the EUL2OSEM screen.**

```
(TM = {
      -0.25 * {1, 1, 1, 1},
      0.25 * {-1 / TMH1V, 1 / TMH2V, -1 / TMH3V, 1 / TMH4V},
      0.25 * {1 / TMH1T, 1 / TMH2T, -1 / TMH3T, -1 / TMH4T}
     } /. vals
 ) // TableForm
```

-0.25	-0.25	-0.25	-0.25
-1.98626	1.98626	-1.98626	1.98626
1.98626	1.98626	-1.98626	-1.98626

- **Multiplying the diagonalization matrix and the test matrix should give the identity matrix.**

```
TM.TMtest // TableForm
```

1.	0.	0.
0.	1.	0.
0.	0.	1.

IM Geometry

- **The test matrix, which gives the change at each sensor for excursions in L, T, V, R, P and Y**

```
(IMtest = Transpose[{
         {0, 0, 0, -1, 0, 0}, (* L *)
         {0, 0, 0, 0, -1, 1}, (* T *)
         {-1, -1, -1, 0, 0, 0}, (* V *)
         {0, -IMV2T, IMV3T, 0, 0, 0}, (* R *)
         {IMV1L, -IMV2L, -IMV3L, 0, 0, 0}, (* P *)
         {0, 0, 0, 0, -IMH2L, -IMH3L} (* Y *)
        }] /. vals // N
  ) // Transpose // TableForm
```

0.	0.	0.	-1.	0.	0.
0.	0.	0.	0.	-1.	1.
-1.	-1.	-1.	0.	0.	0.
0.	-0.129904	0.129904	0.	0.	0.
0.15	-0.075	-0.075	0.	0.	0.
0.	0.	0.	0.	-0.135	-0.135

- **The diagonalization matrix, which attempts to undo the test matrix. This matrix should be typed as-is into the OSEM2EUL screen and transposed into the EUL2OSEM screen.**

```
(IM = {
      {0, 0, 0, -1, 0, 0}, (* L *)
      {0, 0, 0, 0, -1, 1} / 2, (* T *)
      {-1, -1, -1, 0, 0, 0} / 3, (* V *)
      {0, -1 / IMV2T, 1 / IMV3T, 0, 0, 0} / 2, (* R *)
      {4 / IMV1L, -1 / IMV2L, -1 / IMV3L, 0, 0, 0} / 6, (* P *)
      {0, 0, 0, 0, -1 / IMH2L, -1 / IMH3L} / 2 (* Y *)
     } /. vals // N
 ) // TableForm
```

0.	0.	0.	-1.	0.	0.
0.	0.	0.	0.	-0.5	0.5
-0.333333	-0.333333	-0.333333	0.	0.	0.
0.	-3.849	3.849	0.	0.	0.
4.44444	-2.22222	-2.22222	0.	0.	0.
0.	0.	0.	0.	-3.7037	-3.7037

- **Multiplying the diagonalization matrix and the test matrix should give the identity matrix.**

```
IM.IMtest // TableForm
```

1.	0.	0.	0.	0.	0.
0.	1.	0.	0.	0.	0.
0.	0.	1.	0.	0.	0.
0.	0.	0.	1.	0.	0.
0.	0.	0.	0.	1.	0.
0.	0.	0.	0.	0.	1.

PI Geometry

Calculation

- **90° Rotation matrix for x/y -> L/T.**

```
Ninety = RotationMatrix[90 °]
```

```
{{0, -1}, {1, 0}}
```

- **Angular locations of the various items**

IP legs

```
Iangles = {0 °, 120 °, 240 °};
```

Geophones

```
Gangles = 15 ° + Iangles
```

```
{15 °, 135 °, 255 °}
```

LVDTs

```
Langles = 30 ° + Iangles
```

```
{30 °, 150 °, 270 °}
```

Fishing rod steppers

```
Fangles = -30 ° + Iangles
```

```
{-30 °, 90 °, 210 °}
```

EQ stop positions

```
Eangles = -60 ° + Iangles
```

$\{-60\,°,\ 60\,°,\ 180\,°\}$

- **Positions of the various items as vectors**

These are calculated by rotating the x axis {1,0} by the corresponding angles.

```
Ipositions = Iradius * Table[RotationMatrix[Iangles[[i]]].{1, 0}, {i, 1, 3}]
```

$\left\{\{\text{Iradius}, 0\}, \left\{-\frac{\text{Iradius}}{2}, \frac{\sqrt{3}\ \text{Iradius}}{2}\right\}, \left\{-\frac{\text{Iradius}}{2}, -\frac{\sqrt{3}\ \text{Iradius}}{2}\right\}\right\}$

```
Fpositions = Fradius * Table[RotationMatrix[Fangles[[i]]].{1, 0}, {i, 1, 3}]
```

$\left\{\left\{\frac{\sqrt{3}\ \text{Fradius}}{2}, -\frac{\text{Fradius}}{2}\right\}, \{0, \text{Fradius}\}, \left\{-\frac{\sqrt{3}\ \text{Fradius}}{2}, -\frac{\text{Fradius}}{2}\right\}\right\}$

$\left\{\left\{\frac{\sqrt{3}\ \text{Fradius}}{2}, -\frac{\text{Fradius}}{2}\right\}, \{0, \text{Fradius}\}, \left\{-\frac{\sqrt{3}\ \text{Fradius}}{2}, -\frac{\text{Fradius}}{2}\right\}\right\}$

$\left\{\left\{\frac{\sqrt{3}\ \text{Fradius}}{2}, -\frac{\text{Fradius}}{2}\right\}, \{0, \text{Fradius}\}, \left\{-\frac{\sqrt{3}\ \text{Fradius}}{2}, -\frac{\text{Fradius}}{2}\right\}\right\}$

```
Gpositions =  Gradius * Table[RotationMatrix[Gangles[[i]]].{1, 0}, {i, 1, 3}]
```

$\left\{\left\{\frac{\left(1+\sqrt{3}\right)\ \text{Gradius}}{2\ \sqrt{2}}, \frac{\left(-1+\sqrt{3}\right)\ \text{Gradius}}{2\ \sqrt{2}}\right\}, \left\{-\frac{\text{Gradius}}{\sqrt{2}}, \frac{\text{Gradius}}{\sqrt{2}}\right\}, \left\{-\frac{\left(-1+\sqrt{3}\right)\ \text{Gradius}}{2\ \sqrt{2}}, -\frac{\left(1+\sqrt{3}\right)\ \text{Gradius}}{2\ \sqrt{2}}\right\}\right\}$

```
Lpositions =  Lradius * Table[RotationMatrix[Langles[[i]]].{1, 0}, {i, 1, 3}]
```

$\left\{\left\{\frac{\sqrt{3}\ \text{Lradius}}{2}, \frac{\text{Lradius}}{2}\right\}, \left\{-\frac{\sqrt{3}\ \text{Lradius}}{2}, \frac{\text{Lradius}}{2}\right\}, \{0, -\text{Lradius}\}\right\}$

```
Epositions =  Eradius * Table[RotationMatrix[Eangles[[i]]].{1, 0}, {i, 1, 3}]
```

$\left\{\left\{\frac{\text{Eradius}}{2}, -\frac{\sqrt{3}\ \text{Eradius}}{2}\right\}, \left\{\frac{\text{Eradius}}{2}, \frac{\sqrt{3}\ \text{Eradius}}{2}\right\}, \{-\text{Eradius}, 0\}\right\}$

- **Orientations of the various items as unit vectors**

These are calculated by rotating a unit y vector {0,1} (i.e., the tangent vector at {1,0}) by the same set of angles.

```
Fvectors = Table[RotationMatrix[Fangles[[i]]].{0, 1}, {i, 1, 3}]
```

$\left\{\left\{\frac{1}{2}, \frac{\sqrt{3}}{2}\right\}, \{-1, 0\}, \left\{\frac{1}{2}, -\frac{\sqrt{3}}{2}\right\}\right\}$

```
Gvectors = Table[RotationMatrix[Gangles[[i]]].{0, 1}, {i, 1, 3}]
```

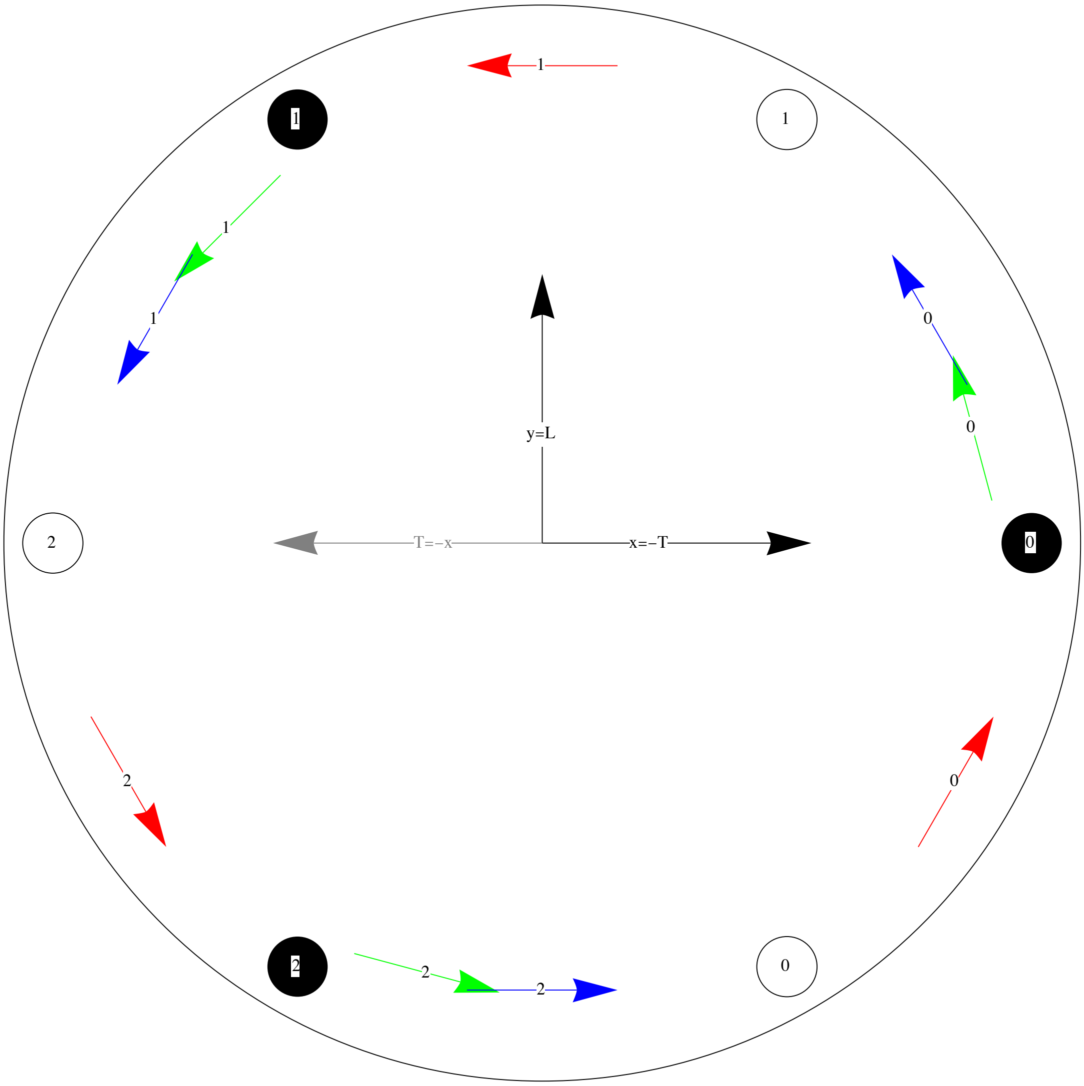
$\left\{\left\{-\frac{-1+\sqrt{3}}{2\ \sqrt{2}}, \frac{1+\sqrt{3}}{2\ \sqrt{2}}\right\}, \left\{-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right\}, \left\{\frac{1+\sqrt{3}}{2\ \sqrt{2}}, -\frac{-1+\sqrt{3}}{2\ \sqrt{2}}\right\}\right\}$

```
Lvectors = Table[RotationMatrix[Langles[[i]]].{0, 1}, {i, 1, 3}]
```

$\left\{\left\{-\frac{1}{2}, \frac{\sqrt{3}}{2}\right\}, \left\{-\frac{1}{2}, -\frac{\sqrt{3}}{2}\right\}, \{1, 0\}\right\}$

- **Diagram**

```
Graphics[{
   Circle[{0, 0}, R],
   Sequence[Table[Disk[Ipositions[[i]], Iradius2],
     {i, 1, 3}
    ]],
   Sequence[Table[Text[i - 1, Ipositions[[i]], Background → White], {i, 1, 3}]],
   Sequence[Table[Circle[Epositions[[i]], Eradius2],
     {i, 1, 3}
    ]],
   Sequence[Table[Text[i - 1, Epositions[[i]], Background → White], {i, 1, 3}]],
   Red,
   Sequence[Table[Arrow[{Fpositions[[i]] - Fvectors[[i]] * l / 2,
       Fpositions[[i]] + Fvectors[[i]] * l / 2}], {i, 1, 3}
    ]],
   Black, Sequence[Table[Text[i - 1, Fpositions[[i]], Background → White], {i, 1, 3}]],
   Green,
   Sequence[Table[Arrow[{Gpositions[[i]] - Gvectors[[i]] * l / 2,
       Gpositions[[i]] + Gvectors[[i]] * l / 2}], {i, 1, 3}
    ]],
   Black, Sequence[Table[Text[i - 1, Gpositions[[i]], Background → White], {i, 1, 3}]],
   Blue,
   Sequence[Table[Arrow[{Lpositions[[i]] - Lvectors[[i]] * l / 2,
       Lpositions[[i]] + Lvectors[[i]] * l / 2}], {i, 1, 3}
    ]],
   Black, Sequence[Table[Text[i - 1, Lpositions[[i]], Background → White], {i, 1, 3}]],
   Arrow[{{0, 0}, {R / 2, 0}}], Text["x=-T", {R / 5, 0}, Background → White],
   Arrow[{{0, 0}, {0, R / 2}}], Text["y=L", {0, R / 5}, Background → White],
   Gray,
   Arrow[{{0, 0}, {-R / 2, 0}}], Text["T=-x", {-R / 5, 0}, Background → White]
  }] /. vals
```



■ Matrices between L/T/Y and LVDTs

Usage: {L0, L1, L2}= LVDTfromLTY.{L, T, Y}

```
(LVDTfromLTY = {
    Table[Lvectors[[i]].Ninety.{1, 0}, {i, 1, 3}],
    Table[Lvectors[[i]].Ninety.{0, 1}, {i, 1, 3}],
    Table[Lradius, {i, 1, 3}]
  }) // TableForm
```

$\frac{\sqrt{3}}{2}$	$-\frac{\sqrt{3}}{2}$	0
$\frac{1}{2}$	$\frac{1}{2}$	-1
Lradius	Lradius	Lradius

Usage: type these numbers into EUL2COIL

```
Transpose[LVDTfromLTY] /. vals // N // TableForm
```

0.866025	0.5	0.594
-0.866025	0.5	0.594
0.	-1.	0.594

Usage: {L, T, Y}= LTYfromLVDT.{L0, L1, L2}

```
(LTYfromLVDT = Inverse[LVDTfromLTY]) // TableForm
```

$$\begin{array}{lll} \frac{1}{\sqrt{3}} & \frac{1}{3} & \frac{1}{3\,\text{Lradius}} \\ -\frac{1}{\sqrt{3}} & \frac{1}{3} & \frac{1}{3\,\text{Lradius}} \\ 0 & -\frac{2}{3} & \frac{1}{3\,\text{Lradius}} \end{array}$$

Usage: type these numbers into LVDT2EUL

```
Transpose[LTYfromLVDT] /. vals // N // TableForm
```

0.57735	-0.57735	0.
0.333333	0.333333	-0.666667
0.561167	0.561167	0.561167

■ Matrices between L/T/Y and Geophones

Usage: {G0, G1, G2}= GfromLTY.{L, T, Y} (assuming the geophones had actuation)

```
(GfromLTY = {
    Table[Gvectors[[i]].Ninety.{1, 0}, {i, 1, 3}],
    Table[Gvectors[[i]].Ninety.{0, 1}, {i, 1, 3}],
    Table[Gradius, {i, 1, 3}]
  }) // TableForm
```

$$\begin{array}{lll} \frac{1+\sqrt{3}}{2\sqrt{2}} & -\frac{1}{\sqrt{2}} & -\frac{-1+\sqrt{3}}{2\sqrt{2}} \\ \frac{-1+\sqrt{3}}{2\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{1+\sqrt{3}}{2\sqrt{2}} \\ \text{Gradius} & \text{Gradius} & \text{Gradius} \end{array}$$

Usage: you would type these numbers into EUL2ACC if it existed, which it doesn't because geophones don't have actuation.

```
Transpose[GfromLTY] /. vals // N // TableForm
```

0.965926	0.258819	0.5915
-0.707107	0.707107	0.5915
-0.258819	-0.965926	0.5915

Usage: {L, T, Y}= LTYfromG.{G0, G1, G2}

```
(LTYfromG = Inverse[GfromLTY] // Simplify) // TableForm
```

$$\begin{array}{lll} \frac{3+\sqrt{3}}{3\sqrt{6}} & \frac{-1+\sqrt{3}}{3\sqrt{2}} & \frac{1}{3\,\text{Gradius}} \\ -\frac{\sqrt{2}}{3} & \frac{\sqrt{2}}{3} & \frac{1}{3\,\text{Gradius}} \\ \frac{-3+\sqrt{3}}{3\sqrt{6}} & -\frac{3+\sqrt{3}}{3\sqrt{6}} & \frac{1}{3\,\text{Gradius}} \end{array}$$

Usage: type these numbers into ACC2EUL

```
Transpose[LTYfromG] /. vals // N // TableForm
```

0.643951	-0.471405	-0.172546
0.172546	0.471405	-0.643951
0.563539	0.563539	0.563539

■ Matrices between L/T/Y and FRs

Usage: {F0, F1, F2}= FfromLTY.{L, T, Y}

```
(FfromLTY = {
    Table[Fvectors[[i]].Ninety.{1, 0}, {i, 1, 3}],
    Table[Fvectors[[i]].Ninety.{0, 1}, {i, 1, 3}],
    Table[Fradius, {i, 1, 3}]
   }) // TableForm
```

$\frac{\sqrt{3}}{2}$	0	$-\frac{\sqrt{3}}{2}$
$-\frac{1}{2}$	1	$-\frac{1}{2}$
Fradius	Fradius	Fradius

$\frac{\sqrt{3}}{2}$	0	$-\frac{\sqrt{3}}{2}$
$-\frac{1}{2}$	1	$-\frac{1}{2}$
Fradius	Fradius	Fradius

$$\left\{\left\{\frac{\sqrt{3}}{2}, 0, -\frac{\sqrt{3}}{2}\right\}, \left\{-\frac{1}{2}, 1, -\frac{1}{2}\right\}, \{\text{Fradius}, \text{Fradius}, \text{Fradius}\}\right\}$$

Usage: write a Python script to take DC force/torque requests in LTY coordinates, multiply by this matrix and output stepper motor movements in steps.

```
FfromLTY /. vals // N // TableForm
```

0.866025	0.	-0.866025
-0.5	1.	-0.5
0.6344	0.6344	0.6344

Usage: {L, T, Y}= LTYfromF.{F0, F1, F2}

```
(LTYfromF = Inverse[FfromLTY] // Simplify) // TableForm
```

$\frac{1}{\sqrt{3}}$	$-\frac{1}{3}$	$\frac{1}{3\,\text{Fradius}}$
0	$\frac{2}{3}$	$\frac{1}{3\,\text{Fradius}}$
$-\frac{1}{\sqrt{3}}$	$-\frac{1}{3}$	$\frac{1}{3\,\text{Fradius}}$

$\frac{1}{\sqrt{3}}$	$-\frac{1}{3}$	$\frac{1}{3\,\text{Fradius}}$
0	$\frac{2}{3}$	$\frac{1}{3\,\text{Fradius}}$
$-\frac{1}{\sqrt{3}}$	$-\frac{1}{3}$	$\frac{1}{3\,\text{Fradius}}$

$$\left\{\left\{\frac{1}{\sqrt{3}}, -\frac{1}{3}, \frac{1}{3\,\text{Fradius}}\right\}, \left\{0, \frac{2}{3}, \frac{1}{3\,\text{Fradius}}\right\}, \left\{-\frac{1}{\sqrt{3}}, -\frac{1}{3}, \frac{1}{3\,\text{Fradius}}\right\}\right\}$$

Usage: you would type these numbers into FR2EUL, if it existed, which it doesn't because FRs don't have sensing.

```
Transpose[LTYfromF] /. vals // N // TableForm
```

0.57735	0.	-0.57735
-0.333333	0.666667	-0.333333
0.525431	0.525431	0.525431

- **Matrices between L/T/Y and LVDTs (for comparison with Enzo's -v3; completely wrong; don't use ever)**
- **Matrices between x/y/yaw and LVDTs (for comparison with -v2; don't use unless you want EUL=x/y)**