FreeCAD World

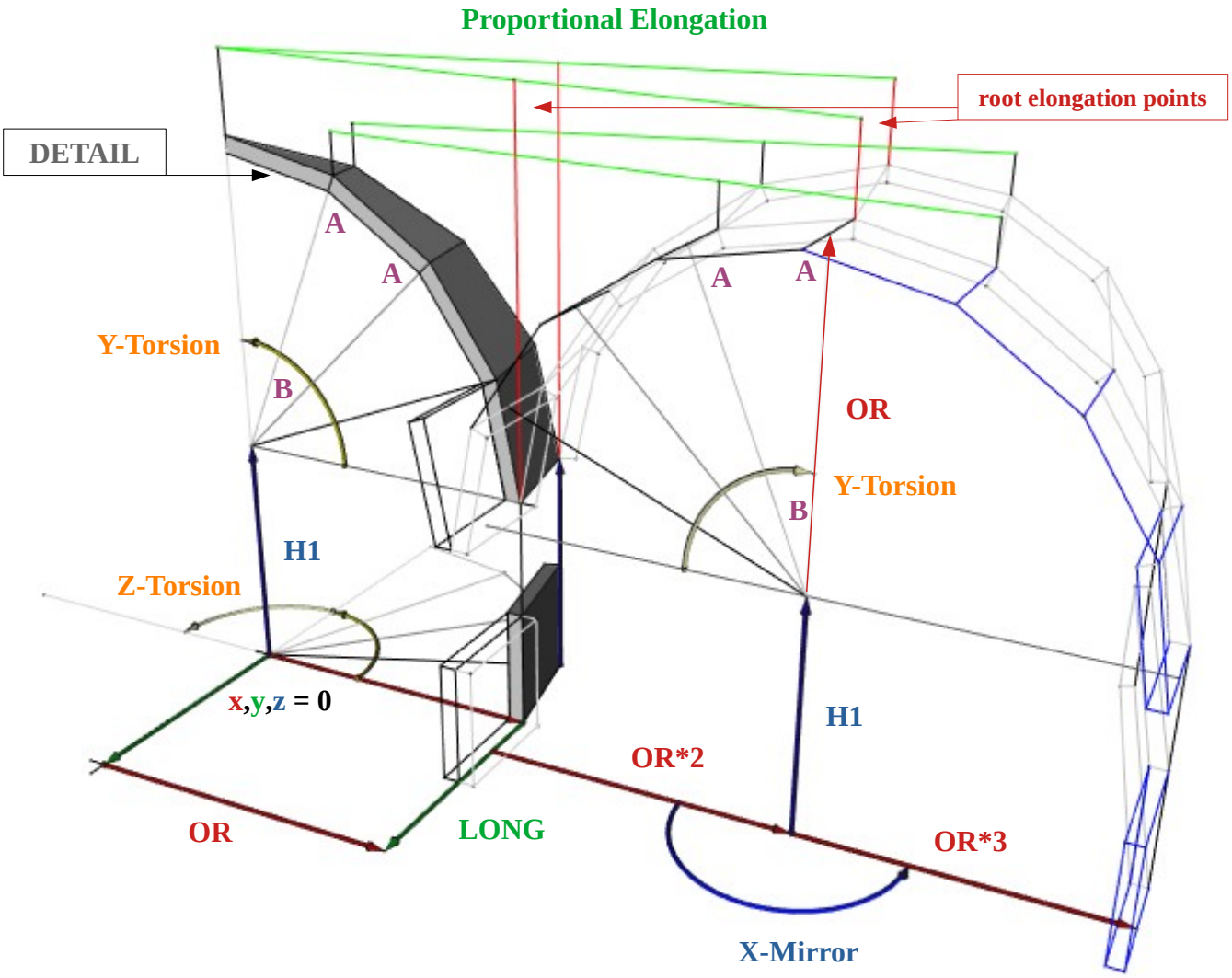
[notes]

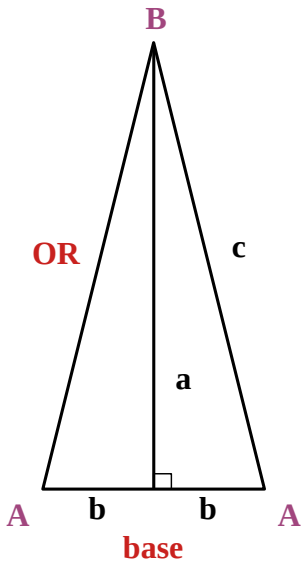
Config types

[all types and values (except **str("eg. Value or valUe")**) are **case sensetive**]

- **int():** integers: (1, 2, 15, 144, 2048, etc);
- **float():** floating point numbers: (1.2, 2.3, 15.123, 144.9, 2048.1024, ¾, ½, etc);
- **bool():** boolean **True** or **False**, 1 or 0. (In real life is **On/Off**);
- **tuple():** in config used as **comma separated** tuple of integers or floats;
- **str():** string as any human readable words, eg **str("CORNER")**;
- **dict():** **comma separated** dictionary of options: str(key) = value, where value may be **one** of described above types;

The Principle





B = 360°/**DETAILS**

A + **B**/2 = 90°

A = (180° - **B**)/2

Isosceles Triangle as base of construction

a = **OR** * sin(**A**) = cathetus

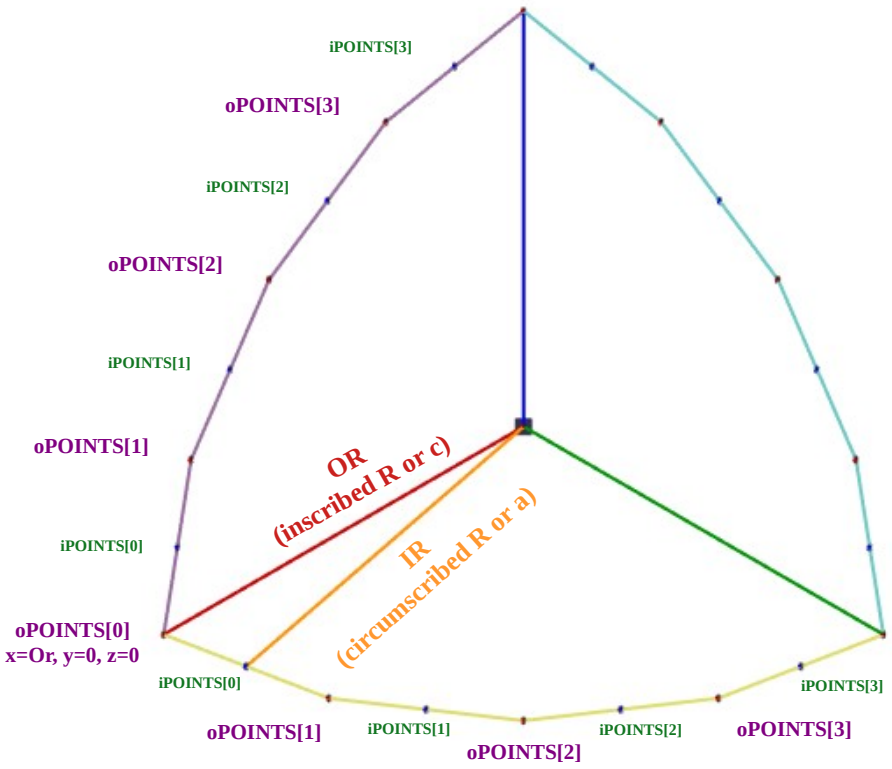
b = **OR** * cos(**A**) = side **b**

OR = **a** / cos(**B**/2) = **c** = hypotenuse

base = **b** * 2 = **OR** * 2 * cos(**A**)

Pointing

- **oPOINTS;**
- **iPOINTS;**



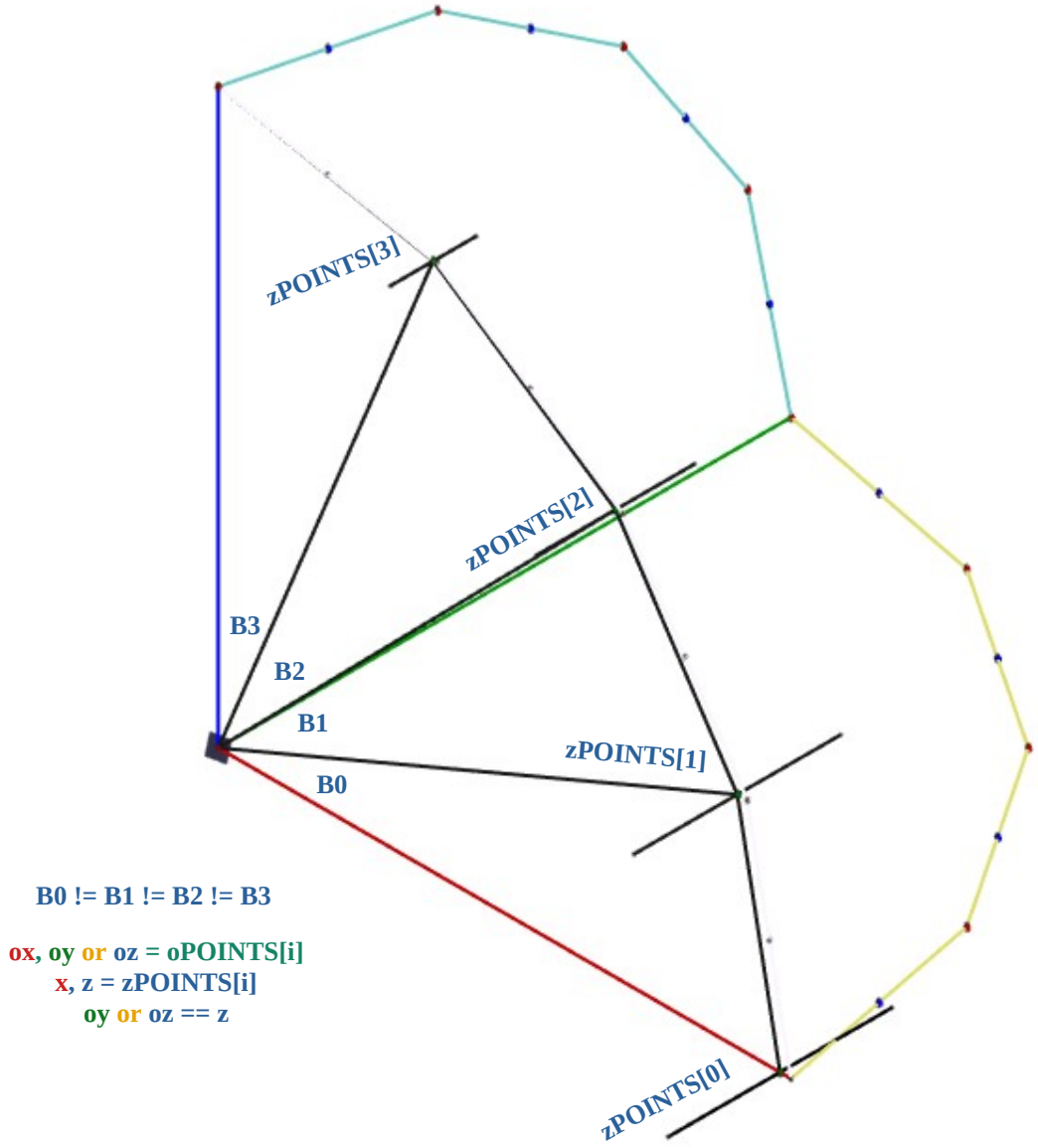
See:

- **clockWiseArray(x, y);**
- **oi_pointing.FCStd**

```
class PairOfCompasses(Trigon):
    _deviation_ = float(0.01)

    def clockWiseArray(self, x, y):
        """
            Clockwise bottom view
        """
        A = self.DETAIL_A
        B = self.DETAIL_B
        """
            Extra solution for thorus to find proportions
            on y axis of higher 'horison poly wireframe'.
            System-wide single deviation is:
                x == 0.01 mm if 0 on x & y axis
                in case of self.DETAILS % 4 == 0
        """
        if x < self.DEVIATION: x = self.DEVIATION
        base = self.isoscelesBase(x)
        points = list( [[ x, y ]] )
        for i in range(self.POLY_QUARTER):
            x -= self.rightSideB_ByA(base, A)
            y += self.rightCathetusA_ByA(base, A)
            points.append([ x, y ])
            A -= B
        return points
```

- **zPOINTS;**



See:

- **zPolyPoints(Or);**
- **z_pointing.FCStd;**

```
def zPolyPoints(self, Or):
    """
        Receives: OR of inscribed into circle polygon;
        Returns: sequence of mid x, z points of base
        of isosceless triangle on y=0 and ZERO_Z=0;
    """
    oPOINTS = self.oPolyPoints(Or)
    points  = list()
    for x, z in oPOINTS:
        x, y = self.firstSectionToCircumscribed(x, z)
        x, y = self.circumscribedCounterClockWiseOnce(x, y)
        points.append([ x, z ])
    return points
```