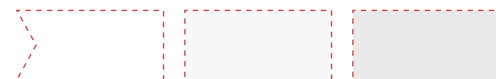
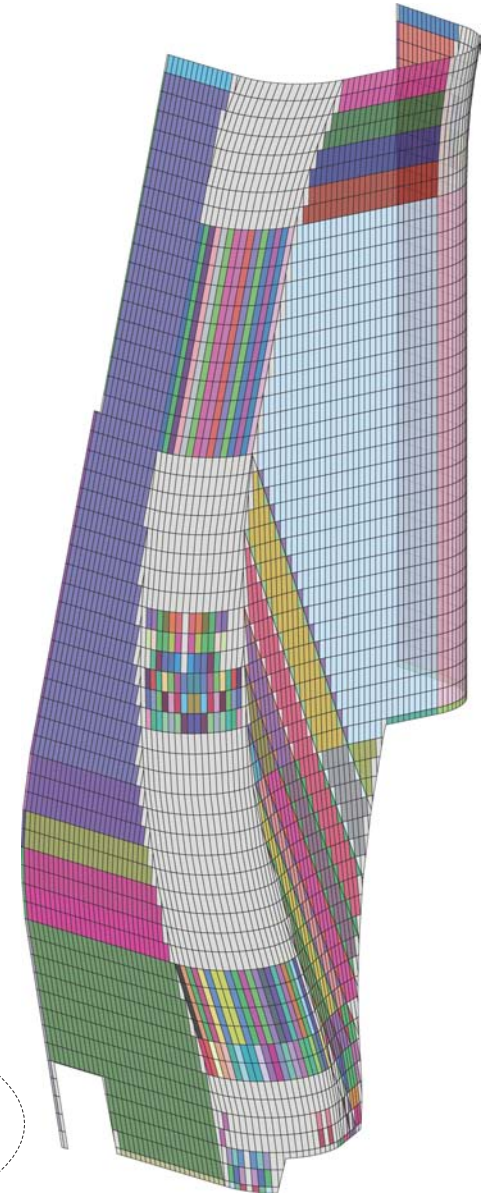
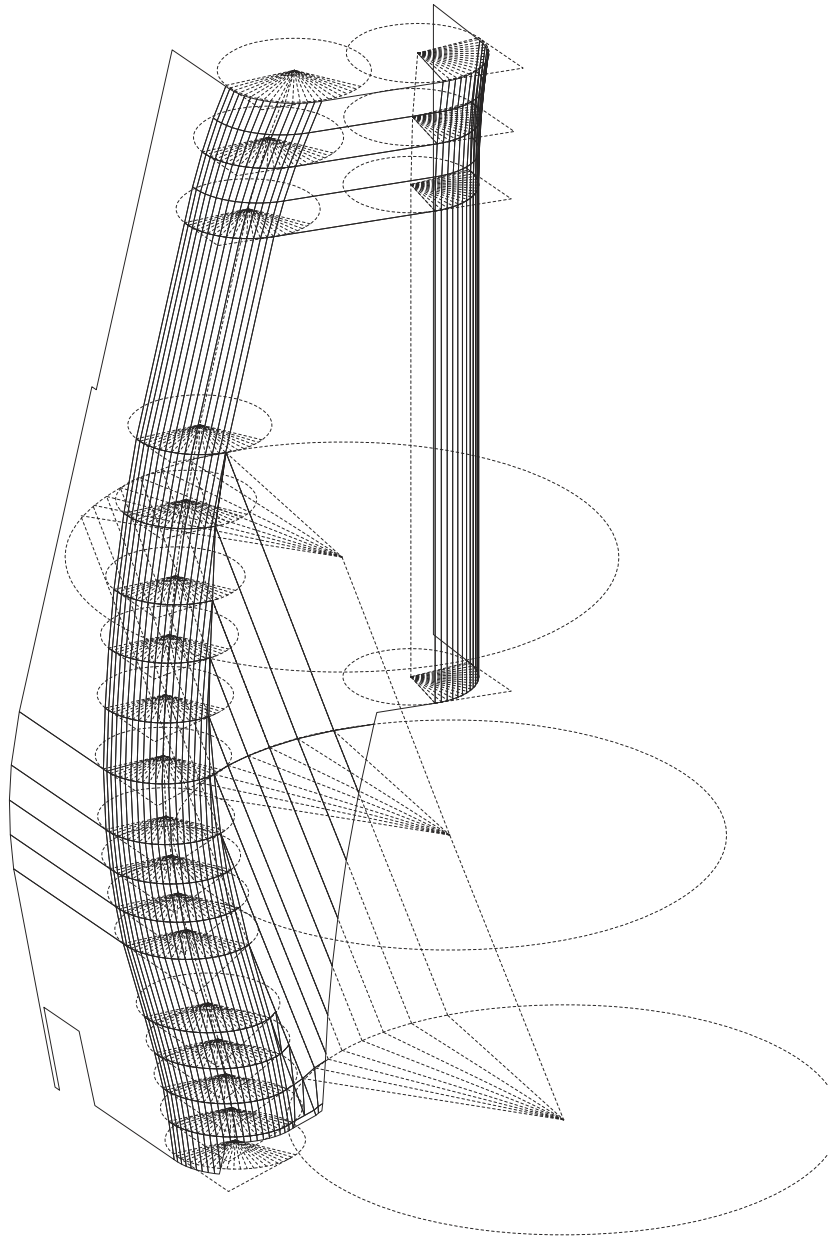
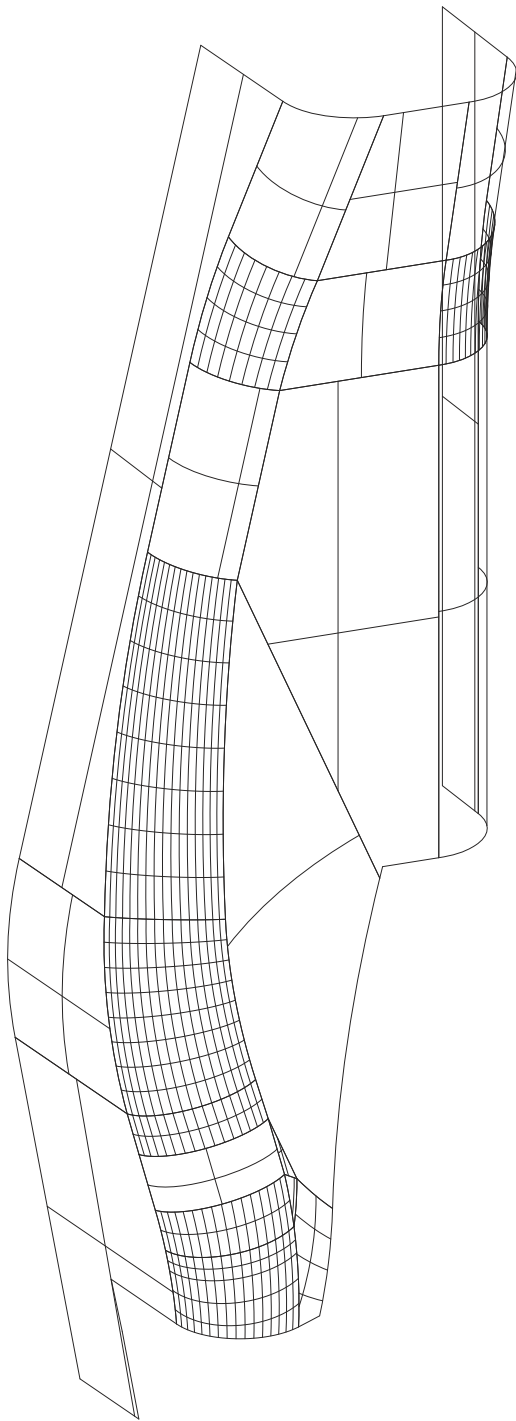
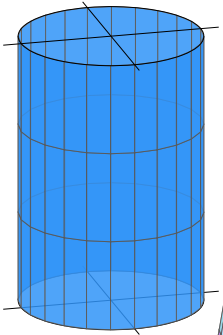


MORPH SEMINAR 2009
MATHEMATICS FOR SYSTEMATIC MODELING
WEEK1 POINT AND VECTOR

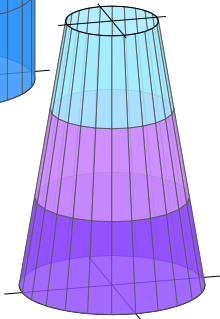




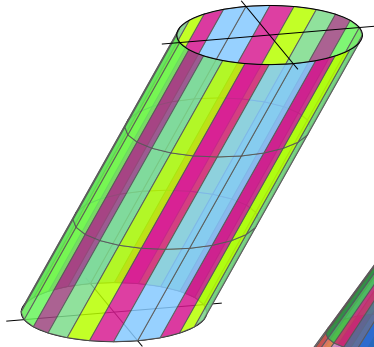
cylinder



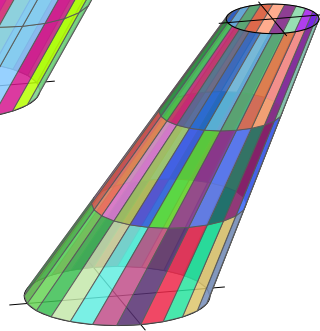
cone



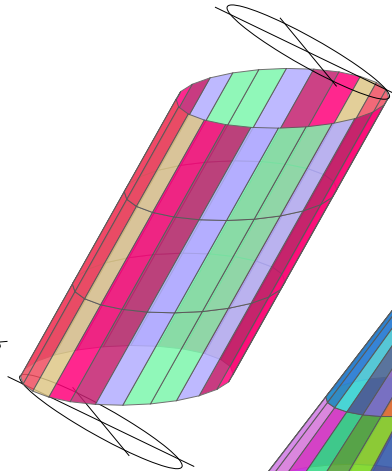
sheared cylinder



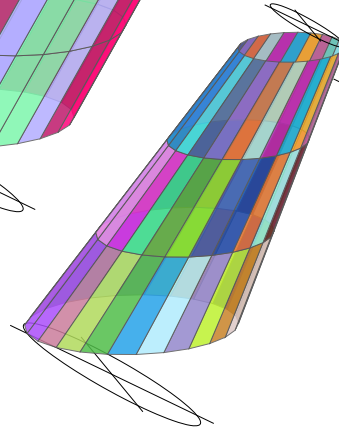
sheared cone



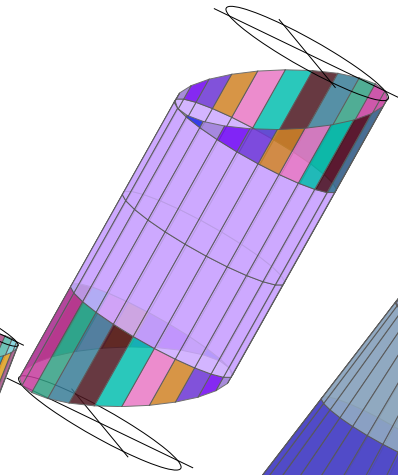
tilted cylinder



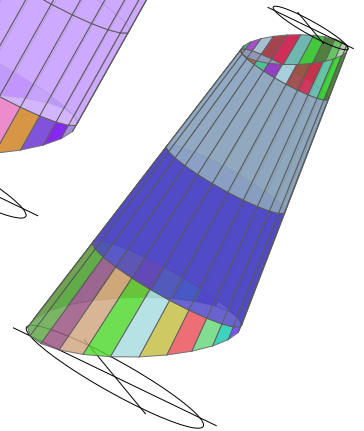
tilted cone



tilted cylinder



tilted cone

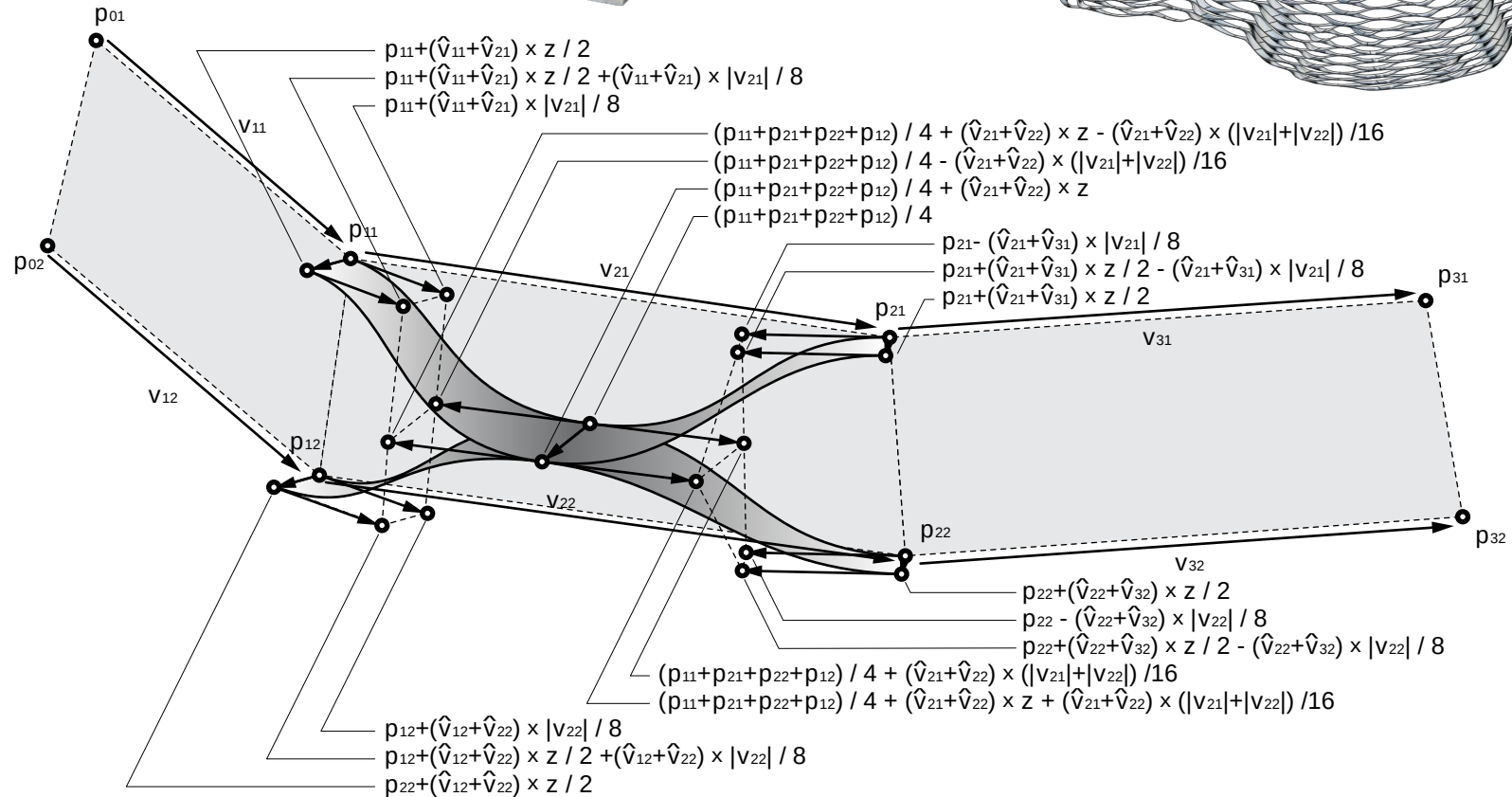
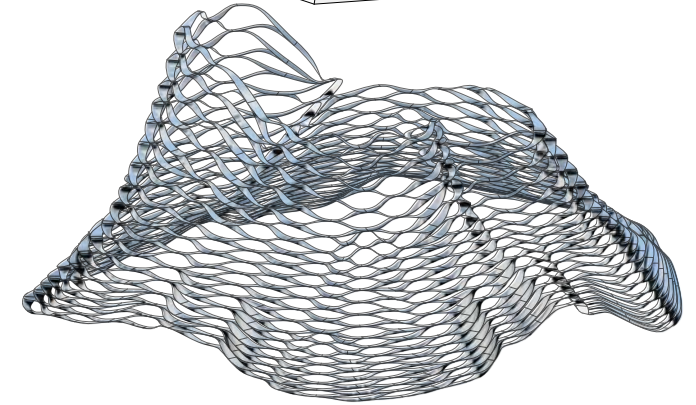
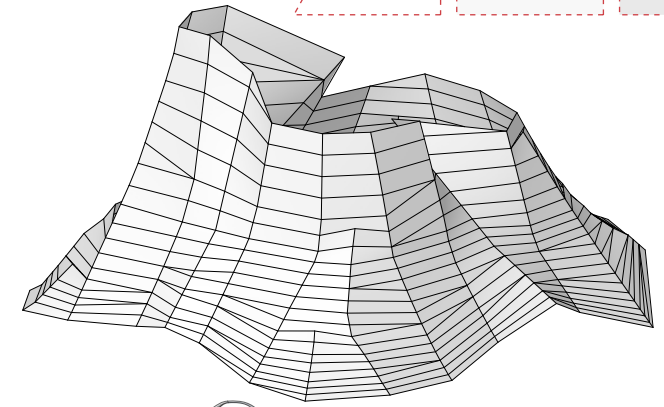


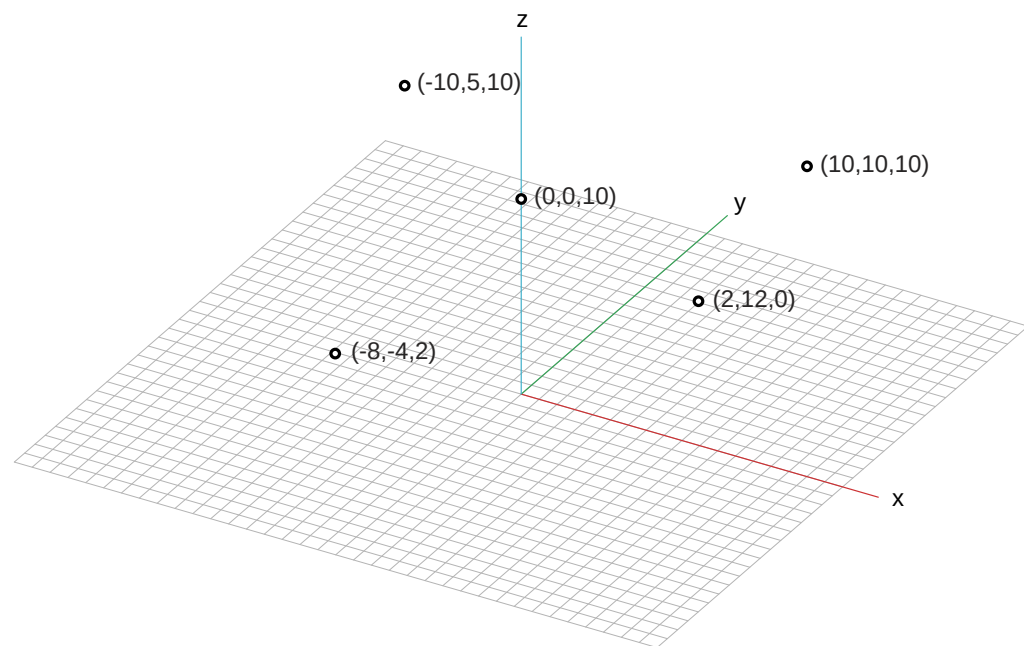
MATHEMATICS FOR SYSTEMATIC MODELING_1 POINT AND VECTOR

VECTOR MODELING EXAMPLE



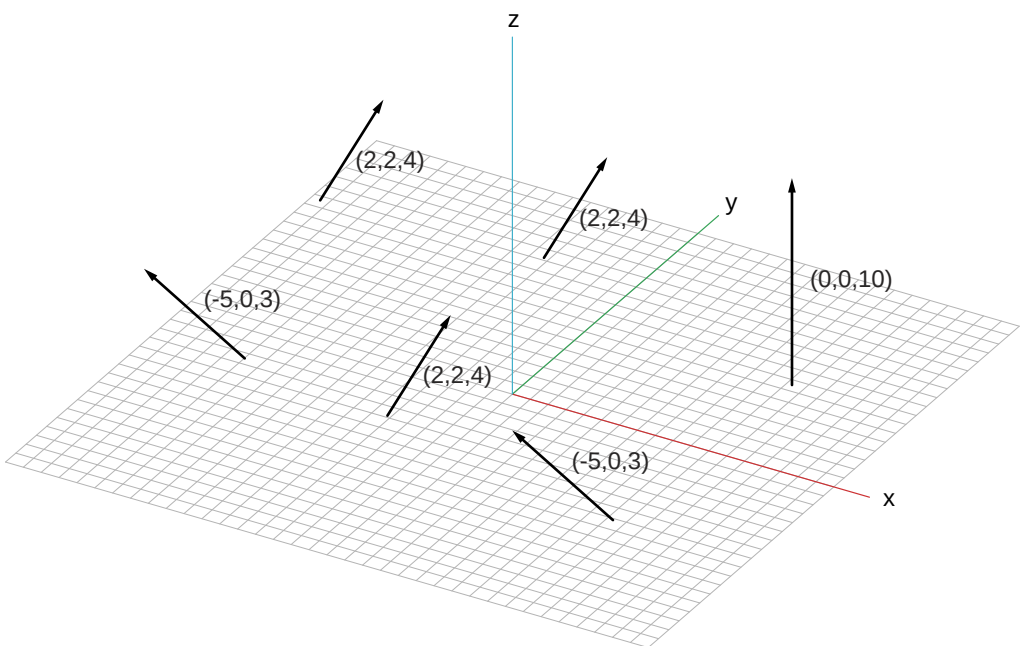
(c) DR_D





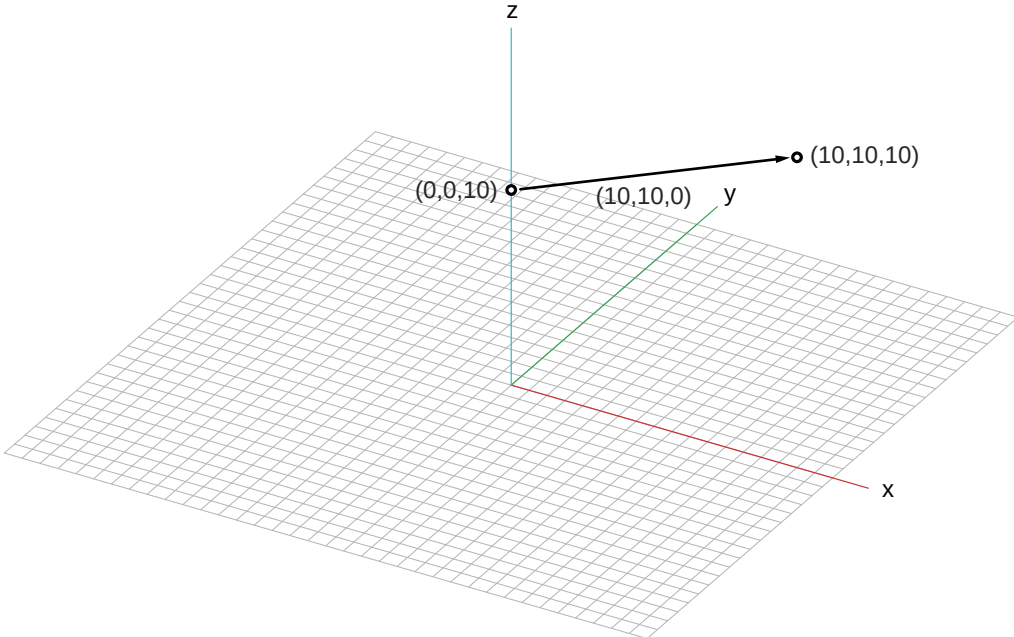
POINT

$p = (x, y, z)$
 $p = (x, y, z)$
 $\vec{p} = (x, y, z)$

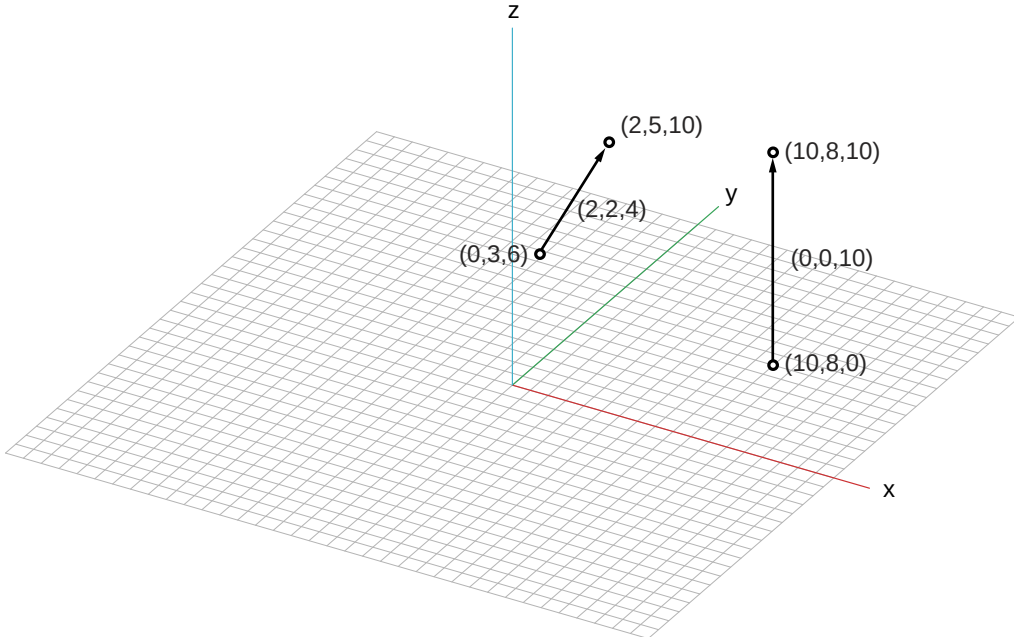


VECTOR

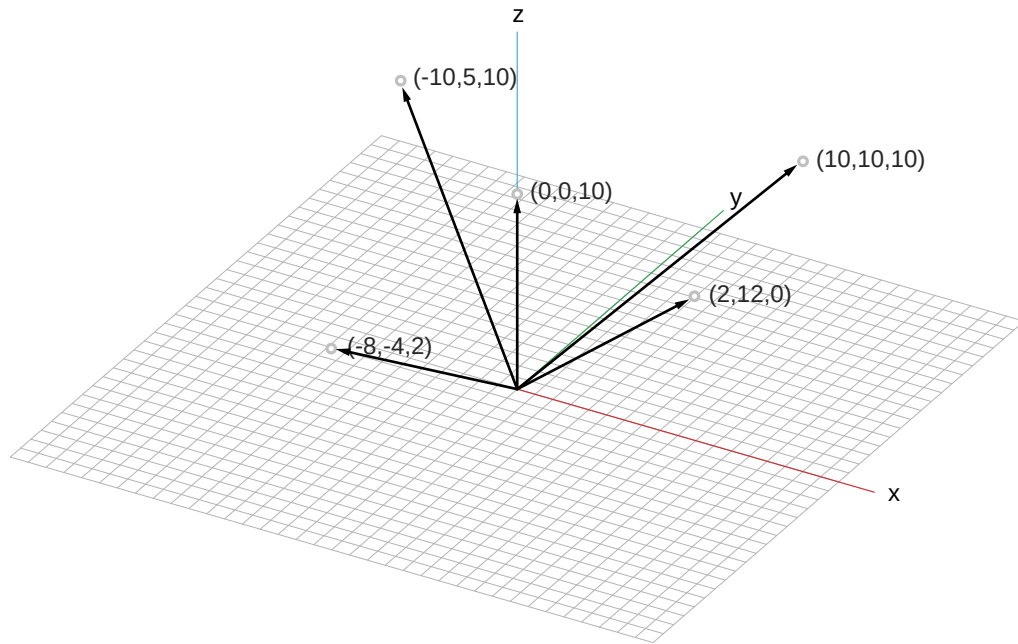
$v = (x, y, z)$
 $v = (x, y, z)$
 $\vec{v} = (x, y, z)$



$p = (x, y, z)$

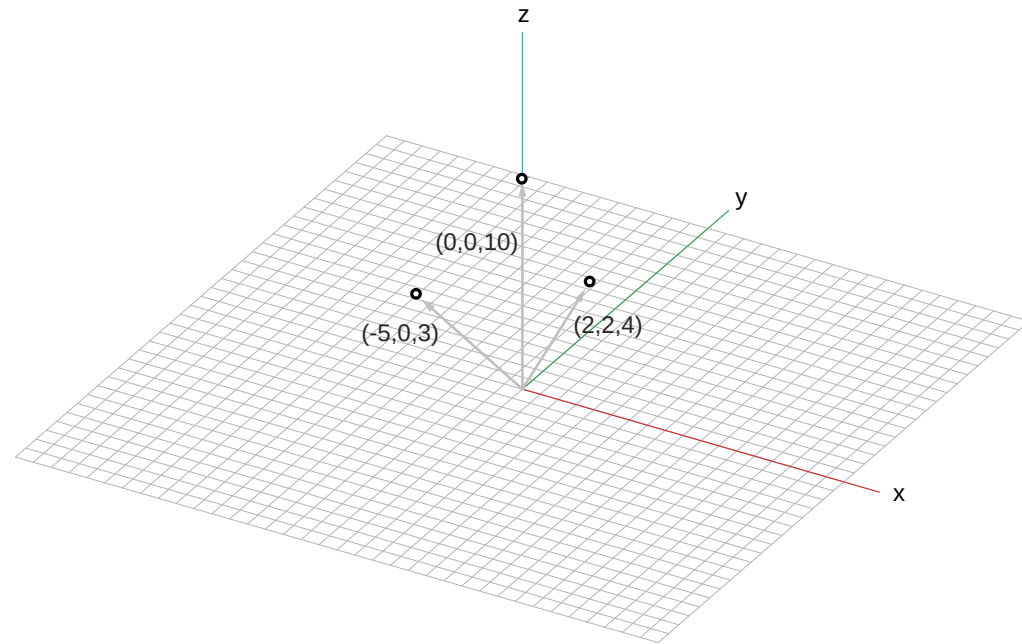


$v = (x, y, z)$



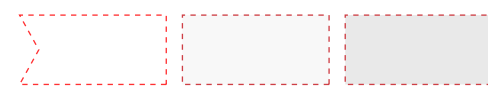
VECTOR

$$v = (x, y, z)$$

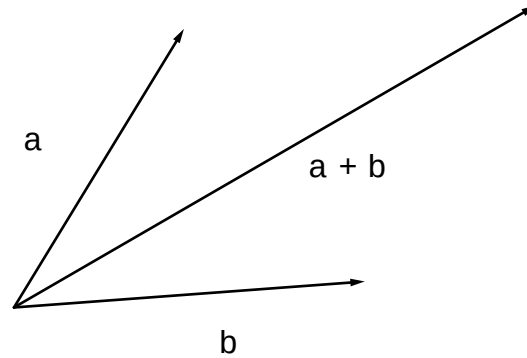


POINT

$$p = (x, y, z)$$



Definition

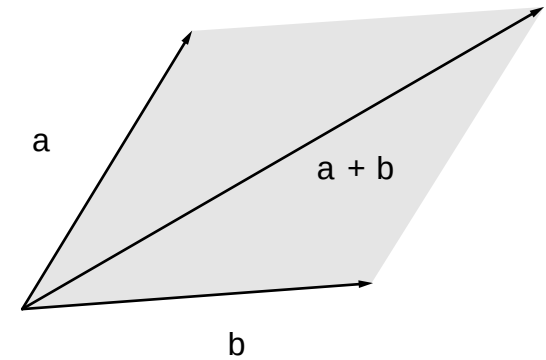
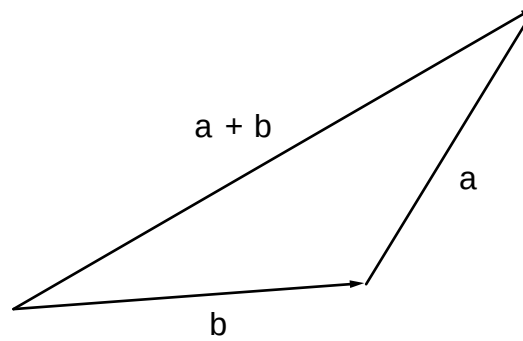
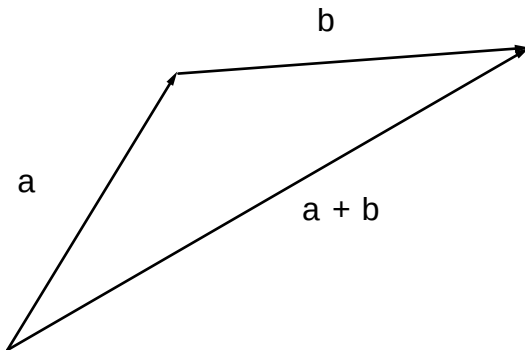


$$\mathbf{a} = (a_x, a_y, a_z)$$

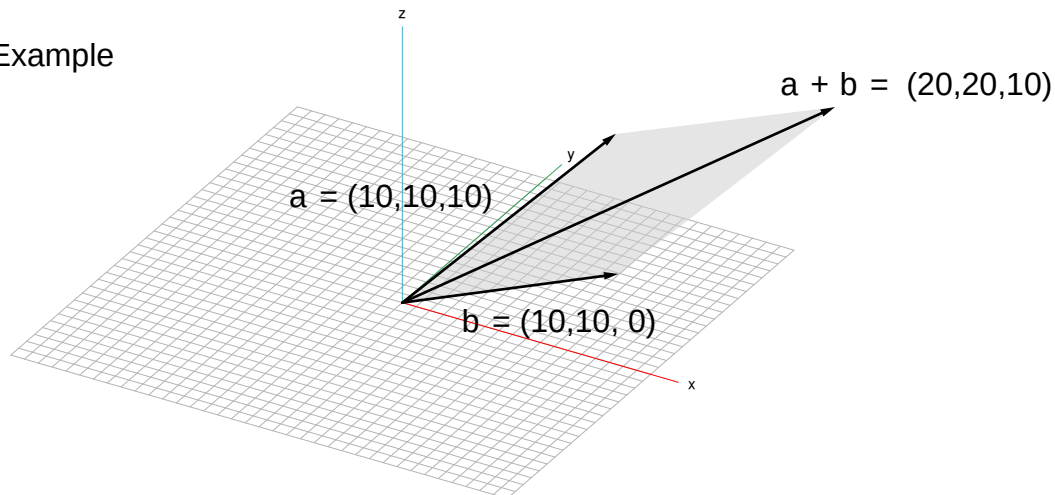
$$\mathbf{b} = (b_x, b_y, b_z)$$

$$\mathbf{a} + \mathbf{b} = (a_x + b_x, a_y + b_y, a_z + b_z)$$

Geometrical Interpretation



Example

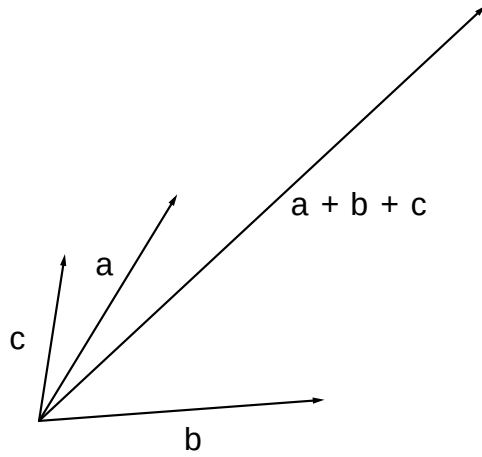


$$\mathbf{a} = (10, 10, 10)$$

$$\mathbf{b} = (10, 10, 0)$$

$$\mathbf{a} + \mathbf{b} = (10+10, 10+10, 10+0)$$

$$= (20, 20, 10)$$

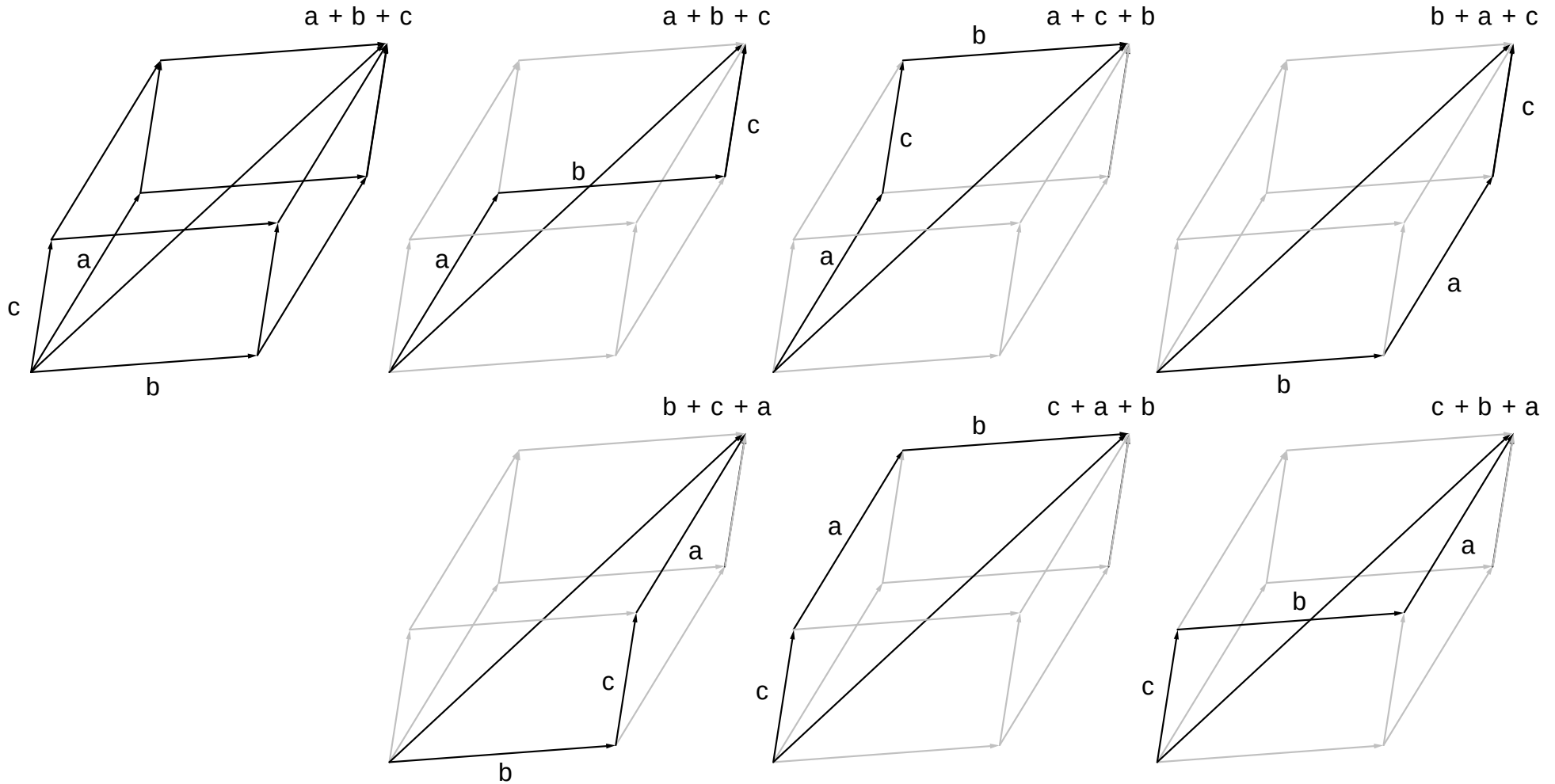


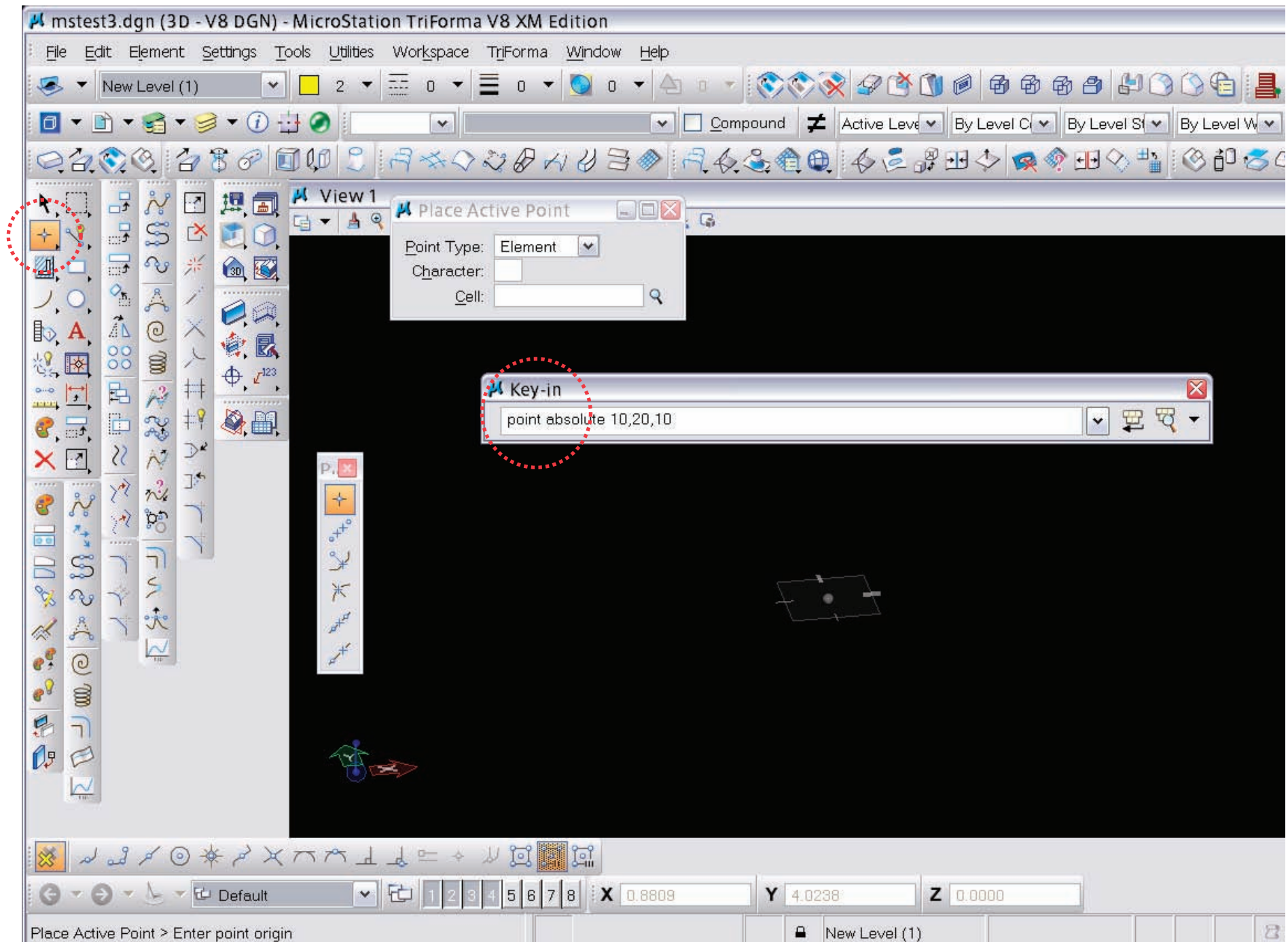
$$a = (a_x, a_y, a_z)$$

$$b = (b_x, b_y, b_z)$$

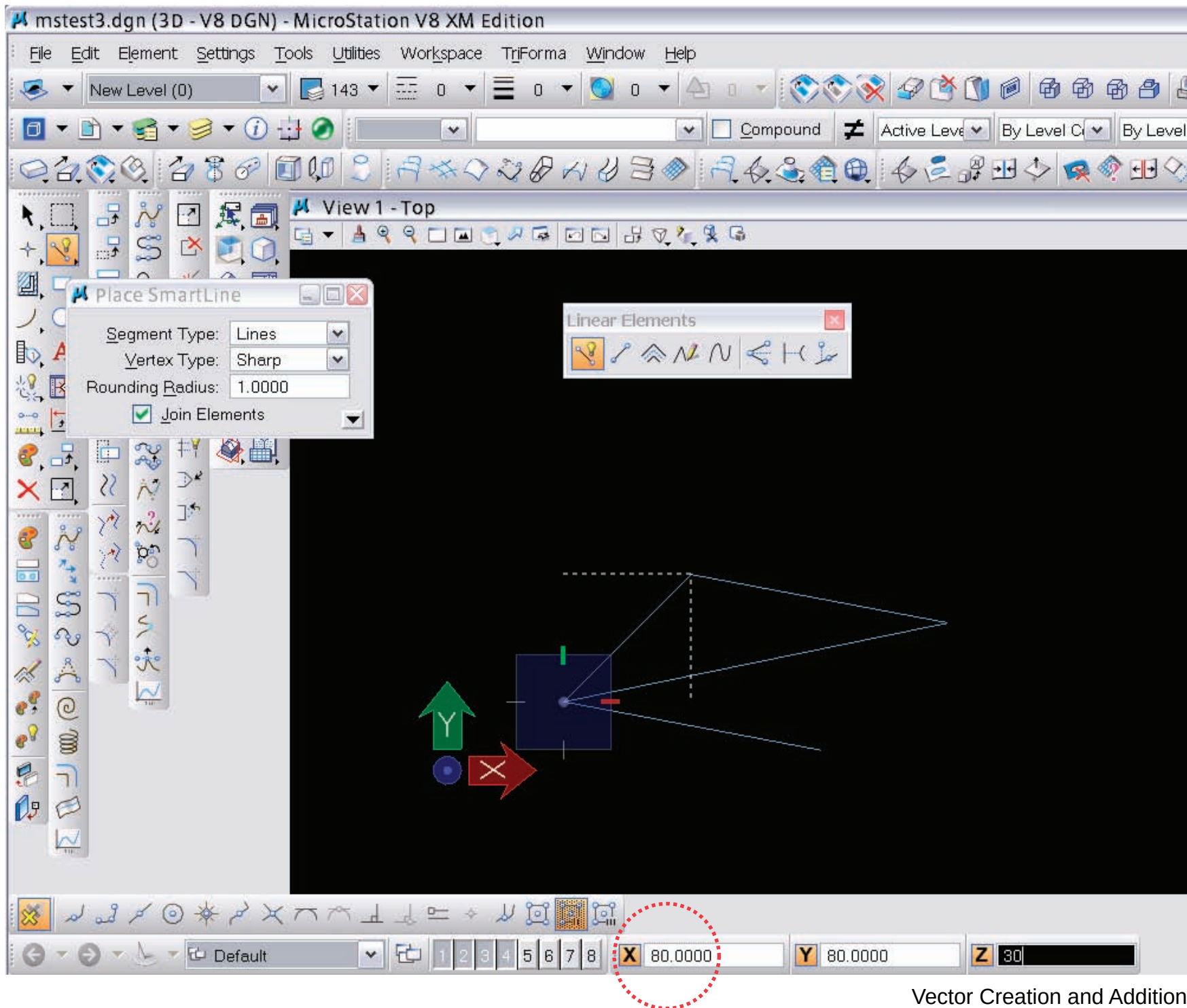
$$c = (c_x, c_y, c_z)$$

$$a + b + c = (a_x + b_x + c_x, a_y + b_y + c_y, a_z + b_z + c_z)$$

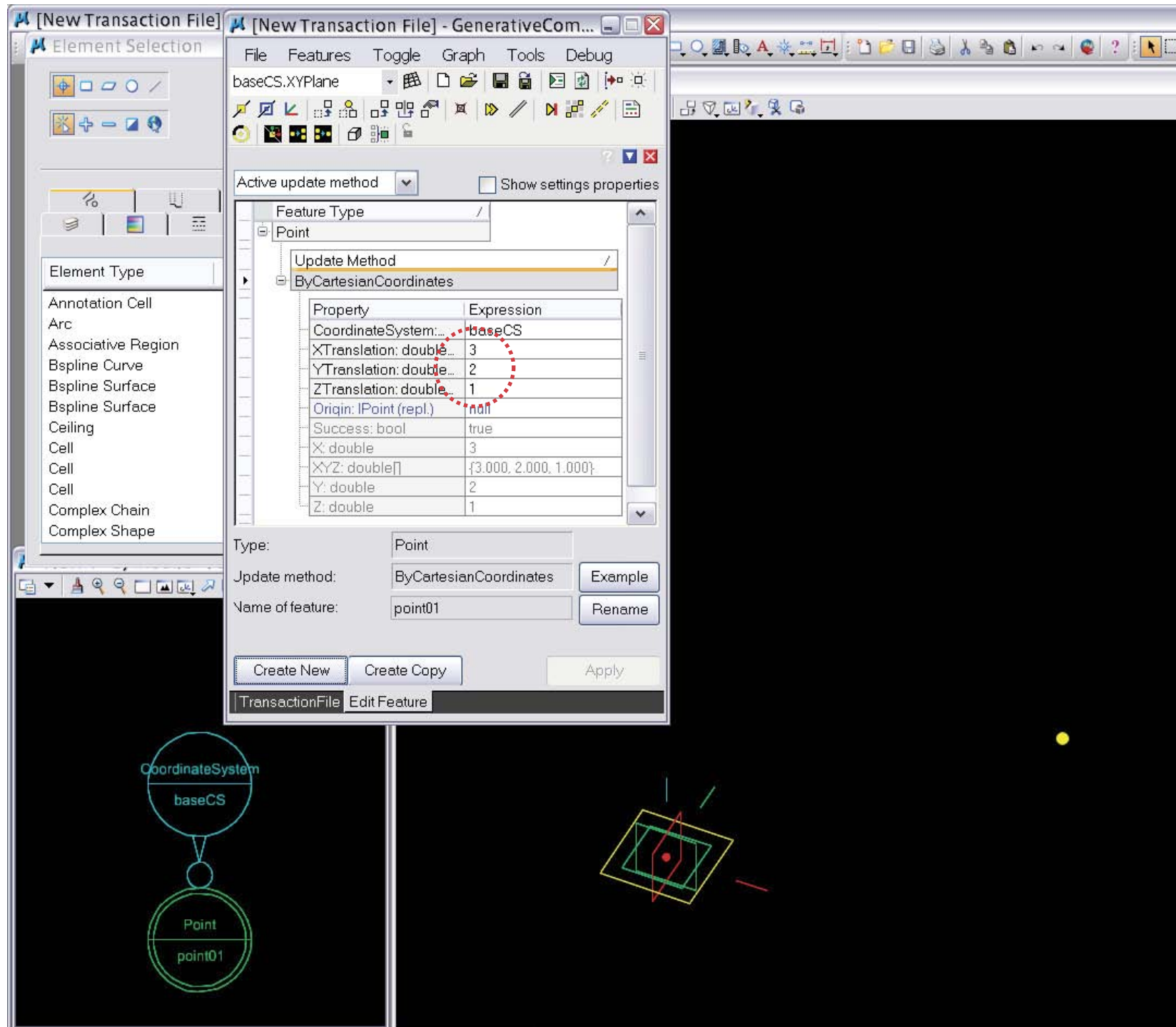




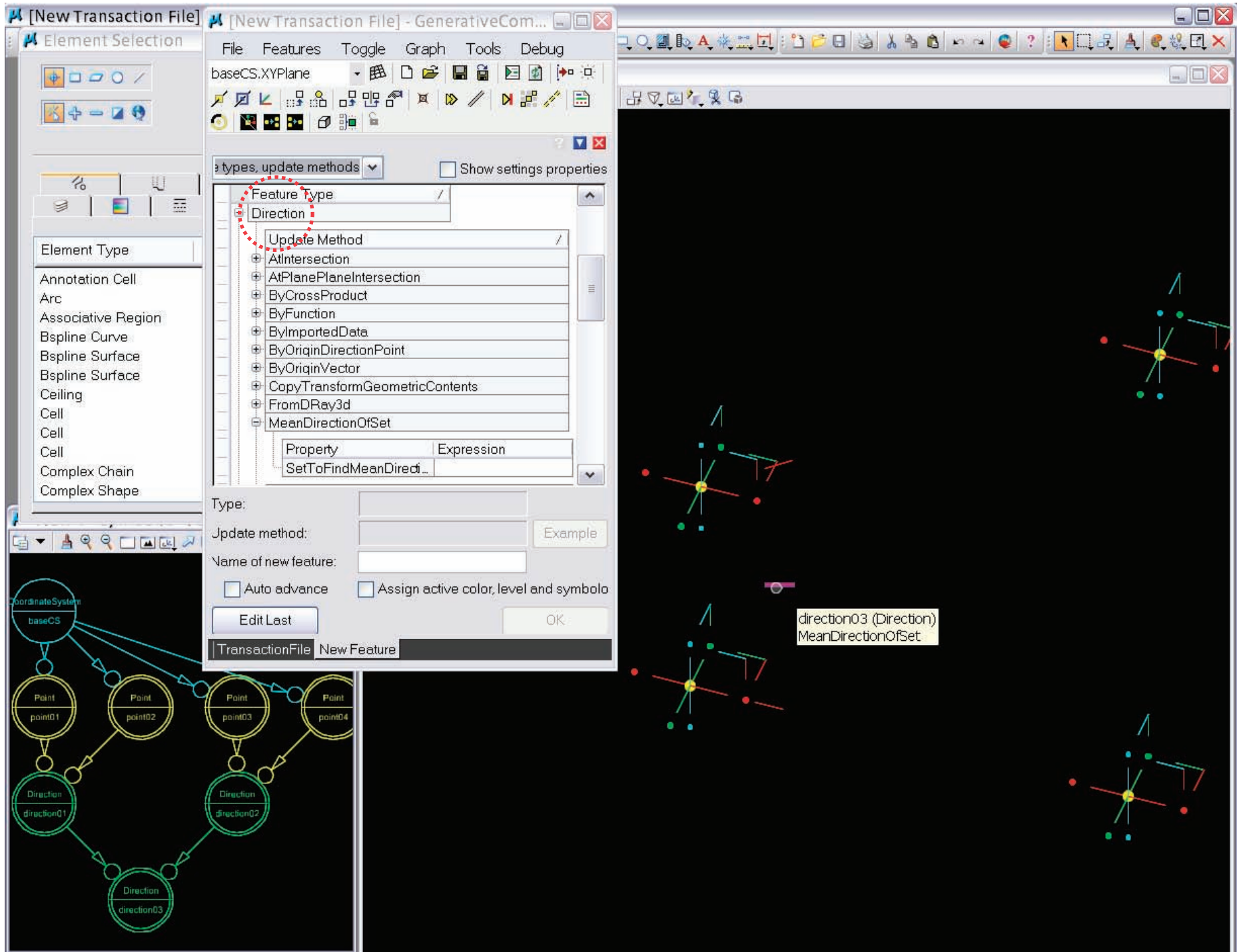
Point Creation by Coordinates



Vector Creation and Addition



Point Creation by Coordinates



The screenshot displays the GenerativeComponents V8 XM Edition software interface. The main window shows the 'Script Expression Builder' with the 'Geometry' tab selected. The 'Feature Dictionary' pane on the left lists various vector operations. The 'Feature types, methods' pane on the right shows a list of methods for the 'Geometry.DVector3d' type. A callout box highlights the 'new Geometry.DVector3d(double ax, double ay, double az)' constructor. Another callout box highlights the 'void SumOf(Geometry.DVector3d Origin, Geometry.DVector3d Vector)' method.

Feature Dictionary (Left Pane):

Type	Description
new Geometry.DVector3d()	Copy from a point.
new Geometry.DVector3d(Source)	Copy from a point.
new Geometry.DVector3d(Source)	Copy from DVector2d.
new Geometry.DVector3d(Origin, Target)	Construct a vector from origin to target
new Geometry.DVector3d(Origin, Target)	Construct a vector from origin to target
new Geometry.DVector3d(ax, ay)	Constructor from immediate xyz
new Geometry.DVector3d(Origin, Target)	Inplace vector addition.
new Geometry.DVector3d(ax, ay, az)	Angle from the instance vector to parameter...
AddInPlace	
AngleTo	
AngleToXY	
AngleXY	
AngleXYPlaneTowardsZ	
CrossProduct	
CrossProduct	
CrossProductToPoints	
CrossProductXY	
DifferenceOf	
DifferenceOf	
new Geometry.DVector3d(ax, ay, az)	

Feature types, methods (Right Pane):

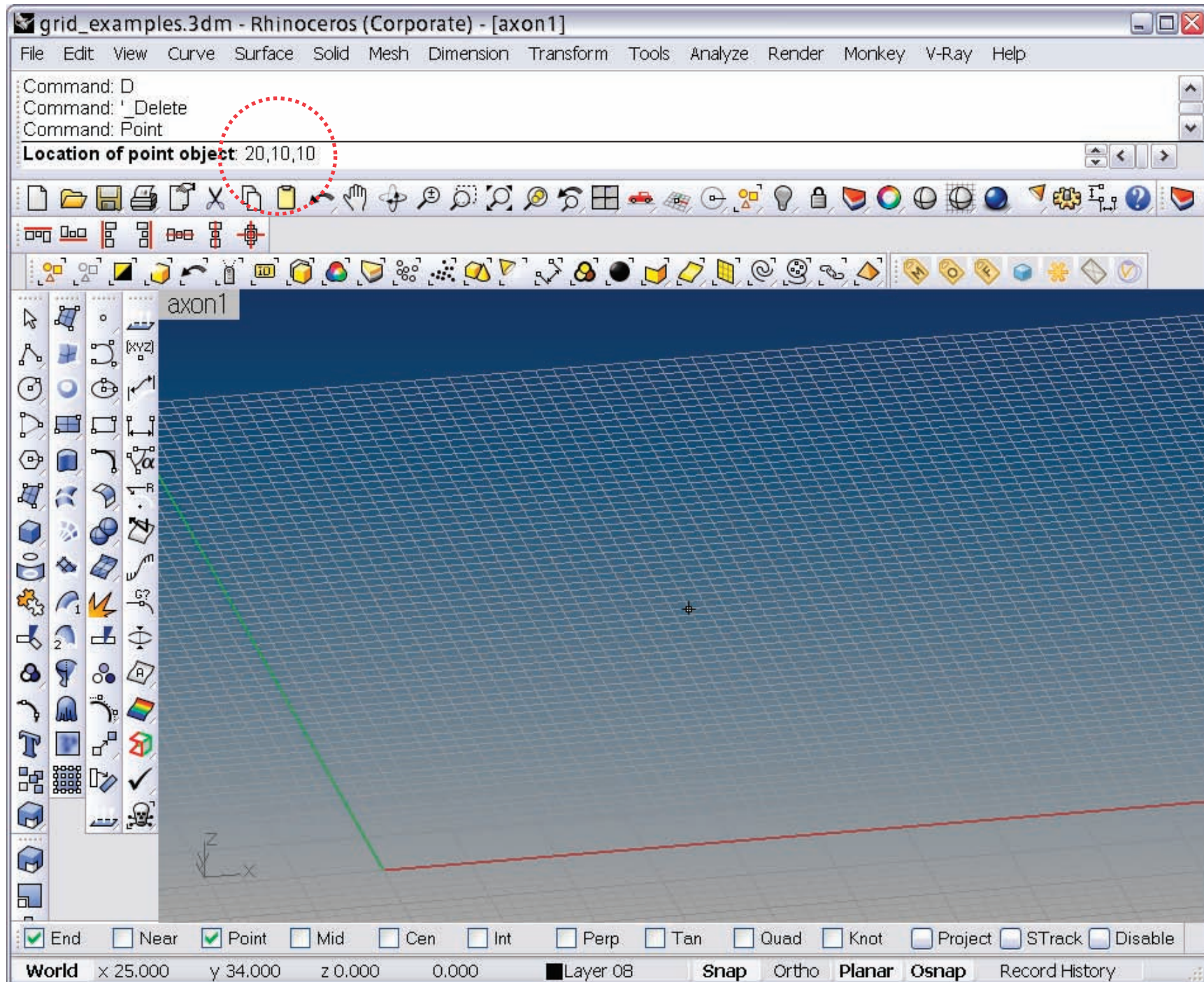
Type	Description
ProductOfTranspose	Replace the instance vector by the product of transposed Matrix...
ProductOfTranspose	Replace the instance vector by the product of the transposed matrix...
ProductOfTranspose	Replace the instance vector by a transposed matrix times a vector...
RotateXY	Rotate the instance vector in place around the z axis.
RotateXY	Replace the instance vector by Vector rotated by angle theta around...
Scale	Replace instance vector by scaled vector.
ScaleInPlace	Inplace scaling
ScaleToLength	Replace instance vector by contents scaled to given length...
ScaleToLength	Replace instance vector by Vector scaled to given length. Return...
SetComponent	Set a single component identified by cyclic index (0,1,2 = x,y,z).
SignedAngleTo	Angle from instance vector to parameter vector2, in their common...
SignedCrossProduct	Replace instance vector by cross product of Vector1 and Vector2...
SmallerUnorientedAngleTo	Smaller angle from the instance vector to the line containing...
SmallerUnorientedAngleToXY	
SubtractInPlace	
SumOf	Add an origin and vector. Result overwrites the instance.
SumOf	Sum an origin and the product of a matrix and vector. Place result in...
SumOf	Sum an origin and scaled vector. Place result in instance.
SumOf	Replace contents by sum of scaled __THISTYPE__'s
SumOf	Sum an origin and 2 scaled vectors. Place result in instance.
SumOf	Sum an origin and the product of a matrix and vector, with the...
SumOf	Replace contents by sum of scaled __THISTYPE__'s
SumOf	Sum an origin and 3 scaled vectors. Place result in instance.
ToString	Return String representation
TripleProduct	Return the triple product of the instance vector and two other vectors.

Script Expression Builder (Bottom Pane):

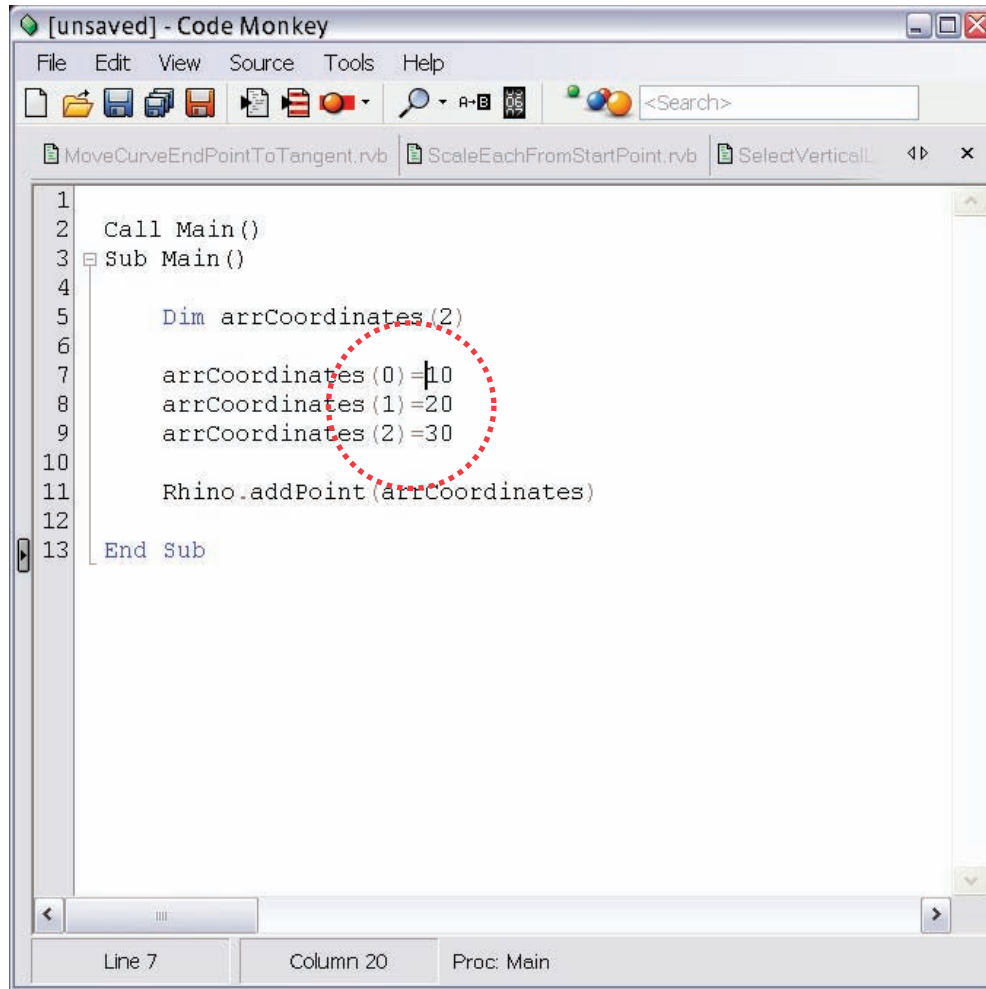
SumOf(Origin, Vector)

Buttons: Replace, Insert, Close

Vector Creation and Addition



Point Creation by Coordinates

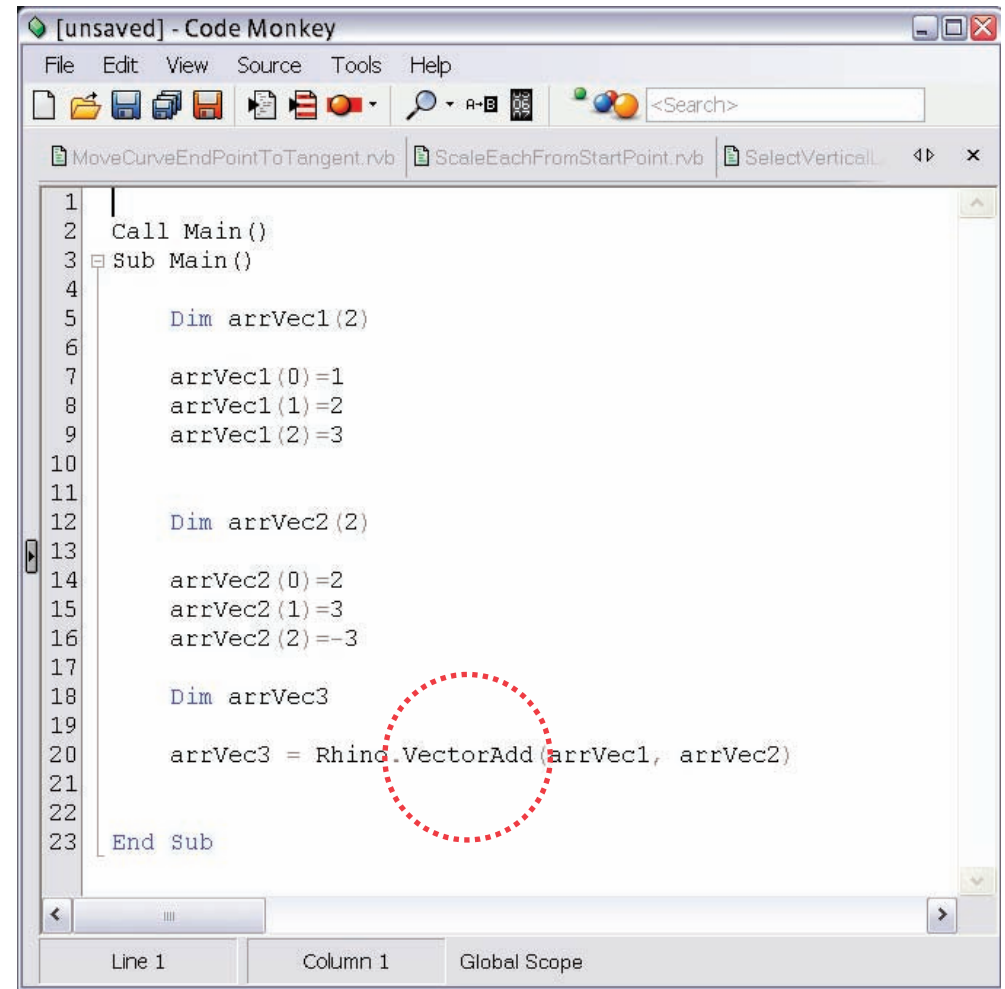


```

1
2 Call Main()
3 Sub Main()
4
5   Dim arrCoordinates(2)
6
7   arrCoordinates(0)=10
8   arrCoordinates(1)=20
9   arrCoordinates(2)=30
10
11   Rhino.addPoint(arrCoordinates)
12
13 End Sub
  
```

Line 7 Column 20 Proc: Main

Point Creation by Coordinates

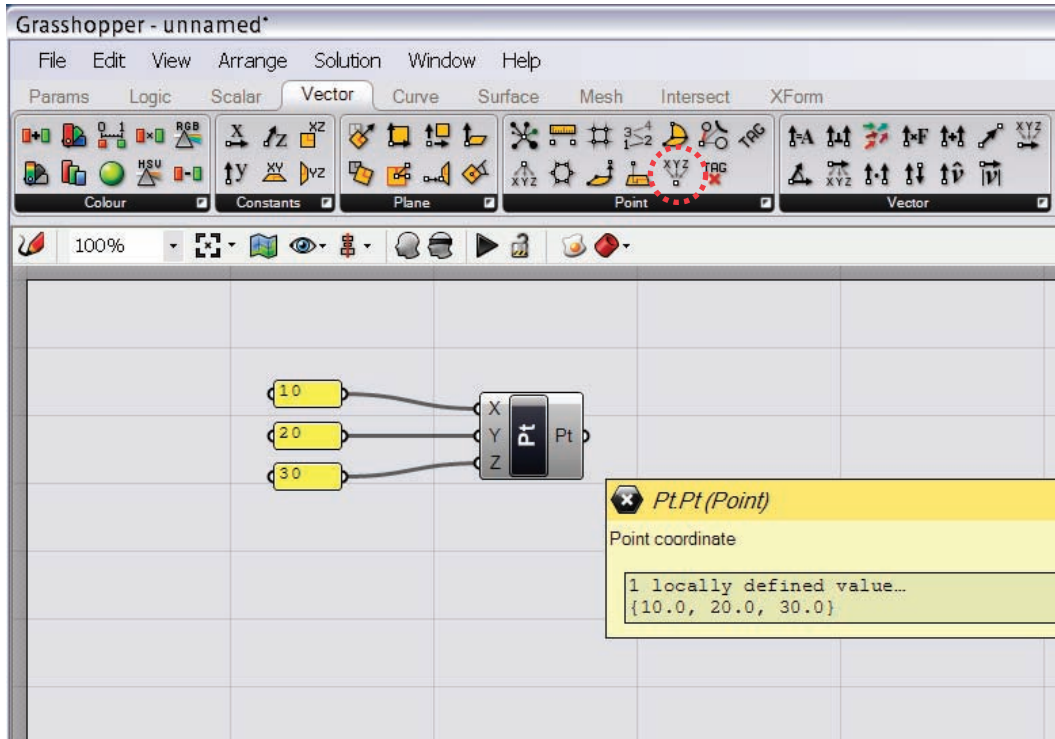


```

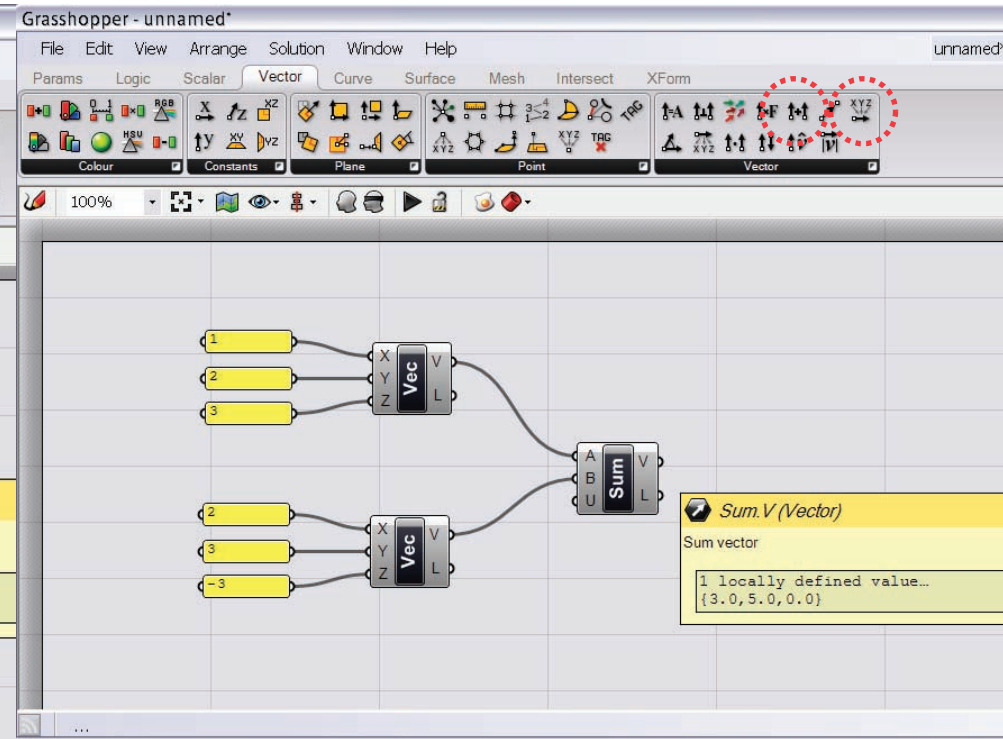
1
2 Call Main()
3 Sub Main()
4
5   Dim arrVec1(2)
6
7   arrVec1(0)=1
8   arrVec1(1)=2
9   arrVec1(2)=3
10
11
12   Dim arrVec2(2)
13
14   arrVec2(0)=2
15   arrVec2(1)=3
16   arrVec2(2)=-3
17
18   Dim arrVec3
19
20   arrVec3 = Rhino.VectorAdd(arrVec1, arrVec2)
21
22
23 End Sub
  
```

Line 1 Column 1 Global Scope

Vector Creation and Addition



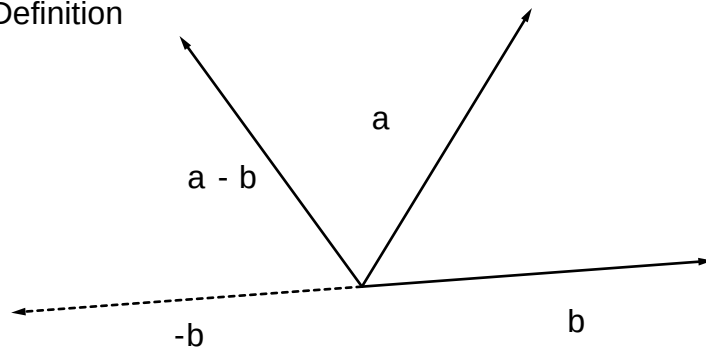
Point Creation by Coordinates



Vector Creation and Addition



Definition

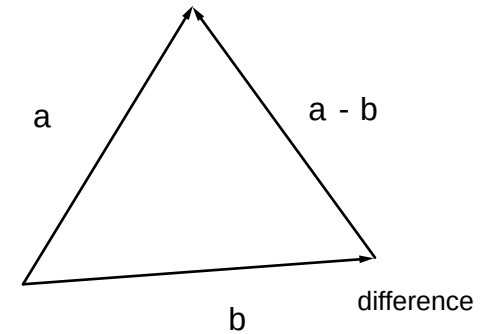
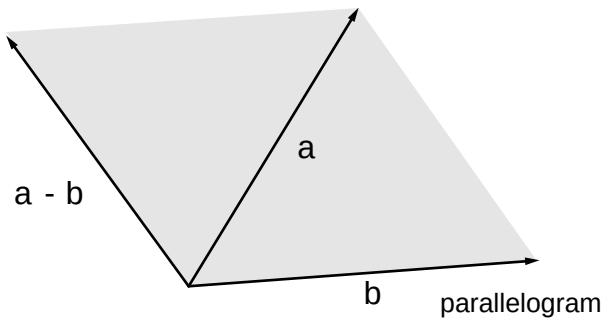
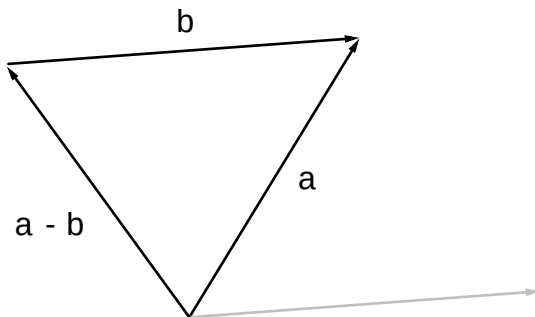


$$a = (a_x, a_y, a_z)$$

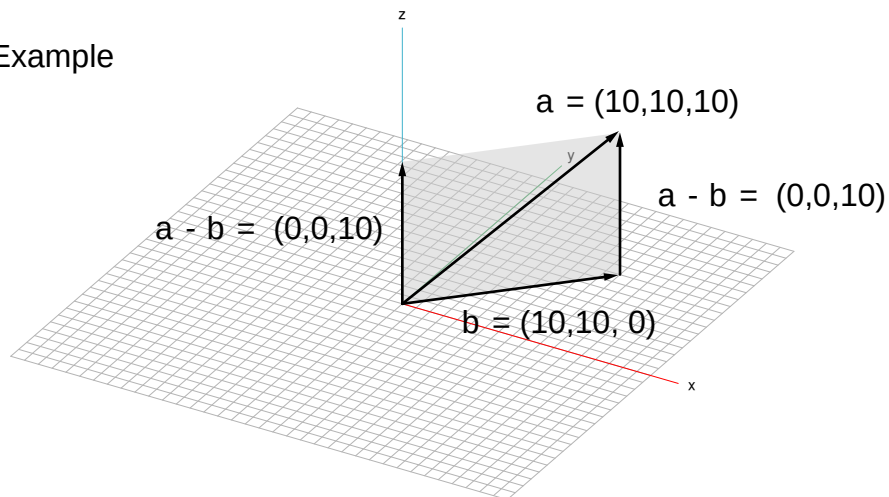
$$b = (b_x, b_y, b_z)$$

$$a - b = (a_x - b_x, a_y - b_y, a_z - b_z)$$

Geometrical Interpretation



Example



$$a = (10, 10, 10)$$

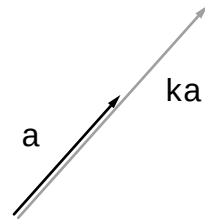
$$b = (10, 10, 0)$$

$$a - b = (10 - 10, 10 - 10, 10 - 0)$$

$$= (0, 0, 10)$$



Definition

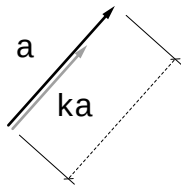


$$a = (a_x, a_y, a_z)$$

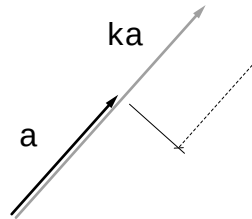
$$ka = (ka_x, ka_y, ka_z)$$

Geometrical Interpretation

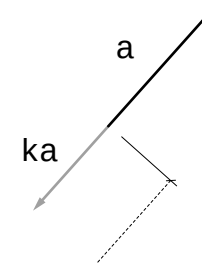
$$0 < k < 1$$



$$k > 1$$



$$k < 0$$

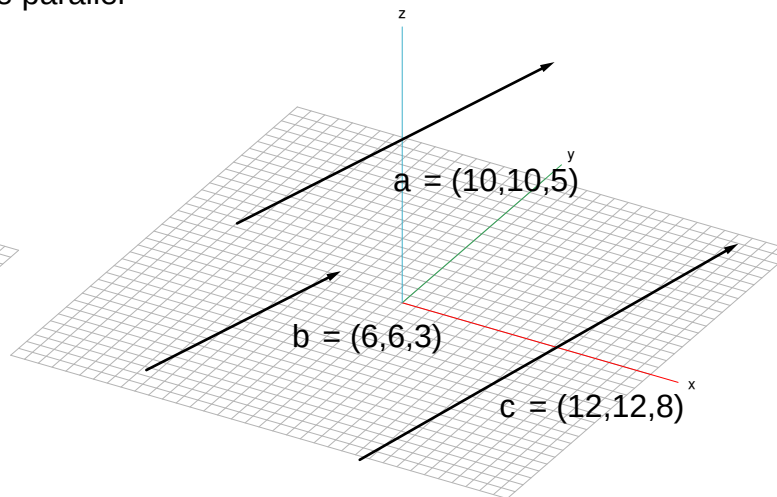
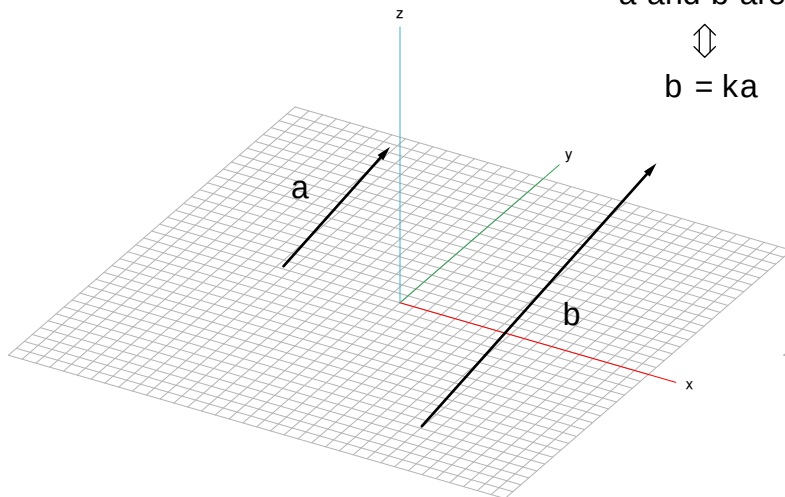


Parallel Vectors

a and b are parallel

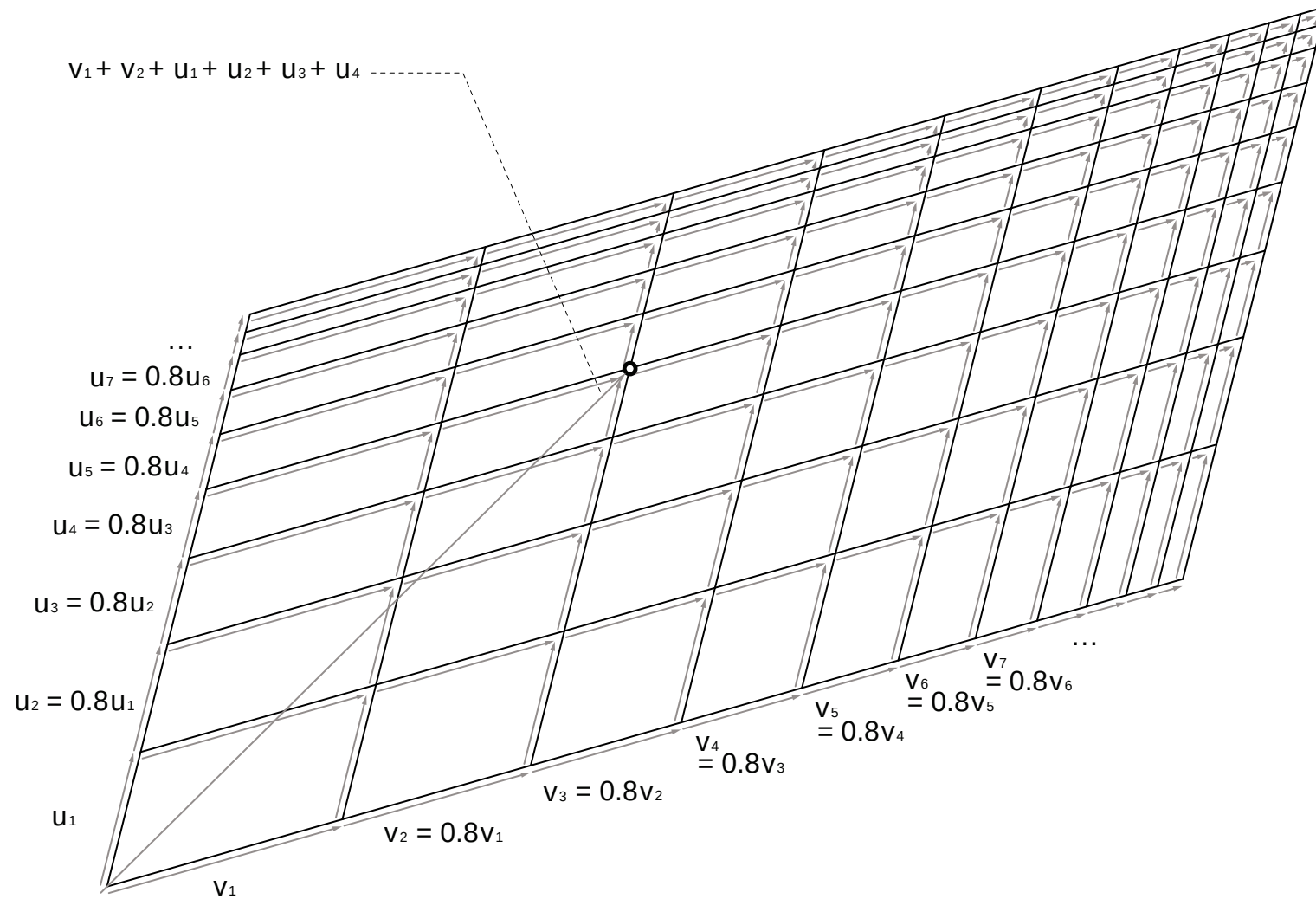


$$b = ka$$

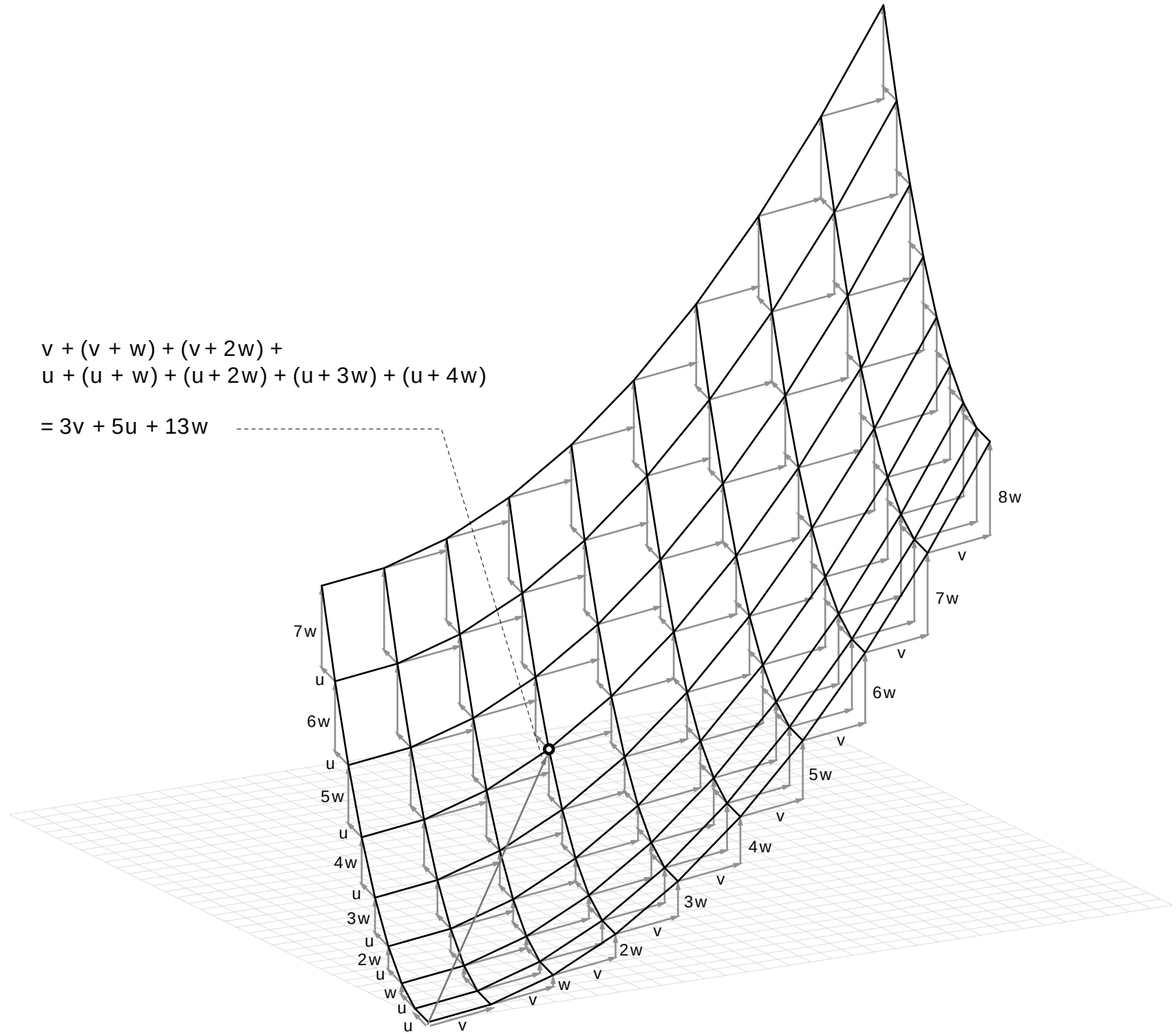


$$\begin{aligned} b &= (6, 6, 3) \\ &= (10 \times 0.6, 10 \times 0.6, 5 \times 0.6) \\ &= 0.6 \times (10, 10, 5) \\ &= 0.6 \times a \\ &\Rightarrow a \text{ and } b \text{ are parallel} \end{aligned}$$

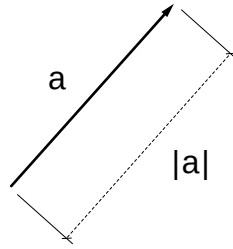
$$\begin{aligned} c &= (12, 12, 8) \\ &= (10 \times 1.2, 10 \times 1.2, 5 \times \underline{1.6}) \\ &\neq 1.2 \times a \\ &\Rightarrow a \text{ and } c \text{ are not parallel} \end{aligned}$$



$$\begin{aligned}
 &v + (v + w) + (v + 2w) + \\
 &u + (u + w) + (u + 2w) + (u + 3w) + (u + 4w) \\
 &= 3v + 5u + 13w
 \end{aligned}$$



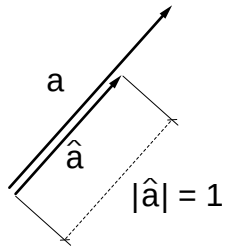
Length



$$\mathbf{a} = (a_x, a_y, a_z)$$

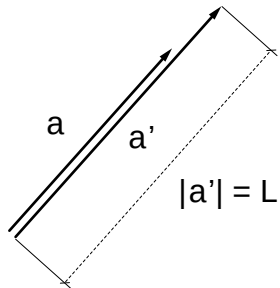
$$|\mathbf{a}| = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

Unitization

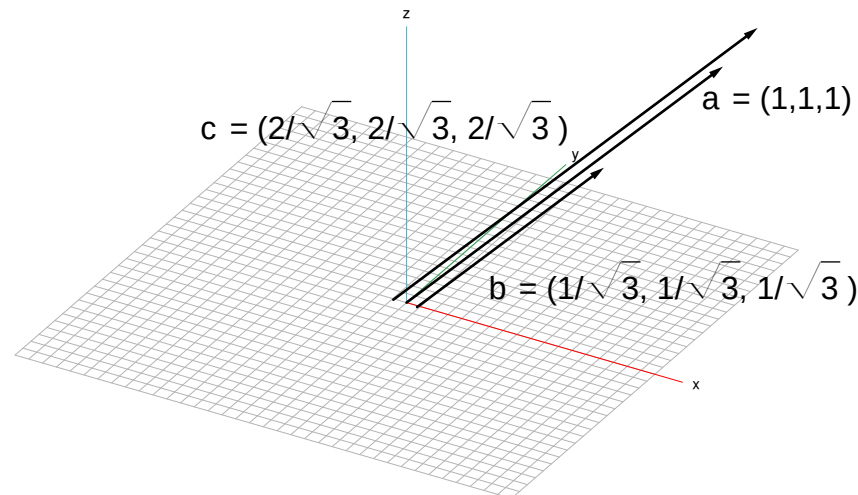


$$\hat{\mathbf{a}} = \mathbf{a}/|\mathbf{a}|$$

Setting Length



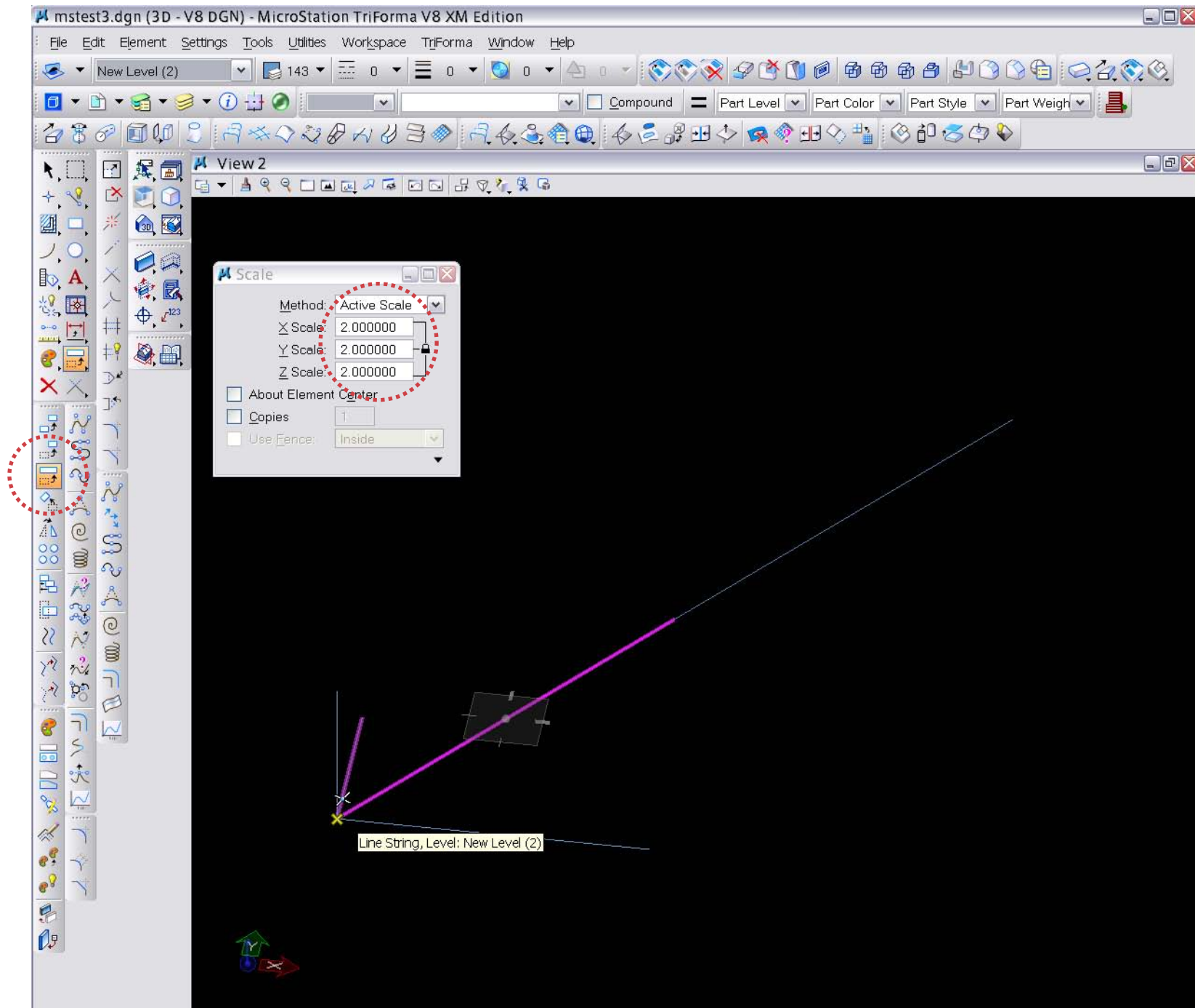
$$\mathbf{a}' = \hat{\mathbf{a}} \times L = \mathbf{a}/|\mathbf{a}| \times L$$



$$|\mathbf{a}| = \sqrt{1 + 1 + 1} = \sqrt{3}$$

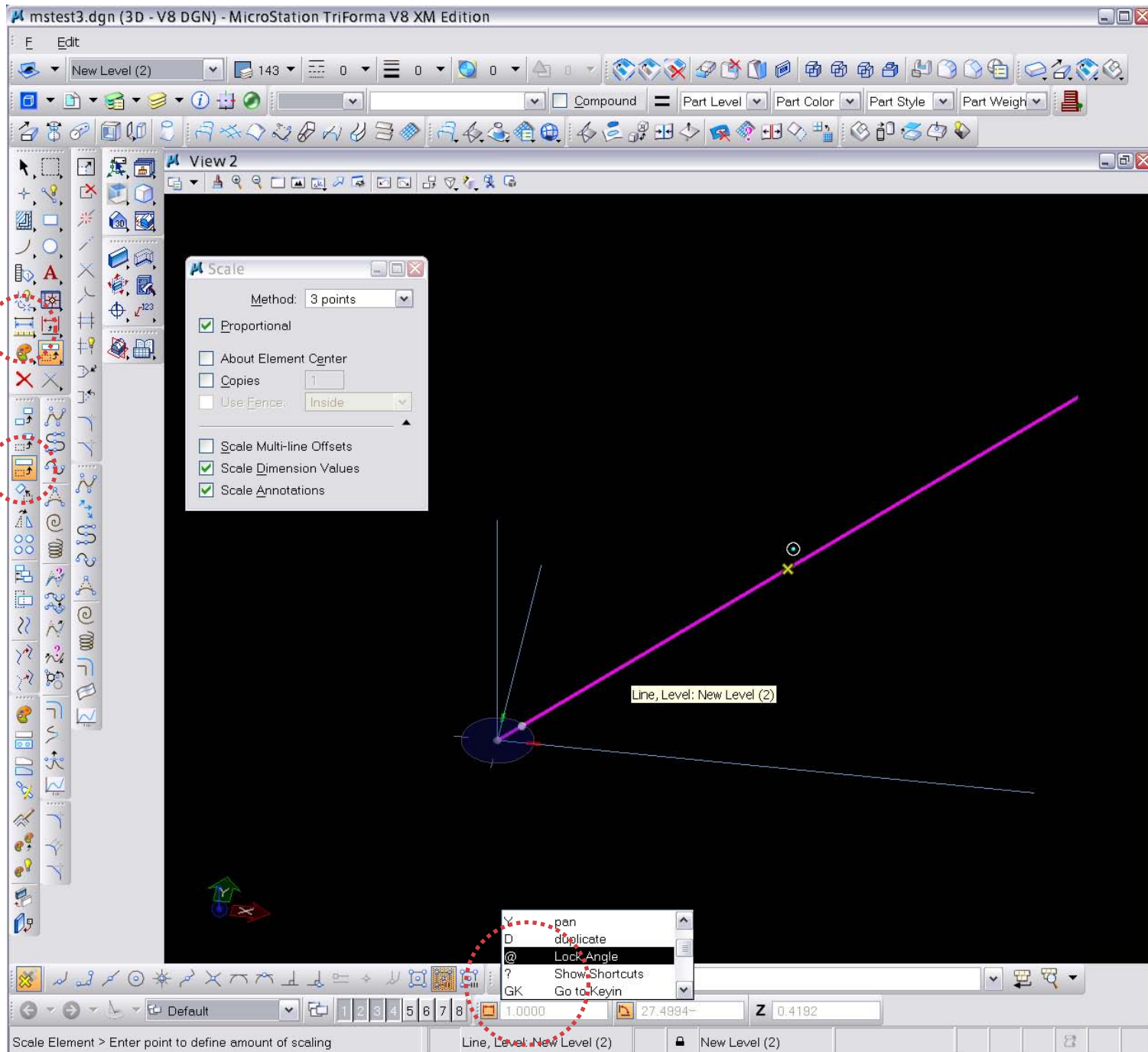
$$|\mathbf{b}| = 1$$

$$|\mathbf{c}| = 2$$

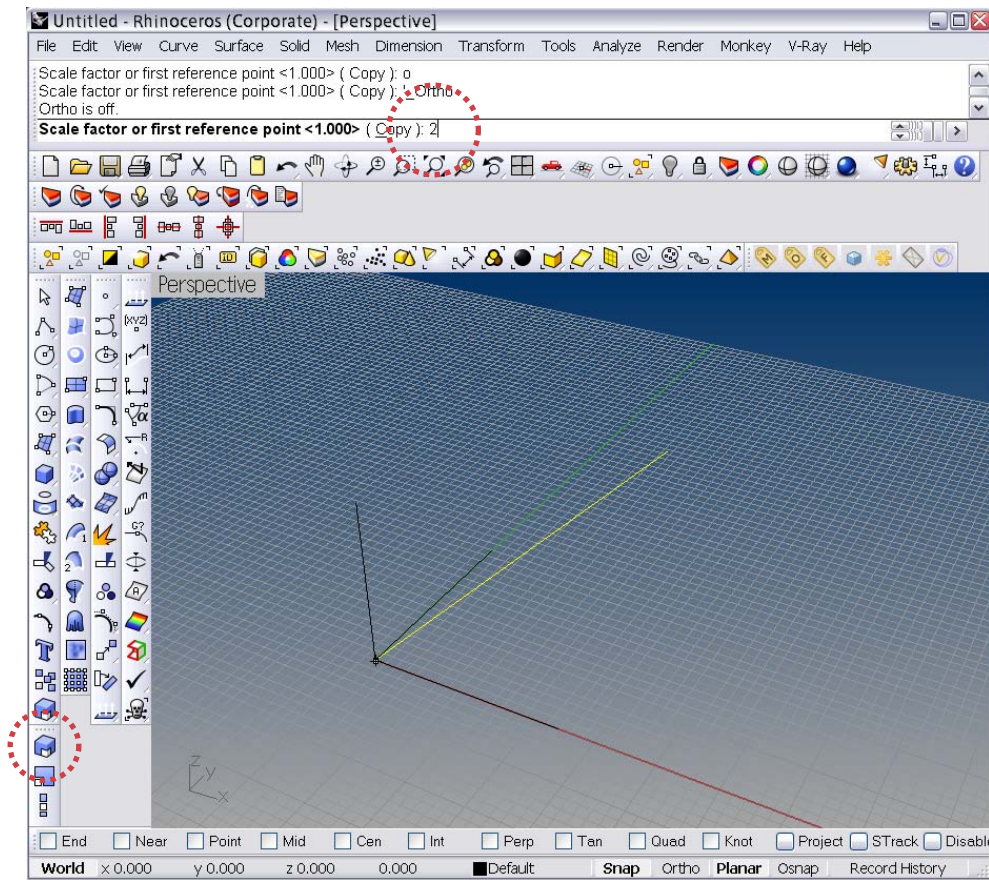


Scaling Vector (Line)

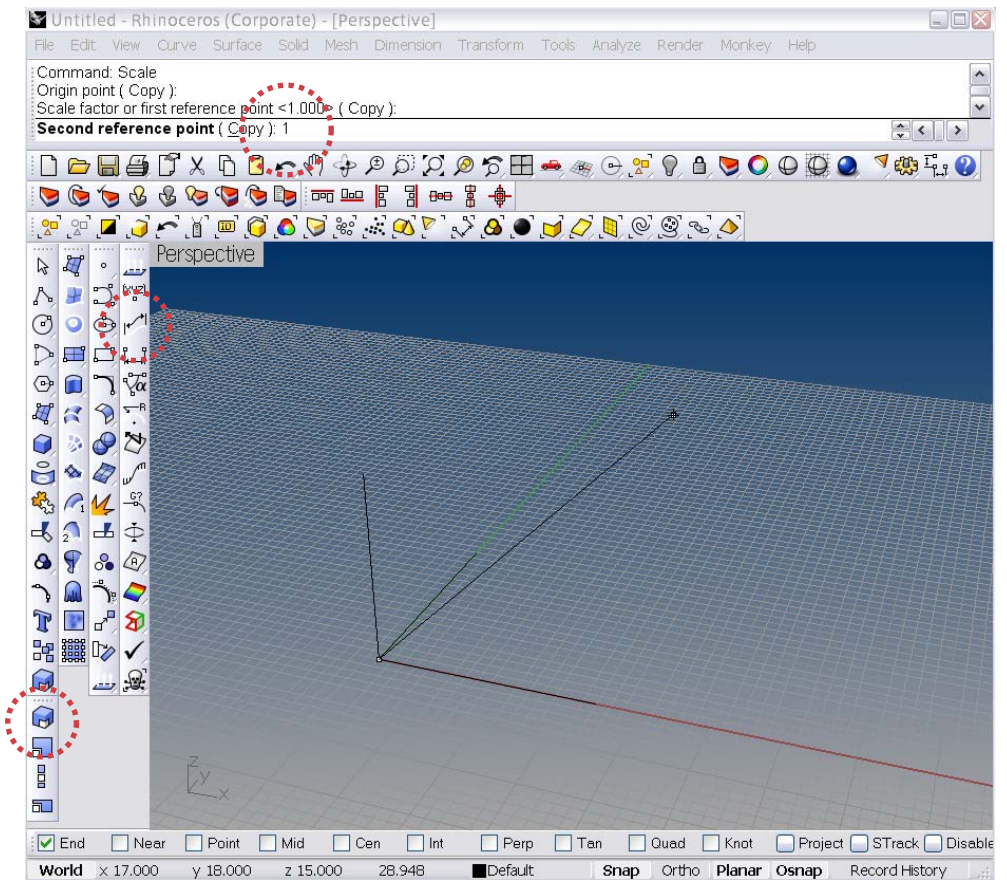
Length



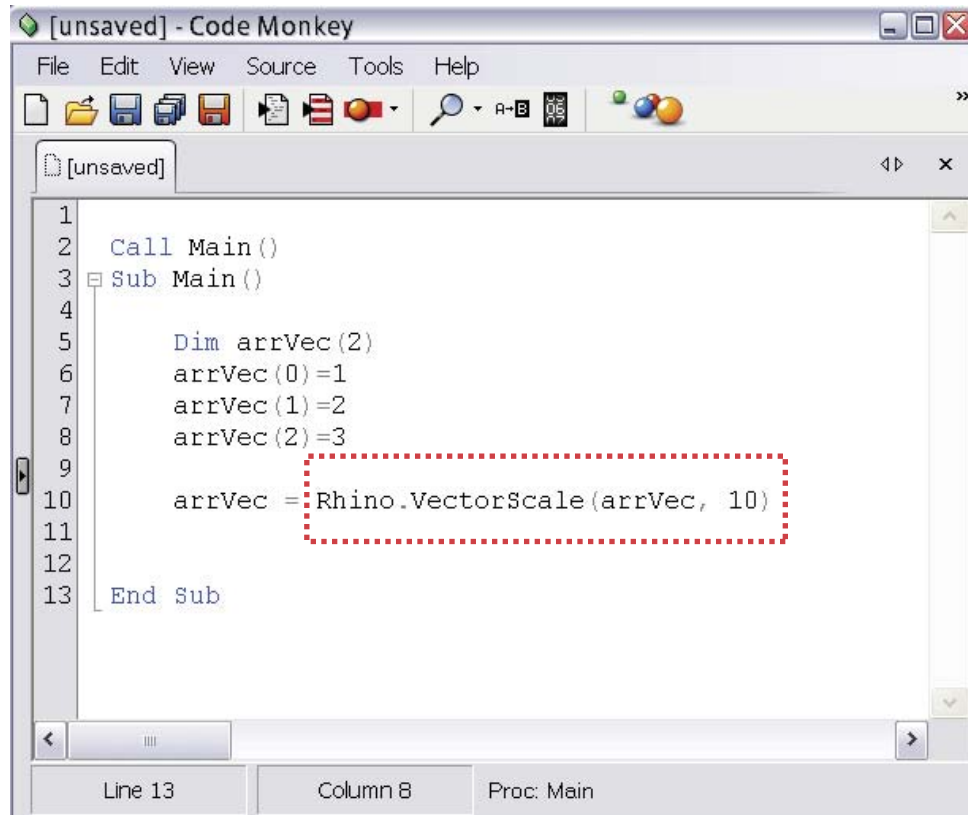
Unitizing Vector (Line)
and Vector Length



Scaling Vector (Line)



Unitizing Vector (Line)
and Vector Length

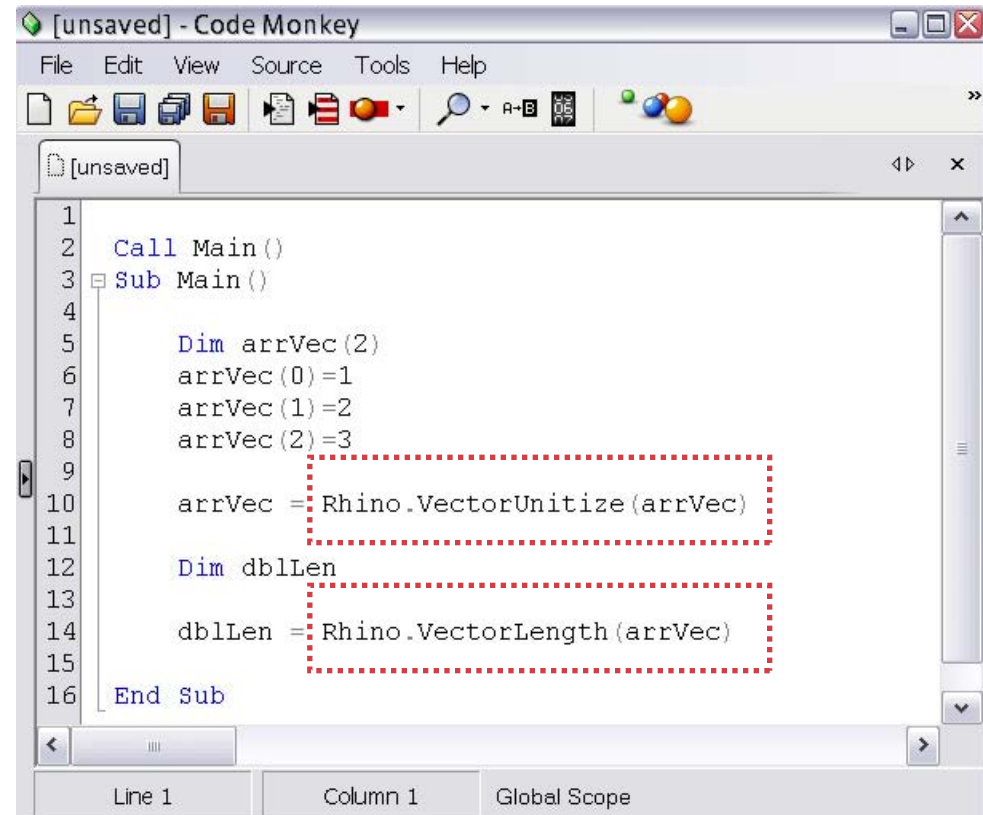


```

1
2 Call Main()
3 Sub Main()
4
5   Dim arrVec(2)
6   arrVec(0)=1
7   arrVec(1)=2
8   arrVec(2)=3
9
10  arrVec = Rhino.VectorScale(arrVec, 10)
11
12
13 End Sub
  
```

Line 13 Column 8 Proc: Main

Scaling Vector

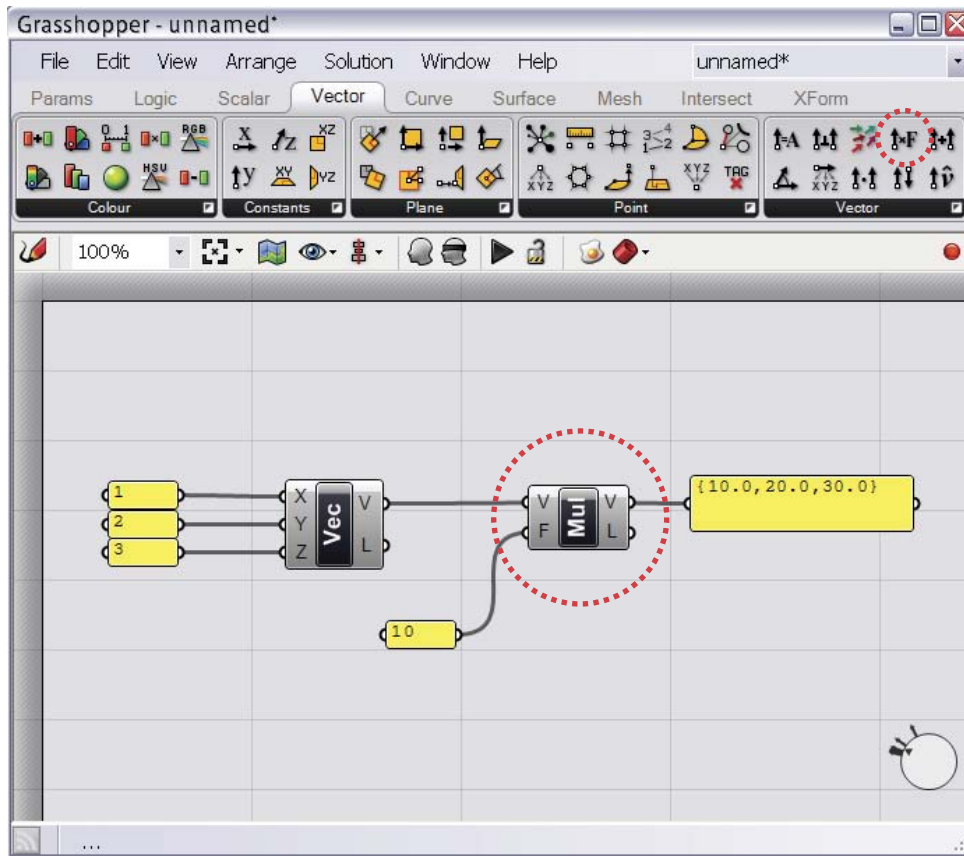


```

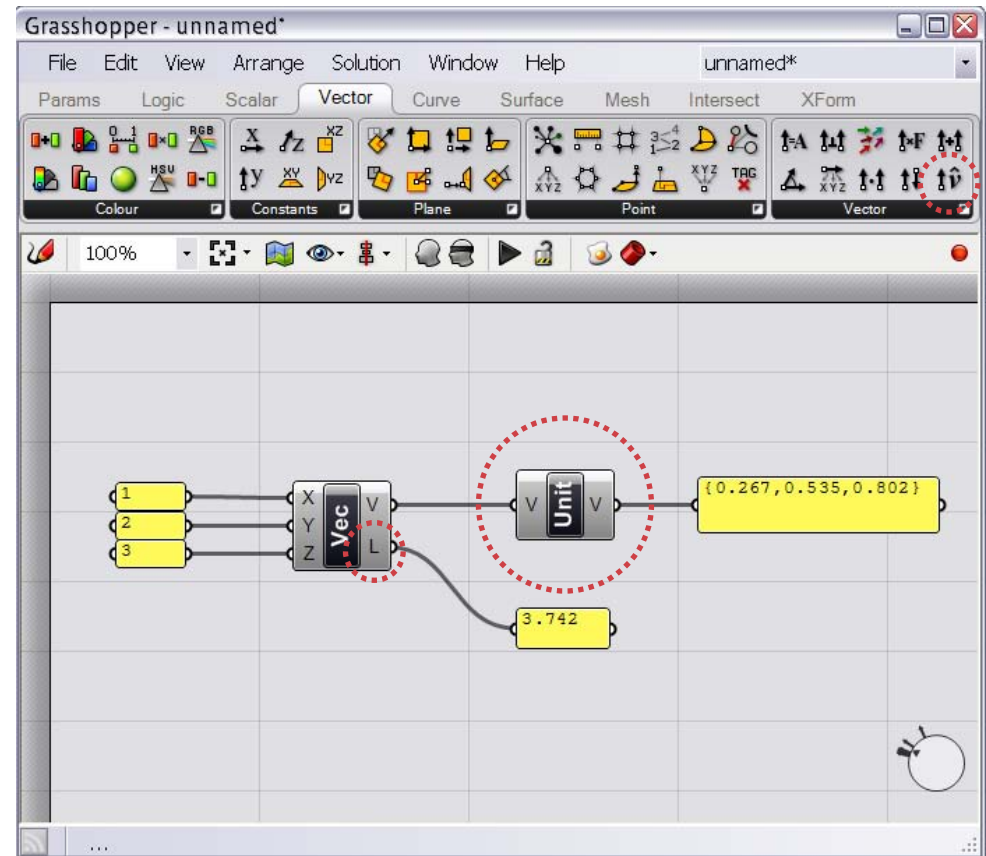
1
2 Call Main()
3 Sub Main()
4
5   Dim arrVec(2)
6   arrVec(0)=1
7   arrVec(1)=2
8   arrVec(2)=3
9
10  arrVec = Rhino.VectorUnitize(arrVec)
11
12  Dim dblLen
13
14  dblLen = Rhino.VectorLength(arrVec)
15
16 End Sub
  
```

Line 1 Column 1 Global Scope

Unitizing Vector
and Vector Length



Scaling Vector



Unitizing Vector
and Vector Length



Definition

$$\mathbf{a} = (a_x, a_y, a_z)$$

$$\mathbf{b} = (b_x, b_y, b_z)$$

$$\mathbf{a} \cdot \mathbf{b} = a_x \times b_x + a_y \times b_y + a_z \times b_z$$

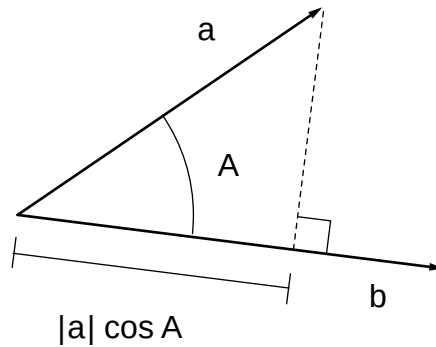
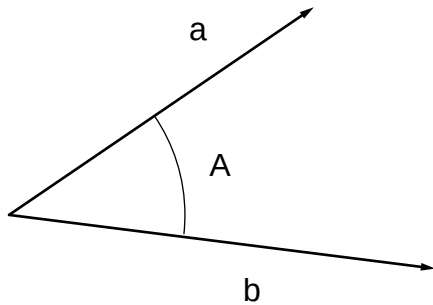
Dot Product

(Inner Product)

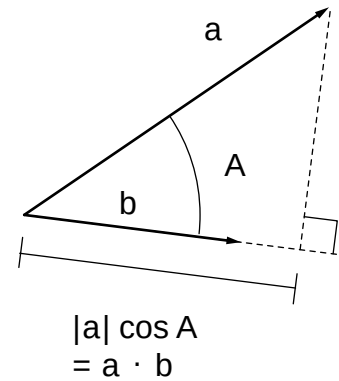
(Scalar Product)

Geometrical Interpretation

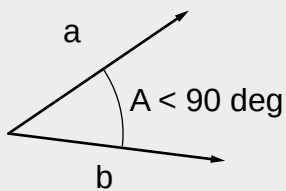
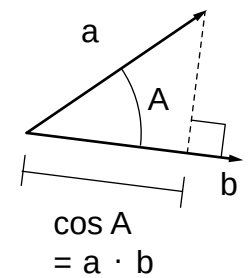
$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}| |\mathbf{b}| \cos A$$



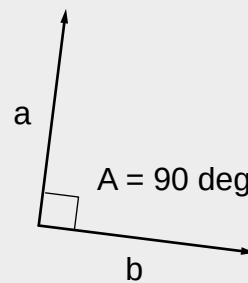
if $|\mathbf{b}| = 1$



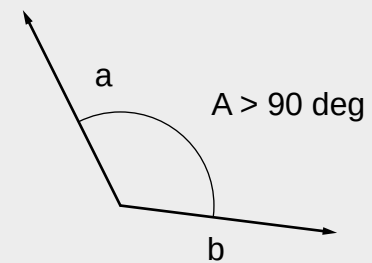
if $|\mathbf{a}| = 1, |\mathbf{b}| = 1$



$$0 < \mathbf{a} \cdot \mathbf{b}$$



$$\mathbf{a} \cdot \mathbf{b} = 0$$



$$\mathbf{a} \cdot \mathbf{b} < 0$$



Definition

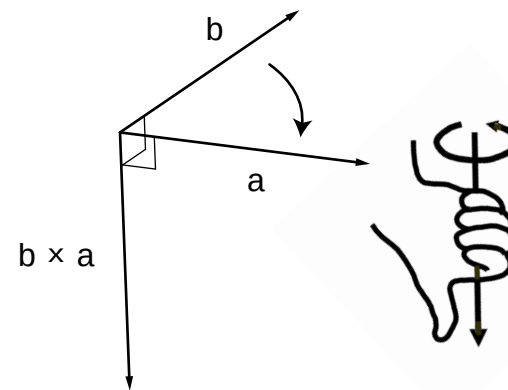
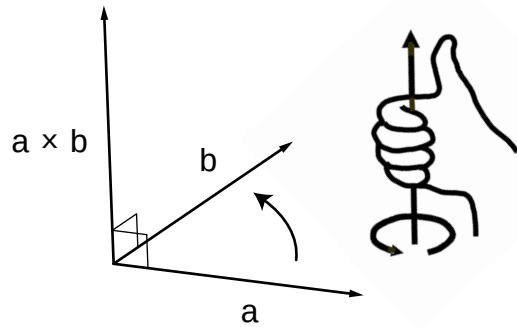
$$\mathbf{a} = (a_x, a_y, a_z) \quad \mathbf{b} = (b_x, b_y, b_z)$$

Cross Product
 (Outer Product)
 (Vector Product)

$$\mathbf{a} \times \mathbf{b} = \begin{pmatrix} a_y \times b_z - a_z \times b_y \\ a_z \times b_x - a_x \times b_z \\ a_x \times b_y - a_y \times b_x \end{pmatrix}$$

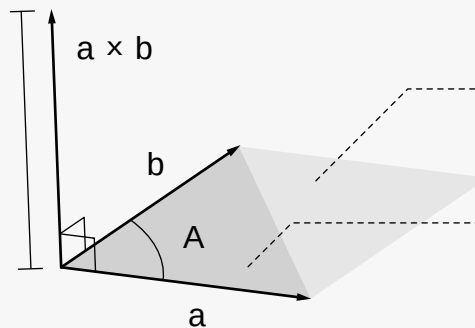
Geometrical Interpretation

- Perpendicular Direction



- Length & Area

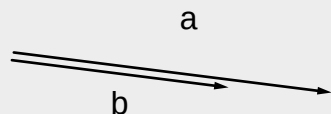
$$|\mathbf{a} \times \mathbf{b}| = |\mathbf{a}| |\mathbf{b}| \sin A$$



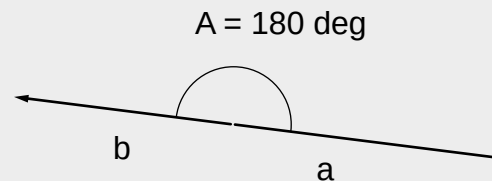
$$\text{Area of Parallelogram} = |\mathbf{a}| |\mathbf{b}| \sin A = |\mathbf{a} \times \mathbf{b}|$$

$$\text{Area of Triangle} = \frac{1}{2} \times |\mathbf{a}| |\mathbf{b}| \sin A = |\mathbf{a} \times \mathbf{b}| / 2$$

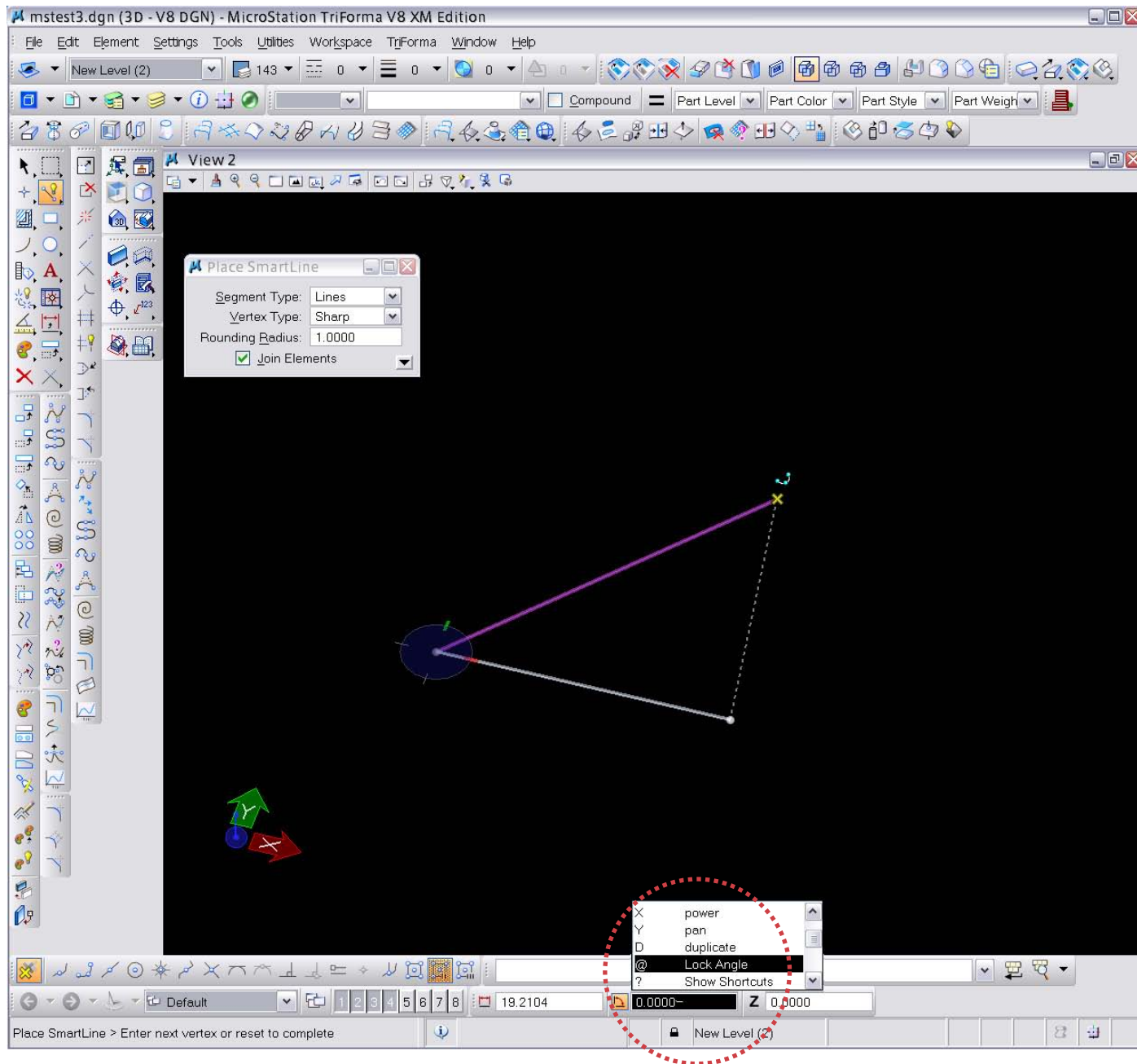
- Parallel



$$|\mathbf{a} \times \mathbf{b}| = 0$$

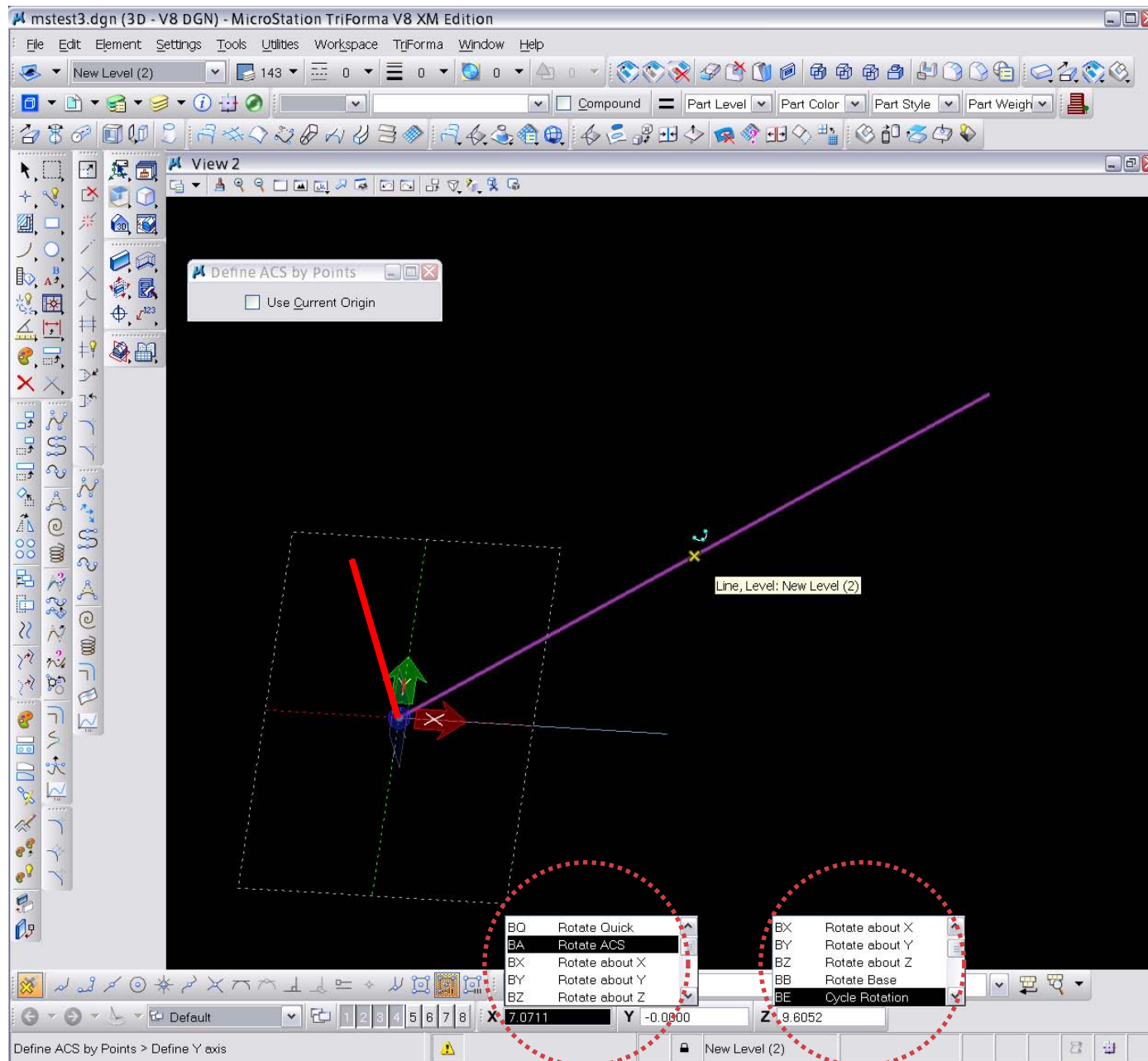


$$|\mathbf{a} \times \mathbf{b}| = 0$$

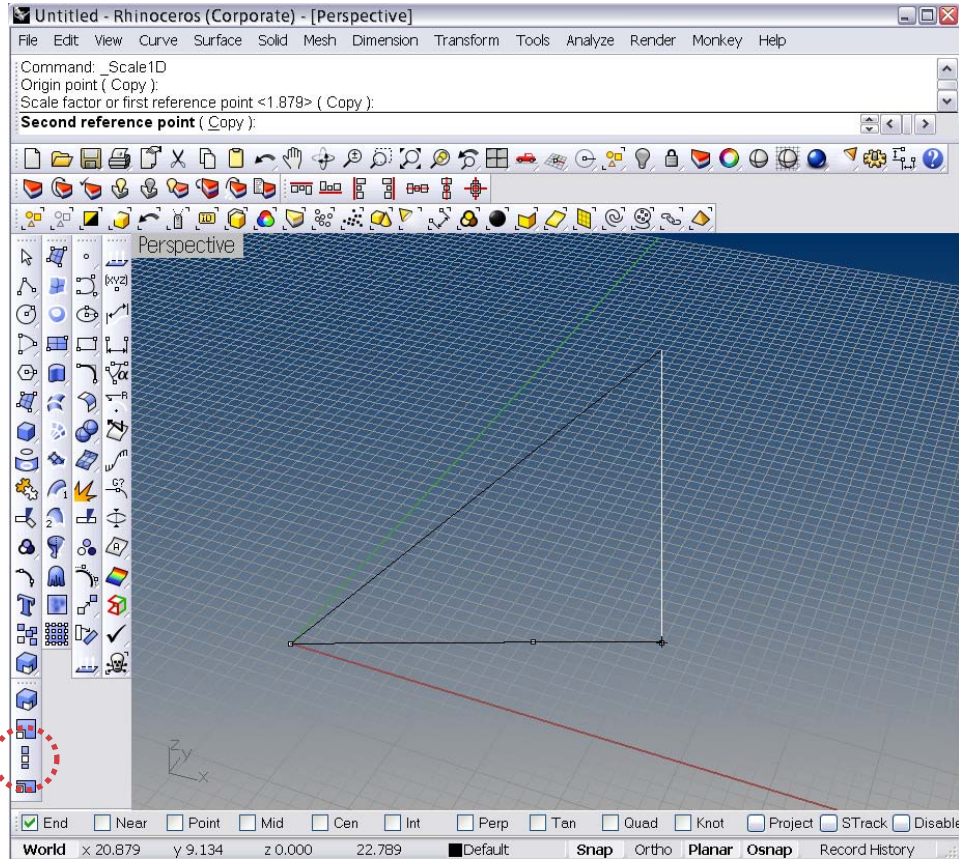


Dot Product as Line Projection

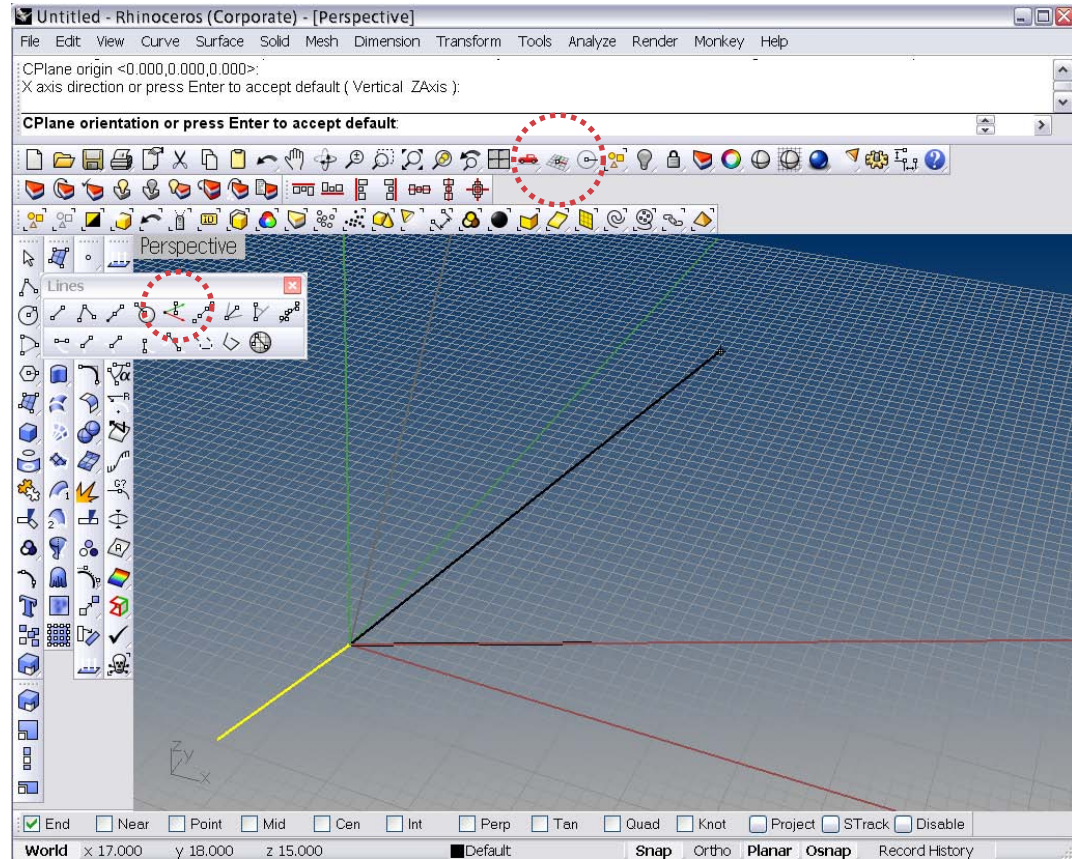
MATHEMATICS FOR SYSTEMATIC MODELING_1 POINT AND VECTOR OPERATIONS IN MICROSTATION



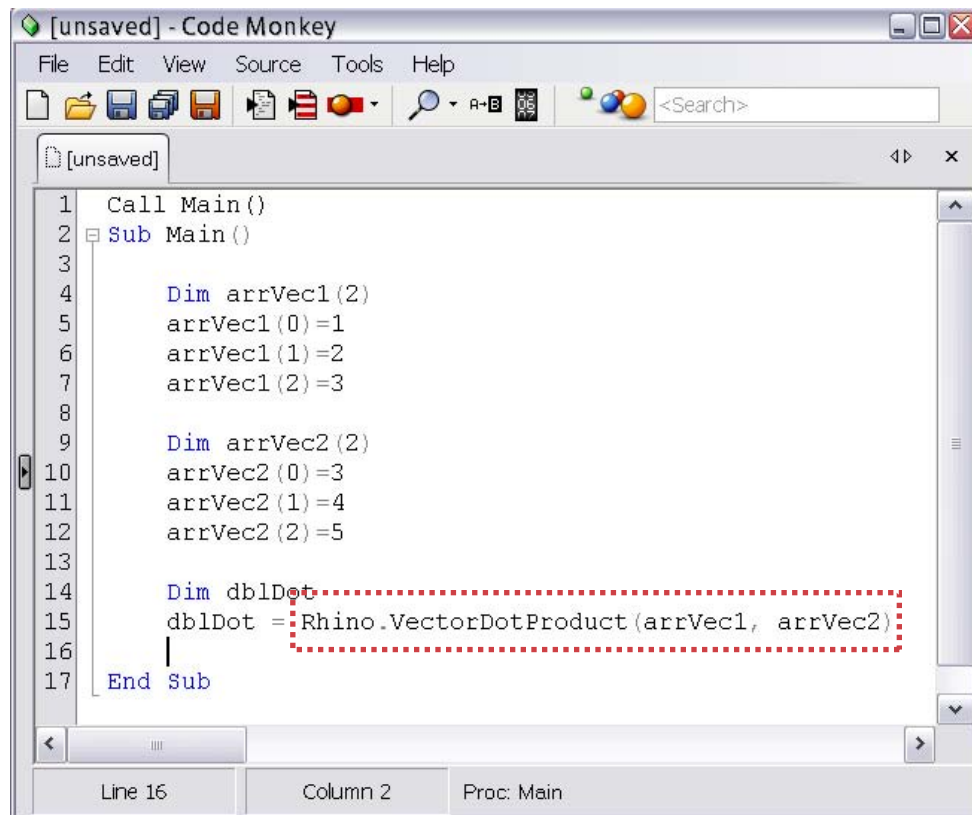
Cross Product as Perpendicular Line



Dot Product as Line Projection



Cross Product as Perpendicular Line



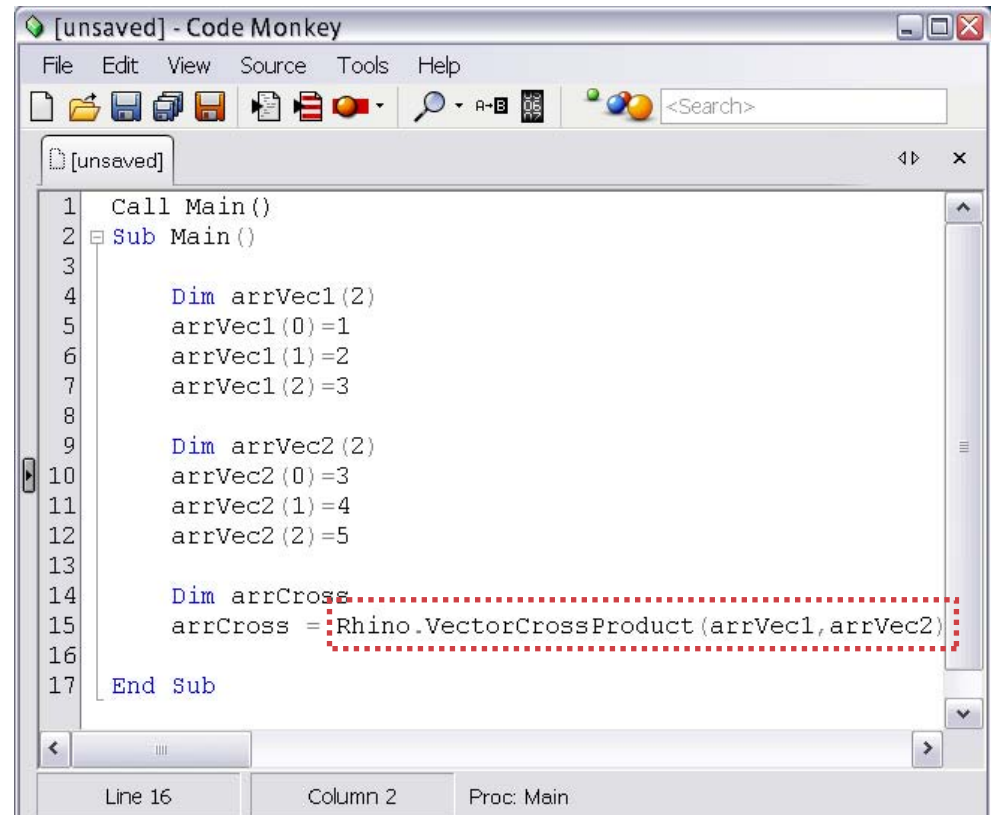
The screenshot shows the Code Monkey script editor with the following code:

```

1 Call Main()
2 Sub Main()
3
4   Dim arrVec1(2)
5   arrVec1(0)=1
6   arrVec1(1)=2
7   arrVec1(2)=3
8
9   Dim arrVec2(2)
10  arrVec2(0)=3
11  arrVec2(1)=4
12  arrVec2(2)=5
13
14  Dim dblDot
15  dblDot = Rhino.VectorDotProduct(arrVec1, arrVec2)
16
17 End Sub
  
```

The line `dblDot = Rhino.VectorDotProduct(arrVec1, arrVec2)` is highlighted with a red dashed box. The status bar at the bottom indicates "Line 16", "Column 2", and "Proc: Main".

Dot Product



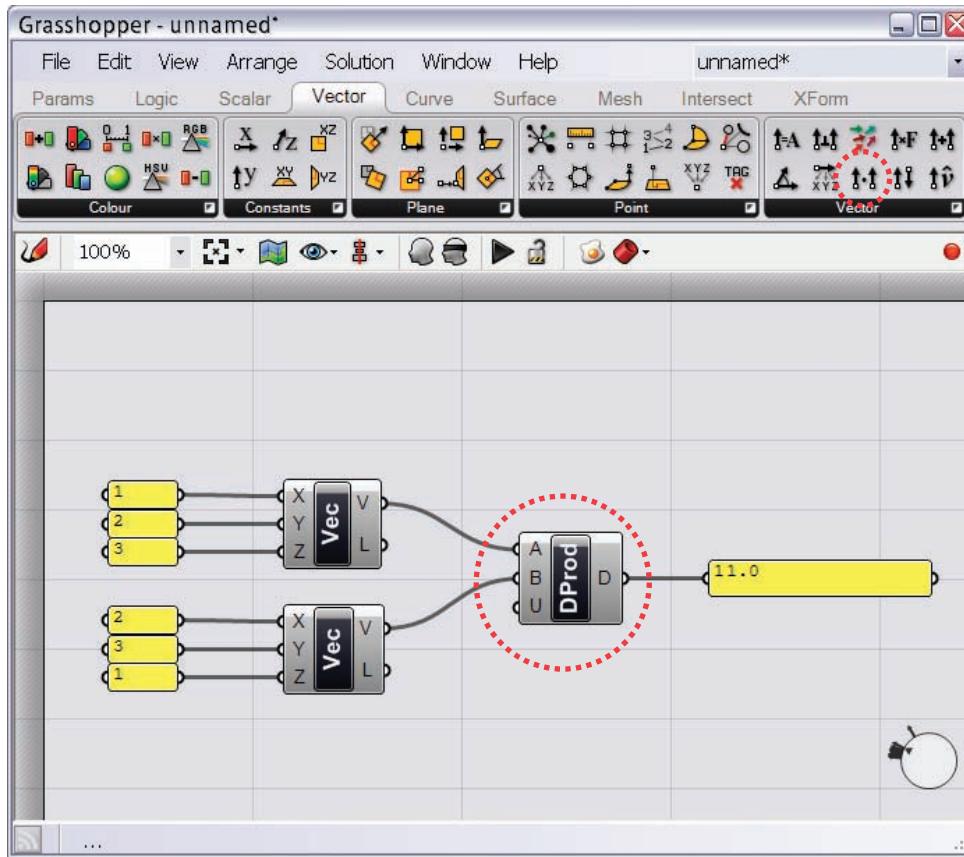
The screenshot shows the Code Monkey script editor with the following code:

```

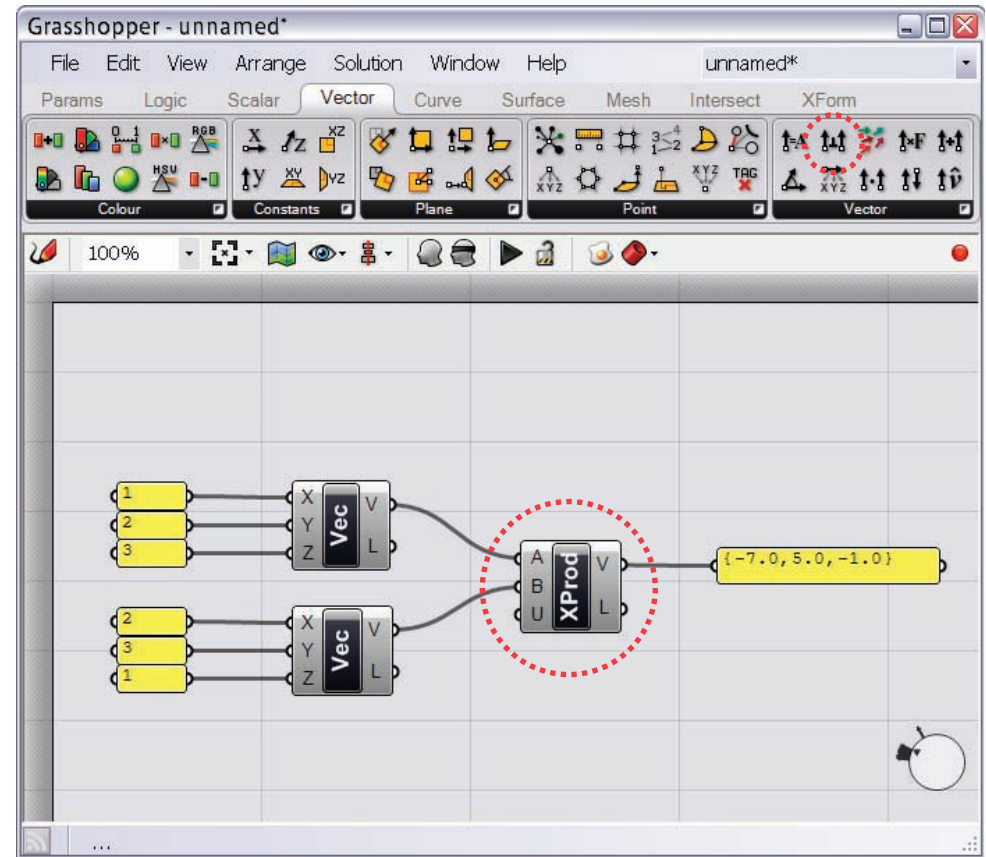
1 Call Main()
2 Sub Main()
3
4   Dim arrVec1(2)
5   arrVec1(0)=1
6   arrVec1(1)=2
7   arrVec1(2)=3
8
9   Dim arrVec2(2)
10  arrVec2(0)=3
11  arrVec2(1)=4
12  arrVec2(2)=5
13
14  Dim arrCross
15  arrCross = Rhino.VectorCrossProduct(arrVec1, arrVec2)
16
17 End Sub
  
```

The line `arrCross = Rhino.VectorCrossProduct(arrVec1, arrVec2)` is highlighted with a red dashed box. The status bar at the bottom indicates "Line 16", "Column 2", and "Proc: Main".

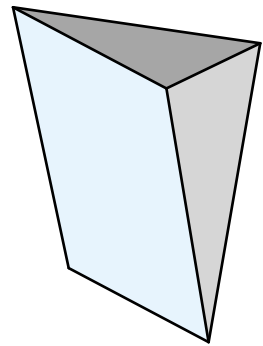
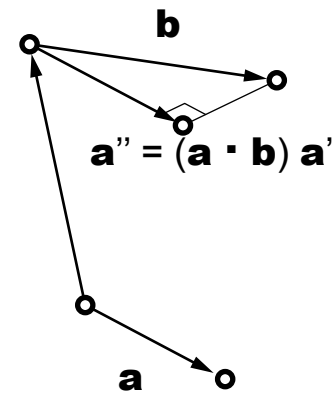
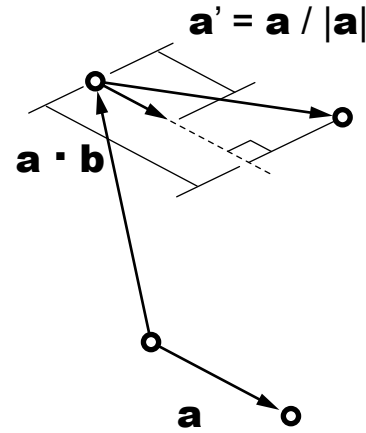
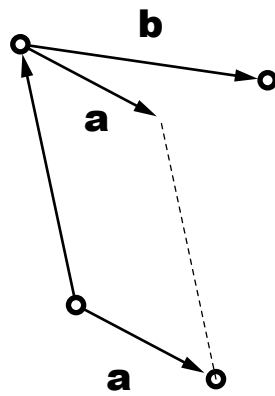
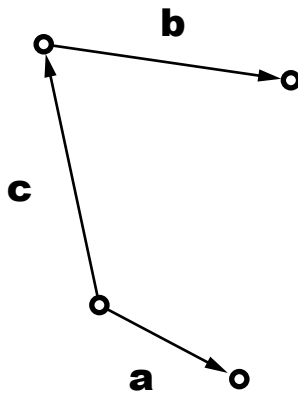
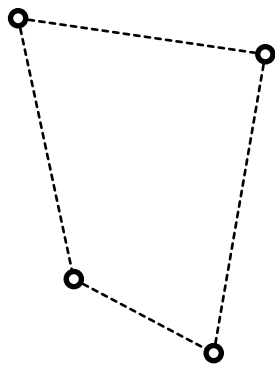
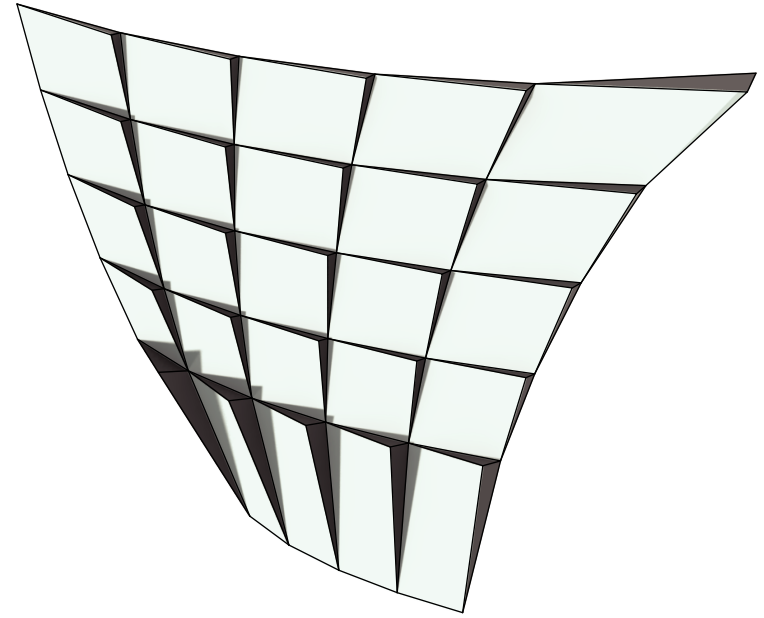
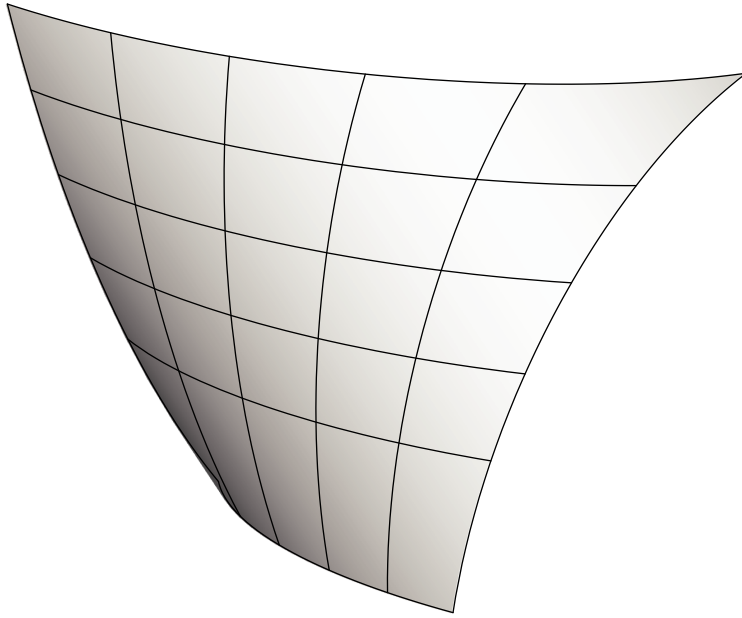
Cross Product

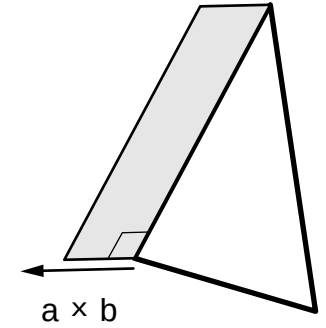
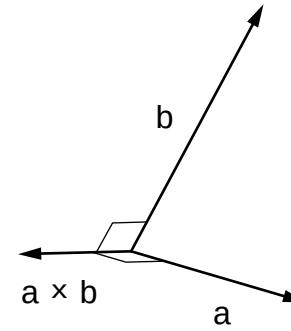
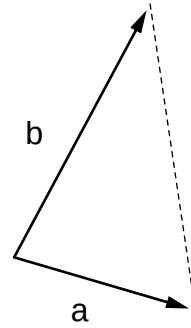
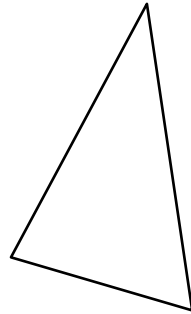
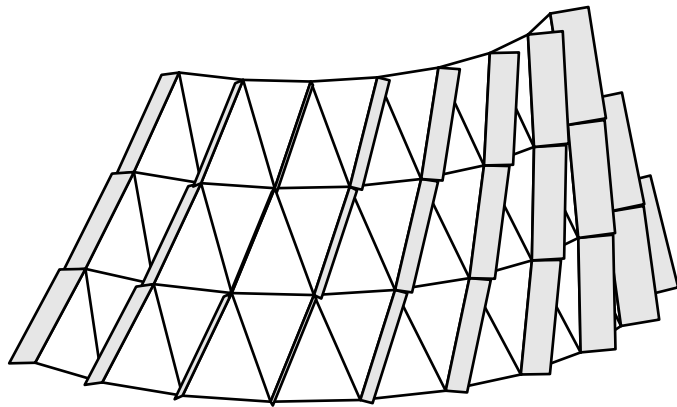


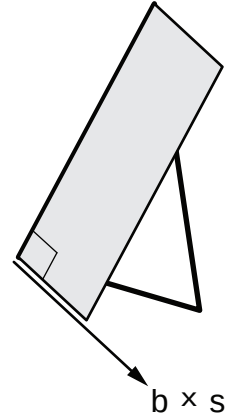
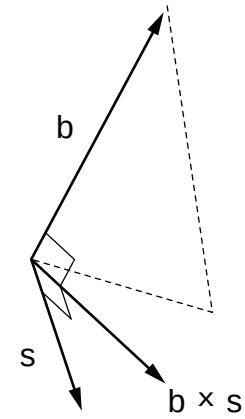
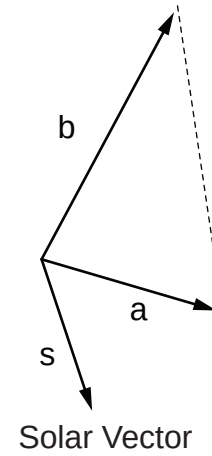
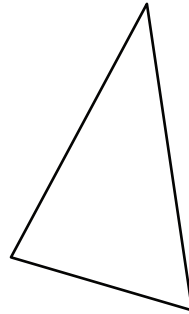
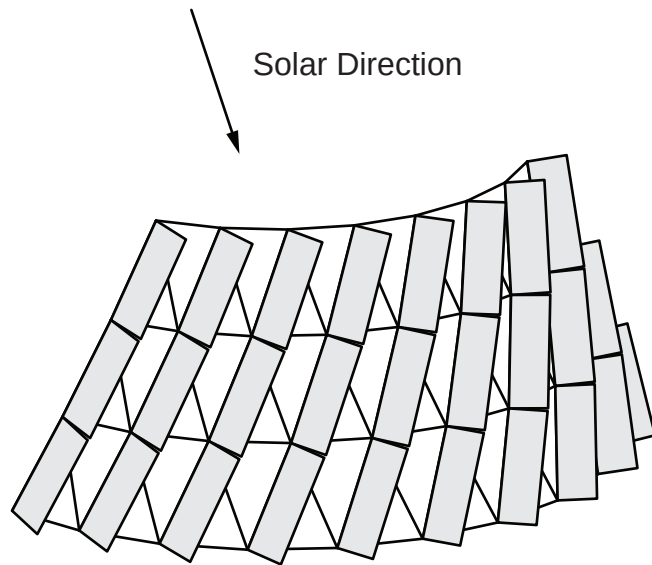
Dot Product

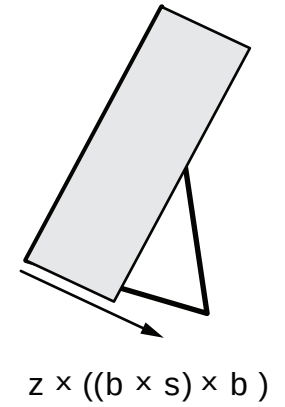
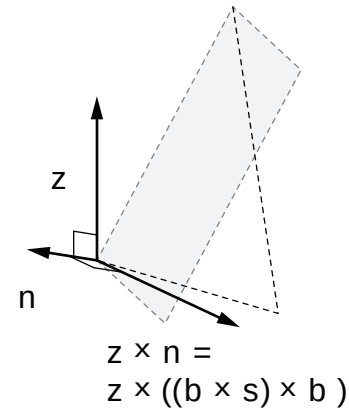
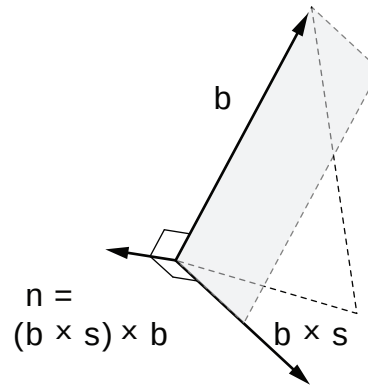
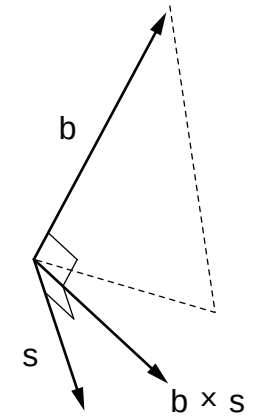
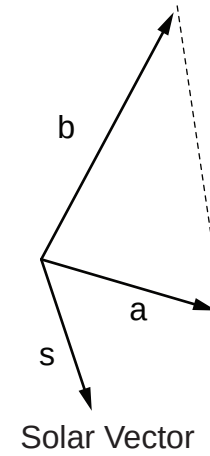
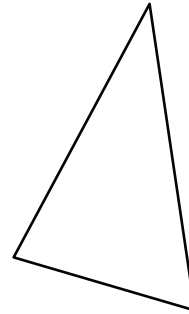
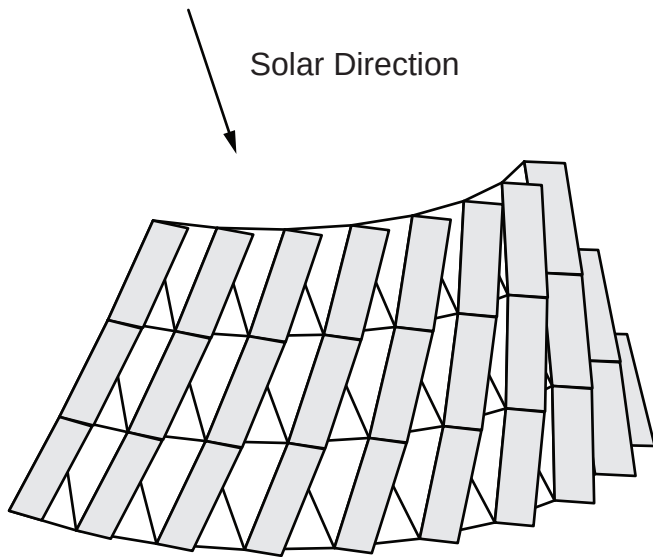


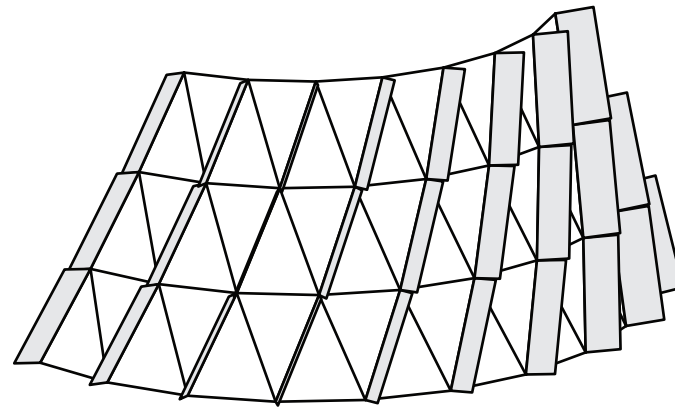
Cross Product







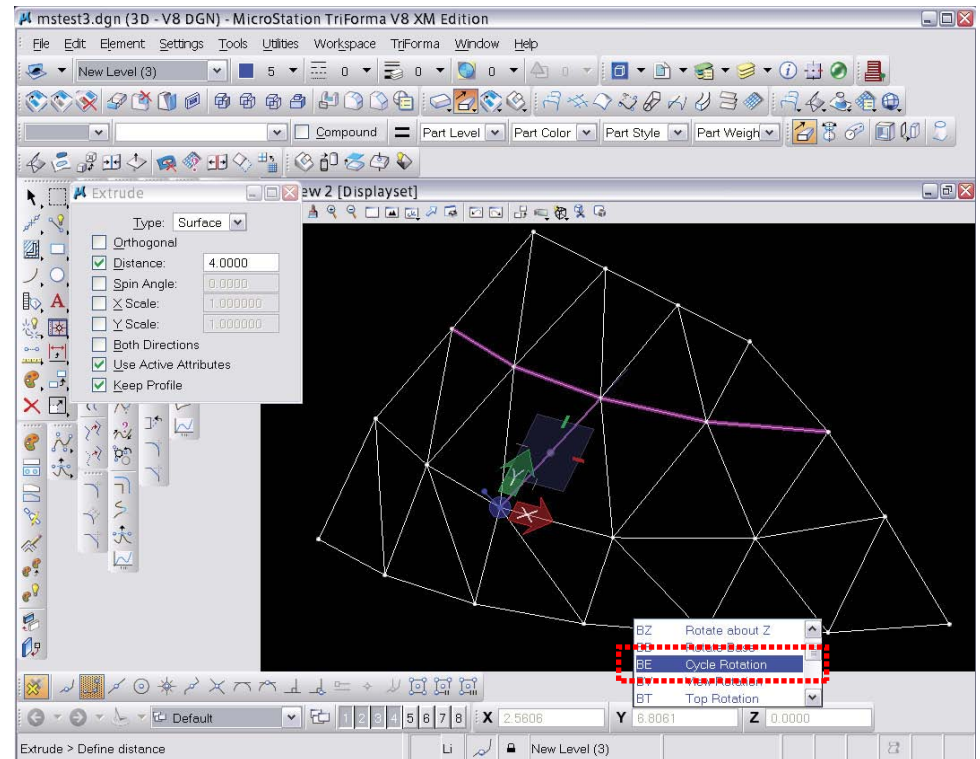
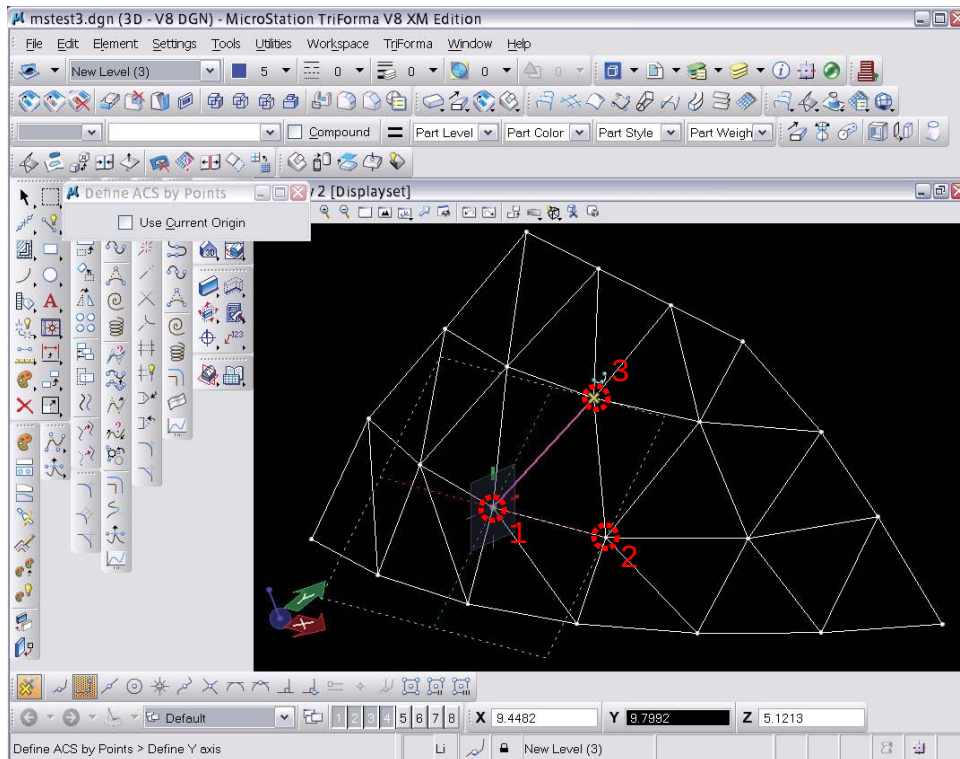
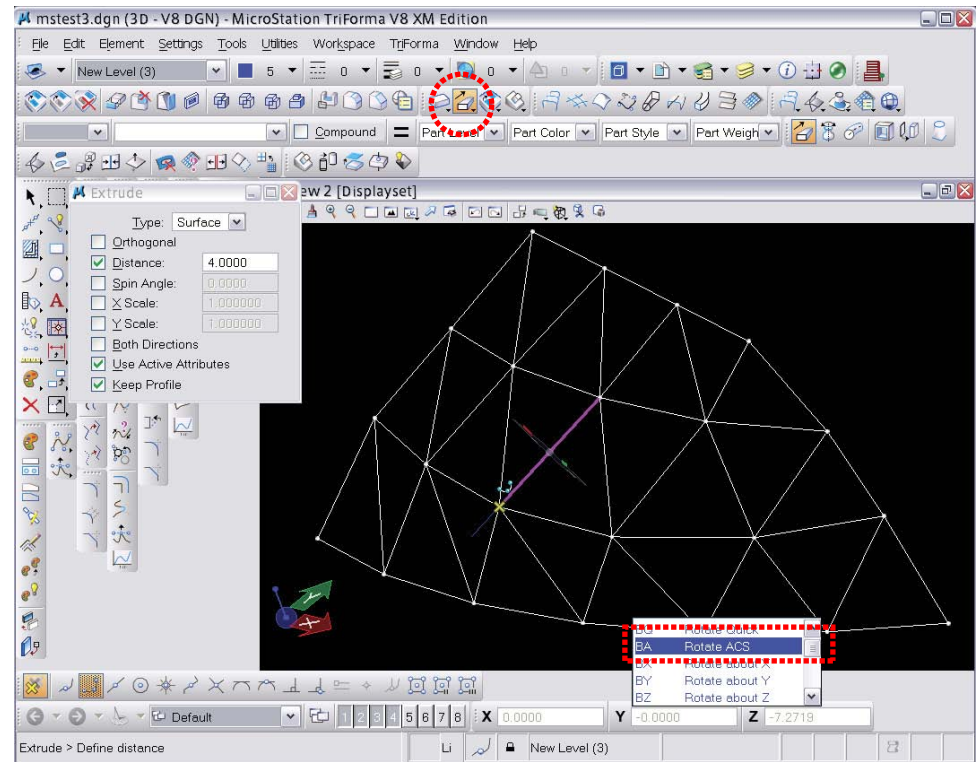
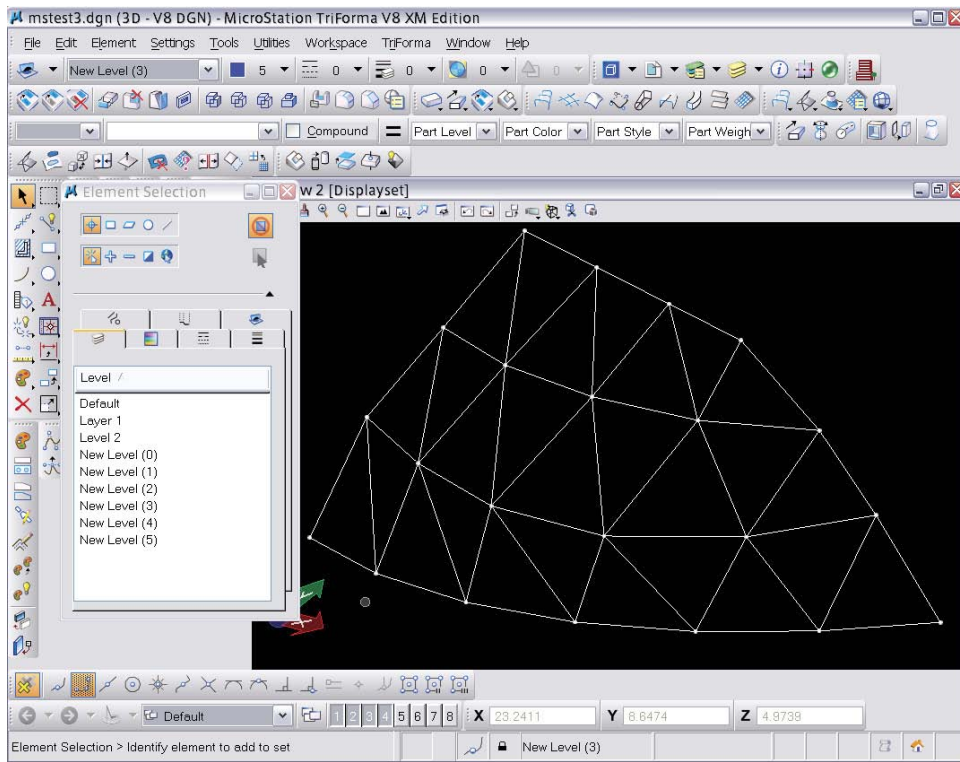




Make small diagrid and model fins perpendicular to the diagrid on any platform you choose.

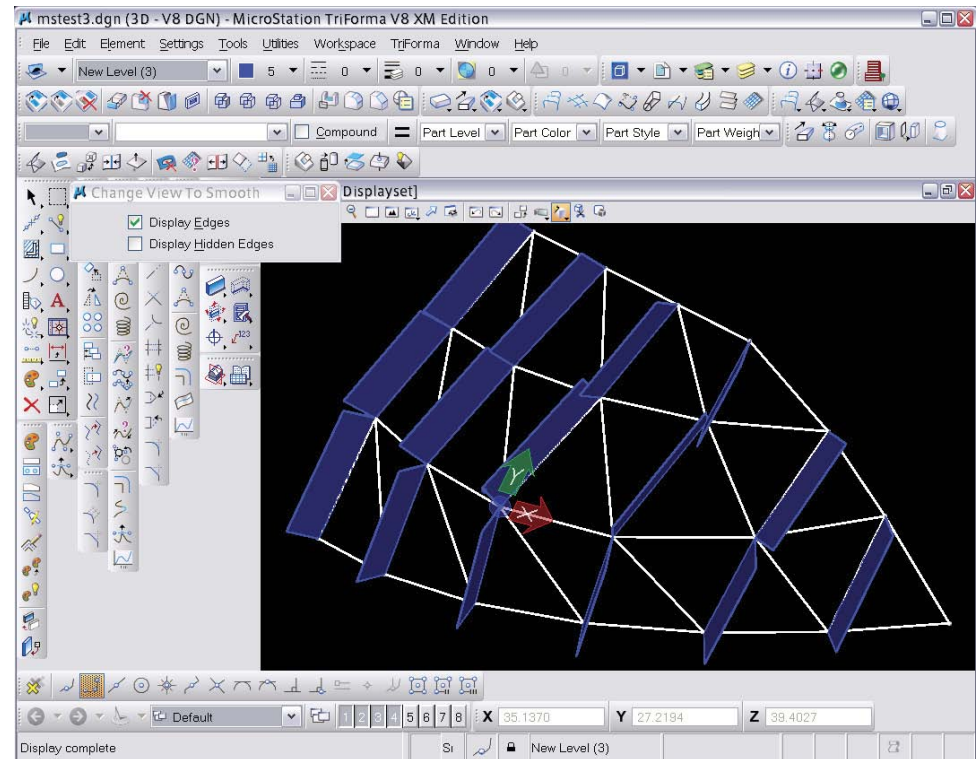
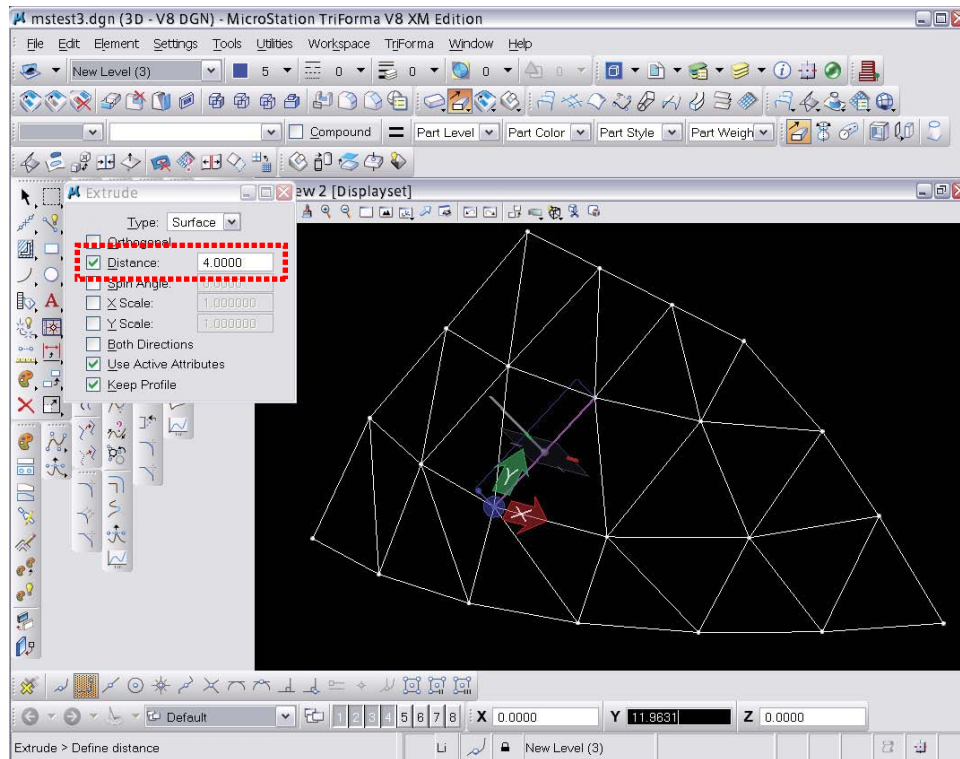
MATHEMATICS FOR SYSTEMATIC MODELING_1 POINT AND VECTOR

POINT AND VECTOR EXERCISE



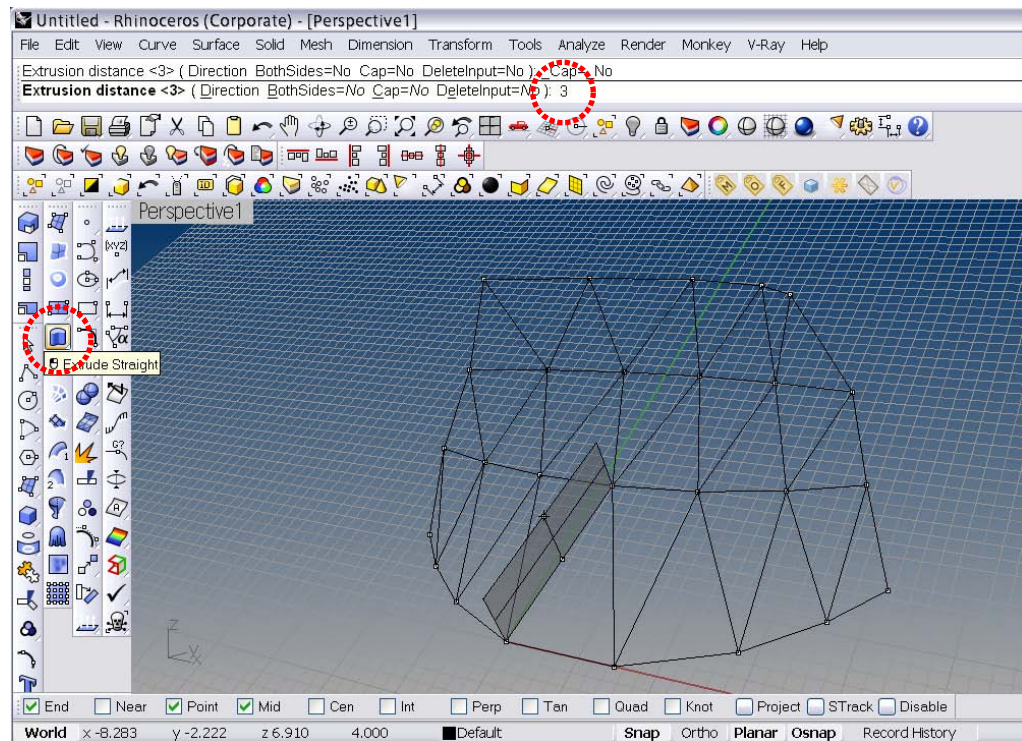
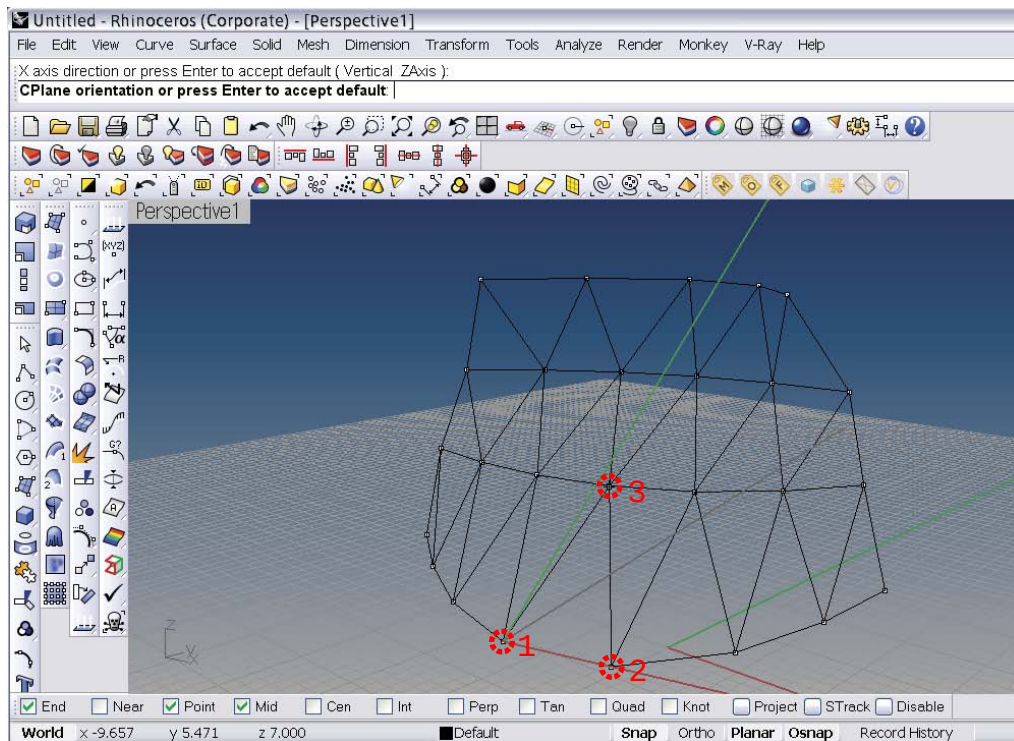
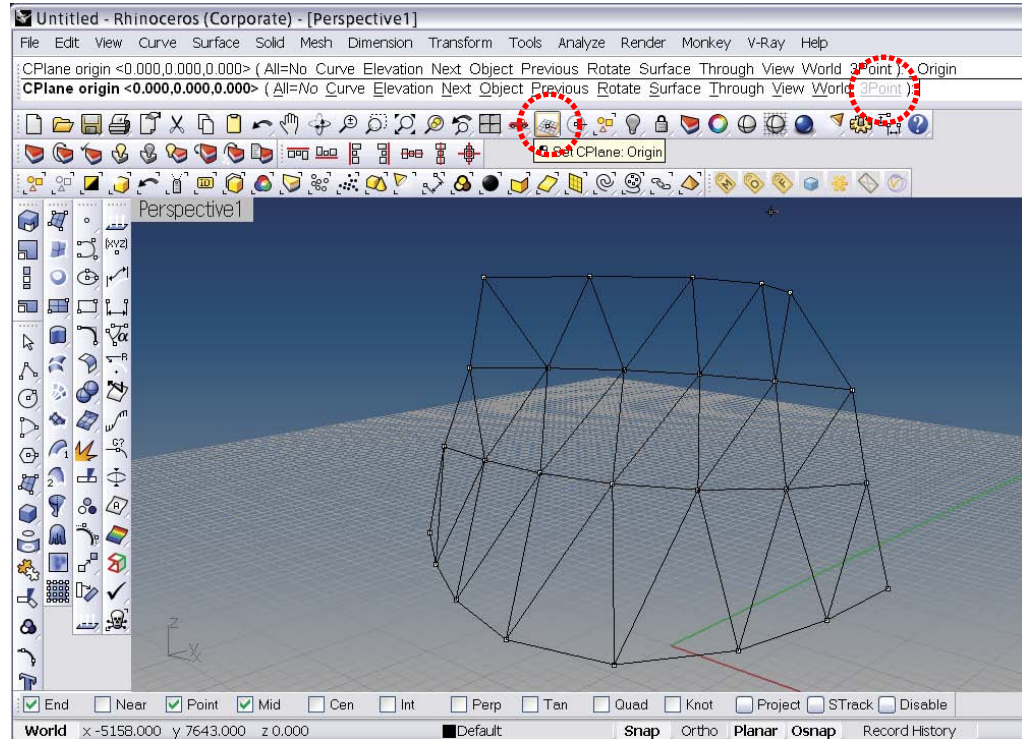
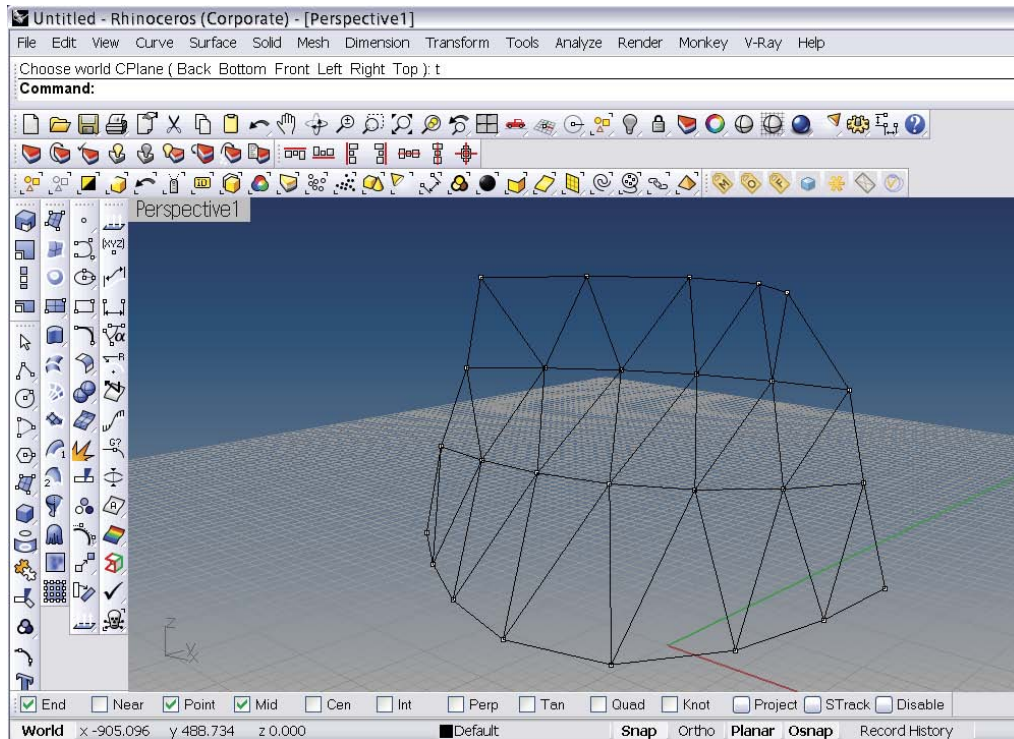
MATHEMATICS FOR SYSTEMATIC MODELING_1 POINT AND VECTOR

POINT AND VECTOR EXERCISE



MATHEMATICS FOR SYSTEMATIC MODELING_1 POINT AND VECTOR

POINT AND VECTOR EXERCISE



MATHEMATICS FOR SYSTEMATIC MODELING_1 POINT AND VECTOR

POINT AND VECTOR EXERCISE

