3D in Ti-Nspire and GeoGebra

Øystein Nordvik

e-mail: oystein.nordvik@hsh.no

Stord Haugesund University College

Abstract:

This article explains how it is possible to construct a three dimensional coordinate system in Ti-Nspire and GeoGebra. Both software packages offer dynamic geometry in two dimensions and the possibility to enter formulas that will be recalculated continuously. Teachers often need three dimensional visualizations to accompany explanations. Points in space, vectors, straight lines and planes are examples of typical syllabus content in higher mathematics. The focus of the article is both the explanation of the mathematical problems that lie behind such a construction and also to show how we can actually construct true perspective 3d.

Introduction:

The objective of this article is to explain how it is possible to construct a three dimensional coordinate system in a two dimensional environment. I will go through the mathematical thinking behind such a construction in some detail, so that it can be carried out in both Ti-Nspire and GeoGebra. The two software packages offer similar possibilities, but sometimes, to achieve the same result, it must be carried out in different ways. In order to construct a coordinate system I will demonstrate that what is needed are 9 points, three points of infinity and 6 points to build a cube. To be able to enter points defined by coordinates, three growth factors and a formula for each of the units **i**, **j** and **k** are needed.

In the pursuit of the points and the formulas I focus on the following 4 issues:

1. Rotation
2. Points of infinity
3. Relative size of objects when rotated
4. Construction technique

I will address each problem in this order and give examples of use at the end of the article.

1. Rotation: There is need for both horizontal and vertical rotation. However, it is not necessary, but convenient to construct a two dimensional slider that gives you both horizontal (**ho**) and vertical (**ve**) angles by moving one point. Such a slider is built by putting a point P inside a rectangle (**l** by **w**). Next measure horizontal (**h**) and vertical (**v**) distance to the left and the upper side, and calculate the formulas $-180 + \frac{h}{l} \cdot 360$ and $-180 + \frac{v}{w} \cdot 360$ using **h**, **v**, **l** and **w**. When the point inside is moved, the distances **h** and **v** will vary from 0 to **l** and 0 to **w** respectively. The result will be angles varying from -180 to 180. Hide everything in this construction except P which will be the movable point that will rotate the coordinate system (see Figure 1). Be aware that in GeoGebra, **ho** and **ve** are defined as numbers (not angles) when we construct this slider. As a consequence, when used with trigonometric functions, **ho** and **ve** should be transformed to $\frac{\pi}{180}\cdot\boldsymbol{ho}$ and $\frac{\pi}{180}\cdot\boldsymbol{ve}$.

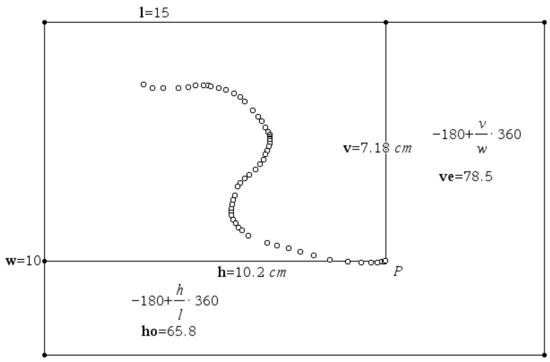


Figure 1

Start the construction of the coordinate system by defining independent variable **s** and construct a square with sides 2**s**. Rotate the square around its centre O using the horizontal angle **ho** (see Figure 3 and 4).

Vertical rotation is based on the idea that an object that is rotated towards or away from the eye, is in fact a projection as shown in Figure 2 constructed in Ti-Nspire.

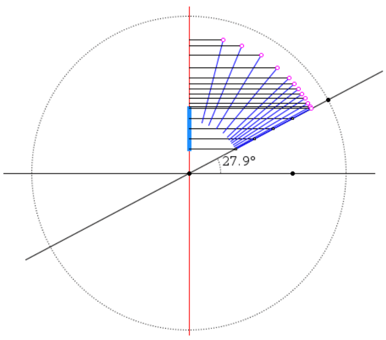


Figure 2

The red vertical line represents our screen and the light blue segment is what we see of the rotated object.

Through each vertex of the rotated square, construct a line perpendicular to the horizontal center line. In Ti-Nspire draw vertical vectors from the horizontal center line to each vertex. Measure the magnitude of the vectors and calculate new variables using a formula like "$distance\cdot\sin(ve)$". Use *Measurement Transfer* and transfer these values to their respective vectors, blue and red as seen in Figure 3. In GeoGebra, define the factor $\boldsymbol{fa}=\sin(\frac{\pi}{180}\cdot ve)$. Activate the command *Enlarge Object from Point by Factor*, click on a vertex, then the point on the horizontal center line and write **fa** in the dialog box. There is no need for measuring the distances. In both Ti-Nspire and GeoGebra this produces 4 points to construct the bold polygon seen in Figures 3 and 4.

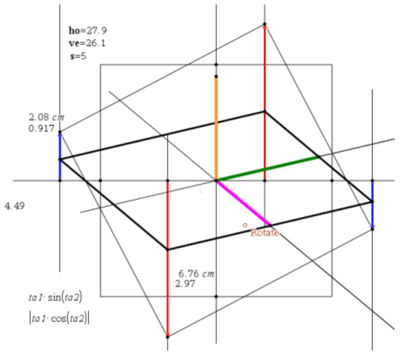


Figure 3

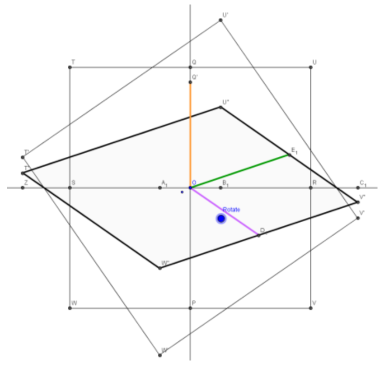


Figure 4

2. Points of infinity: In perspective drawings lines that are parallel, are not drawn as parallel lines, see Figure 5, they intersect in points of infinity.

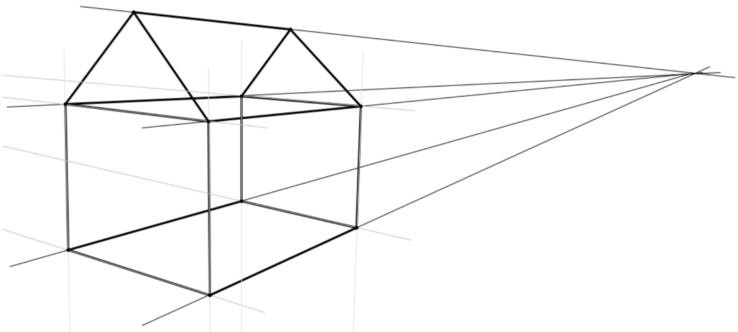


Figure 5

To calculate and construct the points of infinity, consider the length of the purple (**xp**), green (**yp**) and orange (**zp**=$|\boldsymbol{s} \cdot \cos(\boldsymbol{ve})|$) segment in Figure 3 and 4. **xp**, **yp** and **zp** will vary from 0 to **s** as we rotate. When either of these variables is close to **s**, the point of infinity should approach ∞, and when either is close to 0, so should the point of infinity be. As a consequence, we need a formula that will approach 0 when the length of a segment approaches zero and that approaches infinity when the length approaches **s**. The formula will be: $\frac{\boldsymbol{v1 \cdot xp}}{\boldsymbol{s-xp}}$. For a positive constant **v1** that we define, $\mathbf{lim}_{\boldsymbol{xp \to 0}} \frac{\boldsymbol{v1 \cdot xp}}{\boldsymbol{s-xp}} = \mathbf{0}$ and $\mathbf{lim}_{\boldsymbol{xp \to s}} \frac{\boldsymbol{v1 \cdot xp}}{\boldsymbol{s-xp}} = \infty$. (A small **v1** will give the impression of closeness to the construction, a bigger **v1** will remove ourselves from it.) The same argument can be used for y- and z-direction. Three positive variables **vx**, **vy** and **vz** will result from this. However, as we rotate through [-180,180] horizontally and vertically, a point of infinity should always be away from the eye. Boolean expressions will change the sign of **vx** as follows:

ix:=when((−180<ve<−90 or 90<ve<180) and −90<ho<90 or −90<ve<90 and (−180<ho<−90 or 90<ho<180),vx,−vx)

Similarly the sign of **vy** and **vz** are:

iy:=when(0<ho<180 and −90<ve<90 or −180<ho<0 and (−180<ve<−90 or 90<ve<180),vy,−vy)

iz:=when(−180<ve<−90 or 0<ve<90,−vz,vz)

The when-command needs to be replaced by an if-command in GeoGebra. In addition use || for *or* as well as *&&* for *and*. There are different techniques for Ti-Nspire and GeoGebra on how to transfer these values to the axes. In Ti-Nspire we place rays on the axes from O in positive directions and use the *Measurement Transfer* command to do so. In GeoGebra, the tool *Enlarge Object from Point by Factor* can be used. For the infinity point in the x-direction, activate the command, first click on the point O followed by the point xp and then enter **ix** in the dialog box. Do the same with y- and z-directions.

3. Relative size of objects when rotated: In perspective drawings the size of an object will depend on how close an object is to the eye. For example we will have that for the xy-plane, the distance from O to 5 on the x-axis, is equal to the distance from O to -5. These equal lengths must be different in a perspective drawing. Their relative size can be can be explained by Figure 6:

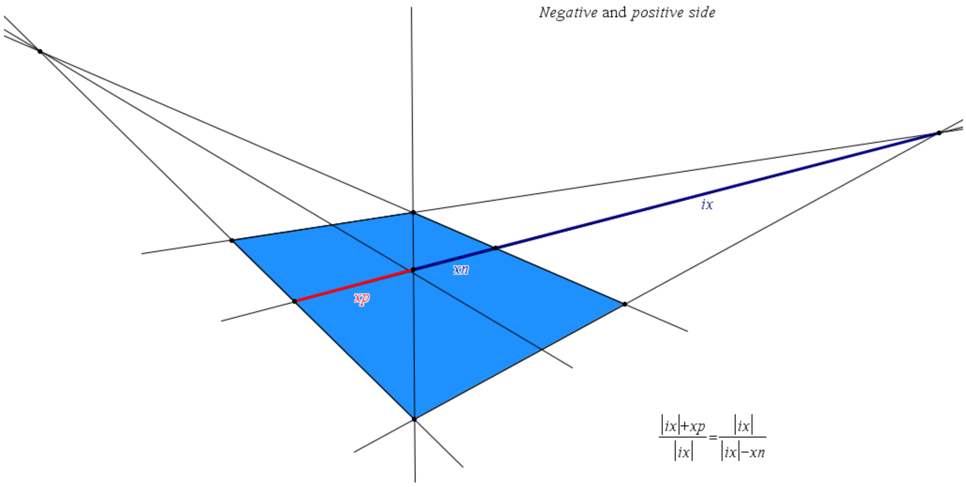


Figure 6

The solution of the equation $\frac{|ix|+xp}{|ix|}=\frac{|ix|}{|ix|-xn}$ between two ratios with respect to **xn,** is $\frac{|ix|\cdot xp}{|ix|+xp}$. Find **yn** and **zn** similarly. However, as with the points of infinity, as we rotate, sometimes **xp** is closest to the eye, sometimes **xn**. Boolean formulas take care of that:

xpos:=when((−180<ve<−90 or 90<ve<180) and −90<ho<90 or −90<ve<90 and (−180<ho<−90 or 90<ho<180),xn,xp)

xneg:=when((−180<ve<−90 or 90<ve<180) and −90<ho<90 or −90<ve<90 and (−180<ho<−90 or 90<ho<180),−xp,−xn)

ypos:=when(0< ho <180 and −90<ve<90 or −180< ho <0 and (−180<ve<−90 or 90<ve<180),yn,yp)

yneg:=when(0< ho <180 and −90<ve<90 or −180< ho <0 and (−180<ve<−90 or 90<ve<180),−yp,−yn)

zpos:=when(−180<ve<−90,−zn,when(−90<ve<0,zn,when(0<ve<90,zp, −zp)))

zneg:=when(−180<ve<−90,zp,when(−90<ve<0,−zp,when(0<ve<90,−zn, zn)))

In GeoGebra the formulas are similar except for the fact that **xp** should be replaced by **1** and **xn** should be replaced by $\frac{xn}{xp}$. When transferring these numbers, first click on the point **xp** followed by the point **O** and then enter **xpos** in the dialog box. Similar procedures should be used for the other 5 variables.

After transferring these 6 remaining variables to their respective axis, we can construct by straight lines and intersection points a cube that will represent our foundation for the coordinate system.

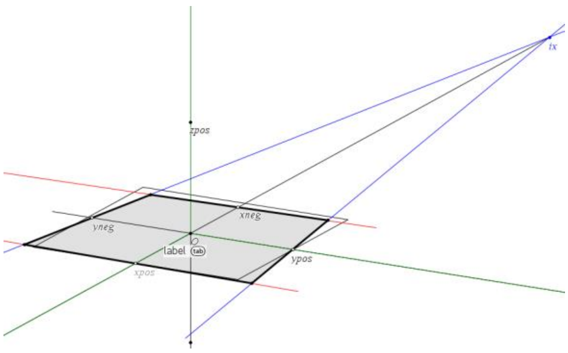


Figure 7

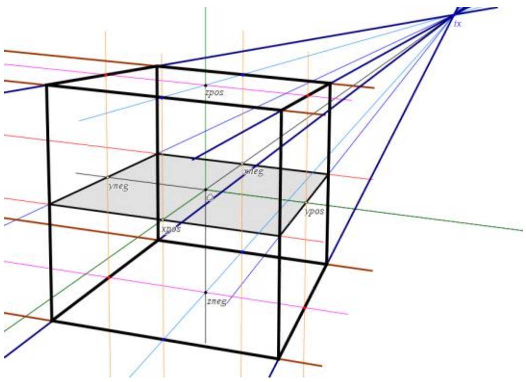


Figure 8

Figures 7 and 8 are screenshots from a construction in Ti-Nspire, but the following link will show how this construction works in GeoGebra:
http://ans.hsh.no/home/ono/Matematikk/Geogebra/Basisplan_nov_29_2012.html

4. Construction technique: As we have pointed out, there is a difference in length of **xpos** and **xneg** and for **ypos** and **yneg** as well as for **zpos** and **zneg**. The ratio $\frac{xpos}{|xneg|}$ is the basis for the factor for x. Since this ratio represents lengths of size **s,** we define the factor for x to be: $fx = \left(\frac{xpos}{|xneg|}\right)^{\frac{1}{s}}$. Similarly we define $fy = \left(\frac{ypos}{|yneg|}\right)^{\frac{1}{s}}$ and $fz = \left(\frac{|zpos|}{|zneg|}\right)^{\frac{1}{s}}$. The factors are growth factors, sometimes bigger than 1, sometimes smaller than 1, but always positive numbers.

Consider the values 1, 2, 3, 4, … on the x-axis (similar arguments can be made for y and z); when the positive side of the x-axis is towards the eye, the distance between these points will increase like a geometric sequence with a positive ratio bigger than 1.When the positive side is away from the eye there will be a similar change of distances, but the ratio will be smaller than 1. Every new value will increase by the factor **fx**. The distance from 1 to 2 will be the distance **i** from 0 to 1 multiplied by **fx**. Therefore we have:

$$\mathbf{2} = \boldsymbol{i} + \boldsymbol{i} \cdot (\boldsymbol{fx})$$

$$\mathbf{3} = \boldsymbol{i} + \boldsymbol{i} \cdot (\boldsymbol{fx}) + \boldsymbol{i} \cdot (\boldsymbol{fx})^{\mathbf{2}}$$

$$\boldsymbol{n} = \boldsymbol{i} + \boldsymbol{i} \cdot (\boldsymbol{fx}) + \boldsymbol{i} \cdot (\boldsymbol{fx})^{\mathbf{2}} + \cdots + \boldsymbol{i} \cdot (\boldsymbol{fx})^{\boldsymbol{n}-\mathbf{1}}$$

We observe that any value we want to put on one of the axes needs to be evaluated as a geometric series. A number $\boldsymbol{n}$ is represented in the coordinate system by the number

$$\boldsymbol{i} \cdot \frac{(\boldsymbol{fx})^{\boldsymbol{n}} - \mathbf{1}}{(\boldsymbol{fx}) - \mathbf{1}}$$

Finally we need to find the size of the units; **i**, **j** and **k**. The equation $\boldsymbol{i} \cdot \frac{(\boldsymbol{fx})^{s}-\mathbf{1}}{(\boldsymbol{fx})-\mathbf{1}} = \boldsymbol{xpos}$ solved with respect to **i** and similar equations for **j** and **k** give us the units. The remaining formulas are: $\boldsymbol{i} = \frac{((\boldsymbol{fx})-\mathbf{1})\cdot \boldsymbol{xpos}}{(\boldsymbol{fx})^{s}-\mathbf{1}}$, $\boldsymbol{j} = \frac{((\boldsymbol{fy})-\mathbf{1})\cdot \boldsymbol{ypos}}{(\boldsymbol{fy})^{s}-\mathbf{1}}$ and $\boldsymbol{k} = \frac{((\boldsymbol{fz})-\mathbf{1})\cdot \boldsymbol{zpos}}{(\boldsymbol{fz})^{s}-\mathbf{1}}$.

Once the coordinate system has been made, it may be saved and used as basis for any construction you would like to develop it into, as the examples below show us. It is also possible to include a zoom option into the construction by replacing **s** with $\frac{s}{zoom}$ in every formula. A variable defined as **zoom** and adjusted by a slider will thus make it possible to zoom in or out. Another feature seen in my files is the grid in the xy-plane. Place values ranging from -15 to 15 on both x- and y-axis. Draw lines to **ix** and **iy**, find intersection with sides of the polygon that is the xy-plane. These intersection points will only exist for elements of [-15,15] that are smaller than or equal to $\frac{s}{zoom}$.

Conclusion: There is a variety of problems that can be visualized by such a construction. In my homepage (www.oysteinnordvik.no) more examples can be found and all the *.tns files can be downloaded from this site. Examples 2 to 7 below are just some of the concepts I have found it useful to visualize. In http://www.geogebratube.org/user/profile/id/7876 I have uploaded further examples.

Some of the GeoGebra files in the following examples are constructed in GeoGebra 4, some in Geogebra 4.2. It should be pointed out that files made in GeoGebra 4 will work well in GeoGebra 4.2 but not necessarily the other way.

Example 1:

Construction of the point (2,3,4), see Figure 9. The coordinates of the point must first be calculated the following way: 2 will be $\boldsymbol{i} \cdot \frac{(\boldsymbol{fx})^{(2)}-1}{(\boldsymbol{fx})-1}$, 3 will be $\boldsymbol{j} \cdot \frac{(\boldsymbol{fy})^{(3)}-1}{(\boldsymbol{fy})-1}$ and

4 will be $\boldsymbol{k} \cdot \frac{(\boldsymbol{fz})^{(4)}-1}{(\boldsymbol{fz})-1}$. In GeoGebra the formulas will be: $\frac{1}{\boldsymbol{xpos}} \cdot \boldsymbol{i} \cdot \frac{(\boldsymbol{fx})^{(2)}-1}{(\boldsymbol{fx})-1}$, $\frac{1}{\boldsymbol{ypos}} \cdot \boldsymbol{j} \cdot \frac{(\boldsymbol{fy})^{(3)}-1}{(\boldsymbol{fy})-1}$ and $\frac{1}{\boldsymbol{zpos}} \cdot \boldsymbol{k} \cdot \frac{(\boldsymbol{fz})^{(4)}-1}{(\boldsymbol{fz})-1}$. When we have done these calculations, we transfer these values to the appropriate axis where they will be shown as 2 on the x-axes, 3 on the y-axes and 4 on the z-axes. Draw lines from the x-value and the y-value to **iz** (orange lines), lines from the y-value and the z-value to **ix** (blue lines) and lines from the x-value and the z-value to **iy** (red lines). You will have to rotate the system to do so. Next, find intersection point between orange line through y-value and red line through z-value and then intersection point between orange line through x-value and blue line through z-value. Draw a line through the first of these intersection points and **ix** and a line through the second intersection point and **iy**. Intersection point (2,3,4) between the last two lines has been found.

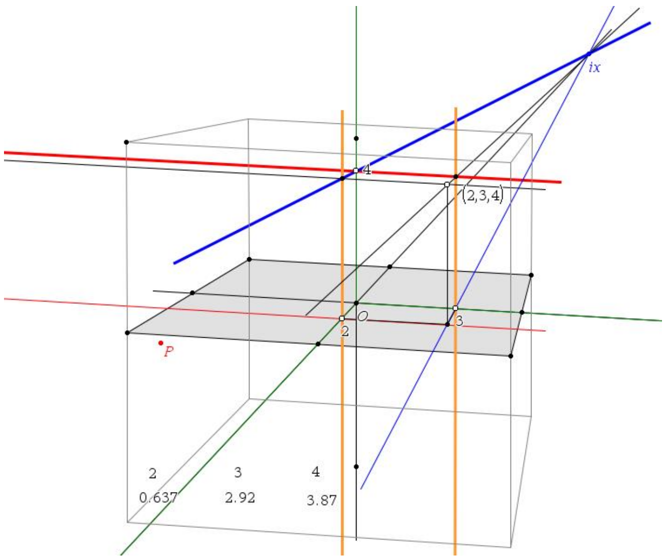


Figure 9

The entire construction can be seen in details in this video:
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Construction%20of%20Point/2_3_4_construction.wmv

Example 2:

Parallelepiped constructed in Ti_Nspire

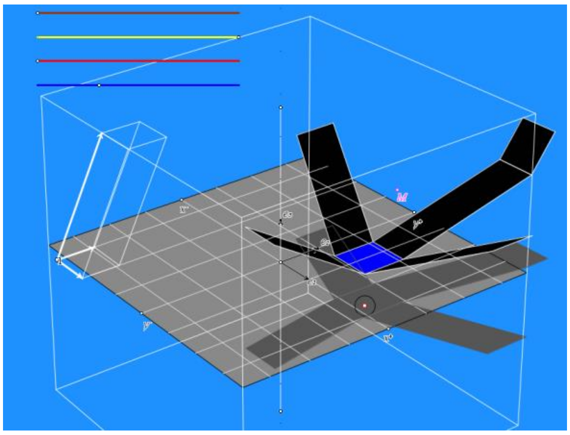


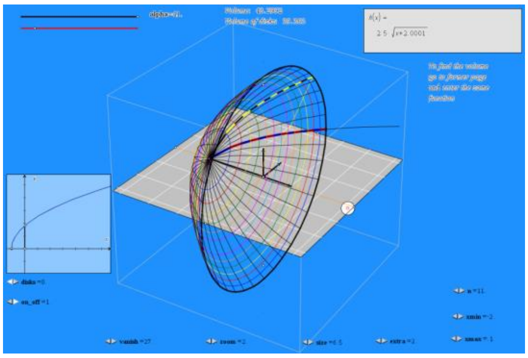
Figure 10

See video:
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Parallelepiped/Parallelepiped_feb_22_2012.wmv

Get the file:
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Parallelepiped/

Example 3:

Solid of Revolution constructed in Ti-Nspire Cas and Geogebra 4.2



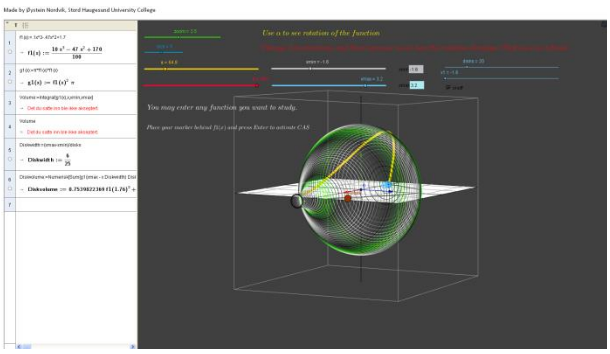


Figure 11 Figure 12

See video of file in Ti-Nspire:
http://ans.hsh.no/home/ono/Matematikk/Calculus/3d%20perspective/Solid%20of%20revolution/Solid_of_revolution_nov_2012.wmv

Get the Ti-Nspire file:
http://ans.hsh.no/home/ono/Matematikk/Calculus/3d%20perspective/Solid%20of%20revolution/

The Geogebra file: http://www.geogebratube.org/material/show/id/24686

Example 4:

Two straight lines constructed in Geogebra

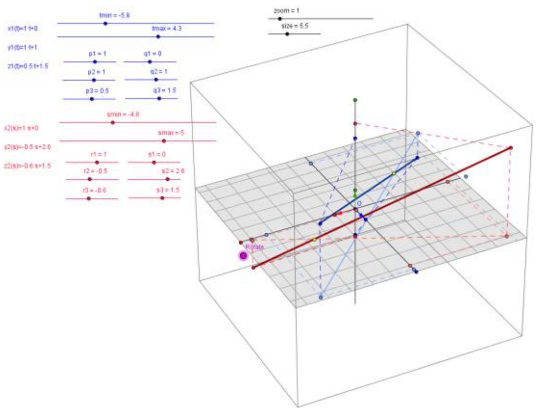


Figure 13

http://www.geogebratube.org/student/m20847

Example 5

The volume-of-the-box problem. From a piece of paper, a square is cut off in each corner to build a box. How big should the squares be in order to get maximum volume?

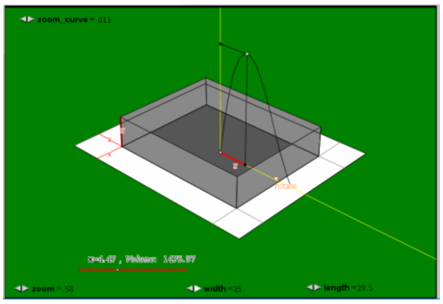


Figure 14

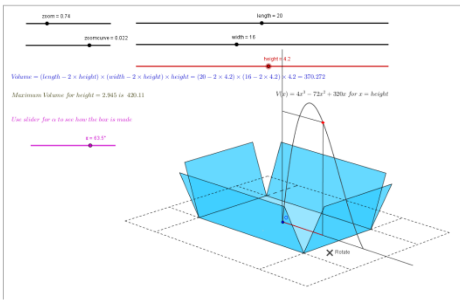


Figure 15

Video of the Ti-Nspire file:
http://ans.hsh.no/home/ono/Matematikk/Calculus/Volume_of_the_box/Volume_of_the_box_3.wmv

The Ti-Nspire file: http://ans.hsh.no/home/ono/Matematikk/Calculus/Volume_of_the_box/

Construction in GeoGebra: http://www.geogebratube.org/material/show/id/23580

Example 6

This file constructed in Ti-Nspire is meant to visualize the difference in observed and expected values in a chi-square test for independence. Data are entered in a spreadsheet, 2-6 rows by 2-6 columns. The Chi-square is calculated and showed in the graph together with the critical value.

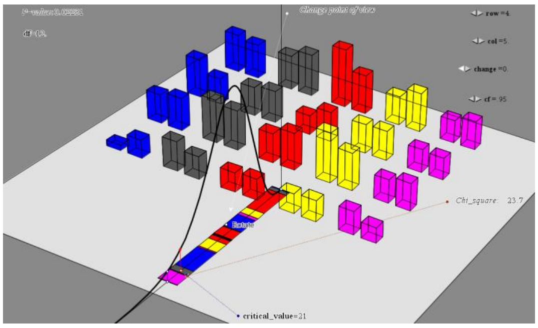


Figure 16

Video of the file:
http://ans.hsh.no/home/ono/Matematikk/Statistics/Hypotheses%20testing/Chi%20square%20for%20independence/Chi_square_independence.wmv

The file:
http://ans.hsh.no/home/ono/Matematikk/Statistics/Hypotheses%20testing/Chi%20square%20for%20independence/

Example 7

3D function constructed in GeoGebra 4.2. This particular file will not work in GeoGebra 4.0.

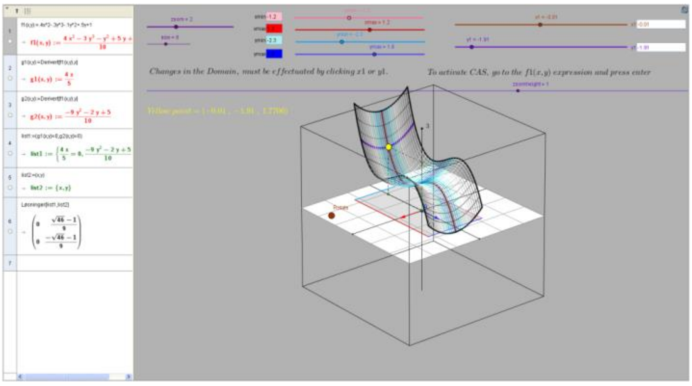


Figure 17

http://www.geogebratube.org/material/show/id/25287

References:

Software Packages:

[TI-Nspire] TI-Nspire CAS Teacher Version, version 3.2.0.1219, a product from Texas Instruments

[Geogebra] GeoGebra-Dynamic Mathematics for everyone, version 4.2.15.0

Manuals:

Ti-Nspire/Ti-Nspire CAS Teacher Software Guidebook (English):
http://education.ti.com/calculators/downloads/US/Software/Download/en/6775/7605/TI-NSpire_TS_Guidebook_EN.pdf

GeoGebra 4.0 Manual: http://wiki.geogebra.org/en/Manual:Main_Page, http://wiki.geogebra.org/en/Release_Notes_GeoGebra_4.2

Supplemental Electronic material

[1] Ø.Nordvik, Chi-Square test for independence, 2012, Ti-Nspire CAS
http://ans.hsh.no/home/ono/Matematikk/Statistics/Hypotheses%20testing/Chi%20square%20for%20independence/Chi_square_independence.wmv

http://ans.hsh.no/home/ono/Matematikk/Statistics/Hypotheses%20testing/Chi%20square%20for%20independence/

[2] Ø.Nordvik, Volume of the box problem, 2011, Ti-Nspire CAS
http://ans.hsh.no/home/ono/Matematikk/Calculus/Volume_of_the_box/Volume_of_the_box_3.wmv

http://ans.hsh.no/home/ono/Matematikk/Calculus/Volume_of_the_box/

[3] Ø.Nordvik, Space Curve, 2012, Ti-Nspire CAS and GeoGebra
http://ans.hsh.no/home/ono/Matematikk/Calculus/3d%20perspective/Space%20curve/Space_curve_jan_2012.wmv

http://ans.hsh.no/home/ono/Matematikk/Calculus/3d%20perspective/Space%20curve/

http://www.geogebratube.org/material/show/id/24116

[4] Ø.Nordvik, Unfolding Cone, 2012, Ti-Nspire CAS
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Unfolding_cone/Unfolding%20cone.wmv

http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Unfolding_cone/

[5] Ø.Nordvik, Dodecahedron, 2012, Ti-Nspire CAS
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Dodecahedron/Dodecahedron.wmv

http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Dodecahedron/

[6] Ø.Nordvik, 3D function and constrained optimization, 2012, Ti-Nspire CAS
http://ans.hsh.no/home/ono/Matematikk/Calculus/3d%20perspective/3d%20Function%20oct%202012%20path/3dFunction_sep_26.wmv

http://ans.hsh.no/home/ono/Matematikk/Calculus/3d%20perspective/3d%20Function%20oct%202012%20path/

[7] Ø.Nordvik, Pyramide, 2012, Ti-Nspire CAS
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Pyramide/Pyramide2.wmv

http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Pyramide/

[8] Ø.Nordvik, Rhombicosidodecahedron, 2012,
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Rhombicosidodecahedron/Rhombicosidodecahedron.wmv

http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Rhombicosidodecahedron/

[9] Ø.Nordvik, Unfolding Tetrahedron, 2012, Ti-Nspire CAS
http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Tetrahedron/Tetrahedron.wmv

http://ans.hsh.no/home/ono/Matematikk/Geometry/3D%20perspective/Tetrahedron/

[10] Ø.Nordvik, Points, Planes, Line and Pyramide, 2012,GeoGebra,
http://www.geogebratube.org/student/m22020

[11] Ø.Nordvik, Plane with normal vector, 2012,GeoGebra,
http://www.geogebratube.org/student/m21240

[12] Ø.Nordvik, 3D function, 2012, GeoGebra

http://www.geogebratube.org/material/show/id/25287