Pointalign mit Blender

PointAlign-Script 0.7 Documentation

Angleichung der ausgewählten Punkte an einen Punkt im 3D-Raum in Python
Foto

Preface

I need many times alignment of different vertexes to a coordinate.

Installation

1. Download the script pointalign.py
2. Copy the script in the **scripts**-directory of your Blender

Execute the script

Run the script from the 3d View's Mesh->Scripts menu
screen1

Align to the 3D-Cursor or to X,Y,Z-values

To align selected vertexes of a mesh to 3D-Cursor. Also it's possible to align to absolute X,Y,Z-values. Take actual 3D-Cursor coordinates by button.
Functions

1. align to the X-axis
2. align to the Y-axis
3. align to the Z-axis
4. X-value
5. Y-value
6. Z-value
7. refresh 3D-Cursor coordinates

Example

1. Add a new cube (**shift + a**)
   screen7
2. Set 3D-Cursor to a position (**left mouse button**)
   screen8
3. Deselect all vertexes (**a**)
4. Select the four right vertexes of the cube (**shift + right mouse button**)
   screen9
5. Align this vertexes to the X-axis of the 3D-Cursor's X-position (**script X-button** = 1)
   Functions
6. Finished
   screena

Align to local/global object-bounds

To align selected vertexes to the maximum or minimum boundaries of the object in all three directions.
Functions

1. options for local boundry
2. options for global boundry
3. X-axis local
4. Y-axis local
5. Z-axis local
6. X-axis global
7. Y-axis global
8. Z-axis global

Example

1. Add a new cube (**shift + a**)
   screen7
2. Deselect all vertexes (**a**)
3. Select the four upper vertexes of the cube (**shift + right mouse button**)
   screenb
4. Merge the selectes vertexes at the center (**alt-m + 1**)
   screenc
5. Align this vertex to the object's local X-axis minimum (**press X-button**)
   screend

Align to 3D-viewport

To align selected meshes to the point of view. Switching to editmode is not nesessary.
Functions

Example

1. Add a new cube (**shift + a**)
   screen7
2. Switch the editmode off (**Tab**)
3. Rotate to other point of view (**Numpad-2 + Numpad-4**)
   screene
4. Align the mesh to the view (**press "Align"-button**)
   Functions
5. Finished
   screenf

Align to Object

To align selected meshes to the point of view. Switching to editmode is not nesessary.
Functions

1. Direction to alignment (X,Y,Z-axis)
2. Source properties (minimum,maximum,middle,pivot)
3. Target properties (minimum,maximum,middle,pivot)
4. Execute alignment

Example

1. Add a new cube (**shift + a**)
   screen7
2. Switch the editmode off (**Tab**)
3. Add a new sphere (**shift + a)
4. Switch the editmode off (**Tab**)
   screen9
5. Select first the target(which will modified)-object, the cube and to the last the source object, the shere (**shift + right mouse button**)
   screenh
6. Set options to align the minimum X-boundary of the cube to the maximum X-boundary of the sphere
   screeni
7. Align the meshes (press **"Align"-button**)
   screenj

Planed Functions

- Alignment and Set Camera to Viewport.

Links

- Blender
- My Repeat-script

Script

```
#!BPY

""" Registrationsinformationen fuer Blender-Menues
Name: 'PointAlign'
Blender: 235
Group: 'Mesh'
Tip: 'Align selected vertices'
"""
__author__ = "Thomas Buschhardt"
__url__ = ["Script Site,www.buschhardt.de/pointalign"]
__version__ = "0.7 20050117"
__email__ = ["Thomas Buschhardt, thomas:buschhardt*de"]
__bpydoc__ = """\
This script implements vertex alignment in Blender.

Usage:

Select the mesh you want to work on, enter Edit Mode and select the vertices
to alignment. Set the 3D-Cursor on a target position.
Then run this script from the 3d View's Mesh->Scripts menu.

You can control the alignment on the global X,Y and Z axis.
To quit the script press 'ESC' or 'Q'.

Notes: You can undo and redo your steps just like with normal mesh operations in
```

```
Blender.
"""
##
# Hauptmodule einladen
##
import Blender

##
# globale Variablen
##
EREIGNIS_KEIN = 1
EREIGNIS_ZEICHNEN = 2
EREIGNIS_BEENDEN = 3
EREIGNIS_EINGABE = 4
EREIGNIS_CURSOR = 5
EREIGNIS_LOKAL = 6
EREIGNIS_GLOBAL = 7
EREIGNIS_VIEWPORT = 8
EREIGNIS_OBJEKT = 9
AENDERE_X = Blender.Draw.Create(0)
AENDERE_Y = Blender.Draw.Create(0)
AENDERE_Z = Blender.Draw.Create(0)
ORIGINALWERTE=[]
ORIGINALOBJEKT=0
ORIGINALINDEX=[]
XX = Blender.Draw.Create("")
YY = Blender.Draw.Create("")
ZZ = Blender.Draw.Create("")
XX.val = str(Blender.Window.GetCursorPos()[0])
YY.val = str(Blender.Window.GetCursorPos()[1])
ZZ.val = str(Blender.Window.GetCursorPos()[2])
LOK_X=Blender.Draw.Create(0)
LOK_Y=Blender.Draw.Create(0)
LOK_Z=Blender.Draw.Create(0)
GLO_X=Blender.Draw.Create(0)
GLO_Y=Blender.Draw.Create(0)
GLO_Z=Blender.Draw.Create(0)
DIR_X = Blender.Draw.Create(0)
DIR_Y = Blender.Draw.Create(0)
DIR_Z = Blender.Draw.Create(0)
ZIEL = Blender.Draw.Create(0)
QUELLE = Blender.Draw.Create(0)
RICHTUNG = Blender.Draw.Create(0)

##
# Hilfsfunktionen
##

# Umwandlungsfunktion Vektor
def mulmatvec4x3(a, b):
  # a is vector, b is matrix
  r = [0, 0, 0]
  r[0] = a[0]*b[0][0]+a[1]*b[1][0]+a[2]*b[2][0]+b[3][0]
  r[1] = a[0]*b[0][1]+a[1]*b[1][1]+a[2]*b[2][1]+b[3][1]
```

```
  r[2] = a[0]*b[0][2]+a[1]*b[1][2]+a[2]*b[2][2]+b[3][2]
  return r

# Originalwerte des Objektes speichern
def original():
  global ORIGINALWERTE
  global ORIGINALOBJEKT
  global ORIGINALINDEX
  ORIGINALWERTE=[]
  ORIGINALINDEX=[]
  Ausgangszustand = Blender.Window.EditMode()
  if Ausgangszustand==1:Blender.Window.EditMode(0)
  Objekt = Blender.Object.GetSelected()[0]
  ORIGINALOBJEKT=Objekt
  if Objekt.getType()=="Mesh":#Vorpruefung auf Mesh-Objekt
    Gitter = Objekt.getData()
    for i in Gitter.verts:
      if i.sel:ORIGINALWERTE.append([i.co[0],i.co[1],i.co[2]])
      if i.sel:ORIGINALINDEX.append(i.index)
  ORIGINALINDEX.sort()
  Blender.Window.EditMode(Ausgangszustand)

# Testet ob gespeichertes Objekt noch aktuell ist
def oritest():
  global ORIGINALWERTE
  global ORIGINALOBJEKT
  global ORIGINALINDEX
  anzahlPunkte=0
  indexliste=[]
  Ausgangszustand = Blender.Window.EditMode()
  if Ausgangszustand==1:Blender.Window.EditMode(0)
  Objekt = Blender.Object.GetSelected()[0]
  if Objekt.getType()=="Mesh":#Vorpruefung auf Mesh-Objekt
    Gitter = Objekt.getData()
    for i in Gitter.verts:
      if i.sel:
        indexliste.append(i.index)
        anzahlPunkte+=1
  indexliste.sort()
  if len(ORIGINALWERTE)!=anzahlPunkte or indexliste!=ORIGINALINDEX or ORIGINALOBJEKT!=Objekt:original()
  Blender.Window.EditMode(Ausgangszustand)


# Loeschen aller Aenderungen
def undochange(Gitter):
  Zaehler = 0
  for i in Gitter.verts:
    if i.sel:
      i.co[0]=ORIGINALWERTE[Zaehler][0]
      i.co[1]=ORIGINALWERTE[Zaehler][1]
      i.co[2]=ORIGINALWERTE[Zaehler][2]
      Zaehler+=1
    Gitter.update()
```

```
# sucht die Objektgrenzen
def limitfind(Objekt,lokal=1):
  if lokal==1:hilf=Objekt.getData().verts[0].co
  else:hilf=mulmatvec4x3(Objekt.getData().verts[0].co,Objekt.getMatrix())
  grenzen=[hilf[0],hilf[0],hilf[1],hilf[1],hilf[2],hilf[2]]
  for i in Objekt.getData().verts:
    if lokal==1:a=i.co
    else:a=mulmatvec4x3(i.co,Objekt.getMatrix())
    if a[0]<grenzen[0]:grenzen[0]=a[0]
    if a[0]>grenzen[1]:grenzen[1]=a[0]
    if a[1]<grenzen[2]:grenzen[2]=a[1]
    if a[1]>grenzen[3]:grenzen[3]=a[1]
    if a[2]<grenzen[4]:grenzen[4]=a[2]
    if a[2]>grenzen[5]:grenzen[5]=a[2]
  return grenzen


# Angleichen des Objektes an den 3D-View
def alignview():
  for i in Blender.Object.GetSelected():
    if i.getType()=="Mesh":#Vorpruefung auf Mesh-Objekt
      sm=Blender.Window.GetViewMatrix()
      m=Blender.Mathutils.Matrix([sm[0][0],sm[1][0],sm[2][0]],[sm[0][1],sm[1][1],sm[2][1]],[sm[0][2],sm
      pi2=180/3.1415926
      i.setEuler(m[0]/pi2,m[1]/pi2,m[2]/pi2)
  return

##
# grafische Oberflaeche
##
def fenster():
  global EREIGNIS_ZEICHNEN,EREIGNIS_BEENDEN,EREIGNIS_EINGABE,AENDERE_X,AENDERE_Y,AENDERE_Z,XX,YY,ZZ
  global EREIGNIS_LOKAL,EREIGNIS_GLOBAL,GLO_X,GLO_Y,GLO_Z,LOK_X,LOK_Y,LOK_Z,EREIGNIS_VIEWPORT
  global DIR_X,DIR_Y,DIR_Z,EREIGNIS_OBJEKT,EREIGNIS_KEIN,ZIEL,QUELLE,RICHTUNG

  Blender.BGL.glColor3f(1,1,1)
  Blender.BGL.glRasterPos2i(175,120)
  Blender.Draw.Text("PointAlign 0.7")
  Blender.BGL.glRasterPos2i(175,105)
  Blender.Draw.Text("by Thomas Buschhardt")

###3D Cursor
  AENDERE_X=Blender.Draw.Toggle("X",EREIGNIS_ZEICHNEN,10,50,50,15,AENDERE_X.val,"an X-Achse angleichen")
  AENDERE_Y=Blender.Draw.Toggle("Y",EREIGNIS_ZEICHNEN,10,30,50,15,AENDERE_Y.val,"an Y-Achse angleichen")
  AENDERE_Z=Blender.Draw.Toggle("Z",EREIGNIS_ZEICHNEN,10,10,50,15,AENDERE_Z.val,"an Z-Achse angleichen")

  XX=Blender.Draw.String("",EREIGNIS_EINGABE,70,50,90,15,XX.val,30,"tool")
  YY=Blender.Draw.String("",EREIGNIS_EINGABE,70,30,90,15,YY.val,30,"tool")
  ZZ=Blender.Draw.String("",EREIGNIS_EINGABE,70,10,90,15,ZZ.val,30,"tool")

  Blender.Draw.PushButton("Refresh",EREIGNIS_CURSOR,70,70,90,15,"tool")

  Blender.BGL.glColor3f(0,0,0)
  Blender.BGL.glRasterPos2i(8,87)
```

```
  Blender.Draw.Text("3D-Cursor")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
  Blender.BGL.glVertex2i(5,90)
  Blender.BGL.glVertex2i(5,5)
  Blender.BGL.glVertex2i(165,5)
  Blender.BGL.glVertex2i(165,90)
  Blender.BGL.glVertex2i(70,90)
  Blender.BGL.glEnd()

###lokale
  LOK_X=Blender.Draw.Slider(" X ",EREIGNIS_LOKAL,175,50,90,15,LOK_X.val,-1,1)
  LOK_Y=Blender.Draw.Slider(" Y ",EREIGNIS_LOKAL,175,30,90,15,LOK_Y.val,-1,1)
  LOK_Z=Blender.Draw.Slider(" Z ",EREIGNIS_LOKAL,175,10,90,15,LOK_Z.val,-1,1)

  Blender.BGL.glColor3f(1,1,1)
  Blender.BGL.glRasterPos2i(175,70)
  Blender.Draw.Text("-1=min   1=max")
  Blender.BGL.glColor3f(0,0,0)
  Blender.BGL.glRasterPos2i(172,87)
  Blender.Draw.Text("local Limits")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
  Blender.BGL.glVertex2i(170,90)
  Blender.BGL.glVertex2i(170,5)
  Blender.BGL.glVertex2i(270,5)
  Blender.BGL.glVertex2i(270,90)
  Blender.BGL.glVertex2i(235,90)
  Blender.BGL.glEnd()

###globale
  GLO_X=Blender.Draw.Slider(" X ",EREIGNIS_GLOBAL,280,50,90,15,GLO_X.val,-1,1)
  GLO_Y=Blender.Draw.Slider(" Y ",EREIGNIS_GLOBAL,280,30,90,15,GLO_Y.val,-1,1)
  GLO_Z=Blender.Draw.Slider(" Z ",EREIGNIS_GLOBAL,280,10,90,15,GLO_Z.val,-1,1)

  Blender.BGL.glColor3f(1,1,1)
  Blender.BGL.glRasterPos2i(280,70)
  Blender.Draw.Text("-1=min   1=max")
  Blender.BGL.glColor3f(0,0,0)
  Blender.BGL.glRasterPos2i(277,87)
  Blender.Draw.Text("global Limits")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
  Blender.BGL.glVertex2i(275,90)
  Blender.BGL.glVertex2i(275,5)
  Blender.BGL.glVertex2i(375,5)
  Blender.BGL.glVertex2i(375,90)
  Blender.BGL.glVertex2i(350,90)
  Blender.BGL.glEnd()

###Viewport
  Blender.Draw.PushButton("Align",EREIGNIS_VIEWPORT,10,105,150,15,"tool")

  Blender.BGL.glColor3f(0,0,0)
  Blender.BGL.glRasterPos2i(8,127)
  Blender.Draw.Text("Align to 3D-view")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
```

```
  Blender.BGL.glVertex2i(5,130)
  Blender.BGL.glVertex2i(5,100)
  Blender.BGL.glVertex2i(165,100)
  Blender.BGL.glVertex2i(165,130)
  Blender.BGL.glVertex2i(107,130)
  Blender.BGL.glEnd()

###Objektangleich
  Blender.BGL.glColor3f(0,0,0)
  Blender.BGL.glRasterPos2i(8,332)
  Blender.Draw.Text("Align to Object")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
  Blender.BGL.glVertex2i(5,335)
  Blender.BGL.glVertex2i(5,140)
  Blender.BGL.glVertex2i(210,140)
  Blender.BGL.glVertex2i(210,335)
  Blender.BGL.glVertex2i(95,335)
  Blender.BGL.glEnd()

  Blender.BGL.glColor3f(1,1,1)
  Blender.BGL.glRasterPos2i(15,192)
  Blender.Draw.Text("0=min  1=max  2=mid  3=pivot")
  Blender.BGL.glColor3f(0,0,0)
  Blender.BGL.glRasterPos2i(12,207)
  Blender.Draw.Text("Target-Object")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
  Blender.BGL.glVertex2i(10,210)
  Blender.BGL.glVertex2i(10,165)
  Blender.BGL.glVertex2i(205,165)
  Blender.BGL.glVertex2i(205,210)
  Blender.BGL.glVertex2i(95,210)
  Blender.BGL.glEnd()
  ZIEL=Blender.Draw.Slider(" Objectvalue ",EREIGNIS_KEIN,15,170,185,15,ZIEL.val,0,3)

  Blender.BGL.glColor3f(1,1,1)
  Blender.BGL.glRasterPos2i(15,249)
  Blender.Draw.Text("0=min  1=max  2=mid  3=pivot")
  Blender.BGL.glColor3f(0,0,0)
  Blender.BGL.glRasterPos2i(12,262)
  Blender.Draw.Text("Source-Object")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
  Blender.BGL.glVertex2i(10,265)
  Blender.BGL.glVertex2i(10,220)
  Blender.BGL.glVertex2i(205,220)
  Blender.BGL.glVertex2i(205,265)
  Blender.BGL.glVertex2i(100,265)
  Blender.BGL.glEnd()
  QUELLE=Blender.Draw.Slider(" Objectvalue ",EREIGNIS_KEIN,15,225,185,15,QUELLE.val,0,3)

  RICHTUNG=Blender.Draw.Slider(" Direction ",EREIGNIS_KEIN,15,280,185,15,RICHTUNG.val,0,2)
  Blender.BGL.glColor3f(1,1,1)
  Blender.BGL.glRasterPos2i(15,302)
  Blender.Draw.Text("0=X  1=Y  2=Z")
  Blender.BGL.glColor3f(0,0,0)
```

```
  Blender.BGL.glRasterPos2i(12,317)
  Blender.Draw.Text("Direction")
  Blender.BGL.glBegin(Blender.BGL.GL_LINE_STRIP)
  Blender.BGL.glVertex2i(10,320)
  Blender.BGL.glVertex2i(10,275)
  Blender.BGL.glVertex2i(205,275)
  Blender.BGL.glVertex2i(205,320)
  Blender.BGL.glVertex2i(65,320)
  Blender.BGL.glEnd()


  Blender.Draw.PushButton("Align",EREIGNIS_OBJEKT,10,145,195,15,"tool")
  Blender.Draw.Redraw(1)

##
# Auswerten von allgemeinen Ereignissen
##
def allg_ereignis(erg,wert):
  if erg==Blender.Draw.ESCKEY or erg==Blender.Draw.QKEY:schalter_ereignis(EREIGNIS_BEENDEN)

##
# Auswerten von Schalter-Ereignissen
##
def schalter_ereignis(erg):
  global XX,YY,ZZ,LOK_X,LOK_Y,LOK_Z
  if erg==EREIGNIS_BEENDEN:Blender.Draw.Exit()
  elif erg==EREIGNIS_CURSOR:
    XX.val = str(Blender.Window.GetCursorPos()[0])
    YY.val = str(Blender.Window.GetCursorPos()[1])
    ZZ.val = str(Blender.Window.GetCursorPos()[2])
  elif erg==EREIGNIS_LOKAL:
    GLO_X.val=0
    GLO_Y.val=0
    GLO_Z.val=0
    AENDERE_X.val=0
    AENDERE_Y.val=0
    AENDERE_Z.val=0
    Ausgangszustand = Blender.Window.EditMode()
    if Ausgangszustand==1:Blender.Window.EditMode(0)
     Objekt = Blender.Object.GetSelected()[0]
    if Objekt.getType()=="Mesh":#Vorpruefung auf Mesh-Objekt
      Gitter = Objekt.getData()
      undochange(Gitter)
      grenze=limitfind(Objekt,1)
      for i in Gitter.verts:
        if i.sel:
          if LOK_X.val==-1:i.co[0]=grenze[0]
          if LOK_X.val==1:i.co[0]=grenze[1]
          if LOK_Y.val==-1:i.co[1]=grenze[2]
          if LOK_Y.val==1:i.co[1]=grenze[3]
          if LOK_Z.val==-1:i.co[2]=grenze[4]
          if LOK_Z.val==1:i.co[2]=grenze[5]
        Gitter.update()
    Blender.Window.EditMode(Ausgangszustand)
```

```
elif erg==EREIGNIS_GLOBAL:
  LOK_X.val=0
  LOK_Y.val=0
  LOK_Z.val=0
  AENDERE_X.val=0
  AENDERE_Y.val=0
  AENDERE_Z.val=0
  Ausgangszustand = Blender.Window.EditMode()
  if Ausgangszustand==1:Blender.Window.EditMode(0)
   Objekt = Blender.Object.GetSelected()[0]
  if Objekt.getType()=="Mesh":#Vorpruefung auf Mesh-Objekt
    Gitter = Objekt.getData()
    undochange(Gitter)
    grenze=limitfind(Objekt,0)
    for i in Gitter.verts:
      if i.sel:
        a=mulmatvec4x3(i.co,Objekt.getMatrix())
        ziel=a
        if GLO_X.val==-1:ziel[0]=grenze[0]
        if GLO_X.val==1:ziel[0]=grenze[1]
        if GLO_Y.val==-1:ziel[2]=grenze[4]
        if GLO_Y.val==1:ziel[2]=grenze[5]
        if GLO_Z.val==-1:ziel[1]=grenze[2]
        if GLO_Z.val==1:ziel[1]=grenze[3]
        objektmatrix = mulmatvec4x3(ziel,Objekt.getInverseMatrix())
        i.co[0] = objektmatrix[0]
        i.co[1] = objektmatrix[1]
        i.co[2] = objektmatrix[2]
      Gitter.update()
  Blender.Window.EditMode(Ausgangszustand)

elif erg==EREIGNIS_VIEWPORT:
  alignview()
  Blender.Redraw()
elif erg==EREIGNIS_OBJEKT:
  ix=[0,2,1][RICHTUNG.val]#0:x,1:y,2:z
  or_wert=QUELLE.val#0:min,1:max,2:mid,3:pivot
  ed_wert=ZIEL.val#
  erst_lim=limitfind(Blender.Object.GetSelected()[0],0)
  erst_loc=Blender.Object.GetSelected()[0].getLocation()
  for i in range(1,len(Blender.Object.GetSelected())):
    ob=Blender.Object.GetSelected()[i]
    ob_lim=limitfind(ob,0)
    ob_loc=ob.getLocation()
    Wert=[ob_loc[0],ob_loc[1],ob_loc[2]]
    if or_wert==2:teil1=(erst_lim[1+2*ix]-erst_lim[2*ix])/2.0+erst_lim[2*ix]+erst_loc[ix]
    elif or_wert==3:teil1=erst_loc[ix]
    else:teil1=erst_lim[or_wert+2*ix]
    if ed_wert==0:teil2=-ob_lim[2*ix]+ob_loc[ix]
    elif ed_wert==1:teil2=-ob_lim[1+2*ix]+ob_loc[ix]
    elif ed_wert==2:teil2=-((ob_lim[1+2*ix]-ob_lim[2*ix])/2.0+ob_lim[2*ix])+ob_loc[ix]
    else:teil2=0
    Wert[ix]=teil1+teil2
```

```
      ob.setLocation(Wert)
    Blender.Redraw()
  elif erg==EREIGNIS_ZEICHNEN:
    GLO_X.val=0
    GLO_Y.val=0
    GLO_Z.val=0
    LOK_X.val=0
    LOK_Y.val=0
    LOK_Z.val=0
    oritest()
    try:
      Ausgangszustand = Blender.Window.EditMode()
      if Ausgangszustand==1:Blender.Window.EditMode(0)
       Objekt = Blender.Object.GetSelected()[0]
      if Objekt.getType()=="Mesh":#Vorpruefung auf Mesh-Objekt
        Gitter = Objekt.getData()
        undochange(Gitter)
        ##
        # Punktkoordinaten veraendern
        ##
        if AENDERE_X.val+AENDERE_Y.val+AENDERE_Z.val!=0:
          ziel=[float(XX.val),float(YY.val),float(ZZ.val)]
          for i in Gitter.verts:
            if i.sel:
              if AENDERE_X.val==0:ziel[0] = mulmatvec4x3(i.co,Objekt.getMatrix())[0]
              if AENDERE_Y.val==0:ziel[2] = mulmatvec4x3(i.co,Objekt.getMatrix())[2]
              if AENDERE_Z.val==0:ziel[1] = mulmatvec4x3(i.co,Objekt.getMatrix())[1]
              objektmatrix = mulmatvec4x3(ziel,Objekt.getInverseMatrix())
              i.co[0] = objektmatrix[0]
              i.co[1] = objektmatrix[1]
              i.co[2] = objektmatrix[2]
            Gitter.update()
        Blender.Redraw()
      Blender.Window.EditMode(Ausgangszustand)
    except:pass

original()

Blender.Draw.Register(fenster,allg_ereignis,schalter_ereignis)
```