3D Camera Controls in Python

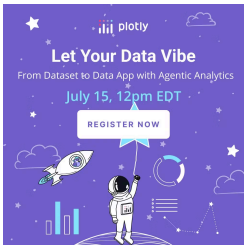
How to Control the Camera in your 3D Charts in Python with Plotly.

How camera controls work

The camera position and direction is determined by three vectors: *up*, *center*, *eye*. Their coordinates refer to the 3-d domain, i.e., (0, 0, 0) is always the center of the domain, no matter data values. The `eye` vector determines the position of the camera. The default is $(x=1.25, y=1.25, z=1.25)$.

The `up` vector determines the `up` direction on the page. The default is $(x=0, y=0, z=1)$, that is, the z-axis points up.

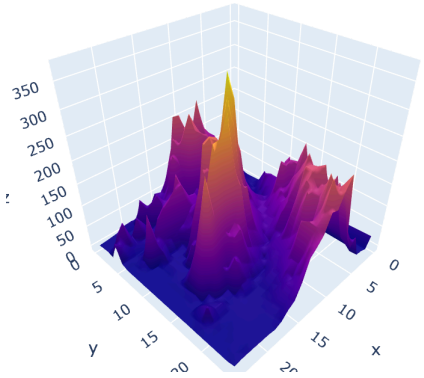
The projection of the `center` point lies at the center of the view. By default it is $(x=0, y=0, z=0)$.



Default parameters

```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=40, r=0, l=20, b=20)
)

name = 'default'
# Default parameters which are used when `layout.scene.camera` is not provided
camera = dict(
    up=dict(x=0, y=0, z=1),
    center=dict(x=0, y=0, z=0),
    eye=dict(x=1.25, y=1.25, z=1.25)
)

fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

default



Changing the camera position by setting the eye parameter

Lower the View Point

by setting `eye.z` to a smaller value.

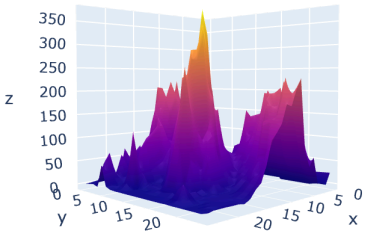
```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=30, r=0, l=20, b=10)
)

name = 'eye = (x:2, y:2, z:0.1)'
camera = dict(
    eye=dict(x=2, y=2, z=0.1)
)

fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

tting the

; the up

center

eye = (x:2, y:2, z:0.1)



X-Z plane

set eye.x and eye.z to zero

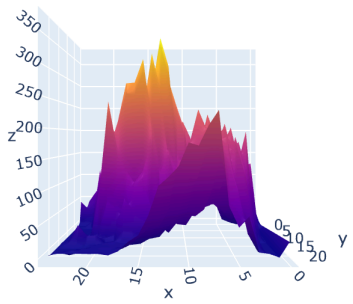
```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=30, r=0, l=20, b=10)
)

name = 'eye = (x:0., y:2.5, z:0.)'
camera = dict(
    eye=dict(x=0., y=2.5, z=0.)
)


fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

tting the

; the up

center

eye = (x:0., y:2.5, z:0.)



Y-Z plane

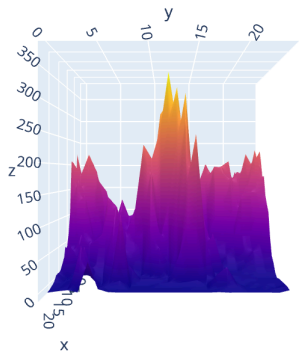
tting the

; the up

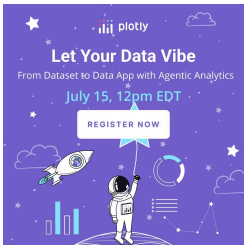
center

```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=30, r=0, l=20, b=10)
)

name = 'eye = (x:2.5, y:0., z:0.)'
camera = dict(
    eye=dict(x=2.5, y=0., z=0.)
)

fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

eye = (x:2.5, y:0., z:0.)



View from Above (X-Y plane)



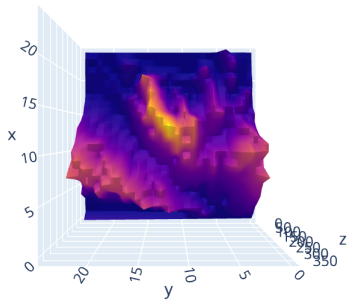
tting the

; the up

center

```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=30, r=0, l=20, b=10)
)

name = 'eye = (x:0., y:0., z:2.5)'
camera = dict(
    eye=dict(x=0., y=0., z=2.5)
)

fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

eye = (x:0., y:0., z:2.5)



Zooming In

... by placing the camera closer to the origin (eye with a smaller norm)

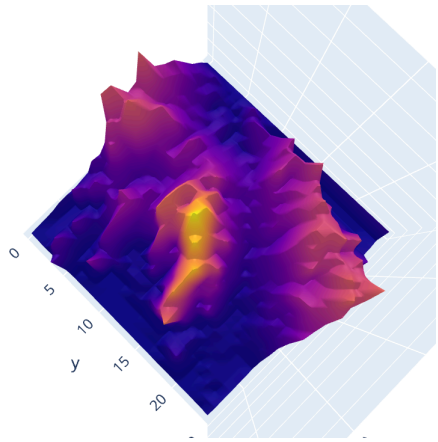
```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=30, r=0, l=20, b=10)
)

name = 'eye = (x:0.1, y:0.1, z:1.5)'
camera = dict(
    eye=dict(x=0.1, y=0.1, z=1.5)
)

fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

tting the

; the up

center

eye = (x:0.1, y:0.1, z:1.5)



Tilting the camera vertical by setting the up parameter

Tilt camera by changing the up vector: here the vertical of the view points in the x direction.

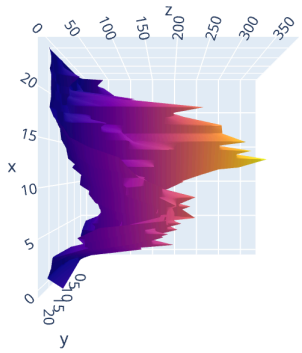
tting the

; the up

center

```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=30, r=0, l=20, b=10)
)

name = 'eye = (x:0., y:2.5, z:0.), point along x'
camera = dict(
    up=dict(x=1, y=0., z=0),
    eye=dict(x=0., y=2.5, z=0.)
)

fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

eye = (x:0., y:2.5, z:0.), point along x



Note when `up` does not correspond to the direction of an axis, you also need to set `layout.scene.dragmode='orbit'`.

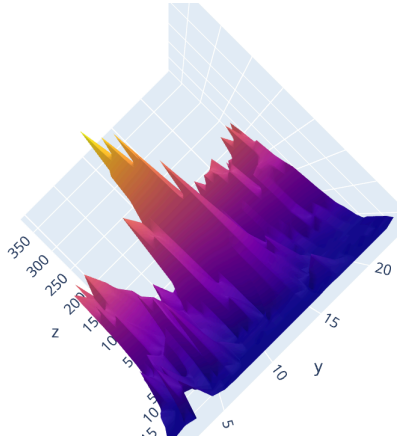
tting the

; the up

center

```
import math
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=30, r=0, l=20, b=10)
)

angle = math.pi / 4 # 45 degrees

name = 'vertical is along y+z'
camera = dict(
    up=dict(x=0, y=math.cos(angle), z=math.sin(angle)),
    eye=dict(x=2, y=0, z=0)
)

fig.update_layout(scene_camera=camera, scene_dragmode='orbit', title=name)
fig.show()
```

vertical is along y+z

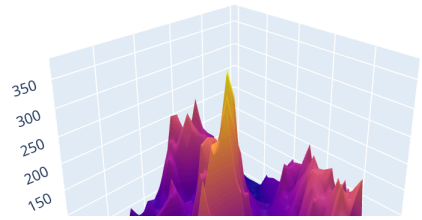


Changing the focal point by setting center

You can change the focal point (a point which projection lies at the center of the view) by setting the `center` parameter of `camera`. Note how a part of the data is cropped below because the camera is looking up.

```
import plotly.graph_objects as go
import pandas as pd

# Read data from a csv
z_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/api_docs/mt_bruno_elevation.csv')

fig = go.Figure(data=go.Surface(z=z_data, showscale=False))
fig.update_layout(
    title=dict(text='Mt Bruno Elevation'),
    width=400, height=400,
    margin=dict(t=25, r=0, l=20, b=30)
)

name = 'looking up'
camera = dict(
    center=dict(x=0, y=0, z=0.7))


fig.update_layout(scene_camera=camera, title=name)
fig.show()
```

looking up



Reference

See https://plotly.com/python/reference/layout/scene/#layout-scene-camera (https://plotly.com/python/reference/layout/scene/#layout-scene-camera) for more information and chart attribute options!

What About Dash?

Dash (https://dash.plot.ly/) is an open-source framework for building analytical applications, with no Javascript required, and it is tightly integrated with the Plotly graphing library.

Learn about how to install Dash at https://dash.plot.ly/installation (https://dash.plot.ly/installation).

Everywhere in this page that you see fig.show(), you can display the same figure in a Dash application by passing it to the figure argument of the Graph component (https://dash.plot.ly/dash-core-components/graph) from the built-in dash_core_components package like this:

```
import plotly.graph_objects as go # or plotly.express as px
fig = go.Figure() # or any Plotly Express function e.g. px.bar(...)
# fig.add_trace( ... )
# fig.update_layout( ... )

from dash import Dash, dcc, html


lv([
=fig)


use_reloader=False)  # Turn off reloader if inside Jupyter
```