
imgviz Documentation

Release 1.7.4

Matthew Matl

Aug 22, 2023

CONTENTS

1	Installation	3
1.1	Getting Started	3
1.2	API Reference	5
2	Indices and tables	17
	Index	19

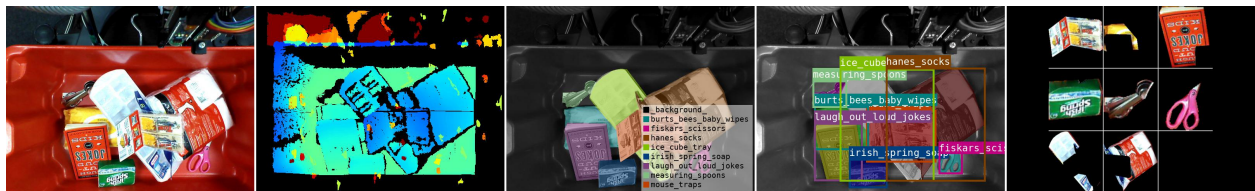
INSTALLATION

```
pip install imgviz
```

there are optional dependencies like skimage, below installs all.

```
pip install imgviz[all]
```

1.1 Getting Started



```
#!/usr/bin/env python
# flake8: noqa

import os.path as osp

import matplotlib.pyplot as plt

here = osp.dirname(osp.abspath(__file__)) # NOQA

# -----
# GETTING_STARTED {{
import imgviz

# sample data of rgb, depth, class label and instance masks
data = imgviz.data.arc2017()

rgb = data["rgb"]
gray = imgviz.rgb2gray(rgb)

# colorize depth image with JET colormap
depth = data["depth"]
depthviz = imgviz.depth2rgb(depth, min_value=0.3, max_value=1)

# colorize label image
```

(continues on next page)

(continued from previous page)

```
class_label = data["class_label"]
labelviz = imgviz.label2rgb(class_label, image=gray, label_names=data["class_names"],
    ↪font_size=20)

# instance bboxes
bboxes = data["bboxes"].astype(int)
labels = data["labels"]
masks = data["masks"] == 1
captions = [data["class_names"][l] for l in labels]
maskviz = imgviz.instances2rgb(gray, masks=masks, labels=labels, captions=captions)

# tile instance masks
insviz = [(rgb * m[:, :, None])[b[0] : b[2], b[1] : b[3]] for b, m in zip(bboxes, masks)]
insviz = imgviz.tile(imgs=insviz, border=(255, 255, 255))
insviz = imgviz.resize(insviz, height=rgb.shape[0])

# tile visualization
tiled = imgviz.tile(
    [rgb, depthviz, labelviz, maskviz, insviz],
    shape=(1, 5),
    border=(255, 255, 255),
    border_width=5,
)
# }} GETTING_STARTED
# -----

out_file = osp.join(here, ".readme/getting_started.jpg")
imgviz.io.imsave(out_file, tiled)

img = imgviz.io.imread(out_file)
plt.imshow(img)
plt.axis("off")
plt.show()
```


1.2 API Reference

1.2.1 Functions

<code>imgviz.asgray</code>	Convert any array to gray image.
<code>imgviz.gray2rgb</code>	Convert gray to rgb.
<code>imgviz.rgb2gray</code>	Convert rgb to gray.
<code>imgviz.rgb2rgba</code>	Convert rgb to rgba.
<code>imgviz.rgb2hsv</code>	Convert rgb to hsv.
<code>imgviz.rgba2rgb</code>	Convert rgba to rgb.
<code>imgviz.hsv2rgb</code>	Convert hsv to rgb.
<code>imgviz.depth2rgb</code>	Convert depth to rgb.
<code>imgviz.flow2rgb</code>	Visualize optical flow.
<code>imgviz.instances2rgb</code>	Convert instances to rgb.
<code>imgviz.label_colormap</code>	Label colormap.
<code>imgviz.label2rgb</code>	Convert label to rgb.
<code>imgviz.nchannel2rgb</code>	Convert nchannel array to rgb by PCA.
<code>imgviz.plot_trajectory</code>	Plot the trajectory using transform matrices
<code>imgviz.centerize</code>	Centerize image for specified image size
<code>imgviz.normalize</code>	Normalize image.
<code>imgviz.resize</code>	Resize image.
<code>imgviz.tile</code>	Tile images.

imgviz.asgray

`imgviz.asgray(img: ndarray) → ndarray`

Convert any array to gray image.

Parameters

img (*numpy.ndarray*) – Input image.

Returns

gray – Output gray image.

Return type

numpy.ndarray, (H, W), np.uint8

imgviz.gray2rgb

`imgviz.gray2rgb(gray: ndarray) → ndarray`

Convert gray to rgb.

Parameters

gray (*numpy.ndarray*, (H, W), np.uint8) – Input gray image.

Returns

rgb – Output rgb image.

Return type

numpy.ndarray, (H, W, 3), np.uint8

imgviz.rgb2gray

`imgviz.rgb2gray(rgb: ndarray) → ndarray`

Convert rgb to gray.

Parameters

rgb (*numpy.ndarray*, (*H*, *W*, 3), *np.uint8*) – Input rgb image.

Returns

gray – Output gray image.

Return type

numpy.ndarray, (*H*, *W*)

imgviz.rgb2rgba

`imgviz.rgb2rgba(rgb: ndarray) → ndarray`

Convert rgb to rgba.

Parameters

rgb (*numpy.ndarray*, (*H*, *W*, 3), *np.uint8*) – Input rgb image.

Returns

rgba – Output rgba image.

Return type

numpy.ndarray, (*H*, *W*, 4), *np.uint8*

imgviz.rgb2hsv

`imgviz.rgb2hsv(rgb: ndarray) → ndarray`

Convert rgb to hsv.

Parameters

rgb (*numpy.ndarray*, (*H*, *W*, 3), *np.uint8*) – Input rgb image.

Returns

hsv – Output hsv image.

Return type

numpy.ndarray, (*H*, *W*, 3), *np.uint8*

imgviz.rgba2rgb

`imgviz.rgba2rgb(rgba: ndarray) → ndarray`

Convert rgba to rgb.

Parameters

rgba (*numpy.ndarray*, (*H*, *W*, 4), *np.uint8*) – Input rgba image.

Returns

rgb – Output rgb image.

Return type

numpy.ndarray, (*H*, *W*, 3), *np.uint8*

imgviz.hsv2rgb

`imgviz.hsv2rgb(hsv: ndarray) → ndarray`

Convert hsv to rgb.

Parameters

hsv (*numpy.ndarray*, (*H*, *W*, 3), *np.uint8*) – Input hsv image.

Returns

rgb – Output rgb image.

Return type

numpy.ndarray, (*H*, *W*, 3), *np.uint8*

imgviz.depth2rgb

`imgviz.depth2rgb(depth: ~numpy.ndarray, min_value: ~typing.Optional[float] = None, max_value: ~typing.Optional[float] = None, colormap: str = 'jet', dtype: ~typing.Type = <class 'numpy.uint8'>) → ndarray`

Convert depth to rgb.

Parameters

- **depth** (*numpy.ndarray*, (*H*, *W*), *float*) – Depth image.
- **dtype** (*numpy.dtype*) – Dtype of output image. default: *np.uint8*
- **min_value** (*float*, *optional*) – Minimum value for colorizing.
- **max_value** (*float*, *optional*) – Maximum value for colorizing.
- **colormap** (*str*, *optional*) – Colormap, default: 'jet'.

Returns

rgb – Output colored image.

Return type

numpy.ndarray, (*H*, *W*, 3), *np.uint8*

imgviz.flow2rgb

`imgviz.flow2rgb(flow_uv)`

Visualize optical flow.

Parameters

flow_uv (*numpy.ndarray*, (*H*, *W*, 2), *float*) – Optical flow.

Returns

dst – RGB image.

Return type

numpy.ndarray

imgviz.instances2rgb

`imgviz.instances2rgb(image, labels, bboxes=None, masks=None, captions=None, font_size=25, line_width=5, boundary_width=1, alpha=0.7, colormap=None, font_path=None)`

Convert instances to rgb.

Parameters

- **image** (*numpy.ndarray*, (H, W, 3), *numpy.uint8*) – RGB image.
- **labels** (*list of int*, (N,)) – Labels.
- **bboxes** (*list of numpy.ndarray*, (N, 4), *float*) – Bounding boxes.
- **masks** (*numpy.ndarray*, (N, H, W), *bool*) – Masks.
- **captions** (*list of str*) – Captions.
- **font_size** (*int*) – Font size.
- **line_width** (*int*) – Line width.
- **alpha** (*float*) – Alpha of RGB.
- **colormap** (*numpy.ndarray*, (M, 3), *numpy.uint8*) – Label id to RGB color.

Returns

dst – Visualized image.

Return type

numpy.ndarray, (H, W, 3), *numpy.uint8*

imgviz.label_colormap

`imgviz.label_colormap(n_label=256, value=None)`

Label colormap.

Parameters

- **n_labels** (*int*) – Number of labels (default: 256).
- **value** (*float* or *int*) – Value scale or value of label color in HSV space.

Returns

cmap – Label id to colormap.

Return type

numpy.ndarray, (N, 3), *numpy.uint8*

imgviz.label2rgb

`imgviz.label2rgb(label, image=None, alpha=0.5, label_names=None, font_size=30, thresh_suppress=0, colormap=None, loc='rb', font_path=None)`

Convert label to rgb.

Parameters

- **label** (*numpy.ndarray*, (H, W), *int*) – Label image.
- **image** (*numpy.ndarray*, (H, W, 3), *numpy.uint8*) – RGB image.

- **alpha** (*float*, or *list* or *dict of float*) – Alpha of RGB (default: 0.5). If given as a list or dict, it is treated as alpha for each class according to the index or key.
- **label_names** (*list* or *dict of string*) – Label id to label name.
- **font_size** (*int*) – Font size (default: 30).
- **thresh_suppress** (*float*) – Threshold of label ratio in the label image.
- **colormap** (*numpy.ndarray*, (*M*, 3), *numpy.uint8*) – Label id to color. By default, `label_colormap()` is used.
- **loc** (*string*) – Location of legend (default: 'rb'). 'centroid', 'lt' and 'rb' are supported.
- **font_path** (*str*) – Font path.

Returns

res – Visualized image.

Return type

`numpy.ndarray`, (H, W, 3), `numpy.uint8`

imgviz.nchannel2rgb

`imgviz.nchannel2rgb(nchannel, dtype=<class 'numpy.uint8'>, pca=None)`

Convert nchannel array to rgb by PCA.

Parameters

- **nchannel** (*numpy.ndarray*, (*H*, *W*, *C*), *float*) – N channel image.
- **dtype** (*numpy.dtype*) – Dtype (default: `numpy.uint8`).
- **pca** (*sklearn.decomposition.PCA*) – PCA.

Returns

dst – Visualized image.

Return type

`numpy.ndarray`, (H, W, 3), `numpy.uint8`

imgviz.plot_trajectory

`imgviz.plot_trajectory(transforms, is_relative=False, mode='xz', style='b.', axis=True)`

Plot the trajectory using transform matrices

Parameters

- **transforms** (*numpy.ndarray*) – transform matrices with the shape of [N, 4, 4] where N is the # of poses.
- **is_relative** (*bool*) – True for relative poses. default: False.
- **mode** (*str*) – x and y axis of trajectory. default: 'xz' following kitti format.
- **style** (*str*) – style of plotting, default: 'b.'
- **axis** (*bool*) – False to disable axis.

Returns

dst – trajectory

Return type`numpy.ndarray`**imgviz.centerize**`imgviz.centerize(src, shape, cval=None, return_mask=False, interpolation='linear', loc='center')`

Centerize image for specified image size

Parameters

- **src** (`numpy.ndarray`) – Image to centerize
- **shape** (*tuple of int*) – Image shape (height, width) or (height, width, channel)
- **cval** (*int or float or numpy.ndarray*) – Color to be filled in the blank.
- **return_mask** (`numpy.ndarray`) – Mask for centerized image.
- **interpolation** (*str*) – Interpolation method (default: 'linear').
- **loc** (*str*) – Location of image ('center', 'lt', 'rb'). (default: 'center')

Returns**dst** – Centerized image.**Return type**`numpy.ndarray`**imgviz.normalize**`imgviz.normalize(src, min_value=None, max_value=None, return_minmax=False)`

Normalize image.

Parameters

- **src** (`numpy.ndarray`, (*H, W*) or (*H, W, C*), *float*) – Input image.
- **min_value** (*float*) – Minimum value.
- **max_value** (*float*) – Maximum value.
- **return_minmax** (*bool*) – Flag to return min_value and max_value.

Returns**dst** – Normalized image in [0, 1].**Return type**`numpy.ndarray, float`**imgviz.resize**`imgviz.resize(src, height=None, width=None, interpolation='linear', backend='auto')`

Resize image.

Parameters

- **src** (`numpy.ndarray`, (*H, W*) or (*H, W, C*)) – Input image.
- **height** (*int, optional*) – Height of image. If not given, the image is resized based on width keeping image ratio.

- **width** (*int*, *optional*) – Width of image. If not given, the image is resized based on height keeping image ratio.
- **interpolation** (*str*) – Resizing interpolation (default: 'linear').
 - 'linear':**
Linear interpolation.
 - 'nearest':**
Interpolate with the nearest value.
- **backend** (*str*) – Resizing backend (default: 'auto').
 - 'pillow':**
Pillow is used.
 - 'opencv':**
OpenCV is used.

Returns**dst** – Resized image.**Return type***numpy.ndarray***imgviz.tile**`imgviz.tile(imgs, shape=None, cval=None, border=None, border_width=None)`

Tile images.

Parameters

- **imgs** (*numpy.ndarray*) – Image list which should be tiled.
- **shape** (*tuple of int*) – Tile shape.
- **cval** (*array-like, optional*) – Color to fill the background. Default is (0, 0, 0).
- **border** (*array-like, optional*) – Color for the border. If None, the border is not drawn.
- **border_width** (*int*) – Pixel size of the border.

Returns**dst** – Tiled image.**Return type***numpy.ndarray***1.2.2 Classes**

<i>imgviz.Depth2RGB</i>	Convert depth array to rgb.
<i>imgviz.Nchannel2RGB</i>	Convert nchannel array to rgb by PCA.

imgviz.Depth2RGB

```
class imgviz.Depth2RGB(min_value=None, max_value=None, colormap='jet')
```

Convert depth array to rgb.

Parameters

- **min_value** (*float*, *optional*) – Minimum value for coloring.
- **max_value** (*float*, *optional*) – Maximum value for coloring.
- **colormap** (*str*, *optional*) – Colormap, default: 'jet'.

```
__init__(min_value=None, max_value=None, colormap='jet')
```

Methods

```
__init__([min_value, max_value, colormap])
```

Attributes

max_value	Maximum value of depth.
min_value	Minimum value of depth.

imgviz.Nchannel2RGB

```
class imgviz.Nchannel2RGB(pca=None)
```

Convert nchannel array to rgb by PCA.

Parameters

pca (*sklearn.decomposition.PCA*) – PCA.

```
__init__(pca=None)
```

Methods

```
__init__([pca])
```

Attributes

pca	PCA for N channel to 3.
-----	-------------------------

1.2.3 Draw Module

<code>imgviz.draw.circle</code>	Draw circle on numpy array with Pillow.
<code>imgviz.draw.rectangle</code>	Draw rectangle on numpy array with Pillow.
<code>imgviz.draw.star</code>	Draw star on numpy array with Pillow.
<code>imgviz.draw.text</code>	Draw text on numpy array with Pillow.
<code>imgviz.draw.text_in_rectangle</code>	Draw text in a rectangle.
<code>imgviz.draw.text_size</code>	Get text size (height and width).
<code>imgviz.draw.triangle</code>	Draw triangle on numpy array with Pillow.

imgviz.draw.circle

`imgviz.draw.circle(src, center, diameter, fill=None, outline=None, width=0)`

Draw circle on numpy array with Pillow.

Parameters

- **src** (*numpy.ndarray*) – Input image.
- **center** ((2,) array-like) – center is (cy, cx).
- **diameter** (*float*) – Diameter of the circle.
- **fill** (*int* or (3,) array-like, optional) – RGB color to fill the mark. None for no fill. (default: None)
- **outline** (*int* or (3,) array-like, optional) – RGB color to draw the outline.
- **width** (*int*, optional) – Rectangle line width. (default: 0)

Returns

dst – Output image.

Return type

numpy.ndarray

imgviz.draw.rectangle

`imgviz.draw.rectangle(src, aabb1, aabb2, fill=None, outline=None, width=0)`

Draw rectangle on numpy array with Pillow.

Parameters

- **src** (*numpy.ndarray*) – Input image.
- **aabb1** (array-like, (2,)) – Minimum vertex (y_min, x_min) of the axis aligned bounding box (AABB).
- **aabb2** (array-like, (2,)) – Maximum vertex (y_max, x_max) of the AABB.
- **fill** (*int* or array-like, (3,), optional) – RGB color to fill the mark. None for no fill. (default: None)
- **outline** (*int* or array-like, (3,), optional) – RGB color to draw the outline.
- **width** (*int*, optional) – Rectangle line width. (default: 0)

Returns

dst – Output image.

Return type`numpy.ndarray`**imgviz.draw.star**`imgviz.draw.star(src, center, size, fill=None, outline=None)`

Draw star on numpy array with Pillow.

Parameters

- **src** (`numpy.ndarray`) – Input image.
- **center** ((2,) array-like) – center is (cy, cx).
- **size** (`float`) – Diameter to create the star.
- **fill** (`int` or (3,) array-like, optional) – RGB color to fill the mark. None for no fill. (default: None)
- **outline** (`int` or (3,) array-like, optional) – RGB color to draw the outline.

Returns

dst – Output image.

Return type`numpy.ndarray`**imgviz.draw.text**`imgviz.draw.text(src, yx, text, size, color=(0, 0, 0), font_path=None)`

Draw text on numpy array with Pillow.

Parameters

- **src** (`numpy.ndarray`) – Input image.
- **yx** ((2,) array-like) – Left top point of the text.
- **text** (`str`) – Text to draw.
- **size** (`int`) – Text size in pixel.
- **color** ((3,) array-like) – Text RGB color in uint8. Default is (0, 0, 0), which is black.
- **font_path** (`str`) – Default font is DejaVuSansMono in matplotlib.

Returns

dst – Output image.

Return type`numpy.ndarray`

imgviz.draw.text_in_rectangle

`imgviz.draw.text_in_rectangle(src, loc, text, size, background, color=None, aabb1=None, aabb2=None, font_path=None, keep_size=False)`

Draw text in a rectangle.

Parameters

- **src** (*numpy.ndarray*) – Input image.
- **loc** (*str*) – Location of text. It must be one of following: lt, rt, lb, rb, lt+, rt+, lb-, rb-.
- **text** (*str*) – Text to draw.
- **size** (*int*) – Text size in pixel.
- **background** ((3,) *array-like*) – Background color in uint8.
- **color** ((3,) *array-like*) – Text RGB color in uint8. If None, the color is determined by background color. (default: None)
- **aabb1** ((2,) *array-like*) – Coordinate of the rectangle (y_min, x_min), (y_max, x_max). Default is (0, 0), (height, width).
- **aabb2** ((2,) *array-like*) – Coordinate of the rectangle (y_min, x_min), (y_max, x_max). Default is (0, 0), (height, width).
- **keep_size** (*bool*) – Force to keep original size (size change happens with loc=xx+, xx-).

Returns

dst – Output image.

Return type

numpy.ndarray

imgviz.draw.text_size

`imgviz.draw.text_size(text, size, font_path=None)`

Get text size (height and width).

Parameters

- **text** (*str*) – Text.
- **size** (*int*) – Pixel font size.

Returns

- **height** (*int*) – Text height.
- **width** (*int*) – Text width.

imgviz.draw.triangle

`imgviz.draw.triangle(src, center, size, fill=None, outline=None)`

Draw triangle on numpy array with Pillow.

Parameters

- **src** (*numpy.ndarray*) – Input image.
- **center** ((2,) array-like) – center is (cy, cx).
- **size** (*float*) – Diameter to create the star.
- **fill** (*int* or (3,) array-like, optional) – RGB color to fill the mark. None for no fill. (default: None)
- **outline** (*int* or (3,) array-like, optional) – RGB color to draw the outline.

Returns

dst – Output image.

Return type

numpy.ndarray

1.2.4 IO Module

<code>imgviz.io.imread</code>	Read image from file.
<code>imgviz.io.imsave</code>	Save image to file.
<code>imgviz.io.cv_imshow</code>	Show image with OpenCV.
<code>imgviz.io.cv_waitkey</code>	Wait key for the OpenCV window.
<code>imgviz.io.pyplot_to_numpy</code>	Convert pyplot state to numpy array.
<code>imgviz.io.pyglet_imshow</code>	Show image with pyglet.
<code>imgviz.io.pyglet_run</code>	Start pyglet mainloop.

imgviz.io.imread

imgviz.io.imsave

imgviz.io.cv_imshow

imgviz.io.cv_waitkey

imgviz.io.pyplot_to_numpy

imgviz.io.pyglet_imshow

imgviz.io.pyglet_run

INDICES AND TABLES

- `genindex`
- `search`

Symbols

`__init__()` (*imgviz.Depth2RGB method*), 12
`__init__()` (*imgviz.Nchannel2RGB method*), 12

A

`asgray()` (*in module imgviz*), 5

C

`centerize()` (*in module imgviz*), 10
`circle()` (*in module imgviz.draw*), 13

D

`Depth2RGB` (*class in imgviz*), 12
`depth2rgb()` (*in module imgviz*), 7

F

`flow2rgb()` (*in module imgviz*), 7

G

`gray2rgb()` (*in module imgviz*), 5

H

`hsv2rgb()` (*in module imgviz*), 7

I

`instances2rgb()` (*in module imgviz*), 8

L

`label2rgb()` (*in module imgviz*), 8
`label_colormap()` (*in module imgviz*), 8

N

`Nchannel2RGB` (*class in imgviz*), 12
`nchannel2rgb()` (*in module imgviz*), 9
`normalize()` (*in module imgviz*), 10

P

`plot_trajectory()` (*in module imgviz*), 9

R

`rectangle()` (*in module imgviz.draw*), 13

`resize()` (*in module imgviz*), 10
`rgb2gray()` (*in module imgviz*), 6
`rgb2hsv()` (*in module imgviz*), 6
`rgb2rgba()` (*in module imgviz*), 6
`rgba2rgb()` (*in module imgviz*), 6

S

`star()` (*in module imgviz.draw*), 14

T

`text()` (*in module imgviz.draw*), 14
`text_in_rectangle()` (*in module imgviz.draw*), 15
`text_size()` (*in module imgviz.draw*), 15
`tile()` (*in module imgviz*), 11
`triangle()` (*in module imgviz.draw*), 16