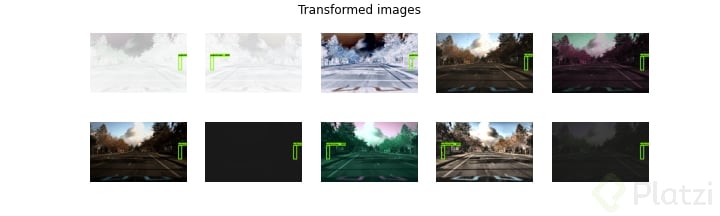
**Data augmentation**

Estuve leyendo la documentación de albumentations y conseguí las siguientes transfomaciones:



El código es el siguiente:

# Definición de transformaciones:

transforms = A.Compose([

A.HorizontalFlip(p=0.3),

A.RandomBrightnessContrast(p=0.3),

A.ChannelShuffle(p=0.3),

A.CLAHE(p=0.3),

A.ColorJitter(brightness=(0, 1), contrast=(0, 1), saturation=(0, 1), hue=(-0.3, 0.3), p=0.3),

A.FancyPCA(p=0.3, alpha=0.2, always\_apply=**False**),

A.GaussNoise(p=0.3),

A.InvertImg(p=0.3),

A.MultiplicativeNoise(p=0.3),

A.PixelDropout(p=0.3)

], bbox\_params = A.BboxParams(format='albumentations'))

# Creación de imágenes tranformadas:

transformed\_images = []

transformed\_bboxes = []

for i in range(len(transforms)):

t\_im = transforms(image=train\_images[0], bboxes=bbox)

t\_bbox = (list(t\_im['bboxes'][0])[:-1])

t\_bbox = [t\_bbox[1], t\_bbox[0], t\_bbox[3], t\_bbox[2]]

transformed\_images.append(t\_im)

transformed\_bboxes.append(t\_bbox)

# Visualzación de data augmentation

plt.figure(figsize=(10, 3))

for idx in range(0, 10):

bboxes = list(gt\_boxes.values())[idx]

dummy\_score = np.ones(shape=[bboxes.shape[0]], dtype=np.float32)

plt.subplot(2, 5, idx+1)

plt.axis('off')

plot\_detections(

transformed\_images[idx]['image'],

np.array([transformed\_bboxes[idx]]),

np.ones(shape=[bboxes.shape[0]], dtype=np.int32),

dummy\_score,

category\_index

)

plt.suptitle('Transformed images')

plt.savefig('transformed\_images.png')

plt.show()

Aquí les dejo la documentación de [Transformeciones con Albumentations](https://albumentations.ai/docs/api_reference/augmentations/transforms/)