In [ ]:
```
## import torch
import tqdm
import numpy as np
import cv2
import imageio
from tqdm.notebook import tqdm as tqdm
import matplotlib.pyplot as plt
from skimage.io import imread
from skimage.io import imsave
from skimage.transform import warp
from skimage.metrics import peak_signal_noise_ratio as compare_psnr
from skimage.metrics import structural_similarity as ssim
from utils.common_utils import *
import warnings
from torchsummary import summary
from skimage import segmentation
from networks.conv_layers import *
from networks.skip import skip
from networks.unet import UNet
CUDA_LAUNCH_BLOCKING=1

import os
os.environ['CUDA_VISIBLE_DEVICES'] = '0'
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print(device)

torch.backends.cudnn.enabled = True
torch.backends.cudnn.benchmark =True
dtype = torch.cuda.FloatTensor

warnings.filterwarnings("ignore")

imsize = (96,96)

batch_size = 25

pname = 'collect630'
start_f = 1
fname = 'Ablation_data/{}/img'.format(pname)
fgt = 'Ablation_data/{}/GT/{}_GT.png'.format(pname,pname)
# fgt = 'test_data/Our/Synthetic/Set1/GT/{}.jpg'.format(pname)
scale_factor = 1


fresult = 'result/{}'.format(pname)


if not os.path.exists(fresult):
    os.makedirs(fresult)
```

```
cuda
```

Utility Functions

In [ ]:
```
def backwarp(tenInput, tenFlow):
    backwarp_tenGrid = {}
```

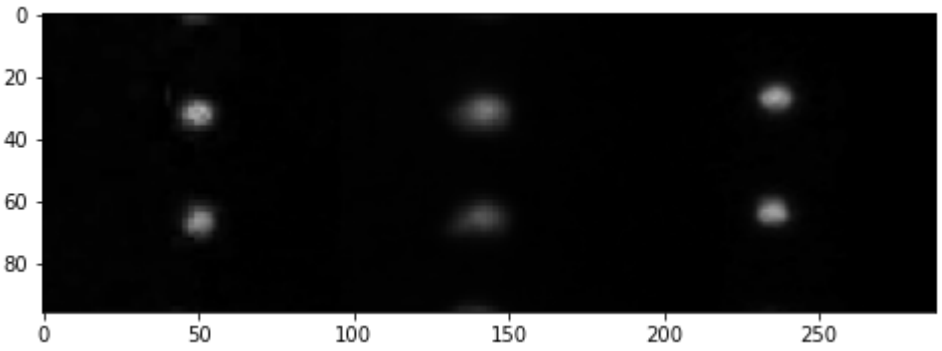
```
    if str(tenFlow.size()) not in backwarp_tenGrid:
        tenHorizontal = torch.linspace(-1.0, 1.0, tenFlow.shape[3]).view(1, 1,
        tenVertical = torch.linspace(-1.0, 1.0, tenFlow.shape[2]).view(1, 1, t

        backwarp_tenGrid[str(tenFlow.size())] = torch.cat([ tenHorizontal, ten
    # end

    tenFlow = torch.cat([ tenFlow[:, 0:1, :, :] / ((tenInput.shape[3] - 1.0) /

    return torch.nn.functional.grid_sample(input=tenInput, grid=(backwarp_tenG

def backwarp_grid(tenInput, tenFlow_xy):

    return torch.nn.functional.grid_sample(input=tenInput, grid = tenFlow_xy.p

def im_resize(im, scale_factor):
    width = int(im.size[1] * scale_factor)
    height = int(im.size[0] * scale_factor)
    newsize = (height,width)
#     im1 = im.resize(newsize)
    return im.resize(newsize)

def visualize_rgb(warp_np):
#     warp_np = warp_np.transpose(1,2,0)
    nr = warp_np.shape[0]
    nc = warp_np.shape[1]
    warp_np = (warp_np - np.amin(warp_np))/(np.amax(warp_np) - np.amin(warp_np
    one_pad = np.ones((nr, nc, 1))
    out_warp_np = np.concatenate((warp_np,one_pad),axis = -1)
    return out_warp_np

def visualize_rgb_norm(warp_np):
#     warp_np = warp_np.transpose(1,2,0)
    nr = warp_np.shape[0]
    nc = warp_np.shape[1]
#     warp_np = (warp_np - np.amin(warp_np))/(np.amax(warp_np) - np.amin(warp_
    one_pad = np.ones((nr, nc, 1))
    out_warp_np = np.concatenate((warp_np,one_pad),axis = -1)
    return out_warp_np

def has_file_allowed_extension(filename, extensions):
    """Checks if a file is an allowed extension.

    Args:
        filename (string): path to a file

    Returns:
        bool: True if the filename ends with a known image extension
    """
    filename_lower = filename.lower()
    return any(filename_lower.endswith(ext) for ext in extensions)
```

Load Turbulence Images

In [ ]:
```
# Load reference GT pattern. If none, load a single turbulence image
img_gt_rgb,img_gt_np = get_image(fgt, imsize)
img_gt_np  = img_gt_np[:3,:,:]
```

```
dim_gt = img_gt_np.shape[0]
if dim_gt ==1:
    img_gt_np = np.concatenate((img_gt_np, img_gt_np,img_gt_np), 0)

images = []
i = 0
# # Load turbulence image batch

extensions = ['.jpg', '.JPG', '.png', '.ppm', '.bmp', '.pgm', '.tif']
# Load image by sorted name
for target in sorted(os.listdir(fname)):
    d = os.path.join(fname, target)
    if has_file_allowed_extension(d, extensions) and i < batch_size:
#         print(d)
        i = i+1
        rgb, imgs = get_image(d, imsize)
        imgs = pil_to_np(im_resize(rgb,scale_factor))
        dim = imgs.shape[0]
        if dim ==1:
            imgs = np.concatenate((imgs, imgs,imgs), 0)
        images.append(imgs)


images_warp_np = np.array(images)
print(images_warp_np.shape)
images_mean_np = np.mean(images_warp_np, axis = 0)
# print(images_mean_np.shape)
dim, nr, nc = images_mean_np.shape


if dim>1:
    img_gt_np = cv2.resize(img_gt_np.transpose(1,2,0), dsize=(nc, nr), interpo
else:
    img_gt_np = cv2.resize(img_gt_np.transpose(1,2,0), dsize=(nc, nr), interpo

out_imshow = np.concatenate([images_warp_np[0].transpose(1,2,0),images_mean_np
plt.figure(figsize=(8,5))
plt.imshow(out_imshow)
plt.show()
```

```
(25, 3, 96, 96)
```



Setup Fourier Feature Transform function

In [ ]:

```
class GaussianFourierFeatureTransform_B(torch.nn.Module):
    """
```

```
    An implementation of Gaussian Fourier feature mapping.

    "Fourier Features Let Networks Learn High Frequency Functions in Low Dimen
       https://arxiv.org/abs/2006.10739
       https://people.eecs.berkeley.edu/~bmild/fourfeat/index.html

    Given an input of size [batches, num_input_channels, width, height],
     returns a tensor of size [batches, mapping_size*2, width, height].
    """

    def __init__(self, num_input_channels, B, mapping_size=256, scale=10):
        super().__init__()

        self._num_input_channels = num_input_channels
        self._mapping_size = mapping_size
        self._B = B*scale
#         self._B = torch.load('{}/{}_tensor_B.pt'.format(fresult,pname))

    def forward(self, x):
        assert x.dim() == 4, 'Expected 4D input (got {}D input)'.format(x.dim(

        batches, channels, width, height = x.shape

        assert channels == self._num_input_channels,\
            "Expected input to have {} channels (got {} channels)".format(self

        # Make shape compatible for matmul with _B.
        # From [B, C, W, H] to [(B*W*H), C].
        x = x.permute(0, 2, 3, 1).reshape(batches * width * height, channels)

        x = x @ self._B.to(x.device)

        # From [(B*W*H), C] to [B, W, H, C]
        x = x.view(batches, width, height, self._mapping_size)
        # From [B, W, H, C] to [B, C, W, H]
        x = x.permute(0, 3, 1, 2)

        x = 2 * np.pi * x
        return torch.cat([torch.sin(x), torch.cos(x)], dim=1)
```

In [ ]:
```
# Generate straight grid batch for shape image
xy_grid_batch = []
coords_x = np.linspace(-1, 1, nc)
coords_y = np.linspace(-1, 1, nr)
xy_grid = np.stack(np.meshgrid(coords_x, coords_y), -1)

xy_grid_var = np_to_torch(xy_grid.transpose(2,0,1)).type(dtype).cuda()
xy_grid_batch_var = xy_grid_var.repeat(batch_size, 1, 1, 1)
print(xy_grid_batch_var.shape)
```

```
torch.Size([25, 2, 96, 96])
```

Setup Image Generator

In [ ]:
```
model_imgen = conv_layers(256,3)
model_imgen = model_imgen.type(dtype)
# print(model_imgen)
# summary(model_imgen,(256, nr, nc))
```

```
torch.manual_seed(0)

B_var = torch.randn(2,128)
print(B_var.shape)
```

```
torch.Size([2, 128])
```

Setup Grid deformer network

In [ ]:
```
# Use the skip net as grid deformor

input_depth_warp = 8
pad = 'reflection'

# ------------------------------------Use 10 grid-deform networks -----------
model_grid = []
for i in range(batch_size):
    model_grid.append(conv_layers(2,2, need_sigmoid = False, need_tanh = True)

sum1 = summary(model_grid[0],(2, nr, nc))

# ------------------------------------Use 1 grid-deform networks ------------
# model_grid = conv_layers(2,2, need_sigmoid = False, need_tanh = True)

# model_grid = skip(2, 2,
#             num_channels_down = [128, 128, 128, 128, 128],
#             num_channels_up   = [128, 128, 128, 128, 128],
#             num_channels_skip = [16, 16, 16, 16, 16],
#             upsample_mode='bilinear',
#             need_sigmoid=False, need_tanh=True, need_bias=True, pad=pad, act

# model_grid = model_grid.type(dtype)
# # sum1 = summary(model_grid,(256, nr, nc))
# sum1 = summary(model_grid,(2, nr, nc))
```

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 256, 96, 96]             768
              ReLU-2          [-1, 256, 96, 96]               0
       BatchNorm2d-3          [-1, 256, 96, 96]             512
            Conv2d-4          [-1, 256, 96, 96]          65,792
              ReLU-5          [-1, 256, 96, 96]               0
            Conv2d-6          [-1, 256, 96, 96]          65,792
              ReLU-7          [-1, 256, 96, 96]               0
            Conv2d-8            [-1, 2, 96, 96]             514
              Tanh-9            [-1, 2, 96, 96]               0
================================================================
Total params: 133,378
Trainable params: 133,378
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.07
Forward/backward pass size (MB): 126.28
Params size (MB): 0.51
Estimated Total Size (MB): 126.86
----------------------------------------------------------------
```

Network Initializaiton

In [ ]:
```
# The input turbulent images

# The frequence bandwith for the turbulence field
FB_img = 8

vec_scale = 1.1

reg_noise_std = 1./30. # set to 1./30 works fine

img_gt_batch_var = torch.from_numpy(images_warp_np).type(dtype).cuda()
# straight_grid_input = GaussianFourierFeatureTransform_B(2, B_var, 128, FB_im
# ------------------------------SETUP Grid deformer-------------------------
grid_input_single_gd = xy_grid_var.detach().clone()
grid_input_gd = xy_grid_batch_var.detach().clone()
# --------------------------------------------------------------------------
grid_input = GaussianFourierFeatureTransform_B(2, B_var, 128, FB_img)(xy_grid_
print(sys.getsizeof(grid_input))
```

72

In [ ]:
```
model_params_list = [{'params':model_grid[i].parameters()} for i in range(batc

model_params_list.append({'params':model_imgen.parameters()})

# print(model_params_list)
```

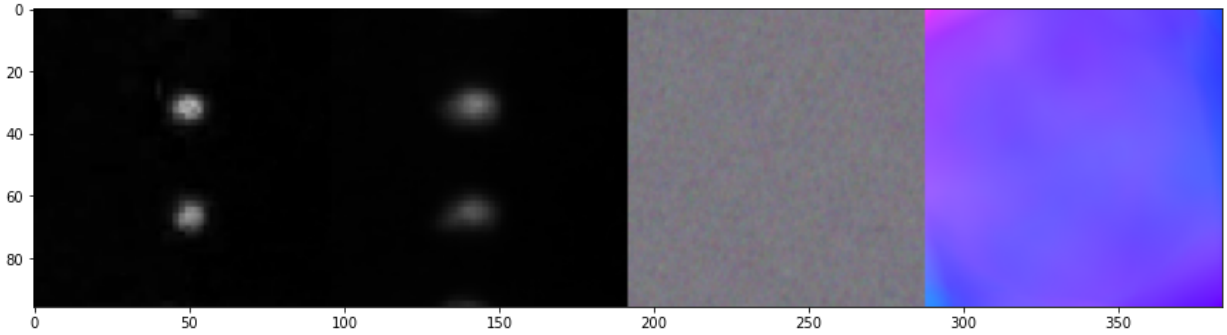
In [ ]:
```
optimizer = torch.optim.Adam(model_params_list, lr=1e-4)

num_iter_i = 1000

# imsave('{}/{}_turb_img_frame_{}.png'.format(fresult,pname,0), images_warp_np
imsave('{}/{}_avg_img_{}.png'.format(fresult,pname,batch_size), images_mean_np
for epoch in tqdm(range(num_iter_i)):
    optimizer.zero_grad()

# ------------------------------SETUP Grid deformer-------------------------
    refined_xy = []
    for b in range(batch_size):
        vec_input = grid_input_single_gd
        refined_xy.append(model_grid[b](vec_input))
    refined_xy = vec_scale*torch.cat(refined_xy)

    generated =  model_imgen(grid_input)

    loss = torch.nn.functional.l1_loss(img_gt_batch_var, generated)
    loss += torch.nn.functional.l1_loss(xy_grid_batch_var, refined_xy)

    loss.backward()
    optimizer.step()

    if epoch % 100 == 0:
        print('Epoch %d, loss = %.03f' % (epoch, float(loss)))
        out_img = generated[0].detach().cpu().numpy().transpose(1,2,0)
        pred_xy = refined_xy[0].detach().cpu().numpy().transpose(1,2,0)
        out_imshow = np.concatenate([images_warp_np[0].transpose(1,2,0),images
        plt.figure(figsize=(15,5))
```

```
        plt.imshow(np.clip(out_imshow,0,1))
        plt.show()

imsave('{}/{}_ini_{}_{}.png'.format(fresult,pname,FB_img,batch_size), out_img)
```

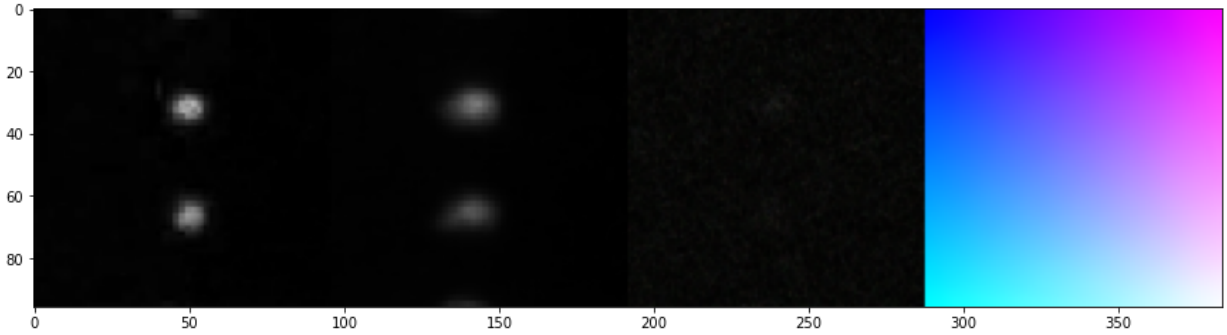
```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```

```
  0%|          | 0/1000 [00:00<?, ?it/s]
```

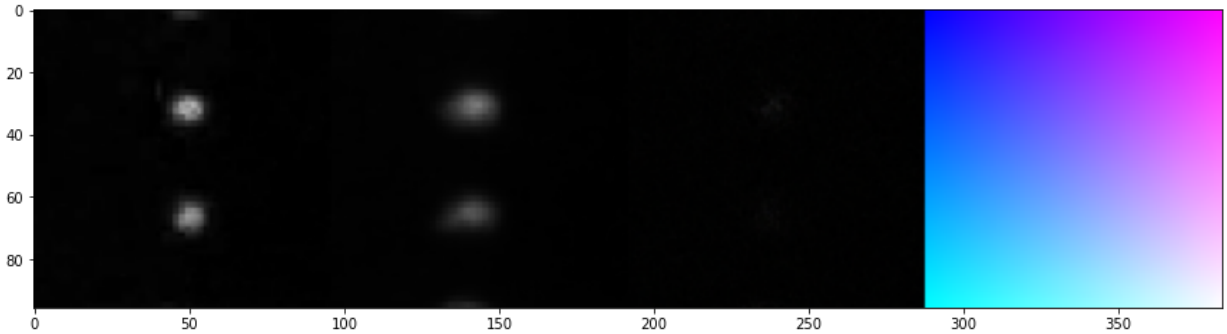
Epoch 0, loss = 0.981



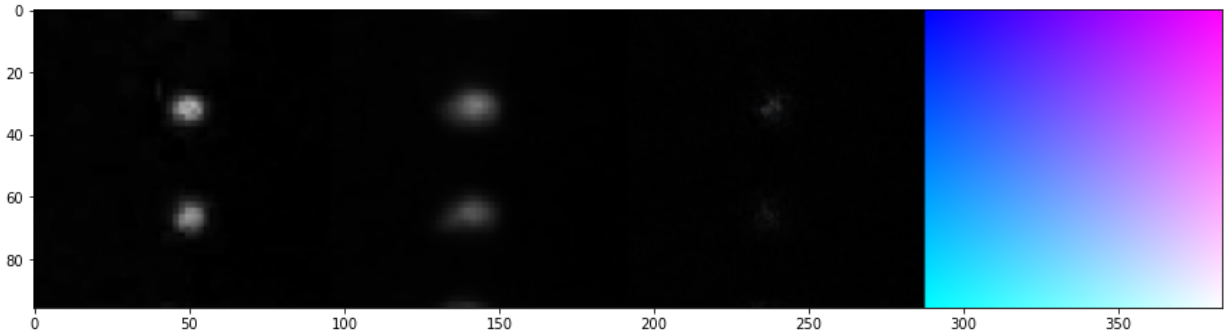
Epoch 100, loss = 0.035



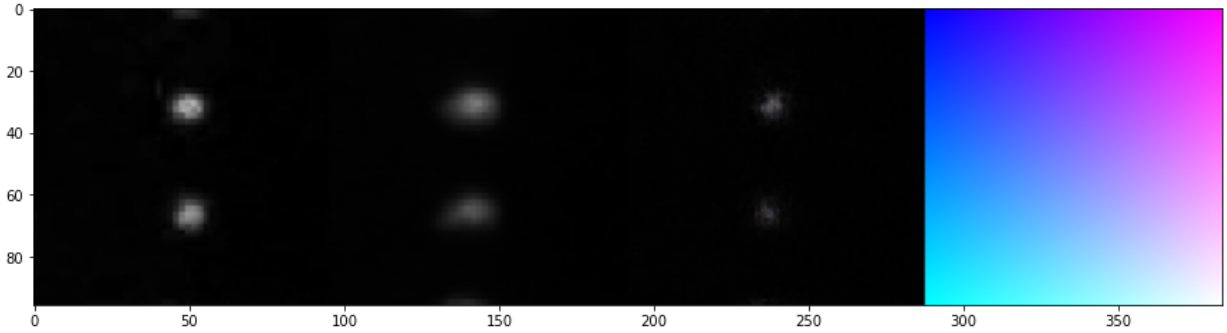
Epoch 200, loss = 0.016



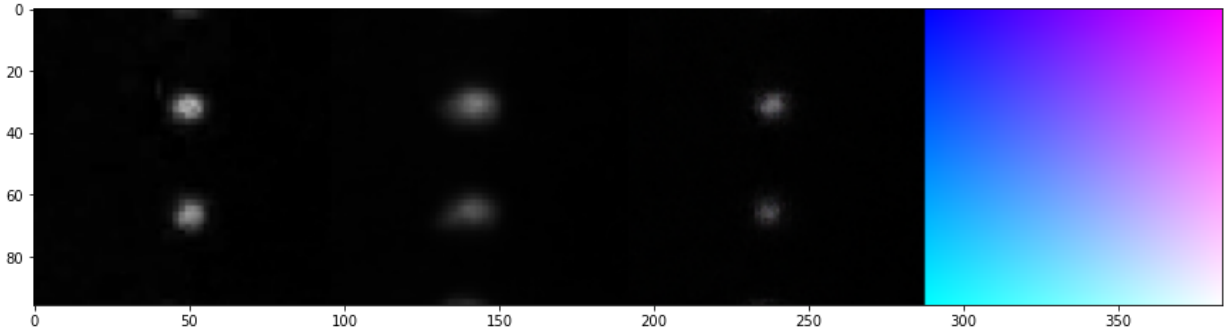
Epoch 300, loss = 0.015



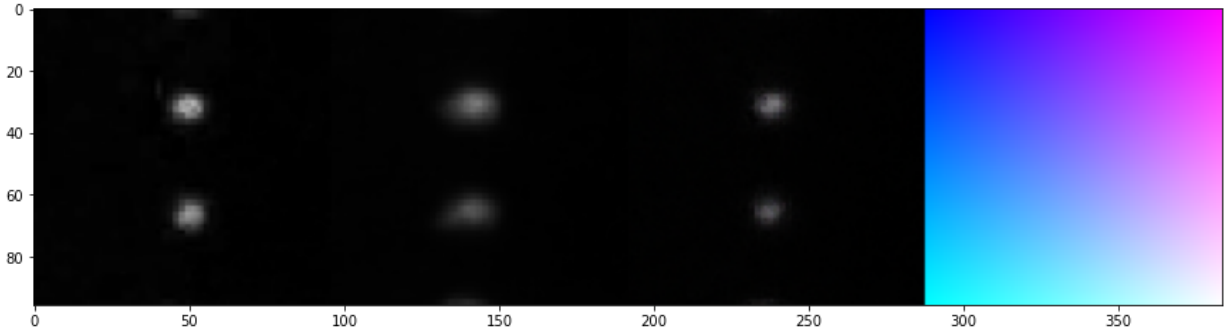
Epoch 400, loss = 0.013



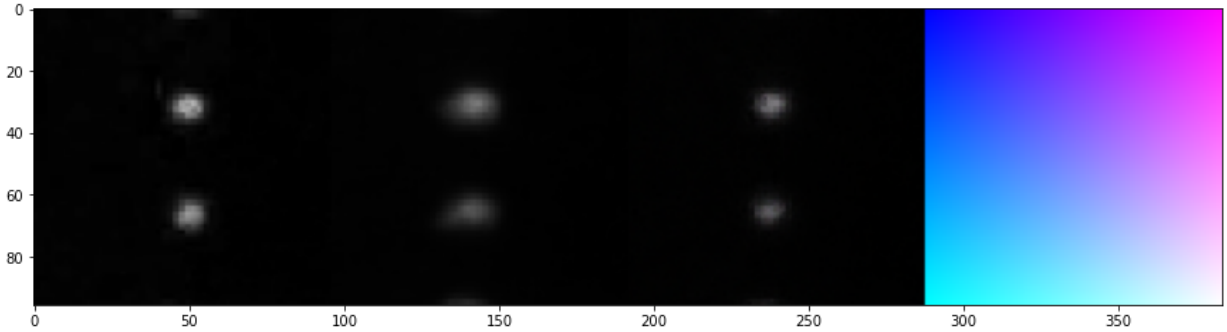
Epoch 500, loss = 0.013



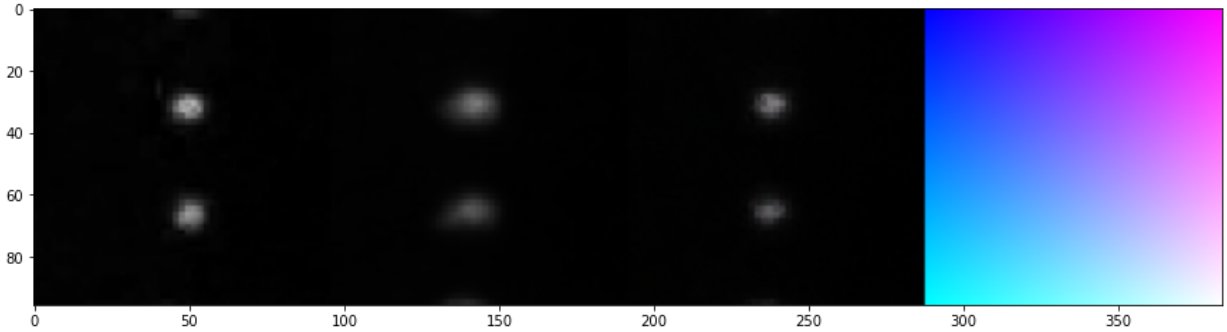
Epoch 600, loss = 0.012



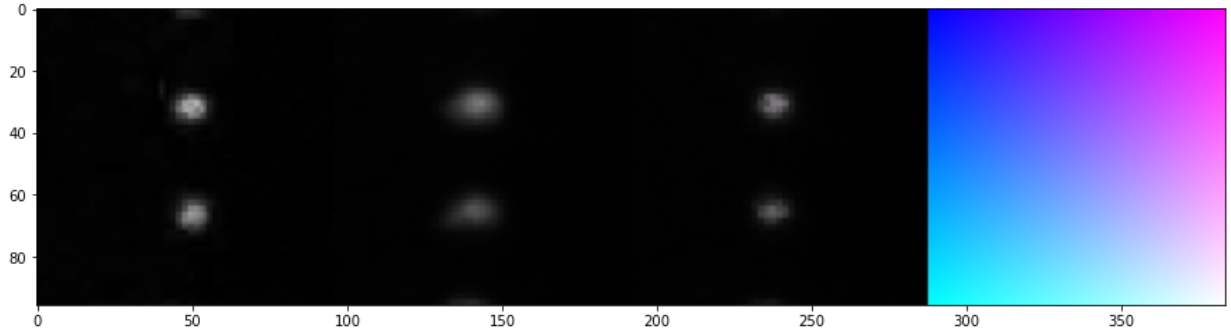
Epoch 700, loss = 0.012



Epoch 800, loss = 0.012



Epoch 900, loss = 0.012



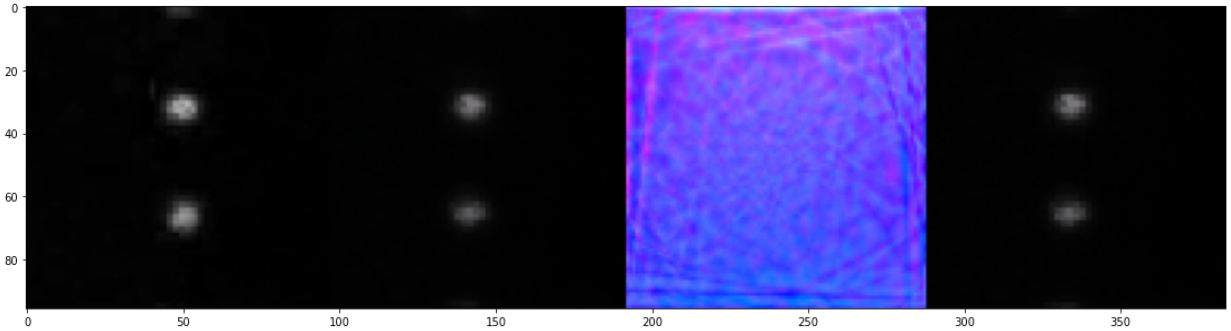

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```

In [ ]:
```
torch.save(model_imgen, '{}/{}_ig_scale_{}_FB_{}_{}.pth'.format(fresult,pname,
torch.save(model_grid, '{}/{}_gd_scale_{}_FB_{}_{}.pth'.format(fresult,pname,s
# model_imgen = torch.load('{}/{}_ig_scale_{}_FB_{}_{}.pth'.format(fresult,pna
# model_grid = torch.load('{}/{}_gd_scale_{}_FB_{}_{}.pth'.format(fresult,pnam

img_gt_np = img_gt_np.clip(0,1)
```

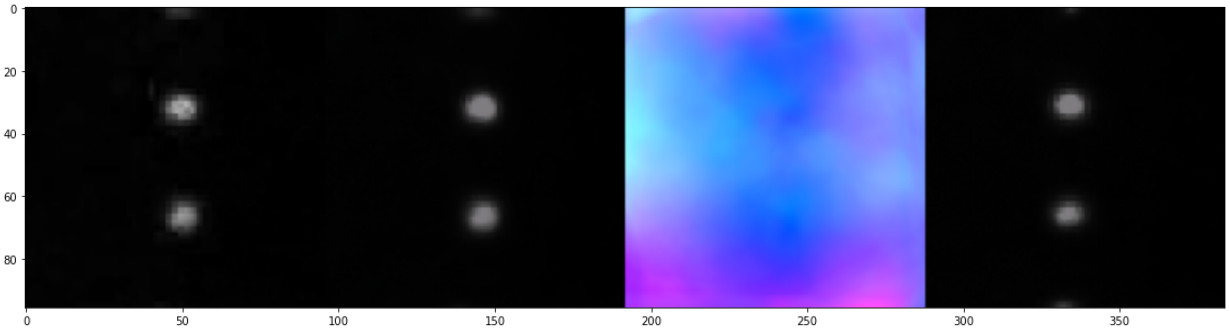
In [ ]:
```
num_iter = 1000
reg_noise_std = 1./30

# the reference frame in the plot
i = 0

loss_arr = torch.zeros(num_iter)
psnr_arr_sharp = torch.zeros(num_iter)
psnr_arr_turb = torch.zeros(num_iter)
ssim_arr_sharp = torch.zeros(num_iter)
ssim_arr_turb = torch.zeros(num_iter)
# seed = torch.seed()

optimizer = torch.optim.Adam(model_params_list, lr=1e-4)

for epoch in tqdm(range(num_iter)):

    optimizer.zero_grad()

# ------------------------------SETUP Grid deformer-------------------------
    refined_xy = []
    for b in range(batch_size):
        vec_input = grid_input_single_gd
        refined_xy.append(model_grid[b](vec_input))

    refined_xy = vec_scale*torch.cat(refined_xy)
    refined_warp = refined_xy - xy_grid_batch_var
    refined_uv = torch.cat(((nc - 1.0)*refined_warp[:, 0:1, :, :]/2 , (nr - 1.

    # Get mask for the warp field
    mask_u1 = (refined_xy[:,0:1,:,:] > -1).float() * 1
    mask_u2 = (refined_xy[:,0:1,:,:] < 1).float() * 1
    mask_v1 = (refined_xy[:,1:2,:,:] > -1).float() * 1
    mask_v2 = (refined_xy[:,1:2,:,:] < 1).float() * 1
    mask = mask_u1*mask_u2*mask_v1*mask_v2

    # predict sharp image using straight grid
    sharp_imgs_predict = model_imgen(grid_input)
```

```
    # predict turbulent image using forward mapping
    refined_turb_imgs = backwarp_grid(sharp_imgs_predict,refined_xy)

    # predict turbulent images using sampling grid
    generated_turb_imgs = model_imgen(GaussianFourierFeatureTransform_B(2, B_v

    # loss function
    loss = torch.nn.functional.l1_loss(generated_turb_imgs*mask,img_gt_batch_v
    loss += torch.nn.functional.l1_loss(refined_turb_imgs*mask,img_gt_batch_va
    loss += torch.nn.functional.l1_loss(generated_turb_imgs*mask,refined_turb_

#     loss += torch.nn.functional.l1_loss(img_gt_batch_var*mask, sharp_imgs_pr
#     loss += torch.nn.functional.l1_loss(xy_grid_batch_var, refined_xy)

    loss_arr[epoch] = loss
    psnr_arr_sharp[epoch] = compare_psnr(img_gt_np, sharp_imgs_predict[i].deta
    psnr_arr_turb[epoch]   = compare_psnr(images_warp_np[i], generated_turb_im
    ssim_arr_sharp[epoch] = float(ssim(img_gt_np.transpose(1,2,0), sharp_imgs_
    ssim_arr_turb[epoch]   = float(ssim(images_warp_np[i].transpose(1,2,0), ge

    loss.backward()
    optimizer.step()

    if epoch % 100 == 0:
        print('Epoch %d, loss = %.03f, psnr_sharp = %.03f, psnr_turb = %.03f' '

        out_img = refined_turb_imgs[i]*mask[i]
        out_img = out_img.detach().cpu().numpy().transpose(1,2,0)
        sharp_img = sharp_imgs_predict[i].detach().cpu().numpy().transpose(1,2
        warp_img= refined_uv[i].detach().cpu().numpy().transpose(1,2,0)
        out_target = images_warp_np[i].transpose(1,2,0)

        out_imshow = np.concatenate([out_target,out_img,visualize_rgb(warp_img
        plt.figure(figsize=(20,5))
        plt.imshow(np.clip(out_imshow,0,1))
        plt.show()
```

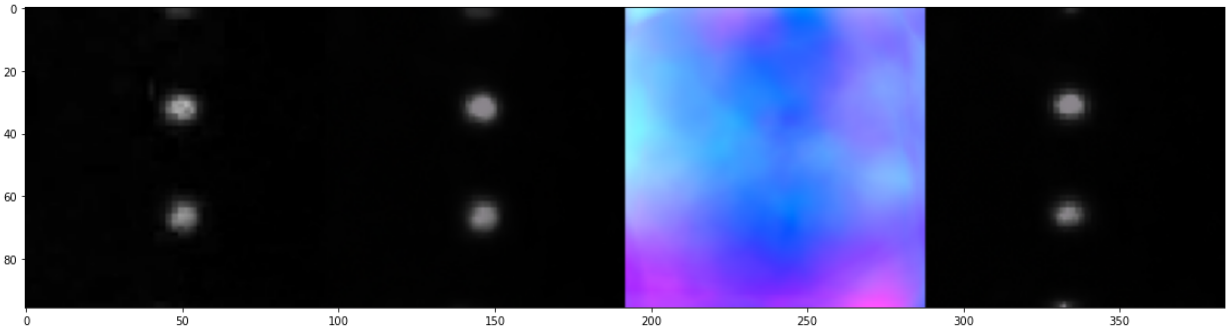
```
  0%|          | 0/1000 [00:00<?, ?it/s]
Epoch 0, loss = 0.021, psnr_sharp = 26.870, psnr_turb = 27.275
```



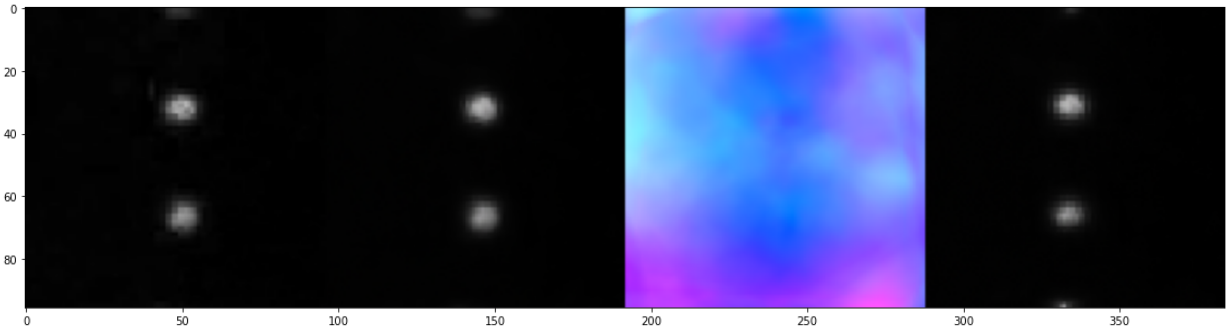

```
Epoch 100, loss = 0.009, psnr_sharp = 26.520, psnr_turb = 39.145
```



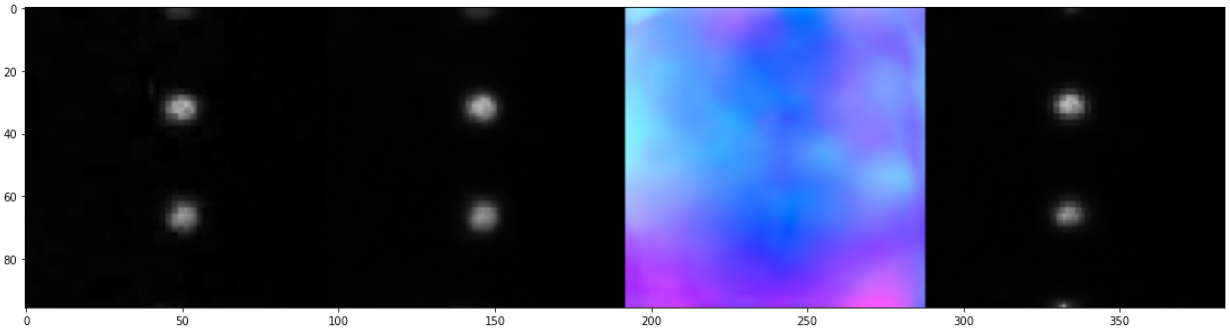
Epoch 200, loss = 0.008, psnr_sharp = 26.293, psnr_turb = 40.202



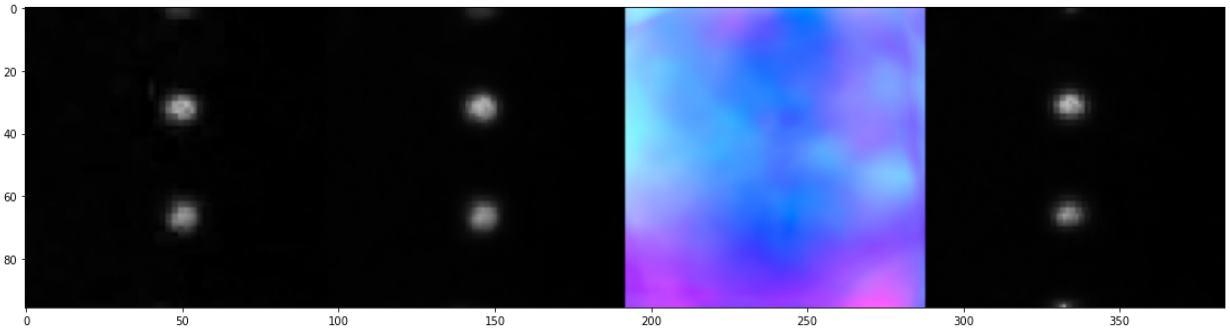
Epoch 300, loss = 0.008, psnr_sharp = 25.961, psnr_turb = 40.915



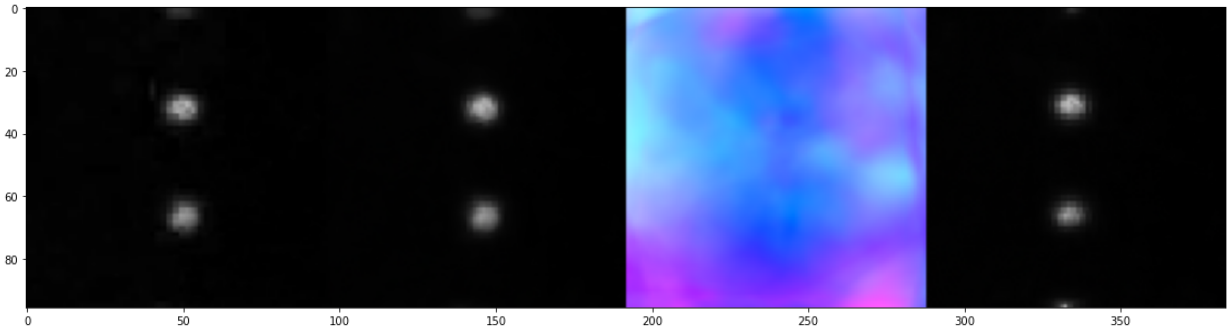
Epoch 400, loss = 0.007, psnr_sharp = 25.948, psnr_turb = 41.115



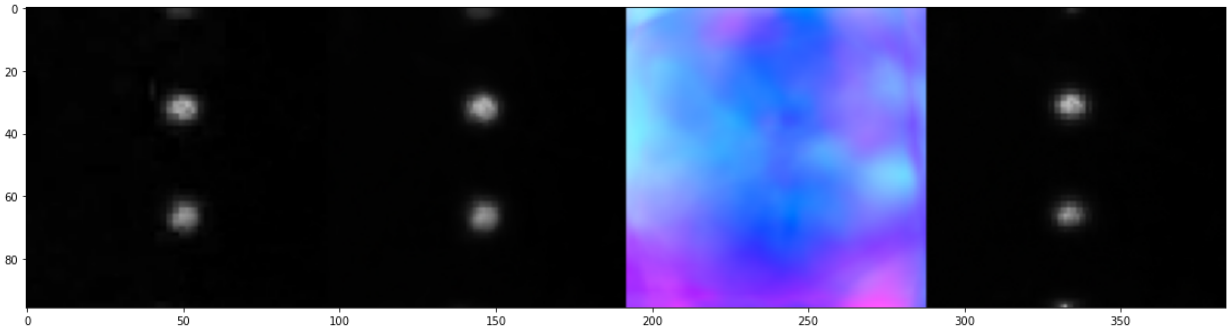
Epoch 500, loss = 0.007, psnr_sharp = 25.942, psnr_turb = 41.147



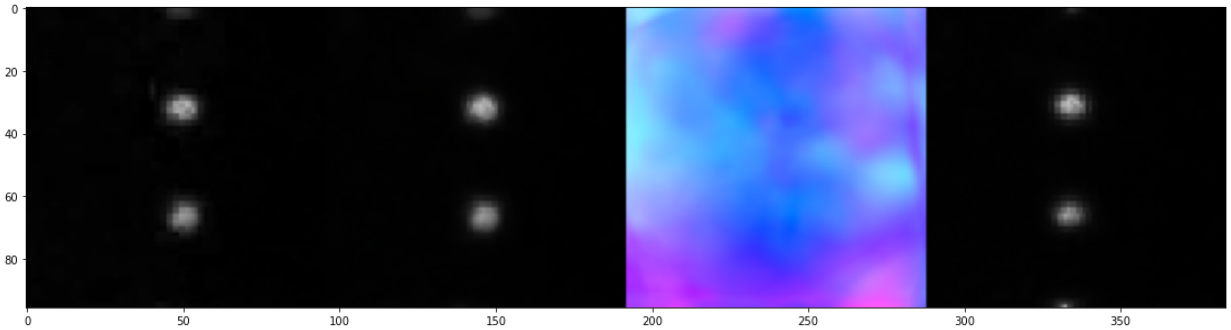
Epoch 600, loss = 0.007, psnr_sharp = 25.939, psnr_turb = 41.535



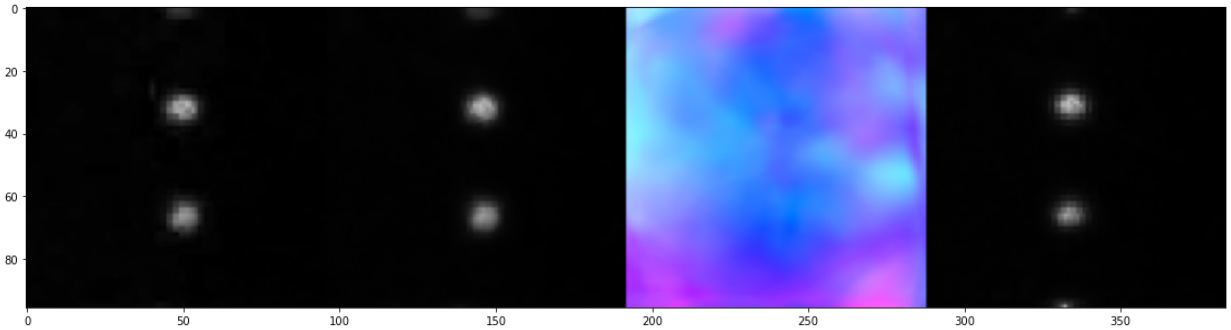
Epoch 700, loss = 0.007, psnr_sharp = 25.946, psnr_turb = 41.693



Epoch 800, loss = 0.007, psnr_sharp = 25.948, psnr_turb = 41.803

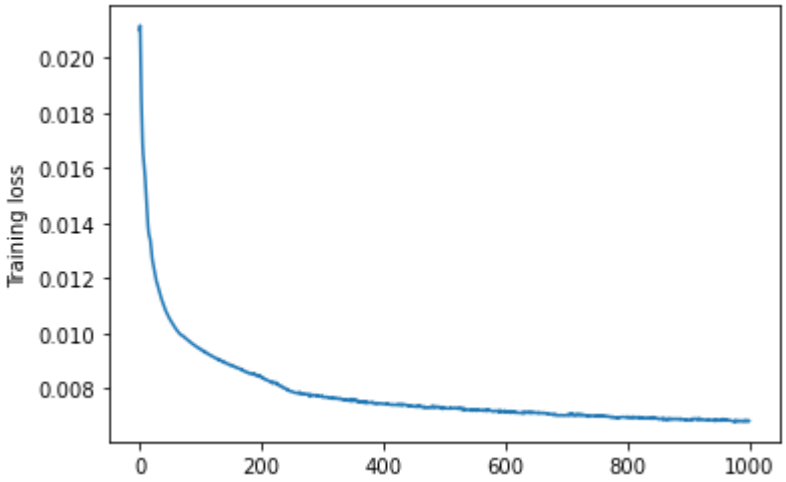


Epoch 900, loss = 0.007, psnr_sharp = 25.951, psnr_turb = 41.849



In [ ]:
```
print(float(loss))
loss_arr_np = loss_arr.detach().cpu().numpy()
plt.plot(loss_arr_np[:])
plt.ylabel('Training loss')
plt.show()
```

0.006831178907305002



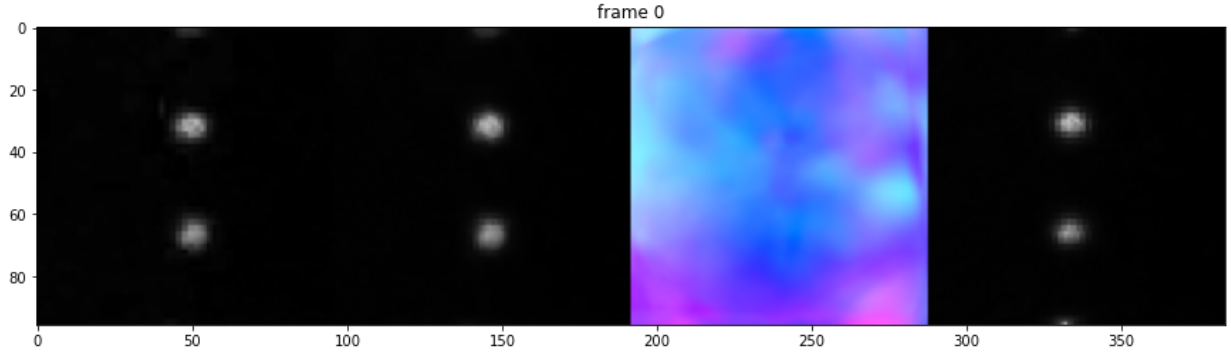
In [ ]:
```
torch.save(model_grid, '{}/{}_gd_final_{}_{}.pth'.format(fresult,pname,FB_img,
torch.save(model_imgen, '{}/{}_ig_final_{}_{}.pth'.format(fresult,pname,FB_img
torch.save(loss_arr, '{}/{}_loss_final_{}_{}.pth'.format(fresult,pname,FB_img,
torch.save(psnr_arr_sharp, '{}/{}_psnr_sharp_final_{}_{}.pth'.format(fresult,p
torch.save(psnr_arr_turb, '{}/{}_psnr_turb_final_{}_{}.pth'.format(fresult,pna
torch.save(ssim_arr_sharp, '{}/{}_ssim_sharp_final_{}_{}.pth'.format(fresult,p
torch.save(ssim_arr_turb, '{}/{}_ssim_turb_final_{}_{}.pth'.format(fresult,pna
```

Test

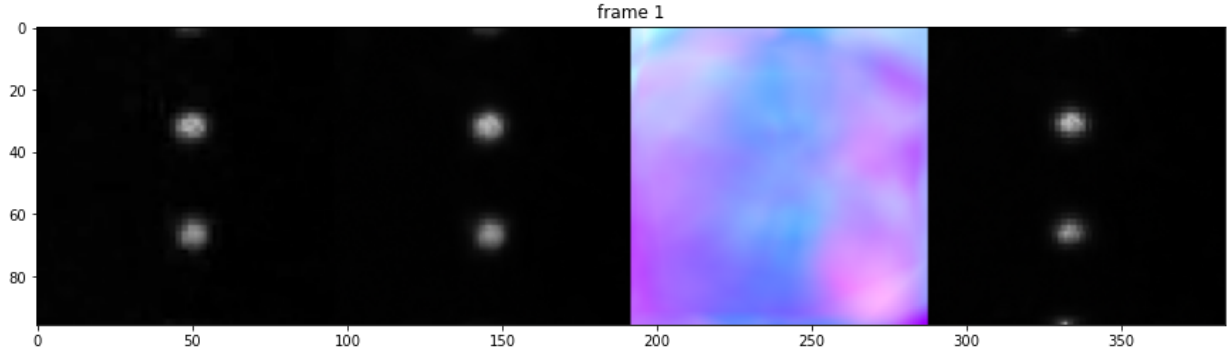
In [ ]:
```
# model_grid = torch.load('{}/{}_gd_final_{}_{}.pth'.format(fresult,pname,FB_i
# model_imgen = torch.load('{}/{}_ig_final_{}_{}.pth'.format(fresult,pname,FB_

fresult = '{}/{}/'.format(fresult,batch_size)
if not os.path.exists(fresult):
    os.makedirs(fresult)

# -------------------------------SETUP Grid deformer-------------------------
refined_xy = []
for b in range(batch_size):
    refined_xy.append(model_grid[b](grid_input_single_gd))

refined_xy = vec_scale*torch.cat(refined_xy)
refined_warp = refined_xy - xy_grid_batch_var
refined_uv = torch.cat([(nc - 1.0)*refined_warp[:, 0:1, :, :]/2 , (nr - 1.0)*r

# refined_uv -= refined_uv.min()
# refined_uv /= refined_uv.max()

generated_turb_imgs = model_imgen(GaussianFourierFeatureTransform_B(2, B_var,

sharp_imgs_predict = model_imgen(grid_input)

psnr_arr= compare_psnr(img_gt_np, sharp_imgs_predict[0].detach().cpu().numpy()
ssim_arr = ssim(img_gt_np.transpose(1,2,0), sharp_imgs_predict[0].detach().cpu

print('PSNR: {}, SSIM: {}'.format(psnr_arr,ssim_arr))
for j in range(batch_size):

    out_img = generated_turb_imgs[j].detach().cpu().numpy().transpose(1,2,0)
    sharp_img = sharp_imgs_predict[j].detach().cpu().numpy().transpose(1,2,0)
```

```
    warp_img= refined_uv[j].detach().cpu().numpy().transpose(1,2,0)
    warp_img -= np.min(warp_img)
    warp_img /= np.max(warp_img)

    out_target = img_gt_batch_var[j].detach().cpu().numpy().transpose(1,2,0)
    out_imshow = np.concatenate([out_target,out_img,visualize_rgb_norm(warp_im
    plt.figure(figsize=(15,5))
    plt.title('frame {}'.format(j))
    plt.imshow(out_imshow)
    plt.show()

    imsave('{}/{}_turb_img_gt_{}_FB_{}.png'.format(fresult,pname,j+start_f,FB_
    imsave('{}/{}_turb_img_{}_FB_{}.png'.format(fresult,pname,j+start_f,FB_img
    imsave('{}/{}_sharp_img_{}_FB_{}.png'.format(fresult,pname,j+start_f,FB_im
    imsave('{}/{}_warp_img_{}_FB_{}.png'.format(fresult,pname,j+start_f,FB_img
```

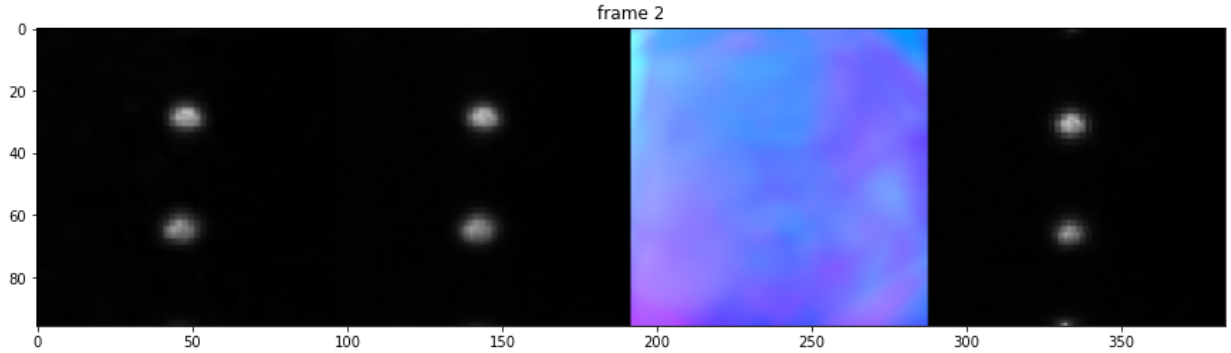
```
PSNR: 25.94705367971506, SSIM: 0.8911992907524109
```



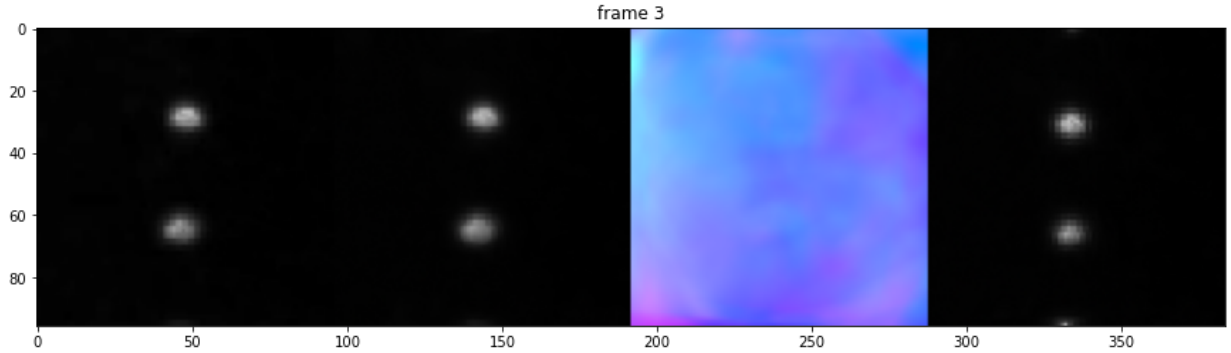

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



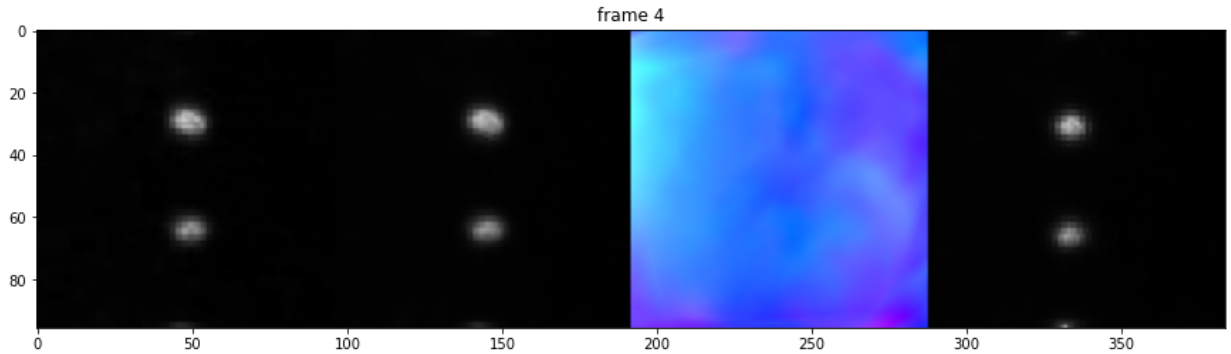

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



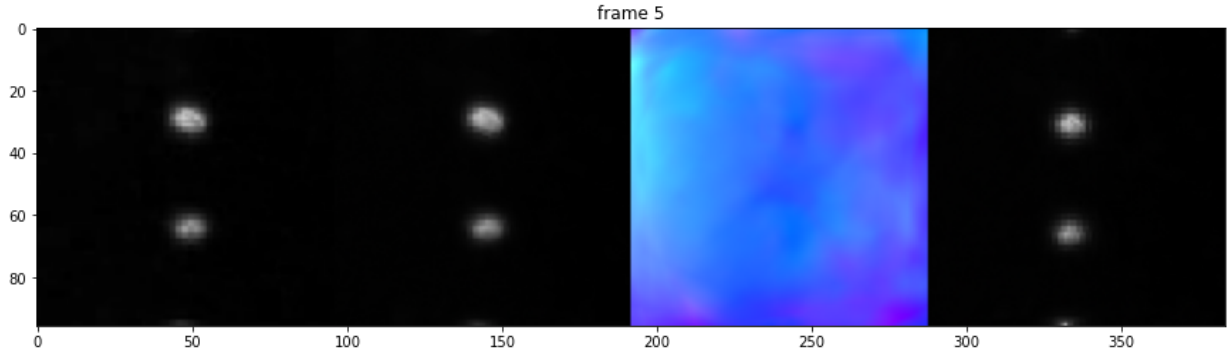

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



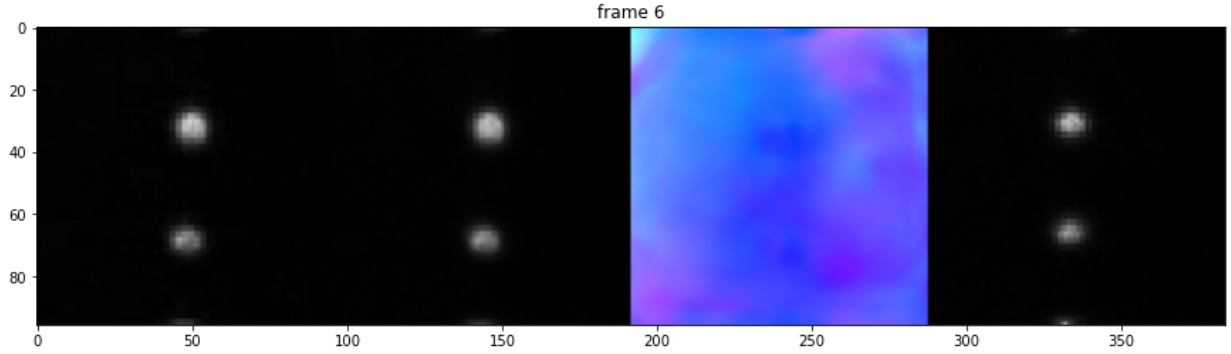

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



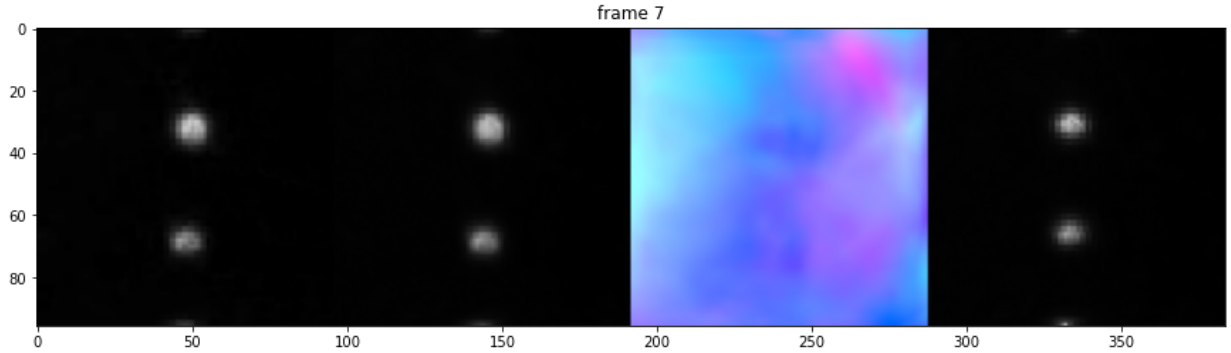

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



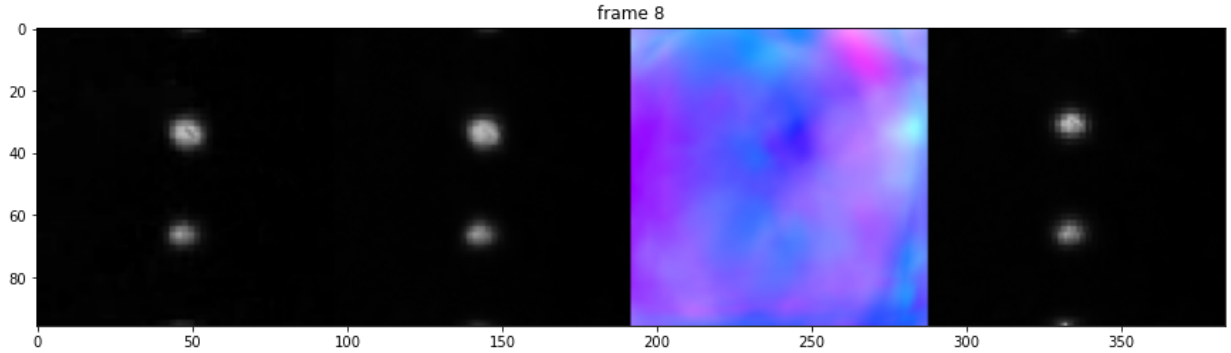

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



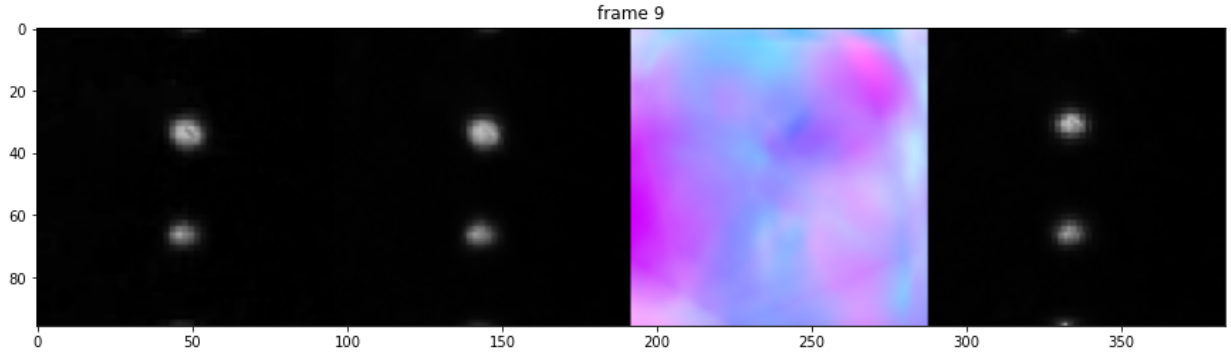

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



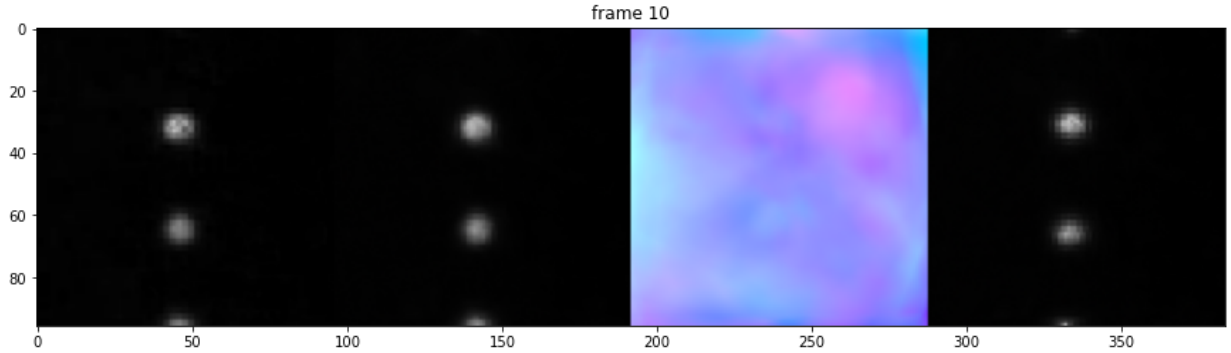

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



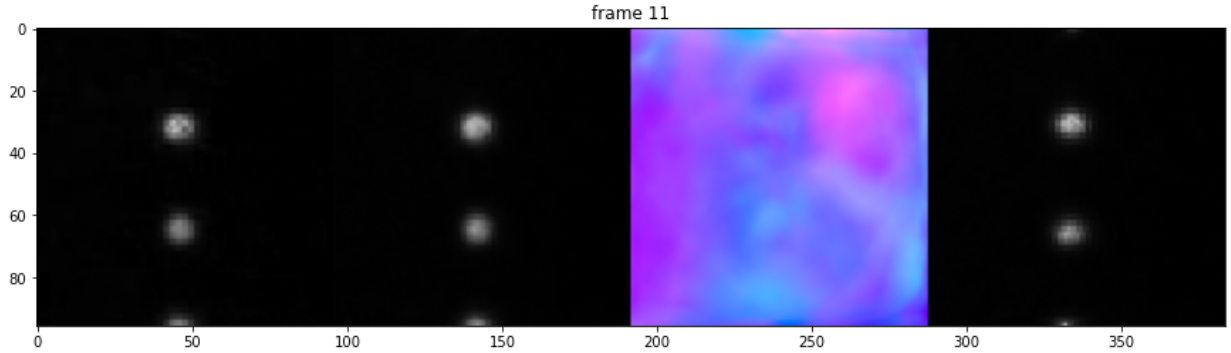

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



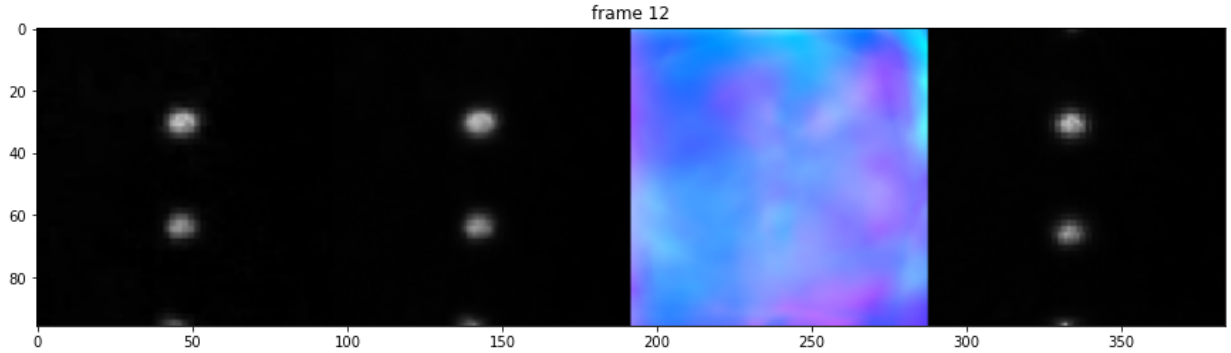

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



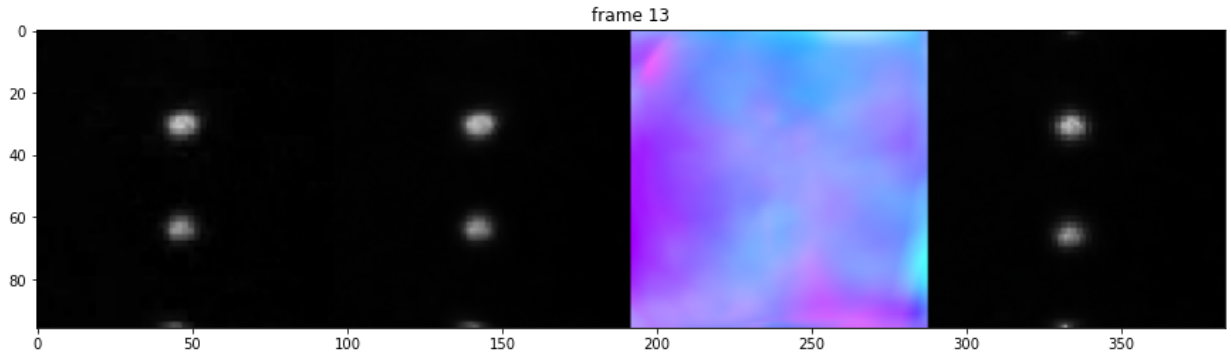

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



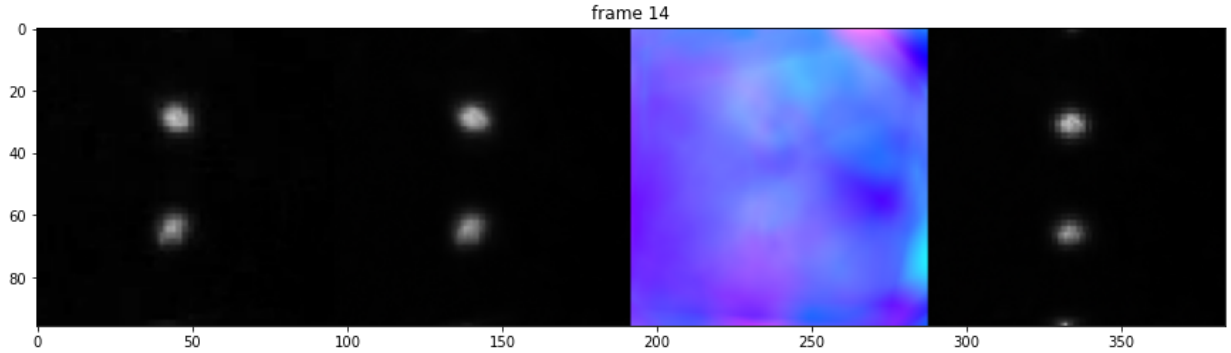

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



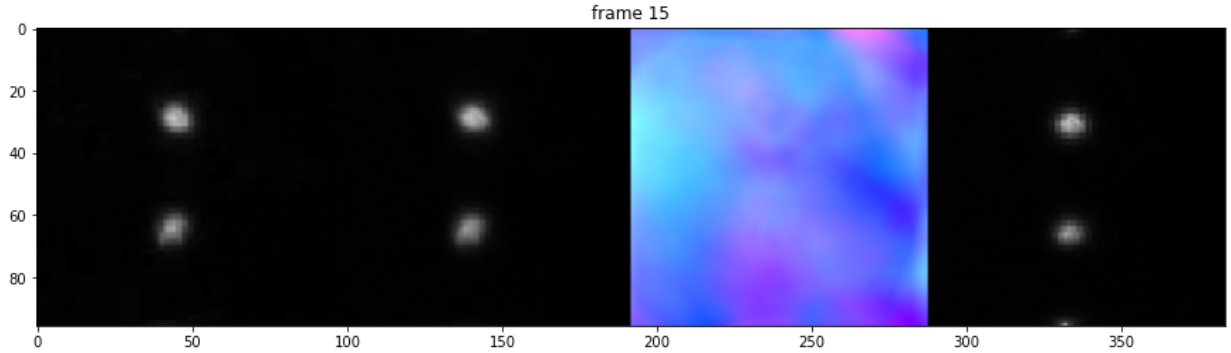

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



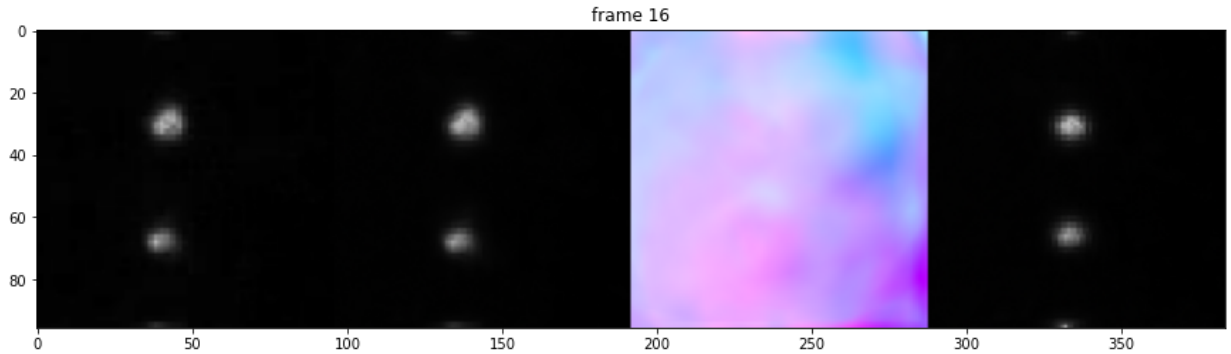

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



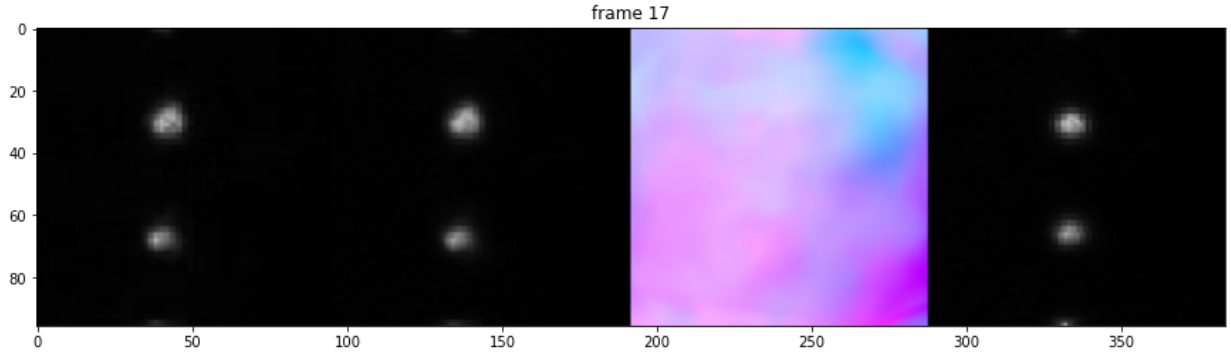

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



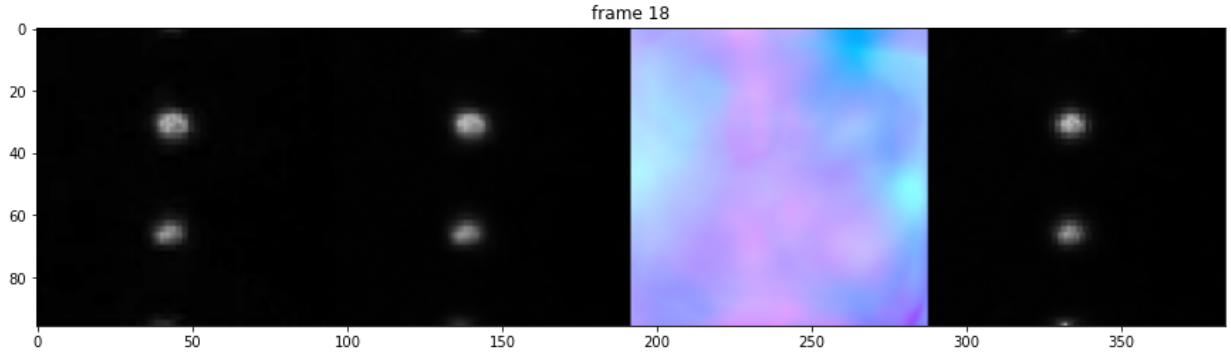

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



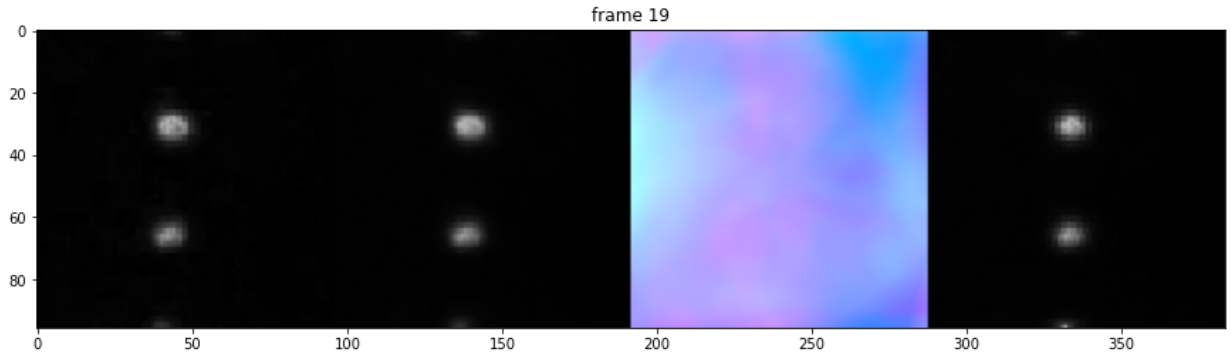

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



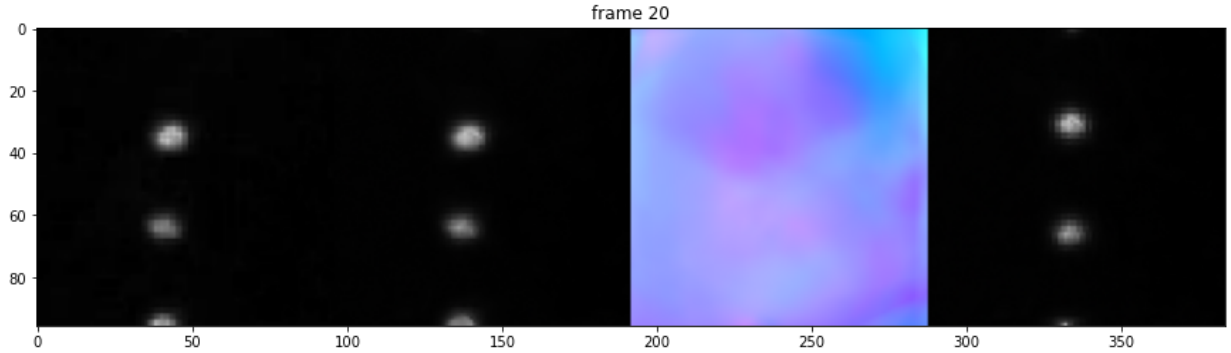

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



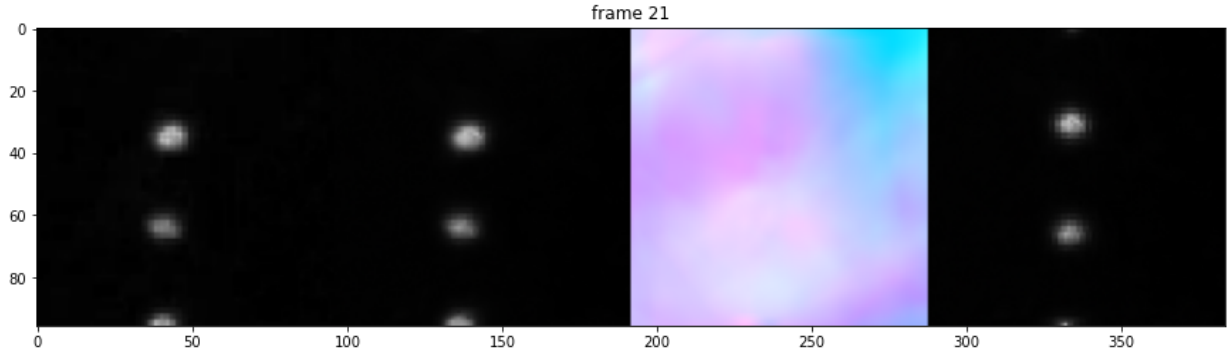

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



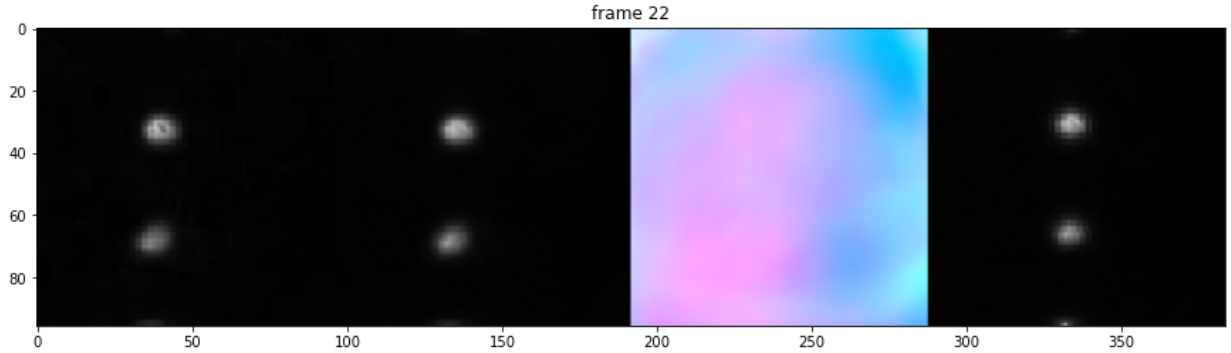

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



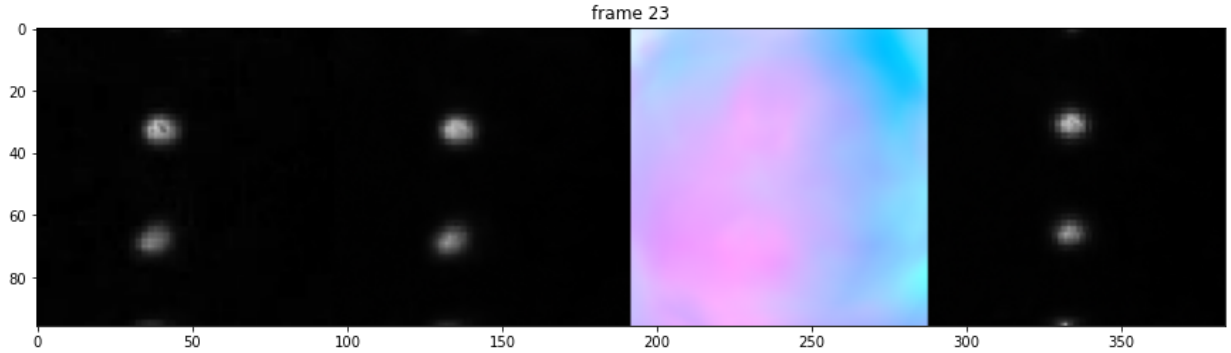

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



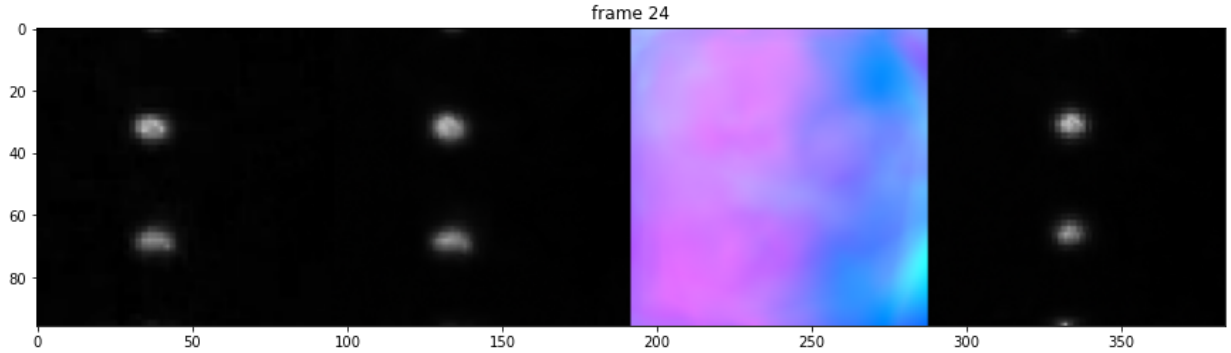

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```




```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```



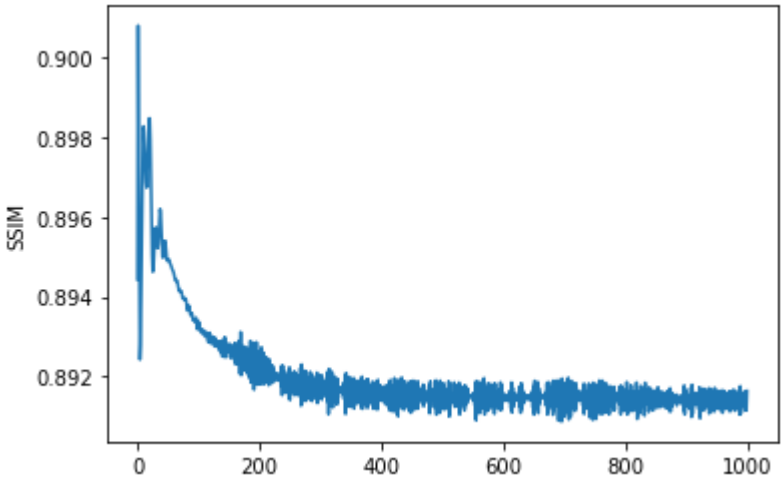

```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```




```
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
Lossy conversion from float64 to uint8. Range [0, 1]. Convert image to uint8 p
rior to saving to suppress this warning.
```

In [ ]:

```
# print(float(loss))
ssim_arr_sharp_np = ssim_arr_sharp.detach().cpu().numpy()
plt.plot(ssim_arr_sharp_np[:])
plt.ylabel('SSIM')
plt.show()
```



In [ ]:
```
psnr_arr_sharp_np = psnr_arr_sharp.detach().cpu().numpy()
plt.plot(psnr_arr_sharp_np[:])
plt.ylabel('PSNR')
plt.show()
```