FinalProject

May 11, 2022

0.0.1 Note to reader

This notebook has been set up s.t. the activation functions and loss weight combination experiments can be performed.

0.0.2 Import modules + Check GPU

[23]:
```
import torch
import torchvision
from torch import nn
import torch.nn.functional as F
from torch.utils.data import DataLoader, Dataset
import os
from PIL import Image
from torchvision.io import read_image
from torchvision.transforms import Resize, Compose, ToTensor, Normalize
import numpy as np
import skimage
import matplotlib.pyplot as plt
import time
from skimage import io
from tqdm import tqdm
import scipy.ndimage
from torch.utils.tensorboard import SummaryWriter
from pathlib import Path

# for SSIM
import math

# for beta selection
import random
```

[4]:
```
print("GPU available: {}".format(torch.cuda.is_available()))
print("Device: {}".format(torch.cuda.get_device_name(0)))
```

```
GPU available: True
Device: Tesla K80
```

0.0.3 Data Generation

[5]:
```
def isotropic_diffusion(img, niter=1, kappa=50, gamma=0.1, voxelspacing=None):

    # initialize output array
    out = np.array(img, dtype=np.float32, copy=True)

    # set default voxel spacing if not supplied
    if voxelspacing is None:
        voxelspacing = tuple([1.] * img.ndim)

    # initialize some internal variables
    deltas = [np.zeros_like(out) for _ in range(out.ndim)]


    time = 0

    results_pixels = []
    results_dIdt = []
    results_time = []

    results_pixels.append(out.astype(img.dtype))
    results_time.append(time)
    #results_dIdt.append(np.zeros_like(out))

    for iter in tqdm(range(niter)):
        # calculate the diffs
        for i in range(out.ndim):
            slicer = [slice(None, -1) if j == i else slice(None) for j in␣
↪range(out.ndim)]
            diff_local = np.diff(out, axis=i)
            deltas[i][tuple(slicer)] = diff_local

        matrices = [delta for delta, spacing in zip(deltas, voxelspacing)]

        # second derivative
        for i in range(out.ndim):
            slicer = [slice(1, None) if j == i else slice(None) for j in␣
↪range(out.ndim)]
            matrices[i][tuple(slicer)] = np.diff(matrices[i], axis=i)


        dIdt = np.sum(matrices, axis=0)
        #print(dIdt)

        # update the image
        out += gamma * (dIdt)
        time += gamma

        results_dIdt.append(dIdt.astype(img.dtype))
        if iter < niter - 1:
            results_pixels.append(out.astype(img.dtype))
            results_time.append(time)

    return results_pixels, results_dIdt, results_time

def get_mgrid(sidelen=256, dim=2):

    '''Generates a flattened grid of (x,y,...) coordinates in a range of -1 to␣
 ↪1.
    sidelen: int
    dim: int'''

    tensors = tuple(dim * [torch.linspace(-1, 1, steps=sidelen)])
    mgrid = torch.stack(torch.meshgrid(*tensors), dim=-1)
    mgrid = mgrid.reshape(-1, dim)

    return mgrid

class ImageFitting(Dataset):

    def __init__(self, img_path, niter):

        self.transform = Compose([
            Resize(256),
            ToTensor(),
            Normalize(torch.Tensor([0.5]), torch.Tensor([0.5]))
        ])
        self.coords = get_mgrid()

        print("-----Generating Data-----")
        self.base_img = io.imread(img_path)
        self.imgs_pixels, self.imgs_dIdt, self.imgs_time =␣
 ↪isotropic_diffusion(self.base_img, niter=niter, kappa=50, gamma=1/(niter+1))

        print("-----Finished-----")

        self.len = len(self.imgs_pixels)

    def __len__(self):

        return self.len

    def __getitem__(self, idx):
        image = self.imgs_pixels[idx]
        image = self.transform(Image.fromarray(image))

        pixels = image.permute(1, 2, 0).view(-1, 1)
        step_val = torch.full((self.coords.size(0),1), self.imgs_time[idx])

        model_input = torch.cat((self.coords, step_val), 1)

        # Compute gradient and laplacian
        grads_x = scipy.ndimage.sobel(image.numpy(), axis=1).squeeze(0)[...,␣
↪None]
        grads_y = scipy.ndimage.sobel(image.numpy(), axis=2).squeeze(0)[...,␣
↪None]
        grads_x, grads_y = torch.from_numpy(grads_x), torch.from_numpy(grads_y)

        grads = torch.stack((grads_x, grads_y), dim=-1).view(-1, 2)
        laplace = scipy.ndimage.laplace(image.numpy()).squeeze(0)[..., None]
        laplace = torch.from_numpy(laplace).view(-1, 1)

        dIdt = torch.from_numpy(self.imgs_dIdt[idx])
        dIdt = dIdt.permute(0,1).view(-1)

        return model_input, {'pixels':pixels, 'grads':grads, 'laplace':laplace,␣
↪'dIdt':dIdt}
```

0.0.4 Loss Calculation

[6]:
```
def computeJacobianFull(x, outputs, create_graph):

    dy_dx = torch.autograd.grad(outputs=outputs, inputs=x, grad_outputs=torch.
↪ones_like(outputs),
            retain_graph=True, create_graph=create_graph, allow_unused=True)[0]

    dy_dx = dy_dx.view(outputs.size(0), outputs.size(1), dy_dx.size(2))

    return dy_dx

def computeLaplaceFull(x, jacobian, create_graph):

    div = 0
    for j in range(jacobian.size(-1)):

        dy_dx2 = torch.autograd.grad(outputs=jacobian[:, :, j], inputs=x,␣
↪grad_outputs=torch.ones_like(jacobian[:, :, j]),
            retain_graph=True, create_graph=create_graph)[0][..., j:j+1]
        div += dy_dx2

    return div

def calcLoss(coords, model_output, gt):

    pixel_loss = ((model_output - gt['pixels'])**2).mean()

    gradients = computeJacobianFull(coords, model_output, create_graph=True)
    grad_loss = ((gradients[:,:,:-1] - gt['grads']).pow(2).sum(-1)).mean()

    laplacian = computeLaplaceFull(coords, gradients[:,:,:-1],␣
↪create_graph=False)
    laplacian_loss = ((laplacian - gt['laplace'])**2).mean()

    dIdt_loss = ((gradients[:,:,-1] - gt['dIdt'])**2).mean()

    pixel_ssim = mean_ssim(gt['pixels'][0].cpu().view(1, 256,256).detach(),␣
↪model_output[0].cpu().view(1, 256,256).detach(), val_range=255)

    grad_ssim = mean_ssim(gt['grads'][0].norm(dim=-1).cpu().view(1, 256,256).
↪detach(), gradients[0][:,:-1].norm(dim=-1).cpu().view(1, 256,256).detach(),␣
↪val_range=255)

    laplacian_ssim = mean_ssim(gt['laplace'][0].cpu().view(1, 256,256).
↪detach(), laplacian[0].cpu().view(1, 256,256).detach(), val_range=255)

    dIdt_ssim = mean_ssim(gt['dIdt'][0].cpu().view(1, 256,256).float().
↪detach(), gradients[0][:,-1].cpu().view(1, 256,256).detach(), val_range=255)

    return pixel_loss, grad_loss, laplacian_loss, dIdt_loss, pixel_ssim,␣
↪grad_ssim, laplacian_ssim, dIdt_ssim
```

0.0.5 SSIM (Structural Similarity Index Measure)

original SSIM paper: https://www.cns.nyu.edu/pub/eero/wang03-reprint.pdf
code source: https://github.com/pranjaldatta/SSIM-PyTorch
explanation: https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e

[7]:
```
def gaussian(window_size=11, sigma=1.5):
    """
    Generates a list of Tensor values drawn from a gaussian distribution with␣
↪standard
    diviation = sigma and sum of all elements = 1.

    Length of list = window_size
    """
    gauss =  torch.Tensor([math.exp(-(x - window_size//2)**2/float(2*sigma**2))␣
↪for x in range(window_size)])
    return gauss/gauss.sum()
```

[8]:
```
def create_window(window_size=11, channel=1):

    # Generate an 1D tensor containing values sampled from a gaussian␣
↪distribution
    _1d_window = gaussian(window_size=window_size, sigma=1.5).unsqueeze(1)

    # Converting to 2D
    _2d_window = _1d_window.mm(_1d_window.t()).float().unsqueeze(0).unsqueeze(0)

    window = torch.Tensor(_2d_window.expand(channel, 1, window_size,␣
↪window_size).contiguous())

    return window
```

[9]:
```
# mean SSIM with SSIM applied locally over moving windows
# output = 1: the same image, output = 0 (or -1): very different
def mean_ssim(img1, img2, val_range, window_size=11, window=None,␣
↪size_average=True, full=False):

    L = val_range # L is the dynamic range of the pixel values (255 for 8-bit␣
↪grayscale images),

    pad = window_size // 2

    try:
        _, channels, height, width = img1.size()
    except:
        channels, height, width = img1.size()

    # if window is not provided, init one
    if window is None:
        real_size = min(window_size, height, width) # window should be atleast␣
↪11x11
        window = create_window(real_size, channel=channels).to(img1.device)

    # calculating the mu parameter (locally) for both images using a gaussian␣
↪filter
    # calculates the luminosity params
    mu1 = F.conv2d(img1, window, padding=pad, groups=channels)
    mu2 = F.conv2d(img2, window, padding=pad, groups=channels)
    mu1_sq = mu1 ** 2
    mu2_sq = mu2 ** 2
    mu12 = mu1 * mu2

    # now we calculate the sigma square parameter
    # Sigma deals with the contrast component
    sigma1_sq = F.conv2d(img1 * img1, window, padding=pad, groups=channels) -␣
↪mu1_sq
    sigma2_sq = F.conv2d(img2 * img2, window, padding=pad, groups=channels) -␣
↪mu2_sq
    sigma12 =  F.conv2d(img1 * img2, window, padding=pad, groups=channels) -␣
↪mu12

    # Some constants for stability
    C1 = (0.01 ) ** 2  # NOTE: Removed L from here (ref PT implementation)
    C2 = (0.03 ) ** 2

    contrast_metric = (2.0 * sigma12 + C2) / (sigma1_sq + sigma2_sq + C2)
    contrast_metric = torch.mean(contrast_metric)

    numerator1 = 2 * mu12 + C1
    numerator2 = 2 * sigma12 + C2
    denominator1 = mu1_sq + mu2_sq + C1
    denominator2 = sigma1_sq + sigma2_sq + C2

    ssim_score = (numerator1 * numerator2) / (denominator1 * denominator2)

    if size_average:
        ret = ssim_score.mean()
    else:
        ret = ssim_score.mean(1).mean(1).mean(1)

    if full:
        return ret, contrast_metric

    return ret
```

[10]:
```
# Helper functions to convert to Tensors
tensorify = lambda x: torch.Tensor(x.transpose((1, 0))).unsqueeze(0).float().
↪div(255.0)
```

[9]:
```
# ### Example Usage ###

# img_path_temp = 'original/cameraman.png'
# img1 = io.imread(img_path_temp)
# img2 = io.imread(img_path_temp)
# # Check SSIM score of True image vs False Image
# _img1 = tensorify(img1)
# _img2 = tensorify(img2)
# true_vs_false = mean_ssim(_img1, _img2, val_range=255)
# print("True vs False Image SSIM Score:", true_vs_false)
```

0.0.6 SIREN Network Architecture

[11]:
```
class SineLayer(nn.Module):
    # See paper sec. 3.2, final paragraph, and supplement Sec. 1.5 for␣
 ↪discussion of omega_0.

    # If is_first=True, omega_0 is a frequency factor which simply multiplies␣
 ↪the activations before the
    # nonlinearity. Different signals may require different omega_0 in the␣
 ↪first layer - this is a
    # hyperparameter.

    # If is_first=False, then the weights will be divided by omega_0 so as to␣
 ↪keep the magnitude of
    # activations constant, but boost gradients to the weight matrix (see␣
 ↪supplement Sec. 1.5)

    def __init__(self, in_features, out_features, bias=True,
                 is_first=False, omega_0=30):
        super().__init__()
        self.omega_0 = omega_0
        self.is_first = is_first

        self.in_features = in_features
        self.linear = nn.Linear(in_features, out_features, bias=bias)

        self.init_weights()

    def init_weights(self):
        with torch.no_grad():
            if self.is_first:
                self.linear.weight.uniform_(-1 / self.in_features,
                                             1 / self.in_features)
            else:
                self.linear.weight.uniform_(-np.sqrt(6 / self.in_features) /␣
 ↪self.omega_0,
                                             np.sqrt(6 / self.in_features) /␣
 ↪self.omega_0)

    def forward(self, input):
        return torch.sin(self.omega_0 * self.linear(input))

    def forward_with_intermediate(self, input):
        # For visualization of activation distributions
        intermediate = self.omega_0 * self.linear(input)
        return torch.sin(intermediate), intermediate


class Siren(nn.Module):
    def __init__(self, in_features, hidden_features, hidden_layers,␣
 ↪out_features, outermost_linear=False,
                 first_omega_0=30, hidden_omega_0=30.):
        super().__init__()

        self.net = []
        self.net.append(SineLayer(in_features, hidden_features,
                                  is_first=True, omega_0=first_omega_0))

        for i in range(hidden_layers):
            self.net.append(SineLayer(hidden_features, hidden_features,
                                      is_first=False, omega_0=hidden_omega_0))

        if outermost_linear:
            final_linear = nn.Linear(hidden_features, out_features)

            with torch.no_grad():
                final_linear.weight.uniform_(-np.sqrt(6 / hidden_features) /␣
 ↪hidden_omega_0,
                                              np.sqrt(6 / hidden_features) /␣
 ↪hidden_omega_0)

            self.net.append(final_linear)
        else:
            self.net.append(SineLayer(hidden_features, out_features,
                                      is_first=False, omega_0=hidden_omega_0))

        self.net = nn.Sequential(*self.net)

    def forward(self, coords):
        coords = coords.clone().detach().requires_grad_(True) # allows to take␣
 ↪derivative w.r.t. input
        output = self.net(coords)
        return output, coords

    def forward_with_activations(self, coords, retain_grad=False):
        '''Returns not only model output, but also intermediate activations.
        Only used for visualizing activations later!'''
        activations = OrderedDict()

        activation_count = 0
        x = coords.clone().detach().requires_grad_(True)
        activations['input'] = x
        for i, layer in enumerate(self.net):
            if isinstance(layer, SineLayer):
                x, intermed = layer.forward_with_intermediate(x)

                if retain_grad:
                    x.retain_grad()
                    intermed.retain_grad()

                activations['_'.join((str(layer.__class__), "%d" %␣
↪activation_count))] = intermed
                activation_count += 1
            else:
                x = layer(x)

                if retain_grad:
                    x.retain_grad()

            activations['_'.join((str(layer.__class__), "%d" %␣
↪activation_count))] = x
            activation_count += 1

        return activations
```

0.0.7 ELU Network Architecture

[12]:
```
class ELULayer(nn.Module):

    def __init__(self, in_features, out_features, bias=True):
        super().__init__()

        self.in_features = in_features
        self.linear = nn.Linear(in_features, out_features, bias=bias)

        self.init_weights()

    def init_weights(self):
        with torch.no_grad():
            nn.init.xavier_uniform_(self.linear.weight)

    def forward(self, input):
        return F.elu(self.linear(input))
    def forward_with_intermediate(self, input):
        # For visualization of activation distributions
        intermediate = self.linear(input)
        return F.elu(intermediate), intermediate


class Base(nn.Module):
    def __init__(self, in_features, hidden_features, hidden_layers,␣
↪out_features, outermost_linear=False,
                 first_omega_0=30, hidden_omega_0=30.):
        super().__init__()

        self.net = []
        self.net.append(ELULayer(in_features, hidden_features))

        for i in range(hidden_layers):
            self.net.append(ELULayer(hidden_features, hidden_features))

        if outermost_linear:
            final_linear = nn.Linear(hidden_features, out_features)

            with torch.no_grad():
                nn.init.xavier_uniform_(final_linear.weight)

            self.net.append(final_linear)
        else:
            self.net.append(ELULayer(hidden_features, out_features,
                                     is_first=False, omega_0=hidden_omega_0))

        self.net = nn.Sequential(*self.net)

    def forward(self, coords):
        coords = coords.clone().detach().requires_grad_(True) # allows to take␣
↪derivative w.r.t. input
        output = self.net(coords)
        return output, coords

    def forward_with_activations(self, coords, retain_grad=False):
        '''Returns not only model output, but also intermediate activations.
        Only used for visualizing activations later!'''
        activations = OrderedDict()

        activation_count = 0
        x = coords.clone().detach().requires_grad_(True)
        activations['input'] = x
        for i, layer in enumerate(self.net):
            if isinstance(layer, SineLayer):
                x, intermed = layer.forward_with_intermediate(x)

                if retain_grad:
                    x.retain_grad()
                    intermed.retain_grad()

                activations['_'.join((str(layer.__class__), "%d" %␣
↪activation_count))] = intermed
                activation_count += 1
            else:
                x = layer(x)

                if retain_grad:
                    x.retain_grad()

            activations['_'.join((str(layer.__class__), "%d" %␣
↪activation_count))] = x
            activation_count += 1

        return activations
```

0.0.8 Save video

[13]:
```
import shutil
import subprocess

def output_video(net, img_path, niter, vidName='video_name.mp4'):

    image = ImageFitting(img_path=img_path, niter=niter)
    dataloader = DataLoader(image, batch_size=1, pin_memory=True, num_workers=0)

    net.cuda()

    if os.path.exists("tmp"):
        shutil.rmtree("tmp")
    os.makedirs("tmp")

    for step, batch in tqdm(enumerate(dataloader)):

        model_input = batch[0].cuda()
        gt = {key: value.cuda() for key, value in batch[1].items()}

        model_output, coords = net(model_input)
        img_grad = computeJacobianFull(coords, model_output, create_graph=True)
        img_laplacian = computeLaplaceFull(coords, img_grad, create_graph=False)

        fig, axes = plt.subplots(2,4, figsize=(18,6))
        axes[0,0].imshow(gt['pixels'][0].cpu().view(256,256).detach().numpy())
        axes[0,1].imshow(gt['grads'][0].norm(dim=-1).cpu().view(256,256).
↪detach().numpy())
        axes[0,2].imshow(gt['laplace'][0].cpu().view(256,256).detach().numpy())
        axes[0,3].imshow(gt['dIdt'][0].cpu().view(256,256).detach().numpy())
        axes[1,0].imshow(model_output[0].cpu().view(256,256).detach().numpy())
        axes[1,1].imshow(img_grad[0][:,:-1].norm(dim=-1).cpu().view(256,256).
↪detach().numpy())
        axes[1,2].imshow(img_laplacian[0].cpu().view(256,256).detach().numpy())
        axes[1,3].imshow(img_grad[0][:,-1].cpu().view(256,256).detach().numpy())

        fig.savefig("tmp/file%02d.png" % step)


    subprocess.call([
        'ffmpeg', '-framerate', '2', '-i', 'tmp/file%02d.png', '-r', '30',␣
↪'-pix_fmt', 'yuv420p',
        vidName
    ])

    shutil.rmtree("tmp")
```

0.0.9 Train Network

[14]:
```
def train(net, writer, img_path, niter, total_epochs=50, lr=[1e-4], beta_0=1,␣
↪beta_1=1, beta_2=1, beta_3=1,
          cyclic=False, decay_exp=False, decay_multi=False):

    """Args:
        net: Network to Train
        writer: SummaryWriter for logging
        img_path: path to default state image
        niter: number of steps to apply diffusion (0 means only 1 image)
        total_epochs: number of epochs to train
        beta_0: constant for loss on pixel value
        beta_1: constant for loss on gradients
        beta_2: constant for loss on laplacian
        beta_3: constant for loss on pixel time derivative
        cyclic: CyclicLearning rate (allows better learning)"""


    image = ImageFitting(img_path=img_path, niter=niter)
    dataloader = DataLoader(image, batch_size=1, pin_memory=True, num_workers=0)

    net.cuda()

    epochs_til_summary = 10 #UPDATE ACCORDINGLY
    steps_til_summary = 5 #UPDATE ACCORDINGLY

    optim = torch.optim.Adam(lr=lr[0], params=net.parameters())

    if decay_multi:
        m = np.floor(total_epochs/4)
        scheduler = torch.optim.lr_scheduler.MultiStepLR(optim,␣
↪milestones=[m*1,m*2,m*3], gamma=0.1)

    if decay_exp:
        scheduler = torch.optim.lr_scheduler.ExponentialLR(optim, gamma=0.9)

    if cyclic:
        scheduler = torch.optim.lr_scheduler.CyclicLR(optim, base_lr=lr[1],␣
↪max_lr=lr[0], step_size_up=250, cycle_momentum=False)

    print("-----Begin Training-----")
    for epoch in range(1, total_epochs + 1):

        epoch_loss = 0.0
        epoch_pixel_loss = 0.0
        epoch_grad_loss = 0.0
        epoch_laplacian_loss = 0.0
        epoch_dIdt_loss = 0.0

        epoch_pixel_ssim = 0.0
        epoch_grad_ssim = 0.0
        epoch_laplacian_ssim = 0.0
        epoch_dIdt_ssim = 0.0

        for step, batch in tqdm(enumerate(dataloader)):

            model_input = batch[0].cuda()
            gt = {key: value.cuda() for key, value in batch[1].items()}

            model_output, coords = net(model_input)

            pixel_loss, grad_loss, laplacian_loss, dIdt_loss, pixel_ssim,␣
↪grad_ssim, laplacian_ssim, dIdt_ssim = calcLoss(coords, model_output, gt)

            loss = beta_0 * pixel_loss + beta_1 * grad_loss + beta_2 *␣
↪laplacian_loss + beta_3 * dIdt_loss

            epoch_loss += model_output.shape[0] * loss.item()
            epoch_pixel_loss += model_output.shape[0] * pixel_loss.item()
            epoch_grad_loss += model_output.shape[0] * grad_loss.item()
            epoch_laplacian_loss += model_output.shape[0] * laplacian_loss.
↪item()
            epoch_dIdt_loss += model_output.shape[0] * dIdt_loss.item()

            epoch_pixel_ssim += model_output.shape[0] * pixel_ssim.item()
            epoch_grad_ssim += model_output.shape[0] * grad_ssim.item()
            epoch_laplacian_ssim += model_output.shape[0] * laplacian_ssim.
↪item()
            epoch_dIdt_ssim += model_output.shape[0] * dIdt_ssim.item()

            if not epoch % epochs_til_summary and step % steps_til_summary ==␣
↪steps_til_summary - 1:

                pixel_output = model_output[0].view(1, -1, 256, 256)
                pixel_gt = gt['pixels'][0].view(1, -1, 256, 256)
                img_grid_pixel = torchvision.utils.make_grid(torch.
↪cat((pixel_gt, pixel_output), 0), 2)
                img_grid_pixel = img_grid_pixel * 0.5 + 0.5
                writer.add_image('pixels', img_grid_pixel, epoch *␣
↪len(dataloader) + step + 1)


                img_grad = computeJacobianFull(coords, model_output,␣
↪create_graph=True)
                grad_output = img_grad[0,:,:-1].norm(dim=-1).view(1, -1, 256,␣
↪256)
                grad_gt = gt['grads'][0].norm(dim=-1).view(1, -1, 256, 256)
                img_grid_grad = torchvision.utils.make_grid(torch.cat((grad_gt,␣
↪grad_output), 0), 2)
                writer.add_image('grads', img_grid_grad, epoch *␣
↪len(dataloader) + step + 1)


                img_laplacian = computeLaplaceFull(coords, img_grad,␣
↪create_graph=False)
                laplacian_output = img_laplacian[0].view(1, -1, 256, 256)
                laplacian_gt = gt['laplace'][0].view(1, -1, 256, 256)
                img_grid_laplacian = torchvision.utils.make_grid(torch.
↪cat((laplacian_gt, laplacian_output), 0), 2)
                writer.add_image('laplacians', img_grid_laplacian, epoch *␣
↪len(dataloader) + step + 1)

                dIdt_output = img_grad[0,:,-1].view(1, -1, 256, 256)
                dIdt_gt = gt['dIdt'][0].view(1, -1, 256, 256)
                img_grid_dIdt = torchvision.utils.make_grid(torch.cat((dIdt_gt,␣
↪dIdt_output), 0), 2)
```

```
                writer.add_image('dIdt', img_grid_dIdt, epoch * len(dataloader)␣
↪+ step + 1)

                # fig, axes = plt.subplots(2,4, figsize=(18,6))
                # axes[0,0].imshow(gt['pixels'][0].cpu().view(256,256).detach().
↪numpy())
                # axes[0,1].imshow(gt['grads'][0].norm(dim=-1).cpu().
↪view(256,256).detach().numpy())
                # axes[0,2].imshow(gt['laplace'][0].cpu().view(256,256).
↪detach().numpy())
                # axes[0,3].imshow(gt['dIdt'][0].cpu().view(256,256).detach().
↪numpy())
                # axes[1,0].imshow(model_output[0].cpu().view(256,256).detach().
↪numpy())
                # axes[1,1].imshow(img_grad[0][:,:-1].norm(dim=-1).cpu().
↪view(256,256).detach().numpy())
                # axes[1,2].imshow(img_laplacian[0].cpu().view(256,256).
↪detach().numpy())
                # axes[1,3].imshow(img_grad[0][:,-1].cpu().view(256,256).
↪detach().numpy())
                # plt.show()

            optim.zero_grad()
            loss.backward()
            optim.step()

            if cyclic or decay_exp or decay_multi:
                scheduler.step()

        # logging epoch loss
        writer.add_scalar('epoch_loss/total', epoch_loss/len(image), epoch)
        writer.add_scalar('epoch_loss/pixel', epoch_pixel_loss/len(image),␣
↪epoch)
        writer.add_scalar('epoch_loss/grad', epoch_grad_loss/len(image), epoch)
        writer.add_scalar('epoch_loss/laplacian', epoch_laplacian_loss/
↪len(image), epoch)
        writer.add_scalar('epoch_loss/dIdt', epoch_dIdt_loss/len(image), epoch)
        print("Epoch %d, Epoch loss: total %0.6f, pixel %0.6f, grad %0.6f,␣
↪laplacian %0.6f, dIdt %0.6f" % (epoch, epoch_loss/len(image),␣
↪epoch_pixel_loss/len(image), epoch_grad_loss/len(image),␣
↪epoch_laplacian_loss/len(image), epoch_dIdt_loss/len(image)))

        # logging ssim loss
        writer.add_scalar('epoch_ssim/pixel', epoch_pixel_ssim/len(image),␣
↪epoch)
        writer.add_scalar('epoch_ssim/grad', epoch_grad_ssim/len(image), epoch)
        writer.add_scalar('epoch_ssim/laplacian', epoch_laplacian_ssim/
↪len(image), epoch)
        writer.add_scalar('epoch_ssim/dIdt', epoch_dIdt_ssim/len(image), epoch)
        print("Epoch %d, Epoch SSIM: pixel %0.6f, grad %0.6f, laplacian %0.6f,␣
↪dIdt %0.6f" % (epoch, epoch_pixel_ssim/len(image), epoch_grad_ssim/
↪len(image), epoch_laplacian_ssim/len(image), epoch_dIdt_ssim/len(image)))

    writer.add_graph(net, model_input)
    print("-----Finished-----")
```

0.0.10 Baselines & Activation Experiments

SIREN Baselines

```
[ ]: torch.cuda.empty_cache()
```

```
[ ]: total_epochs = 20
```

```
[ ]: # SIREN, learn only with the observed pixel values
writer = SummaryWriter('runs/siren/cameraman_experiment_pixels')

img_siren = Siren(in_features=3, out_features=1, hidden_features=512,
                  hidden_layers=3, outermost_linear=True)

train(img_siren, writer, img_path='original/cameraman.png', niter=10,␣
 ↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=1, beta_1=0, beta_2=0, beta_3=0)

writer.close()

output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='videos/siren/cameraman_experiment_pixels'+ '_video.mp4')
```

```
[ ]: # SIREN, learn only with the observed jacobians (first derivative in space)
writer = SummaryWriter('runs/siren/cameraman_experiment_grads')

img_siren = Siren(in_features=3, out_features=1, hidden_features=512,
                  hidden_layers=3, outermost_linear=True)

train(img_siren, writer, img_path='original/cameraman.png', niter=1,␣
 ↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=0, beta_1=1, beta_2=0, beta_3=0)

writer.close()
```

```
output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
↪vidName='videos/siren/cameraman_experiment_grads'+ '_video.mp4')
```

```
[ ]: # SIREN, learns only with the observed laplacians (2nd derivative in space)
writer = SummaryWriter('runs/siren/cameraman_experiment_laplace')

img_siren = Siren(in_features=3, out_features=1, hidden_features=512,
                  hidden_layers=3, outermost_linear=True)

train(img_siren, writer, img_path='original/cameraman.png', niter=1,␣
↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=0, beta_1=0, beta_2=1, beta_3=0)

writer.close()

output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
↪vidName='videos/siren/cameraman_experiment_laplace'+ '_video.mp4')
```

```
[ ]: # SIREN, learn only with the observed derivative in time (3rd derivative)
writer = SummaryWriter('runs/siren/cameraman_experiment_dIdt')

img_siren = Siren(in_features=3, out_features=1, hidden_features=512,
                  hidden_layers=3, outermost_linear=True)

train(img_siren, writer, img_path='original/cameraman.png', niter=1,␣
↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=0, beta_1=0, beta_2=0, beta_3=1)

writer.close()

output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
↪vidName='videos/siren/cameraman_experiment_dIdt'+ '_video.mp4')
```

```
[ ]: # SIREN, learn with all data, equally weighted
writer = SummaryWriter('runs/siren/cameraman_experiment_all')

img_siren = Siren(in_features=3, out_features=1, hidden_features=512,
                  hidden_layers=3, outermost_linear=True)

train(img_siren, writer, img_path='original/cameraman.png', niter=1,␣
↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=1, beta_1=1, beta_2=1, beta_3=1)

writer.close()

output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
↪vidName='videos/siren/cameraman_experiment_all'+ '_video.mp4')
```

Elu Baselines

[ ]:
```
# Base, learn only with the observed pixel values
writer = SummaryWriter('runs/base/cameraman_experiment_pixels')

img_base = Base(in_features=3, out_features=1, hidden_features=512,
                      hidden_layers=3, outermost_linear=True)

train(img_base, writer, img_path='original/cameraman.png', niter=1,␣
 ↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=1, beta_1=0, beta_2=0, beta_3=0)

writer.close()
output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='videos/base/cameraman_experiment_pixels'+ '_video.mp4')
```

[ ]:
```
# Base, learn only with the observed jacobians (first derivative in space)
writer = SummaryWriter('runs/base/cameraman_experiment_grads')

img_base = Base(in_features=3, out_features=1, hidden_features=512,
                      hidden_layers=3, outermost_linear=True)

train(img_base, writer, img_path='original/cameraman.png', niter=1,␣
 ↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=0, beta_1=1, beta_2=0, beta_3=0)

writer.close()

output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='videos/base/cameraman_experiment_grads'+ '_video.mp4')
```

[ ]:
```
# Base, learn only with the observed laplacians (2nd derivative in space)
writer = SummaryWriter('runs/base/cameraman_experiment_laplace')

img_base = Base(in_features=3, out_features=1, hidden_features=512,
                      hidden_layers=3, outermost_linear=True)

train(img_base, writer, img_path='original/cameraman.png', niter=1,␣
 ↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=0, beta_1=0, beta_2=1, beta_3=0)

writer.close()
output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='videos/base/cameraman_experiment_laplace'+ '_video.mp4')
```

[ ]:
```
# Base, learn only with the observed derivative in time (3rd derivative)
writer = SummaryWriter('runs/base/cameraman_experiment_dIdt')
```

```
img_base = Base(in_features=3, out_features=1, hidden_features=512,
                      hidden_layers=3, outermost_linear=True)

train(img_base, writer, img_path='original/cameraman.png', niter=1,␣
 ↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=0, beta_1=0, beta_2=0, beta_3=1)

writer.close()

output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='videos/base/cameraman_experiment_dIdt'+ '_video.mp4')
```

[ ]:
```
# Base, learn with all data, equally weighted
writer = SummaryWriter('runs/base/cameraman_experiment_all')

img_base = Base(in_features=3, out_features=1, hidden_features=512,
                      hidden_layers=3, outermost_linear=True)

train(img_base, writer, img_path='original/cameraman.png', niter=1,␣
 ↪total_epochs=total_epochs, lr=[1e-4],
      beta_0=1, beta_1=1, beta_2=1, beta_3=1)

writer.close()

output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='videos/base/cameraman_experiment_all'+ '_video.mp4')
```

0.0.11 Experiments

Experiments Helper Functions

[ ]:
```
# @min_beta_sum: minimum sum of all four beta values
# @return: a list of 4 beta values, summing to at least min_beta_sum
def generate_random_beta_combos(min_beta_sum=0.1):
    possible_values = [1.0, 0.1, 0.01, 0.001, 0.0]
    betas = [0, 0, 0, 0]

    while np.sum(betas) <= min_beta_sum:
        betas = [random.choice(possible_values) for i in range(4)]

    return betas

# @betas: a list of beta values
# @return: a string with '_' between all beta values
def b_to_string(betas):
    return '_'.join(map(str, betas))
```

[ ]:
```
# runs one experiment with the elu activation function
def run_elu(model_path, betas, total_epochs, lr, cyclic=False, decay_exp=False,␣
 ↪decay_multi=False):
    writer = SummaryWriter(model_path)

    img_base = Base(in_features=3, out_features=1, hidden_features=512,
                    hidden_layers=3, outermost_linear=True)

    train(img_base, writer, img_path='original/cameraman.png', niter=10,
          total_epochs=total_epochs, lr=lr,
          beta_0=betas[0], beta_1=betas[1], beta_2=betas[2], beta_3=betas[3],
          cyclic=cyclic, decay_exp=decay_exp, decay_multi=decay_multi)

    writer.close()

    output_video(img_base, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='/home/jupyter/videos/' + model_path+ '_video.mp4')
```

[ ]:
```
# runs one experiment with the SIREN (periodic) activation function
def run_siren(model_path, betas, total_epochs, lr, cyclic=False,␣
 ↪decay_exp=False, decay_multi=False):
    writer = SummaryWriter(model_path)

    img_siren = Siren(in_features=3, out_features=1, hidden_features=512,
                      hidden_layers=3, outermost_linear=True)

    train(img_siren, writer, img_path='original/cameraman.png', niter=10,
          total_epochs=total_epochs, lr=lr,
          beta_0=betas[0], beta_1=betas[1], beta_2=betas[2], beta_3=betas[3],
          cyclic=cyclic, decay_exp=decay_exp, decay_multi=decay_multi)

    writer.close()

    output_video(img_siren, img_path='original/cameraman.png', niter=10,␣
 ↪vidName='/home/jupyter/videos/' + model_path+ '_video.mp4')
```

Run the experiments

[25]:
```
# settings
model_path = 'runs/cameraman/experiments'
total_epochs = 20 #100
#learning_rates = [1e-4, 1e-5, 1e-6, 1e-7]
learning_rates = [1e-4]

# keep conducting experiments until we've reached the desired amount
num_experiments = 0
while num_experiments < 100:
    torch.cuda.empty_cache()

    # get a random combination of betas
    betas = generate_random_beta_combos()
    model_path_b = model_path + "/" + b_to_string(betas)

    if Path(model_path_b).exists():
        continue

    os.mkdir('/home/jupyter/videos/' + model_path_b)

    ### Part A. Models with elu activation ###
#     model_path_act = model_path + '/elu_'

#     ## learning rate experiments ##

#     # 1. run with uniform learning rates
#     for uniform_lr in learning_rates:
#         model_path_full = model_path_act + 'uniformlr_' + "{:.0e}".
 ↪format(uniform_lr)
#         run_elu(model_path_full, betas, total_epochs, [uniform_lr])

#     # 2. run with decaying learning rates
#     initial_lr = learning_rates[0]

#     # 2.1 multi-step: decay_multi = True
#     model_path_full = model_path_act + '_decay_multi_' + "{:.0e}".
 ↪format(initial_lr)
#     run_elu(model_path_full, betas, total_epochs, [initial_lr],␣
 ↪decay_multi=True)

#     # 2.2 exponential: decay_exp = True
#     model_path_full = model_path_act + '_decay_exp_' + "{:.0e}".
 ↪format(initial_lr)
#     run_elu(model_path_full, betas, total_epochs, [initial_lr],␣
 ↪decay_exp=True)

#     # 3. run with cyclic learning rate
#     max_lr = learning_rates[0]
#     min_lr = learning_rates[-1]
#     model_path_full = model_path_act + '_cyclic_' + "{:.0e}".format(max_lr) +␣
 ↪"_" + "{:.0e}".format(min_lr)
#     run_elu(model_path_full, betas, total_epochs, [max_lr, min_lr],␣
 ↪cyclic=True)
```

```
    ### Part B. Models with SIREN (periodic) activation ###
    model_path_act = model_path_b + '/siren_'

    ## learning rate experiments ##

    # 1. run with uniform learning rates
    for uniform_lr in learning_rates:
        model_path_full = model_path_act + 'uniformlr_' + "{:.0e}".
 ↪format(uniform_lr)
        run_siren(model_path_full, betas, total_epochs, [uniform_lr])

    # 2. run with decaying learning rates
#     initial_lr = learning_rates[0]

#     # 2.1 multi-step: decay_multi = True
#     model_path_full = model_path_act + '_decay_multi_' + "{:.0e}".
 ↪format(initial_lr)
#     run_siren(model_path_full, betas, total_epochs, [initial_lr],␣
 ↪decay_multi=True)

#     # 2.2 exponential: decay_exp = True
#     model_path_full = model_path_act + '_decay_exp_' + "{:.0e}".
 ↪format(initial_lr)
#     run_siren(model_path_full, betas, total_epochs, [initial_lr],␣
 ↪decay_exp=True)

#     # 3. run with cyclic learning rate
#     max_lr = learning_rates[0]
#     min_lr = learning_rates[-1]
#     model_path_full = model_path_act + '_cyclic_' + "{:.0e}".format(max_lr) +␣
 ↪"_" + "{:.0e}".format(min_lr)
#     run_siren(model_path_full, betas, total_epochs, [max_lr, min_lr],␣
 ↪cyclic=True)

    # finished one more experiment
    num_experiments += 1
    print("finished experiment #", num_experiments)
```

```
-----Generating Data-----

100%|      | 10/10 [00:00<00:00, 1167.06it/s]

-----Finished-----
-----Begin Training-----

10it [00:25,  2.54s/it]

Epoch 1, Epoch loss: total 1792.176624, pixel 0.341292, grad 8.266688, laplacian
132.000001, dIdt 17916.950391
```

```
Epoch 1, Epoch SSIM: pixel 0.065567, grad 0.027473, laplacian 0.000044, dIdt
0.000925

10it [00:26,  2.60s/it]

Epoch 2, Epoch loss: total 1779.828564, pixel 0.397523, grad 10.770234,
laplacian 1250.852451, dIdt 17781.694043
Epoch 2, Epoch SSIM: pixel 0.002419, grad 0.012909, laplacian 0.000003, dIdt
0.000297

10it [00:25,  2.58s/it]

Epoch 3, Epoch loss: total 1968.496521, pixel 0.328358, grad 19.897638,
laplacian 170625.655823, dIdt 17975.225586
Epoch 3, Epoch SSIM: pixel 0.052166, grad 0.013906, laplacian 0.000006, dIdt
0.000214

10it [00:26,  2.63s/it]

Epoch 4, Epoch loss: total 2140.969055, pixel 0.339396, grad 24.207743,
laplacian 341306.188379, dIdt 17992.992383
Epoch 4, Epoch SSIM: pixel 0.022648, grad 0.010504, laplacian -0.000002, dIdt
0.000158

10it [00:25,  2.57s/it]

Epoch 5, Epoch loss: total 2082.272510, pixel 0.336864, grad 24.372157,
laplacian 285695.541211, dIdt 17962.156738
Epoch 5, Epoch SSIM: pixel -0.005028, grad 0.013174, laplacian 0.000001, dIdt
-0.000004

10it [00:26,  2.64s/it]

Epoch 6, Epoch loss: total 2030.863867, pixel 0.330846, grad 21.713527,
laplacian 238419.892285, dIdt 17920.913770
Epoch 6, Epoch SSIM: pixel 0.020352, grad 0.010625, laplacian 0.000001, dIdt
0.000051

10it [00:25,  2.55s/it]

Epoch 7, Epoch loss: total 2007.651086, pixel 0.335302, grad 22.325742,
laplacian 219646.560547, dIdt 17876.468848
Epoch 7, Epoch SSIM: pixel 0.038611, grad 0.011424, laplacian 0.000001, dIdt
0.000057

10it [00:25,  2.58s/it]

Epoch 8, Epoch loss: total 2051.837268, pixel 0.340840, grad 23.098742,
laplacian 264609.363672, dIdt 17868.639453
Epoch 8, Epoch SSIM: pixel 0.055094, grad 0.010403, laplacian 0.000001, dIdt
0.000050

10it [00:26,  2.64s/it]

Epoch 9, Epoch loss: total 2079.527124, pixel 0.345991, grad 27.621856,
laplacian 297686.437500, dIdt 17814.670801
```

```
Epoch 9, Epoch SSIM: pixel 0.068804, grad 0.006542, laplacian 0.000000, dIdt
-0.000033

10it [00:28,  2.84s/it]

Epoch 10, Epoch loss: total 2043.798621, pixel 0.377362, grad 32.715271,
laplacian 263866.433984, dIdt 17795.220508
Epoch 10, Epoch SSIM: pixel 0.088739, grad 0.006923, laplacian 0.000001, dIdt
-0.000082

10it [00:26,  2.62s/it]

Epoch 11, Epoch loss: total 2009.568262, pixel 0.392238, grad 34.392525,
laplacian 224966.730469, dIdt 17841.748730
Epoch 11, Epoch SSIM: pixel 0.074478, grad 0.007866, laplacian -0.000000, dIdt
-0.000023

10it [00:25,  2.54s/it]

Epoch 12, Epoch loss: total 2110.206665, pixel 0.402601, grad 34.824779,
laplacian 326729.953516, dIdt 17830.392871
Epoch 12, Epoch SSIM: pixel 0.070939, grad 0.005185, laplacian 0.000000, dIdt
-0.000122

10it [00:25,  2.59s/it]

Epoch 13, Epoch loss: total 2138.444836, pixel 0.428828, grad 35.287197,
laplacian 360075.812891, dIdt 17779.048828
Epoch 13, Epoch SSIM: pixel 0.108333, grad 0.006109, laplacian 0.000000, dIdt
-0.000063

10it [00:25,  2.54s/it]

Epoch 14, Epoch loss: total 2212.184265, pixel 0.497210, grad 52.824970,
laplacian 444346.478906, dIdt 17672.876563
Epoch 14, Epoch SSIM: pixel 0.093855, grad 0.005503, laplacian -0.000000, dIdt
-0.000104

10it [00:25,  2.56s/it]

Epoch 15, Epoch loss: total 2292.813879, pixel 0.589648, grad 58.364088,
laplacian 524507.364844, dIdt 17676.584180
Epoch 15, Epoch SSIM: pixel 0.107817, grad 0.006651, laplacian 0.000000, dIdt
-0.000115

10it [00:26,  2.65s/it]

Epoch 16, Epoch loss: total 2330.021143, pixel 0.584426, grad 66.697298,
laplacian 560413.314063, dIdt 17689.566211
Epoch 16, Epoch SSIM: pixel 0.103204, grad 0.005465, laplacian 0.000000, dIdt
-0.000078

10it [00:25,  2.53s/it]

Epoch 17, Epoch loss: total 2218.156934, pixel 0.763054, grad 64.608762,
laplacian 464656.876563, dIdt 17526.723535
```

```
Epoch 17, Epoch SSIM: pixel 0.106037, grad 0.005756, laplacian -0.000000, dIdt
0.000014

10it [00:26,  2.61s/it]

Epoch 18, Epoch loss: total 2318.970398, pixel 0.832172, grad 89.430962,
laplacian 560110.987500, dIdt 17579.377637
Epoch 18, Epoch SSIM: pixel 0.069979, grad 0.003898, laplacian 0.000000, dIdt
0.000029

10it [00:25,  2.55s/it]

Epoch 19, Epoch loss: total 2567.778101, pixel 0.859720, grad 104.167003,
laplacian 810264.751563, dIdt 17565.493945
Epoch 19, Epoch SSIM: pixel 0.085459, grad 0.005708, laplacian 0.000000, dIdt
-0.000080

10it [00:28,  2.87s/it]

Epoch 20, Epoch loss: total 2467.536633, pixel 0.888009, grad 109.146054,
laplacian 705958.093750, dIdt 17605.813672
Epoch 20, Epoch SSIM: pixel 0.074682, grad 0.002666, laplacian 0.000000, dIdt
-0.000151
-----Finished-----
-----Generating Data-----

100%|     | 10/10 [00:00<00:00, 1255.82it/s]

-----Finished-----

10it [00:15,  1.57s/it]
ffmpeg version 4.1.9-0+deb10u1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 8 (Debian 8.3.0-6)
  configuration: --prefix=/usr --extra-version=0+deb10u1 --toolchain=hardened
--libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
--arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-
filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-
libaom --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca
--enable-libcdio --enable-libcodec2 --enable-libflite --enable-libfontconfig
--enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm
--enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg
--enable-libopenmpt --enable-libopus --enable-libpulse --enable-librsvg
--enable-librubberband --enable-libshine --enable-libsnappy --enable-libsoxr
--enable-libspeex --enable-libssh --enable-libtheora --enable-libtwolame
--enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack
--enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-
libzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-openal --enable-
opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-libiec61883
--enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
  libavutil      56. 22.100 / 56. 22.100
  libavcodec     58. 35.100 / 58. 35.100
  libavformat    58. 20.100 / 58. 20.100
  libavdevice    58.  5.100 / 58.  5.100
  libavfilter     7. 40.101 /  7. 40.101
  libavresample   4.  0.  0 /  4.  0.  0
  libswscale      5.  3.100 /  5.  3.100
  libswresample   3.  3.100 /  3.  3.100
  libpostproc    55.  3.100 / 55.  3.100
Input #0, image2, from 'tmp/file%02d.png':
  Duration: 00:00:05.00, start: 0.000000, bitrate: N/A
    Stream #0:0: Video: png, rgba(pc), 1296x432 [SAR 2835:2835 DAR 3:1], 2 fps,
2 tbr, 2 tbn, 2 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (png (native) -> h264 (libx264))
Press [q] to stop, [?] for help
[libx264 @ 0x563c29a93f00] using SAR=1/1
[libx264 @ 0x563c29a93f00] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2
AVX FMA3 BMI2 AVX2
[libx264 @ 0x563c29a93f00] profile High, level 3.1
[libx264 @ 0x563c29a93f00] 264 - core 155 r2917 0a84d98 - H.264/MPEG-4 AVC codec
- Copyleft 2003-2018 - http://www.videolan.org/x264.html - options: cabac=1
ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00
mixed_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11
fast_pskip=1 chroma_qp_offset=-2 threads=3 lookahead_threads=1 sliced_threads=0
nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3
b_pyramid=2 b_adapt=1 b_bias=0 direct=1 weightb=1 open_gop=0 weightp=2
keyint=250 keyint_min=25 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf
mbtree=1 crf=23.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to '/home/jupyter/videos/runs/cameraman/experiments/1.0_0.001_0.
001_0.1/siren_uniformlr_1e-04_video.mp4':
  Metadata:
    encoder         : Lavf58.20.100
    Stream #0:0: Video: h264 (libx264) (avc1 / 0x31637661), yuv420p, 1296x432
[SAR 1:1 DAR 3:1], q=-1--1, 30 fps, 15360 tbn, 30 tbc
    Metadata:
      encoder         : Lavc58.35.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: -1
frame=  150 fps=101 q=-1.0 Lsize=     259kB time=00:00:04.90 bitrate=
433.6kbits/s dup=140 drop=0 speed= 3.3x
video:257kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing
overhead: 1.008557%
[libx264 @ 0x563c29a93f00] frame I:1     Avg QP:15.97  size: 51228
[libx264 @ 0x563c29a93f00] frame P:38    Avg QP:19.11  size:  5304
[libx264 @ 0x563c29a93f00] frame B:111   Avg QP:14.80  size:    85
[libx264 @ 0x563c29a93f00] consecutive B-frames:  1.3%  0.0%  0.0% 98.7%
[libx264 @ 0x563c29a93f00] mb I  I16..4: 51.8% 23.2% 25.0%
[libx264 @ 0x563c29a93f00] mb P  I16..4:  0.6%  2.0%  1.0%  P16..4:  3.8%  0.9%
1.0%  0.0%  0.0%    skip:90.6%
[libx264 @ 0x563c29a93f00] mb B  I16..4:  0.1%  0.0%  0.0%  B16..8:  2.9%  0.0%
```

```
0.0%  direct: 0.0%  skip:96.9%  L0:47.4% L1:52.5% BI: 0.1%
[libx264 @ 0x563c29a93f00] 8x8 transform intra:41.5% inter:60.4%
[libx264 @ 0x563c29a93f00] coded y,uvDC,uvAC intra: 56.7% 61.7% 58.2% inter:
0.8% 1.1% 0.6%
[libx264 @ 0x563c29a93f00] i16 v,h,dc,p: 68% 25%  7%  1%
[libx264 @ 0x563c29a93f00] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 17% 13% 20%  8%  7%
8%  8%  9% 12%
[libx264 @ 0x563c29a93f00] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 23% 21% 15%  7%  6%
7%  7%  7%  7%
[libx264 @ 0x563c29a93f00] i8c dc,h,v,p: 57% 19% 17%  7%
[libx264 @ 0x563c29a93f00] Weighted P-Frames: Y:0.0% UV:0.0%
[libx264 @ 0x563c29a93f00] ref P L0: 73.9% 18.9%  6.5%  0.7%
[libx264 @ 0x563c29a93f00] ref B L0: 69.1% 30.5%  0.4%
[libx264 @ 0x563c29a93f00] ref B L1: 97.1%  2.9%
[libx264 @ 0x563c29a93f00] kb/s:419.62

finished experiment # 1
-----Generating Data-----

100%|     | 10/10 [00:00<00:00, 1172.08it/s]

-----Finished-----
-----Begin Training-----

10it [00:25,  2.52s/it]

Epoch 1, Epoch loss: total 18.795853, pixel 0.293181, grad 7.466433, laplacian
113.264881, dIdt 17918.147949
Epoch 1, Epoch SSIM: pixel 0.024657, grad 0.034233, laplacian 0.000022, dIdt
0.001166

10it [00:26,  2.60s/it]

Epoch 2, Epoch loss: total 56.686100, pixel 0.270249, grad 6.993146, laplacian
496.902505, dIdt 17916.885645
Epoch 2, Epoch SSIM: pixel 0.099422, grad 0.036360, laplacian 0.000010, dIdt
0.002163

10it [00:25,  2.52s/it]

Epoch 3, Epoch loss: total 114.659120, pixel 0.272526, grad 7.131689, laplacian
1075.247021, dIdt 17926.952930
Epoch 3, Epoch SSIM: pixel 0.103362, grad 0.041289, laplacian 0.000006, dIdt
0.002324

10it [00:25,  2.59s/it]

Epoch 4, Epoch loss: total 66.207528, pixel 0.288026, grad 7.578935, laplacian
586.257114, dIdt 17923.577637
Epoch 4, Epoch SSIM: pixel 0.129646, grad 0.033112, laplacian 0.000006, dIdt
0.003137

10it [00:25,  2.55s/it]
```

```
Epoch 5, Epoch loss: total 99.082601, pixel 0.266842, grad 6.913469, laplacian
921.664624, dIdt 17924.849121
Epoch 5, Epoch SSIM: pixel 0.166034, grad 0.045877, laplacian 0.000004, dIdt
0.001998

10it [00:26,  2.61s/it]

Epoch 6, Epoch loss: total 151.001302, pixel 0.271800, grad 6.837004, laplacian
1441.615771, dIdt 17923.748633
Epoch 6, Epoch SSIM: pixel 0.127570, grad 0.048240, laplacian 0.000004, dIdt
0.003150

10it [00:25,  2.53s/it]

Epoch 7, Epoch loss: total 144.870363, pixel 0.272695, grad 7.092230, laplacian
1377.754041, dIdt 17922.755762
Epoch 7, Epoch SSIM: pixel 0.114187, grad 0.049100, laplacian 0.000004, dIdt
0.003397

10it [00:25,  2.55s/it]

Epoch 8, Epoch loss: total 144.143900, pixel 0.271950, grad 6.656942, laplacian
1374.842346, dIdt 17923.143945
Epoch 8, Epoch SSIM: pixel 0.124894, grad 0.063830, laplacian 0.000004, dIdt
0.003758

10it [00:25,  2.57s/it]

Epoch 9, Epoch loss: total 305.923668, pixel 0.250375, grad 6.159220, laplacian
2997.619458, dIdt 17924.524121
Epoch 9, Epoch SSIM: pixel 0.160711, grad 0.081737, laplacian 0.000003, dIdt
0.003438

10it [00:28,  2.81s/it]

Epoch 10, Epoch loss: total 229.865292, pixel 0.270687, grad 7.171700, laplacian
2226.908813, dIdt 17925.224414
Epoch 10, Epoch SSIM: pixel 0.130257, grad 0.055432, laplacian 0.000003, dIdt
0.003692

10it [00:26,  2.61s/it]

Epoch 11, Epoch loss: total 136.622093, pixel 0.272660, grad 6.719372, laplacian
1298.999915, dIdt 17924.213965
Epoch 11, Epoch SSIM: pixel 0.113547, grad 0.067140, laplacian 0.000002, dIdt
0.003150

10it [00:25,  2.55s/it]

Epoch 12, Epoch loss: total 382.101712, pixel 0.244614, grad 5.672535, laplacian
3764.267285, dIdt 17922.825879
Epoch 12, Epoch SSIM: pixel 0.198117, grad 0.097671, laplacian 0.000002, dIdt
0.001798

10it [00:25,  2.58s/it]
```

```
Epoch 13, Epoch loss: total 486.522043, pixel 0.250277, grad 6.514315, laplacian
4800.052246, dIdt 17922.826367
Epoch 13, Epoch SSIM: pixel 0.178635, grad 0.072523, laplacian 0.000001, dIdt
0.001957

10it [00:25,  2.52s/it]

Epoch 14, Epoch loss: total 202.811334, pixel 0.276339, grad 7.330801, laplacian
1954.777625, dIdt 17923.857422
Epoch 14, Epoch SSIM: pixel 0.096058, grad 0.052143, laplacian 0.000002, dIdt
0.002490

10it [00:25,  2.53s/it]

Epoch 15, Epoch loss: total 208.771985, pixel 0.265436, grad 6.119286, laplacian
2026.500403, dIdt 17921.657324
Epoch 15, Epoch SSIM: pixel 0.131113, grad 0.083950, laplacian 0.000002, dIdt
0.002041

10it [00:25,  2.58s/it]

Epoch 16, Epoch loss: total 672.771747, pixel 0.236812, grad 5.251298, laplacian
6675.180615, dIdt 17922.410742
Epoch 16, Epoch SSIM: pixel 0.192263, grad 0.121163, laplacian 0.000001, dIdt
0.001410

10it [00:25,  2.52s/it]

Epoch 17, Epoch loss: total 500.858636, pixel 0.254457, grad 6.946769, laplacian
4939.093286, dIdt 17919.806641
Epoch 17, Epoch SSIM: pixel 0.132466, grad 0.073124, laplacian 0.000001, dIdt
0.001824

10it [00:26,  2.62s/it]

Epoch 18, Epoch loss: total 224.621533, pixel 0.275744, grad 6.920702, laplacian
2176.980688, dIdt 17922.766504
Epoch 18, Epoch SSIM: pixel 0.095047, grad 0.065678, laplacian 0.000002, dIdt
0.002811

10it [00:25,  2.54s/it]

Epoch 19, Epoch loss: total 415.270639, pixel 0.259027, grad 5.761030, laplacian
4095.070068, dIdt 17920.496680
Epoch 19, Epoch SSIM: pixel 0.142183, grad 0.102761, laplacian 0.000001, dIdt
0.001561

10it [00:29,  2.90s/it]

Epoch 20, Epoch loss: total 873.464124, pixel 0.242809, grad 5.472133, laplacian
8679.895654, dIdt 17922.616504
Epoch 20, Epoch SSIM: pixel 0.179072, grad 0.112573, laplacian 0.000001, dIdt
0.001012
-----Finished-----
-----Generating Data-----
```

```
100%|     | 10/10 [00:00<00:00, 1203.84it/s]

-----Finished-----

10it [00:15,  1.58s/it]
ffmpeg version 4.1.9-0+deb10u1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 8 (Debian 8.3.0-6)
  configuration: --prefix=/usr --extra-version=0+deb10u1 --toolchain=hardened
--libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
--arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-
filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-
libaom --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca
--enable-libcdio --enable-libcodec2 --enable-libflite --enable-libfontconfig
--enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm
--enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg
--enable-libopenmpt --enable-libopus --enable-libpulse --enable-librsvg
--enable-librubberband --enable-libshine --enable-libsnappy --enable-libsoxr
--enable-libspeex --enable-libssh --enable-libtheora --enable-libtwolame
--enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack
--enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-
libzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-openal --enable-
opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-libiec61883
--enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
  libavutil      56. 22.100 / 56. 22.100
  libavcodec     58. 35.100 / 58. 35.100
  libavformat    58. 20.100 / 58. 20.100
  libavdevice    58.  5.100 / 58.  5.100
  libavfilter     7. 40.101 /  7. 40.101
  libavresample   4.  0.  0 /  4.  0.  0
  libswscale      5.  3.100 /  5.  3.100
  libswresample   3.  3.100 /  3.  3.100
  libpostproc    55.  3.100 / 55.  3.100
Input #0, image2, from 'tmp/file%02d.png':
  Duration: 00:00:05.00, start: 0.000000, bitrate: N/A
    Stream #0:0: Video: png, rgba(pc), 1296x432 [SAR 2835:2835 DAR 3:1], 2 fps,
2 tbr, 2 tbn, 2 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (png (native) -> h264 (libx264))
Press [q] to stop, [?] for help
[libx264 @ 0x5634a8239f00] using SAR=1/1
[libx264 @ 0x5634a8239f00] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2
AVX FMA3 BMI2 AVX2
[libx264 @ 0x5634a8239f00] profile High, level 3.1
[libx264 @ 0x5634a8239f00] 264 - core 155 r2917 0a84d98 - H.264/MPEG-4 AVC codec
- Copyleft 2003-2018 - http://www.videolan.org/x264.html - options: cabac=1
ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00
mixed_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11
fast_pskip=1 chroma_qp_offset=-2 threads=3 lookahead_threads=1 sliced_threads=0
nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3
```

```
b_pyramid=2 b_adapt=1 b_bias=0 direct=1 weightb=1 open_gop=0 weightp=2
keyint=250 keyint_min=25 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf
mbtree=1 crf=23.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to '/home/jupyter/videos/runs/cameraman/experiments/0.01_1.0_0.1
_0.0/siren_uniformlr_1e-04_video.mp4':
  Metadata:
    encoder         : Lavf58.20.100
    Stream #0:0: Video: h264 (libx264) (avc1 / 0x31637661), yuv420p, 1296x432
[SAR 1:1 DAR 3:1], q=-1--1, 30 fps, 15360 tbn, 30 tbc
    Metadata:
      encoder         : Lavc58.35.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: -1
frame=  150 fps=103 q=-1.0 Lsize=     250kB time=00:00:04.90 bitrate=
417.9kbits/s dup=140 drop=0 speed=3.37x
video:247kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing
overhead: 1.046852%
[libx264 @ 0x5634a8239f00] frame I:1     Avg QP:15.96  size: 50966
[libx264 @ 0x5634a8239f00] frame P:38    Avg QP:19.14  size:  5071
[libx264 @ 0x5634a8239f00] frame B:111   Avg QP:14.80  size:    81
[libx264 @ 0x5634a8239f00] consecutive B-frames:  1.3%  0.0%  0.0% 98.7%
[libx264 @ 0x5634a8239f00] mb I  I16..4: 52.2% 24.4% 23.5%
[libx264 @ 0x5634a8239f00] mb P  I16..4:  0.5%  1.5%  0.4%  P16..4:  4.2%  1.3%
1.3%  0.0%  0.0%    skip:90.7%
[libx264 @ 0x5634a8239f00] mb B  I16..4:  0.1%  0.0%  0.0%  B16..8:  2.7%  0.0%
0.0%  direct: 0.0%  skip:97.2%  L0:45.9% L1:54.1% BI: 0.1%
[libx264 @ 0x5634a8239f00] 8x8 transform intra:41.6% inter:65.7%
[libx264 @ 0x5634a8239f00] coded y,uvDC,uvAC intra: 50.5% 54.0% 50.6% inter:
1.0% 1.4% 0.8%
[libx264 @ 0x5634a8239f00] i16 v,h,dc,p: 69% 23%  8%  1%
[libx264 @ 0x5634a8239f00] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 18% 11% 23%  6%  8%
9%  7%  7%  9%
[libx264 @ 0x5634a8239f00] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 25% 24% 19%  5%  6%
6%  5%  5%  5%
[libx264 @ 0x5634a8239f00] i8c dc,h,v,p: 58% 19% 18%  6%
[libx264 @ 0x5634a8239f00] Weighted P-Frames: Y:0.0% UV:0.0%
[libx264 @ 0x5634a8239f00] ref P L0: 72.1% 21.6%  5.7%  0.6%
[libx264 @ 0x5634a8239f00] ref B L0: 72.7% 26.8%  0.5%
[libx264 @ 0x5634a8239f00] ref B L1: 97.6%  2.4%
[libx264 @ 0x5634a8239f00] kb/s:404.23

finished experiment # 2
-----Generating Data-----

100%|     | 10/10 [00:00<00:00, 1250.20it/s]

-----Finished-----
-----Begin Training-----

10it [00:25,  2.52s/it]
```

```
Epoch 1, Epoch loss: total 26.621803, pixel 0.293029, grad 7.418119, laplacian
124.365458, dIdt 17930.726074
Epoch 1, Epoch SSIM: pixel 0.144434, grad 0.034668, laplacian 0.000018, dIdt
0.001226

10it [00:25,  2.58s/it]

Epoch 2, Epoch loss: total 32.793078, pixel 0.270055, grad 6.800129, laplacian
803.625409, dIdt 17929.688965
Epoch 2, Epoch SSIM: pixel 0.171307, grad 0.047028, laplacian 0.000006, dIdt
0.002169

10it [00:25,  2.52s/it]

Epoch 3, Epoch loss: total 34.653031, pixel 0.267791, grad 7.240684, laplacian
946.015164, dIdt 17925.415527
Epoch 3, Epoch SSIM: pixel 0.132266, grad 0.037886, laplacian 0.000005, dIdt
0.002539

10it [00:26,  2.61s/it]

Epoch 4, Epoch loss: total 32.987681, pixel 0.282658, grad 7.398408, laplacian
763.746381, dIdt 17923.541895
Epoch 4, Epoch SSIM: pixel 0.074605, grad 0.036620, laplacian 0.000005, dIdt
0.002800

10it [00:25,  2.54s/it]

Epoch 5, Epoch loss: total 36.060338, pixel 0.269357, grad 6.874941, laplacian
1123.810162, dIdt 17920.359180
Epoch 5, Epoch SSIM: pixel 0.127593, grad 0.042870, laplacian 0.000003, dIdt
0.002469

10it [00:25,  2.56s/it]

Epoch 6, Epoch loss: total 43.156741, pixel 0.265568, grad 6.978813, laplacian
1823.233777, dIdt 17919.032813
Epoch 6, Epoch SSIM: pixel 0.131061, grad 0.048071, laplacian 0.000002, dIdt
0.003255

10it [00:25,  2.54s/it]

Epoch 7, Epoch loss: total 35.063009, pixel 0.278535, grad 7.116309, laplacian
1000.033496, dIdt 17918.510645
Epoch 7, Epoch SSIM: pixel 0.129885, grad 0.048056, laplacian 0.000004, dIdt
0.003179

10it [00:25,  2.53s/it]

Epoch 8, Epoch loss: total 43.848952, pixel 0.261201, grad 6.329889, laplacian
1957.696680, dIdt 17915.975293
Epoch 8, Epoch SSIM: pixel 0.171981, grad 0.073916, laplacian 0.000002, dIdt
0.002677

10it [00:26,  2.62s/it]
```

```
Epoch 9, Epoch loss: total 53.777078, pixel 0.261862, grad 6.611146, laplacian
2922.459351, dIdt 17915.150879
Epoch 9, Epoch SSIM: pixel 0.156788, grad 0.069717, laplacian 0.000002, dIdt
0.002660

10it [00:27,  2.80s/it]

Epoch 10, Epoch loss: total 42.246780, pixel 0.274739, grad 7.115699, laplacian
1718.996826, dIdt 17913.637695
Epoch 10, Epoch SSIM: pixel 0.087605, grad 0.057229, laplacian 0.000002, dIdt
0.003546

10it [00:25,  2.57s/it]

Epoch 11, Epoch loss: total 47.746038, pixel 0.260828, grad 6.159050, laplacian
2364.864758, dIdt 17912.257617
Epoch 11, Epoch SSIM: pixel 0.137628, grad 0.085086, laplacian 0.000002, dIdt
0.002645

10it [00:25,  2.52s/it]

Epoch 12, Epoch loss: total 72.796843, pixel 0.245811, grad 5.922017, laplacian
4894.022437, dIdt 17910.020410
Epoch 12, Epoch SSIM: pixel 0.164734, grad 0.095903, laplacian 0.000002, dIdt
0.002601

10it [00:26,  2.62s/it]

Epoch 13, Epoch loss: total 50.957817, pixel 0.272831, grad 7.106015, laplacian
2591.329468, dIdt 17911.224707
Epoch 13, Epoch SSIM: pixel 0.130631, grad 0.058398, laplacian 0.000002, dIdt
0.002785

10it [00:25,  2.51s/it]

Epoch 14, Epoch loss: total 50.526639, pixel 0.266455, grad 6.305798, laplacian
2628.300171, dIdt 17911.192090
Epoch 14, Epoch SSIM: pixel 0.155778, grad 0.079204, laplacian 0.000002, dIdt
0.002232

10it [00:25,  2.54s/it]

Epoch 15, Epoch loss: total 76.133932, pixel 0.250388, grad 5.790076, laplacian
5241.235937, dIdt 17906.457910
Epoch 15, Epoch SSIM: pixel 0.167398, grad 0.102950, laplacian 0.000001, dIdt
0.001513

10it [00:25,  2.57s/it]

Epoch 16, Epoch loss: total 65.769697, pixel 0.263554, grad 6.825569, laplacian
4101.187622, dIdt 17905.896484
Epoch 16, Epoch SSIM: pixel 0.152439, grad 0.064349, laplacian 0.000001, dIdt
0.002249

10it [00:25,  2.51s/it]
```

```
Epoch 17, Epoch loss: total 50.029660, pixel 0.268612, grad 6.552466, laplacian
2554.009302, dIdt 17910.239844
Epoch 17, Epoch SSIM: pixel 0.137277, grad 0.075862, laplacian 0.000001, dIdt
0.002381

10it [00:25,  2.58s/it]

Epoch 18, Epoch loss: total 84.075417, pixel 0.241623, grad 5.531142, laplacian
6061.698853, dIdt 17903.125000
Epoch 18, Epoch SSIM: pixel 0.208097, grad 0.115840, laplacian 0.000001, dIdt
0.001382

10it [00:25,  2.54s/it]

Epoch 19, Epoch loss: total 90.826101, pixel 0.249573, grad 6.195786, laplacian
6670.383716, dIdt 17901.524512
Epoch 19, Epoch SSIM: pixel 0.192020, grad 0.089923, laplacian 0.000001, dIdt
0.001848

10it [00:28,  2.86s/it]

Epoch 20, Epoch loss: total 56.914589, pixel 0.272576, grad 6.785977, laplacian
3219.546289, dIdt 17905.890723
Epoch 20, Epoch SSIM: pixel 0.133891, grad 0.067046, laplacian 0.000001, dIdt
0.002562
-----Finished-----
-----Generating Data-----

100%|      | 10/10 [00:00<00:00, 1287.90it/s]

-----Finished-----

0it [00:00, ?it/s]/opt/conda/lib/python3.7/site-
packages/ipykernel_launcher.py:24: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may
consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
10it [00:15,  1.58s/it]
ffmpeg version 4.1.9-0+deb10u1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 8 (Debian 8.3.0-6)
  configuration: --prefix=/usr --extra-version=0+deb10u1 --toolchain=hardened
--libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
--arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-
filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-
libaom --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca
--enable-libcdio --enable-libcodec2 --enable-libflite --enable-libfontconfig
--enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm
--enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg
--enable-libopenmpt --enable-libopus --enable-libpulse --enable-librsvg
--enable-librubberband --enable-libshine --enable-libsnappy --enable-libsoxr
--enable-libspeex --enable-libssh --enable-libtheora --enable-libtwolame
```

```
--enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack
--enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-
libzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-openal --enable-
opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-libiec61883
--enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
  libavutil      56. 22.100 / 56. 22.100
  libavcodec     58. 35.100 / 58. 35.100
  libavformat    58. 20.100 / 58. 20.100
  libavdevice    58.  5.100 / 58.  5.100
  libavfilter     7. 40.101 /  7. 40.101
  libavresample   4.  0.  0 /  4.  0.  0
  libswscale      5.  3.100 /  5.  3.100
  libswresample   3.  3.100 /  3.  3.100
  libpostproc    55.  3.100 / 55.  3.100
Input #0, image2, from 'tmp/file%02d.png':
  Duration: 00:00:05.00, start: 0.000000, bitrate: N/A
    Stream #0:0: Video: png, rgba(pc), 1296x432 [SAR 2835:2835 DAR 3:1], 2 fps,
2 tbr, 2 tbn, 2 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (png (native) -> h264 (libx264))
Press [q] to stop, [?] for help
[libx264 @ 0x561f8d9d5f00] using SAR=1/1
[libx264 @ 0x561f8d9d5f00] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2
AVX FMA3 BMI2 AVX2
[libx264 @ 0x561f8d9d5f00] profile High, level 3.1
[libx264 @ 0x561f8d9d5f00] 264 - core 155 r2917 0a84d98 - H.264/MPEG-4 AVC codec
- Copyleft 2003-2018 - http://www.videolan.org/x264.html - options: cabac=1
ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00
mixed_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11
fast_pskip=1 chroma_qp_offset=-2 threads=3 lookahead_threads=1 sliced_threads=0
nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3
b_pyramid=2 b_adapt=1 b_bias=0 direct=1 weightb=1 open_gop=0 weightp=2
keyint=250 keyint_min=25 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf
mbtree=1 crf=23.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to '/home/jupyter/videos/runs/cameraman/experiments/0.1_1.0_0.01
_0.001/siren_uniformlr_1e-04_video.mp4':
  Metadata:
    encoder         : Lavf58.20.100
    Stream #0:0: Video: h264 (libx264) (avc1 / 0x31637661), yuv420p, 1296x432
[SAR 1:1 DAR 3:1], q=-1--1, 30 fps, 15360 tbn, 30 tbc
    Metadata:
      encoder         : Lavc58.35.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: -1
frame=  150 fps=104 q=-1.0 Lsize=     260kB time=00:00:04.90 bitrate=
434.8kbits/s dup=140 drop=0 speed=3.39x
video:258kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing
overhead: 1.005734%
```

```
[libx264 @ 0x561f8d9d5f00] frame I:1     Avg QP:16.82  size: 51330
[libx264 @ 0x561f8d9d5f00] frame P:38    Avg QP:19.06  size:  5331
[libx264 @ 0x561f8d9d5f00] frame B:111   Avg QP:14.84  size:    82
[libx264 @ 0x561f8d9d5f00] consecutive B-frames:  1.3%  0.0%  0.0% 98.7%
[libx264 @ 0x561f8d9d5f00] mb I  I16..4: 50.7% 26.2% 23.1%
[libx264 @ 0x561f8d9d5f00] mb P  I16..4:  0.5%  1.6%  0.4%  P16..4:  4.2%  1.4%
1.3%  0.0%  0.0%    skip:90.7%
[libx264 @ 0x561f8d9d5f00] mb B  I16..4:  0.1%  0.0%  0.0%  B16..8:  2.8%  0.0%
0.0%  direct: 0.0%  skip:97.1%  L0:47.8% L1:52.1% BI: 0.1%
[libx264 @ 0x561f8d9d5f00] 8x8 transform intra:44.0% inter:67.3%
[libx264 @ 0x561f8d9d5f00] coded y,uvDC,uvAC intra: 52.8% 54.8% 52.0% inter:
1.0% 1.4% 0.8%
[libx264 @ 0x561f8d9d5f00] i16 v,h,dc,p: 69% 22%  8%  0%
[libx264 @ 0x561f8d9d5f00] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 19% 11% 21%  7%  8%
9%  7%  8% 10%
[libx264 @ 0x561f8d9d5f00] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 25% 24% 18%  6%  5%
6%  5%  5%  5%
[libx264 @ 0x561f8d9d5f00] i8c dc,h,v,p: 58% 18% 18%  6%
[libx264 @ 0x561f8d9d5f00] Weighted P-Frames: Y:0.0% UV:0.0%
[libx264 @ 0x561f8d9d5f00] ref P L0: 70.8% 22.2%  6.3%  0.7%
[libx264 @ 0x561f8d9d5f00] ref B L0: 68.6% 31.1%  0.4%
[libx264 @ 0x561f8d9d5f00] ref B L1: 97.5%  2.5%
[libx264 @ 0x561f8d9d5f00] kb/s:420.80

finished experiment # 3
-----Generating Data-----

100%|     | 10/10 [00:00<00:00, 1108.17it/s]

-----Finished-----
-----Begin Training-----

10it [00:24,  2.49s/it]

Epoch 1, Epoch loss: total 1794.866235, pixel 0.344890, grad 8.248509, laplacian
109.372216, dIdt 17947.564844
Epoch 1, Epoch SSIM: pixel 0.101625, grad 0.026833, laplacian 0.000026, dIdt
0.001582

10it [00:25,  2.60s/it]

Epoch 2, Epoch loss: total 1775.744592, pixel 0.408818, grad 11.107056,
laplacian 1666.020834, dIdt 17740.780859
Epoch 2, Epoch SSIM: pixel 0.007082, grad 0.014969, laplacian 0.000005, dIdt
0.000191

10it [00:25,  2.51s/it]

Epoch 3, Epoch loss: total 1809.437537, pixel 0.349068, grad 14.187024,
laplacian 10766.420215, dIdt 17986.707422
Epoch 3, Epoch SSIM: pixel 0.042241, grad 0.014762, laplacian 0.000002, dIdt
0.000714
```

```
10it [00:26,  2.61s/it]

Epoch 4, Epoch loss: total 2100.757666, pixel 0.343609, grad 30.208072,
laplacian 304889.844873, dIdt 17958.674316
Epoch 4, Epoch SSIM: pixel 0.040865, grad 0.013085, laplacian 0.000003, dIdt
0.000097

10it [00:25,  2.54s/it]

Epoch 5, Epoch loss: total 2041.692188, pixel 0.329382, grad 25.694508,
laplacian 249898.331641, dIdt 17917.934668
Epoch 5, Epoch SSIM: pixel 0.050619, grad 0.010567, laplacian 0.000001, dIdt
-0.000011

10it [00:25,  2.53s/it]

Epoch 6, Epoch loss: total 1988.387708, pixel 0.346298, grad 26.383153,
laplacian 195183.581250, dIdt 17932.037793
Epoch 6, Epoch SSIM: pixel 0.041036, grad 0.009976, laplacian 0.000001, dIdt
-0.000056

10it [00:25,  2.56s/it]

Epoch 7, Epoch loss: total 2053.435474, pixel 0.348727, grad 28.968291,
laplacian 263050.827734, dIdt 17903.842773
Epoch 7, Epoch SSIM: pixel 0.061297, grad 0.008113, laplacian 0.000000, dIdt
-0.000084

10it [00:25,  2.52s/it]

Epoch 8, Epoch loss: total 2202.734277, pixel 0.350646, grad 37.785837,
laplacian 414788.740625, dIdt 17879.451172
Epoch 8, Epoch SSIM: pixel 0.066285, grad 0.009018, laplacian 0.000000, dIdt
-0.000071

10it [00:26,  2.61s/it]

Epoch 9, Epoch loss: total 2238.020190, pixel 0.368077, grad 42.810662,
laplacian 452185.175781, dIdt 17858.345410
Epoch 9, Epoch SSIM: pixel 0.065114, grad 0.007509, laplacian 0.000000, dIdt
-0.000064

10it [00:28,  2.80s/it]

Epoch 10, Epoch loss: total 2253.463062, pixel 0.402176, grad 44.785559,
laplacian 471901.925000, dIdt 17815.606543
Epoch 10, Epoch SSIM: pixel 0.080571, grad 0.006661, laplacian 0.000000, dIdt
-0.000145

10it [00:25,  2.56s/it]

Epoch 11, Epoch loss: total 2131.985803, pixel 0.458949, grad 46.644411,
laplacian 349151.180469, dIdt 17828.341211
Epoch 11, Epoch SSIM: pixel 0.079338, grad 0.007240, laplacian 0.000000, dIdt
-0.000152
```

```
10it [00:25,  2.52s/it]

Epoch 12, Epoch loss: total 2197.925232, pixel 0.499348, grad 45.420939,
laplacian 419567.213281, dIdt 17783.574902
Epoch 12, Epoch SSIM: pixel 0.109144, grad 0.007339, laplacian 0.000000, dIdt
-0.000089

10it [00:25,  2.52s/it]

Epoch 13, Epoch loss: total 2217.877588, pixel 0.552990, grad 55.927468,
laplacian 430840.253125, dIdt 17870.367188
Epoch 13, Epoch SSIM: pixel 0.068456, grad 0.006207, laplacian 0.000000, dIdt
-0.000125

10it [00:25,  2.57s/it]

Epoch 14, Epoch loss: total 2149.671509, pixel 0.526460, grad 45.183729,
laplacian 371920.375781, dIdt 17777.505762
Epoch 14, Epoch SSIM: pixel 0.091550, grad 0.006252, laplacian 0.000000, dIdt
-0.000080

10it [00:25,  2.51s/it]

Epoch 15, Epoch loss: total 2254.380457, pixel 0.643062, grad 57.125675,
laplacian 476514.600000, dIdt 17778.651660
Epoch 15, Epoch SSIM: pixel 0.072173, grad 0.005382, laplacian 0.000000, dIdt
-0.000172

10it [00:25,  2.60s/it]

Epoch 16, Epoch loss: total 2263.104126, pixel 0.638717, grad 59.793797,
laplacian 489054.079688, dIdt 17740.493164
Epoch 16, Epoch SSIM: pixel 0.087960, grad 0.006100, laplacian 0.000000, dIdt
-0.000144

10it [00:25,  2.51s/it]

Epoch 17, Epoch loss: total 2363.737866, pixel 0.753830, grad 69.708949,
laplacian 596823.770312, dIdt 17669.132910
Epoch 17, Epoch SSIM: pixel 0.079889, grad 0.005548, laplacian 0.000000, dIdt
-0.000092

10it [00:26,  2.61s/it]

Epoch 18, Epoch loss: total 2445.548071, pixel 0.870792, grad 76.627430,
laplacian 681998.400000, dIdt 17635.487500
Epoch 18, Epoch SSIM: pixel 0.079763, grad 0.004916, laplacian -0.000000, dIdt
-0.000127

10it [00:25,  2.54s/it]

Epoch 19, Epoch loss: total 2680.395239, pixel 0.880309, grad 105.306596,
laplacian 926203.262500, dIdt 17541.909961
Epoch 19, Epoch SSIM: pixel 0.066640, grad 0.004182, laplacian 0.000000, dIdt
-0.000085
```

```
10it [00:27,  2.80s/it]

Epoch 20, Epoch loss: total 2818.872852, pixel 1.090704, grad 107.666595,
laplacian 1066049.231250, dIdt 17528.225000
Epoch 20, Epoch SSIM: pixel 0.063074, grad 0.001992, laplacian -0.000000, dIdt
-0.000045
-----Finished-----
-----Generating Data-----

100%|     | 10/10 [00:00<00:00, 1285.45it/s]

-----Finished-----

10it [00:16,  1.61s/it]
ffmpeg version 4.1.9-0+deb10u1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 8 (Debian 8.3.0-6)
  configuration: --prefix=/usr --extra-version=0+deb10u1 --toolchain=hardened
--libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
--arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-
filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-
libaom --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca
--enable-libcdio --enable-libcodec2 --enable-libflite --enable-libfontconfig
--enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm
--enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg
--enable-libopenmpt --enable-libopus --enable-libpulse --enable-librsvg
--enable-librubberband --enable-libshine --enable-libsnappy --enable-libsoxr
--enable-libspeex --enable-libssh --enable-libtheora --enable-libtwolame
--enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack
--enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-
libzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-openal --enable-
opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-libiec61883
--enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
  libavutil      56. 22.100 / 56. 22.100
  libavcodec     58. 35.100 / 58. 35.100
  libavformat    58. 20.100 / 58. 20.100
  libavdevice    58.  5.100 / 58.  5.100
  libavfilter     7. 40.101 /  7. 40.101
  libavresample   4.  0.  0 /  4.  0.  0
  libswscale      5.  3.100 /  5.  3.100
  libswresample   3.  3.100 /  3.  3.100
  libpostproc    55.  3.100 / 55.  3.100
Input #0, image2, from 'tmp/file%02d.png':
  Duration: 00:00:05.00, start: 0.000000, bitrate: N/A
    Stream #0:0: Video: png, rgba(pc), 1296x432 [SAR 2835:2835 DAR 3:1], 2 fps,
2 tbr, 2 tbn, 2 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (png (native) -> h264 (libx264))
Press [q] to stop, [?] for help
[libx264 @ 0x555fe683af00] using SAR=1/1
[libx264 @ 0x555fe683af00] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2
```

```
AVX FMA3 BMI2 AVX2
[libx264 @ 0x555fe683af00] profile High, level 3.1
[libx264 @ 0x555fe683af00] 264 - core 155 r2917 0a84d98 - H.264/MPEG-4 AVC codec
- Copyleft 2003-2018 - http://www.videolan.org/x264.html - options: cabac=1
ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00
mixed_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11
fast_pskip=1 chroma_qp_offset=-2 threads=3 lookahead_threads=1 sliced_threads=0
nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3
b_pyramid=2 b_adapt=1 b_bias=0 direct=1 weightb=1 open_gop=0 weightp=2
keyint=250 keyint_min=25 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf
mbtree=1 crf=23.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to '/home/jupyter/videos/runs/cameraman/experiments/0.001_0.0_0.
001_0.1/siren_uniformlr_1e-04_video.mp4':
  Metadata:
    encoder         : Lavf58.20.100
    Stream #0:0: Video: h264 (libx264) (avc1 / 0x31637661), yuv420p, 1296x432
[SAR 1:1 DAR 3:1], q=-1--1, 30 fps, 15360 tbn, 30 tbc
    Metadata:
      encoder         : Lavc58.35.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: -1
frame=  150 fps=103 q=-1.0 Lsize=     267kB time=00:00:04.90 bitrate=
445.9kbits/s dup=140 drop=0 speed=3.36x
video:264kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing
overhead: 0.980566%
[libx264 @ 0x555fe683af00] frame I:1     Avg QP:16.81  size: 51178
[libx264 @ 0x555fe683af00] frame P:38    Avg QP:19.13  size:  5514
[libx264 @ 0x555fe683af00] frame B:111   Avg QP:14.75  size:    82
[libx264 @ 0x555fe683af00] consecutive B-frames:  1.3%  0.0%  0.0% 98.7%
[libx264 @ 0x555fe683af00] mb I  I16..4: 51.1% 23.9% 25.0%
[libx264 @ 0x555fe683af00] mb P  I16..4:  0.6%  1.9%  1.1%  P16..4:  3.8%  0.9%
1.0%  0.0%  0.0%    skip:90.6%
[libx264 @ 0x555fe683af00] mb B  I16..4:  0.1%  0.0%  0.0%  B16..8:  2.8%  0.0%
0.0%  direct: 0.0%  skip:97.1%  L0:46.7% L1:53.2% BI: 0.1%
[libx264 @ 0x555fe683af00] 8x8 transform intra:40.2% inter:59.5%
[libx264 @ 0x555fe683af00] coded y,uvDC,uvAC intra: 57.9% 62.8% 59.3% inter:
0.8% 1.1% 0.6%
[libx264 @ 0x555fe683af00] i16 v,h,dc,p: 70% 21%  8%  1%
[libx264 @ 0x555fe683af00] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 18% 13% 19%  5% 10%
9% 10%  7%  9%
[libx264 @ 0x555fe683af00] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 21% 22% 14%  5%  9%
8%  9%  5%  6%
[libx264 @ 0x555fe683af00] i8c dc,h,v,p: 57% 19% 17%  7%
[libx264 @ 0x555fe683af00] Weighted P-Frames: Y:0.0% UV:0.0%
[libx264 @ 0x555fe683af00] ref P L0: 73.7% 19.3%  6.4%  0.6%
[libx264 @ 0x555fe683af00] ref B L0: 70.0% 29.5%  0.4%
[libx264 @ 0x555fe683af00] ref B L1: 97.6%  2.4%
[libx264 @ 0x555fe683af00] kb/s:431.63
```

```
finished experiment # 4
-----Generating Data-----

100%|     | 10/10 [00:00<00:00, 1280.47it/s]

-----Finished-----
-----Begin Training-----

10it [00:24,  2.47s/it]

Epoch 1, Epoch loss: total 1814.866809, pixel 0.346538, grad 8.280921, laplacian
145.447729, dIdt 17916.945215
Epoch 1, Epoch SSIM: pixel 0.066716, grad 0.027082, laplacian 0.000023, dIdt
0.001077

10it [00:25,  2.58s/it]

Epoch 2, Epoch loss: total 1860.611877, pixel 0.380354, grad 9.547269, laplacian
526.857953, dIdt 17979.984668
Epoch 2, Epoch SSIM: pixel 0.026077, grad 0.020811, laplacian 0.000011, dIdt
0.000759

10it [00:25,  2.52s/it]

Epoch 3, Epoch loss: total 2007.487341, pixel 0.338607, grad 9.528469, laplacian
2000.423846, dIdt 17975.778418
Epoch 3, Epoch SSIM: pixel 0.052165, grad 0.020526, laplacian 0.000005, dIdt
0.000811

10it [00:25,  2.51s/it]

Epoch 4, Epoch loss: total 1915.231946, pixel 0.323519, grad 8.647112, laplacian
1095.440375, dIdt 17967.172852
Epoch 4, Epoch SSIM: pixel 0.066524, grad 0.026051, laplacian 0.000006, dIdt
0.001681

10it [00:25,  2.58s/it]

Epoch 5, Epoch loss: total 2000.744348, pixel 0.311763, grad 8.571822, laplacian
1984.589478, dIdt 17934.017676
Epoch 5, Epoch SSIM: pixel 0.034062, grad 0.021815, laplacian 0.000003, dIdt
0.001552

10it [00:25,  2.52s/it]

Epoch 6, Epoch loss: total 2007.131433, pixel 0.316525, grad 8.451786, laplacian
2063.998999, dIdt 17919.631836
Epoch 6, Epoch SSIM: pixel -0.000337, grad 0.021222, laplacian 0.000005, dIdt
0.001306

10it [00:26,  2.60s/it]

Epoch 7, Epoch loss: total 2127.891479, pixel 0.316967, grad 8.532283, laplacian
3310.638397, dIdt 17879.783398
Epoch 7, Epoch SSIM: pixel 0.048897, grad 0.022880, laplacian 0.000003, dIdt
0.001553
```

```
10it [00:25,  2.52s/it]

Epoch 8, Epoch loss: total 2045.645618, pixel 0.325292, grad 8.582417, laplacian
2518.142517, dIdt 17849.236426
Epoch 8, Epoch SSIM: pixel 0.111544, grad 0.018265, laplacian 0.000004, dIdt
0.001153

10it [00:26,  2.60s/it]

Epoch 9, Epoch loss: total 2195.640747, pixel 0.326631, grad 9.105421, laplacian
4041.611841, dIdt 17820.474316
Epoch 9, Epoch SSIM: pixel 0.149717, grad 0.019854, laplacian 0.000001, dIdt
0.000889

10it [00:27,  2.80s/it]

Epoch 10, Epoch loss: total 2338.577417, pixel 0.340185, grad 9.802977,
laplacian 5436.615308, dIdt 17847.726270
Epoch 10, Epoch SSIM: pixel 0.144651, grad 0.016809, laplacian 0.000001, dIdt
0.000337

10it [00:25,  2.51s/it]

Epoch 11, Epoch loss: total 2447.192749, pixel 0.341265, grad 10.091131,
laplacian 6479.799939, dIdt 17887.802930
Epoch 11, Epoch SSIM: pixel 0.174936, grad 0.015297, laplacian 0.000003, dIdt
0.000655

10it [00:25,  2.59s/it]

Epoch 12, Epoch loss: total 2319.233435, pixel 0.359956, grad 9.305313,
laplacian 5283.228564, dIdt 17812.452246
Epoch 12, Epoch SSIM: pixel 0.179723, grad 0.019449, laplacian 0.000001, dIdt
0.000341

10it [00:25,  2.51s/it]

Epoch 13, Epoch loss: total 2448.509143, pixel 0.396323, grad 9.817775,
laplacian 6651.720801, dIdt 17731.228809
Epoch 13, Epoch SSIM: pixel 0.197913, grad 0.019172, laplacian 0.000001, dIdt
0.000515

10it [00:25,  2.58s/it]

Epoch 14, Epoch loss: total 2672.038867, pixel 0.403108, grad 11.305337,
laplacian 8852.258960, dIdt 17751.044727
Epoch 14, Epoch SSIM: pixel 0.158298, grad 0.013831, laplacian 0.000001, dIdt
0.000052

10it [00:25,  2.51s/it]

Epoch 15, Epoch loss: total 3214.529370, pixel 0.459044, grad 12.215571,
laplacian 14272.570410, dIdt 17745.975977
Epoch 15, Epoch SSIM: pixel 0.171228, grad 0.014736, laplacian 0.000000, dIdt
0.000307
```

```
10it [00:25,  2.59s/it]

Epoch 16, Epoch loss: total 3742.840369, pixel 0.519699, grad 12.350855,
laplacian 19620.176904, dIdt 17679.520605
Epoch 16, Epoch SSIM: pixel 0.158672, grad 0.015579, laplacian 0.000000, dIdt
0.000220

10it [00:25,  2.52s/it]

Epoch 17, Epoch loss: total 3773.448828, pixel 0.454644, grad 13.829960,
laplacian 19860.530664, dIdt 17731.110742
Epoch 17, Epoch SSIM: pixel 0.179908, grad 0.015471, laplacian 0.000000, dIdt
0.000116

10it [00:25,  2.51s/it]

Epoch 18, Epoch loss: total 3802.279419, pixel 0.490274, grad 13.280104,
laplacian 20207.345020, dIdt 17677.744922
Epoch 18, Epoch SSIM: pixel 0.168784, grad 0.011985, laplacian 0.000000, dIdt
0.000094

10it [00:25,  2.58s/it]

Epoch 19, Epoch loss: total 3800.307642, pixel 0.537183, grad 13.228332,
laplacian 20240.468457, dIdt 17624.952148
Epoch 19, Epoch SSIM: pixel 0.174455, grad 0.014050, laplacian 0.000000, dIdt
0.000132

10it [00:27,  2.80s/it]

Epoch 20, Epoch loss: total 4174.798901, pixel 0.581214, grad 14.603018,
laplacian 24003.438037, dIdt 17592.708789
Epoch 20, Epoch SSIM: pixel 0.150035, grad 0.015564, laplacian 0.000000, dIdt
0.000074
-----Finished-----
-----Generating Data-----

100%|      | 10/10 [00:00<00:00, 1286.28it/s]

-----Finished-----

10it [00:21,  2.10s/it]
ffmpeg version 4.1.9-0+deb10u1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 8 (Debian 8.3.0-6)
  configuration: --prefix=/usr --extra-version=0+deb10u1 --toolchain=hardened
--libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
--arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-
filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-
libaom --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca
--enable-libcdio --enable-libcodec2 --enable-libflite --enable-libfontconfig
--enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm
--enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg
--enable-libopenmpt --enable-libopus --enable-libpulse --enable-librsvg
--enable-librubberband --enable-libshine --enable-libsnappy --enable-libsoxr
```

```
--enable-libspeex --enable-libssh --enable-libtheora --enable-libtwolame
--enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack
--enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-
libzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-openal --enable-
opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-libiec61883
--enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
  libavutil      56. 22.100 / 56. 22.100
  libavcodec     58. 35.100 / 58. 35.100
  libavformat    58. 20.100 / 58. 20.100
  libavdevice    58.  5.100 / 58.  5.100
  libavfilter     7. 40.101 /  7. 40.101
  libavresample   4.  0.  0 /  4.  0.  0
  libswscale      5.  3.100 /  5.  3.100
  libswresample   3.  3.100 /  3.  3.100
  libpostproc    55.  3.100 / 55.  3.100
Input #0, image2, from 'tmp/file%02d.png':
  Duration: 00:00:05.00, start: 0.000000, bitrate: N/A
    Stream #0:0: Video: png, rgba(pc), 1296x432 [SAR 2835:2835 DAR 3:1], 2 fps,
2 tbr, 2 tbn, 2 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (png (native) -> h264 (libx264))
Press [q] to stop, [?] for help
[libx264 @ 0x55a062731f00] using SAR=1/1
[libx264 @ 0x55a062731f00] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2
AVX FMA3 BMI2 AVX2
[libx264 @ 0x55a062731f00] profile High, level 3.1
[libx264 @ 0x55a062731f00] 264 - core 155 r2917 0a84d98 - H.264/MPEG-4 AVC codec
- Copyleft 2003-2018 - http://www.videolan.org/x264.html - options: cabac=1
ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00
mixed_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11
fast_pskip=1 chroma_qp_offset=-2 threads=3 lookahead_threads=1 sliced_threads=0
nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3
b_pyramid=2 b_adapt=1 b_bias=0 direct=1 weightb=1 open_gop=0 weightp=2
keyint=250 keyint_min=25 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf
mbtree=1 crf=23.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to '/home/jupyter/videos/runs/cameraman/experiments/1.0_1.0_0.1_
0.1/siren_uniformlr_1e-04_video.mp4':
  Metadata:
    encoder         : Lavf58.20.100
    Stream #0:0: Video: h264 (libx264) (avc1 / 0x31637661), yuv420p, 1296x432
[SAR 1:1 DAR 3:1], q=-1--1, 30 fps, 15360 tbn, 30 tbc
    Metadata:
      encoder         : Lavc58.35.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: -1
frame=  150 fps=104 q=-1.0 Lsize=     254kB time=00:00:04.90 bitrate=
424.9kbits/s dup=140 drop=0 speed= 3.4x
video:252kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing
```

```
overhead: 1.029547%
[libx264 @ 0x55a062731f00] frame I:1     Avg QP:15.70  size: 47904
[libx264 @ 0x55a062731f00] frame P:38    Avg QP:19.00  size:  5259
[libx264 @ 0x55a062731f00] frame B:111   Avg QP:14.73  size:    82
[libx264 @ 0x55a062731f00] consecutive B-frames:  1.3%  0.0%  0.0% 98.7%
[libx264 @ 0x55a062731f00] mb I  I16..4: 58.3% 17.0% 24.7%
[libx264 @ 0x55a062731f00] mb P  I16..4:  0.6%  2.3%  0.9%  P16..4:  3.8%  0.9%
0.9%  0.0%  0.0%    skip:90.6%
[libx264 @ 0x55a062731f00] mb B  I16..4:  0.1%  0.0%  0.0%  B16..8:  2.9%  0.0%
0.0%  direct: 0.0%  skip:97.0%  L0:46.6% L1:53.3% BI: 0.1%
[libx264 @ 0x55a062731f00] 8x8 transform intra:42.0% inter:59.6%
[libx264 @ 0x55a062731f00] coded y,uvDC,uvAC intra: 58.4% 63.5% 60.5% inter:
0.8% 1.1% 0.6%
[libx264 @ 0x55a062731f00] i16 v,h,dc,p: 73% 20%  6%  1%
[libx264 @ 0x55a062731f00] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 14% 13% 17%  7% 10%
10%  9%  9% 12%
[libx264 @ 0x55a062731f00] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 25% 22% 16%  5%  7%
6%  7%  5%  6%
[libx264 @ 0x55a062731f00] i8c dc,h,v,p: 54% 20% 19%  8%
[libx264 @ 0x55a062731f00] Weighted P-Frames: Y:0.0% UV:0.0%
[libx264 @ 0x55a062731f00] ref P L0: 73.8% 18.4%  7.1%  0.7%
[libx264 @ 0x55a062731f00] ref B L0: 71.1% 28.5%  0.4%
[libx264 @ 0x55a062731f00] ref B L1: 97.3%  2.7%
[libx264 @ 0x55a062731f00] kb/s:411.04

finished experiment # 5
-----Generating Data-----

100%|     | 10/10 [00:00<00:00, 1159.22it/s]

-----Finished-----
-----Begin Training-----

10it [00:24,  2.48s/it]

Epoch 1, Epoch loss: total 190.568272, pixel 0.344656, grad 8.148641, laplacian
104.195838, dIdt 17933.348535
Epoch 1, Epoch SSIM: pixel 0.032766, grad 0.027506, laplacian 0.000024, dIdt
0.001690

10it [00:25,  2.60s/it]

Epoch 2, Epoch loss: total 223.934142, pixel 0.345734, grad 9.043941, laplacian
444.560037, dIdt 17857.340234
Epoch 2, Epoch SSIM: pixel 0.018716, grad 0.018599, laplacian 0.000010, dIdt
0.000977

10it [00:25,  2.51s/it]

Epoch 3, Epoch loss: total 255.237726, pixel 0.334775, grad 8.909799, laplacian
747.814856, dIdt 17956.492871
```

```
Epoch 3, Epoch SSIM: pixel 0.020453, grad 0.020034, laplacian 0.000008, dIdt
0.000884

10it [00:25,  2.53s/it]

Epoch 4, Epoch loss: total 462.159679, pixel 0.319729, grad 9.529771, laplacian
2812.653687, dIdt 17994.100098
Epoch 4, Epoch SSIM: pixel 0.056193, grad 0.020033, laplacian 0.000003, dIdt
0.001105

10it [00:25,  2.58s/it]

Epoch 5, Epoch loss: total 452.182953, pixel 0.319670, grad 8.941711, laplacian
2718.746960, dIdt 17941.377734
Epoch 5, Epoch SSIM: pixel 0.035697, grad 0.023118, laplacian 0.000003, dIdt
0.001593

10it [00:25,  2.54s/it]

Epoch 6, Epoch loss: total 353.849197, pixel 0.308936, grad 8.682603, laplacian
1736.287311, dIdt 17935.189844
Epoch 6, Epoch SSIM: pixel 0.077749, grad 0.021318, laplacian 0.000005, dIdt
0.000978

10it [00:25,  2.58s/it]

Epoch 7, Epoch loss: total 295.821411, pixel 0.312839, grad 8.332347, laplacian
1160.650034, dIdt 17892.285449
Epoch 7, Epoch SSIM: pixel 0.074073, grad 0.020851, laplacian 0.000004, dIdt
0.001449

10it [00:25,  2.51s/it]

Epoch 8, Epoch loss: total 368.288702, pixel 0.315093, grad 8.561647, laplacian
1888.020422, dIdt 17863.017383
Epoch 8, Epoch SSIM: pixel 0.058733, grad 0.018582, laplacian 0.000001, dIdt
0.000614

10it [00:26,  2.60s/it]

Epoch 9, Epoch loss: total 467.217441, pixel 0.319720, grad 8.641083, laplacian
2881.013983, dIdt 17825.162793
Epoch 9, Epoch SSIM: pixel 0.124318, grad 0.017580, laplacian 0.000002, dIdt
0.000401

10it [00:27,  2.80s/it]

Epoch 10, Epoch loss: total 713.758423, pixel 0.339931, grad 9.718257, laplacian
5347.353027, dIdt 17805.094336
Epoch 10, Epoch SSIM: pixel 0.186489, grad 0.015669, laplacian 0.000001, dIdt
0.000432

10it [00:25,  2.54s/it]

Epoch 11, Epoch loss: total 842.909290, pixel 0.325829, grad 10.048907,
laplacian 6636.391064, dIdt 17826.495996
```

```
Epoch 11, Epoch SSIM: pixel 0.165996, grad 0.014063, laplacian 0.000001, dIdt
0.000339

10it [00:26,  2.62s/it]

Epoch 12, Epoch loss: total 1419.564450, pixel 0.368746, grad 11.414354,
laplacian 12403.762891, dIdt 17804.634473
Epoch 12, Epoch SSIM: pixel 0.187385, grad 0.014755, laplacian 0.000001, dIdt
0.000213

10it [00:25,  2.52s/it]

Epoch 13, Epoch loss: total 1643.796716, pixel 0.365290, grad 12.368138,
laplacian 14645.331445, dIdt 17802.634863
Epoch 13, Epoch SSIM: pixel 0.184158, grad 0.012082, laplacian 0.000001, dIdt
0.000158

10it [00:25,  2.57s/it]

Epoch 14, Epoch loss: total 1752.933618, pixel 0.381586, grad 12.499214,
laplacian 15737.570801, dIdt 17792.616895
Epoch 14, Epoch SSIM: pixel 0.189889, grad 0.011957, laplacian 0.000000, dIdt
0.000182

10it [00:25,  2.51s/it]

Epoch 15, Epoch loss: total 1622.605798, pixel 0.404907, grad 11.703171,
laplacian 14444.354102, dIdt 17699.966504
Epoch 15, Epoch SSIM: pixel 0.206009, grad 0.017348, laplacian 0.000000, dIdt
0.000115

10it [00:25,  2.56s/it]

Epoch 16, Epoch loss: total 1844.198645, pixel 0.483079, grad 12.193454,
laplacian 16666.670264, dIdt 17631.177051
Epoch 16, Epoch SSIM: pixel 0.190710, grad 0.019222, laplacian 0.000000, dIdt
0.000139

10it [00:25,  2.52s/it]

Epoch 17, Epoch loss: total 2519.064728, pixel 0.548760, grad 14.309171,
laplacian 23420.258789, dIdt 17560.734570
Epoch 17, Epoch SSIM: pixel 0.164187, grad 0.019163, laplacian 0.000000, dIdt
0.000040

10it [00:25,  2.51s/it]

Epoch 18, Epoch loss: total 3187.327197, pixel 0.652958, grad 17.297070,
laplacian 30100.005762, dIdt 17559.616992
Epoch 18, Epoch SSIM: pixel 0.151996, grad 0.016163, laplacian 0.000000, dIdt
0.000067

10it [00:26,  2.61s/it]

Epoch 19, Epoch loss: total 2795.569495, pixel 0.672258, grad 17.069961,
laplacian 26184.390332, dIdt 17542.270313
```

```
Epoch 19, Epoch SSIM: pixel 0.131105, grad 0.014221, laplacian 0.000000, dIdt
0.000086
```

```
10it [00:27,  2.79s/it]
```

```
Epoch 20, Epoch loss: total 3663.679553, pixel 0.769440, grad 18.601243,
laplacian 34880.887598, dIdt 17372.982812
Epoch 20, Epoch SSIM: pixel 0.125092, grad 0.013949, laplacian 0.000000, dIdt
0.000175
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/tmp/ipykernel_2988/1545087382.py in <module>
     55     for uniform_lr in learning_rates:
     56         model_path_full = model_path_act + 'uniformlr_' + "{:.0e}".
 ↪format(uniform_lr)
---> 57         run_siren(model_path_full, betas, total_epochs, [uniform_lr])
     58 
     59     # 2. run with decaying learning rates

/tmp/ipykernel_2988/2675429778.py in run_siren(model_path, betas, total_epochs,␣
 ↪lr, cyclic, decay_exp, decay_multi)
      9             total_epochs=total_epochs, lr=lr,
     10             beta_0=betas[0], beta_1=betas[1], beta_2=betas[2],␣
 ↪beta_3=betas[3],
---> 11             cyclic=cyclic, decay_exp=decay_exp, decay_multi=decay_multi)

     12 
     13     writer.close()

/tmp/ipykernel_2988/962146541.py in train(net, writer, img_path, niter,␣
 ↪total_epochs, lr, beta_0, beta_1, beta_2, beta_3, cyclic, decay_exp,␣
 ↪decay_multi)
    131         print("Epoch %d, Epoch SSIM: pixel %0.6f, grad %0.6f, laplacian␣
 ↪%0.6f, dIdt %0.6f" % (epoch, epoch_pixel_ssim/len(image), epoch_grad_ssim/
 ↪len(image), epoch_laplacian_ssim/len(image), epoch_dIdt_ssim/len(image)))
    132 
--> 133     writer.add_graph(net, model_input)
    134     print("-----Finished-----")
    135 

/opt/conda/lib/python3.7/site-packages/torch/utils/tensorboard/writer.py in␣
 ↪add_graph(self, model, input_to_model, verbose, use_strict_trace)
    734         if hasattr(model, 'forward'):
    735             # A valid PyTorch model should have a 'forward' method
--> 736             self._get_file_writer().add_graph(graph(model,␣
 ↪input_to_model, verbose, use_strict_trace))
    737         else:
    738             # Caffe2 models do not have the 'forward' method
```

```
/opt/conda/lib/python3.7/site-packages/torch/utils/tensorboard/_pytorch_graph.p␣
 ↪in graph(model, args, verbose, use_strict_trace)
    289     with torch.onnx.select_model_mode_for_export(model, torch.onnx.
 ↪TrainingMode.EVAL):  # TODO: move outside of torch.onnx?
    290         try:
--> 291             trace = torch.jit.trace(model, args, strict=use_strict_trac )
    292             graph = trace.graph
    293             torch._C._jit_pass_inline(graph)

/opt/conda/lib/python3.7/site-packages/torch/jit/_trace.py in trace(func,␣
 ↪example_inputs, optimize, check_trace, check_inputs, check_tolerance, strict,␣
 ↪_force_outplace, _module_class, _compilation_unit)
    748             strict,
    749             _force_outplace,
--> 750             _module_class,
    751         )
    752

/opt/conda/lib/python3.7/site-packages/torch/jit/_trace.py in trace_module(mod,␣
 ↪inputs, optimize, check_trace, check_inputs, check_tolerance, strict,␣
 ↪_force_outplace, _module_class, _compilation_unit)
    989                         _force_outplace,
    990                         True,
--> 991                         _module_class,
    992                     )
    993     finally:

/opt/conda/lib/python3.7/site-packages/torch/autograd/grad_mode.py in␣
 ↪decorate_context(*args, **kwargs)
     25         def decorate_context(*args, **kwargs):
     26             with self.clone():
---> 27                 return func(*args, **kwargs)
     28         return cast(F, decorate_context)
     29

/opt/conda/lib/python3.7/site-packages/torch/jit/_trace.py in␣
 ↪_check_trace(check_inputs, func, traced_func, check_tolerance, strict,␣
 ↪force_outplace, is_trace_module, _module_class)
    516         traced_outs = run_mod_and_filter_tensor_outputs(traced_func,␣
 ↪inputs, "trace")
    517         fn_outs = run_mod_and_filter_tensor_outputs(func, inputs,␣
 ↪"Python function")
--> 518         if compare_outputs(traced_outs, fn_outs, "Python function"):
    519             check_outs = run_mod_and_filter_tensor_outputs(
    520                 check_mod_func, inputs, "repeated trace"

/opt/conda/lib/python3.7/site-packages/torch/jit/_trace.py in␣
 ↪compare_outputs(original, reference, match_what)
```

```
    495                             rtol=check_tolerance,
    496                             atol=default_tolerances(orig, ref)[1],
--> 497                             equal_nan=True,
    498                         )
    499                     except AssertionError as e:

    [… skipping hidden 2 frame]

/opt/conda/lib/python3.7/site-packages/torch/testing/_comparison.py in␣
 ↪compare(self)
    602             actual, expected = self._equalize_attributes(actual,␣
 ↪expected)
    603
--> 604             self._compare_values(actual, expected)
    605
    606     @contextlib.contextmanager

/opt/conda/lib/python3.7/site-packages/torch/testing/_comparison.py in␣
 ↪_compare_values(self, actual, expected)
    714             compare_fn = self._compare_regular_values_close
    715
--> 716         compare_fn(actual, expected, rtol=self.rtol, atol=self.atol,␣
 ↪equal_nan=self.equal_nan)
    717
    718     def _compare_quantized_values(

/opt/conda/lib/python3.7/site-packages/torch/testing/_comparison.py in␣
 ↪_compare_regular_values_close(self, actual, expected, rtol, atol, equal_nan,␣
 ↪identifier)
    840         actual, expected = self._promote_for_comparison(actual, expecte )
    841         matches = torch.isclose(actual, expected, rtol=rtol, atol=atol,
 ↪equal_nan=equal_nan)
--> 842         if torch.all(matches):
    843             return
    844

KeyboardInterrupt:
```

Error in callback <function install_repl_displayhook.<locals>.post_execute at 0x7fd3ac486ef0> (for post_execute):

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/opt/conda/lib/python3.7/site-packages/matplotlib/pyplot.py in post_execute()
    136     def post_execute():
    137         if matplotlib.is_interactive():
--> 138             draw_all()
```

```
    139
    140     try:  # IPython >= 2

/opt/conda/lib/python3.7/site-packages/matplotlib/_pylab_helpers.py in␣
 ↪draw_all(cls, force)
    135         for manager in cls.get_all_fig_managers():
    136             if force or manager.canvas.figure.stale:
--> 137                 manager.canvas.draw_idle()
    138
    139

/opt/conda/lib/python3.7/site-packages/matplotlib/backend_bases.py in␣
 ↪draw_idle(self, *args, **kwargs)
   2058         if not self._is_idle_drawing:
   2059             with self._idle_draw_cntx():
-> 2060                 self.draw(*args, **kwargs)
   2061
   2062     @property

/opt/conda/lib/python3.7/site-packages/matplotlib/backends/backend_agg.py in␣
 ↪draw(self)
    434               (self.toolbar._wait_cursor_for_draw_cm() if self.toolbar
    435                else nullcontext()):
--> 436             self.figure.draw(self.renderer)
    437             # A GUI class may be need to update a window using this␣
 ↪draw, so
    438             # don't forget to call the superclass.

/opt/conda/lib/python3.7/site-packages/matplotlib/artist.py in␣
 ↪draw_wrapper(artist, renderer, *args, **kwargs)
     71     @wraps(draw)
     72     def draw_wrapper(artist, renderer, *args, **kwargs):
---> 73         result = draw(artist, renderer, *args, **kwargs)
     74         if renderer._rasterizing:
     75             renderer.stop_rasterizing()

/opt/conda/lib/python3.7/site-packages/matplotlib/artist.py in␣
 ↪draw_wrapper(artist, renderer)
     48                 renderer.start_filter()
     49
---> 50             return draw(artist, renderer)
     51         finally:
     52             if artist.get_agg_filter() is not None:

/opt/conda/lib/python3.7/site-packages/matplotlib/figure.py in draw(self,␣
 ↪renderer)
   2809             self.patch.draw(renderer)
   2810             mimage._draw_list_compositing_images(
-> 2811                 renderer, self, artists, self.suppressComposite)

   2812
   2813             for sfig in self.subfigs:

/opt/conda/lib/python3.7/site-packages/matplotlib/image.py in␣
 ↪_draw_list_compositing_images(renderer, parent, artists, suppress_composite)
    130     if not_composite or not has_images:
    131         for a in artists:
--> 132             a.draw(renderer)
    133     else:
    134         # Composite any adjacent images together

/opt/conda/lib/python3.7/site-packages/matplotlib/artist.py in␣
 ↪draw_wrapper(artist, renderer)
     48                 renderer.start_filter()
     49
---> 50             return draw(artist, renderer)
     51         finally:
     52             if artist.get_agg_filter() is not None:

/opt/conda/lib/python3.7/site-packages/matplotlib/axes/_base.py in draw(self,␣
 ↪renderer)
   3081
   3082         mimage._draw_list_compositing_images(
-> 3083             renderer, self, artists, self.figure.suppressComposite)

   3084
   3085         renderer.close_group('axes')

/opt/conda/lib/python3.7/site-packages/matplotlib/image.py in␣
 ↪_draw_list_compositing_images(renderer, parent, artists, suppress_composite)
    130     if not_composite or not has_images:
    131         for a in artists:
--> 132             a.draw(renderer)
    133     else:
    134         # Composite any adjacent images together

/opt/conda/lib/python3.7/site-packages/matplotlib/artist.py in␣
 ↪draw_wrapper(artist, renderer)
     48                 renderer.start_filter()
     49
---> 50             return draw(artist, renderer)
     51         finally:
     52             if artist.get_agg_filter() is not None:

/opt/conda/lib/python3.7/site-packages/matplotlib/axis.py in draw(self,␣
 ↪renderer, *args, **kwargs)
   1161
   1162         for tick in ticks_to_draw:
-> 1163             tick.draw(renderer)
   1164 
   1165         # scale up the axis label box to also find the neighbors, not

/opt/conda/lib/python3.7/site-packages/matplotlib/artist.py in␣
 ↪draw_wrapper(artist, renderer)
     48                 renderer.start_filter()
     49 
---> 50             return draw(artist, renderer)
     51         finally:
     52             if artist.get_agg_filter() is not None:

/opt/conda/lib/python3.7/site-packages/matplotlib/axis.py in draw(self, rendere )
    297         for artist in [self.gridline, self.tick1line, self.tick2line,
    298                        self.label1, self.label2]:
--> 299             artist.draw(renderer)
    300         renderer.close_group(self.__name__)
    301         self.stale = False

/opt/conda/lib/python3.7/site-packages/matplotlib/artist.py in␣
 ↪draw_wrapper(artist, renderer)
     48                 renderer.start_filter()
     49 
---> 50             return draw(artist, renderer)
     51         finally:
     52             if artist.get_agg_filter() is not None:

/opt/conda/lib/python3.7/site-packages/matplotlib/text.py in draw(self, rendere )
    682         renderer.open_group('text', self.get_gid())
    683 
--> 684         with self._cm_set(text=self._get_wrapped_text()):
    685             bbox, info, descent = self._get_layout(renderer)
    686             trans = self.get_transform()

/opt/conda/lib/python3.7/contextlib.py in __enter__(self)
    105         return self.__class__(self.func, self.args, self.kwds)
    106 
--> 107     def __enter__(self):
    108         # do not keep args and kwds alive unnecessarily
    109         # they are only needed for recreation, which is not possible␣
 ↪anymore

KeyboardInterrupt:
```

```
Error in callback <function flush_figures at 0x7fd3acbd4830> (for post_execute):
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/opt/conda/lib/python3.7/site-packages/matplotlib_inline/backend_inline.py in␣
 ↪flush_figures()
    119         # ignore the tracking, just draw and close all figures
    120         try:
--> 121             return show(True)
    122         except Exception as e:
    123             # safely show traceback if in IPython, else raise

/opt/conda/lib/python3.7/site-packages/matplotlib_inline/backend_inline.py in␣
 ↪show(close, block)
     41             display(
     42                 figure_manager.canvas.figure,
---> 43                 metadata=_fetch_figure_metadata(figure_manager.canvas.
 ↪figure)
     44             )
     45     finally:

/opt/conda/lib/python3.7/site-packages/IPython/core/display.py in␣
 ↪display(include, exclude, metadata, transient, display_id, *objs, **kwargs)
    318             publish_display_data(data=obj, metadata=metadata, **kwargs)
    319         else:
--> 320             format_dict, md_dict = format(obj, include=include,␣
 ↪exclude=exclude)
    321             if not format_dict:
    322                 # nothing to display (e.g. _ipython_display_ took over)

/opt/conda/lib/python3.7/site-packages/IPython/core/formatters.py in␣
 ↪format(self, obj, include, exclude)
    178             md = None
    179             try:
--> 180                 data = formatter(obj)
    181             except:
    182                 # FIXME: log the exception

/opt/conda/lib/python3.7/site-packages/decorator.py in fun(*args, **kw)
    230             if not kwsyntax:
    231                 args, kw = fix(args, kw, sig)
--> 232             return caller(func, *(extras + args), **kw)
    233     fun.__name__ = func.__name__
    234     fun.__doc__ = func.__doc__

/opt/conda/lib/python3.7/site-packages/IPython/core/formatters.py in␣
 ↪catch_format_error(method, self, *args, **kwargs)
    222     """show traceback on failed format call"""
    223     try:
```

```
--> 224         r = method(self, *args, **kwargs)
    225     except NotImplementedError:
    226         # don't warn on NotImplementedErrors

/opt/conda/lib/python3.7/site-packages/IPython/core/formatters.py in␣
 ↪__call__(self, obj)
    339                 pass
    340             else:
--> 341                 return printer(obj)
    342             # Finally look for special method names
    343             method = get_real_method(obj, self.print_method)

/opt/conda/lib/python3.7/site-packages/IPython/core/pylabtools.py in␣
 ↪print_figure(fig, fmt, bbox_inches, base64, **kwargs)
    149         FigureCanvasBase(fig)
    150
--> 151     fig.canvas.print_figure(bytes_io, **kw)
    152     data = bytes_io.getvalue()
    153     if fmt == 'svg':

/opt/conda/lib/python3.7/site-packages/matplotlib/backend_bases.py in␣
 ↪print_figure(self, filename, dpi, facecolor, edgecolor, orientation, format,␣
 ↪bbox_inches, pad_inches, bbox_extra_artists, backend, **kwargs)
   2298                 if bbox_inches == "tight":
   2299                     bbox_inches = self.figure.get_tightbbox(
-> 2300                         renderer, bbox_extra_artists=bbox_extra_artists

   2301                     if pad_inches is None:
   2302                         pad_inches = rcParams['savefig.pad_inches']

/opt/conda/lib/python3.7/site-packages/matplotlib/figure.py in␣
 ↪get_tightbbox(self, renderer, bbox_extra_artists)
   1630
   1631         for a in artists:
-> 1632             bbox = a.get_tightbbox(renderer)
   1633             if bbox is not None and (bbox.width != 0 or bbox.height !=␣
 ↪0):
   1634                 bb.append(bbox)

/opt/conda/lib/python3.7/site-packages/matplotlib/axes/_base.py in␣
 ↪get_tightbbox(self, renderer, call_axes_locator, bbox_extra_artists,␣
 ↪for_layout_only)
   4627                 try:
   4628                     bb_yaxis = self.yaxis.get_tightbbox(
-> 4629                         renderer, for_layout_only=for_layout_only)

   4630                 except TypeError:
   4631                     # in case downstream library has redefined axis:
```

```
/opt/conda/lib/python3.7/site-packages/matplotlib/axis.py in get_tightbbox(self ␣
 ↪renderer, for_layout_only)
   1103         ticks_to_draw = self._update_ticks()
   1104 
-> 1105         self._update_label_position(renderer)
   1106 
   1107         # go back to just this axis's tick labels

/opt/conda/lib/python3.7/site-packages/matplotlib/axis.py in␣
 ↪_update_label_position(self, renderer)
   2350         # get bounding boxes for this axis and any siblings
   2351         # that have been set by `fig.align_ylabels()`
-> 2352         bboxes, bboxes2 = self.
 ↪_get_tick_boxes_siblings(renderer=renderer)
   2353 
   2354         x, y = self.label.get_position()

/opt/conda/lib/python3.7/site-packages/matplotlib/axis.py in␣
 ↪_get_tick_boxes_siblings(self, renderer)
   1878         for ax in grouper.get_siblings(self.axes):
   1879             axis = getattr(ax, f"{axis_name}axis")
-> 1880             ticks_to_draw = axis._update_ticks()
   1881             tlb, tlb2 = axis._get_tick_bboxes(ticks_to_draw, renderer)
   1882             bboxes.extend(tlb)

/opt/conda/lib/python3.7/site-packages/matplotlib/axis.py in _update_ticks(self
   1046         major_labels = self.major.formatter.format_ticks(major_locs)
   1047         major_ticks = self.get_major_ticks(len(major_locs))
-> 1048         self.major.formatter.set_locs(major_locs)
   1049         for tick, loc, label in zip(major_ticks, major_locs,␣
 ↪major_labels):
   1050             tick.update_position(loc)

/opt/conda/lib/python3.7/site-packages/matplotlib/ticker.py in set_locs(self,␣
 ↪locs)
    709             if self._useOffset:
    710                 self._compute_offset()
--> 711             self._set_order_of_magnitude()
    712             self._set_format()
    713 

/opt/conda/lib/python3.7/site-packages/matplotlib/ticker.py in␣
 ↪_set_order_of_magnitude(self)
    765         vmin, vmax = sorted(self.axis.get_view_interval())
    766         locs = np.asarray(self.locs)
--> 767         locs = locs[(vmin <= locs) & (locs <= vmax)]
    768         locs = np.abs(locs)
    769         if not len(locs):
```

```
KeyboardInterrupt:
```

0.0.12 Visualize Results

```
[28]: %load_ext tensorboard
      %tensorboard --logdir="runs"
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard

Reusing TensorBoard on port 6006 (pid 2841), started 0:11:02 ago. (Use '!kill␣
↪2841' to kill it.)

<IPython.core.display.HTML object>
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```