HorizonNet 코드 분석

길다영

2021. 07. 19

https://github.com/sunset1995/HorizonNet 참고.

목차
Contents

1

데이터 훈련

Preprocess.py
Train.py

2

Pre-Processing

Preprocess.py

3

Estimating Layout with HorizonNet

inference.py
/misc/post_proc.py

4

Layout 3D Viewer

Layout_viewer.py

1. 데이터 훈련

Preprocess.py
Train.py

결과

1. 데이터 훈련

python preprocess.py --img_glob ssu_dataset/train/img/pic424_1.png --output_dir ssu_dataset/preprocessed --rgbonly

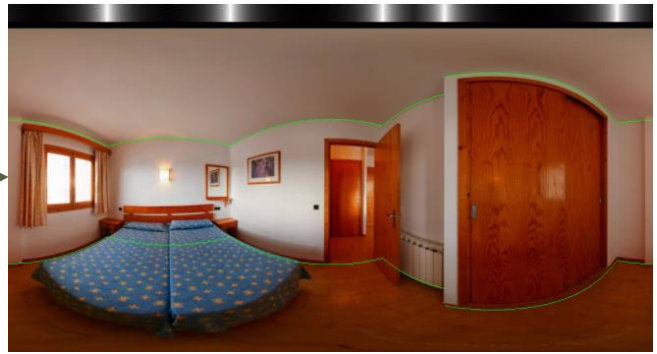
preprocess.py

이미지 조정(정렬)

python train.py --train_root_dir ssu_dataset/train --valid_root_dir ssu_dataset/valid --batch_size_train 1 --num_workers 0 --id model_bs1

train.py

데이터 세트에 대해 학습
Batch_size, epochs, num_workers 지정



https://velog.io/@arittung/%EC%A1%B8%EC%97%85%EC%9E%91%ED%92%88-3D-Room-Reconstruction-Using-HorizonNet-%EC%A7%81%EC%A0%91-%EB%8D%B0%EC%9D%B4%ED%84%B0-%ED%9B%88%EB%A0%A8-%ED%9B%84-%EC%8B%A4%ED%96%89

python preprocess.py --img_glob ssu_dataset/train/img/pic424_1.png --output_dir ssu_dataset/preprocessed --rgbonly

1. 데이터 훈련

```python
# Process each input
for i_path in paths:
    print('Processing', i_path, flush=True)

    # Load and cat input images
    img_ori = np.array(Image.open(i_path).resize((1024, 512), Image.BICUBIC))[..., :3]

    # VP detection and line segment extraction
    _, vp, _, _, panoEdge, _, _ = panoEdgeDetection(img_ori,
                                                    qError=args.q_error,
                                                    refineIter=args.refine_iter)
    panoEdge = (panoEdge > 0)

    # Align images with VP
    i_img = rotatePanorama(img_ori / 255.0, vp[2::-1])
    l_img = rotatePanorama(panoEdge.astype(np.float32), vp[2::-1])
```

소실점 가진 이미지를 align
(구부려진 사진이 펴짐)

preprocess.py

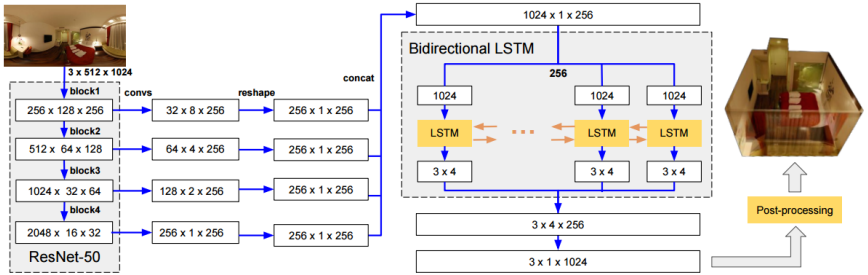
python train.py --train_root_dir ssu_dataset/train --valid_root_dir ssu_dataset/valid --batch_size_train 1 --num_workers 0 --id model_bs1

1. 데이터 훈련

```
13 from model import HorizonNet, ENCODER_RESNET, ENCODER_DENSENET
14 from dataset import PanoCorBonDataset
15 from misc.utils import group_weight, adjust_learning_rate, save_model, load_trained_model
16 from inference import inference
17 from eval_general import test_general
```

Model.py에서
HorizonNet(ResNet50 & LSTM) 가져오기

Model.py로 이동




```
123     # Create model
124     if args.pth is not None:
125         print('Finetune model is given.')
126         print('Ignore --backbone and --no_rnn')
127         net = load_trained_model(HorizonNet, args.pth).to(device)
128     else:
129         net = HorizonNet(args.backbone, not args.no_rnn).to(device)
```

Model.py에서 가져온 HorizonNet 사용.

train.py

ResNet 50 이외의 다른 특징 추출기
선택하여 진행해보기

RNN 존재하는 경우와
존재하지 않는 경우 직접 비교

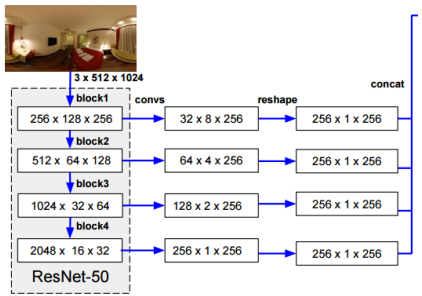
1. 데이터 훈련

CNN(ResNet50)

```python
ENCODER_RESNET = [
    'resnet18', 'resnet34', 'resnet50', 'resnet101', 'resnet152',
    'resnext50_32x4d', 'resnext101_32x8d'
]
```

```python
HorizonNet
'''
class HorizonNet(nn.Module):
    x_mean = torch.FloatTensor(np.array([0.485, 0.456, 0.406])[None, :, None, None])
    x_std = torch.FloatTensor(np.array([0.229, 0.224, 0.225])[None, :, None, None])

    def __init__(self, backbone, use_rnn):
        super(HorizonNet, self).__init__()
        self.backbone = backbone
        self.use_rnn = use_rnn
        self.out_scale = 8
        self.step_cols = 4
        self.rnn_hidden_size = 512

        # Encoder
        if backbone.startswith('res'):
            self.feature_extractor = Resnet(backbone, pretrained=True)
        elif backbone.startswith('dense'):
            self.feature_extractor = Densenet(backbone, pretrained=True)
        else:
            raise NotImplementedError()

        # Inference channels number from each block of the encoder
        with torch.no_grad():
            dummy = torch.zeros(1, 3, 512, 1024)
            c1, c2, c3, c4 = [b.shape[1] for b in self.feature_extractor(dummy)]
            c_last = (c1*8 + c2*4 + c3*2 + c4*1) // self.out_scale

        # Convert features from 4 blocks of the encoder into B x C x 1 x W'
        self.reduce_height_module = GlobalHeightStage(c1, c2, c3, c4, self.out_scale)
```

```python
class Resnet(nn.Module):
    def __init__(self, backbone='resnet50', pretrained=True):
        super(Resnet, self).__init__()
        assert backbone in ENCODER_RESNET
        self.encoder = getattr(models, backbone)(pretrained=pretrained)
        del self.encoder.fc, self.encoder.avgpool

    def forward(self, x):
        features = []
        x = self.encoder.conv1(x)
        x = self.encoder.bn1(x)
        x = self.encoder.relu(x)
        x = self.encoder.maxpool(x)

        x = self.encoder.layer1(x);  features.append(x)  # 1/4
        x = self.encoder.layer2(x);  features.append(x)  # 1/8
        x = self.encoder.layer3(x);  features.append(x)  # 1/16
        x = self.encoder.layer4(x);  features.append(x)  # 1/32
        return features

    def list_blocks(self):
        lst = [m for m in self.encoder.children()]
        block0 = lst[:4]
        block1 = lst[4:5]
        block2 = lst[5:6]
        block3 = lst[6:7]
        block4 = lst[7:8]
        return block0, block1, block2, block3, block4
```



model.py

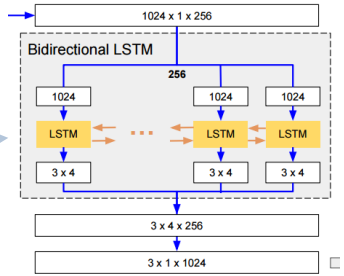
https://velog.io/@arittung/ResNet-50

1.

데이터 훈련

RNN(LSTM)

```
203         # 1D prediction
204         if self.use_rnn:
205             self.bi_rnn = nn.LSTM(input_size=c_last,
206                                   hidden_size=self.rnn_hidden_size,
207                                   num_layers=2,
208                                   dropout=0.5,
209                                   batch_first=False,
210                                   bidirectional=True)
211             self.drop_out = nn.Dropout(0.5)
212             self.linear = nn.Linear(in_features=2 * self.rnn_hidden_size,
213                                     out_features=3 * self.step_cols)
214             self.linear.bias.data[0*self.step_cols:1*self.step_cols].fill_(-1)
215             self.linear.bias.data[1*self.step_cols:2*self.step_cols].fill_(-0.478)
216             self.linear.bias.data[2*self.step_cols:3*self.step_cols].fill_(0.425)
217         else:
218             self.linear = nn.Sequential(
219                 nn.Linear(c_last, self.rnn_hidden_size),
220                 nn.ReLU(inplace=True),
221                 nn.Dropout(0.5),
222                 nn.Linear(self.rnn_hidden_size, 3 * self.step_cols),
223             )
224             self.linear[-1].bias.data[0*self.step_cols:1*self.step_cols].fill_(-1)
225             self.linear[-1].bias.data[1*self.step_cols:2*self.step_cols].fill_(-0.478)
226             self.linear[-1].bias.data[2*self.step_cols:3*self.step_cols].fill_(0.425)
227         self.x_mean.requires_grad = False
228         self.x_std.requires_grad = False
229         wrap_lr_pad(self)
230
```



model.py

https://velog.io/@arittung/ResNet-50

1. 데이터 훈련

```python
    def _prepare_x(self, x):
        if self.x_mean.device != x.device:
            self.x_mean = self.x_mean.to(x.device)
            self.x_std = self.x_std.to(x.device)
        return (x[:, :3] - self.x_mean) / self.x_std

    def forward(self, x):
        if x.shape[2] != 512 or x.shape[3] != 1024:
            raise NotImplementedError()
        x = self._prepare_x(x)
        conv_list = self.feature_extractor(x)
        feature = self.reduce_height_module(conv_list, x.shape[3]//self.step_cols)

        # rnn
        if self.use_rnn:
            feature = feature.permute(2, 0, 1)  # [w, b, c*h]
            output, hidden = self.bi_rnn(feature)  # [seq_len, b, num_directions * hidden_size]
            output = self.drop_out(output)
            output = self.linear(output)  # [seq_len, b, 3 * step_cols]
            output = output.view(output.shape[0], output.shape[1], 3, self.step_cols)  # [seq_len, b, 3, step_cols]
            output = output.permute(1, 2, 0, 3)  # [b, 3, seq_len, step_cols]
            output = output.contiguous().view(output.shape[0], 3, -1)  # [b, 3, seq_len*step_cols]
        else:
            feature = feature.permute(0, 2, 1)  # [b, w, c*h]
            output = self.linear(feature)  # [b, w, 3 * step_cols]
            output = output.view(output.shape[0], output.shape[1], 3, self.step_cols)  # [b, w, 3, step_cols]
            output = output.permute(0, 2, 1, 3)  # [b, 3, w, step_cols]
            output = output.contiguous().view(output.shape[0], 3, -1)  # [b, 3, w*step_cols]

        # output.shape => B x 3 x W
        cor = output[:, :1]  # B x 1 x W
        bon = output[:, 1:]  # B x 2 x W

        return bon, cor
```

model.py

1. 데이터 훈련

```
python train.py --train_root_dir ssu_dataset/train --valid_root_dir ssu_dataset/valid --batch_size_train 1 --num_workers 0 --id model_bs1
```

```
# Create dataloader
dataset_train = PanoCorBonDataset(
    root_dir=args.train_root_dir,
    flip=not args.no_flip, rotate=not args.no_rotate, gamma=not args.no_gamma,
    stretch=not args.no_pano_stretch)
loader_train = DataLoader(dataset_train, args.batch_size_train,
                          shuffle=True, drop_last=True,
                          num_workers=args.num_workers,
                          pin_memory=not args.no_cuda,
                          worker_init_fn=lambda x: np.random.seed())
```

Epoch만큼 for문 돌림.

```
# Start training
for ith_epoch in trange(1, args.epochs + 1, desc='Epoch', unit='ep'):

    # Train phase
    net.train()
    if args.freeze_earlier_blocks != -1:
        b0, b1, b2, b3, b4 = net.feature_extractor.list_blocks()
        blocks = [b0, b1, b2, b3, b4]
        for i in range(args.freeze_earlier_blocks + 1):
            for m in blocks[i]:
                m.eval()
    iterator_train = iter(loader_train)
    for _ in trange(len(loader_train),
                    desc='Train ep%s' % ith_epoch, position=1):
        # Set learning rate
        adjust_learning_rate(optimizer, args)

        args.cur_iter += 1
        x, y_bon, y_cor = next(iterator_train)

        losses = feed_forward(net, x, y_bon, y_cor)
        for k, v in losses.items():
            k = 'train/%s' % k
            tb_writer.add_scalar(k, v.item(), args.cur_iter)
        tb_writer.add_scalar('train/lr', args.running_lr, args.cur_iter)
        loss = losses['total']

        # backprop
        optimizer.zero_grad()
        loss.backward()
        nn.utils.clip_grad_norm_(net.parameters(), 3.0, norm_type='inf')
        optimizer.step()
```

train.py

2. Pre-Processing

Preprocess.py

2. Pre-processing

assets/demo.png

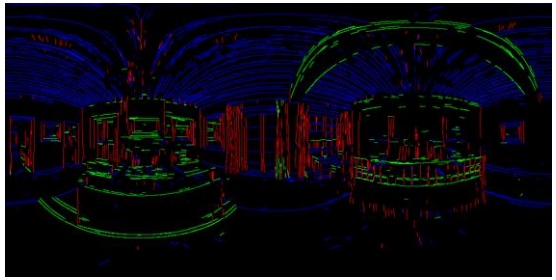
python preprocess.py --img_glob assets/demo.png --output_dir assets/preprocessed/

preprocess.py

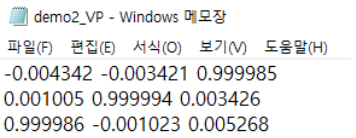
결과

demo.aligned_rgb.png

demo.aligned_line.png



demo_VP.txt



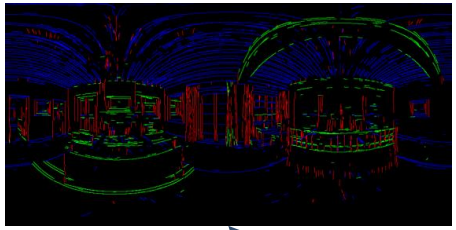
https://velog.io/@arittung/3D-Room-Reconstruction-Using-HorizonNet-%EC%B4%88%EA%B8%B0-%ED%99%98%EA%B2%BD-%EC%84%A4%EC%A0%95-HorizonNet-%EC%8B%A4%ED%96%89

2. Pre-processing

python preprocess.py --img_glob assets/demo.png --output_dir assets/preprocessed/

```python
# Process each input
for i_path in paths:
    print('Processing', i_path, flush=True)

    # Load and cat input images
    img_ori = np.array(Image.open(i_path).resize((1024, 512), Image.BICUBIC))[..., :3]

    # VP detection and line segment extraction
    _, vp, _, _, panoEdge, _, _ = panoEdgeDetection(img_ori,
                                                    qError=args.q_error,
                                                    refineIter=args.refine_iter)
    panoEdge = (panoEdge > 0)

    # Align images with VP
    i_img = rotatePanorama(img_ori / 255.0, vp[2::-1])
    l_img = rotatePanorama(panoEdge.astype(np.float32), vp[2::-1])

    # Dump results
    basename = os.path.splitext(os.path.basename(i_path))[0]
    if args.rgbonly:
        path = os.path.join(args.output_dir, '%s.png' % basename)
        Image.fromarray((i_img * 255).astype(np.uint8)).save(path)
    else:
        path_VP = os.path.join(args.output_dir, '%s_VP.txt' % basename)
        path_i_img = os.path.join(args.output_dir, '%s_aligned_rgb.png' % basename)
        path_l_img = os.path.join(args.output_dir, '%s_aligned_line.png' % basename)

        with open(path_VP, 'w') as f:
            for i in range(3):
                f.write('%.6f %.6f %.6f\n' % (vp[i, 0], vp[i, 1], vp[i, 2]))
        Image.fromarray((i_img * 255).astype(np.uint8)).save(path_i_img)
        Image.fromarray((l_img * 255).astype(np.uint8)).save(path_l_img)
```

preprocess.py



demo2_VP - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

-0.004342 -0.003421 0.999985
0.001005 0.999994 0.003426
0.999986 -0.001023 0.005268

파노라마 사진에서
edge와 소실점을 찾음

소실점 가진 이미지를 align
(구부려진 사진이 펴짐)

3. Estimating Layout with HorizonNet

inference.py
/misc/post_proc.py

3. Estimating Layout with HorizonNet

결과

demo_aligned_rgb.raw.png

python inference.py --pth ckpt/resnet50_rnn__st3d.pth --img_glob assets/preprocessed/demo_aligned_rgb.png --output_dir assets/inferenced --visualize

inference.py

demo2_aligned_rgb.json

```
{"z0": 50.0, "z1": -32.29869079589844,
"uv": [[0.13787418603897095, 0.36454907059669495],
[0.13787418603897095, 0.5889818668365479],
[0.3714289665222168, 0.3713512718677521],
[0.3714289665222168, 0.5841352343559265],
[0.6208978891372681, 0.3664819896221161],
[0.6208978891372681, 0.5875996351242065],
[0.867792546749115, 0.35881996154785156],
[0.867792546749115, 0.5931026935577393]]}
```

demo_aligned_rgb.json

https://velog.io/@arittung/3D-Room-Reconstruction-Using-HorizonNet-%EC%B4%88%EA%B8%B0-%ED%99%98%EA%B2%BD-%EC%84%A4%EC%A0%95-HorizonNet-%EC%8B%A4%ED%96%89

3. python inference.py --pth ckpt/resnet50_rnn__st3d.pth --img_glob assets/preprocessed/demo_aligned_rgb.png --output_dir assets/inferenced --visualize

Estimating Layout with HorizonNet

Preprocess된 이미지 불러옴

```python
# Prepare image to processed
paths = sorted(glob.glob(args.img_glob))
if len(paths) == 0:
    print('no images found')
for path in paths:
    assert os.path.isfile(path), '%s not found' % path

# Check target directory
if not os.path.isdir(args.output_dir):
    print('Output directory %s not existed. Create one.' % args.output_dir)
    os.makedirs(args.output_dir)
device = torch.device('cpu' if args.no_cuda else 'cuda')

# Loaded trained model
net = utils.load_trained_model(HorizonNet, args.pth).to(device)
net.eval()
```

Trained model(pth 파일) 불러옴

Pth 파일(?)

inference.py

Inference()로 넘어감.

```python
# Inferencing
with torch.no_grad():
    for i_path in tqdm(paths, desc='Inferencing'):
        k = os.path.split(i_path)[-1][:-4]

        # Load image
        img_pil = Image.open(i_path)
        if img_pil.size != (1024, 512):
            img_pil = img_pil.resize((1024, 512), Image.BICUBIC)
        img_ori = np.array(img_pil)[..., :3].transpose([2, 0, 1]).copy()
        x = torch.FloatTensor([img_ori / 255])

        # Inferenceing corners
        cor_id, z0, z1, vis_out = inference(net, x, device,
                                            args.flip, args.rotate,
                                            args.visualize,
                                            args.force_cuboid,
                                            args.min_v, args.r)

        # Output result
        with open(os.path.join(args.output_dir, k + '.json'), 'w') as f:
            json.dump({
                'z0': float(z0),
                'z1': float(z1),
                'uv': [[float(u), float(v)] for u, v in cor_id],
            }, f)

        if vis_out is not None:
            vis_path = os.path.join(args.output_dir, k + '.raw.png')
            vh, vw = vis_out.shape[:2]
            Image.fromarray(vis_out)\
                 .resize((vw//2, vh//2), Image.LANCZOS)\
                 .save(vis_path)
```

3. Estimating Horizon with Layout Net

python inference.py --pth ckpt/resnet50_rnn__st3d.pth --img_glob assets/preprocessed/demo_aligned_rgb.png --output_dir assets/inferenced --visualize

```python
def inference(net, x, device, flip=False, rotate=[], visualize=False,
              force_cuboid=True, min_v=None, r=0.05):
    '''

    # Init floor/ceil plane
    z0 = 50
    _, z1 = post_proc.np_refine_by_fix_z(*y_bon_, z0)

    # Detech wall-wall peaks
    if min_v is None:
        min_v = 0 if force_cuboid else 0.05
    r = int(round(W * r / 2))
    N = 4 if force_cuboid else None
    xs_ = find_N_peaks(y_cor_, r=r, min_v=min_v, N=N)[0]

    # Generate wall-walls
    cor, xy_cor = post_proc.gen_ww(xs_, y_bon_[0], z0, tol=abs(0.16 * z1 / 1.6), force_cuboid=force_cuboid)
    if not force_cuboid:
        # Check valid (for fear self-intersection)
        xy2d = np.zeros((len(xy_cor), 2), np.float32)
        for i in range(len(xy_cor)):
            xy2d[i, xy_cor[i]['type']] = xy_cor[i]['val']
            xy2d[i, xy_cor[i-1]['type']] = xy_cor[i-1]['val']
        if not Polygon(xy2d).is_valid:
            print(
                'Fail to generate valid general layout!! '
                'Generate cuboid as fallback.',
                file=sys.stderr)
            xs_ = find_N_peaks(y_cor_, r=r, min_v=0, N=4)[0]
            cor, xy_cor = post_proc.gen_ww(xs_, y_bon_[0], z0, tol=abs(0.16 * z1 / 1.6), force_cuboid=True)
```

1

inference.py

논문 3.2 Recovering the Floor and Ceiling Planes

바닥, 천장 평면 초기화

논문 3.3 Recovering Wall Planes / Figure4 (a)

Find_N_peaks() : prominent peak를 찾는 과정

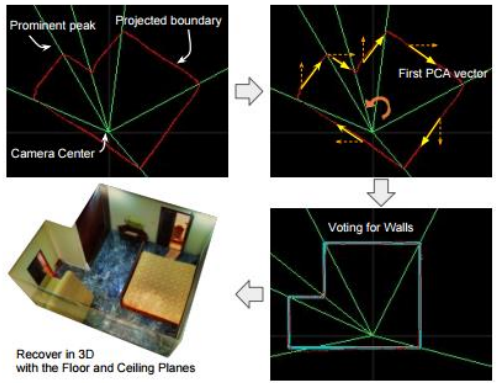
Gen.ww()
: cuboid, non-cuboid 함수 구분
: gen_ww_cuboid() → vote()
: gen_ww_general() → vote()

/misc/post_proc.py로 이동

2

```python
def find_N_peaks(signal, r=29, min_v=0.05, N=None):
    max_v = maximum_filter(signal, size=r, mode='wrap')
    pk_loc = np.where(max_v == signal)[0]
    pk_loc = pk_loc[signal[pk_loc] > min_v]
    if N is not None:
        order = np.argsort(-signal[pk_loc])
        pk_loc = pk_loc[order[:N]]
        pk_loc = pk_loc[np.argsort(pk_loc)]
    return pk_loc, signal[pk_loc]
```

inference.py



(a) Depicting how we recover the wall planes from our model output.

3. Estimating Horizon with Layout Net

1

```python
def gen_ww(init_coorx, coory, z=50, coorW=1024, coorH=512, floorW=1024, floorH=512, tol=3, force_cuboid=True):
    gpid = get_gpid(init_coorx, coorW)
    coor = np.hstack([np.arange(coorW)[:, None], coory[:, None]])
    xy = np_coor2xy(coor, z, coorW, coorH, floorW, floorH)

    # Generate wall-wall
    if force_cuboid:
        xy_cor = gen_ww_cuboid(xy, gpid, tol)
    else:
        xy_cor = gen_ww_general(init_coorx, xy, gpid, tol)
```

Cuboid 경우

Non-cuboid 경우

2

```python
def gen_ww_cuboid(xy, gpid, tol):
    xy_cor = []
    assert len(np.unique(gpid)) == 4

    # For each part seperated by wall-wall peak, voting for a wall
    for j in range(4):
        now_x = xy[gpid == j, 0]
        now_y = xy[gpid == j, 1]
        new_x, x_score, x_l1 = vote(now_x, tol)
        new_y, y_score, y_l1 = vote(now_y, tol)
        if (x_score, -x_l1) > (y_score, -y_l1):
            xy_cor.append({'type': 0, 'val': new_x, 'score': x_score})
        else:
            xy_cor.append({'type': 1, 'val': new_y, 'score': y_score})
```

For문의 개수 차이
Cuboid : 4개
Non-cuboid : 모서리 개수

/misc/post_proc.py

3

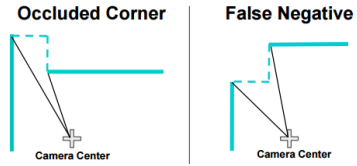
```python
def vote(vec, tol):
    vec = np.sort(vec)
    n = np.arange(len(vec))[::-1]
    n = n[:, None] - n[None, :] + 1.0
    l = squareform(pdist(vec[:, None], 'minkowski', p=1) + 1e-9)

    invalid = (n < len(vec) * 0.4) | (l > tol)
    if (~invalid).sum() == 0 or len(vec) < tol:
        best_fit = np.median(vec)
        p_score = 0
    else:
        l[invalid] = 1e5
        n[invalid] = -1
        score = n
        max_idx = score.argmax()
        max_row = max_idx // len(vec)
        max_col = max_idx % len(vec)
        assert max_col > max_row
        best_fit = vec[max_row:max_col+1].mean()
        p_score = (max_col - max_row + 1) / len(vec)

    l1_score = np.abs(vec - best_fit).mean()

    return best_fit, p_score, l1_score
```

/misc/post_proc.py

Non-cuboid 보완해보기

2

```python
def gen_ww_general(init_coorx, xy, gpid, tol):
    xy_cor = []
    assert len(init_coorx) == len(np.unique(gpid))

    # Candidate for each part seperated by wall-wall boundary
    for j in range(len(init_coorx)):
        now_x = xy[gpid == j, 0]
        now_y = xy[gpid == j, 1]
        new_x, x_score, x_l1 = vote(now_x, tol)
        new_y, y_score, y_l1 = vote(now_y, tol)
        u0 = np_coorx2u(init_coorx[(j - 1 + len(init_coorx)) % len(init_coorx)])
        u1 = np_coorx2u(init_coorx[j])
        if (x_score, -x_l1) > (y_score, -y_l1):
            xy_cor.append({'type': 0, 'val': new_x, 'score': x_score, 'action': 'ori', 'gpid': j, 'u0': u0, 'u1': u1, 'tbd': True})
        else:
            xy_cor.append({'type': 1, 'val': new_y, 'score': y_score, 'action': 'ori', 'gpid': j, 'u0': u0, 'u1': u1, 'tbd': True})
```



벽에 투표하는 대신, 두개의 두드러진 봉우리와 두 개의 벽의 위치에 따라 코너를 추가함.

/misc/post_proc.py

3. python inference.py --pth ckpt/resnet50_rnn__st3d.pth --img_glob assets/preprocessed/demo_aligned_rgb.png --output_dir assets/inferenced --visualize

Estimating Horizon with Layout Net

```
120        # Expand with btn coory
121        cor = np.hstack([cor, post_proc.infer_coory(cor[:, 1], z1 - z0, z0)[:, None]])
122
123        # Collect corner position in equirectangular
124        cor_id = np.zeros((len(cor)*2, 2), np.float32)
125        for j in range(len(cor)):
126            cor_id[j*2] = cor[j, 0], cor[j, 1]
127            cor_id[j*2 + 1] = cor[j, 0], cor[j, 2]
128
129        # Normalized to [0, 1]
130        cor_id[:, 0] /= W
131        cor_id[:, 1] /= H
132
133        return cor_id, z0, z1, vis_out
```

inference.py

4. Layout
3D Viewer

Layout_viewer.py

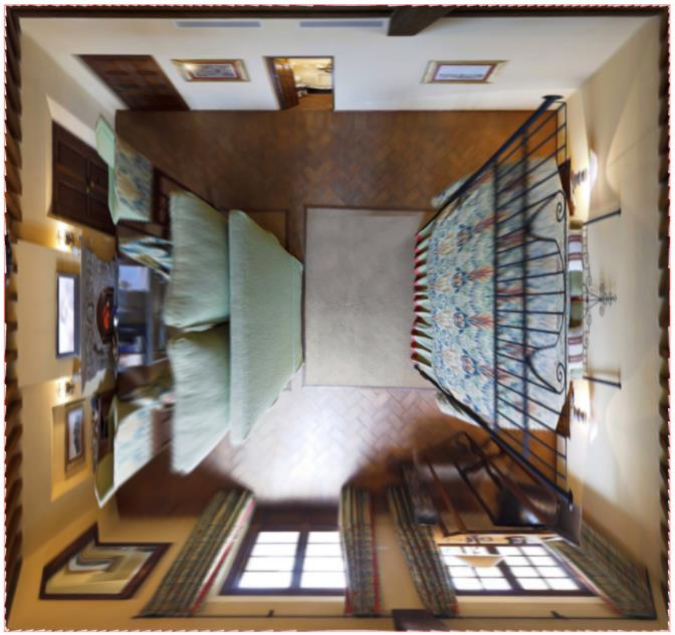
3.

Layout 3D Viewer

결과

python layout_viewer.py --img ssu_dataset/preprocessed/pic424_1_aligned_rgb.png --layout ssu_dataset/train/label_cor/pic424_1_aligned_rgb.json --ignore_ceiling --vis

Layout_viewer.py



https://velog.io/@arittung/3D-Room-Reconstruction-Using-HorizonNet-%EC%B4%88%EA%B8%B0-%ED%99%98%EA%B2%BD-%EC%84%A4%EC%A0%95-HorizonNet-%EC%8B%A4%ED%96%89

감사합니다.

The End