SuperPoint 转 onnx

目的：方便 tensorRT 加速部署

参考 :pytorch 转 onnx https://blog.csdn.net/qq_39967751/article/details/124932242

pytorch 源码链接：https://github.com/magicleap/SuperGluePretrainedNetwork

1）要点：

- 在模型推理一次后执行 torch.onnx.export ，具体含义参见上述链接;

demo_superglue.py

```
last_data = matching.superpoint(frame_tensor)
torch.onnx.export(matching.superpoint, frame_tensor, f='SuperPoint.onnx', export_params=True,
verbose=True,
                  input_names=None,
                  output_names=None, do_constant_folding=True, dynamic_axes=None,
opset_version=11)
exit()
```

- SuperPoint 原始推理代码中包含后处理部分，此部分不能被正确导出，只能导出网络的原始输出，包括描述子（desc）和特征相应图（scores），后处理需要自己根据 python 代码进行实现，修改如下：

  superpoint.py

```
def forward(self, data):
    """ Compute keypoints, scores, descriptors for image """
    # Shared Encoder
    x = self.relu(self.conv1a(data)) #['image']
    x = self.relu(self.conv1b(x))
    x = self.pool(x)
    x = self.relu(self.conv2a(x))
    x = self.relu(self.conv2b(x))
    x = self.pool(x)
    x = self.relu(self.conv3a(x))
    x = self.relu(self.conv3b(x))
    x = self.pool(x)
    x = self.relu(self.conv4a(x))
    x = self.relu(self.conv4b(x))

    # Compute the dense keypoint scores
    cPa = self.relu(self.convPa(x))
```

```
    scores = self.convPb(cPa)
    scores = torch.nn.functional.softmax(scores, 1)[:, :-1]
    b, _, h, w = scores.shape
    scores = scores.permute(0, 2, 3, 1).reshape(b, h, w, 8, 8)
    scores = scores.permute(0, 1, 3, 2, 4).reshape(b, h*8, w*8)
    # scores = simple_nms(scores, self.config['nms_radius'])
    #
    #
    # # Extract keypoints
    # keypoints = [
    #     torch.nonzero(s > self.config['keypoint_threshold'])
    #     for s in scores]
    # scores = [s[tuple(k.t())] for s, k in zip(scores, keypoints)]
    #
    # # Discard keypoints near the image borders
    # keypoints, scores = list(zip(*[
    #     remove_borders(k, s, self.config['remove_borders'], h*8, w*8)
    #     for k, s in zip(keypoints, scores)]))
    #
    # # Keep the k keypoints with highest score
    # if self.config['max_keypoints'] >= 0:
    #     keypoints, scores = list(zip(*[
    #         top_k_keypoints(k, s, self.config['max_keypoints'])
    #         for k, s in zip(keypoints, scores)]))
    #
    # # Convert (h, w) to (x, y)
    # keypoints = [torch.flip(k, [1]).float() for k in keypoints]

    # Compute the dense descriptors
    cDa = self.relu(self.convDa(x))
    descriptors = self.convDb(cDa)
    descriptors = torch.nn.functional.normalize(descriptors, p=2, dim=1)

    # Extract descriptors
    # descriptors = [sample_descriptors(k[None], d[None], 8)[0]
    #                for k, d in zip(keypoints, descriptors)]

    return scores, descriptors
```

- onnx 转换时不支持词典类型的数据输入，修给如下：
  superpoint.py

```
x = self.relu(self.conv1a(data)) #['image']
```

同时其他地方的输入也需要修改， 如
demo_superglue.py

```
last_data = matching.superpoint(frame_tensor)
```

2）

运行：

- 代码 SuperGluePretrainedNetwork，按 ReadME 安装依赖

电脑有摄像头，运行：./demo_superglue.py --resize 640 480

无摄像头：./demo_superglue.py --input images_dir --output_dir output_dir --resize 640 480

可在 SuperGluePretrainedNetwork 下找到 onnx 文件