

一. 环境准备

在 autodl 上租的，配置如下；



项目整体运行跑通了，系统盘镜像已经保存

镜像UUID	镜像名称	大小	状态	共享信息	来源	缓存地区	原基础镜像信息	创建时间	操作
image-2fc0c5cb12	codef+sam_track	24.29GB	● 就绪	私有镜像	AutoDL	重庆区	PyTorch 2.0.0 Python 3.8(ubuntu20.04) CUDA 11.8	2025-11-21 19:52:33	编辑 共享 删除

然后大概我把数据盘打包成压缩包就可以完全移植这个实例了吧

名称	修改时间
CoDeF_main	18小时前
Segment-and-Track-Anything-main	3天前
blackswan.mp4	1天前
CoDeF_main.zip	4天前
Segment-and-Track-Anything-main.zip	3天前

二. Segment-and-Track-Anything-main

新建一个虚拟环境单独安装依赖

```
(base) root@autodl-container-c0d64989c1-c3c508de:~/autodl-tmp/ALL_project# conda env list
# conda environments:
#
base                  * /root/miniconda3
codef_env              /root/miniconda3/envs/codef_env
sam_track               /root/miniconda3/envs/sam_track
```

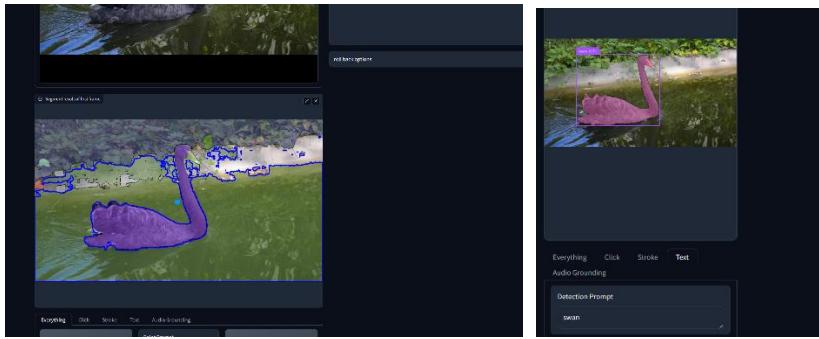
脚本运行过程有一些库版本和缺失，一个个装就行

Autodl 貌似不能直接通过浏览器打开这个项目的网页 ui，需要在控制台自定义服务那找到下面这个

再下面这样，本地浏览器就可以打开网页了



然后效果如下：



这个项目是在终端执行的，下一个光流是在 notebook 执行

三 . 光流

这个问题要少一些，网络问题不好下模型，本地下完传上去就好了

```
download_models.sh
raft-chairs.pth
raft-kitti.pth
raft-sintel.pth
raft-small.pth
raft-things.pth
```

```
[3]: # 引入 argparse 来创建正确的参数对象
import argparse

# 定义配置 (这里根据命令行参数)
video_name = "beauty_1" # 你的视频名
root_dir = "../all_sequences"

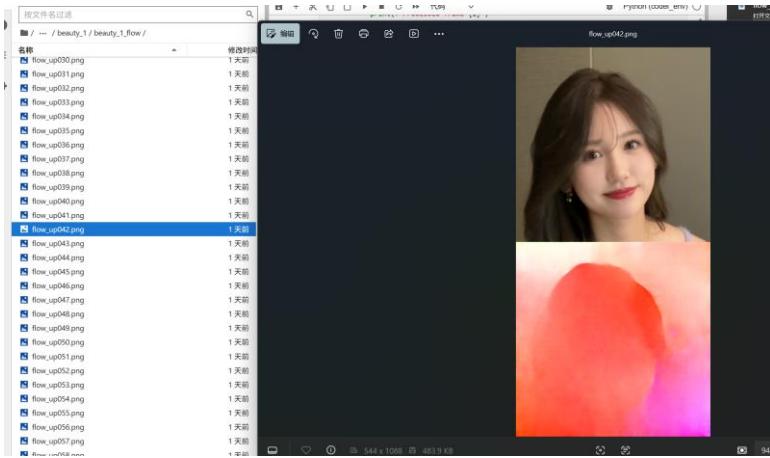
# 使用 Namespace 直接创建参数, 完美模拟 parser.parse_args() 的结果
args = argparse.Namespace(
    model="models/raft-things.pth",
    # 预设配置
    name=video_name,
    pathos.path.join(root_dir, video_name, video_name),
    outdir=os.path.join(root_dir, video_name, f"{(video_name)}_flow"),
    outdir_conf=os.path.join(root_dir, video_name, f"{(video_name)}_flow_confidence"),
    # 顶层参数
    small=False,
    mixed_precision=False,
    alternate_corr=False,
    thresh=4
)
# 动态开关
if mask==False,
    confidence=True,
    discrete=False,
    thresh=4
)

print("# 准备开始...")
print("读取: {args.path}")
print(f"输出: {args.outdir}")

# 运行 demo
# 注意: 因为你的环境是 PyTorch 2.x, 这里运行可能会报 meshgrid 警告或错误
# 如果报错, 请看下面的“关于 meshgrid 的修复”
try:
    demo(args)
    print("运行完成!")
except Exception as e:
    print("发生错误: {e}")
    if "meshgrid" in str(e) or "indexing" in str(e):
        print("\n⚠️ 错误原因: PyTorch 2.0 版本兼容性问题。")
        print("解决方法: 你需要修改 demo 函数中的代码:")
        print("将 grid_x, grid_y=torch.meshgrid(x,y) 改为 grid_x,grid_y=torch.meshgrid(x,y, indexing='ij')")

# 准备开始...
读取: ../all_sequences/beauty_1/beauty_1
```

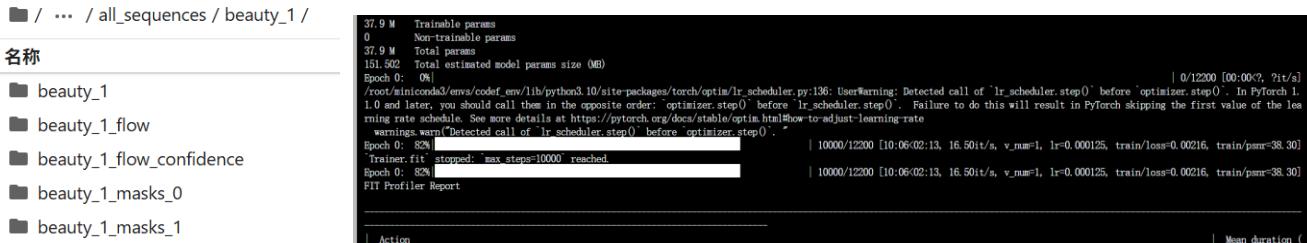
然后获得给光流文件



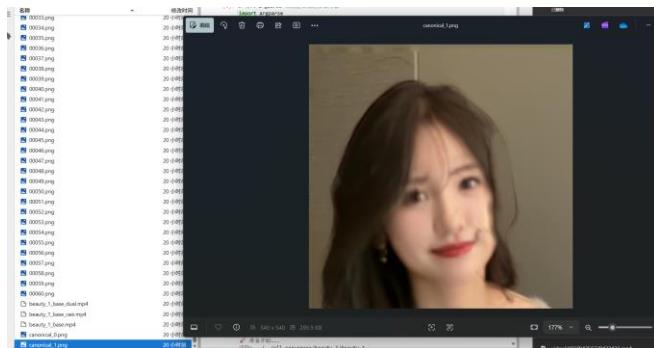
四 . Codef

数据预处理完成后，我是在终端运行和测试的
文件结构

训练



获得规范场图



保存的模型

... / beauty_1 / base /

名称	修改时间
last-v1.ckpt	21小时前
step=10000.ckpt	21小时前
step=5000.ckpt	21小时前

最后测试和视频重建

The screenshot shows a terminal window titled "终端 1" with the command "test_multi.sh" running. The script content is as follows:

```
1 #!/bin/bash
2
3 GPU=0
4 NAME=beauty_1
5 EXP_NAME=base
6
7 # 数据路径
8 ROOT_DIRECTORY="all_sequences/${NAME}/${NAME}"
9 # 结果保存路径 (改成 results 方便找)
10 LOG_SAVE_PATH="results/${NAME}"
11 MASK_DIRECTORY="all_sequences/${NAME}/masks_0 all_sequences/${NAME}/${NAME}_masks_1"
12
13 WEIGHT_PATH="/root/autodl-tmp/ALL_project/CoDef_main/ckpts/beauty_1/base/last-v1.ckpt"
14 # WEIGHT_PATH=ckpts/all_sequences/${NAME}/${EXP_NAME}/step=10000.ckpt
15
16 echo "正在测试视频: ${NAME}"
17 echo "加载模型: ${WEIGHT_PATH}"
18 echo "结果保存: ${LOG_SAVE_PATH}"
19
20 python train.py --test --encode_w \
21   --root_dir ${ROOT_DIRECTORY} \
22   --log_save_path ${LOG_SAVE_PATH} \
23   --mask_dir ${MASK_DIRECTORY} \
24   --weight_path "${WEIGHT_PATH}" \
25   --gpus ${GPU} \
26   --config configs/${NAME}/${EXP_NAME}.yaml \
27   --exp_name ${EXP_NAME} \
28   --save_deform False
29
```

To the left of the terminal is a file explorer window showing a directory structure under "/... / beauty_1 / base /". The directory contains numerous PNG files (0033.png to 0060.png) and three MP4 files: beauty_1_base_dual.mp4, beauty_1_base_raw.mp4, and beauty_1_base.mp4. There are also two canonical_0.png files.