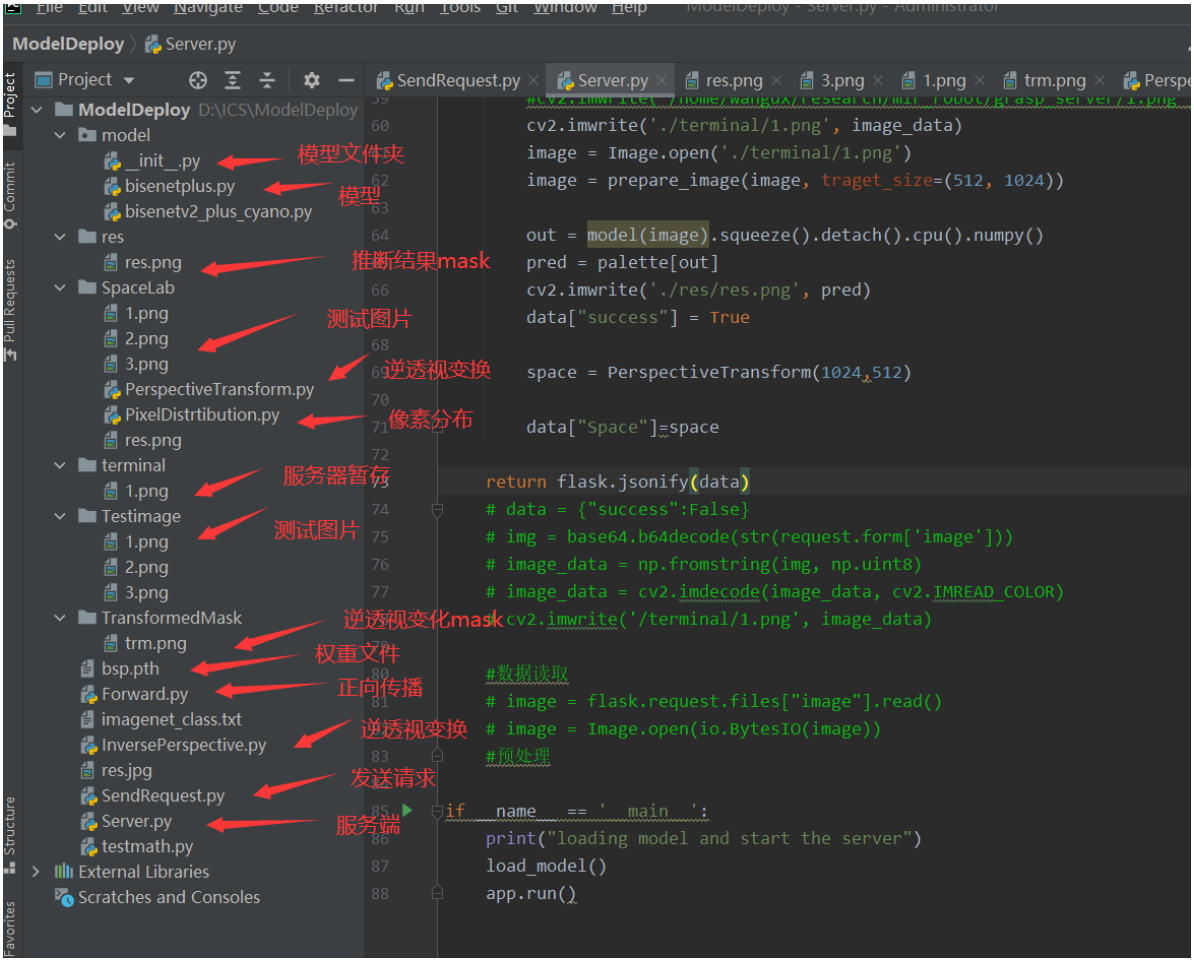
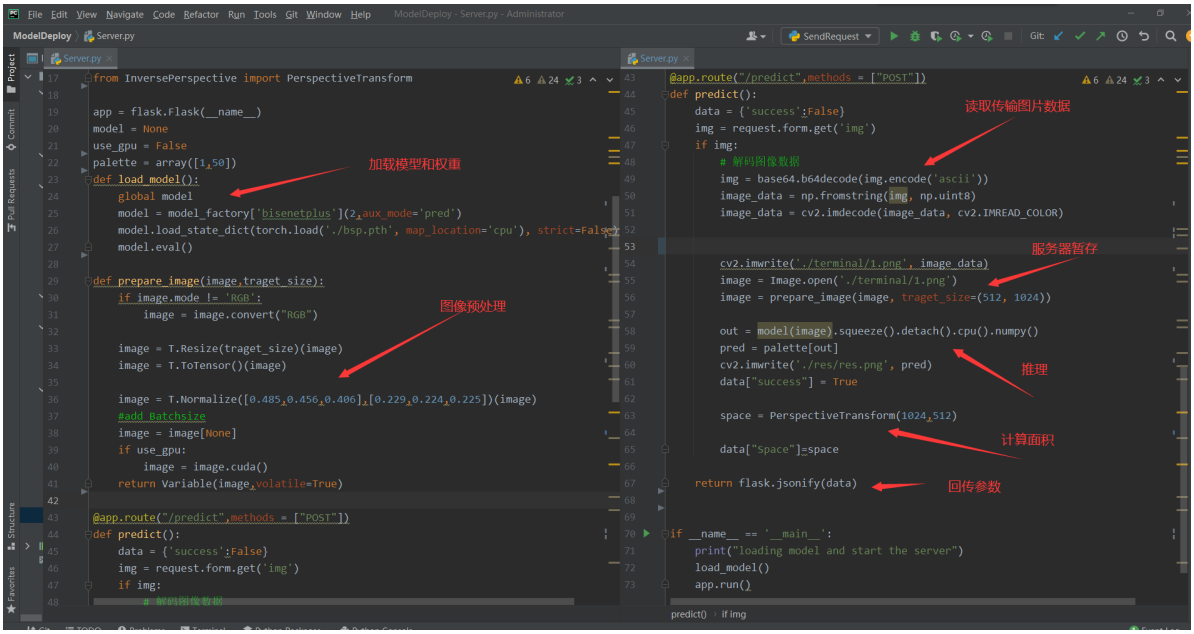
Area estimation

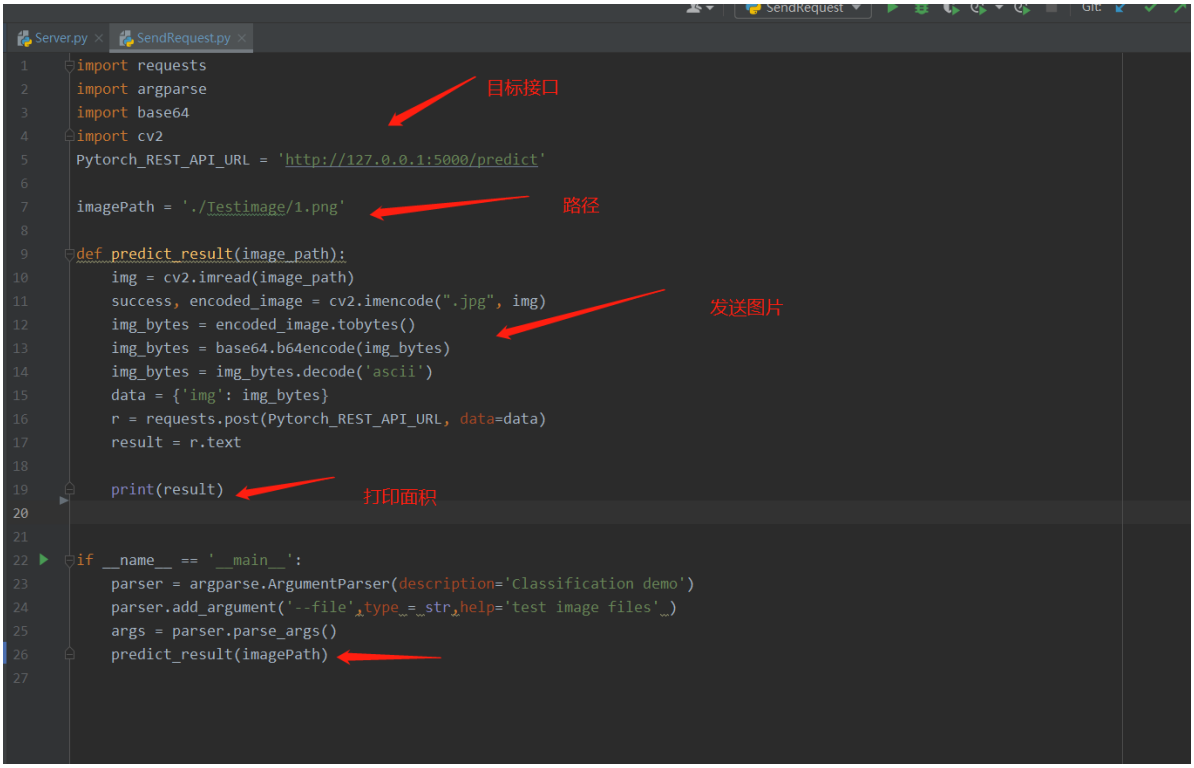
文件结构



Server.py

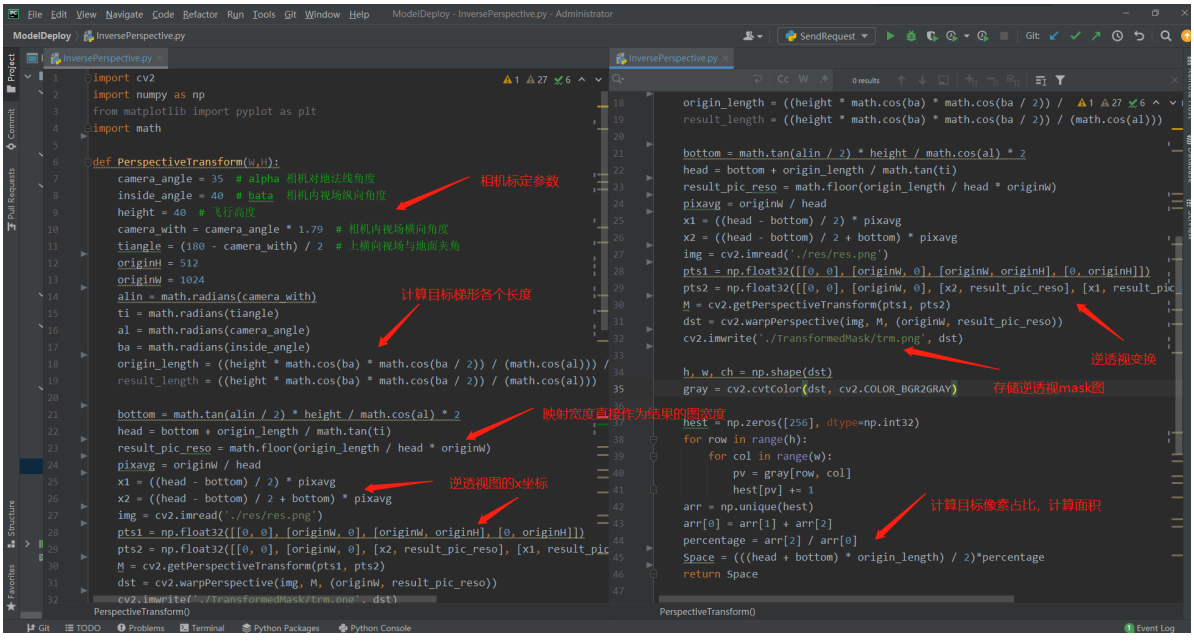


SendRequest.py




```python
import requests
import argparse
import base64
import cv2
Pytorch_REST_API_URL = 'http://127.0.0.1:5000/predict'

imagePath = './Testimage/1.png'

def predict_result(image_path):
    img = cv2.imread(image_path)
    success, encoded_image = cv2.imencode(".jpg", img)
    img_bytes = encoded_image.tobytes()
    img_bytes = base64.b64encode(img_bytes)
    img_bytes = img_bytes.decode('ascii')
    data = {'img': img_bytes}
    r = requests.post(Pytorch_REST_API_URL, data=data)
    result = r.text

    print(result)


if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Classification demo')
    parser.add_argument('--file', type=str, help='test image files')
    args = parser.parse_args()
    predict_result(imagePath)
```

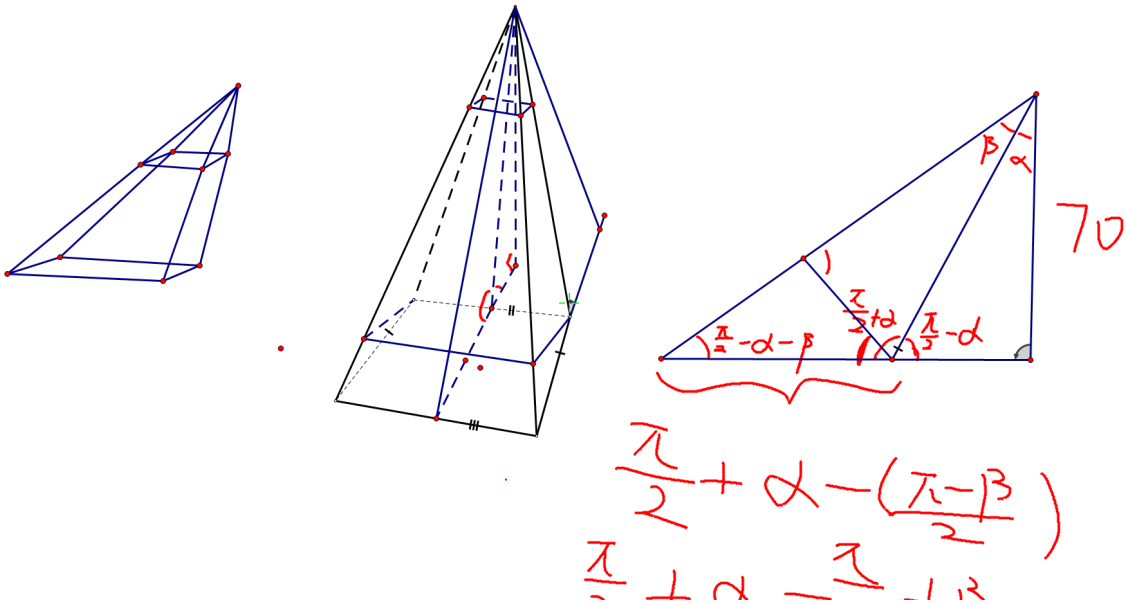
InversePerspective.py



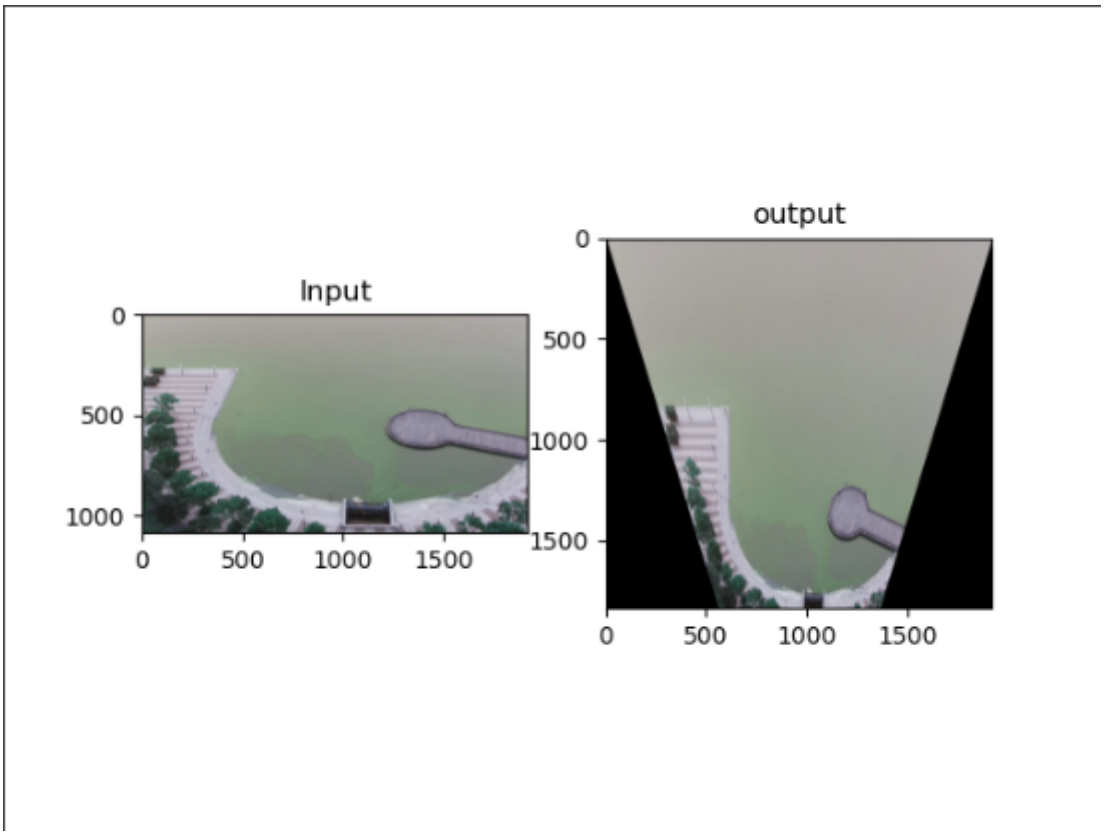

```python
import cv2
import numpy as np
from matplotlib import pyplot as plt
import math

def PerspectiveTransform(w,h):
    camera_angle = 35  # alpha 相机对地法线角度
    inside_angle = 40  # beta  相机内视场纵向角度
    height = 40  # 飞行高度
    camera_with = camera_angle * 1.79  # 相机内视场横向角度
    tiangle = (180 - camera_with) / 2  # 上横向视场与地面夹角
    originH = 512
    originW = 1024
    alin = math.radians(camera_with)
    ti = math.radians(tiangle)
    al = math.radians(camera_angle)
    ba = math.radians(inside_angle)
    origin_length = ((height * math.cos(ba) * math.cos(ba / 2)) / (math.cos(al))) /
    result_length = ((height * math.cos(ba) * math.cos(ba / 2)) / (math.cos(al)))

    bottom = math.tan(alin / 2) * height / math.cos(al) * 2
    head = bottom + origin_length / math.tan(ti)
    result_pic_reso = math.floor(origin_length / head * originW)
    pixavg = originW / head
    x1 = ((head - bottom) / 2) * pixavg
    x2 = ((head - bottom) / 2 + bottom) * pixavg
    img = cv2.imread('./res/res.png')
    pts1 = np.float32([[0, 0], [originW, 0], [originW, originH], [0, originH]])
    pts2 = np.float32([[0, 0], [originW, 0], [x2, result_pic_reso], [x1, result_pic
    M = cv2.getPerspectiveTransform(pts1, pts2)
    dst = cv2.warpPerspective(img, M, (originW, result_pic_reso))
    cv2.imwrite('./TransformedMask/trm.one', dst)
PerspectiveTransform()
```

```python
    origin_length = ((height * math.cos(ba) * math.cos(ba / 2)) /
    result_length = ((height * math.cos(ba) * math.cos(ba / 2)) / (math.cos(al)))

    bottom = math.tan(alin / 2) * height / math.cos(al) * 2
    head = bottom + origin_length / math.tan(ti)
    result_pic_reso = math.floor(origin_length / head * originW)
    pixavg = originW / head
    x1 = ((head - bottom) / 2) * pixavg
    x2 = ((head - bottom) / 2 + bottom) * pixavg
    img = cv2.imread('./res/res.png')
    pts1 = np.float32([[0, 0], [originW, 0], [originW, originH], [0, originH]])
    pts2 = np.float32([[0, 0], [originW, 0], [x2, result_pic_reso], [x1, result_pic
    M = cv2.getPerspectiveTransform(pts1, pts2)
    dst = cv2.warpPerspective(img, M, (originW, result_pic_reso))
    cv2.imwrite('./TransformedMask/trm.png', dst)

    h, w, ch = np.shape(dst)
    gray = cv2.cvtColor(dst, cv2.COLOR_BGR2GRAY)
    hest = np.zeros([256], dtype=np.int32)
    for row in range(h):
        for col in range(w):
            pv = gray[row, col]
            hest[pv] += 1
    arr = np.unique(hest)
    arr[0] = arr[1] + arr[2]
    percentage = arr[2] / arr[0]
    Space = (((head + bottom) * origin_length) / 2)*percentage
    return Space
PerspectiveTransform()
```

示意图



逆透视



估测

```
PerspectiveTransform
D:\ICS\anaconda\python.exe D:/ICS/ModelDeploy/SpaceLab/PerspectiveTransform.py
Space= 13685.25439126393
Cany= 8140.8375821933805

Process finished with exit code 0
```


大龙燚
湖滨新天地
停车场
江南首席
望湖阁码头
喜士多
湖滨楼
鹤家铁板烧
日本料理
王品牛排
望海阁游船
服务中心
地下停车场
圆融雕塑
起点
总长：357米
鳄鱼雕塑
73米
面积：14022.54米2
钓鱼翁雕塑
209米
酒桶狗雕塑
书包狗雕塑
打扑克雕塑
27栋
15栋