2018 年《机器视觉与应用》大作业

本年度《机器视觉与应用》大作业有三题备选：第一题是图像融合，旨在让同学们充分掌握 OpenCV 的使用；第二题是视觉里程计（Visual Odometry），旨在让同学们掌握如何通过分析图像序列估计相机（机器人）位姿，引起同学们对 SLAM（同步定位与建图）问题的兴趣；第三题是用机器学习方法进行物体识别，旨在引起同学们对利用机器学习方法进行物体识别的兴趣。同学们根据自己的兴趣自行选择一个题目，如果两题或三题都做了，可以适当加分，注意不能照搬网上开源程序。

与作业有关的任何问题，可以在微信群中联系助教。

课程助教：张昊若

2018 年 5 月 14 日

第一题 图像融合

http://eric-yuan.me/poisson-blending-2/

http://codepad.org/ANqtikKR

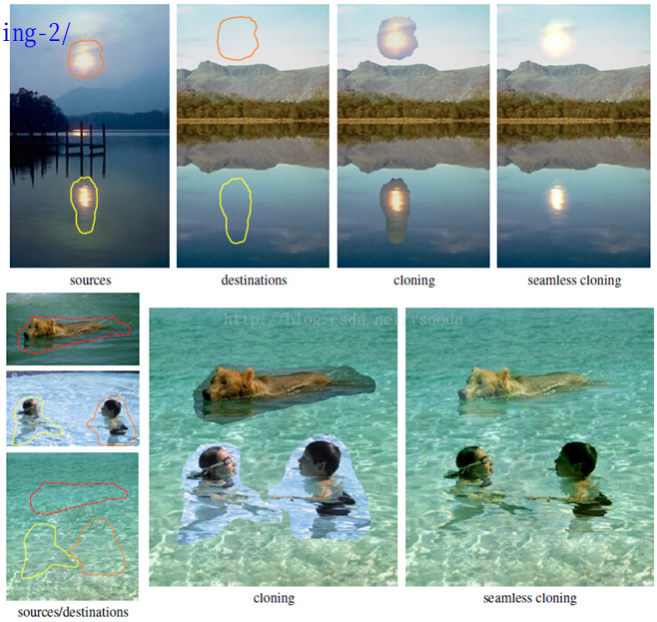

图 1 图像融合示意图

作业要求：基于 OpenCV 实现图像融合领域一种非常经典的算法——泊松克隆，论文题目是《Poisson Image Editing》，要求不可以使用 OpenCV 中的 seamless_cloning 类或 Sobel 函数等高级 API。

可以参考网上相关博客，例如：https://blog.csdn.net/hjimce/article/details/45716603。

同学可以自己选择图片进行处理，需要选择两组图片，每组有三张图片，分别是 source，mask，destination。如最上面那张图，source 就是熊和孩子，mask 是这些前景的掩膜，destination 是背景，这里是水池。

算法基本步骤与每步得分（按**步骤**给分）：

1. 图片读取（10 分）

主要考察大家对 OpenCV 安装、编译的掌握程度，可以参考助教上课的 PPT 进行配置，可以在 Ubuntu 系统中配置。

2. 求取图像的梯度场（20 分）

可以通过差分的方式求取图像的梯度场，包括 source 图像和 destination 图像。

3. 求解融合图像的散度（20 分）

计算融合后图像每个像素的散度值。

4. 泊松重建（40 分）

根据已知的图像散度和边界条件，建立泊松方程，并求解。

5. 图像融合效果改进（10 分）*

如果有其他改进图像融合效果的措施，可以一并在作业文档内说明。

提交材料：

1. Visual Studio 工程（包含 cpp，项目配置等），使用 Ubuntu 系统的同学则提交 cpp 与 cmake 配置文件
2. 说明文档：
   - 图像融合效果截图；
   - 程序使用方法：OpenCV 版本、运行时图片位置等；
   - 算法介绍，对应前面每一步骤，便于后续评分；如果有算法改进措施，也请说明。

上述材料请放置在一个压缩文件内，命名为“**图像融合-学号-姓名.rar**”。

第二题 视觉里程计

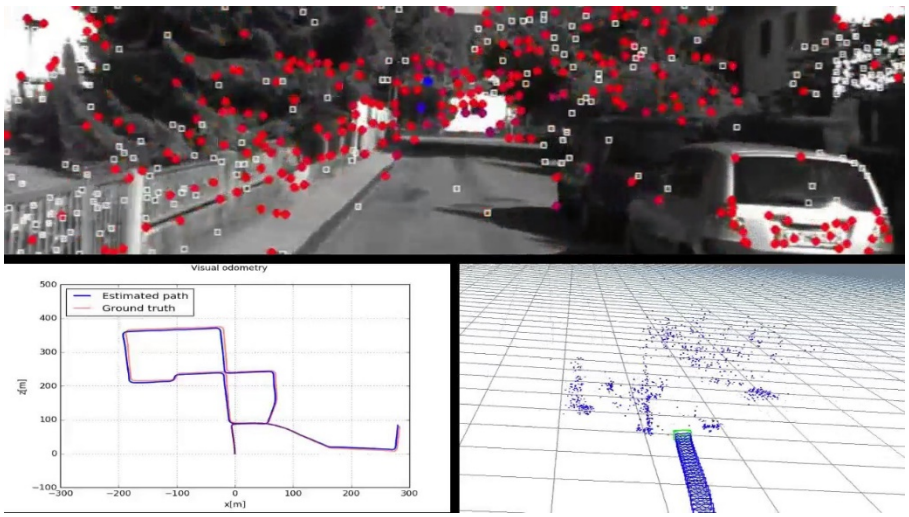

图 2 视觉里程计示意图

视觉里程计(VO)是一种利用连续的图像序列来估计相机或机器人移动距离的方法，也被称为 Visual SLAM 问题的前端，是移动机器人定位导航领域中的关键技术之一。

作业要求：基于 OpenCV 初步实现一个简单的视觉里程计，相关理论可以参考 Davide Scaramuzza 的论文“Visual Odometry: Part I - The First 30 Years and Fundamentals”和“Visual Odometry: Part II - Matching, Robustness, and Applications”，可以参考网上相关书籍或博客。

例如：《视觉 SLAM 十四讲》，http://avisingh599.github.io/vision/monocular-vo/，http://www.fengbing.net/2015/07/26/%E4%B8%80%E4%B8%AA%E7%AE%80%E5%8D%95%E7%9A%84%E8%A7%86%E8%A7%89%E9%87%8C%E7%A8%8B%E8%AE%A1%E5%AE%9E%E7%8E%B01/

同学们可以使用 KITTI 公共数据集来对自己的算法效率和精度进行评估，数据集的地址为：http://www.cvlibs.net/datasets/kitti/eval_odometry.

算法基本步骤与每步得分（按**步骤**给分）：

1. 图片读取（10 分）

主要考察大家对 OpenCV 安装、编译的掌握程度，可以参考助教上课的 PPT 进行配置，可以在 Ubuntu 系统中配置。

2. 特征提取（20 分）

主要考察大家对一些局部特征描述子提取方法的了解，这里一般提取 FAST 关键点，计算 ORB 特征。

3. 特征匹配或跟踪（50 分）

大家可以通过 ORB 特征匹配或者 KLT 关键点跟踪的方式对图像连续帧进行处理，这里包含了通过本质矩阵计算相机位姿变换，单目视觉初始化等问题。

4. 生成相机轨迹与 Ground Truth 比较（10 分）

最后将视觉里程计的计算结果与真实轨迹进行对比，将结果画出。

5. 优化（10 分）*

对视觉里程计有改进，包括 Bundle Adjustment 迭代优化估计相机位姿变换，局部地图等。

提交材料：

1. Visual Studio 工程（包含 cpp，项目配置等），使用 Ubuntu 系统的同学则提交 cpp 与 cmake 配置文件
2. 说明文档：
   - 视觉里程计效果截图；
   - 程序使用方法：OpenCV 版本、运行时图片位置等；
   - 算法介绍，对应前面每一步骤，便于后续评分。

上述材料请放置在一个压缩文件内，命名为“**视觉里程计-学号-姓名.rar**”。

第三题 物体识别

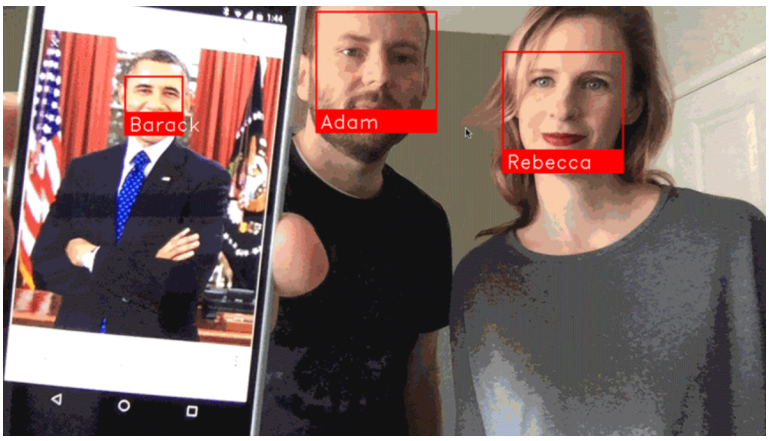

图 3 人脸识别示意图

本题主要通过人脸识别这一应用让大家充分了解如何使用机器学习的方法进行物体识别。

作业要求：使用 MATLAB 完成简单的如图 3 所示的人脸识别应用，可以参考 Python 版本的 Face Recognition API， https://github.com/ageitgey/face_recognition，不能照搬源码，不能使用一些深度学习框架(Caffe, TensorFlow 等)的高级 API。

算法基本步骤与每步得分（按**步骤**给分）：

1. 图片操作（10 分）

这部分主要是考察 MATLAB 中的一些基本图像操作，以及实时视频流的处理。

2. 编写程序完成人脸识别框架（60 分）

这部分大家采用的机器学习方法会有不同，要求不能使用 MATLAB 高级 API 或工具箱。

3. 完成三种 API 接口（30 分）

第一种：Find Faces in Pictures，如图 4 所示；

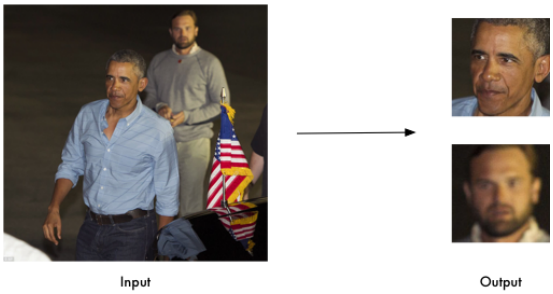

图 4 Find Faces in Pictures

第二种：Identify Faces in Pictures，如图 5 所示；

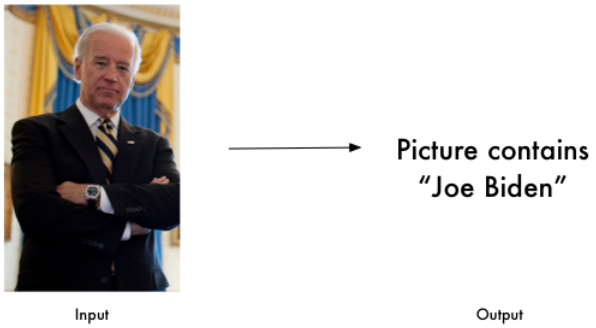

图 5 Identify Faces in Pictures

第三种：Real-time Face Recognition，要求实时视频流可以识别已训练的几张（>2）人脸，如图 3 所示。

提交材料：

1. 所有代码，包含训练和测试图片，可以直接运行；
2. 说明文档：
   - 人脸识别效果截图（三种接口）；
   - 程序使用方法：比如训练步骤，图片位置等等；
   - 算法介绍，对应前面每一步骤，便于后续评分。

上述材料请放置在一个压缩文件内，命名为“**物体识别-学号-姓名.rar**”。