

投影字典对学习 (Dictionary Pair Learning)

Julia 学术代码包开发

曲晓峰

香港理工大学 电子计算学系
人体生物特征识别研究中心

二〇一六年五月一日



目录



① 我的工作

- 自我介绍
- 线扫描掌纹识别系统
- 门把手识别系统

② 投影字典对学习 Projective Dictionary Pair Learning (DPL)

- DPL
- Julia 代码移植



自我介绍

- 曲晓峰，香港理工大学博士生
- 研究方向：人体生物特征识别/机器视觉/嵌入式系统/图像处理/模式识别/机器学习。
- 近期的工作（都发表在 IEEE Transactions on SMC: Systems）
 - ① 线扫描掌纹采集系统
A Novel Line-Scan Palmprint Acquisition System
 - ② 门把手人手识别系统
Door Knob Hand Recognition System
- 从 2015 年 3 月开始学习 Julia
- 现在维护 ProjectiveDictionaryPairLearning.jl
- 希望这个包有帮助；请大家帮忙优化改进。



线扫描掌纹识别系统

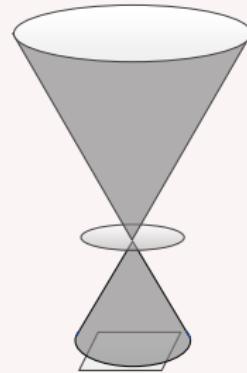
- <http://www.linescanpalmpoint.ml>
- 基于 CIS 线阵传感器的掌纹识别系统
- 体积小；扫描速度自适应；识别性能可靠（等错误率 0.048%）。





线扫描掌纹识别系统

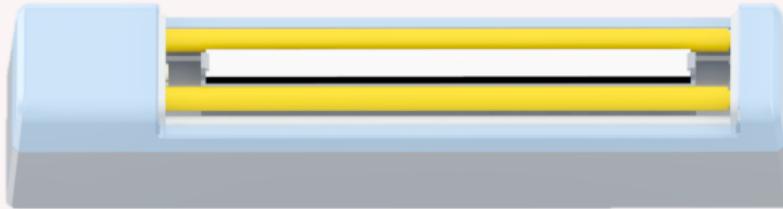
- <http://www.linescanpalmpoint.ml>
- 基于 CIS 线阵传感器的掌纹识别系统
- 体积小；扫描速度自适应；识别性能可靠（等错误率 0.048%）。





线扫描掌纹识别系统

- <http://www.linescanpalmpoint.ml>
- 基于 CIS 线阵传感器的掌纹识别系统
- 体积小；扫描速度自适应；识别性能可靠（等错误率 0.048%）。





门把手识别系统

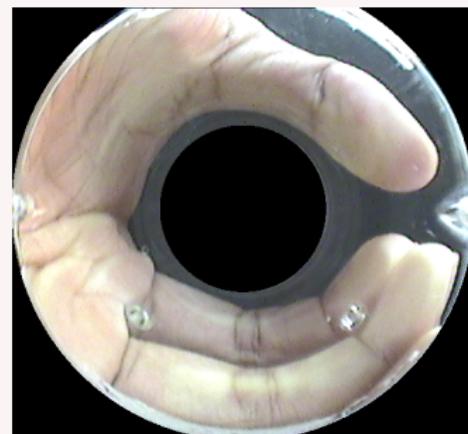
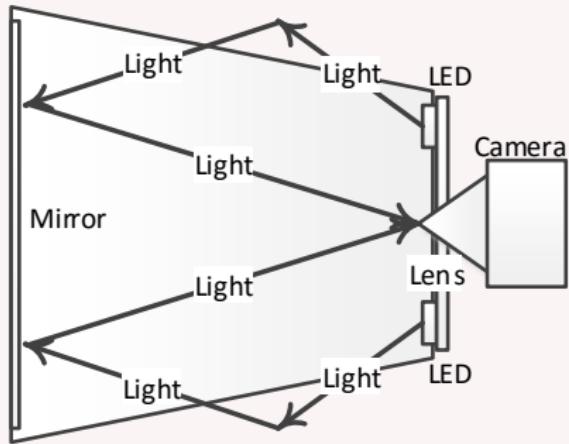
- <http://www.doorknob.ml>
- 为普通人设计的，模拟门把手的人手识别系统





门把手识别系统

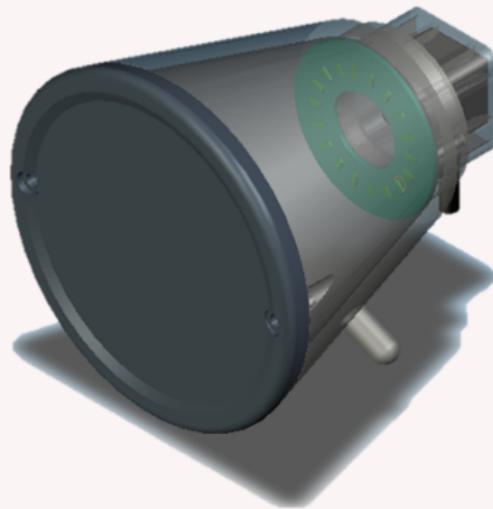
- 结合门把手的外形设计与人体生物特征识别的功能
- “功能与形式的统一”。
- 人体工学优化的消费级生物特征识别





门把手识别系统

性能可靠。优于常见的指纹识别和人脸识别（识别率高于 99.5%；等错误率低于 0.15%）。



Elsevier 的期刊 Pattern Recognition (模式识别) 近期 (七月十五日截止) 有一个 Special Issue on Ubiquitous Biometrics (无所不在的生物特征识别)

字典学习



- 字典学习来源于傅立叶变换和小波变换
- 字典学习是用来学习一个字典，用字典表达和重建信号
- 傅立叶变换中，字典是余弦函数；小波变换中，字典是小波基函数
- 字典学习，是从训练数据中，根据约束学习一个最优字典
- 约束的设置，决定了字典的性能
- 字典学习包括 synthesis 字典和 analysis 字典
- synthesis 字典： $\min_{D,A} \|X - DA\|_F^2 + \lambda \|A\|_0$
- analysis 字典： $\min_{\Omega,X} \|X - \hat{X}\|_F^2 + \lambda \|\Omega \hat{X}\|_0$

DPL 设计



- 在训练中同时学习分析字典和综合字典。
- 保证效率，不用稀疏编码 (ℓ_0 和 ℓ_1 -范数)。
- 保证区分 (discrimination) 能力。这里是保证本类重建的误差最小化。

DPL Learning Function

$$P^*, D^* = \arg \min_{P, D} \sum_{k=1}^K \|X_k - D_k P_k X_k\|_F^2 + \lambda \|P_k \bar{X}_k\|_F^2, \text{ s.t. } \|\mathbf{d}_i\|_2^2 \leq 1$$

优化 I



$$\{P^*, A^*, D^*\} = \arg \min_{P, A, D} \sum_{k=1}^K \|X_k - D_k A_k\|_F^2 + \tau \|P_k X_k - A_k\|_F^2 + \lambda \|P_k \bar{X}_k\|_F^2,$$

s.t. $\|\mathbf{d}_i\|_2^2 \leq 1$

① 固定 D and P ，更新 A

$$A^* = \arg \min_A \sum_{k=1}^K \|X_k - D_k A_k\|_F^2 + \tau \|P_k X_k - A_k\|_F^2$$

有闭合解

$$A_k^* = (D_k^T D_k + \tau I)^{-1} (\tau P_k X_k + D_k^T X_k)$$



优化 II

② 固定 A , 更新 D 和 P

$$\begin{cases} P^* = \arg \min_P \sum_{k=1}^K \tau \|P_k X_k - A_k\|_F^2 + \lambda \|P_k \bar{X}_k\|_F^2; \\ D^* = \arg \min_D \sum_{k=1}^K \|X_k - D_k A_k\|_F^2, \text{ s.t. } \|\mathbf{d}_i\|_2^2 \leq 1 \end{cases}$$

③ 迭代至收敛 (或一定次数)

为什么用 Julia



在开发之前：

- 论文中公式的希腊字母可以用作程序里的变量名。
易于对应论文中公式的概念和代码中的变量。
- 原生的线性代数方面的运算支持。与 MATLAB 类似。
- 免费自由的开发和运行环境，利于成果推广。
- 可以实现敏捷的开发流程 (unit test , regression test ,
continuous integration)
- 学习 Julia
- Julia 运算速度快 (T_T)



为什么用 Julia

完成这个代码包的开发之后：

- 原生的单元测试和测试驱动开发支持。(MATLAB 槽点)
- 不用每次做点什么都要 `import numpy as np`。依赖关系清晰 (REQUIRE)，自动安装，且受版本管理器管理。(Python 槽点)
- 分布式包管理器，便于代码进入官方代码库，易于下载使用。

```
Pkg.add("ProjectiveDictionaryPairLearning")
using ProjectiveDictionaryPairLearning
dpldemo()
```

- 每次修改代码，推送到 GitHub Repo 之后可以触发 Travis-CI 自动回归测试。(可以拿这种方式做实验？)



移植过程

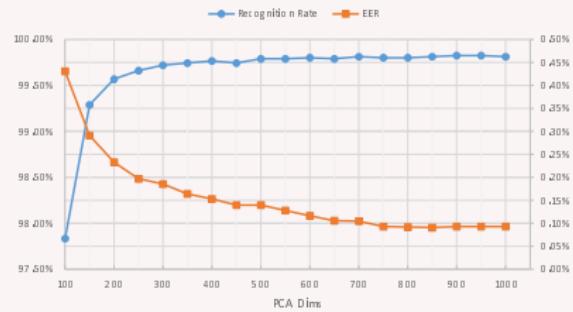
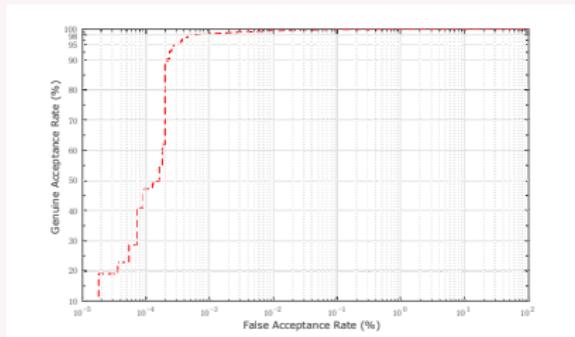
- 根据代码功能，确定程序 package 结构。（移植代码参考 MATLAB 版本）
- /src - 源代码；/test - 测试。
- 确定函数的输入输出，撰写测试用例。
- 使用 “MAT.jl” 包读写 MATLAB 的 MAT 数据文件。充分利用 legacy 数据。
- 重写函数，行行对译，Jupyter 调试，函数基本没有代价。
- 使用持续集成网站进行回归测试，保证功能稳定。
- 打包，并提交到 “MetaData”。
- 优化性能。@profile，@time。
- 维护更新：Julia 版本更新之后的语法更新，优化，注释说明。



识别性能评估

分类方法 $dissimilarity(y) = \|y - D_i P_i y\|_2$

- 在门把手识别系统中，应用 Projective Dictionary Pair Learning
- 200 个人；400 只手；12000 张图片的比对中
- 识别率最高达到 99.814%；等错误率最低达到 0.0910%.





运算性能表现

基于 Projective Dictionary Pair Learning 一文中使用的有遮挡的人脸数据库。该代码与程序包一同发布，可以在自己的环境中重复测试。

```
using ProjectiveDictionaryPairLearning; dpldemo();
```

- Julia 运行速度

The running time for DPL training is : 4.39 s

The running time for DPL testing is : 0.27 s

Recognition rate for DPL is : 97.579%

- MATLAB 代码¹运行速度

The running time for DPL training is : 3.12 s

The running time for DPL testing is : 0.17 s

Recognition rate for DPL is : 0.976

¹可以到 Prof. Lei Zhang 的主页下载:

http://www4.comp.polyu.edu.hk/~cslzhang/code/DPL_NIPS14.zip



过程中遇到的问题

- Julia 的速度并不天然比 MATLAB 快。需要手动进一步的优化。
- 对于 Julia 现有优化工具不熟悉。从头写了整个包。
- 名字起得不好。
 - ① “DPL.jl”
 - ② “ProjectiveDictionaryPairLearning.jl”
- Julia 版本变动剧烈；相关依赖不稳定
 - ① v0.3.7 and NumericalExtension.jl
 - ② From v0.4.x to v0.5.0-dev
- Julia 还缺少计算机视觉方面的库和组织，缺少统一的接口实践（例如 Python 的 scikit-learn 和 scikit-image），难以积累和拓展。

总结和对于在科研中使用 Julia 的建议



- Julia 其实已经足够成熟到可以用来进行学术代码的开发
- 便捷的线性运算；敏捷的开发模式；快速的运行速度。
- 与 Python 相比主要优势是原生的线性运算
- 与 MATLAB 相比主要优势是现代化的开发模式
- 适合科研结果的推广
- 但要注意的是，保证单元测试与回归测试的覆盖，以保证程序包的健壮性
- 还缺乏一些底层的依赖，生态环境不够繁荣
- 适合轻依赖、重数值运算的科研代码