DIP Homework Final README

文件说明

文件名	文件类型	说明
`run_demo.m`	Script	执行测试脚本，输入图片路径在 `generate_init_data` 中设置
`generate_init_data.m`	Script	生成输入图片I、模糊图B、模糊核K、噪点图N和降噪图Nd等数据
`test_denoise.m`	Script	测试降噪功能（仅测试一个通道）
`test_matrix_vector_mul.m`	Script	测试图片的Matrix形式的转换，并测试转换后的矩阵乘法
`generate_blurimg.m`	Function	为输入图片生成模糊图和噪点图
`random_kernel.m`	Function	生成随机的Blur Kernel
`write_kernel.m`	Function	将Kernel以图片形式保存到磁盘
`denoise.m`	Function	为输入图片降噪
`denoise_preprocess.m`	Function	降噪算法预处理数据
`denoise_channel.m`	Function	为RGB图的单一通道（或灰度图）进行降噪，根据参数会调用两种不同的降噪算法
`call_denoi_bls_gsm.m`	Function	BLS-GSM降噪算法的调用函数，在其中进行了一些参数的设置
`FAST_NLM_II.m`	Function	使用的降噪算法之一——Fast NLM
`deblur.m`	Function	根据输入的降噪图片和模糊图片两者对模糊图片进行去模糊
`kernel_estimation.m`	Function	根据传入图片和参数估算Blur Kernel
`mat2kmat.m`	Function	将普通图片矩阵转换为论文中所提到的矩阵形式

文件名	文件类型	说明
`mat2vec.m`	Function	将普通图片矩阵转换为论文中所提到的向量形式
`deconv.m`	Function	封装算法实现的各类反卷积算法
`deconvglucy.m`	Function	Gain-Controlled Richardson-Lucy算法实现
`Gscale.m`	Function	生成Gaussian Pyramid
`compute_Igain_map.m`	Function	生成gain map
`jbfilter2.m`	Function	Joint/Cross Bilateral Filtering算法
`fftconv2.m`	Function	用快速傅里叶变换实现的2d卷积
`/denoise`	Directory	BLS-GSM降噪算法工具箱
`/l1_ls_matlab`	Directory	L1-LS问题求解工具箱
`/images`	Directory	存放输入和输出的图片文件

使用方法

配置

a. Denoise 工具箱的配置

将下述文件夹按照给定顺序加到MATLAB path中：

```
denoise\denoising_subprograms (at the top)
denoise\matlabPyrTools\MEX
denoise\matlabPyrTools
denoise\Simoncelli_PyrTools
denoise\Added_PyrTools
```

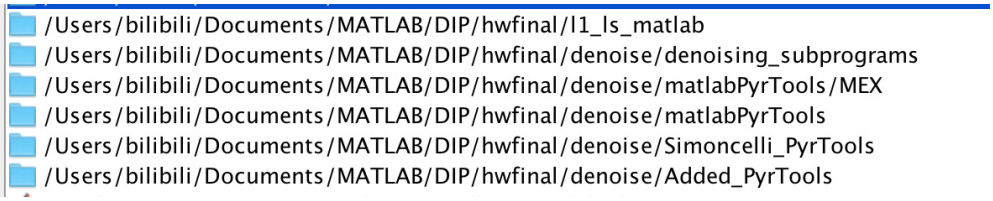
*如果使用Portilla提供的原版代码，需要用 `BLS-GSMmod.zip` 覆盖原有内容，具体方法参见这里。

b. L1-LS 工具箱的配置

向MATLAB path中加入：

```
l1_ls_matlab
```

最终顺序：



使用

a. 使用现有的测试脚本

1. 在`generate_init_data.m`第6行设置好输入的图片文件（原图）
2. 执行`run_demo.m`脚本，会依次执行每一种算法
3. 执行结果保存在`/images`目录下

b. 具体函数调用流程

1. 输入：模糊图片 B ，噪点图片 N
2. 调用`denoise`函数为 N 降噪，得到 N_D

```
Nd = denoise( N );
```

3. 调用`deblur`函数为 B 进行去模糊

```
[ I, K ] = deblur( Nd, B, unikernel, deconvmode, verbose );
```

其中：

- `unikernel`为真时表示对rgb图的每一个通道估算一个统一的Kernel
- `deconvmode`表示反卷积使用的算法，可以为
  - `'reg'`： Matlab自带的Regularized Filter方法（`deconvreg`）
  - `'lucy'`： Matlab自带的Richardson-Lucy算法（`deconvlucy`）
  - `'resRL'`： Residual RL算法，调用自己实现的`deconvglucy`
  - `'gcRL'`： 增益控制RL算法，调用`deconvglucy`
  - `'detailedRL'`： 增加细节的`gcRL`算法，会同时计算`resRL`和`gcRL`

4. 上一步中输出的 I 即为所求的去模糊后的结果， K 为估算的Blur Kernel