# 项目规范文档 - DAS数据添加噪音、降噪一体化处理系统

版本：1.00

最后更新：2025年8月18日07:56:58

作者：yulania

# 项目概述

本系统实现对DAS数据添加噪音、降噪并绘图分析的一体化处理：

1. 读取mat文件形式的DAS数据
2. 在读取的DAS数据中添加可控的高斯噪音和脉冲噪音
3. 对噪音DAS数据用指定降噪方法进行降噪
4. 可实现的降噪方式有：高斯滤波、移动平均滤波、中值滤波、小波降噪、双边滤波
5. 对降噪前后的各指标进行计算：PSNR，SSIM，相对误差e
6. 绘制热力图：原始图像、添加噪音后的图像、降噪后的图像，图像上有噪声的参数和降

噪指标参数

# 模块接口规范表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 模块 | 文件路径 | 工厂类 | 主要方法 | 输入参数 | 返回对象 | 依赖路径 |
| 数据加载模块 | data\_loader/ | DataLoaderFactory | create\_loader(file\_type, \*\*kwargs) | file\_type: str, kwargs: dict | 具体数据加载器实例 | matlab\_loader.py |
| 噪声生成模块 | noise\_generator/ | NoiseGeneratorFactory | create\_generator(noise\_type, \*\*kwargs) | noise\_type: str, kwargs: dict | 具体噪声生成器实例 | gaussian\_noise.py, impulse\_noise.py |
| 降噪模块 | denoising/ | DenoisingFactory | create\_denoiser(denoiser\_type, \*\*kwargs) | denoiser\_type: str, kwargs: dict | 具体降噪器实例 | gaussian\_filter.py, moving\_average\_filter.py, median\_filter.py, wavelet\_denoising.py, bilateral\_filter.py |
| 指标计算模块 | metrics/ | MetricsFactory | create\_calculator(metric\_type, \*\*kwargs) | metric\_type: str, kwargs: dict | 具体指标计算器实例 | ssim\_calculator.py, psnr\_calculator.py,  r\_error\_calculator.py |
| 可视化模块 | visualization/ | VisualizationFactory | create\_visualizer(visualizer\_type, \*\*kwargs) | visualizer\_type: str, kwargs: dict | 具体可视化器实例 | heatmap\_visualizer.py |

# 各模块详细规范

1.数据加载模块

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **组件** | **文件路径** | **类名** | **方法名** | **输入参数** | **返回对象** | **功能说明** |
| 工厂 | data\_loader/data\_loader\_factory.py | DataLoaderFactory | create\_loader | file\_type, \*\*kwargs | 数据加载器实例 | 统一创建数据加载器 |
| MATLAB加载器 | data\_loader/matlab\_loader.py | MatlabLoader | load\_data | file\_path, variable\_name | (data, metadata) | 加载MATLAB文件数据 |
| MATLAB加载器 | data\_loader/matlab\_loader.py | MatlabLoader | get\_data\_info | 无 | dict | 获取数据基本信息 |

1. 噪声生成器详细规范

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **组件** | **文件路径** | **类名** | **方法名** | **输入参数** | **返回对象** | **功能说明** |
| 工厂 | noise\_generator/noise\_factory.py | NoiseGeneratorFactory | create\_generator | noise\_type, \*\*kwargs | 噪声生成器实例 | 统一创建噪声生成器 |
| 高斯噪声 | noise\_generator/gaussian\_noise.py | GaussianNoiseGenerator | add\_noise | data, snr\_db | noisy\_data | 添加高斯白噪声 |
| 高斯噪音 | noise\_generator/gaussian\_noise.py | GaussianNoiseGenerator | \_calculate\_noise\_std\_from\_snr | data, snr\_db | noise\_std | 根据信噪比计算噪声标准差 |
| 高斯噪声 | noise\_generator/gaussian\_noise.py | GaussianNoiseGenerator | generate\_noise\_only | shape | noise | 仅生成高斯噪声 |
| 脉冲噪声 | noise\_generator/impulse\_noise.py | ImpulseNoiseGenerator | add\_noise | data, noise\_ratio, salt\_ratio | noisy\_data | 添加脉冲噪声 |
| 脉冲噪声 | noise\_generator/impulse\_noise.py | ImpulseNoiseGenerator | generate\_mask | shape, noise\_ratio | mask | 生成噪声掩码 |

1. 降噪器详细规范

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **组件** | **文件路径** | **类名** | **方法名** | **输入参数** | **返回对象** | **功能说明** |
| 工厂 | denoising/denoising\_factory.py | DenoisingFactory | create\_denoiser | denoiser\_type, \*\*kwargs | 降噪器实例 | 统一创建降噪器 |
| 高斯滤波 | denoising/gaussian\_filter.py | GaussianFilter | denoise | data, sigma | denoised\_data | 高斯滤波降噪 |
| 高斯滤波 | denoising/gaussian\_filter.py | GaussianFilter | apply\_separable\_filter | data, sigma\_row, sigma\_col | denoised\_data | 可分离高斯滤波 |
| 移动平均 | denoising/moving\_average\_filter.py | MovingAverageFilter | denoise | data, window\_size, axis | denoised\_data | 移动平均滤波降噪 |
| 移动平均 | denoising/moving\_average\_filter.py | MovingAverageFilter | apply\_directional\_filter | data, time\_window, distance\_window | denoised\_data | 方向性移动平均滤波 |
| 中值滤波 | denoising/median\_filter.py | MedianFilter | denoise | data, size | denoised\_data | 中值滤波降噪 |
| 中值滤波 | denoising/median\_filter.py | MedianFilter | apply\_directional\_median\_filter | data, size\_time, size\_distance | denoised\_data | 方向性中值滤波 |
| 小波降噪 | denoising/wavelet\_denoising.py | WaveletDenoising | denoise | data, wavelet, level, threshold\_mode | denoised\_data | 小波变换降噪 |
| 小波降噪 | denoising/wavelet\_denoising.py | WaveletDenoising | apply\_visu\_shrink | data, wavelet, level | denoised\_data | VisuShrink方法降噪 |
| 小波降噪 | denoising/wavelet\_denoising.py | WaveletDenoising | get\_wavelet\_coefficients | data, wavelet, level | coeffs | 获取小波系数 |
| 双边滤波 | denoising/bilateral\_filter.py | BilateralFilter | denoise | data, spatial\_sigma, intensity\_sigma, window\_size | denoised\_data | 双边滤波降噪 |
| 双边滤波 | denoising/bilateral\_filter.py | BilateralFilter | apply\_separable\_bilateral\_filter | data, spatial\_sigma, intensity\_sigma， | denoised\_data | 可分离双边滤波 |

1. 可视化器详细规范

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **组件** | **文件路径** | **类名** | **方法名** | **输入参数** | **返回对象** | **功能说明** |
| 热力图可视化 | visualization/heatmap\_visualizer.py | HeatmapVisualizer | plot\_heatmap | data, time\_axis, distance\_axis, title, xlabel, ylabel, vmin, vmax, colorbar, colorbar\_label, show, save\_path, \*\*kwargs | fig, ax | 绘制数据热力图 |
| 热力图可视化 | visualization/heatmap\_visualizer.py | HeatmapVisualizer | plot\_heatmap\_with\_stats | data, time\_axis, distance\_axis, title, show\_stats, \*\*kwargs | fig, ax | 绘制带统计信息的热力图 |
| 热力图可视化 | visualization/heatmap\_visualizer.py | HeatmapVisualizer | plot\_multiple\_heatmaps | data\_list, titles, time\_axis, distance\_axis, figsize, \*\*kwargs | fig, axes\_list | 绘制多个热力图进行比较 |
| 热力图可视化 | visualization/heatmap\_visualizer.py | HeatmapVisualizer | plot\_heatmap\_contour | data, time\_axis, distance\_axis, title, contour\_levels, \*\*kwargs | fig, ax | 绘制带等高线的热力图 |

1. 指标计算器详细规范

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **组件** | **文件路径** | **类名** | **方法名** | **输入参数** | **返回对象** | **功能说明** |
| 工厂 | metrics/metrics\_factory.py | MetricsFactory | create\_calculator | metric\_type, \*\*kwargs | 指标计算器实例 | 统一创建指标计算器 |
| SSIM计算器 | metrics/ssim\_calculator.py | SSIMCalculator | calculate | original\_data, processed\_data, \*\*kwargs | ssim\_value: float | 计算SSIM值 |
| SSIM计算器 | metrics/ssim\_calculator.py | SSIMCalculator | calculate\_per\_channel | original\_data, processed\_data, \*\*kwargs | ssim\_values: list | 分别计算每个通道的SSIM值 |
| PSNR计算器 | metrics/psnr\_calculator.py | PSNRCalculator | calculate | original\_data, processed\_data | psnr\_value: float | 计算PSNR值 |
| 相对误差计算器 | metrics/r\_error\_calculator.py | RelativeErrorCalculator | calculate | original\_data, processed\_data | relative\_error: float | 计算相对误差值 |

根目录：C:\Users\17981\Desktop\科研\optic\_code\new\_denoise

├── data\_loader

│ ├── \_\_init\_\_.py

│ ├── data\_loader\_factory.py

│ └── matlab\_loader.py

├── denosing

│ ├── \_\_init\_\_.py

│ ├── bilateral\_filter.py

│ ├── denoising\_factory.py

│ ├── gaussian\_filter.py

│ ├── median\_filter.py

│ ├── moving\_average\_filter.py

│ └── wavelet\_denoising.py

├── docs

│ └── readme.docx

├── main.py

├── metrics

│ ├── \_\_init\_\_.py

│ ├── metrics\_factory.py

│ ├── psnr\_calculator.py

│ ├── r\_error\_calculator.py

│ └── ssim\_calculator.py

├── noise\_generator

│ ├── \_\_init\_\_.py

│ ├── gaussian\_noise.py

│ ├── impulse\_noise.py

│ └── noise\_factory.py

├── output

├── test

│ ├── generate\_directory\_tree.py

│ ├── noisy\_data\_heatmap.png

│ ├── noisy\_data\_with\_stats.png

│ ├── original\_data\_heatmap.png

│ ├── test\_data\_loader.py

│ ├── test\_noise\_generator.py

│ └── test\_noise\_visualization.py

├── utils

│ └── file\_utils.py

└── visualization

├── \_\_init\_\_.py

├── heatmap\_visualizer.py

└── visualization\_factory.py