# 历史记录

|  |  |  |
| --- | --- | --- |
| 版本 | 修订 | 责任人 |
| 20181020 | 起草 | 刘娇、刘焕云 |
| 20181123 | 添加新定义 | 刘焕云、刘娇、李松松 |

# 目 录

[历史记录 1](#_Toc1021563565)

[目 录 1](#_Toc1251042699)

[1 Matching初始化 2](#_Toc1860834624)

[2 Matching释放 2](#_Toc482541282)

[3 Matching注册 2](#_Toc1929785805)

[4 Matching匹配 3](#_Toc1437665760)

[5 分时段匹配 4](#_Toc444674620)

# 1 Matching初始化

初始化matching参数并读取已注册信息

mm = MatchingModule(MM\_CONFIG)

输入参数

MM\_CONFIG配置参数，dict类型，解释见表1

输出参数

-mm Matching库中申请的全局句柄

表1 MM\_CONFIG参数

|  |  |  |
| --- | --- | --- |
| 关键字 | 类型 | 含义 |
| 'cache\_dir' | string | Matching模块所需的缓存目录，必须具有读写权限 |
| 'feature\_dim' | int | 特征维度 |
| 'distance\_method' | string | 距离方法，支持euclidean, cosine |
| 'threshold' | float | 得分阈值 |
| 'use\_time\_match' | list | [可选参数]是否按照时间对比，默认为false，更多解释请参考<[5 分时段匹配](#_5 分时段匹配)> |
| 'time\_match\_interval' | list<int | [可选参数]当'use\_time\_match'为True时启用，默认为[-100, -10]，单位second |

# 2 Matching释放

释放在Matching初始化中申请内存

mm.free()

输入参数

无

输出参数

无

# 3 Matching注册

实现注册功能，更新注册列表

result= mm.register(feature)

输入参数

- feature matching所需输入特征信息。格式为字典型：

{ input\_id0:{‘camera\_id’:i,’time\_stamp’:\*\*:\*\*.\*\*,’feature’:[[]]},

input\_id1:{...},

…}

输出参数

-result 输出register相关信息。格式为字典型：

{ input\_id0:{‘camera\_id’:i,‘time\_stamp’:\*\*:\*\*.\*\*, ‘object\_id’:[x]}

input\_id1:{…},

...}

表2 注册函数输入输出参数

|  |  |  |
| --- | --- | --- |
| 关键字 | 类型 | 含义 |
| ‘input\_id\*’ | string | 输入目标编号 |
| 'camera\_id' | int | 输入目标对应的摄像头编号 |
| 'time\_stamp' | string | 输入目标出现时间，格式:’YY-MM-DD HH:MM:SS’ |
| 'feature' | list<list<float | 目标的特征，每一个目标特征为一个list，每一个特征值为float类型 |
| 'object\_id' | list<uint | 当前输入中注册的目标id，从1开始自增，list长度为1 |

# 4 Matching匹配

实现匹配，当匹配失败时可实现注册功能并更新注册列表，小于config中设置的threshold即为匹配成功

result= mm.matching(feature, register=true, rank=1)

输入参数

- feature，与register函数输入参数相同

- register(true)，匹配失败时是否注册目标

false表示匹配失败时不注册，当匹配失败时，返回结果中object\_id=[-1]，score=[0.0]

true表示匹配失败时进行注册，注册新的object\_id，返回结果中object\_id=[new\_id]，score=[None]

- rank(1)，返回匹配成功的的结果个数

0或-1表示返回所有匹配结果

N表示返回N个匹配结果，默认只返回一个结果

输出参数

-result 输出matching相关信息。格式为字典型：

{ input\_id0:{‘camera\_id’:i,‘time\_stamp’:\*\*:\*\*.\*\*, ‘object\_id’:[x],‘score’:[y]}

input\_id1:{…},

...}

表3 匹配函数输入输出参数

|  |  |  |
| --- | --- | --- |
| 关键字 | 类型 | 含义 |
| ‘input\_id\*’ | string | 输入目标编号 |
| 'camera\_id' | int | 输入目标对应的摄像头编号 |
| 'time\_stamp' | string | 输入目标出现时间，格式:’YY-MM-DD HH:MM:SS’ |
| 'feature' | list<list<float | 目标的特征，每一个目标特征为一个list，每一个特征值为float类型 |
| 'object\_id' | list<uint | 当前输入中注册的目标id，从1开始自增，list长度不大于rank或register\_number |
| 'score' | list<float | 匹配得分，每一个得分为一个float，list长度不大于rank或register\_number |

# 5 分时段匹配



* 普通匹配时，需要将待匹配的特征与数据库中注册的所有特征进行匹配，也就是上图中5个图片都需要计算得分
* 分时段匹配时，仅将待匹配特征与指定时间间隔内的注册特征进行匹配，具体时间间隔可以设置time\_match\_interval，可能只匹配2\3\4共3张图像（设查询图片1出现时间为time\_query\_1，只有这三种图片在[time\_query\_1+time\_match\_interval]之间）