**基于云服务的SLAM算法框架1.0**

|  |
| --- |
|  |
| **协同工作流程图** |

# 客户端：

* IMU：
  + 获取IMU数据
  + 处理后得到state\_imu
* 图像特征提取：
  + 获取图片
  + 图片预处理, 获取feature\_camera
* 与服务器通信：
  + 上传state\_imu
  + 上传feature\_camera
  + 下载state\_fused
* 延迟修正：
  + 输入：state\_fused，延迟的IMU数据
  + 输出：state\_fused\_modified

# 负载均衡器

* 解析客户端请求，分配给后端服务器

# 后端服务器：

* Visual Slam
  + 输入：feature\_camera
  + 输出：state\_visual
* IMU与Visual Slam融合
  + 输入：state\_imu，state\_visual
  + 输出：state\_fused

Tips：

1. 定义：

feature\_camera:

timestamp

index\_of\_keypoint\_in\_previous\_frame

index\_of\_keypoint\_in\_current\_frame

positon\_of\_keypoint\_in\_previous\_frame

positon\_of\_keypoint\_in\_current\_frame

state\_imu:

timestamp

position\_imu

cov\_position\_imu

orientation\_imu

cov\_position\_imu

state\_visual:

timestamp

position\_visual

cov\_position\_visual

orientation\_visual

cov\_position\_visual

state\_fused:

timestamp

position\_fused

cov\_position\_fused

orientation\_fused

cov\_orientation\_fused

state\_fused\_modified:

timestamp

position\_fused\_modified

cov\_position\_fused\_modified

orientation\_fused\_modified

cov\_orientation\_fused\_modified

1. 目前这个框架的特点类似于服务器驱动，服务器处理发送state\_fused之后，客户端再去采集下一帧图像。

另一个思路是客户端驱动，客户端不断地采集图像，服务器不断地处理图像和IMU数据。

相比较而言客户端驱动的执行效率要高，但实现较为复杂，可以在实现服务器驱动之后再实现客户端驱动。