深蓝学院
shenlanxueyuan.com

融合定位-第10章作业分享

主讲人 翰文

代码补充

深蓝学院
shenlanxueyuan.com

1. 实现预积分、地图匹配、边缘化、帧间匹配四种优化因子
2. 将上述四种约束因子加入划窗，进行优化

参考：https://github.com/AlexGeControl/Sensor-Fusion

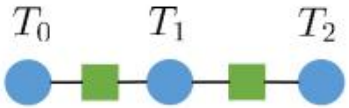
深蓝学院
shenlanxueyuan.com

1. 基于地图定位的滑动窗口模型

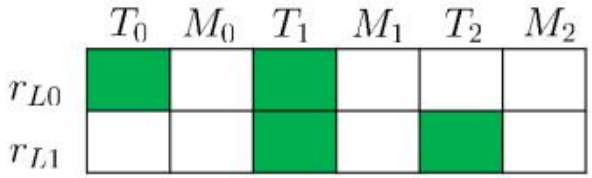
3）激光里程计相对位姿和优化变量的残差

该残差对应的因子为激光里程计因子。

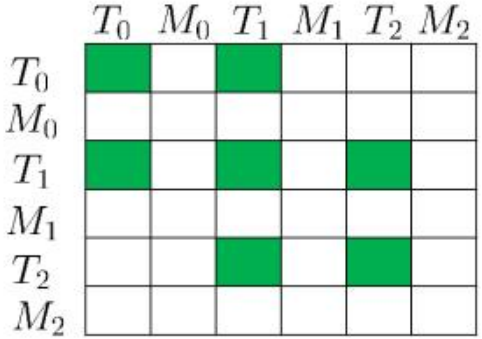
一个因子约束两个位姿，其模型如下：



残差关于优化变量的雅可比，可视化如下：



因此，对应的Hessian矩阵可视化为：



帧间匹配的部分代码：

深蓝学院
shenlanxueyuan.com

```
if ( jacobians ) {
  // compute shared intermediate results:
  const Eigen::Matrix3d R_i_inv = ori_i.inverse().matrix();
  const Eigen::Matrix3d J_r_inv = JacobianRInv(residual.block(INDEX_R, 0 ,3 , 1));    //  右雅克比
  const Eigen::Vector3d pos_ij = ori_i.inverse() * (pos_j - pos_i) ;

  if ( jacobians[0] ) {      //   残差rL0(rp  rq )  对 T0(p q)  M0(v ba bg) 的雅克比
    // implement computing:
    Eigen::Map<Eigen::Matrix<double, 6 , 15,  Eigen::RowMajor>>  jacobian_i (jacobians[0] );  //  col : rp
    jacobian_i.setZero();

    jacobian_i.block<3, 3>(INDEX_P, INDEX_P) =  -R_i_inv;
    jacobian_i.block<3, 3>(INDEX_R,INDEX_R)  =  -J_r_inv*(ori_ij*ori_j.inverse()*ori_i).matrix();
    jacobian_i.block<3, 3>(INDEX_P,INDEX_R) =  Sophus::SO3d::hat(pos_ij).matrix();

    jacobian_i = sqrt_info * jacobian_i ;        //   注意 sqrt_i 为对角的协方差矩阵对角线为观测的方差，可理解
  }

  if ( jacobians[1] ) {      //  残差rL0(rp  rq )  对 T0(p q)  M0(v ba bg) 的雅克比
    // implement computing:
    Eigen::Map<Eigen::Matrix<double, 6 ,15, Eigen::RowMajor>> jacobian_j (jacobians[1]);
    jacobian_j.setZero();

    jacobian_j.block<3, 3>(INDEX_P, INDEX_P) = R_i_inv;
    jacobian_j.block<3, 3>(INDEX_R,INDEX_R)  = J_r_inv*ori_ij.matrix();

    jacobian_j = sqrt_info * jacobian_j ;
  }
```

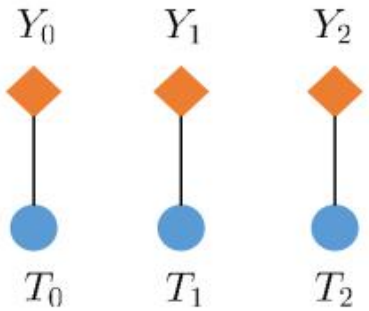
深蓝学院
shenlanxueyuan.com

1. 基于地图定位的滑动窗口模型

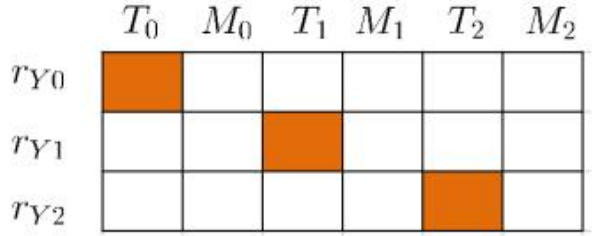
2）地图匹配位姿和优化变量的残差

该残差对应的因子为地图先验因子。

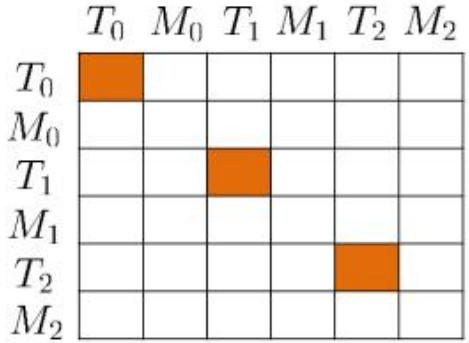
一个因子仅约束一个位姿，其模型如下：



残差关于优化变量的雅可比，可视化如下：



因此，对应的Hessian矩阵的可视化为：



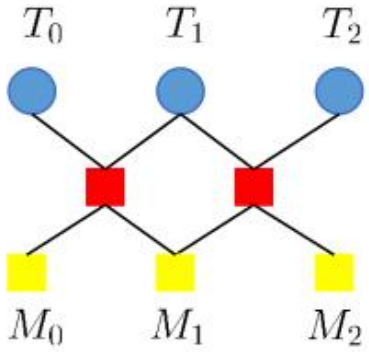
深蓝学院
shenlanxueyuan.com

1. 基于地图定位的滑动窗口模型

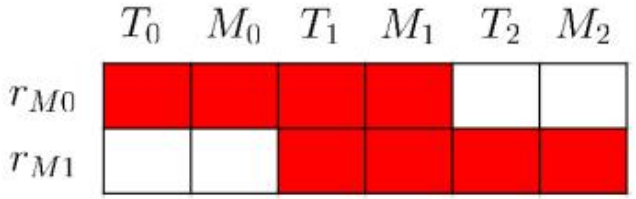
4）IMU预积分和优化变量的残差

该残差对应的因子为IMU因子。

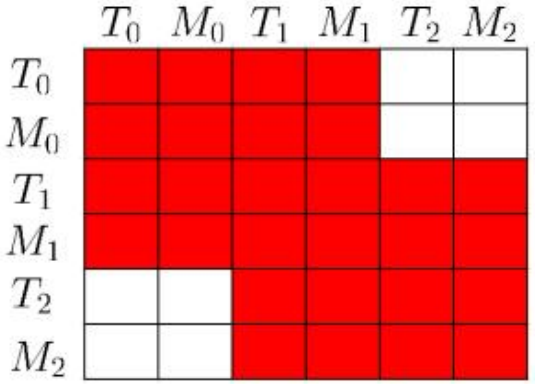
一个因子约束两个位姿，并约束两个时刻 IMU 的速度与 bias。



残差关于优化变量的雅可比，可视化如下：



因此，对应的Hessian矩阵可视化为：



加入滑窗，进行优化，边缘化及更新先验残差参考GEYAO助教和VINS的代码
由于ceres求解需要雅可比，所以多了一个将H矩阵反解出雅可比的步骤

雅各比矩阵和残差获取

每次迭代需要使用奇异值分解，从H和b中获取相应的雅各比矩阵和残差便于更新：

$$H = J^T J = QSQ^T$$

其中，S为奇异值，Q为对应的右奇异向量，那么：

$$J = \sqrt{S} Q^T$$

根据：

$$b = J^T e$$

有：

$$e = (J^T)^{-1} Q^T b$$

代码逻辑也是类似的，我们首先使用Eigen进行SVD分解，然后按照公式推导一步步走下来就可以了

将因子添加到优化框架中

```
if ( sliding_window_ptr_->GetNumParamBlocks() == 0 ) {
    // TODO: add init key frame
    sliding_window_ptr_->AddPRVAGParam(current_key_frame_, true);
} else {
    // TODO: add current key frame
    sliding_window_ptr_->AddPRVAGParam(current_key_frame_, false);
}
```

Ceres中添加残差块：

1. 创建问题
2. 添加待优化参数块
3. 添加四种残差块

```
problem.AddParameterBlock(target_key_frame.prvag, 15, local_parameterization);

if( target_key_frame.fixed ) {
        problem.SetParameterBlockConstant(target_key_frame.prvag);
}
```

```
problem.AddResidualBlock(
            factor_map_matching_pose,
            NULL,      // loss_function
            key_frame.prvag      //  一元边
);
```

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening