1. install Lab2 package from google drive
2.put ORBSLAM2 package in catkin_ws/src/

CODING

tracking.cc

search “//coding”

How to build?

$cd catkin_ws
$source devel/setup.bash
$cd src/ORB_SLAM2

chmod +x build.sh
./build.sh

chmod +x build_ros.sh
./build_ros.sh

How to run?

$ rosrun ORB_SLAM2 Stereo Vocabulary/ORBvoc.txt Examples/Stereo/zed.yaml true 0

1.Constant motion model

frame Class:

mCurrentFrame
mLastFrame

mTcw: translation from world to camera

Relocalization:

```
    // Relocalization is performed when tracking is lost
    // Track Lost: Query KeyFrame Database for keyframe candidates for relocalisation
    vector<KeyFrame*> vpCandidateKFs = mpKeyFrameDB-
>DetectRelocalizationCandidates(&mCurrentFrame);

    if(vpCandidateKFs.empty())
        return false;

    const int nKFs = vpCandidateKFs.size();


//check matching
    if(!bMatch)
    {
        mCurrentFrame.mTcw = cv::Mat::zeros(0, 0, CV_32F); // set mTcw back to empty if
relocation is failed
        return false;
    }
    else
    {
        mnLastRelocFrameId = mCurrentFrame.mnId;
        return true;
    }




    int nCandidates=0;

    for(int i=0; i<nKFs; i++)
    {
        KeyFrame* pKF = vpCandidateKFs[i];
        if(pKF->isBad())
            vbDiscarded[i] = true;
        else
        {
            int nmatches = matcher.SearchByBoW(pKF,mCurrentFrame,vvpMapPointMatches[i]);
            if(nmatches<15)
            {
                vbDiscarded[i] = true;
                continue;
            }
            else
            {
                PnPsolver* pSolver = new PnPsolver(mCurrentFrame,vvpMapPointMatches[i]);
                pSolver->SetRansacParameters(0.99,10,300,4,0.5,5.991);
                vpPnPsolvers[i] = pSolver;
                nCandidates++;
```

```
            }
        }
    }

// We perform first an ORB matching with each candidate
    // If enough matches are found we setup a PnP solver
    ORBmatcher matcher(0.75,true);

    vector<PnPsolver*> vpPnPsolvers;
    vpPnPsolvers.resize(nKFs);

    vector<vector<MapPoint*> > vvpMapPointMatches;
    vvpMapPointMatches.resize(nKFs);

    vector<bool> vbDiscarded;
    vbDiscarded.resize(nKFs);



    // Compute Bag of Words Vector
    mCurrentFrame.ComputeBoW();


    // Alternatively perform some iterations of P4P RANSAC
    // Until we found a camera pose supported by enough inliers
    bool bMatch = false;
    ORBmatcher matcher2(0.9,true);
```