代码主要做以下改动：

run_euroc.cpp 中：

```
void PubImuData()
{
//  string sImu_data_file = sConfig_path + "MH_05_imu0.txt";
    string sImu_data_file = "/media/yxt/storage/github_useful_tools/vio_data_simulation/bin/imu_pose.txt";
    cout << "1 PubImuData start sImu_data_file: " << sImu_data_file << endl;
    ifstream fsImu;
    fsImu.open(sImu_data_file.c_str());
    if (!fsImu.is_open())
    {
        cerr << "Failed to open imu file! " << sImu_data_file << endl;
        return;
    }

    std::string sImu_line;
    double dStampNSec = 0.0;
    Vector3d vAcc;
    Vector3d vGyr;
    double tmp;
    while (std::getline(fsImu, sImu_line) && !sImu_line.empty()) // read imu data
    {
        std::istringstream ssImuData(sImu_line);
        ssImuData >> dStampNSec;//时间戳
        for(int i=0;i<7;i++) //利用循环跳过 imu quaternion(4)和 imu position(3)
            ssImuData>>tmp;
        ssImuData >> vGyr.x() >> vGyr.y() >> vGyr.z() >> vAcc.x() >> vAcc.y() >> vAcc.z();
        // cout << "Imu t: " << fixed << dStampNSec << " gyr: " << vGyr.transpose() << " acc: " << vAcc.transpose() << endl;
        pSystem->PubImuData(dStampNSec, vGyr, vAcc);
        usleep(5000*nDelayTimes);//10000um = 0.01s
    }
    fsImu.close();
}
```

```
void PubSimImageData(){
    string sImage_file = "/media/yxt/storage/github_useful_tools/vio_data_simulation/bin/cam_pose.txt";

    cout << "1 PubImageData start sImage_file: " << sImage_file << endl;

    ifstream fsImage;
    fsImage.open(sImage_file.c_str());
    if (!fsImage.is_open())
    {
        cerr << "Failed to open image file! " << sImage_file << endl;
        return;
    }

    std::string sImage_line;
    double dStampNSec;
```

```
//    string sImgFileName;
    int n = 0;

    while (std::getline(fsImage, sImage_line) && !sImage_line.empty()){
        std::istringstream ssImuData(sImage_line);
        ssImuData >> dStampNSec;
        cout<<"cam time: "<<fixed<<dStampNSec<<endl;
          //cam_pose.txt 中相机数与 keyframe 中每一帧一一对应，从 all_points_0.txt 到 all_points_600
        string imagePath = "/media/yxt/storage/github_useful_tools/vio_data_simulation/bin/keyframe/all_points_"
                + std::to_string(n) + ".txt";
        cout<<"points_file: "<<imagePath<<endl;
         vector<cv::Point2f> FeaturePoints;//容器 FeaturePoints 存放一个相机的特征点(归一化坐标)
        ifstream f;
        f.open(imagePath.c_str());

        if (!f.is_open())
        {
            cerr << "Failed to open image file! " << imagePath << endl;
            return;
        }

        std::string s;
        while (std::getline(f, s) && !s.empty()){
            std::istringstream ss(s);
            double tmp;
            for (int i = 0; i < 4; i++)
                ss >> tmp;
            float px, py;
            ss >。。 px;
            ss >> py;
            cv::Point2f pt(px, py);
            FeaturePoints.push_back(pt);

        }
        pSystem->PubSimImageData(FeaturePoints, dStampNSec);
             usleep(100000*nDelayTimes);//usleep 延时时间单位为微秒，百万分之一,200000us = 0.2s,5Hz
        n++;
    }
    fsImage.close();

}
```

System.cpp 中

```
 void System::PubSimImageData(const vector<cv::Point2f> &feature, const double &dStampSec) {

    last_image_time = dStampSec;

    PUB_THIS_FRAME = true;
    TicToc t_r;

    trackerData[0].loaddata(feature, dStampSec);
```

```
9.
10.   for (unsigned int i = 0;; i++)
11.   {
12.       bool completed = false;
13.       completed |= trackerData[0].updateID(i); //a|=b;就是 a=a|b
14.
15.       if (!completed)
16.           break;
17.   }
18.
19.   if (PUB_THIS_FRAME) {
20.       pub_count++;
21.       shared_ptr<IMG_MSG> feature_points(new IMG_MSG());
22.       feature_points->header = dStampSec;
23.
24.       for (int i = 0; i < NUM_OF_CAM; i++) {
25.           auto &pts_velocity = trackerData[i].pts_velocity;
26.           for (int j = 0; j < feature.size(); j++) {
27.               int p_id = j;
28.                feature_points->points.push_back(Vector3d(feature[j].x, feature[j].y, 1));

29.               feature_points->id_of_point.push_back(p_id * NUM_OF_CAM + i);
30.               cv::Point2f pixel_point;
31.               pixel_point.x = 460 * feature[j].x + 255;
32.               pixel_point.y = 460 * feature[j].y + 255;
33.               feature_points->u_of_point.push_back(pixel_point.x);
34.               feature_points->v_of_point.push_back(pixel_point.y);
35.               feature_points->velocity_x_of_point.push_back(pts_velocity[j].x);
36.               feature_points->velocity_y_of_point.push_back(pts_velocity[j].y);
37. //              feature_points->velocity_x_of_point.push_back(0);
38. //              feature_points->velocity_y_of_point.push_back(0);
39.           }
40.           if (!init_pub)
41.           {
42.               cout << "4 PubImage init_pub skip the first image!" << endl;
43.               init_pub = 1;
44.           }
45.           else {
46.               m_buf.lock();
47.               feature_buf.push(feature_points);
48.               m_buf.unlock();
49.               con.notify_one();
50.           }
51.       }
52.   }
53.   // cout << "5 PubImage t : " << fixed << feature_points->header
54.   //     << " feature_buf size: " << feature_buf.size() << endl;
55. }
```

IMU 噪声与轨迹精度如下表所示

	acc_n	gyr_n	acc_w	gyr_w	max	mean	min	rmse
0	0	0	0	0	0.20	0.08	0.013	0.094
1	0.0019	0.0015	1e-5	1e-6	0.22	0.11	0.026	0.11
2	0.0038	0.0030	2e-5	2e-6	0.57	0.25	0.069	0.28
3	0.0076	0.0060	4e-5	4e-6	1.3	0.81	0.9	0.86
4	0.019	0.015	0.0001	1e-5	4.95	2.20	0.66	2.41

下面四张图分别对应上表 1、2、3、4 四种情况