## 1.中值法解算

在 imu_integration/src/estimator/activity.cpp TODO 中添加如下代码，即可实现

```
Eigen::Vector3d angular_delta;
Eigen::Matrix3d R_curr, R_prev;
double delta_t;
Eigen::Vector3d velocity_delta;

GetAngularDelta(1, 0, angular_delta);

// update orientation:
UpdateOrientation(angular_delta, R_curr, R_prev);

// get velocity delta:
GetVelocityDelta(1, 0, R_curr, R_prev, delta_t, velocity_delta);

// update position:
UpdatePosition(delta_t, velocity_delta);
```

## 2.完成中值法和欧拉法解算并进行精度对比

通过布尔变量 useEuler 判断使用中值法还是欧拉法 ,useEuler 为 true 时使用欧拉法。在 bool Activity::GetAngularDelta 中：

```
if (useEuler){
    angular_delta = delta_t*angular_vel_prev;
}
else
    angular_delta = 0.5*delta_t*(angular_vel_curr + angular_vel_prev);
```

在 bool Activity::GetVelocityDelta 中：

```
if (useEuler){
    velocity_delta = delta_t*linear_acc_prev;
}
else
    velocity_delta = 0.5*delta_t*(linear_acc_curr + linear_acc_prev);
```

接下来添加轨迹保存代码

在 estimator/activity.hpp 中添加成员函数 save_Pose_asTUM，用于保存 IMU 积分得出的估计轨迹。

```
void save_Pose_asTUM(std::string filename, Eigen::Matrix4d pose_, double time) {


```

```
std::ofstream save_points;
save_points.setf(std::ios::fixed, std::ios::floatfield);
save_points.open(filename.c_str(), std::ios::out|std::ios::app);

if(!save_points.is_open()){
  std::cout << "fail to open file" << std::endl;
  return;
}

Eigen::Quaterniond q(pose_.block<3, 3>(0, 0));

save_points.precision(9);
save_points <<time<<" ";
save_points.precision(5);
save_points <<pose_(0, 3)<<" "
          <<pose_(1, 3)<<" "
          <<pose_(2, 3)<<" "
          <<q.x()<<" "
          <<q.y()<<" "
          <<q.z()<<" "
          <<q.w() <<std::endl;

save_points.close();
}
```

为保存真值的轨迹，本作业另起一个节点订阅真值话题，该节点对应的 saveDataInTum.cpp 如下所示

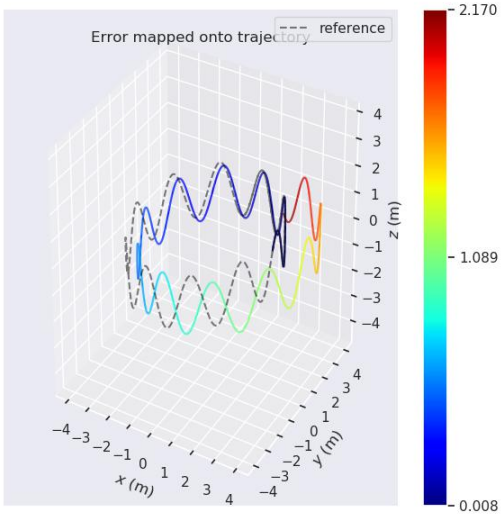
```
#include <fstream>
#include <string>
#include <ros/ros.h>
#include <nav_msgs/Odometry.h>
#include <Eigen/Core>
#include <Eigen/Dense>
#include <boost/filesystem.hpp>
#include "stdio.h"

static std::ofstream ground_truth_str;

bool CreateDirectory(std::string directory_path){
   if(!boost::filesystem::is_directory(directory_path))
   boost::filesystem::create_directory(directory_path);

   if(!boost::filesystem::is_directory(directory_path)){
      std::cout<< "cannot create directory" << std::endl;
```

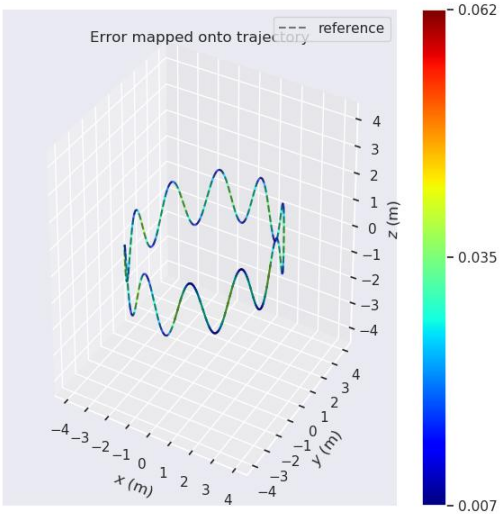
```
        return false;
    }

    std::string filename{"/workspace/assignments/06-imu-navigation/src/imu_integration/result/sim/gt.txt"};
    remove(filename.c_str());
    return true;
}

bool CreateFile(std::ofstream& ofs, std::string file_path){
    ofs.setf(std::ios::fixed, std::ios::floatfield);
    ofs.open(file_path.c_str(), std::ios::out|std::ios::app);
    if(!ofs){
        std::cout << "cannot open file" << std::endl;
        return false;
    }
    return true;
}

void callback_truth(const nav_msgs::OdometryConstPtr& ground_truth){

    static bool is_file_created = false;

    if(!is_file_created){
        if(!CreateDirectory("/workspace/assignments/06-imu-navigation/src/imu_integration/result/sim"))
        return;
        if(!CreateFile(ground_truth_str,"/workspace/assignments/06-imu-navigation/src/imu_integration/result/sim/gt.txt"))
        return;
        is_file_created = true;
    }

    Eigen::Vector3d t;
    t(0) = ground_truth->pose.pose.position.x;
    t(1) = ground_truth->pose.pose.position.y;
    t(2) = ground_truth->pose.pose.position.z;
    Eigen::Quaterniond q;
    q.w() = ground_truth->pose.pose.orientation.w;
    q.x() = ground_truth->pose.pose.orientation.x;
    q.y() = ground_truth->pose.pose.orientation.y;
    q.z() = ground_truth->pose.pose.orientation.z;

    ros::Time timestamp_ = ground_truth->header.stamp;
```

```
    double timestamp_in_sec = timestamp_.toSec();

    ground_truth_str.precision(9);
    ground_truth_str <<timestamp_in_sec<<" ";
    ground_truth_str.precision(5);
    ground_truth_str <<t(0)<<" "
                     <<t(1)<<" "
                     <<t(2)<<" "
                     <<q.x()<<" "
                     <<q.y()<<" "
                     <<q.z()<<" "
                     <<q.w() <<std::endl;

}

int main(int argc, char** argv){
    std::string nodename{"saveDataInTum"};
    ros::init(argc, argv, nodename);

    ros::NodeHandle nh;
    ros::Subscriber gt_sub = nh.subscribe("/sim/gt", 10, &callback_truth);

    ros::Rate loop_rate(100);
    while(ros::ok()){
        ros::spinOnce();
        loop_rate.sleep();
    }

    ground_truth_str.close();
    return 0;
}
```

精度对比结果：

| 欧拉法 | 中值法 |
| --- | --- |



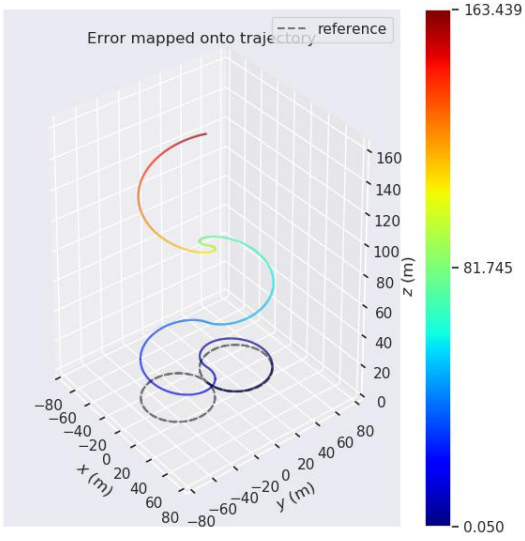




| | |
|---|---|
| max | 2.170356 |
| mean | 0.778399 |
| median | 0.625838 |
| min | 0.007901 |
| rmse | 1.007943 |
| sse | 2602.861335 |
| std | 0.640346 |

| | |
|---|---|
| max | 0.061941 |
| mean | 0.023919 |
| median | 0.024830 |
| min | 0.007162 |
| rmse | 0.025269 |
| sse | 1.547172 |
| std | 0.008149 |

## 3.生成不同运动状况并进行精度分析

本作业共生成了 5 种运动状态，分别为绕 8 字、静止、匀速运动、加速运动、先加速后减速。在绕 8 字运动仿真中，发现 z 轴方向有很大偏移：

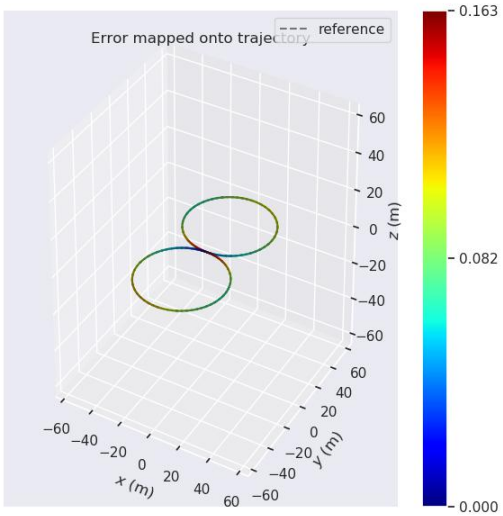
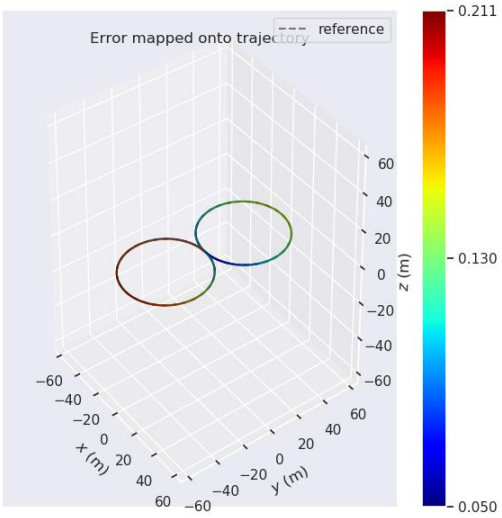
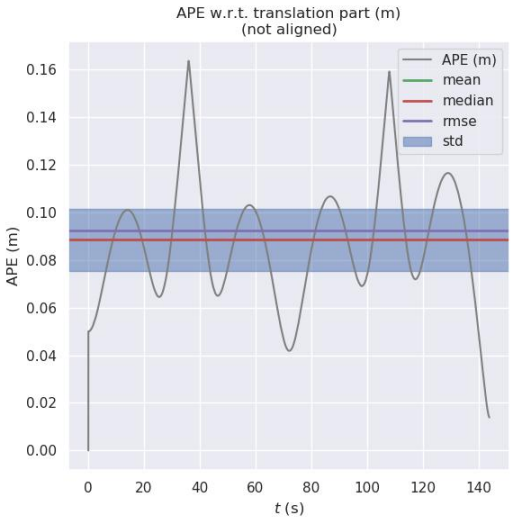
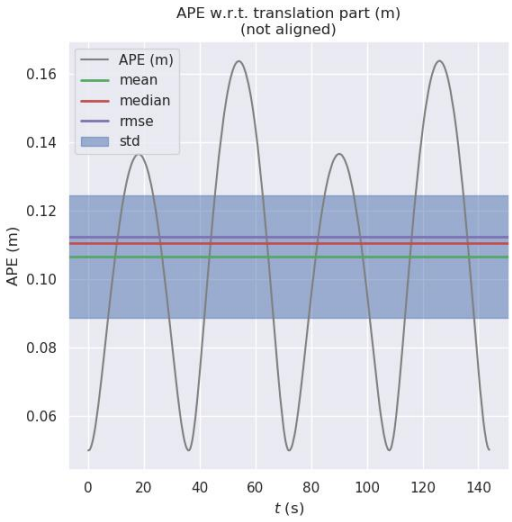



分析其原因，通过 rqt_bag 查看仿真数据的 rosbag 中的重力值，发现该重力

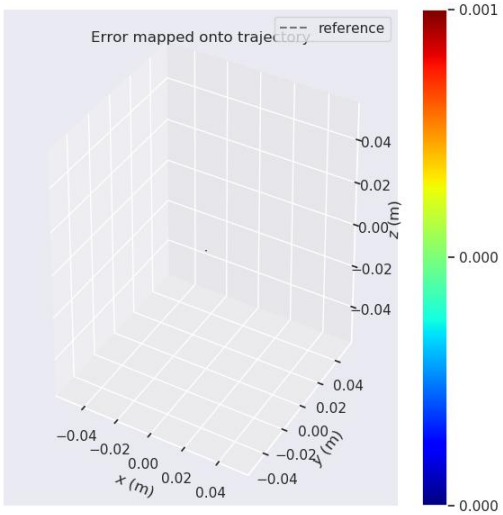
值为-9.794216，需要修改 estimate_node 加载的 yaml 文件中 gravity/z = -9.794216。修改后仿真结果正常。

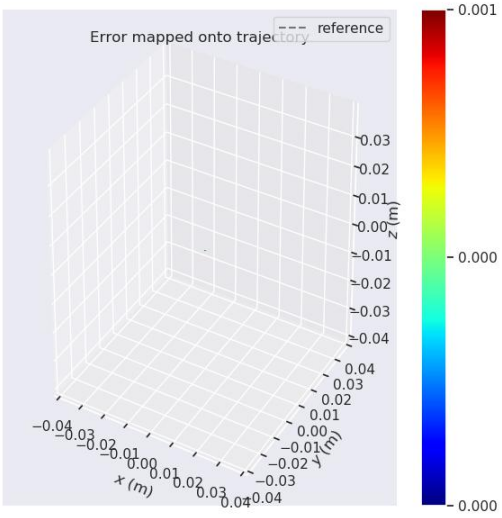
欧拉法和中值法结果如下，左列为欧拉法，右列为中值法。

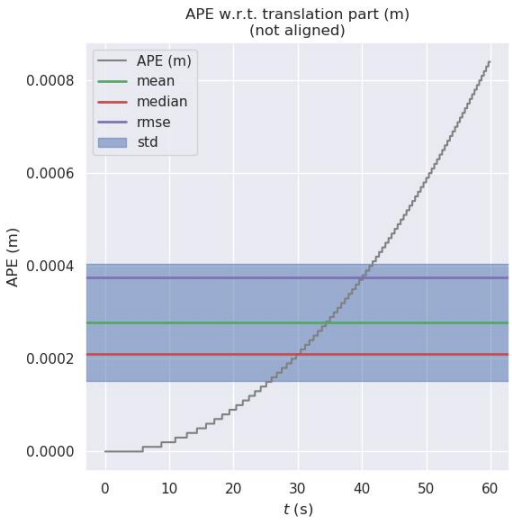
绕 8 字

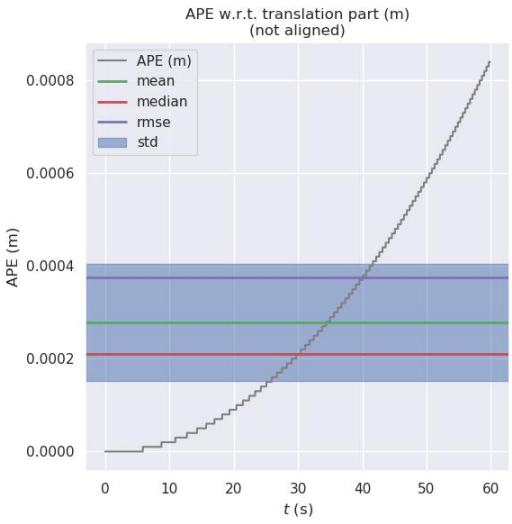
<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>欧拉法</td>
<td>中值法</td>
</tr>
</table>

# 静止


Error mapped onto trajectory
reference
0.001
0.000
0.000
x (m)
y (m)
z (m)


Error mapped onto trajectory
reference
0.001
0.000
0.000
x (m)
y (m)
z (m)

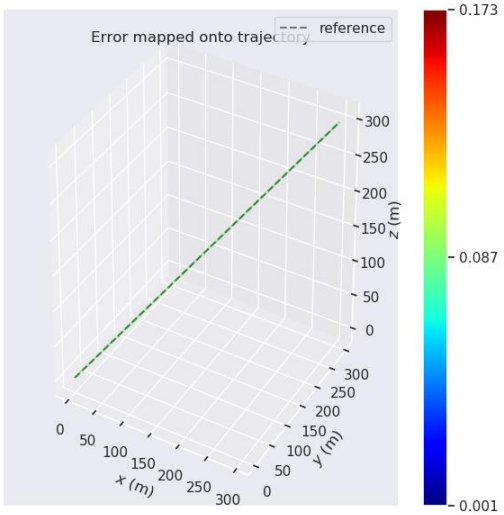

APE w.r.t. translation part (m)
(not aligned)
APE (m)
mean
median
rmse
std
APE (m)
t (s)

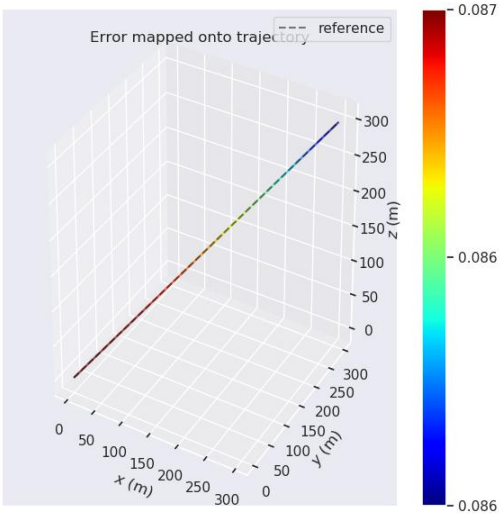

APE w.r.t. translation part (m)
(not aligned)
APE (m)
mean
median
rmse
std
APE (m)
t (s)

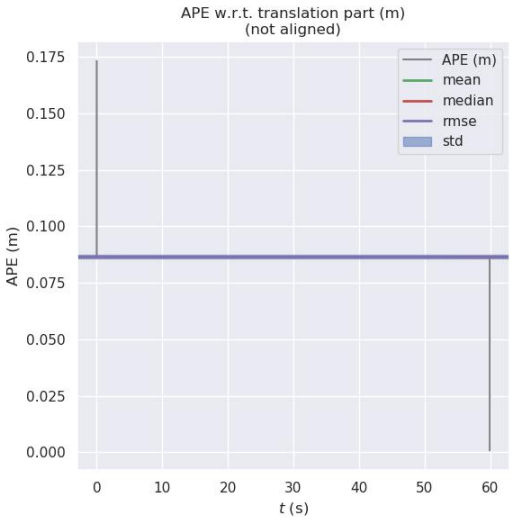
欧拉法

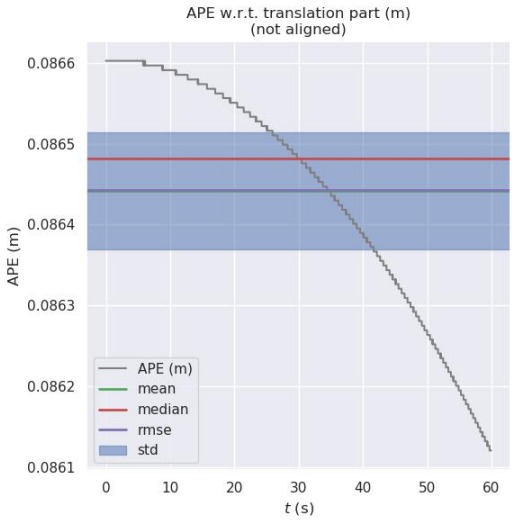
中值法

# 匀速运动


Error mapped onto trajectory
reference
0.173
0.087
0.001
x (m)
y (m)
z (m)

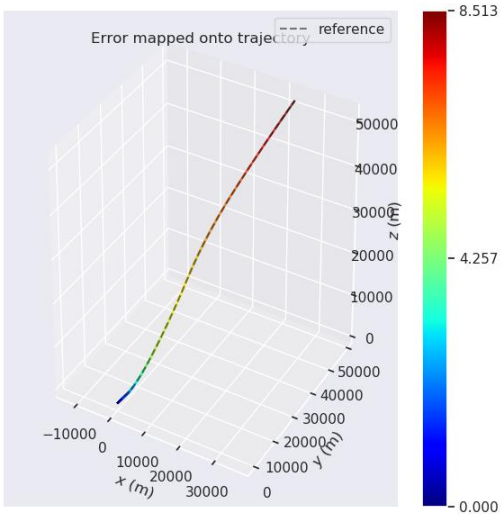

Error mapped onto trajectory
reference
0.087
0.086
0.086
x (m)
y (m)
z (m)

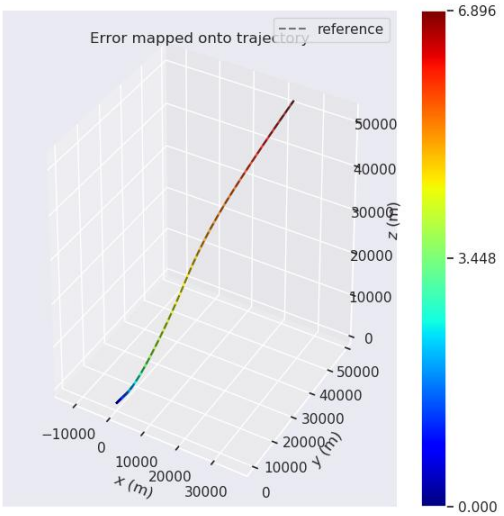

APE w.r.t. translation part (m)
(not aligned)
APE (m)
mean
median
rmse
std
APE (m)
t (s)

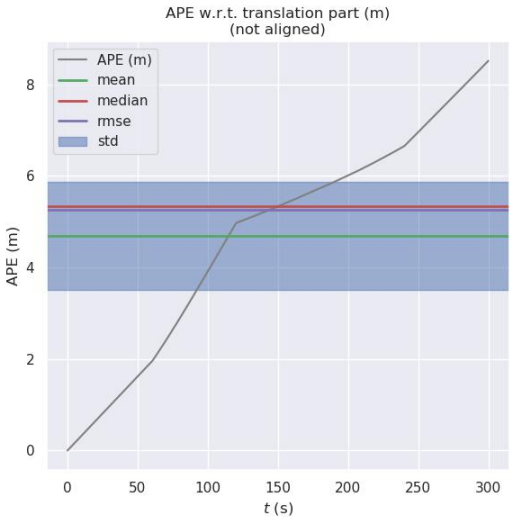

APE w.r.t. translation part (m)
(not aligned)
APE (m)
mean
median
rmse
std
APE (m)
t (s)

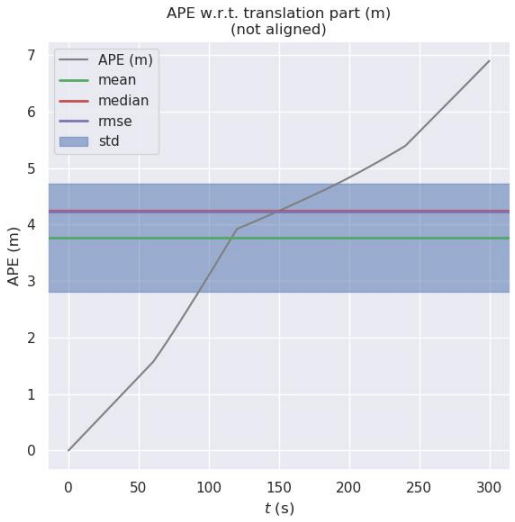
| 欧拉法 | 中值法 |
| --- | --- |

# 加速运动


Error mapped onto trajectory
reference
8.513
4.257
0.000
x (m)
y (m)
z (m)

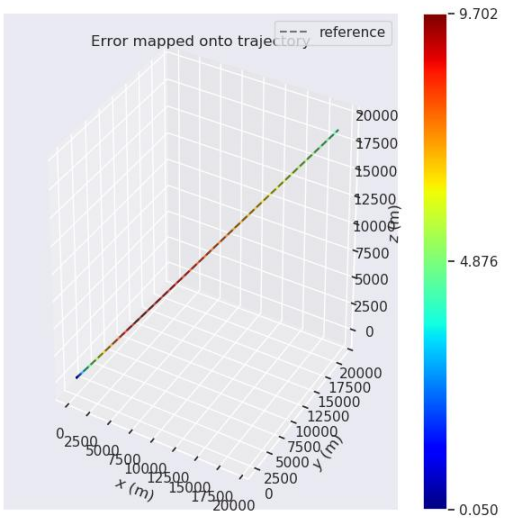

Error mapped onto trajectory
reference
6.896
3.448
0.000
x (m)
y (m)
z (m)

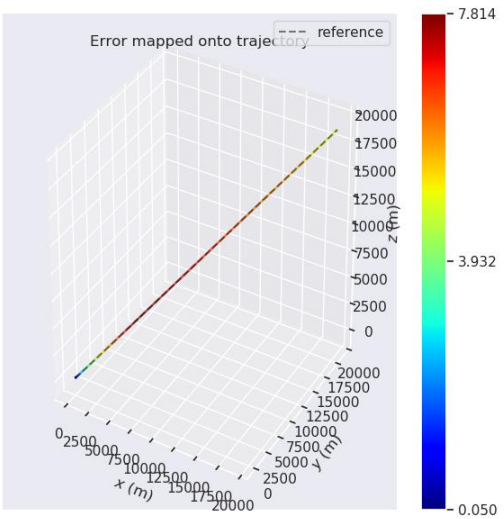

APE w.r.t. translation part (m)
(not aligned)
APE (m)
mean
median
rmse
std
APE (m)
t (s)

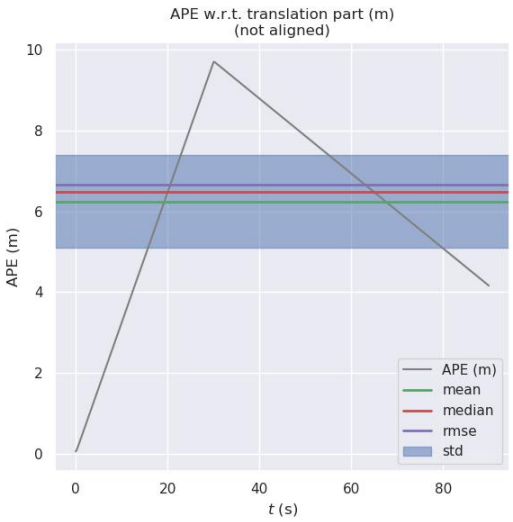

APE w.r.t. translation part (m)
(not aligned)
APE (m)
mean
median
rmse
std
APE (m)
t (s)

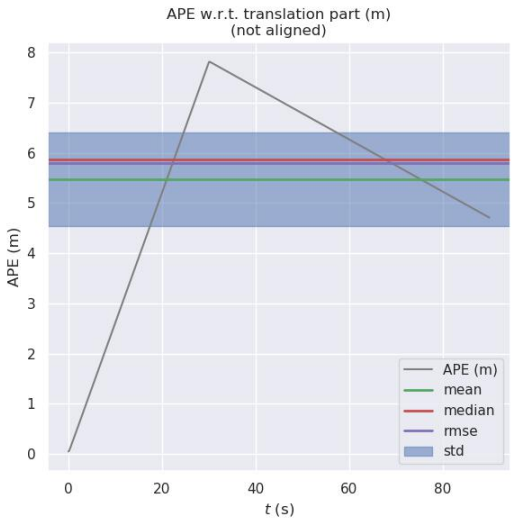
| 欧拉法 | 中值法 |
| --- | --- |

先加速后减速




| 欧拉法 | 中值法 |
| --- | --- |

结论：

对于静止或者匀速运动，欧拉法的精度和中值法精度相当。绕 8 字运动欧拉法的精度比中值法高，可以理解为绕 8 字在一段时间内的角速度相同，使用中值法并不能获得太多优势。

对于变速运动，如加速和先加速后减速运动，由于存在加速度变化，中值法取平均值进行积分比较合理，试验结果也表明，变速运动时，中值法精度优于欧拉法。