*********when mapping by 3d motion:

1.
if we use imu, odom, velodyne lidar sensor:
use_imu_data = **true**,
use_odometry = **true**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.pose_tracker.use_odom_angle = **false**,

if we find the imu can not provide a good angle
use_odometry = **true**,

if we find the imu and odom can not provide a good angle
use_odometry = **false**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
when use_pose_filter = **true** scan_matching time will increase, **set** the parameter:
running_search_angle can **get** a suitebale time.
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.109 (0.109 means 0.109 radians)

if we **do** not match by using high_grid:
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_high_grid = **false**

if we map outside, some parameter can be adjust:
*/**********these parameter just are suitable in zizhu, other scenario need further test**********/*
TRAJECTORY_BUILDER_3D.high_resolution_adaptive_voxel_filter.max_length = 20.0
TRAJECTORY_BUILDER_3D.low_resolution_adaptive_voxel_filter.max_length = 20.0
TRAJECTORY_BUILDER_3D.high_resolution_adaptive_voxel_filter.min_num_points = 2000
TRAJECTORY_BUILDER_3D.low_resolution_adaptive_voxel_filter.min_num_points = 2000
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_high_grid = **false**

2.
if we use imu, velodyne lidar sensor:
use_imu_data = **true**,
use_odometry = **false**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.pose_tracker.use_odom_angle = **false**,

if we find the imu can not provide a good angle
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
when use_pose_filter = **true** scan_matching time will increase, **set** the parameter:
running_search_angle can **get** a suitebale time.
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.109 (0.109 means 0.109 radians)

the other **is** same **as** above.

3.
if we use odom, velodyne lidar sensor:
use_imu_data = **false**,

use_odometry = **true**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.pose_tracker.use_odom_angle = **true**,

if we find the odom can not provide a good angle
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
when use_pose_filter = **true** scan_matching time will increase, **set** the parameter:
running_search_angle can **get** a suitebale time.
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.109 (0.109 means 0.109 radians)

the other **is** same **as** above.

4.
if we use just velodyne lidar sensor:
use_imu_data = **false**,
use_odometry = **false**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.872 (0.872 means 0.872 radians)
TRAJECTORY_BUILDER_3D.evaluate.angle_interver = 0.00436 (**this** means the min resolution angle when searching angle)

*********when localization by 3d motion:
1.
if we use imu, odom, velodyne lidar sensor:
use_imu_data = **true**,
use_odometry = **true**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.pose_tracker.use_odom_angle = **false**,

if we find the imu can not provide a good angle
use_odometry = **true**,

if we find the imu and odom can not provide a good angle
use_odometry = **false**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
when use_pose_filter = **true** scan_matching time will increase, **set** the parameter:
running_search_angle can **get** a suitebale time.
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.109 (0.109 means 0.109 radians)

if we **do** not match by using high_grid:
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_high_grid = **false**

if we map outside, some parameter can be adjust:
*/**********these parameter just are suitable in zizhu, other scenario need further test**********/*
TRAJECTORY_BUILDER_3D.high_resolution_adaptive_voxel_filter.max_length = 20.0
TRAJECTORY_BUILDER_3D.low_resolution_adaptive_voxel_filter.max_length = 20.0

TRAJECTORY_BUILDER_3D.high_resolution_adaptive_voxel_filter.min_num_points = 2000
TRAJECTORY_BUILDER_3D.low_resolution_adaptive_voxel_filter.min_num_points = 2000
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_high_grid = **false**

2.
if we use imu, velodyne lidar sensor:
use_imu_data = **true**,
use_odometry = **false**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.pose_tracker.use_odom_angle = **false**,

if we find the imu can not provide a good angle
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
when use_pose_filter = **true** scan_matching time will increase, **set** the parameter:
running_search_angle can **get** a suitebale time.
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.109 (0.109 means 0.109 radians)

the other **is** same **as** above.

3.
if we use odom, velodyne lidar sensor:
use_imu_data = **false**,
use_odometry = **true**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.pose_tracker.use_odom_angle = **true**,

if we find the odom can not provide a good angle
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
when use_pose_filter = **true** scan_matching time will increase, **set** the parameter:
running_search_angle can **get** a suitebale time.
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.109 (0.109 means 0.109 radians)

the other **is** same **as** above.

4.
if we use just velodyne lidar sensor:
use_imu_data = **false**,
use_odometry = **false**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.use_pose_filter = **true**,
TRAJECTORY_BUILDER_3D.kalman_local_trajectory_builder.running_search_angle = 0.872 (0.872 means 0.872 radians)
TRAJECTORY_BUILDER_3D.evaluate.angle_interver = 0.00436 (**this** means the min resolution angle when searching angle)

now our forklift just use odom and velodyne, the parameter **in** localization_by_velodyne.lua **is** using **in** forklift.
when mapping, recommend record bag and using mapping_by_velodyne_points_sim.launch