



دانشگاه صنعتی امیرکبیر  
(پلی تکنیک تهران)  
دانشکده مهندسی کامپیووتر

پایان نامه کارشناسی  
مهندسی کامپیووتر

مکان یابی مبتنی بر لایدار در خودروهای خودران

نگارش  
مهرداد رحمانی

استاد راهنما  
دکتر مهدی جوانمردی

شهریور ۱۴۰۲

لِمَنْ يُرِكَتُ الْأَرْضُ  
بِسْرَهُ تَحْمِلُهُ  
لِمَنْ يَرْجِعَ الْأَرْضُ  
يَوْمَ الْحِجَارَةِ



به نام خدا

تاریخ:

## تعهدنامه اصالت اثر

اینجانب مهدی رحمانی متعهد می‌شوم که مطالب مندرج در این پایان نامه حاصل کار پژوهشی اینجانب تحت نظارت و راهنمایی استادی دانشگاه صنعتی امیرکبیر بوده و به دستاوردهای دیگران که در این پژوهش از آنها استفاده شده است مطابق مقررات و روال متعارف ارجاع و در فهرست منابع و مأخذ ذکر گردیده است. این پایان نامه قبلاً برای احراز هیچ مدرک هم‌سطح یا بالاتر ارائه نگردیده است.

در صورت اثبات تخلف در هر زمان، مدرک تحصیلی صادر شده توسط دانشگاه از درجه اعتبار ساقط بوده و دانشگاه حق پیگیری قانونی خواهد داشت.

کلیه نتایج و حقوق حاصل از این پایان نامه متعلق به دانشگاه صنعتی امیرکبیر می‌باشد. هرگونه استفاده از نتایج علمی و عملی، و اگذاری اطلاعات به دیگران یا چاپ و تکثیر، نسخه‌برداری، ترجمه و اقتباس از این پایان نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است.  
نقل مطالب با ذکر مأخذ بلامانع است.

مهدی رحمانی

امضا

## تقدیم

تقدیم به پدر، مادر و برادر عزیزم که در تمامی این سال‌ها تحصیل را برایم تسهیل نمودند و همواره در سختی‌ها و دشواری‌های زندگی، یاوری دلسوز و فداکار و پشتیبانی محکم برایم بودند.

## تقدیر و تشکر

بدینوسیله مراتب قدردانی و امتنان خود را خدمت،  
استاد دلسوز و گرانقدرم؛ جناب آقای دکتر مهدی جوانمردی که در کمال سعه صدر، با حسن  
خلق و فروتنی، از هیچ کمکی در این عرصه بر من دریغ نداشتند،  
جناب آقای دکتر احسان جوانمردی بابت آموزش‌ها، کمک‌های بی دریغ و راهنمایی‌هایشان،  
تمامی دوستان و همراهان که در این سال‌ها در کنار من بودند،  
ابراز و از تمامی زحمات آنان تشکر مینمایم.

مهدی رحمانی

۱۴۰۲ شهریور

## چکیده

در صنعت وسایل نقلیه‌ی خودران و همچنین رباتیک، یکی از مهم‌ترین قابلیت‌های جسم متحرک مورد نظر، قابلیت مکان‌یابی می‌باشد. چنانچه مکان جسم در محیط با دقت مناسبی مشخص نباشد؛ امكان مسیریابی نادرست و همچنین تصادف وجود دارد. با توجه به آنکه در خودروهای خودران سرعت جسم ممکن است زیاد باشد، بنابراین نرخ به روزرسانی مکان جسم نیز باید بالا باشد تا در هر لحظه نیز از مکان آن مطلع باشیم.

در روش‌های سنتی با به کارگیری حسگر GNSS مکان یک جسم متحرک را تخمین می‌زنند که به علت نرخ به روزرسانی پایین و وابسته بودن به امواج ماهواره‌های خارجی، همواره مشکلاتی را به همراه داشته است. در پژوهش‌هایی حسگر IMU نیز به عنوان مکملی با نرخ به روزرسانی بالاتر، با هدف رفع نواقص GNSS به کار گرفته شده است.

در این پژوهش قصد داریم تا از لایدار که حسگری پرکاربرد در صنعت خودران است، برای بهبود دقت مکان‌یابی و رفع نواقص دو حسگر دیگر، استفاده نماییم. برای این منظور، عملکرد این حسگر برای مکان‌یابی در دو بخش شبیه‌سازی و عملی، نمایش داده شده است و سپس با ترکیب سه حسگر ذکر شده به کمک فیلتر کالمن توسعه یافته، به مکان‌یابی دقیق‌تری برای خودروی خودران دست یافتیم.

### واژه‌های کلیدی:

مکان‌یابی، لایدار، خودروی خودران، فیلتر کالمن

## صفحه

## فهرست مطالب

|    |   |
|----|---|
| ۱  | چکیده.....                                      |
| ۲  | فصل اول مقدمه.....                              |
| ۳  | مقدمه.....                                      |
| ۴  | ۱- هدف پژوهش.....                               |
| ۵  | ۲- بیان صورت مسئله.....                         |
| ۶  | ۳- بیان راه حل.....                             |
| ۷  | ۴- تاریخچه.....                                 |
| ۱۱ | فصل دوم نرم افزارهای مورد استفاده.....          |
| ۱۲ | نرم افزارهای مورد استفاده.....                  |
| ۱۲ | ۱- شبیه ساز کارلا.....                          |
| ۱۳ | ۱-۱- معماری کارلا.....                          |
| ۱۴ | ۱-۲- اجزای مختلف کارلا.....                     |
| ۱۶ | ۲- پیاده سازی.....                              |
| ۱۷ | ۲-۱- سیستم عامل رأس.....                        |
| ۱۷ | ۲-۲- عملکرد رأس.....                            |
| ۱۹ | ۲-۲-۱- ساختار رأس.....                          |
| ۲۱ | ۲-۲-۲- ابزارهای ROSbag و Rviz.....              |
| ۲۲ | ۳- پیاده سازی.....                              |
| ۲۳ | ۳-۱- آستر استادیو و کیت توسعه نرم افزار.....    |
| ۲۵ | فصل سوم مکان یابی.....                          |
| ۲۶ | مکان یابی.....                                  |
| ۲۷ | ۱- SLAM.....                                    |
| ۲۸ | ۱-۱- SLAM چیست؟.....                            |
| ۳۰ | ۱-۲- SLAM چگونه کار می کند؟.....                |
| ۳۱ | ۱-۳- نقش لایدار در SLAM.....                    |
| ۳۲ | ۲- نقشه برداری.....                             |
| ۳۳ | ۲-۱- نقشه و انواع آن.....                       |
| ۳۵ | ۲-۲- مراحل تهیه نقشه.....                       |
| ۳۶ | ۳- مکان یابی مبتنی بر لایدار در پژوهش فعلی..... |

|    |   |
|----|---|
| ۳۷ | ۱-۳-۳- ابرنقط چیست؟                       |
| ۳۹ | ۲-۳-۳- الگوریتم ثبت نقاط                  |
| ۴۲ | <b>فصل چهارم ترکیب حسگرها</b>             |
| ۴۳ | <b>ترکیب حسگرها</b>                       |
| ۴۳ | ۱- انواع مختلف ترکیب حسگرها               |
| ۴۵ | ۲- سطوح ترکیب حسگرها                      |
| ۴۷ | ۳- الگوریتم‌های ترکیب حسگر                |
| ۴۷ | ۱-۳-۴- الگوریتم فیلتر کالمن               |
| ۵۰ | ۲-۳-۴- الگوریتم‌های دیگر                  |
| ۵۱ | <b>فصل پنجم نتایج</b>                     |
| ۵۲ | <b>نتایج</b>                              |
| ۵۳ | ۱-۱-۵- مکان‌یابی به کمک GNSS              |
| ۵۴ | ۱-۱-۵- نحوه‌ی مکان‌یابی با GNSS           |
| ۵۵ | ۲-۱-۵- نتایج مکان‌یابی با GNSS در کارلا   |
| ۵۶ | ۲-۱-۵- مکان‌یابی به کمک IMU               |
| ۵۶ | ۱-۲-۵- نحوه‌ی مکان‌یابی با IMU            |
| ۵۷ | ۲-۲-۵- نتایج مکان‌یابی با IMU در کارلا    |
| ۵۸ | ۳-۳-۵- مکان‌یابی به کمک لایدار            |
| ۵۸ | ۱-۳-۵- نحوه‌ی مکان‌یابی با لایدار         |
| ۶۰ | ۲-۳-۵- نتایج مکان‌یابی با لایدار در کارلا |
| ۶۱ | ۴-۴-۵- مکان‌یابی به کمک ترکیب حسگرها      |
| ۶۱ | ۱-۴-۵- ترکیب حسگر GNSS و IMU              |
| ۶۲ | ۲-۴-۵- ترکیب حسگر GNSS و IMU و لایدار     |
| ۶۴ | ۵-۵- مکان‌یابی با لایدار به صورت عملی     |
| ۶۸ | <b>فصل ششم جمع‌بندی و پیشنهادات</b>       |
| ۷۲ | <b>منابع و مراجع</b>                      |
| ۷۵ | <b>پیوست‌ها</b>                           |
| ۷۷ | <b>Abstract</b>                           |

## صفحه

## فهرست اشکال

|          |  |
|----------|--|
| ..... ۴  | ..... شکل ۱-۱ نمایی از حسگر INS و نمای داخلی آن  |
| ..... ۵  | ..... شکل ۱-۲ پدیده‌های چند مسیری و NLOS از معایب استفاده از GNSS [۴]                      |
| ..... ۷  | ..... شکل ۱-۳ نمونه‌ای از اسکن ثبت شده توسط حسگر لایدار                                    |
| ..... ۱۳ | ..... شکل ۱-۴ تصویری از محیط شبیه‌ساز کارلا [۱۳]   |
| ..... ۱۳ | ..... شکل ۲-۱ نحوه‌ی تعامل کد کاربر در قسمت مشتری با شبیه ساز اجرا شده در سرویس دهنده [۱۳] |
| ..... ۱۸ | ..... شکل ۲-۲ نحوه‌ی ارتباط جستجوگر با سرویس دهنده وب [۱۴]                                 |
| ..... ۱۹ | ..... شکل ۲-۳ نحوه‌ی ارتباط رأس-بگ با سرویس دهنده وب [۱۵]                                  |
| ..... ۲۱ | ..... شکل ۲-۴ ساختار و چگونگی ارتباطات در رأس [۱۶]   |
| ..... ۲۲ | ..... شکل ۲-۵ چگونگی عملکرد رأس-بگ   |
| ..... ۲۴ | ..... شکل ۲-۶ چگونگی عملکرد رأس-بگ   |
| ..... ۲۴ | ..... شکل ۷-۱ نحوه‌ی اتصال حسگر لایدار به لپتاپ [۱۷]                                       |
| ..... ۲۶ | ..... شکل ۷-۲ خروجی حسگر لایدار در آستر استادیو  |
| ..... ۲۸ | ..... شکل ۷-۳ مکان‌یابی ربات و عدم برخورد آن با موانع در رسیدن به مقصد [۱۸]                |
| ..... ۲۹ | ..... شکل ۷-۴ نمونه‌ای از فرآیند SLAM انجام شده [۲۰]                                       |
| ..... ۳۱ | ..... شکل ۷-۵ نمونه‌ای از فرآیند SLAM انجام شده [۲۱]                                       |
| ..... ۳۲ | ..... شکل ۷-۶ ربات مجهر به لایدار در SLAM [۲۱]   |
| ..... ۳۳ | ..... شکل ۷-۷ نمونه‌ای از نقشه تهیه شده برای خودروی خودران [۲۰]                            |
| ..... ۳۴ | ..... شکل ۷-۸ انواع نقشه   |
| ..... ۳۸ | ..... شکل ۷-۹ نقشه توپولوژیکی [۲۳]   |
| ..... ۳۸ | ..... شکل ۸-۱ اسکن لایدار ثبت شده در کارلا   |
| ..... ۳۹ | ..... شکل ۸-۲ نمونه‌ای از خروجی الگوریتم ثبت نقاط [۲۶]                                     |
| ..... ۴۱ | ..... شکل ۱۱-۱ مقایسه‌ی دقیق روش ICP و NDT [۲۹]  |
| ..... ۴۴ | ..... شکل ۱-۱ روش‌های مختلف ترکیب حسگرها [۳۰]  |
| ..... ۴۵ | ..... شکل ۲-۱ طرح‌های ارتباطی حسگرها [۳۰]  |
| ..... ۴۷ | ..... شکل ۳-۱ طرح‌های ارتباطی حسگرها [۳۰]  |
| ..... ۴۸ | ..... شکل ۴-۱ نحوه‌ی کارکرد فیلتر کالمون در این پژوهش [۳۱]                                 |
| ..... ۵۲ | ..... شکل ۱-۲ نمای بالا از شهر مورد آزمایش در کارلا  |
| ..... ۵۳ | ..... شکل ۲-۲ نمایی از خودرو در حال جمع آوری داده در سطح شهر                               |
| ..... ۵۴ | ..... شکل ۳-۱ نحوه‌ی کارکرد حسگر GPS در تعیین مکان [۳۲]                                    |
| ..... ۵۹ | ..... شکل ۴-۲ داده‌برداری خودرو با لایدار در دو گام زمانی متوالی                           |

|          |  |
|----------|--|
| ..... ۵۹ | شکل ۵-۵ نحوه‌ی تطابق اسکن برای یافتن جابه‌جایی نسبی بین دو ابرنقطا[۳۱] |
| ..... ۶۳ | شکل ۵-۶ نحوه‌ی ترکیب حسگرهای IMU و GNSS و لایدار با فیلتر کالمن [۳۱]   |
| ..... ۶۴ | شکل ۵-۷ حسگر لایدار آستر مورد استفاده                                  |
| ..... ۶۴ | شکل ۵-۸ حسگر لایدار آستر مورد استفاده                                  |
| ..... ۶۵ | شکل ۵-۹ نمای بالای نقشه‌ی نمونه برداری شده از دانشکده کامپیوتر         |
| ..... ۶۶ | شکل ۵-۱۰ نمای ایزومتریک نقشه‌ی نمونه برداری شده از دانشکده کامپیوتر    |
| ..... ۶۶ | شکل ۵-۱۱ نمای بالای نقشه‌ی بدون نمونه برداری از دانشکده کامپیوتر       |
| ..... ۶۷ | شکل ۵-۱۲ نمای ایزومتریک نقشه‌ی بدون نمونه برداری از دانشکده کامپیوتر   |
| ..... ۷۵ | شکل پ-۱ سیستم آماده شده برای داده‌برداری                               |
| ..... ۷۵ | شکل پ-۲ آماده‌سازی Rosbag برای ضبط داده‌ها                             |
| ..... ۷۶ | شکل پ-۳ جمع‌آوری داده از داخل طبقه همکف دانشکده کامپیوتر               |
| ..... ۷۶ | شکل پ-۴ جمع‌آوری داده بیرون از محوطه‌ی دانشکده                         |

صفحه

فهرست نمودارها

|    |  |
|----|--|
| ۵۵ | نمودار ۱-۵ تخمین مکان خودرو در سطح شهر با حسگر GPS                   |
| ۵۷ | نمودار ۲-۵ تخمین مکان خودرو در سطح شهر با حسگر IMU                   |
| ۶۰ | نمودار ۳-۵ تخمین مکان خودرو در سطح شهر با حسگر لایدار                |
| ۶۲ | نمودار ۴-۵ تخمین مکان خودرو با ترکیب حسگرهای GNSS و IMU              |
| ۶۳ | نمودار ۵-۵ تخمین مکان خودرو با ترکیب حسگرهای GNSS و IMU و لایدار     |
| ۶۵ | نمودار ۶-۵ مکانیابی با لایدار در دانشکده‌ی کامپیوتر دانشگاه امیرکبیر |

## فهرست علائم

### علائم لاتین

|                                       |             |
|---------------------------------------|-------------|
| مختصه در راستای افق                   | x           |
| مختصه در راستای عمود به صفحه          | y           |
| مختصه در راستای عمودی و بیانگر ارتفاع | z           |
| شرایط اولیه                           | $x_0$       |
| حالت پیش‌بینی شده                     | $\check{x}$ |
| حالت تصحیح شده                        | $\hat{x}$   |

### علائم یونانی

|                                     |          |
|-------------------------------------|----------|
| سرعت زاویه‌ای                       | $\omega$ |
| زاویه‌ی قرارگیری جسم در صفحه‌ی افقی | $\theta$ |

## فصل اول

### مقدمه

## مقدمه

امروزه به تکنولوژی خودروهای خودران، به عنوان یک راه امیدوارکننده و در عین حال چالش برانگیز برای کاهش تصادفات جاده‌ای و همچنین بهبود شرایط ترافیکی نگاه می‌شود. تکنولوژی مذکور، امکاناتی نظیر حسگرهای مختلف و دوربین‌ها و همچنین روش‌های هوش مصنوعی را به کار می‌گیرد تا بتواند تردد خودروی بدون راننده را امکان‌پذیر سازد و این امر موجب می‌شود تا خطاهای انسانی در رانندگی کاهش یابد.

اصلی‌ترین و مهم‌ترین ویژگی خودروهای خودران، توانایی خودرو در شناخت محیط اطراف، حرکت و رانندگی در مسیری مشخص و بدون دخالت انسان است. این خودروهای هوشمند می‌توانند به صورت خودکار در خیابان‌ها تردد کنند، سرعت خود را بنابر موانعی مانند ترافیک، حضور عابر پیاده و دیگر موانع فیزیکی افزایش یا کاهش دهنند، در زمان نیاز به درستی بپیچند یا توقف کنند و در نهایت بدون هیچ مشکلی به مقصد تعیین شده برسند. خودرو برای دستیابی به این اهداف با استفاده از چند دوربین، انواع سنسور نصب شده روی خودرو و با کمک هوش مصنوعی محیط اطراف و وضعیت جغرافیایی را بررسی کرده و موانع را تشخیص می‌دهد. در حقیقت هوش مصنوعی به عنوان مغز خودرو عمل کرده و سنسورها و دوربین‌ها وظیفه چشم راننده را بر عهده دارند. در این پژوهش نیز سعی شده است گامی مهم در این عرصه برداشته شود، که در ادامه به صورت کامل‌تر به آن پرداخته شده است.

### ۱-۱- هدف پژوهش

برای آنکه وسیله نقلیه خودران به صورت ایمن در محیط‌های شهری رانندگی کند، لازم است تا موقعیت دقیق خود را بداند. مکان‌یابی نقشی اساسی در بسیاری از کاربردها مانند مسیریابی ربات‌ها یا تخمین مکان خودروهای خودران دارد. امروزه با پیشرفت حسگرهای ظهور ا نوع جدیدی از آن‌ها و همچنین ابداع الگوریتم‌های پیشرفته‌تر تخمین دقیق‌تر مکان خودرو امکان‌پذیر شده است. امروزه ساده‌ترین روشی که

برای یافتن مکان یک جسم متحرک به نظرمان می‌رسد، استفاده از سیستم ناوبری ماهواره‌ای جهانی<sup>۱</sup> می‌باشد. این روش قطعاً از بهترین روش‌هایی می‌باشد که بشر برای یافتن مکان اجسام تاکنون به کار گرفته است ولی همچنان نقاط ضعفی دارد که استفاده‌ی تنها از آن، برای بسیاری از کاربردها و مسائل در زمینه‌ی رباتیک و خودروهای خودران، نامناسب می‌باشد.

در این پژوهش سعی بر آن است تا با شناسایی روش‌ها و حسگرهای موجود، بتوانیم راه حلی برای تخمین مکان یک خودروی خودران به صورت دقیق‌تر و همچنین به صورت بلادرنگ، بیابیم. سپس با کمک الگوریتم و برنامه‌ی طراحی شده در محیط شبیه‌ساز، به آزمایش عملی آن روش در محیط دانشگاه، بپردازیم. برای درک بهتر مسئله، لازم است تا در این بخش، به صورت مختصر با سنسورها و مشکلاتی که در مکان‌یابی با آن‌ها وجود دارد آشنا شویم تا در بخش‌های آتی به ابعاد دیگر پژوهش بپردازیم.

## ۱-۲- بیان صورت مسئله

به صورت کلی، دو روش برای مکان‌یابی یک ربات یا خودروی متحرک داریم: روش محلی<sup>۲</sup> و روش جهانی<sup>۳</sup>. در ادامه هریک را با ذکر مثالی از آن توضیح می‌دهیم.

**روش محلی:** در این روش، با دانستن یک موقعیت اولیه‌ی تقریبی از ربات، سعی می‌کنیم تا به کمک داده‌های دریافت شده از حسگرها در دستگاه مختصات محلی خودشان، جابه‌جایی و چرخش ربات را نسبت به داده‌های قبلی دریافت شده، بیابیم و با جمع مقادیر آن‌ها با یک‌دیگر، قادر باشیم تا مکان فعلی ربات و مسیر پیموده شده توسط آن را تخمین بزنیم. در این روش، چنانچه دنبال کردن مسیر ربات با مشکلی مواجه شود و یا داده‌های سنسور به هر دلیلی از دست برود، ربات قادر به بازیابی موقعیت درست خود نمی‌باشد و خطای مکان‌یابی به صورت تجمعی، زیاد می‌شود[۱].

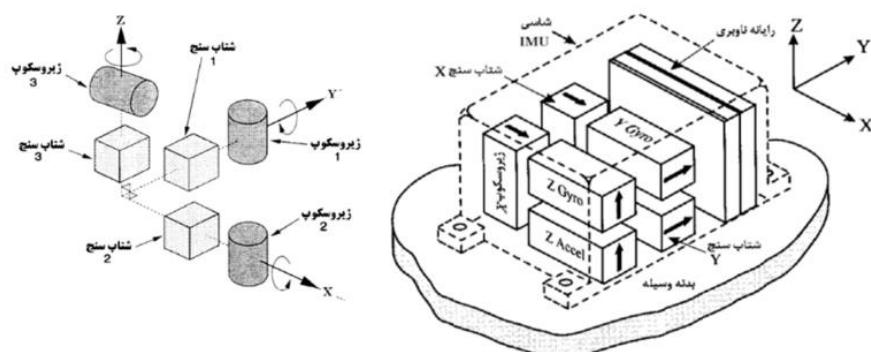
از جمله کارهای انجام شده در روش محلی، کیلومترشماری<sup>۱</sup> به کمک واحد اندازه‌گیری لختی (IMU)<sup>۲</sup> می‌باشد. ناوبری لختی، از انواع روش‌های محاسبه مسیر می‌باشد که بر اساس اندازه‌گیری شتاب حرکت

<sup>1</sup> Global Navigation Satellite System (GNSS)

<sup>2</sup> Local

<sup>3</sup> Global

جسم عمل می‌کند؛ به این صورت است که از شتاب برای تعیین سرعت و مجدد از سرعت برای تعیین موقعیت جسم، انتگرال گیری می‌کند. با توجه به اینکه در ساختار سیستم ناوبری لختی<sup>۳</sup> از ژیروسکوپ نیز استفاده می‌شود، در کنار تعیین متغیرهای حرکتی جسم میتوان وضعیت (سمت‌گیری) جسم را نیز تعیین کرد. لذا یکی از مزیت‌های روش ناوبری لختی توانایی آن در تعیین توازن موقعیت و وضعیت جسم می‌باشد. برتری دیگر آن، قابلیت تعیین متغیرهای حرکتی هم در دستگاه مختصات اینرسی و هم دستگاه جغرافیایی می‌باشد. اگر مجموعه سنسورهای لختی (شامل ژیروسکوپ‌ها و شتاب‌سنج‌ها) به صورت صلب بر روی بدن وسیله نقلیه نصب شوند، به مجموعه‌ی آن، واحد اندازه‌گیری لختی گفته می‌شود و سیستم ناوبری را در این صورت سیستم ناوبری لختی بدون صفحه پایدار گویند. به کمک حسگر IMU میتوان میزان جابه‌جایی نسبی را با توجه به مدل حرکتی محاسبه کرد و با دانستن مکان اولیه‌ی خودرو و جمع کردن جابه‌جایی‌های نسبی با آن، موقعیت خودرو را در هر لحظه محاسبه کرد. یکی از نقاط مهم سیستم‌های ناوبری لختی نامحدود بودن خطای آن‌ها با زمان و انباست خطا می‌باشد که باعث شده است عموماً در کنار آن‌ها از سیستم‌های ناوبری کمکی غیر لختی استفاده شود. این نامحدود بودن خطای در سیستم‌های ناوبری لختی به گونه‌ای است که با گذشت کمتر از چند دقیقه میزان خطا بیش از دو برابر می‌شود و رشد خطا به صورت نمایی است. همچنین افزایش دقت این حسگر، افزایش هزینه، وزن و حجم آن را در پی خواهد داشت<sup>[۲]</sup>. در شکل ۱-۱ نمایی از ساختار درونی این حسگر نمایش داده شده است.



شکل ۱-۱ نمایی از حسگر INS و نمای داخلی آن.

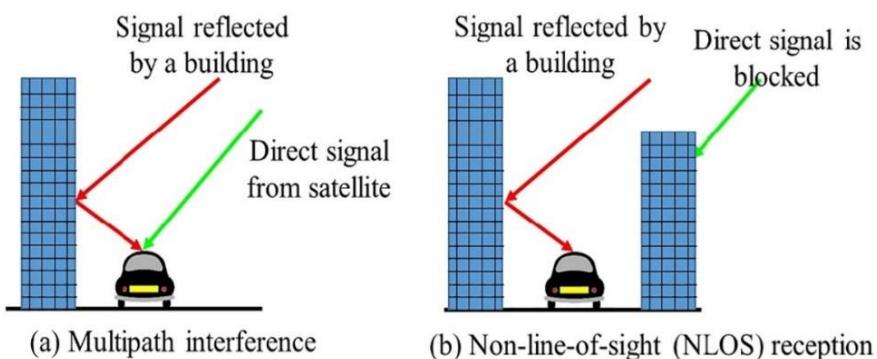
<sup>1</sup> Odometry

<sup>2</sup> Inertial Measurement Unit (IMU)

<sup>3</sup> Inertial Navigation System (INS)

**روش جهانی:** در روش جهانی، می‌توان مکان یک ربات را بدون داشتن دانشی قبلی از موقعیت ربات، تخمین زد. بنابراین در این روش وابسته به دنبال کردن اطلاعات دریافت شده از سنسور نیستیم و در هر زمان که نیاز باشد، می‌توانیم به کمک این روش مکان ربات را تعیین نماییم. روش‌های مکان‌یابی جهانی بسیار پر قدرت‌تر از روش‌های محلی هستند و در موقعي که خطاهایی در مکان‌یابی ربات پیش بیاید، بهتر می‌توانند شرایط را کنترل نمایند و مکان درست ربات را بازیابی کنند.

سیستم ناوبری ماهواره‌ای جهانی که پیشتر به آن اشاره شد، نمونه‌ای از روش‌های موقعیت‌یابی جهانی می‌باشد. این سیستم، شبکه‌ای متشکل از ماهواره‌ها می‌باشد که اطلاعات زمانی و مکانی را به زمین ارسال می‌کند. برای مکان‌یابی در این سیستم، اطلاعات دریافت شده از حداقل چهار ماهواره لازم است و این اطلاعات توسط گیرنده‌های مربوطه ثبت می‌شوند و برای تعیین موقعیت اجسام دارای این گیرنده‌ها مانند خودروها، مورد استفاده قرار می‌گیرند. گیرنده شامل آنتن و واحد پردازش می‌باشد که واحد پردازش از طریق محاسباتی پیچیده بر روی سیگنال‌های دریافت شده از ماهواره‌ها، موقعیت خودرو را به دست می‌آورد. با این حال، ممکن است حتی با یک سیستم پیشرفته مبتنی بر GNSS، مکان دقیق یک وسیله نقلیه با مکان تخمین زده شده توسط آن، تا چند متر تفاوت داشته باشد؛ زیرا ممکن است که ماهواره‌های مربوطه از میدان دید گیرنده خارج شوند و یا در محیط‌های شهری به علت وجود ساختمان‌های بلند یا تونل‌ها، خط دید گیرنده مسدود شود و یا پدیده چند مسیری<sup>۱</sup> رخ دهد که درواقع ترکیبی از خط دید و غیر خط دید می‌باشد که یک یا چند بار قبل از رسیدن به آنتن گیرنده از موانع نزدیک بازتاب می‌شوند [۲، ۳]. در شکل ۲-۱ مشکل ذکرشده به خوبی نمایش داده شده است.



شکل ۲-۱ پدیده‌های چند مسیری و NLOS از معایب استفاده از GNSS [۴].

<sup>1</sup> Multipath

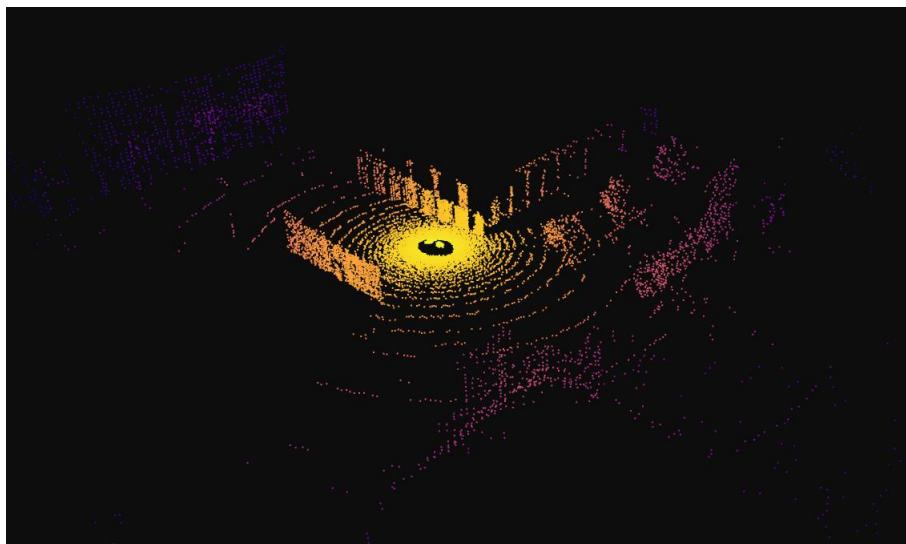
از جمله مشکلات مهم دیگر سنسور GPS به خصوص در صنعت خودروهای خودران، آن است که این حسگر، فرکانس به روزرسانی پایینی دارد. این فرکانس در حدود ۱ هرتز می‌باشد و چنانچه خودرویی با سرعت ۱۰۰ کیلومتر بر ساعت حرکت کند، آنگاه پس از طی کردن مسافتی به اندازهٔ حدوداً ۳۰ متر، مجدداً مکان خودرو توسط این حسگر به روزرسانی می‌شود. برای یک خودروی کاملاً خودران این عدد مقدار کاملاً غیرقابل قبول می‌باشد و می‌تواند منجر به تصادف وسیله‌ی نقلیه شود.

باتوجه به آنچه ذکر شد، استفاده از هریک از سنسورهای IMU یا GPS به تنها یکی مشکلات فراوانی دارد. بنابراین در تحقیقاتی به ترکیب داده‌های این دو حسگر با یک دیگر پرداختند تا هریک بتوانند نواقص دیگری را تا حدی بپوشانند. با این وجود، در گذر از داخل تونل‌ها و یا پارکینگ‌هایی که برای مدت قابل ملاحظه‌ای دسترسی به حسگر GPS مختل می‌شود؛ به علت خطای تجمعی که در روش کیلومتر شماری با IMU داریم، بازهم ممکن است مکان‌یابی خودرو با مشکل مواجه شود. در این پژوهش به دنبال آن هستیم که راه حلی برای پر کردن خلاصهٔ ذکر شده بیابیم.

### ۱-۳- بیان راه حل

برای مسئله‌ی بیان شده، راه حلی که در این پژوهش به کار گرفته شده‌است، استفاده از حسگرهای نوین جهت بالابردن دقت مکان‌یابی می‌باشد. امروزه باتوجه به پیشرفت در ساخت حسگرهای بینایی می‌توان از آن‌ها برای کیلومترشماری و مکان‌یابی بهره برد. اولین حسگر بینایی که به ذهن می‌رسد، دوربین خواهد بود. دوربین‌ها نیز محدودیت‌هایی دارند از جمله آنکه در تاریکی عملکرد خوبی ندارند. همچنین می‌دانیم که دوربین‌های معمولی تصاویری دو بعدی از محیط تهیه می‌کنند و نمی‌توان به کمک آن‌ها درک مناسبی از بعد سوم (ارتفاع یا عمق) پیدا کرد. برای رفع مشکلاتی که دوربین‌های دو بعدی ایجاد می‌کنند، ربات‌ها یا خودروهای خودران را به حسگر لایدار مجهز می‌نماییم. لایدار یک حسگر نوری فعال است که می‌تواند اندازه گیری بسیار دقیقی را از فاصله ارائه دهد که در آن خطاهای معمولاً بدون توجه به فاصله ثابت هستند. لایدار با سرعت خیلی زیادی، می‌تواند تمامی سطوح اطراف خود را اسکن کرده و پستی و بلندی‌های اجسام محیطی را با دقت خیلی خوبی نشان دهد. همچنین این کار را با ارسال پرتوهای لیزری به جسم و ضبط بازتاب و پراش‌های پرتو ارسالی انجام می‌دهد و در نتیجه اطلاعات کاملی در خصوص سطح و عمق و همچنین فاصله جسم تا منبع را از طریق محاسبه زمان رفت و

برگشت پرتو نور به دست می‌آورد. به خروجی حاصل از یک دور چرخش کامل لایدار به دور خود یک اسکن می‌گوییم که نمونه‌ای از آن در شکل ۳-۱ آمده است. در مرحله‌ی بعد به کمک الگوریتم‌های ثبت ابر نقاط<sup>۱</sup> می‌توان ماتریس تبدیل بین هر دو اسکن متوالی را یافت و سپس به کمک ماتریس مذکور بردار جابه‌جایی بین هردو اسکن متوالی را پیدا کرد. در نهایت با دانستن موقعیت اولیه‌ی خودرو و جمع کردن مجموع جابه‌جایی‌های به دست آمده با آن، می‌توان موقعیت خودرو را در هر زمان با دقت نسبتاً بالایی به دست آورد[۵].



شکل ۳-۱ نمونه‌ای از اسکن ثبت شده توسط حسگر لایدار.

لازم به ذکر است که روش کیلومترشماری انجام گرفته با کمک لایدار دقت بسیار بالاتری نسبت به حسگر IMU دارد و درواقع خطای تجمعی آن کمتر خواهد بود. همچنین از نکات مثبت دیگر آن، فراهم آوردن بینایی برای خودرو و همچنین انجام الگوریتم مکان‌یابی به کمک آن، در شب و در تاریکی می‌باشد و همچنین تغییرات آب و هوایی در دقت الگوریتم تاثیر بسیار ناچیزی دارد.

#### ۴-۱- تاریخچه

مکان‌یابی یکی از توانایی‌های اساسی و مهم در اکثر وسیله‌های هوشمند به حساب می‌آید. با گذشت زمان و پیشرفت تکنولوژی در حوزه‌ی حسگرها، زمینه‌های تحقیقاتی مختلفی مهیا شد و روش‌های

<sup>۱</sup> Point cloud registration

جدیدی برای مکان‌یابی دقیق‌تر ابداع شد. می‌دانیم که در یک ربات یا خودروی خودران حسگرهای متفاوتی برای درک هرچه بهتر محیط استفاده می‌شود. به مرور برای بهبود توانایی مکان‌یابی سعی بر آن شد تا از اطلاعات جمع‌آوری شده توسط این حسگرها نیز در کنار حسگرهایی از قبیل سیستم موقعیت‌یاب جهانی، استفاده کنند.

در سال‌های گذشته الگوریتم‌های مکان‌یابی بسیاری برای حسگرهای بصری مختلف توسعه یافته است. چندین الگوریتم مانند LIMO و ... مکان را به کمک یک دوربین تک چشمی یا استریو تخمین می‌زنند و همچنین از ابرنقاط به دست آمده توسط لایدار، برای پشتیبانی و بهبود دقت استفاده می‌کنند [۶, ۷].

استفاده از لایدار به تنها یی دشوار است و مکان‌یابی را با خطا مواجه می‌سازد؛ زیرا لایدار به طور مداوم در حال حرکت است و اعوجاج حرکتی در ابرهای نقطه‌ای وجود دارد که در سرعت‌های بالای خودرو به میزان بیشتری قابل مشاهده است. یکی از راه‌های کم کردن این خطای ایجاد شده ترکیب اطلاعات به دست آمده از سنسورهای مختلف با آن می‌باشد. به عنوان مثال، سیستم ناوبری Scherer و همکاران [۸] از فاصله سنجی بصری استریو ادغام شده با IMU به همراه ابرنقاط به دست آمده از لایدار برای تخمین مکان وسیله نقلیه‌ی مورد نظرشان استفاده می‌کند.

امروزه به علت بالا بودن هزینه‌ی استفاده از دوربین‌ها و همچنین عدم وجود دقت کافی در مکان‌یابی به علت وابستگی به میزان روشنایی و سایر عوامل محیطی، تمایل به استفاده از حسگر لایدار در این زمینه بیشتر شده است. خروجی حسگر لایدار در هر فریم یک اسکن از ابر نقاط می‌باشد که برای تطابق دو اسکن متوالی و به دست آوردن ماتریس تبدیل میان آن دو لازم از الگوریتم‌های ثبت ابر نقاط استفاده کیم. از جمله مهم‌ترین روش‌ها می‌توان به ICP و NDT اشاره کرد.

پیر دلناخ و همکاران [۹]، با استفاده از لایدار به تنها یی و با به کارگیری روش CT-ICP توانستند تا به صورت بلادرنگ بر روی دیتاستهای موجود، مکان‌یابی را با دقت قابل قبولی انجام دهند. هنگامی که نرخ اسکن لایدار در مقایسه با جایه‌جایی آن در محیط بالا باشد، اعوجاج و اختلاف حرکت در اسکن‌ها اغلب نادیده گرفته می‌شود. به این ترتیب در این مقاله با به کارگیری روش مبتنی بر ICP توانستند به دقت مطلوب برسند.

بوجو و همکاران [۱۰]، از تطابق اسکن‌ها به کمک روش NDT به دقت مطلوبی رسیدند. آن‌ها از INS با فرکانس بالا برای تخمین مکان اولیه استفاده میکردند و از مکان یابی مبتنی بر لایدار با فرکانس پایین‌تر برای جلوگیری از انباشت خطأ در طول حرکت ربات، بهره میبردند.

لازم به ذکر است که همیشه در پی تطابق دو اسکن متوالی نیستیم. می‌توان از تطبیق یک اسکن با نقشه نیز بهره برد که در بسیاری از پژوهش‌ها نشان داده شده است که دقت به مراتب بالاتری را دریافت خواهیم کرد و با نام مکان یابی و نقشه برداری به صورت همزمان<sup>۱</sup> (SLAM) شناخته می‌شود. برای این منظور می‌توان دو رویکرد کلی را مدنظر قرار داد. در حالت اول یک نقشه‌ی ژئورفرنس شده که مکان تمامی نقاط آن نقشه در مختصات جهانی مشخص می‌باشد را داریم و با تطبیق اسکن با نقشه می‌توان به صورت نسبتاً دقیقی به مکان خودرو دست یافت. در حالت دوم برای مثال می‌توان هر ده اسکن متوالی را ذخیره کرد و به دستگاه مختصات جهانی انتقال داد و سپس اسکن یازدهم را با ده اسکن قبلی تطبیق داد و مکان فعلی خودرو را به دست آورد. این امر به دلیل مترادف‌تر بودن ابرنقاط نتیجه‌ی بهتری را بر می‌گرداند. شوبین چن و همکاران [۱۱]، با به کارگیری تکنیک SLAM و همچنین استفاده از روش wNDT برای ثبت ابر نقاط، توانستند به صورت بلاذرنگ و با دقت بالایی، مکان یابی مبتنی بر لایدار را انجام دهند.

یک رویکرد پیشرفته‌ی اخیر در فاصله سنجی با استفاده از حسگرهای لایدار، مکان یابی و نقشه سازی به صورت بلاذرنگ مبتنی بر لایدار است که ویژگی‌هایی را که روی لبه‌های تیز و صفحات مسطح هستند با استفاده از اطلاعات جابجایی اسکن استخراج می‌کند. همچنین، داده‌های واحدهای IMU را برای بهبود دقت تخمین ادغام می‌کند [۳].

همانطور که پیش‌تر نیز اشاره شد، در خیلی از موارد ناچاریم که از ترکیب اطلاعات حسگرهای مختلف برای مکان یابی استفاده کنیم و این امر موجب افزایش دقت نیز می‌شود. برای ترکیب اطلاعات حسگرهای روش‌های مختلفی وجود دارد. از معروف‌ترین آن‌ها می‌توان به فیلتر کالمن اشاره کرد که در پروژه‌ی آپولوی ناسا نیز برای تخمین مکان و یافتن مسیر به کار گرفته شد. مونتمرو و همکاران [۱۲]، از روش فیلتر کالمن توسعه یافته برای تخمین مکان یک ربات استفاده کردند. به کمک این روش می‌توان از

<sup>۱</sup> Simultaneous Localization and Mapping (SLAM)

ترکیب اطلاعات سنسورهای مختلف مانند GNSS، IMU و ... به خطای کمتری در تخمین موقعیت دست یافت.

باتوجه به توضیحات داده شده در نهایت الگوریتمی خواهیم داشت که با دقت قابل قبولی مکان خودرو را به ما می‌دهد. لذا می‌توان آن را به عنوان گامی مهم در راستای پیاده‌سازی به صورت تجربی در یک ربات یا خودروی کوچک دانست که قادر خواهد بود به کمک حسگر لایدار و سایر حسگرهای موردنیاز و با داده برداری در محیط دانشگاه، مکان خود را مشخص نماید.

در ادامه، در رابطه با نرم‌افزارهای شبیه‌ساز و روش‌های تعیین مکان خودرو در این شبیه‌سازها صحبت خواهیم کرد. سپس به روش‌های ترکیب داده‌های حسگرهای مختلف می‌پردازیم و در نهایت نتایج به دست آمده در شبیه‌ساز و همچنین آزمایش عملی را تشریح خواهیم کرد.

## فصل دوم

### نرم افزارهای مورد استفاده

## نرم افزارهای مورد استفاده

در این پژوهه قصد داریم تا در ابتدای کار، الگوریتم‌های موردنظر را توسعه دهیم و از نرم افزارها و شبیه‌سازهای مربوطه برای آزمایش آن‌ها کمک بگیریم. همچنین پس از آنکه گام شبیه‌سازی را به خوبی پشت سر نهادیم، لازم است تا به صورت عملی کار خود را صحت سنجی نماییم. برای بخش عملی لازم است تا به نحوی بتوانیم به سنسور لایدار متصل شویم و از آن داده برداری نماییم و به صورت بلاذرنگ عمل مکان یابی را نیز اجام دهیم. بنابراین ضروری است تا با ابزارهای مورد استفاده در این پژوهش، آشنا شویم. از شبیه‌ساز کارلا برای بخش شبیه‌سازی و از نرم افزار راس و آستر استادیو برای بخش عملی استفاده کردیم که در این فصل به آن‌ها می‌پردازیم.

### ۱-۲- شبیه ساز کارلا<sup>۱</sup>

کارلا یک شبیه‌ساز متن باز<sup>۲</sup> می‌باشد که از ابتدا به صورت ماژولار ساخته شد و دارای رابطهای برنامه‌نویسی کاربردی<sup>۳</sup> متنوعی به منظور رسیدگی به طیف وسیعی از مشکلات موجود در رانندگی خودروهای خودران می‌باشد. بنابراین یکی از اهداف اصلی کارلا، بهبود تحقیق و توسعه در زمینه‌ی خودروهای خودران است تا به عنوان ابزاری عمل کند که امکان دسترسی و سفارشی‌سازی آن را توسط کاربران، فراهم می‌سازد. با توجه به اینکه پیاده‌سازی ایده‌های مختلف در حوزه‌ی خودروهای خودران به صورت واقعی ممکن است هزینه‌بر باشد و همچنین در صورت عدم عملکرد مناسب باعث بروز خطأ و حادثه شود، در این حوزه تا حد ممکن از شبیه‌ساز استفاده می‌شود. در این راستا، شبیه‌ساز مورد استفاده باید نیازمندی‌هایی از جمله توانایی یادگیری قوانین راهنمایی و رانندگی توسط خودرو، امکان اجرای الگوریتم‌های درک محیط و ... را برآورده سازد.

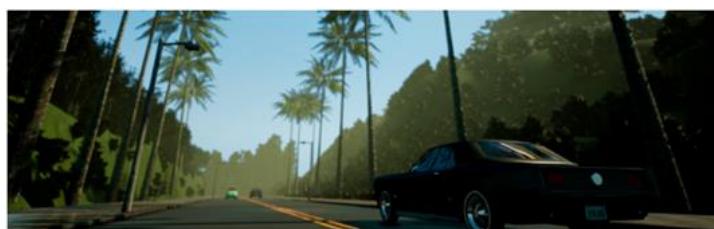
کارلا بر روی موتور فیزیکی Unreal Engine پیاده سازی شده است تا بتواند شبیه ساز را اجرا کند و همچنین از استاندارد OpenDRIVE برای تعریف جاده‌ها و تنظیمات شهری استفاده می‌کند. کاربر از

<sup>1</sup> Carla Simulator

<sup>2</sup> Open source

<sup>3</sup> Application Programming Interface (API)

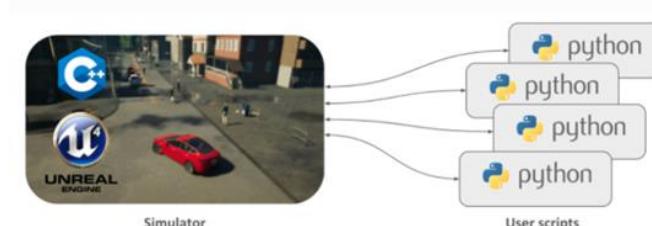
طریق رابطهایی که به کمک زبان‌های برنامه نویسی پایتون و C++ پیاده‌سازی شده‌اند، بر روی شبیه‌ساز کنترل دارد. در شکل ۱-۲ می‌توانید تصویری از محیط این شبیه‌ساز را مشاهده نمایید. در ادامه به توضیح کامل‌تری درباره‌ی این شبیه‌ساز می‌پردازیم.



شکل ۱-۲ تصویری از محیط شبیه‌ساز کارلا [۱۳].

### ۱-۱-۲- معماری کارلا

کارلا از معماری مشتری-سرمیس‌دهنده<sup>۱</sup> و به صورت مقیاس پذیر تشکیل شده‌است. سرویس‌دهنده مسئول تمامی امکانات مربوط به شبیه‌سازی؛ از جمله، ارائه و شبیه‌سازی حسگرهای محاسبه‌ی فیزیک، به روزرسانی حالت‌ها و عملگرهای ... می‌باشد. از آنجایی که در این شبیه‌ساز، هدف آن است که نتایج به واقعیت نزدیک باشد، بنابراین اجرای سرویس‌دهنده به کمک یک واحد پردازش گرافیکی اختصاصی می‌باشد و نیاز به این ویژگی در زمانی که با الگوریتم‌های یادگیری ماشین سر و کار داریم، به صورت به خصوصی احساس می‌شود. بخش مشتری، از مجموعه‌ای از ماثوله‌ای برای کنترل منطق عملگرهای شرایط محیطی در شبیه‌ساز تشکیل شده است. همانطور که اشاره شد، این امر به کمک رابط برنامه نویسی کاربردی کارلا انجام می‌گیرد که به عنوان لایه‌ای واسط بین سرویس‌دهنده و مشتری قرار می‌گیرد. برای درک بهتر این لایه می‌توانید به شکل ۲-۲ مراجعه نمایید.



شکل ۲-۲ نحوه تعامل کد کاربر در قسمت مشتری با شبیه‌ساز اجرا شده در سرویس‌دهنده [۱۳].

<sup>۱</sup> Client-Server

## ۲-۱-۲- اجزای مختلف کارلا

در شبیه‌ساز کارلا مفاهیم و ماذول‌های متعددی داریم. در یک دسته بندی می‌توان به چهارتا از مهم‌ترین قسمت‌های این شبیه‌ساز اشاره کرد:

**۱- جهان<sup>۱</sup> و مشتری:** مشتری در حقیقت یک ماذولی می‌باشد که کاربر با اجرا کردن آن، می‌تواند اطلاعات لازم را از سرویس دهنده دریافت نماید و با دستورات تعریف شده، می‌تواند تغییرات دلخواه را در محیط شبیه‌ساز اعمال نماید. یک مشتری با تعریف یک آدرس آی‌پی<sup>۲</sup> و یک پورت مخصوص اجرا می‌شود. ارتباط آن با سرویس دهنده نیز از طریق ترمینال شکل می‌گیرد. همچنین این امکان وجود دارد که مشتری‌های زیادی به صورت همزمان اجرا شوند که فهم نحوه کارکرد آن نیازمند آن است که با مفاهیم همگام سازی در کارلا آشنا باشید.

جهان نیز یک شیء تعریف شده در محیط شیء‌گرای شبیه‌ساز می‌باشد که در حقیقت معرف شبیه‌ساز می‌باشد. به عنوان یک لایه‌ی انتزاعی عمل می‌کند که در بردارنده‌ی روش‌های اصلی برای ایجاد بازیگران<sup>۳</sup>، تغییردادن وضعیت آب و هوای گرفتن وضعیت فعلی جهان و ... می‌باشد. در هر شبیه‌سازی فقط می‌توان یک جهان را تعریف کرد و می‌تواند با تغییر نقشه‌ی جهان از بین برود و یا با یک جهان جید جایگزین شود.

**۲- بازیگران و طرح‌ها<sup>۴</sup>:** بازیگر هر آنچیزی می‌باشد که یک نقشی را در محیط شبیه‌ساز بازی می‌کند. برای مثال:

- وسایل نقلیه
- پیاده‌روها
- حسگرها
- تماشاگرها
- عالم راهنمایی و رانندگی و چراغ‌های راهنمایی

طرح‌ها، مدل‌های اولیه‌ی بازیگران هستند که از قبل ساخته شده‌اند و برای تولید یک بازیگر، استفاده از آن‌ها ضروری می‌باشد و درواقع، مدل‌هایی با اینیمیشن و مجموعه‌ای از ویژگی‌ها می‌باشند که

<sup>1</sup> World

<sup>2</sup> IP Address

<sup>3</sup> Actors

<sup>4</sup> Blueprints

برخی از این ویژگی‌ها را می‌توان توسط کاربر سفارشی‌سازی کرد و برخی دیگر را نمی‌توان. کتابخانه‌ی بلوپرینت حاوی تمام طرح‌های موجود و همچنین اطلاعات مربوط به آن‌ها می‌باشد.

**۳- نقشه‌ها و ناوبری<sup>۱</sup>:** نقشه درواقع یک شیء است که جهان شبیه‌سازی شده و به خصوص شهر انتخاب شده را نمایش می‌دهد. در آخرین نسخه‌ی این شبیه‌ساز، هشت نقشه برای شبیه‌سازی وجود دارد.

خیابان‌ها، خطوط و تقاطع‌ها به کمک رابطه‌های برنامه‌نویسی پایتون از طریق برنامه‌ی اجرا شده در سمت مشتری، قابل مدیریت می‌باشند. این موارد در کنار کلاس نقطه راه<sup>۲</sup> برای مهیا ساختن وسیله‌ی نقلیه و مسیر ناوبری آن، استفاده می‌شود.

به علائم و چراغ‌های راهنمایی به کمک اشیائی با نام Carla.Landmark می‌توان دسترسی داشت. علاوه براین، شبیه‌ساز به صورت خودکار علائم ایست، میدان‌ها و چراغ‌های راهنمایی را به کمک اطلاعات موجود در فایل OpenDRIVE می‌سازد. درواقع هر کدام از این موارد یک مرزمحصور‌کننده به دور خود دارند و که وقتی وسیله‌ی نقلیه در این مرز وارد می‌شود، از حضور آن شیء آگاه می‌شود.

**۴- حسگرها و داده‌ها:** حسگرها در حالت انتظار خواهند بود تا اتفاقی که برایشان تعریف شده است رخ دهد و از محیط شبیه‌ساز داده را جمع آوری نمایند. همچنین می‌توان برای آن‌ها تابعی را تعریف کرد که چگونه با داده‌ی به دست آمده برخورد نمایند. براساس آنکه نوع حسگر چیست، نوع داده‌ای که بر می‌گرداند نیز فرق دارد.

همچنین حسگر، یک بازیگر می‌باشد که به یک وسیله‌ی نقلیه متصل می‌شود و به همراه آن حرکت می‌کند و از محیط پیرامونش اطلاعات را جمع آوری می‌کند. حسگرهای موجود در این شبیه‌ساز به کمک طرح‌های تعریف شده از آن‌ها در کتابخانه‌ی بلوپرینت در دسترس هستند. از انواع حسگرهای تعریف شده می‌توان به موارد زیر اشاره کرد:

- دوربین‌ها (RGB، عمقی و تقسیم معنایی)
- حسگر GNSS
- حسگر IMU
- لایدار
- تشخیص دهنده‌ی عبور از خط

<sup>1</sup> Maps and navigation

<sup>2</sup> Waypoint

- تشخیص دهنده‌ی مواد
- رادار

### ۳-۱-۲- پیاده‌سازی

در ابتدای کار لازم است تا آخرین نسخه‌ی این شبیه‌ساز را از سایت مربوطه<sup>۱</sup> بارگیری و نصب نماییم. برای این موضوع ابتدا لازم است ملزومات سخت‌افزاری را چک کنیم و سپس مناسب با سیستم عامل خود مراحل نصب را طی کنیم.

پس از نصب می‌توان به کمک ترمینال، سرویس‌دهنده‌ی شبیه‌ساز را با پورت مناسب اجرا کرد. سپس به کمک کد پایتونی که در قسمت مشتری می‌نویسیم می‌توان به سرویس دهنده متصل شد. سپس همانطور که در قسمت قبل توضیح داده شد می‌توان با توجه به واسطه‌های برنامه نویسی تعریف شده، تغییرات لازم را در شبیه‌ساز ایجاد کرد.

ما برای این پژوهش از همان شهری که به صورت پیش فرض تنظیم شده است استفاده می‌نماییم. همچنین لازم است یک وسیله‌ی نقلیه در جهان شبیه‌ساز تعریف نماییم و نحوه‌ی رانندگی آن را هم به صورت خودکار تعریف نماییم؛ یعنی، خودرو با توجه به چراغ‌های راهنمایی و خط کشی‌های خیابان به صورت خودکار در سطح شهر حرکت نماید و در صورت نیاز به صورت خودکار توقف نماید. همچنین حسگرهای GNSS و IMU و لایدار را نیز تعریف می‌کنیم و به خودرو متصل می‌کنیم. لازم است برای GNSS هریک از حسگرها نویز و سایر پارامترها از جمله فرکانس داده‌برداری را تعیین نماییم. برای IMU مقدار فرکانس را برابر با ۱ هرتز، برای IMU برابر با ۲۵۰ هرتز و برای لایدار برابر با ۱۰ هرتز تعریف می‌نماییم. سپس برای هریک از حسگرها نیز تعریف می‌نماییم که در صورت رسیدن داده‌ی مربوط به آن، چگونه آن داده را پردازش نماید و مختصات خودرو را بیابد. برای آنکه با نحوه‌ی ایجاد حسگرها و کار با آن‌ها آشنا شوید لازم است که قبل از شروع نوشتن کد، ابتدا به سراغ مثال‌های خود کارلا بروید. اصلی‌ترین بخش این پژوهش کار با حسگر لایدار می‌باشد که مثالی از آن توسط کارلا در گیت‌هاب<sup>۲</sup> قرار داده شده است.

<sup>1</sup> [https://carla.readthedocs.io/en/latest/start\\_quickstart/](https://carla.readthedocs.io/en/latest/start_quickstart/)

<sup>2</sup> [https://github.com/carla-simulator/carla/blob/master/PythonAPI/examples/open3d\\_lidar.py](https://github.com/carla-simulator/carla/blob/master/PythonAPI/examples/open3d_lidar.py)

برای کار با داده‌های لایدار و ابرنقاط، یکی از بهترین کتابخانه‌های موجود Open3d می‌باشد که لازم است آن را نیز نصب نمایید. به کمک الگوریتم ثبت نقاط موجود در این کتابخانه می‌توان به راحتی الگوریتم کیلومترشماری با کمک لایدار را پیاده‌سازی کرد.

## ۲-۲- سیستم عامل رأس<sup>۱</sup>

رأس یک سیستم عامل رایگان با قابلیت کار بر روی چند سیستم عامل برای ربات است و سرویس‌هایی که شما از یک سیستم عامل انتظار دارید را فراهم می‌کند. رأس شامل شبیه‌ساز سخت‌افزار، کنترل سطح پایین، انتقال پیام‌های بین پردازش‌ها و مدیریت پکیج‌ها می‌باشد. به علاوه رأس شامل ابزارها و کتابخانه‌هایی برای کامپایل کردن و نوشتن و اجرا کردن کد در چند کامپیوتر است. بنابراین رأس جایگزینی برای بقیه سیستم عامل‌ها نیست بلکه در کنار آن‌ها کار می‌کند.

پس از آنکه الگوریتم و کد موردنظرمان بر روی شبیه‌ساز با موفقیت عمل کرد، نوبت به آن می‌رسد که کد خود را بر روی سنسور لایدار به صورت عملی آزمایش نماییم. در این حالت، داده‌ها از حسگر می‌رسند و لازم داریم تا کد ما نیز در حال اجرا باشد و با رسیدن هریک از اسکن‌های این حسگر، الگوریتم را بر روی آن اجرا کند. درواقع در قسمت شبیه‌سازی، محیط کارلا برای ما این امکانات را به راحتی فراهم می‌کرد و تابع‌های تعریف شده برای حسگرها به صورت خودکار با رسیدن اسکن‌ها از سمت سرویس‌دهنده، اجرا می‌شوند. حال در این بخش قرار است با کمک از رأس این مشکل را نیز برطرف نماییم.

## ۱-۲- عملکرد رأس

در ابتدای شکل‌گیری رأس، به دلیل اینکه در کارهای رباتیکی نیاز به دسترسی به فایل‌های سیستمی بوده و این دسترسی در سیستم عامل ویندوز موجود نبود، تیم رأس تصمیم به بنا نهادن سیستم عامل خود بر روی لینوکس گرفت. سیستم عامل لینوکس برخلاف ویندوز، سیستم عاملی متن‌باز بوده که کد منبع‌های آن در دسترس است و به سادگی می‌توان در فایل‌های سیستمی آن تغییر ایجاد کرد.

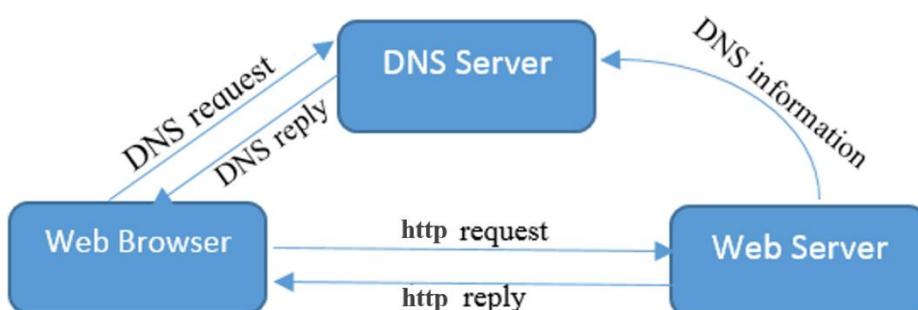
---

<sup>۱</sup> Robot Operating System (ROS)

رآس برای برقراری ارتباط بین وسایل جانبی مختلف و استفاده بهینه از حافظه و پردازنده کامپیوتر، به فایل‌های سیستمی نیاز دارد، به این جهت نمی‌توان رآس را به طور صریح در یکی از دسته‌بندی‌های نرم‌افزاری قرار داد. لذا به خاطر نزدیکی به فایل‌های سیستمی و هسته سیستم‌عامل، آن را سیستم‌عامل می‌نامند، اما از یک نگاه دیگر به دلیل مستقل نبودن آن (چرا که باید بر روی یکی از توزیع‌های سیستم‌عامل لینوکس نظیر اوبونتو نصب شود) نمی‌توان آن را سیستم‌عامل نامید. در نتیجه محیط رآس از این جهت که قابلیت برقراری ارتباط بین قسمت‌های مختلف سخت‌افزاری را دارد، را می‌توان در دسته میان افزارها قرار داد.

کتابخانه‌ها و بسته‌های نرم‌افزاری متعدد که هر روز نیز به تعداد آنها افزوده می‌شود، موجب شده که رآس به ابزاری بی‌نظیر برای کارهای رباتیکی تبدیل شود.

برای درک بهتر عملکرد رآس مقایسه‌ای بین آن و جستجوگرهای اینترنتی انجام می‌دهیم. همانطور که می‌دانید، اطلاعات یک سایت اینترنتی بر روی سرویس‌دهنده‌ی وب قرار دارد و کل این مجموعه تحت نظر سرویس‌دهنده‌ی DNS فعالیت می‌کند. به این ترتیب که هنگامی که کاربر نشانی سایت خاصی را در جستجوگر اینترنتی خود وارد می‌کند، جستجوگر با برقراری ارتباط با سرویس‌دهنده‌ی DNS نحوه ارسال درخواست برای سرویس‌دهنده‌ی وب را دریافت کرده و آن را برای سرور وب ارسال می‌کند. سرویس‌دهنده‌ی وب نیز پس از دریافت درخواست از جستجوگر، اطلاعات درخواست شده را به نمایش در می‌آورد. شما کلی این ارتباط در شکل ۳-۲ آورده شده است.



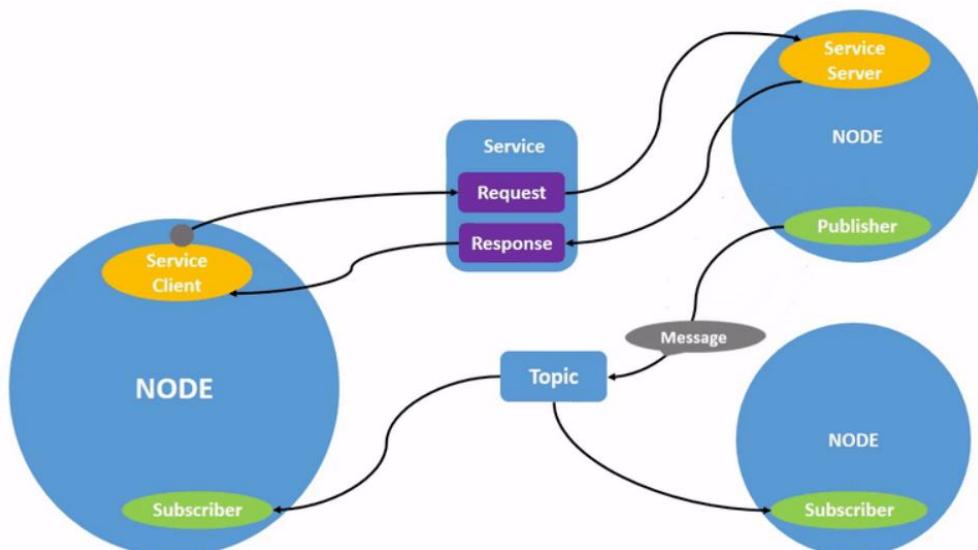
شکل ۳-۲ نحوه ارتباط جستجوگر با سرویس‌دهنده‌ی وب [۱۴].

به این ترتیب کاربر اینترنتی برای دسترسی به اطلاعات مورد نیاز خود، نیازی به دانستن هیچ یک از مطالب بالا و نحوه برقراری ارتباطات بین سرویس‌دهنده‌ی DNS، سرویس‌دهنده‌ی وب و جستجوگر و نوع داده‌های ارسالی آن‌ها ندارد.

محیط رأس نیز برای کاربران خود چنین قابلیتی را فراهم آورده است، به این ترتیب که مثلاً برای برقراری ارتباط یک حسگر و گرفتن داده از آن، کاربر نیازی به دانستن نحوه ارتباط بین قسمت‌های مختلف سخت‌افزار و نرم‌افزار سیستم ندارد و تنها باید نحوه برقراری ارتباط بین نودها در رأس را بداند. در این صورت کاربر می‌تواند داده‌های ارسالی سنسور را با فرمتی که برای داده‌های خروجی از آن نود تعریف شده دریافت کند.

## ۲-۲-۲- ساختار رأس

رأس از یک سری از اجزا و تعاریف تشکیل شده است که ساختار آن را تشکیل می‌دهد. برای کار با رأس لازم است که به خوبی با آن‌ها آشنا باشید. در شکل ۴-۲ این ساختار به خوبی نمایش داده شده است و جلوتر با هریک از این اجزا آشنا می‌شوید.



شکل ۴-۲ ساختار و چگونگی ارتباطات در رأس [۱۵].

**نود<sup>۱</sup>:** برنامه‌هایی که توسط کاربر نوشته می‌شود، برای اجرا باید به عنوان نود به رأس معرفی شود. یک نود در واقع یک فایل قابل اجرا درون یک بسته است. معمولاً برنامه نویسان برای فهم راحت‌تر و اشکال‌زدایی سریع کد، این بخش‌ها را از یکدیگر جدا می‌کنند و تا جای ممکن سعی می‌کنند نودهایی را تعریف نمایند. همچنین هر نود برای ارتباط با نود دیگر از کتابخانه‌ی مشتری استفاده می‌نماید.

**تاپیک<sup>۲</sup>:** در رأس نودها با استفاده از تاپیک‌ها با هم در ارتباط هستند. در حقیقت تاپیک‌ها مانند یک سوییچ شبکه می‌باشند که رایانه‌ها و سیستم‌های مختلف به آن متصل می‌شوند و پیام‌های آنان از این طریق برای یکدیگر ارسال می‌شود. برای این منظور یک نود نقش ناشر<sup>۳</sup> را دارد و می‌تواند پیام‌های خود را بر روی یک تاپیک منتشر نماید و یک یا چند نود دیگر به عنوان مشترکین<sup>۴</sup>، محصول و پیام تولید شده را دریافت نمایند.

**پیام<sup>۵</sup>:** پس از ساختن تاپیک و معرفی ناشر و مشترکین، لازم است تا قالب پیامی که میان آن دو مبادله می‌شود نیز مشخص شود. درواقع یک پیام نمی‌تواند هر نوع داده‌ی دلخواهی باشد و باید ابتدا ساختار آن تعریف شود و به ناشر و مشترک نیز این قالب ساخته شده را معرفی نماییم.

**سرویس:** هانطور که گفته شد، نودها توسط تاپیک‌ها با یکدیگر در ارتباط‌اند. اما نودها از طریق دیگری هم می‌توانند با یکدیگر ارتباط برقرار کنند. این راه دوم در واقع همان سرویس‌ها هستند. نودها می‌توانند درخواستی را به یک سرویس بفرستند و از آن سرویس یک پاسخی را دریافت کنند. درواقع دیگر در این حالت بزومی به همگام‌سازی وجود ندارد و هر زمان که لازم باشد یک نود درخواست را خود را می‌فرستد و پیام موردنظر خود را از سرویس تحويل می‌گیرد. همچنین در این حالت نحوه‌ی ارتباطات یک ارتباط یک به یک می‌باشد.

<sup>1</sup> Node

<sup>2</sup> Topic

<sup>3</sup> Publisher

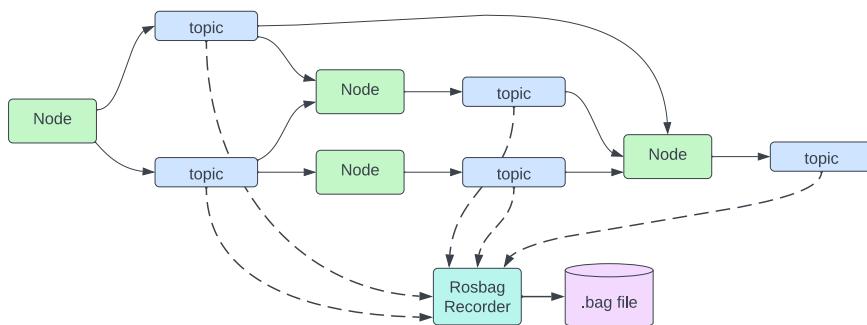
<sup>4</sup> Subscriber

<sup>5</sup> Message

### ۳-۲-۲ - ابزارهای ROSbag و Rviz

در این پژوهش قصد داریم تا با دریافت داده‌های حسگر لایدار، مکان‌یابی سیستم متحرک خود را انجام دهیم و همچنین مسیری را که جسم متحرک طی کرده است در خروجی نمایش دهیم.

برای آنکه ممکن است بخواهیم از داده‌های جمع آوری شده استفاده‌های مختلفی نماییم، به جای آنکه به صورت همزمان با داده برداری کار مکان‌یابی را انجام دهیم، سعی میکنیم یک بار این داده‌ها را جمع آوری نماییم و سپس در زمان دیگری این داده‌ها را مجدد بازپخش کرده و می‌توانیم همان کار مکان‌یابی را به صورت بلاذرنگ انجام دهیم. برای این مسئله، رأس ابزاری به نام ROSbag را معرفی کرده است که می‌توانیم آن را بر روی حالت ضبط گذاشته و داده‌هایی را که روی تاپیک‌های مختلف منتشر می‌شوند را به همراه زمان انتشارشان، در یک فایل ضبط نمود. سپس می‌توان به کمک همان ابزار، هر زمان که بخواهیم داده‌ها را بازپخش نماییم و همان تاپیک‌ها و پیام‌هایشان را شبیه‌سازی نماییم. در شکل ۵-۲ این مفهوم به خوبی نمایش داده شده است.

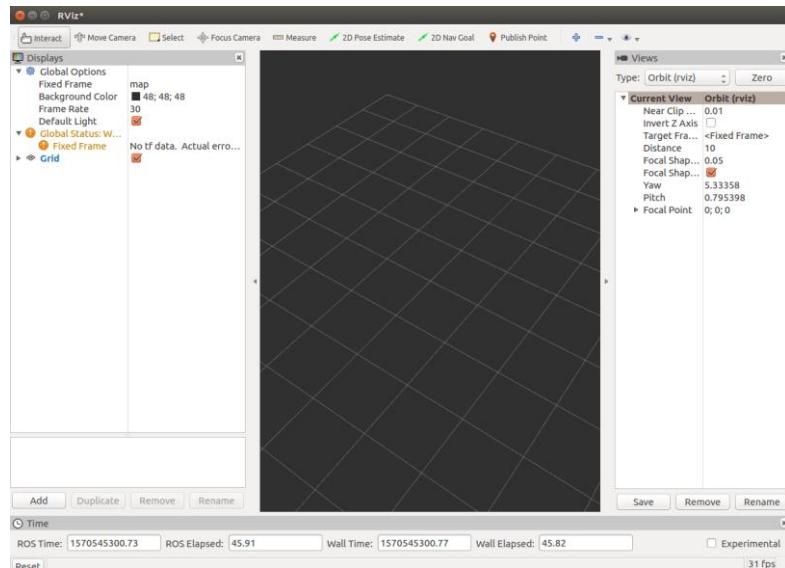


شکل ۵-۲ چگونگی عملکرد رأس-بگ [۱۶].

Rviz یک نمایشگر سه بعدی برای نمایش دیتاهای حسگرها و وضعیت قرارگیری ربات با توجه به اطلاعات رأس است. به کمک Rviz شما می‌توانید مدل ربات واقعی خود را ایجاد نمایید و وضعیت آن را در نمایشگر ببینید. همچنین می‌توانید داده‌هایی که به صورت آنی از حسگرها با استفاده از تاپیک‌ها می‌آید را نمایش دهید.

برای آنکه بتوانیم داده‌های رسیده از حسگر لایدار را به همراه مسیر تولید شده توسط الگوریتم مکان‌یابی، نمایش دهیم؛ می‌توانیم از این ابزار رأس استفاده نماییم. در شکل ۶-۲ تصویری از این

نمایشگر آمده است. لازم به ذکر است که باید به آن، تاپیکها و فریم‌های مربوطه را معرفی نماییم تا بتواند عملیات مصور سازی را به خوبی برای ما انجام دهد.



شکل ۶-۲ چگونگی عملکرد رآس-بگ.

## ۴-۲-۲ پیاده‌سازی

برای پیاده‌سازی لازم است تا پکیج ouster\_ros را نصب نماییم. این پکیج در حقیقت برای ما یک نود از حسگر لایدار متصل شده می‌سازد و داده‌های آن را در در قالب پیام‌هایی مناسب که برای رأس تعريف شده است، و همچنین در تاپیک‌های مناسب منتشر می‌کند. همچنین این پکیج با به کارگیری ROSbag، برای ما قابلیت ضبط داده‌ها و پخش مجدد آن‌ها را فراهم می‌کند.

همچنین لازم است تا خودمان یک پکیج ایجاد نماییم و نودهای موردنظر را در آن تعريف نماییم. یک نود برای انجام الگوریتم کیلومترشماری نیاز داریم. این الگوریتم را پیش‌تر در شبیه‌ساز کارلا توسعه دادیم و با تغییرات اندکی، قابل استفاده می‌شود. این نود لازم است تا بر روی تاپیک داده‌های لیزر منتشر شده، شنود نماید و هریکار که پیامی از نود سنسور می‌آید، الگوریتم را اجرا کند. سپس می‌تواند مکان به دست آمده برای ربات را روی تاپیک مکان‌یابی منتشر نماید.

یک نود نیز برای تبدیل داده‌های به دست آمده از مکان‌یابی به قالب داده‌های کیلومترشماری در رأس، نیاز است. همچنین لازم است که دستگاه مختصات متصل به فریم متحرک و فریم ثابت را تعريف نماییم.

یک نود نیز برای گرفتن داده‌های کیلومترشماری از نود قبل و تبدیل آن‌ها به داده‌ای از جنس مسیر و ذخیره‌ی آن‌ها جهت نمایش باید تعریف شود. در حقیقت وظیقه‌ی این نود ذخیره مختصات‌هایی است که تاکنون ربات ما آن‌ها را طی کرده است و در هر لحظه باید کل این مختصات‌های ثبت شده تا آن لحظه را منتشر نماید. در نهایت با بالا آمدن Rviz می‌توان تاپیک آن شنود کرد و این مسیر طی شده را نمایش داد.

همچنین یک نود هم برای نمایش مختصات‌های طی شده توسط ربات به صورت بلادرنگ خواهیم داشت و درنهایت نموداری با محورهای افقی و عمودی خواهیم داشت.

برای اجرا لازم است که داده‌های ذخیره شده در قالب فایل bag. را بازپخش نماییم و سپس نودهای موردنظر خود را اجرا نماییم که این داده‌ها را دریافت نمایند و خروجی مطلوب را برای ما تولید نمایند.

### ۳-۲- آستر استادیو<sup>۱</sup> و کیت توسعه‌ی نرم افزار<sup>۲</sup>

برای آنکه به حسگر لایدار متصل شویم لازم است تا تنظیماتی را انجام دهیم و نمونه‌ای از خروجی را مشاهده نماییم و سپس به سایر مراحل بخش پیاده‌سازی عملی از جمله داده‌برداری بپردازیم.

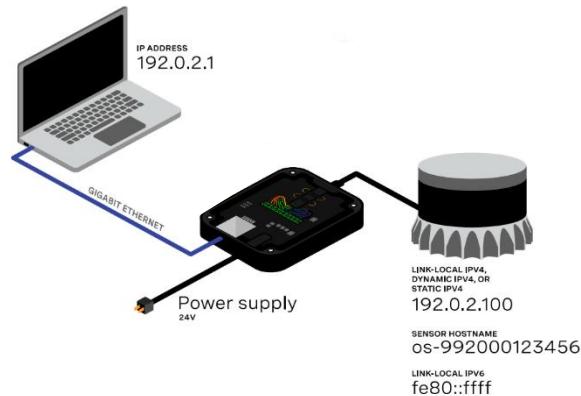
برای این منظور ابتدا باید به راهنمای استفاده از حسگر خریداری شده بپردازیم. حسگر لایدار مورد استفاده، تولید شرکت آستر می‌باشد و یک لایدار ۶۴۴ لایه است. برای اساس می‌توان راهنمای استفاده از این حسگر را دانلود و مطالعه نمود.

ابتدا باید حسگر را به برق شهری متصل کرده و مبدلی دارد که ولتاژ شهر را به ۱۲ ولت رسانیده که برای استفاده‌ی حسگر مناسب می‌باشد. سپس حسگر از طریق یک کابل LAN به لپتاپ متصل می‌شود و داده‌های خود را می‌تواند در قالب بسته‌های UDP از این طریق ارسال نماید. در شکل ۷-۲ نحوه‌ی اتصال این حسگر به لپتاپ نمایش داده شده است.

---

<sup>1</sup> Ouster-Studio

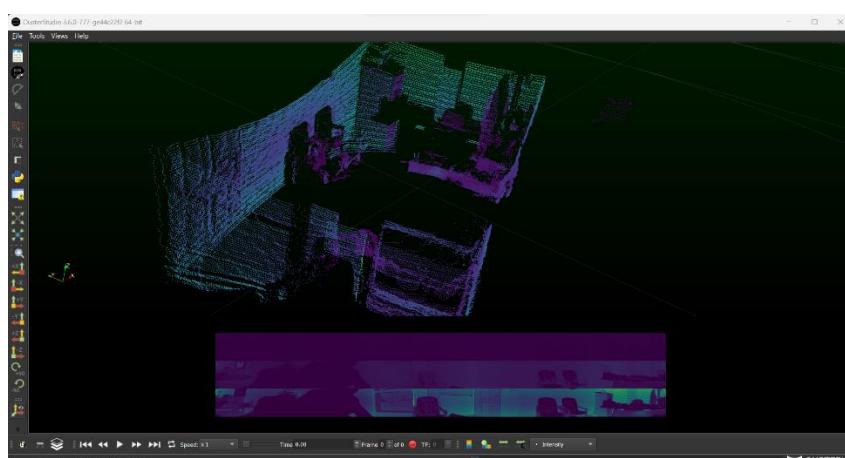
<sup>2</sup> Software Development Kit (SDK)



شکل ۷-۲ نحوه اتصال حسگر لایدار به لپتاپ [۱۷].

در گام بعدی لازم است تا کیت توسعه‌ی نرم افزار مربوط به آستر را نصب نماییم. این کیت توسط همین شرکت برای کار داده‌های به دست آمده از حسگر در سیستم عامل خارجی توسعه داده شده است. این کیت هم برای سیستم عامل لینوکس و هم ویندوز موجود می‌باشد.

پس از چک کردن برقراری اتصال به حسگر که با پینگ کردن آدرس آن و یا سرچ کردن آن در مرورگر سیستم، مشخص می‌شود؛ می‌توان به سراغ گام بعدی که کار با نرم افزار آستر استادیو است، رفت. این نرم افزار را می‌توان از سایت شرکت دانلود کرد. نرم افزار آستر استادیو یک محیط ساده برای پردازش و مصورسازی داده‌های لایدار ساخت این شرکت می‌باشد. همچنین به سادگی حسگر متصل شده را شناسایی می‌کند و داده‌های آن را دریافت می‌کند. همچنین امکان ذخیره‌سازی فایل به صورت pcap. و همچنین بازپخش آن توسط این نرم افزار وجود دارد. در شکل ۸-۲ می‌توانید نمونه‌ای از خروجی این حسگر در نرم افزار را که در دفتر دکتر جوانمردی ضبط شده است، مشاهده نمایید.



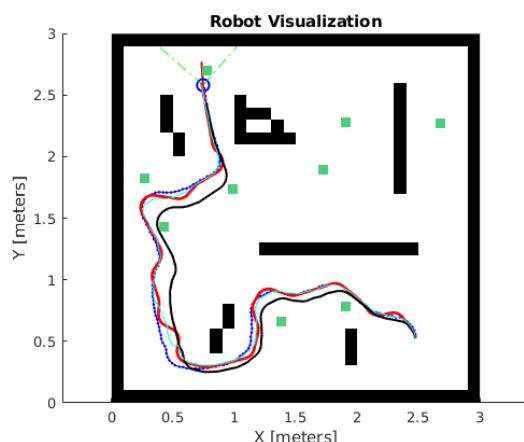
شکل ۸-۲ خروجی حسگر لایدار در آستر استادیو.

## فصل سوم

### مکان‌یابی

## مکان‌یابی

مکان‌یابی یکی از مسائل مهم و چالش برانگیز در هدایت ربات‌های متحرک و همچنین خودروهای خودران است. با توجه به اینکه از ربات‌های متحرک در مکان‌های مختلفی استفاده می‌شود؛ هنگام کار و حرکت ممکن است نیاز به دانستن موقعیت کنونی ربات باشد. برای این منظور، روش‌ها و الگوریتم‌های مختلفی به کار می‌رود. روش به کار رفته در یک ربات، با نوع محیطی که ربات در آن حرکت می‌کند، نوع ربات، سرعت حرکت ربات و همچنین نوع و اندازه ربات، متناسب است. هر یک از روش‌ها از ابزارهای گوناگونی برای موقعیت‌یابی استفاده می‌کنند و با توجه به استفاده از ابزارهای گوناگون دقت‌های متفاوتی ارائه می‌دهند. علاوه بر این موضوع، هزینه‌های هر یک از روش‌ها نیز به نسبت ابزارهای استفاده شده متغیر می‌باشد. بنابراین برای انتخاب روش مکان‌یابی باید با توجه به وسیله نقلیه و کاربرد روش مناسب را انتخاب کرد. در بسیاری از کاربردها برای تضمین سالم رسیدن ربات به مقصد و عدم برخورد آن به موانع باید مکان نسبتاً دقیق ربات را دانست. این موضوع در شکل ۱-۳ نشان داده شده است.



شکل ۱-۳ مکان‌یابی ربات و عدم برخورد آن با موانع در رسیدن به مقصد [۱۸].

مسئله به دست آوردن مکان خود ربات را مسئله مکان‌یابی و مسئله به دست آوردن توصیفی از محیط را مسئله ایجاد نقشه می‌نامیم. به توصیف بدست آمده از محیط، نقشه گفته می‌شود. حل مسئله مکان‌یابی نیاز به استفاده از نقشه محیط دارد، چرا که اگر ربات دانش کافی در رابطه با نقشه محیط داشته باشد با توجه به مشاهدهای که در هر لحظه از محیط انجام می‌دهد و تطبیق آن مشاهده با نقشه موجود می‌تواند مکان خود را تخمین بزند.

در عمل عموماً مدل دقیقی از محیط در دست نیست، یا اغلب قرار است ربات در محیط‌های مختلف و ناشناخته کار کند. بنابراین نیاز داریم که نقشه‌ای از محیط توسط خود ربات و مشاهداتی که از محیط دارد، ایجاد شود. در همین راستا مشکل دوم بروز می‌کند، چرا که برای ایجاد نقشه نیاز به اطلاعات دقیق از مکان ربات داریم. به همین علت است که از این مسئله در مقالات علمی به عنوان مسئله «مرغ و تخم مرغ» هم یاد می‌شود. چرا که برای مکان‌یابی به نقشه نیاز داریم و برای ایجاد نقشه به مکان ربات، مکان‌یابی صرف با فرض وجود نقشه فرآیندی نسبتاً سراسرت و قابل حل است، از طرف دیگر ایجاد نقشه صرف هم با وجود اطلاعات دقیق از مکان ربات مسئله‌ای با راه حلی سراسرت است. اما حل این دو مسئله به صورت همزمان و بدون اطلاعات اولیه از محیط، کاری دشوار می‌نماید.

در این فصل به مفاهیم مکان‌یابی و تهیه نقشه همزمان و همچنین تهیه نقشه به صورت جداگانه و مکان‌یابی به صورت جداگانه که موضوع این پژوهش هست، می‌پردازیم.

### SLAM - ۱-۳

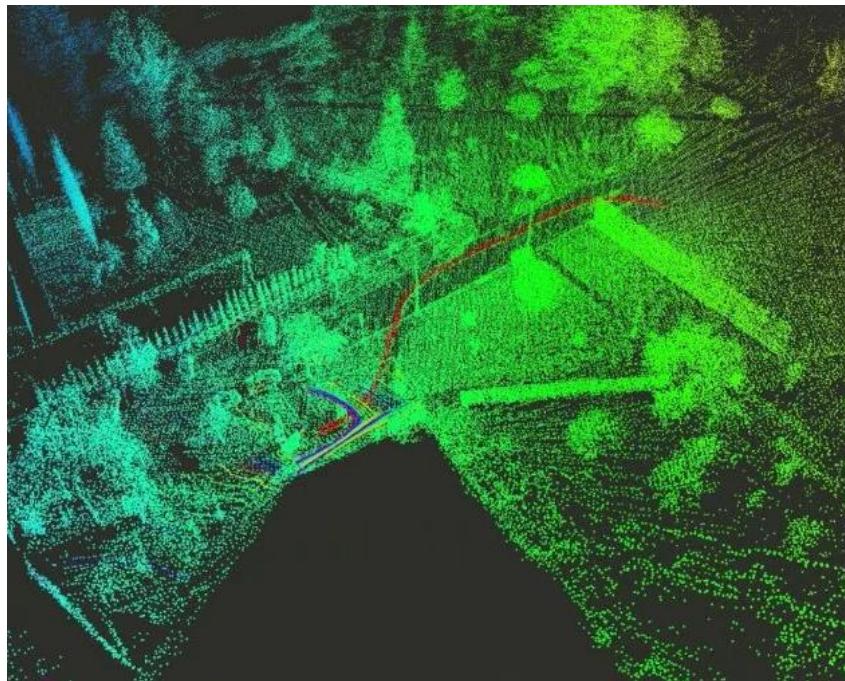
مکان‌یابی و تهیه نقشه همزمان یا همان SLAM یک روش نقشه‌برداری نوین است که با به کارگیری تکنولوژی به ربات‌ها و سایر وسایل نقلیه خودران اجازه می‌دهد تا نقشه‌ای بسازند و همزمان محل خود را نیز بر روی آن نقشه بیابند.

با استفاده از طیف گسترده‌ای از الگوریتم‌ها، محاسبات و داده‌های حسگرهای مختلف، سیستم نرم افزاری SLAM به یک ربات یا وسیله نقلیه دیگرمانند پهپاد یا خودروی خودران اجازه می‌دهد مسیری را در یک محیط ناآشنا ترسیم کند و به طور همزمان موقعیت خود را در آن محیط شناسایی کند. این رویکرد مکان‌یابی، امکان نقشه‌برداری از مناطقی را فراهم می‌کند که ممکن است برای اکتشاف انسان بسیار کوچک یا بسیار خطرناک باشند.

فناوری مکان‌یابی و نقشه‌برداری همزمان در همه چیز از جاروبرقی‌های خانگی تا خودروها استفاده می‌شود. با ارزان‌تر شدن این فناوری و تحقیقات بیشتر در مورد این موضوع، تعدادی از موارد استفاده عملی جدید برای SLAM در طیف گسترده‌ای از صنایع نیز ظاهر شده است.

### ۱-۱-۳ SLAM چیست؟

در توضیحات داده شده تا حد خوبی به این روش آشنا شدیم. نکته‌ی حائز اهمیت در این روش آن است که به یک ربات یا وسیله نقلیه خودران امکان نقشه برداری در یک منطقه‌ی جدید و نا‌آشنا را می‌دهد و در همان حال نیز مکان ربات را در آن منطقه‌ی جدید تعیین می‌کند [۱۹].



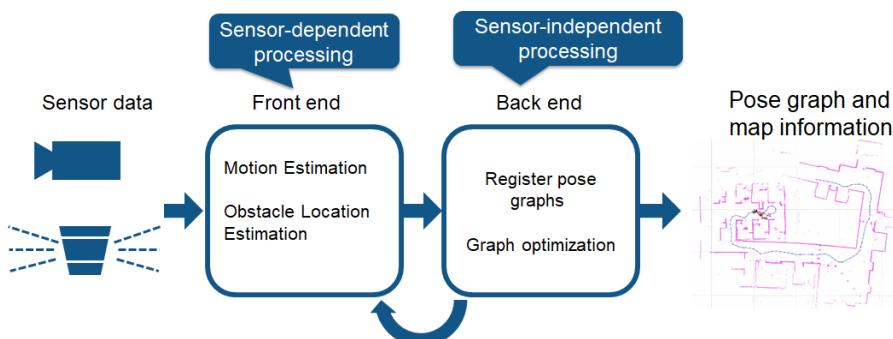
شکل ۲-۳ نمونه‌ای از فرآیند SLAM انجام شده [۲۰].

همانطور که پیش‌تر اشاره شد، این مسئله تا مدت زمانی به مسئله‌ی مرغ و تخم مرغ تبدیل شده بود. با این حال، پس از دهه‌ها تحقیق ریاضی و محاسباتی، تعدادی از راه حل‌های تقریبی مختلف به حل این مسئله پیچیده‌ی الگوریتمی نزدیک شده‌اند.

باید دقت شود که SLAM، یک محصول علمی یا سیستم یکتا نیست؛ به عبارتی، یک مفهوم گسترده با انواع بسیار متنوعی است. تعدادی راه حل نرم‌افزاری و الگوریتمی برای سامانه‌های مبتنی بر SLAM، پیاده سازی شده است که همه‌ی آن‌ها وابسته به محیط، نوع کاربرد و تکنولوژی‌های به کار رفته می‌باشند.

می‌توان گفت که اکثر سامانه‌های مبتنی بر SLAM حداقل دو جزء اصلی دارند:

- **اندازه‌گیری بُرد<sup>۱</sup>**: در تمامی روش‌های SLAM، یک ابزار و یا حسگری برای ربات وجود دارد که به آن اجازه می‌دهد تا محیط پیرامونش را مشاهده و اندازه‌گیری نماید. این می‌تواند به کمک دوربین‌ها، حسگرهای بصری دیگر، حسگر لایدار و یا سونار انجام شود. همچنین هر وسیله‌ای که توانایی اندازه‌گیری برخی مشخصات فیزیکی مانند مکان، فاصله و یا سرعت را داشته باشد، می‌تواند به عنوان جزئی از سیستم SLAM شرکت داشته باشد. این بخش وابسته به حسگر و پردازش‌های مربوط به آن می‌باشد و به آن فرانت-اند<sup>۲</sup> گویند.
- **استخراج داده**: هنگامی که این اندازه‌گیری‌ها و محاسبات انجام شد، سامانه‌ی مبتنی بر SLAM باید یک نوع نرم افزاری برای کمک در تفسیر داده‌ها داشته باشد. برای این بخش نیز راه حل‌های متفاوتی وجود دارد؛ از یک سری از الگوریتم‌های درهم آمیزی گرفته تا الگوریتم‌های پیچیده‌ی تطبیق اسکن، هریک می‌توانند به کار گرفته شوند. تمامی این راه حل‌هایی که به عنوان بک-اند<sup>۳</sup> به کار گرفته می‌شوند، یک هدف را دنبال می‌کنند. آن‌ها داده‌ای اندازه‌گیری برد مربوط به حسگرها را استخراج می‌کنند و از آن‌ها برای شناسایی نشانه‌های ساختگی یا طبیعی<sup>۴</sup> در یک محیط ناشناخته استفاده می‌نمایند. همچنین در این بخش به بهینه‌سازی موقعیت-گراف<sup>۵</sup> می‌پردازند. در شکل ۳-۳ می‌توانید این دو بخش را مشاهده نمایید.



شکل ۳-۳ نمونه‌ای از فرآیند SLAM انجام شده [۲۱].

<sup>1</sup> Range

<sup>2</sup> Front-end

<sup>3</sup> Back-end

<sup>4</sup> Landmarks

<sup>5</sup> Pose-graph optimization

یک سامانه‌ی مبتنی بر SLAM مناسب که به خوبی کار می‌کند، یک ساز و کار ثابت در طول زمان و همگام را بین حسگرهای اندازه‌گیری برد، برنامه‌ی استخراج و تحلیل داده، خود ربات و همچنین سایر نرمافزارها و سخت‌افزارهای به کار گرفته در این سامانه، دارد. دو مورد گفته شده، وابسته به نوع کاربرد می‌توانند تفاوت‌هایی در پیاده‌سازی داشته باشند ولی نکته‌ی مهم آن است؛ برای آنکه یک سامانه‌ی مبتنی بر SLAM به دقت محیط را بررسی کند، لازم دارد تا این اجزای مختلف به صورت یکپارچه با یکدیگر کار کنند.

### ۲-۱-۳ SLAM چگونه کار می‌کند؟

یک وسیله‌ی نقلیه یا ربات که مجهرز به فناوری SLAM می‌باشد، مکان خود را در یک محیط ناشناخته با شناسایی علائم و نشانه‌هایی که در آن محیط است، می‌یابد.

در حقیقت بسیار شبیه به آنچیزی است که انسان انجام می‌دهد. برای مثال زمانی که انسان در یک محیط گم می‌شود، ابتدا به محیط اطراف نگاه می‌کند تا یک نشانه‌ی بزرگ، ثابت و قابل شناسایی را برای خود پیدا کند. چنانچه قبل از نوشی محیط نگاه کرده باشد، عمل یافتن یک نشانه، کاری ساده‌تر خواهد بود. به هر حال ممکن است تا کنون در آن مکان نبوده باشد. بعد از شناسایی یک نشانه در محیط باید محاسبه کند تا فاصله‌اش تا آن چه قدر است. اگر بداند دقیقاً آن نشانه‌ی محیطی در چه مختصاتی قرار دارد و فاصله‌ی نسبی خود را هم تا آن بداند، بنابراین مکان خود را می‌تواند به دست آورد. اگر هم نشانه برای آن فرد نا آشنا باشد و مکانش را نداند، به مسیرش ادامه می‌دهد و بیشتر کاوش می‌کند. چنانچه از آن نشانه دور شود و برگردد به آن نگاه کند می‌تواند مقدار جایه‌جایی خود را محاسبه نماید. به این ترتیب کم کم نقشه‌ی آن محل در ذهن فرد شکل می‌گیرد و هرچه پیش می‌رود و نشانه‌های دیگری نیز می‌بیند، آنگاه می‌تواند یک نقشه‌ی قابل درک برای خود بسازد و مکان خود را در آن مکان مشخص نماید.

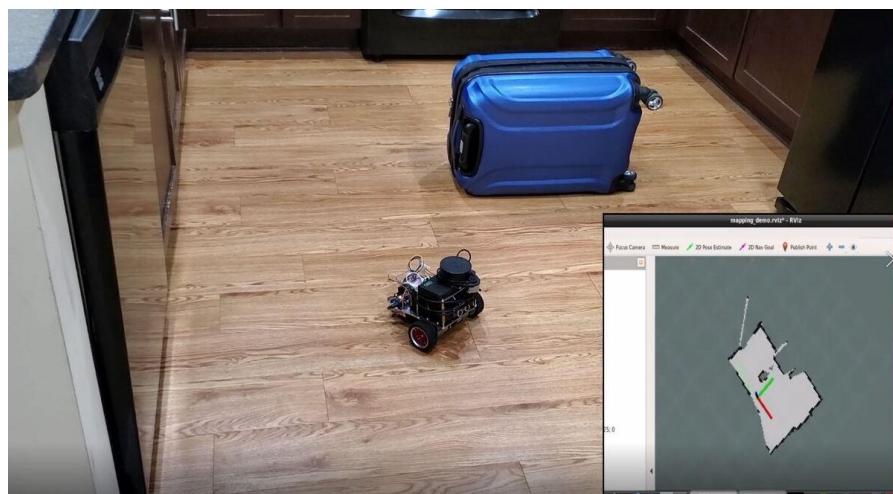
مکان‌یابی و نقشه برداری همزمان نیز تقریباً به همین صورت عمل می‌کند. ابتدا سعی می‌کند نشانه‌ها را شناسایی کند و مکان ربات را به کمک حسگرها نسبت به آن نشانه‌ها بیابد و سپس به مسیر ادامه می‌دهد تا محیط مذکور را به خوبی بررسی نماید و زمانی که تعداد مناسبی نشانه را پیدا کرد، سعی

می‌کند یک نقشه‌ی قابل درک از محیط بسازد. به این طریق، سامانه می‌تواند به صورت همزمان هم یک نقشه از محیط ایجاد نماید و هم مکان خود را در آن بیابد.

### ۳-۱-۳ - نقش لایدار در SLAM

یک حسگر اندازه‌گیری برد که از نور برای مشخص کردن مکان اشیاء ناشناخته نسبت به خودش استفاده می‌کند، حسگر لایدار می‌باشد. حسگر لایدار یکی از بهترین و معروف‌ترین حسگرهای مورد استفاده در مکان‌یابی و نقشه‌برداری همزمان می‌باشد.

حسگر لایدار به این صورت عمل می‌کند که نور لیزر را به یک سطح پرتاب می‌کند و مدت زمان برگشت پرتو را اندازه می‌گیرد. به این صورت فاصله‌ی هر نقطه را تا خودش با دقت نسبتاً بالایی می‌یابد. این داده‌ها برای ایجاد مدل‌های سه بعدی و نقشه‌ها کاربرد دارند. از آنجایی که لایدار به میزان روشنایی محیط نیز وابسته نیست، می‌تواند حتی در موقعی که نشانه‌ها برای چشم انسان نیز قابل روئیت نیستند، آن‌ها را شناسایی کند و فرآیند SLAM را پیش ببرد. به کمک مدل سه بعدی که در هر اسکن از محیط به ما ارائه می‌دهد؛ می‌توان نقشه‌ی محیط را نیز ساخت. نمونه‌ای از ربات مجهر شده به لایدار برای انجام SLAM را در شکل ۴-۳ مشاهده می‌کنید.



شکل ۴-۳ ربات مجهر به لایدار در SLAM [۲۱].

البته موقعی وجود دارد که حسگر لایدار نتواند به خوبی عمل کند. چنانچه موانع زیادی در اطراف نباشد یا موانع در فاصله‌ی زیادی از حسگر قرار داشته باشند، انجام عملیات تطابق اسکن‌ها برای ربات سخت

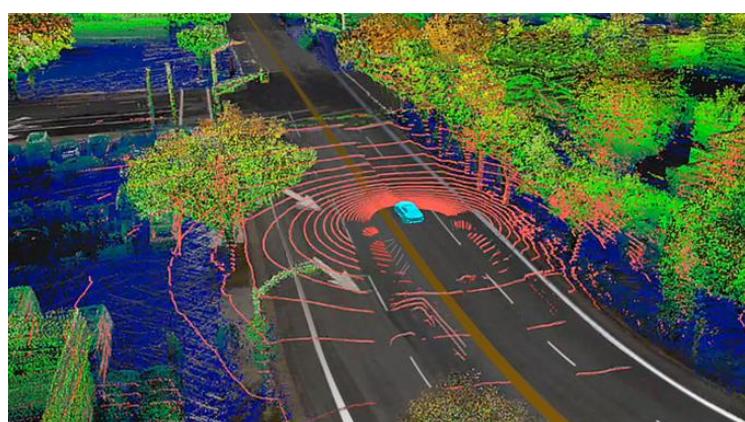
خواهد شد و به این ترتیب ممکن است ربات نتواند مسیری که طی می‌کند را دنبال نماید. از طرف دیگر تکنولوژی لایدار، توان پردازشی زیادی مصرف می‌کند. در بعضی موارد شاید دوربین‌های سنتی و یا سونار بسته به کاربرد بهتر عمل نمایند.

### ۳-۲- نقشه برداری

همانطور که یک گردشگر که به شهر جدیدی وارد می‌شود نیاز به نقشه‌ی آن شهر دارد، ربات‌های متحرک و وسایل نقلیه خودران نیز برای درک محیط اطراف خود نیاز به نقشه دارند. نقشه‌ها همیشه آماده نیستند که به راحتی در اختیار ربات‌ها قرار گیرند و با وجود مشکلاتی که در تهیه‌ی نقشه وجود دارد، ناگزیر به درست کردن آن می‌باشیم. همچنین اگر نقشه موجود باشد، باز هم به اندازه‌ی درک یک انسان متخصص از محیط اطراف به هنگام درست کردن نقشه، جزئیات دارد. بنابراین وسایل نقلیه برخلاف انسان‌ها به یک نقشه با جزئیات بیشتر برای درک محیط و تصمیم گیری نیاز دارند [۲۲].

همچنین از طرف دیگر نقشه‌ها به شرایط آب و هوایی بستگی ندارند و همچنین همیشه در دسترس خواهند بود و امکان از کار افتادن آن‌ها وجود ندارد. بنابراین می‌توان نقشه را به عنوان خروجی یک حسگر با قابلیت اطمینان بالاتری دید، که در درک محیط به ما کمک می‌کند.

نکته‌ی دیگر آن است که با گذشت زمان و پیشرفت و توسعه‌ی شهرها، نمی‌توان همیشه از یک نقشه‌ی ثابت برای محیط استفاده کرد. پس ناچار هستیم تا با روش‌های ساخت و به روزرسانی نقشه‌ها آشنا شویم. در شکل ۳-۵ نمونه‌ای از این نقشه‌های ساخته شده مشاهده می‌شود.

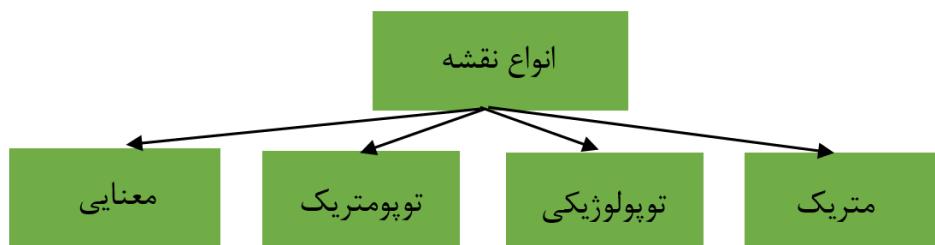


شکل ۳-۵ نمونه‌ای از نقشه تهیه شده برای خودروی خودران [۲۰].

### ۱-۲-۳- نقشه و انواع آن

تعریف متعددی از آنچه که باید یک نقشه از آن تشکیل شده باشد، وجود دارد. به صورت کلی، نقشه نمایشی از ویژگی‌ها و اشیاء موجود در یک دنیای واقعی و موقعیت مکانی آن اشیاء می‌باشد. در واقع یک توصیف نمادین یا انتزاعی از اجسام فیزیکی مانند خانه‌ها، رودها و ... و یا مفاهیم غیرفیزیکی مانند فشار هوا می‌باشد. در این پژوهش منظور از نقشه، اطلاعاتی از یک جهان واقعی و پدیده‌ها به همراه مکان آن‌ها در یک فضای دو یا سه بعدی می‌باشد.

همچنین عبارت "نقشه" یا "نقشه برداری" بعضی اوقات به جای یکدیگر به کار می‌روند؛ در صورتی که اسم آن به نتیجه و محصول نهایی اشاره دارد، در حالی که مصدر آن به فرآیند تهیی نقشه دلالت دارد. برای خودروهای خودران، نقشه می‌تواند یا از قبل تهیی شده باشد و یا به صورت بلاذرنگ و همزمان ایجاد شود. دقت شود که این نقشه‌های از قبل تهیی شده، غالباً جزئیات و اطلاعات بیشتری نسبت به نقشه‌های سنتی که برای انسان‌ها از قبل تهیی می‌شود، در بر دارد و از آن‌ها به عنوان نقشه‌های با کیفیت بالا<sup>۱</sup> یاد می‌شود. به صورت کلی نقشه‌ها انواع مختلفی دارند که چهار نوع مهم آن‌ها با نام‌های نقشه‌های متريک<sup>۲</sup>، نقشه‌های توپولوژيکی<sup>۳</sup>، نقشه‌های توپومتریک<sup>۴</sup> و نقشه‌های معنایی<sup>۵</sup> وجود دارد. در ادامه هریک را توضیح می‌دهیم.



شکل ۳-۶ انواع نقشه.

<sup>1</sup> High Definition (HD) maps

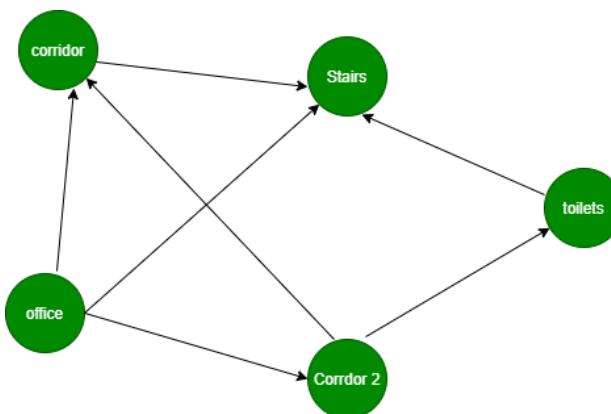
<sup>2</sup> Metric Maps

<sup>3</sup> Topological Maps

<sup>4</sup> Geometric Maps

<sup>5</sup> Semantic Maps

- نقشه‌های متريک: ربات از حسگر برای درک محیط استفاده می‌کند. اين نقشه‌ها در حقیقت نمایش دهنده‌ی اطلاعاتی از قبیل مکان، فاصله و زاویه از نقطه‌ی مرجع می‌باشند. در این نقشه‌ها اندازه‌گیری‌های دقیقی وجود دارد و به عنوان یک نمایش سه بعدی دقیق از محیط به شمار می‌روند.
- نقشه‌های توپولوژیکی: نگهداری نقشه‌های طولانی متريکی کاری دشوار می‌باشد و یک راه حل جایگزین برای آن‌ها، نقشه‌های توپولوژیکی که مبتنی بر گراف اند، می‌باشند. در این نقشه‌ها، نودها بیانگر مکان‌ها و یال‌ها بیانگر مسیرهای متصل کننده می‌باشند. به طور کلی، این نقشه‌ها را به صورت  $E = G = (V, E)$  نمایش می‌دهند که نودها با  $V = (V_1, V_2, \dots, V_N)$  و یال‌ها با  $E = (e_1, e_2, \dots, e_n)$  مشخص می‌شوند. ترتیب نودها اهمیت دارد چراکه گراف‌ها جهت‌دار می‌باشند و جهت مسیر به کمک ترتیب آن‌ها مشخص می‌شود. این نقشه‌ها به صورت دقیق مکان‌ها را مشخص نمی‌کنند و به جای آن رابطه‌ی بین نقاط مختلف محیط را نمایش می‌دهند. بیشتر برای کار مسیریابی مفید هستند. در شکل ۳-۷ نمونه‌ای از آن آمده است.



شکل ۳-۷ نقشه توپولوژیکی [۲۳].

- نقشه‌های توپومتریک: این نقشه‌ها ترکیبی از نقشه‌های متريک و توپولوژیکی هستند. از مختصات دقیقی برای اندازه‌گیری مکان اشیاء استفاده می‌کند ولی از طرفی محیط را با توجه به ارتباط نسبی بین نقاط نمایش می‌دهد.
- نقشه‌های معنایی: این نقشه‌ها دربردارنده‌ی اطلاعات اضافی هستند که مشخص کننده‌ی نوع اشیاء و سطوح موجود در محیط، می‌باشند. همچنین می‌توانند اطلاعاتی شامل هندسه‌ی اشیاء، سایز آن‌ها، جنس و ... داشته باشند.

### ۲-۲-۳ - مراحل تهیه‌ی نقشه

فرآیند تهیه‌ی نقشه را نقشه‌برداری گویند. به صورت کای فرآیند نقشه‌برداری به صورت زیر می‌باشد:

#### الف) مرحله‌ی اول: انتخاب حسگرهای لازم برای کار نقشه‌برداری

باتوجه به نوع کاربردی که می‌خواهیم و همچنین هزینه و دقت مورد نیاز می‌توان از سنسورهای متفاوتی استفاده کرد. به صورت کلی به دو دسته از حسگرهای درونی و بیرونی نیاز است. حسگرهای درونی برخی پارامترهای داخلی مانند سرعت و شتاب را اندازه می‌گیرند و حسگرهای بیرونی برخی پارامترهای خارجی را اندازه می‌گیرند. از جمله مهم‌ترین حسگرهای مورد استفاده، GNSS و IMU می‌باشند که GNSS کار اصلی تصحیح خطای انجام می‌دهد. همچنین حسگر لایدار از جمله حسگرهای معروف می‌باشد که انواع مختلفی دارد. برای مثال یک حسگر لایدار ۶۴ لایه با برد ۱۰۰ متر می‌تواند برای خیلی از کارهای حرفه‌ای در خودروهای خودران استفاده شود که طبیعتاً هزینه‌ی بیشتری هم دارد ولی در کارهای رباتیک می‌توان از حسگرهای لایدار تک لایه و برای نقشه‌برداری دو بعدی استفاده کرد.

#### ب) مرحله‌ی دوم: انتخاب نوع نقشه

براساس نوع کاربردی که از نقشه انتظار داریم، می‌توان یکی از انواع نقشه‌ها را که در بخش قبل توضیح دادیم انتخاب کرد. زمانی که ربات ما در محیط حرکت می‌کند و داده برداری می‌کند لازم است تا متناسب با نوع نقشه با آن داده‌خا برخورد کند.

#### ج) مرحله‌ی سوم: اهمیت و کاربرد نقشه‌برداری

دقت و میزان کیفیت یک نقشه‌ی درحال ساخت به کاربرد نهایی آن بستگی دارد. برای مثال، نقشه‌ی محیط یک ربات ممکن است برای مسیریابی تا هدف نیاز به یک نقشه‌ی جهانی سازگار نیز داشته باشد و یا آنکه صرفاً به نقشه‌های محلی نیاز داشته باشد تا فقط موانعی که به صورت ناگهانی در مسیر رسیدن به هدف قرار می‌گیرند را تشخیص دهد. این موضوع می‌تواند بر دقت نقشه و همچنین تصمیماتی که براساس اطلاعات داخل نقشه مانند احتمال رسیدن موفقیت آمیز به هدف و بدون برخورد به موانع، گرفته می‌شود؛ تاثیر بگذارد.

#### د) مرحله‌ی چهارم: به روزرسانی مداوم نقشه

لازم است همانطور که ربات درحال حرکت در محیط و جمع‌آوری اطلاعات توسط حسگرهاش هست، به صورت مداوم نقشه را به روزرسانی نماید.

### ۳-۳- مکان‌یابی مبتنی بر لایدار در پژوهش فعلی

در این پژوهش به دنبال مکان‌یابی صرف هستیم. در حقیقت در روش‌هایی که تا به حال گفته شد، چه در روش مکان‌یابی و نقشه برداری همزمان و چه نقشه برداری صرف، ما عمل نقشه‌برداری نیز برایمان حائز اهمیت بود.

در این پژوهش تمرکز اصلی بر روی مکان‌یابی صرف می‌باشد و چنانچه بتوانیم این عمل را با موفقیت انجام دهیم، می‌توانیم به راحتی در گام بعد نقشه‌ی محیط را نیز ایجاد کنیم که این مهم در بخش عملی پژوهش اتفاق خواهد افتاد.

در این پژوهش تمرکز اصلی بر خودروهای خودران می‌باشد که با کارهای رباتیکی که در سرعت‌های پایین انجام می‌شوند و در بسیاری از موارد برای کاربردهای داخل ساختمان یا یک محیط کوچک طراحی شده‌اند، تفاوت دارد. براین اساس از حسگرهایی که برای این کاربرد در واقعیت نیز پاسخ‌گو باشند استفاده خواهیم کرد.

در گام اول می‌دانیم که به کمک سامانه‌ی تعیین موقعیت جهانی به راحتی می‌توان با دقت نسبتاً خوبی مکان خودرو را تعیین نمود. همانطور که پیشتر اشاره شد این سامانه محدودیت‌هایی دارد که به تنها‌ی پاسخ‌گوی نیازهای یک خودروی خودران نمی‌باشد. از جمله این محدودیت‌ها می‌توان به در دسترس نبودن در بعضی از مواقع مانند داخل تونل‌ها و پارکینگ‌ها و یا نویز پذیری در برخی موارد و همچنین نرخ به روزرسانی نسبتاً پایین اشاره کرد.

در گام دوم برای بهبود آن از حسگرهای دیگر مانند IMU استفاده می‌کنیم. در حقیقت به کمک روش کیلومترشماری و با داشتن موقعیت اولیه توسط حسگر GPS می‌توان با نرخ به روزرسانی بالاتری مکان خودرو را به روزرسانی کرد. اما همانطور که اشاره کردیم این حسگر هم نمی‌تواند کاستی‌های GPS را

بپوشاند؛ چراکه بعد از زمان اندکی روش کیلومتر شماری با IMU به علت خطای جمع شونده‌ای که دارد، نمی‌تواند موقعیت درستی از خودرو را در اختیار قرار دهد.

در گام بعد به سراغ حسگری رفتیم که حتی به تنها یی تا مدت زمانی می‌تواند با خطای جمع شونده‌ی نستا کمی و با داشتن مکان اولیه‌ی خودرو، مکان آن را در هر لحظه به ما بدهد. این حسگر لایدار نام دارد. این حسگر را در بخش‌های قبل معرفی نمودیم. به کمک به کارگیری الگوریتم‌های ثبت ابرنقاط<sup>۱</sup> مانند روش نزدیک‌ترین نقطه‌ی تکراری (ICP)<sup>۲</sup> یا روش تبدیل توزیع‌های نرمال (NDT)<sup>۳</sup> می‌توان جایه‌جایی نسبی هر دو اسکن متوالی را یافت و سپس با روش کیلومترشماری با حسگر لایدار می‌توان مسیری را که خودرو یا جسم متحرک می‌رود دنبال کرد. در ادامه ابتدا با مفهوم ابر نقطه و سپس دو تا از مهمترین الگوریتم‌های ثبت ابرنقاط که نام برده‌یم، آشنا می‌شویم.

### ۳-۱-۳-۴- ابرنقاط<sup>۴</sup> چیست؟

ابرنقاط مجموعه‌ای از نقاط در یک فضای سه بعدی است که اطلاعاتی از محیط یا اشیاء سه بعدی را نمایان می‌کند. این نقاط با استفاده از حسگرهای مختلفی نظیر لایدارها، دوربین‌های سه بعدی و یا رادارها جمع آوری می‌شوند. اطلاعاتی که از این نقاط به دست می‌آید شامل مختصات مکانی هر نقطه به صورت (x, y, z) و همچنین شدت نور برگردانده شده و همچنین رنگ نقاط در صورت استفاده از دوربین سه بعدی می‌باشد.

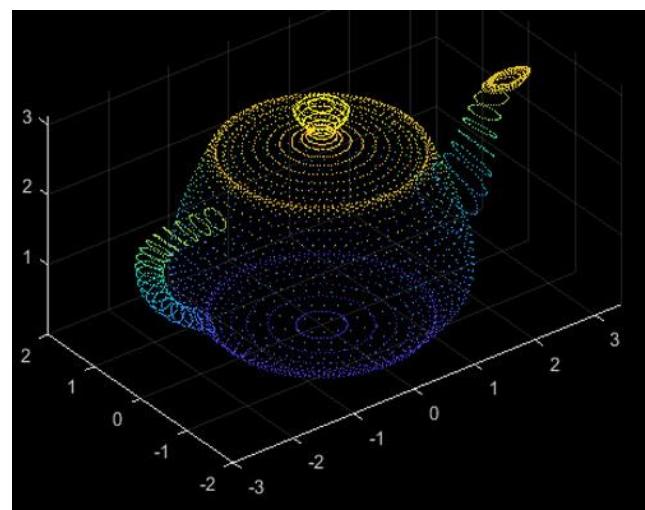
ابرنقاط برای اهدافی نظیر اندازه‌گیری و بازرسی کیفیت، مهندسی معکوس، اینیمیشن، رندر، نقشه‌سازی و در این پژوهش برای مکان‌یابی استفاده می‌شوند. برای مثال در شکل ۳-۸ نمونه‌ای از قوری ساخته شده به کمک ابرنقاط را مشاهده می‌کنید.

<sup>1</sup> Point cloud Registration

<sup>2</sup> Iterative Closest Point (ICP)

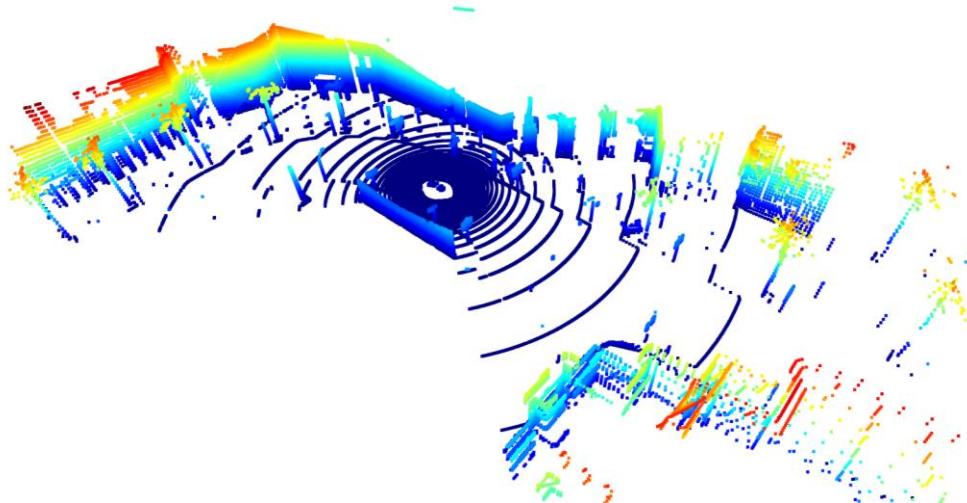
<sup>3</sup> Normal Distributions Transform (NDT)

<sup>4</sup> Point cloud



شکل ۸-۳ مدل سه بعدی قوری با ابرنقطا [۲۴].

می‌دانیم حسگر لایدار مورد استفاده در این پژوهش با فرکانس ۱۰ هرتز داده برداری می‌کند و ۱۰ اسکن از محیط را در هر ثانیه به عنوان خروجی برمی‌گرداند. هریک ازین اسکن‌ها در حقیقت یک ابرنقطا می‌باشد. این ابرنقطا کاملاً محیط پیرامون خودرو را برای ما نمایش می‌دهد. در حقیقت هر جسم صلی که در مسیر اشعه‌های لیزر این حسگر قرار بگیرد، نور لیزر را بازتاب می‌کند و شکل سه بعدی و مکان آن نسبت به خودرو مشخص می‌شود. در شکل ۹-۳ تصویری از یک اسکن لایدار ثبت شده در شبیه ساز کارلا نمایش داده شده است.

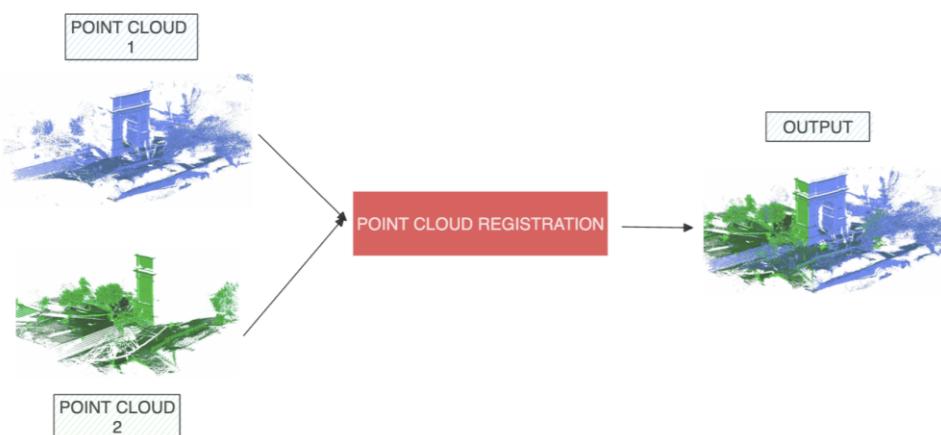


شکل ۹-۳ اسکن لایدار ثبت شده در کارلا.

### ۳-۲-۳- الگوریتم ثبت نقاط

الگوریتم ثبت نقاط یک روش محاسباتی است که برای تطبیق و ترکیب دو یا چند مجموعه ابرنقطاً به منظور تشکیل یک مجموعه ابر نقاط جدید به کار می‌رود. این الگوریتم‌ها عمولاً با استفاده از تطابق نقاط مشابه در دو مجموعه نقاط مختلف، ترانسفورمیشن‌های مکانی (مانند تغییرات مکان و زاویه) به عنوان خروجی ایجاد می‌کنند. این فرآیند برای انجام وظایفی مانند تطبیق مدل‌ها به داده‌های واقعی، پیگیری تغییرات در زمان، و ایجاد نقشه‌های دقیق محیط مورد استفاده قرار می‌گیرند [۲۵].

اصول کلی الگوریتم ثبت نقاط است که، ابتدا باید نقاط مشابهی را در دو مجموعه نقاط ورودی انتخاب کنیم. این نقاط عمولاً نقاطی هستند که ویژگی‌های مشابهی دارند، مانند نزدیکی مکانی یا تطابق شکل هندسی. سپس با استفاده از نقاط مشابه انتخاب شده، یک ماتریس تبدیل (شامل تغییرات مکان و زاویه) محاسبه می‌شود که به ما کمک می‌کند دو مجموعه نقاط را به یکدیگر تطبیق دهیم. سپس ماتریس تبدیل به صورت تکراری و با بهینه‌سازی متغیرهای مختلف، تطابق بین دو مجموعه نقاط را به حداقل می‌رساند و دائماً این ماتریس به روزرسانی می‌شود. پس از بهینه‌سازی، مجموعه ابر نقاط جدیدی به دست می‌آید که نقاط از دو مجموعه ورودی را در خود دارد. مجموعه نقاط ابر جدید برای انجام وظایف مختلفی مانند تحلیل محیط، مسیریابی خودروهای خودران، ردیابی تغییرات زمینه، و بسیاری دیگر از کاربردها استفاده می‌شود. برای درک بهتر این این روش می‌توانید به شکل ۱۰-۳ مراجعه کنید.



شکل ۱۰-۳ نمونه‌ای از خروجی الگوریتم ثبت نقاط [۲۶].

برای ثبت ابر نقاط الگوریتم‌های مختلفی توسعه یافته است که از جمله معروف‌ترین آن‌ها می‌توان به NDT و ICP اشاره کرد که برای هریک از آن‌ها نیز نسخه‌های مختلفی توسعه یافته که بسته به نوع کاربرد می‌توانند استفاده شوند. در ادامه به صورت مختصر به هریک اشاره می‌کنیم.

**الگوریتم ICP:** این الگوریتم در سال ۱۹۹۲ توسط بسل و همکاران<sup>[۲۷]</sup>، توسعه یافت. این الگوریتم یک ابرنقطه‌ی مبدأ و یک ابرنقطه‌ی هدف و همچنین یک ماتریس تبدیل اولیه را به عنوان ورودی می‌گیرد. سپس نقاط کلیدی را براساس مشخصه‌ی هندسی خاصی انتخاب می‌کند و تلاش می‌کند تا این نقاط را در یک ابرنقطه بر نقاط متناظر در ابر نقاط دیگر، تطبیق دهد. با توجه به این عمل تطابق، پارامترهای حرکتی و یا درواقع ماتریس تبدیل بین دو ابر نقاط محاسبه می‌شود. حال می‌توان میانگین خطای مربعات را نیز حساب کرد. سپس ماتریس تبدیل بر روی ابرنقطه هدف اعمال می‌شود و سپس همان فیچرهای هندسی برای تطابق جدید از ابرنقطه اول انتخاب می‌شوند. این عمل آنقدر تکرار می‌شود تا میانگین خطای مربعات از یک حد آستانه‌ی تعریف شده کمتر شود و ماتریس تبدیل نهایی به دست آید.

همچنین لازم به ذکر است که در روش قبل که اشاره شد سعی می‌کردیم فاصله‌ی بین هر دو نقطه‌ی متناظر از دو ابر نقطه را کمینه کنیم که به آن روش نقطه به نقطه<sup>۱</sup> می‌گویند. اما نسخه‌های دیگری نیز برای ICP توسعه یافته است و از جمله معروف‌ترین آن‌ها روش نقطه به صفحه<sup>۲</sup> می‌باشد. درواقع در این روش از مجموعه نقاط ابرنقطه هدف، صفحاتی می‌گذراند و سعی می‌شود تا فاصله‌ی نقاط ابرنقطه اول تا این صفحات کمینه شود. این روش، هم دقیق و هم سرعت همگرایی بیشتری دارد. در این پژوهش نیز ما از همین روش برای تطابق اسکن‌ها استفاده کردیم.

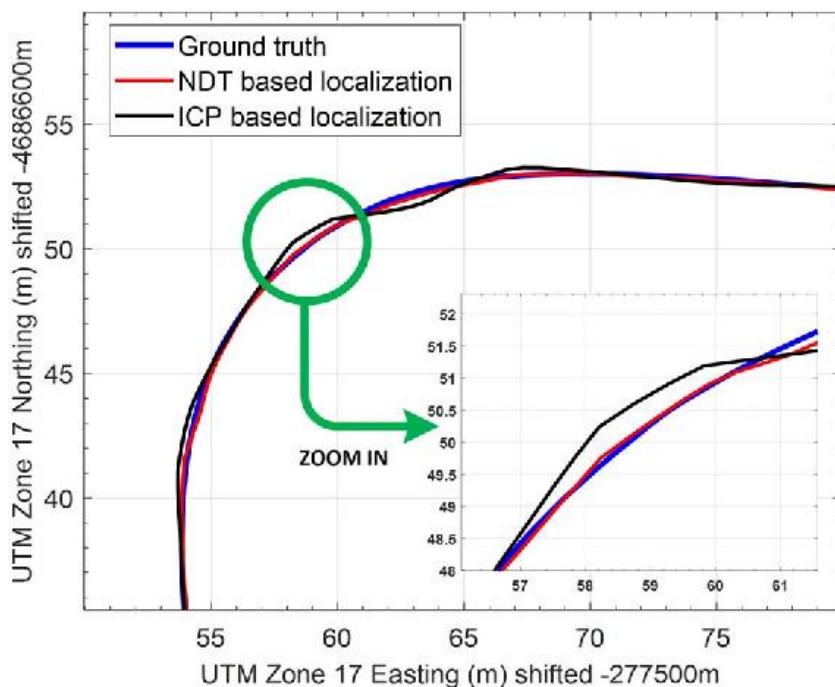
**الگوریتم NDT:** این الگوریتم در سال ۲۰۰۳ توسط بیبر و همکاران توسعه یافت<sup>[۲۸]</sup>. یکی از نسخه‌های اولیه‌ی این الگوریتم یعنی NDT دو بعدی، به جای محاسبه‌ی فواصل تبدیل توزیع نرمال را به دست می‌آورد و می‌توانست با دقیق بالا و به صورت بلادرنگ کار تطابق اسکن را انجام دهد. همچنین،

<sup>1</sup> Point-to-point

<sup>2</sup> Point-to-plain

NDT سه بعدی نیز برای تطابق اسکن سه بعدی، معرفی شد که نسبت به روش ICP سنتی قابل اعتمادتر و سریع‌تر بود.

طبق مقایسه‌ای که بین این دو الگوریتم انجام گرفته است [۲۹]، می‌توان عملکرد بهتر NDT را نسبت به ICP در شکل ۱۱-۳ مشاهده کرد. با این حال چون تا زمان انجام این پروژه کتابخانه‌ی آماده‌ای برای الگوریتم NDT در زبان پایتون موجود نبود و برای سادگی کار از روش ICP نقطه به صفحه در کتابخانه‌ی Open3d استفاده کردیم.



شکل ۱۱-۳ مقایسه‌ی دقت روش NDT و ICP [۲۹]

## فصل چهارم

### ترکیب حسگرها

## ترکیب حسگرها<sup>۱</sup>

همانطور که در فصل‌های قبل اشاره شد، در یک خودروی خودران تعداد زیادی حسگر به کار می‌رود که هریک داده‌های مخصوصی را جمع آوری می‌کند و البته محدودیت‌هایی هم دارد. هرچند هریک از حسگرها داده‌های کاربردی را تهیه می‌کنند، اما این موضوع را تصور کنید که بتوانید در یک لحظه اطلاعات مفید موردنظرتان را از ترکیب خروجی‌های چندین حسگر به دست آورید. ترکیب حسگرها فرآیندی است که ما به کمک آن می‌توانیم به این مهم دست یابیم. این موضوع در مغز انسان نیز اجام می‌شود؛ به این صورت که اطلاعات رسیده از حسگرهای مختلف بدن مانند لامسه، بینایی و شنوایی و ... با یکدیگر ترکیب شده و منجر به درک بهتر و دقیق‌تری از محیط پیرامون می‌شود.

اساساً ترکیب حسگرها با هدف غلبه بر محدودیت‌های سنسورهای منفرد، از طریق جمع‌آوری و ترکیب داده‌ها از چندین حسگر برای تولید اطلاعات قابل اعتمادتر و با عدم قطعیت کمتر انجام می‌شود. فناوری پیاده‌سازی این رویکرد به ظاهر ساده است؛ یک پردازنده با استفاده از الگوریتم‌های نرم‌افزاری، داده‌های حسگرها مختلف را با یکدیگر ترکیب کرده و از این طریق تصویر کامل‌تری از فرآیند یا موقعیت مورد بررسی ارائه می‌دهد. ایده‌ی شکل‌گیری این فناوری آن است که با افزایش درک نسبت به فرآیند یا موقعیت مورد بررسی، بینش نسبت به آن حوزه نیز بیشتر و عمیق‌تر خواهد شد و در نتیجه، پاسخ‌های داده شده به سوالات این حوزه نیز هوشمندتر و دقیق‌تر خواهد بود.

### ۱-۴- انواع مختلف ترکیب حسگرها

ارتباطات بین حسگرها و معماری همچوشی آن‌ها را بر حسب نوع ارتباط بین حسگرها در سیستم می‌توان به انواع زیر تقسیم نمود:

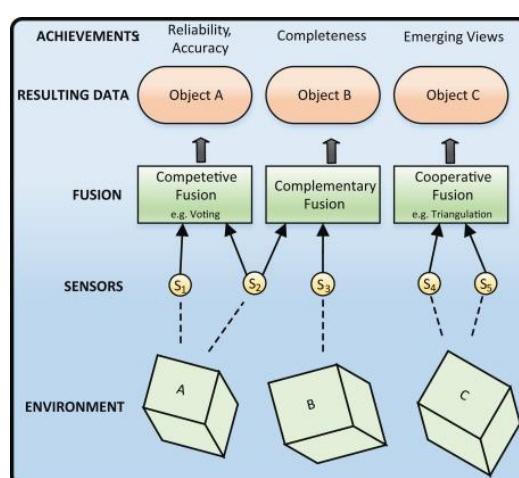
**الف) ترکیب حسگر مکمل:** اطلاعات ارائه شده توسط این حسگرها، توصیف‌کننده محیط اطراف خود از جنبه‌های مختلف هستند. ترکیب این اطلاعات می‌تواند به منظور ایجاد دیدی وسیع‌تر نسبت به محیط مورد استفاده قرار گیرد. در این نوع از همچوشی، حسگرها به صورت مستقل از هم عمل کرده و به طور

<sup>۱</sup> Sensor fusion

مستقیم به یکدیگر وابسته نیستند؛ اما می‌توانند برای رسیدن به تصویری کامل‌تر از پدیده تحت بررسی، با یکدیگر ترکیب شوند. به عنوان مثال می‌توان به سیستم‌های بینایی ماشین اشاره کرد؛ در این سیستم‌ها، تصویر شی مورد بررسی، توسط چندین دوربین، یا ترکیبی از یک دوربین و یک حسگر لایدار گرفته شده و با ترکیب تصاویر گرفته شده از این شی واحد، می‌توان به درک بهتری از محیط دست پیدا کرد.

ب) **ترکیب حسگر رقابتی**: در بعضی کاربردها از چندین حسگر به منظور گرفتن اطلاعات از یک هدف مشترک استفاده شده و خروجی این حسگرها برای ارائه یک خروجی قابل اطمینان با هم ترکیب می‌شوند. در این مدل، دو حالت می‌توانیم داشته باشیم، یک حالت ترکیب حسگرهای مستقل از هم و دیگری هم ترکیب داده‌های یک حسگر واحد، که در زمان‌های مختلف به دست آمده است. این روش بیشترین مقدار کامل بودن را در میان این سه روش دارد و در برابر خطأ مقاوم می‌باشد.

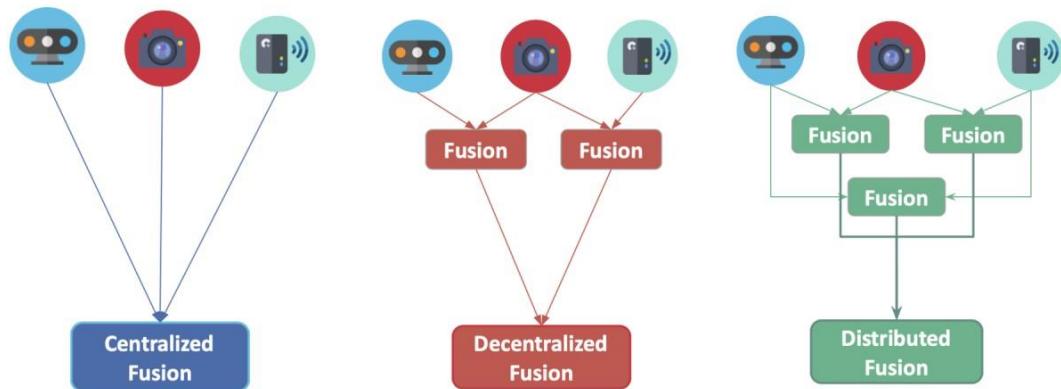
ج) **ترکیب حسگر مشارکتی**: این نوع از همجوشی، داده‌های حسگرهای مختلف مانند حسگرهای صوتی و تصویری را با هم ترکیب کرده و خروجی تولید شده توسط آن نسبت به ورودی تک تک حسگرها به صورت منفرد، پیچیدگی بیشتری خواهد داشت و درنهایت اطلاعات کامل‌تری داریم که با استفاده از یک حسگر تنها امکان پذیر نبود. به طور مثال یکپارچه‌سازی یا ترکیب دو دوربین با زوایای دید مختلف، می‌تواند برای تولید تصویری سه بعدی از محیط مورد بررسی، مورد استفاده قرار گیرد. همچنان مشارکتی حسگرها نسبتاً پیچیده بوده و نتایج بدست آمده از آنها، به دقت تک تک حسگرها بسیار حساس است. هر چند همچنان رقابتی حسگرها می‌توانند منجر به افزایش دقت و سطح اطمینان نتایج شود، اما ترکیب مشارکتی آن‌ها پتانسیل آن را دارد که مقادیر این دو پارامتر ذکر شده را کاهش دهد.



شکل ۱-۴ روش‌های مختلف ترکیب حسگرها [۳۰].

همچنین سه طرح ارتباطی در ترکیب حسگرها استفاده می شود که بسته به نوع کاربرد می توانند استفاده شوند. آن ها عبارتند از:

- **غیر مرکزی<sup>۱</sup>:** ارتباطی بین نود حسگرها رخ نمی دهد و داده ها در مکان اولیه شان ترکیب و پردازش شده و به جای دیگری ارسال نمی شوند.
  - **مرکزی<sup>۲</sup>:** حسگرها با یک نود مرکزی ارتباط برقرار می کنند و داده ها برای ترکیب و پردازش به آن ارسال می شوند.
  - **توزیع شده<sup>۳</sup>:** نودهای حسگر در فواصل زمانی تعیین شده ای با یکدیگر ارتباط برقرار می کنند یا آنکه هر دو سنسوری با یکدیگر ترکیب شده و نتایج آن ها نیز مجددا باهم ترکیب می شود.
- در شکل ۲-۴ این سه طرح ارتباطی به خوبی نمایش داده شده است.



شکل ۲-۴ طرح های ارتباطی حسگرها [۳۰].

## ۲-۴ - سطوح ترکیب حسگرها

تا اینجا کار متوجه شدیم که چه روش هایی برای ترکیب حسگرها وجود دارد. اما سوال بعدی آن است که آیا فقط قادر به ترکیب داده های خام هستیم یا راه حل های دیگری نیز وجود دارد.

<sup>1</sup> Decentralized

<sup>2</sup> Centralized

<sup>3</sup> Distributed

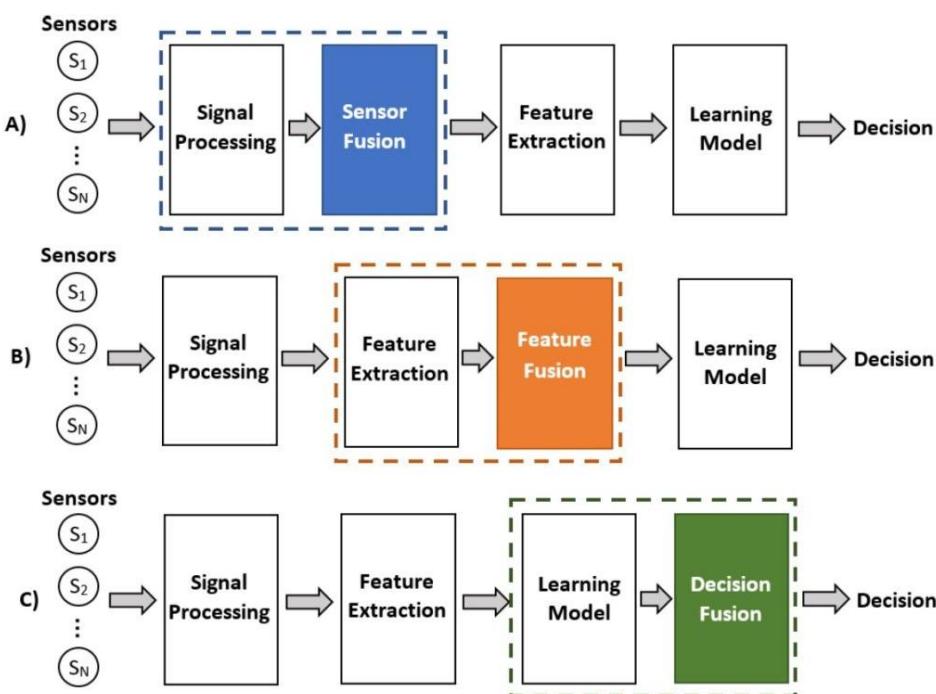
داده‌های خروجی از حسگرها نیازه به پردازش دارند. بعد از آن نیز مراحل دیگری وجود دارد اما آنکه مرحله‌ی ترکیب در کجا قرار بگیرد سطوح مختلفی را به وجود می‌آورد که متناسب با آن میزان حافظ و دقیق مدل تغییر خواهد کرد. به صورت کلی ۳ سطح را می‌توان برای مرحله‌ی ترکیب در نظر گرفت:

- **سطح پایین (سطح داده):** در این سطح، داده‌های حسگر به صورت خام مورد پردازش قرار می‌گیرند تا مطمئن شویم که به صورت ناخواسته در مراحل بعدی تبدیل، نویز به داده‌ها اضافه نمی‌شود. در صورت استفاده از بیش از یک سنسور برای اندازه‌گیری کمیتی فیزیکی، داده‌ها در این سطح ترکیب می‌شوند. در مورد سنسورهایی که کمیت‌های متفاوتی را اندازه‌گیری می‌کنند، ترکیب داده‌ها در سطوح بالاتر انجام می‌شود.

- **سطح متوسط (سطح ویژگی):** در این سطح، به جای استفاده از داده‌های خام، ویژگی‌هایی که از چندین سنسور مستقل از هم استخراج شده که به عبارتی تفسیر داده‌های هر حسگر توسط خود آن حسگر یا یک پردازنده می‌باشد، با یک دیگر ترکیب می‌شوند. در این سطح، درباره‌ی هریک از بردار ویژگی‌های به دست آمده فرضی می‌شود و وزنی در نظر گرفته می‌شود و درواقع به صورت وزن‌دار یکدیگر ترکیب می‌شوند تا به یک جواب واحد برسند.

- **سطح بالا (سطح تصمیم):** تا حدودی مانند سطح متوسط می‌باشد ولی در این بخش به نحوی تصمیم‌ها با یک دیگر ترکیب می‌شوند. در این سطح، ویژگی‌های مختلف طبقه‌بندی شده و از این داده‌ها، برای تصمیم‌گیری در مورد محیط و تشخیص اقدامات موردنیاز استفاده می‌شود.

پارادایم‌های مختلف ترکیب حسگرها در سطوح مختلف اشاره شده، قابل به کار گیری هستند. علاوه بر سه سطح اشاره شده، سطوح هیبریدی نیز قابل مدل‌سازی هستند. به عنوان مثال، داده‌های دو حسگر مختلف را می‌توان برای ایجاد یک مجموعه ویژگی با هم ترکیب کرده و از مدل دسته‌بندی به دست آمده، در سطح سطح تصمیم استفاده نمود. از سوی دیگر، نتایج حاصل از استخراج ویژگی‌ها و مدل‌های دسته‌بندی به دست آمده از چند حسگر، می‌توانند به منظور آموزش و اصلاح الگوریتم‌های دسته‌بندی سطح تصمیم در سایر حسگرها استفاده شوند. در شکل ۳-۴ این سه سطح به خوبی نمایش داده شده است.



شکل ۳-۴ طرح‌های ارتباطی حسگرها.<sup>[۳۰]</sup>

### ۳-۴- الگوریتم‌های ترکیب حسگر

برای ترکیب داده‌ها در فرآیند ترکیب حسگر، نیاز به یک الگوریتم داریم. تا کنون الگوریتم ترکیب کننده را به عنوان بلوک دیاگرامی درنظر می‌گرفتیم که داده‌ها را به عنوان ورودی به آن می‌دادیم و از سوی دیگر خروجی ترکیب شده دریافت می‌کردیم. حال نیاز است که به صورت جزئی‌تری به الگوریتمی که این بین قرار می‌گیرد بپردازیم. الگوریتم‌های مختلفی برای این امر توسعه داده شده‌اند که معروف‌ترین آن‌ها فیلتر کالمن<sup>۱</sup> می‌باشد.

#### ۳-۴-۱- الگوریتم فیلتر کالمن

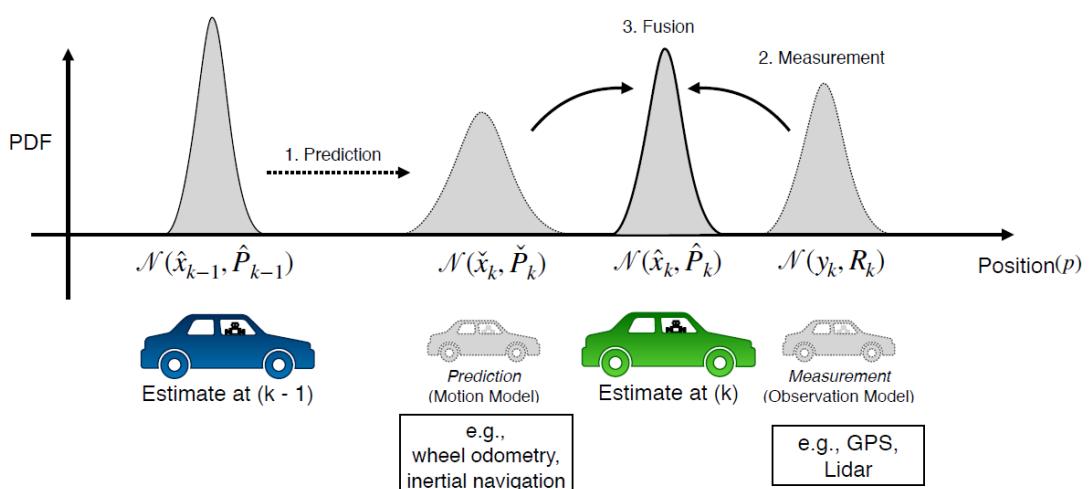
پیدایش این الگوریتم به سال ۱۹۶۰ بازمی‌گردد. از زمانی شهرت زیادی پیدا کرد که در راهنمای کامپیوتری آپولو برای ناوبری جهت رسیدن به ماه استفاده شد. اکنون در گوشی‌های هوشمند، ماهواره‌ها و اهداف مسیریابی استفاده می‌شود.

<sup>1</sup> Kalman-filter

هدف آن تخمین حالت فعلی سیستم دینامیکی یک سامانه می‌باشد، به خصوص زمانی اهمیت آن درک می‌شود که به دنبال تحلیل داده‌های نویزی هستیم. برای مثال، حسگرهای بر روی اتومبیل‌های خودران گاهی اوقات داده‌های نویزی و ناقصی را جمع آوری می‌کنند که می‌تواند توسط الگوریتم کالمن تصحیح شود.

می‌توان گفت که فیلتر کالمن یک نوعی از فیلترهای بیزین می‌باشد. در فیلتر بیزین، الگوریتم بین پیش‌بینی تخمین حالت فعلی سیستم و به روزرسانی مشاهدات حسگرها، جابه‌جا می‌شود. به طور خاص، الگوریتم یک پیش‌بینی را انجام می‌دهد و آن را براساس به روز رسانی که انجام می‌دهد، تصحیح می‌نماید؛ دائماً این دو مرحله را انجام می‌دهد تا به دقت دلخواه برسد.

فیلتر کالمن پیش‌بینی را به صورت بلادرنگ و با استفاده از مدل ریاضی مبتنی بر حالت سیستم که در بردارنده‌ی مکان و سرعت است و همچنین با توجه به عدم قطعیت‌ها، انجام می‌دهد. برای مثال در این پژوهش، چنانچه موقعیت خودرو را در زمان  $k-1$  داشته باشیم، آنگاه به دنبال آنیم که موقعیت آن را در زمان فعلی که  $k$  هست، داشته باشیم. برای این منظور به کمک مدل حرکتی و مقادیری از IMU که داریم و با فرض تغییرات اندک در زمان کم (خطی سازی) می‌توانیم موقعیت فعلی را پیش‌بینی نماییم. در زمان فعلی از حسگرهای دیگر مانند GPS و لایدار نیز داده‌هایی می‌رسد که می‌توانیم موقعیت فعلی را نیز از هرکدام از آن‌ها به دست آوریم. در مرحله‌ی بعد می‌توانیم به کمک این داده‌ها، پیش‌بینی انجام گرفته را تصحیح نماییم. این موضوع به خوبی در شکل ۴-۴ نمایش داده شده است.



شکل ۴-۴ نحوه کارکرد فیلتر کالمن در این پژوهش [۳۱].

دقت شود در مدل ساده‌ی این الگوریتم فرض می‌شود که توابع خطی هستند ولی بعضی از حسگرها خروجی‌های غیرخطی دارند. برای این منظور محققین از دو رویکرد برای خطی سازی مدل استفاده می‌کنند:

- **روش فیلتر کالمن توسعه یافته<sup>۱</sup>:** از مفاهیم ماتریس ژاکوبین و بسط تیلور برای خطی سازی استفاده می‌کند.
- **روش فیلتر کالمن بی اثر<sup>۲</sup>:** از تخمین‌های دقیق‌تری برای خطی سازی استفاده می‌نماید.

باتوجه به آنکه در این پژوهش مدل ما خطی نمی‌باشد، از فیلتر کالمن توسعه یافته استفاده می‌کنیم. برای این منظور لازم است تا با روابط آن آشنا شویم. همانطورکه پیشتر اشاره شد در روش کالمن دو فاز پیش‌بینی و تصحیح داریم. برای پیش‌بینی نیاز به مدل حرکتی خطی سازی‌شده نیاز داریم که در معادله (۴-۱) آمده است.

$$x_k = f_{k-1}(\hat{x}_{k-1}, u_{k-1}, 0) + F_{k-1}(x_{k-1} - \hat{x}_{k-1}) + L_{k-1}w_{k-1} \quad (4-1)$$

در فاز پیش‌بینی باتوجه به مدل حرکتی، حالت تصحیح شده‌ی سامانه در گام قبلی و ورودی گام قبل می‌توان حالت فعلی را پیش‌بینی کرد. همچنین باتوجه به انحراف معیار نویز حرکتی و انحراف معیار تصحیح شده‌ی حالت قبل، می‌توان انحراف معیار را برای حالت فعلی پیش‌بینی کرد.

$$\check{x}_k = f_{k-1}(\hat{x}_{k-1}, u_{k-1}, 0) \quad (4-2)$$

$$\check{P}_k = F_{k-1}\hat{P}_{k-1}F_{k-1}^T + L_{k-1}Q_{k-1}L_{k-1}^T \quad (4-3)$$

در گام بعد مدل اندازه‌گیری خطی شده را باید داشته باشیم که در معادله (۴-۵) آمده است.

$$y_k = h_k(\check{x}_k, 0) + H_k(x_k - \check{x}_k) + M_kv_k \quad (4-4)$$

در گام بعد لازم است تا بهره‌ی بهینه را محاسبه نماییم و سپس به کمک آن حالت فعلی و انحراف معیار تصحیح شده را می‌یابیم.

$$K_k = \check{P}_k\check{H}_k^T \left( H_k\check{P}_kH_k^T + M_kR_kM_k^T \right)^{-1} \quad (4-5)$$

<sup>1</sup> Extended Kalman-filter

<sup>2</sup> Unscented Kalman-filter

$$\hat{x}_k = \check{x}_k + K_k(y_k - h_k(\check{x}_k, 0)) \quad (6-4)$$

$$\hat{P}_k = (1 - K_k H_k) \check{P}_k \quad (7-4)$$

به این ترتیب قادر خواهیم بود داده‌های حسگرهای مختلف را باهم ترکیب کنیم و مکان خودرو را با دقت نسبتاً بالایی به دست آوریم.

### ۲-۳-۴- الگوریتم‌های دیگر

الگوریتم‌های دیگری نیز برای ترکیب حسگرها وجود دارد. با توجه به آنکه تمرکز اصلی در این پژوهش در استفاده از EKF می‌باشد بنابراین به صورت مفصلی به آن‌ها نپرداختیم و در ادامه به آن‌ها اشاره خواهد شد.

**شبکه‌های عصبی<sup>۱</sup>:** یک شبکه‌ی عصبی آموزش دیده در یادگیری عمیق، برای ترکیب داده‌های تصویری از چندین حسگر برای طبقه‌بندی نتایج استفاده می‌شود.

**نظریه‌ی حد مرکزی<sup>۲</sup>:** هدف این نظریه که بر پایه قانون اعداد بزرگ شکل گرفته، خوانش تعداد زیادی نمونه و سپس محاسبه میانگین آن مجموعه داده‌های است. نتیجه این تئوری در قالب یک توزیع نرمال و منحنی زنگوله‌ای نمایش داده می‌شود. پس این روش از داده‌های چندین حسگر میانگین می‌گیرد.

**شبکه‌های بیزی<sup>۳</sup>:** همانطور که پیشتر ذکر کردیم، فیلتر کالمون یکی از انواع فیلتر بیز می‌باشد ولی خب انواع دیگری هم دارد. برای مثال الگوریتم Dempster-Schafer با اندازه‌گیری عدم قطعیت و استنتاجها سعی به تقلید نحوه‌ی استدلال انسان می‌کند.

<sup>1</sup> Neural Networks

<sup>2</sup> Central Limit Theorem (CLT)

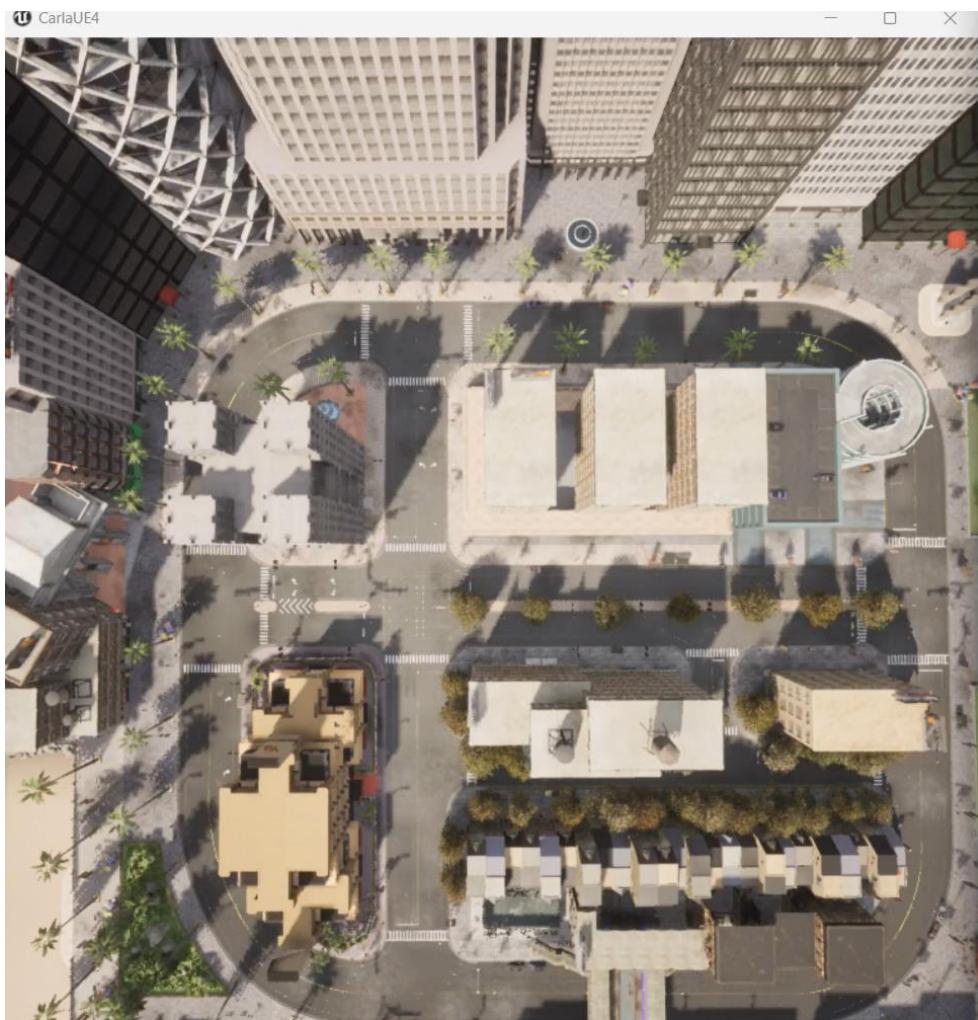
<sup>3</sup> Bayesian Network

## فصل پنجم

### نتایج

## نتایج

در فصل‌های گذشته با نحوه‌ی پیاده‌سازی و مفاهیم به کار گرفته شده در این پژوهش آشنا شدیم. در این پژوهش دو بخش مهم شبیه سازی و آزمایش عملی تعریف شده بود. بخش شبیه سازی به کمک نرم افزار کارلا و برنامه‌نویسی پایتون انجام می‌گرفت و به این صورت بود که باید به کمک حسگرهای GNSS و IMU و لایدار هریک به صورت جداگانه ابتدا مکان خودرو را به دست می‌آوردیم. سپس در گام اول چک می‌کنیم که تاثیر ترکیب دو حسگر GNSS و IMU به چه صورت است و چه ایراداتی دارد. سپس در مرحله‌ی نهایی ترکیب نهایی سه حسگر GNSS و IMU و لایدار بررسی می‌شود. نمای بالای شهری که در شبیه‌ساز کارلا استفاده شده است، در شکل ۱-۵ آمده است.



شکل ۱-۵ نمای بالا از شهر مورد آزمایش در کارلا.

همچنین خودرو در حالت رانندگی خودران تنظیم شده است و ما فقط عمل داده برداری را انجام می‌دهیم. نمای بالای خودروی در حال حرکت در شکل ۲-۵ قرار گرفته است.



شکل ۲-۵ نمایی از خودرو در حال جمع آوری داده در سطح شهر.

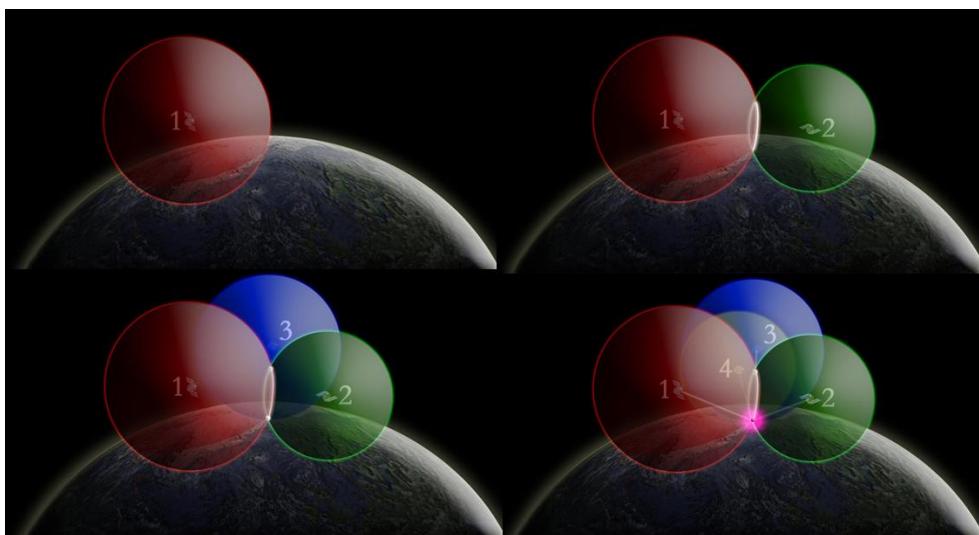
به عنوان بخش عملی نیز به دنبال بررسی این موضوع هستیم که حسگر لایدار به تنها یی چه دققی در مکان‌یابی دارد. همچنین چنانچه با دقت خوبی موفق به مکان‌یابی جسم متحرک شویم، در گام بعد می‌توانیم نقشه‌ی محیط پیرامون را نیز بیابیم.

## ۱-۵ - مکان‌یابی به کمک GNSS

این مرحله ساده‌ترین گام در مکان‌یابی به شمار می‌آید چراکه کار و خروجی اصلی این حسگر موقعیت جسم متحرک می‌باشد و نیاز به اجرای الگوریتمی جهت یافتن مکان نداریم. همانطور که پیشتر اشاره شد، این روش مکان‌یابی جزء روش‌های مکان‌یابی جهانی می‌باشد و در فریم جهانی مختصات جسم متحرک را به ما می‌دهد. ابتدا با نحوه‌ی عملکرد این حسگر آشنا می‌شویم و سپس به سراغ خروجی به دست آمده در شبیه‌ساز می‌رویم.

### ۱-۱-۵- نحوه مکان‌یابی با GNSS

سامانه‌ی GNSS برای تعیین مکان یک جسم از سامانه‌ی ماهواره‌ای استفاده می‌کند. لازم به ذکر است که به تعداد ۲۴ ماهواره در ارتفاع ۲۰۱۹۰ کیلومتری از سطح زمین، در طول هر دوازده ساعت یک دور به دور زمین می‌چرخند. شش مدار حول زمین داریم که در هریک به تعداد چهار ماهواره حضور دارند که با زاویه ۶۰ درجه نسبت به هم قرار دارند. برای تعیین مکان یک جسم مانیاز داریم که حداقل چهار ماهواره در دید گیرنده GPS قرار گیرد. پس درواقع این صفحات اوربیتالی به نحوی قرار گرفتند که در هر زمانی در هر نقطه‌ای از زمین حداقل چهار ماهواره در دید باشد. مکان هر گیرنده GPS به کمک اندازه گیری زمان پرواز<sup>۱</sup> مشخص می‌شود به این صورت ماهواره‌ها مکان اوربیتال خود را به همراه زمانی که این ارسال انجام شده است، می‌فرستند و گیرنده هم مکان خود را می‌یابد. برای تعیین مکان در روی سطح زمین (مکان دو بعدی) فقط سه ماهواره نیاز است که درواقع محل برخورد سه کره می‌باشد و ماهواره چهارم برای اصلاح خطای زمانی موجود در گیرنده GPS است. اما اگر بخواهیم مکان یک جسم دارای ارتفاع از سطح زمین را تعیین کنیم؛ ناگزیریم برای پیدا کردن مکان، از چهار ماهواره استفاده نماییم که درواقع ماهواره چهارم در پیدا کردن ارتفاع و برطرف کردن خطای زمانی ذکر شده کمک می‌کند. در شکل ۳-۵ به خوبی این موضوع نمایش داده شده است.



شکل ۳-۵ نحوه کارکرد حسگر GPS در تعیین مکان [۳۲].

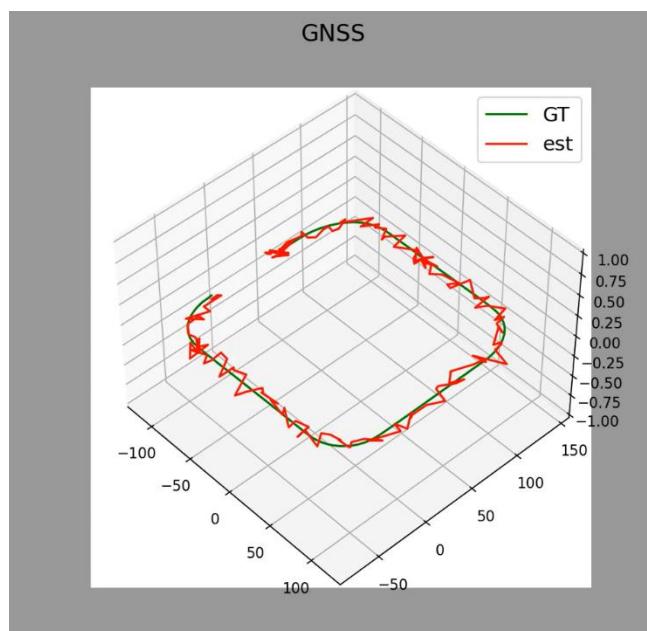
<sup>۱</sup> Time of Flight

## ۲-۱-۵- نتایج مکان‌یابی با GNSS در کارلا

ابتدا لازم است تا در سمت مشتری پس از اتصال به سرویس دهنده و ساخت یک جهان، یک حسگر GPS را به خودوری موردنظر متصل نماییم. نکته‌ی مهم آن است که GPS به ما طول و عرض و ارتفاع جغرافیایی را می‌دهد که ما آن را به کمک یک تابع به مختصات داخل شبیه ساز برحسب  $x$  و  $y$  و  $z$  و با واحد متر تبدیل می‌کنیم. پس لازم است تا در تابعی که حسگر پس از هر بار رسیدن داده، آن را صدای زند، تابع تعریف شده را نیز صدا بزنیم.

نکته‌ی دیگر آن است که GPS موجود در شبیه ساز به صورت پیش فرض با دقت خیلی بالایی مکان را تعیین می‌کند، اما در واقعیت چنین نیست؛ به همین دلیل می‌توانیم برای آن نویز را تعیین نماییم.

همچنین لازم به ذکر است یک پردازه‌ی جدا برای رسم نقشه و مسیری که خودرو طی می‌کند در نظر گرفتیم و داده‌های دریافتی برای آن ارسال می‌شوند و به صورت بلادرنگ این نقشه آپدیت می‌شود. در نهایت نقشه‌ی به دست آمده در نمودار ۱-۵ قابل مشاهده است. داده‌های سبزرنگ داده‌های صحیح از مکان خودرو هستند که مستقیماً از شبیه ساز دریافت کردیم و داده‌های قرمز رنگ در نمودار، بیانگر داده‌های دریافت شده از حسگر GPS می‌باشند. همانطور که انتظار داشتیم این دو نمودار تا حد خوبی بر یک دیگر منطبق می‌باشند و البته خطوط شکسته‌ی قرمز رنگ به علت نویز حسگر می‌باشد.



نمودار ۱-۵ تخمین مکان خودرو در سطح شهر با حسگر GPS.

## ۲-۵- مکان‌یابی به کمک IMU

در مقدمه در رابطه با ساختار داخلی این حسگر صحبت کردیم. این حسگر به طور کلی از دو قسمت اصلی یعنی شتاب‌سنج و ژیروسکوپ تشکیل شده است که هریک توانایی اندازه‌گیری کمیت‌های مربوطه را در سه راستای فضایی  $x$  و  $y$  و  $z$  دارند. همانطور که مشخص است به کمک این پارامترها به تنها یک نمی‌توان مکان یک خودرو را تعیین کرد. بنابراین نیاز است که الگوریتم و کدی نوشته شود که با کمک این پارامترها بتواند جایه‌جایی جسم متحرک را بیابد. در ادامه در رابطه با نحوه محاسبه مکان به کمک IMU و همچنین نتیجه‌ی به دست آمده در شبیه‌ساز توضیح خواهیم داد.

### ۲-۱- نحوه مکان‌یابی با IMU

به کمک حسگر IMU می‌توان کار مکان‌یابی محلی را انجام داد. در حقیقت با توجه به اطلاعات رسیده از شتاب‌سنج‌ها و ژیروسکوپ می‌توان جایه‌جایی نسبی و سرعت نسبی را محاسبه کرد و با جمع این مقادیر به دست آمده با مقدار اولیه، به مقدار نهایی رسید. به صورت تئوری به نظر می‌رسد که این کار باید به طور دقیقی به ما خروجی دهد؛ اما باید این نکته را در نظر گرفت که اندکی خطا در اطلاعات رسیده، در فرکанс بالای این حسگر، به صورت نمایی تجمعی شده و پس از مدت کمی با خطای بالایی مواجه خواهیم بود.

برای آنکه به کمک کمیت‌هایی مانند شتاب خطی و سرعت زاویه‌ای بتوانیم موقعیت را به دست آوریم، نیازمند روابط فیزیکی هستیم تا رابطه‌ی شتاب خطی با مکان و سرعت زاویه‌ای را با مقدار چرخش و زاویه‌ی قرارگیری جسم متحرک بیان کند. بر این اساس می‌توان از روابط (۱-۵) و (۲-۵) استفاده کرد.

$$x = \frac{1}{2}at^2 + v_0t + x_0 \quad (1-5)$$

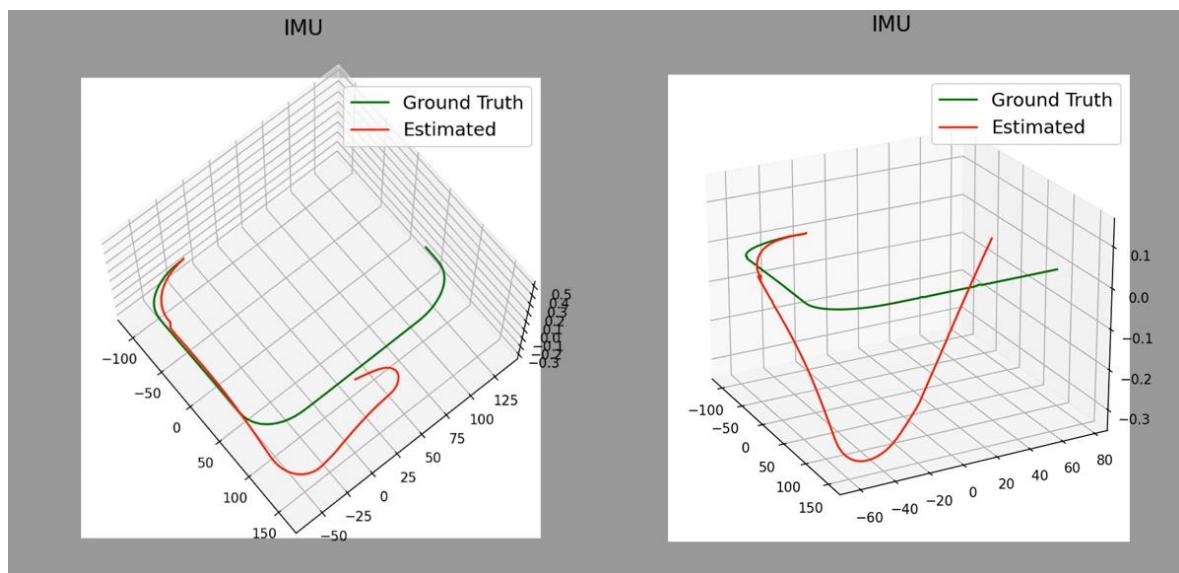
$$\theta = \theta_0 + \int \omega dt \quad (2-5)$$

به این ترتیب به کمک کیلومترشماری با IMU می‌توان مکان خودرو را در هر لحظه به دست آورد. در روابط بالا همانطور که مشاهده می‌شود نیاز است که شرایط اولیه مانند سرعت و مکان اولیه مشخص باشند تا بتوان سرعت و مکان فعلی را تخمین زد.

## ۲-۲-۵ - نتایج مکان‌یابی با IMU در کارلا

ابتدا لازم است تا در سمت مشتری پس از اتصال به سرویس دهنده و ساخت یک جهان، یک حسگر IMU را به خودوری موردنظر متصل نماییم. فرکانس داده‌برداری را بر روی  $200$  هرتز تنظیم می‌نماییم. همچنین لازم است مقداری نویز برای آن در نظر بگیریم. در شبیه‌ساز کارلا برای شتاب و ژیروسکوپ در هر سه راستای فضایی مقدار نویز منحصر به فردی را تنظیم نمود.

در نهایت کدی برای انجام فرآیند کیلومتر شماری با IMU می‌نویسیم که با گرفتن یک مقدار اولیه برای شروع مکان خودرو را با گرفتن مقادیر شتاب و سرعت زاویه‌ای محاسبه نماید. در نهایت نتیجه‌ی به دست آمده در شکل ?? قابل مشاهده می‌باشد.



نمودار ۲-۵ تخمین مکان خودرو در سطح شهر با حسگر IMU

همانطور که مشاهده می‌شود پس از مدت زمانی و با طی مسیر اندکی دیگر داده‌های محاسبه شده برای موقعیت خودرو از طریق این حسگر قابل اعتماد نمی‌باشند و خطای بسیار زیاد می‌شود. بنابراین زمانی که برای مدتی داده‌های GPS جهت تصحیح خطای در دسترس نباشند، مکان محاسبه شده به کمک IMU برای ما قابل اطمینان نمی‌باشد و باید به دنبال چاره‌ای برای آن بود.

### ۳-۵- مکان‌یابی به کمک لایدار

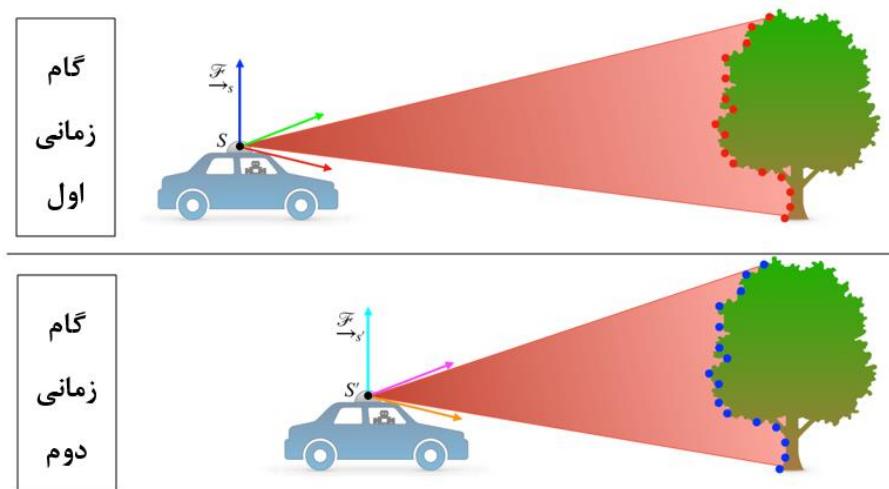
تا این قسمت تا حد خوبی با لایدار آشنا شده‌ایم. لایدار نیز مانند IMU مکان خودرو را به صورت مستقیم نمی‌دهد و باید به کمک الگوریتمی بتوانیم مکان خودرو را به دست آوریم. خروجی این حسگر یک سری ابرنقاط می‌باشد که باید به نحوی به کمک روش کیلومترشماری با لایدار بتوانیم جابه‌جایی نسبی و درنهایت موقعیت خودرو را به دست آوریم.

در این بخش نیز می‌خواهیم ببینیم که حسگر لایدار به تنها یک چه دقیقی در تخمین مکان خودرو در شبیه ساز دارد و در مراحل بعدی با الگوریتم به دست آمده در این مرحله، می‌توان به صورت عملی نیز دقیق این الگوریتم و حسگر را در مکان‌یابی به دست آورد.

### ۳-۱- نحوه مکان‌یابی با لایدار

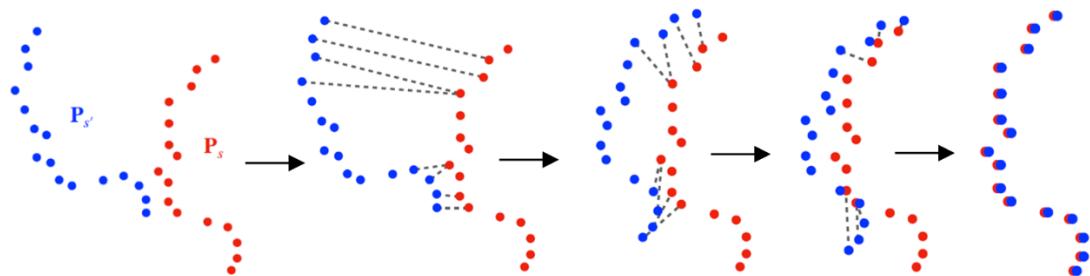
حسگر لایدار در هر ثانیه به تعداد ۱۰ ابر نقطه که هر کدام حداقل شامل یک میلیون نقطه از محیط اطراف هستند را به عنوان خروجی می‌دهد. ما نیاز داریم که جابه‌جایی نسبی را به کمک هر دو اسکن متوالی به دست آوریم.

فرض کنید که حسگر بر روی یک خودرو متصل شده است و در اطراف آن تنها یک درخت وجود دارد. در واحد زمانی اول یک اسکن از محیط شامل آن درخت را به عنوان خروجی می‌دهد. از آنجایی که این خودرو با سرعت در حال حرکت است، کمی جلوتر رفته در واحد زمانی بعد یک اسکن دیگر را از آن محیط به ما می‌دهد. در هردو اسکن متوالی نقاطی از درخت ثبت شده است اما چون وسیله‌ی نقلیه به آن نزدیک‌تر شده است، مقیاس و فاصله‌ی نقاط نسبت به دستگاه مختصات متصل به لایدار تغییر کرده است. با توجه به این تغییراتی که در هر دو اسکن متوالی ثبت شده است می‌توان به نحوی جابه‌جایی نسبی که خودرو داشته است را به دست آورد. مثال گفته شده به خوبی در شکل ۴-۵ نمایش داده شده است.



شکل ۴-۵ داده‌برداری خودرو با لایdar در دو گام زمانی متوالی.

سپس به یک الگوریتم ثبت ابرنقاط نیاز داریم تا بتواند این دو ابر نقطه را بر هم منطبق نماید و جابه‌جایی نسبی و یا به عبارتی ماتریس تبدیل نسبی که برای این انطباق نیاز است را محاسبه نماید. برای این امر از الگوریتم ICP استفاده خواهیم کرد. این الگوریتم سعی می‌کند تا فاصله‌ی بین نقاط متناظر با کمینه کردن تابع خطای دست آورد. نمونه‌ای از نحوه‌ی تطابق اسکن در شکل ۵-۵ به خوبی نمایش داده شده است.



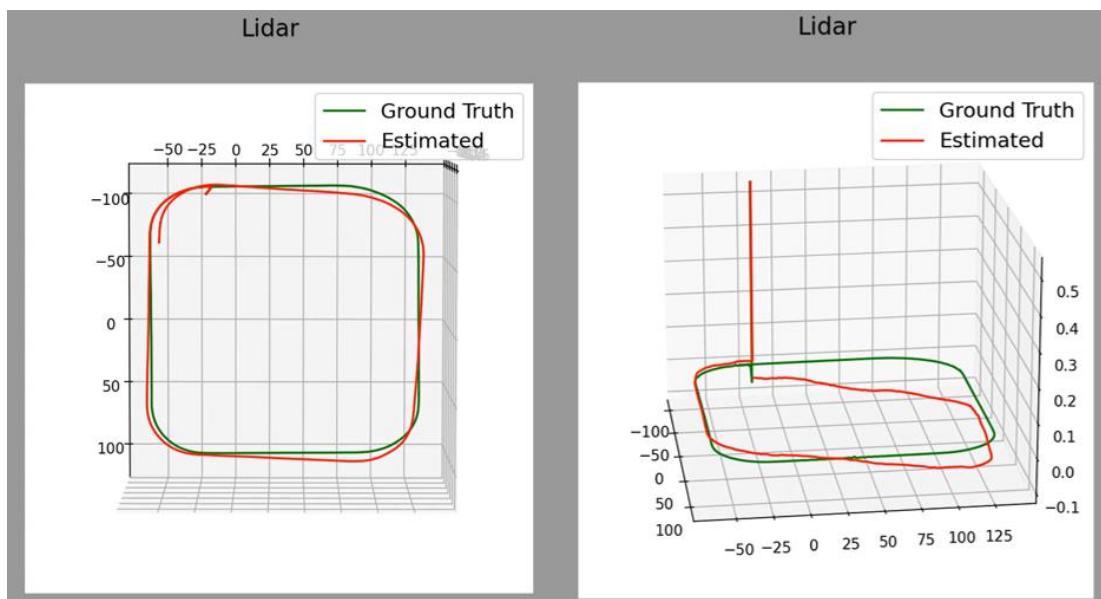
شکل ۵-۵ نحوه‌ی تطابق اسکن برای یافتن جابه‌جایی نسبی بین دو ابرنقاط [۳۱].

به این ترتیب با به دست آوردن ماتریس‌های تبدیل نسبی می‌توان جابه‌جایی و چرخش نسبی بین هر دو گام زمانی را به دست آورد و به این ترتیب با جمع آن‌ها باهم، مکان خودرو را با دقیقیت خوبی به دست آورد.

### ۲-۳-۵ - نتایج مکان‌یابی با لایدار در کارلا

ابتدا لازم است تا در سمت مشتری پس از اتصال به سرویس دهنده و ساخت یک جهان، یک حسگر لایدار را به خودروی موردنظر متصل نماییم. این حسگر در ارتفاع ۱.۷ متری و بر روی سقف خودرو متصل می‌شود. فرکانس داده‌برداری را بر روی ۱۰ هرتز تنظیم می‌نماییم. همچنین لازم است مقداری نویز برابر با ۰.۰۲ متر را برای آن در نظر بگیریم.

با کمک نسخه‌ی نقطه‌ی صفحه‌ی الگوریتم ICP، کدی برای کیلومترشماری به کمک لایدار می‌نویسیم. همچنین لزومی به نوشتن الگوریتم نمی‌باشد و می‌توان از کتابخانه‌ی Open3d استفاده نمود. در نهایت با اجرای کد مشتری، خودرو در شهر به حرکت درآمده و با ارسال ابرنقاط، الگوریتم کیلومترشماری باید جایه‌جایی نسبی محاسبه شده را با مکان اولیه جمع نماید تا مکان نهایی به دست آید. در نهایت نتیجه در نمودار ۳-۵ آمده است.



نمودار ۳-۵ تخمین مکان خودرو در سطح شهر با حسگر لایدار.

همانطور که مشاهده می‌شود مکان خودرو با دقت خوبی تخمین زده شده است. این دقت در صفحه‌ی  $x$ - $y$  بیشتر است و در راستای  $z$  خطأ بیشتر می‌شود. علت این امر آن است که در لایدار ۶۴ لایه درواقع در راستای  $z$  ما ۶۴ نقطه خواهیم داشت ولی در مقابل در صفحه‌ی  $y$ - $x$  با توجه به چرخش کامل لایدار ۳۶۰ نقطه داریم. به نوعی بایاس بیشتر بر روی تطابق در صفحه‌ی  $y$ - $x$  می‌باشد.

## ۴-۵ - مکانیابی به کمک ترکیب حسگرها

همانطور که در بخش قبل مشاهده کردیم، مکانیابی با هریک از حسگرها به تنها ی دقت کافی ندارد. برای این منظور لازم است از ترکیب چندین حسگر برای بالابردن دقت و همچنانین برطرف کردن نقصان‌های هریک از حسگرها بپردازیم.

در این بخش ابتدا به ترکیب دو حسگر IMU و GNSS می‌پردازیم و نتیجه را بررسی می‌کنیم. در گام بعدی حسگر لایدار را نیز می‌افزاییم و نتیجه را مشاهده می‌کنیم. لازم به ذکر است که تنظیمات نویز و فرکанс و دیگر تنظیمات حسگرها را تغییر نمی‌دهیم تا نتایج قابل مقایسه باشند.

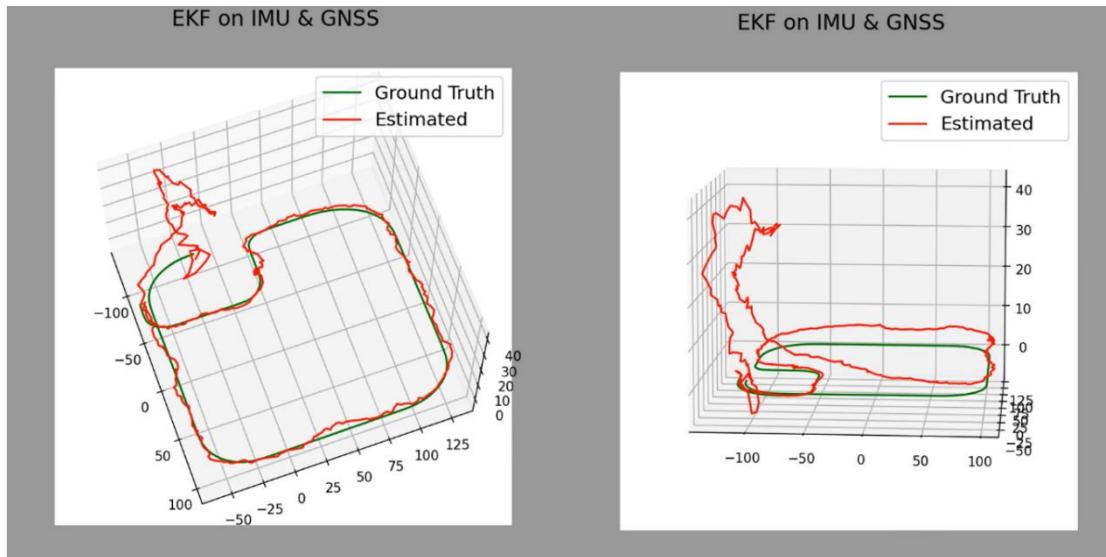
## ۴-۱-۵ - ترکیب حسگر GNSS و IMU

باتوجه به مبانی و روابطی که در فصل قبل در رابطه با فیلتر کالمن توسعه یافته خواندیم، در این قسمت به دنبال آنیم که اطلاعات دو حسگر GPS و IMU را باهم ترکیب نماییم.

در این فیلتر دو بخش داریم؛ یک بخش برای پیش‌بینی و بخش دیگر برای تصحیح می‌باشد. در بخش پیش‌بینی ما به کمک اطلاعات دریافت شده از IMU و مدل دینامیکی که برای حرکت سامانه در نظر گرفتیم، می‌توانیم حالت فعلی را تخمین بزنیم. نکته‌ای که باید در نظر گرفت، این مدل دینامیکی نیازمند شرایط اولیه می‌باشد و همانطور که در بخش قبل دیدیم اگر یک شرط اولیه در ابتدای مکانیابی تنظیم شود و در مراحل بعد فقط به صورت نسبی جایه‌جایی‌ها را با یکدیگر جمع کنیم، پس از مدت زمانی خطا بسیار زیاد می‌شود. با کمک فیلتر کالمن و اطلاعات رسیده توسط GNSS می‌توان این شرایط اولیه را با رسیدن داده از سمت آن، آپدیت کرد. به این ترتیب هم می‌توانیم با فرکانس بالای حسگر IMU مکان خودرو را در هر لحظه داشته باشیم و هم آنکه از دقت GNSS بهره ببریم.

نتیجه‌ی این بررسی در نمودار ۴-۵ قابل مشاهده می‌باشد. طبق این نمودار دقت مکانیابی در صفحه‌ی X-y نسبت به حالت GPS یا IMU ای تنهای، با وجود نویزی که همچنان وجود دارد، افزایش یافته است. همچنانین در ابتدای کار لازم است ماتریس‌های کواریانس مقداردهی اولیه شوند که این کار به صورت رندوم انجام می‌شود. به همین دلیل در ابتدای مکانیابی اختلاف زیادی در مکان پیش‌بینی شده مشاهده

می‌شود اما پس از چندین بار به روزرسانی این ماتریس‌ها، جواب موردنظر همگرا می‌شود و هرچه بیشتر می‌گذرد خطای که در  $Z$  نیز مشاهده می‌شود کمینه می‌شود.



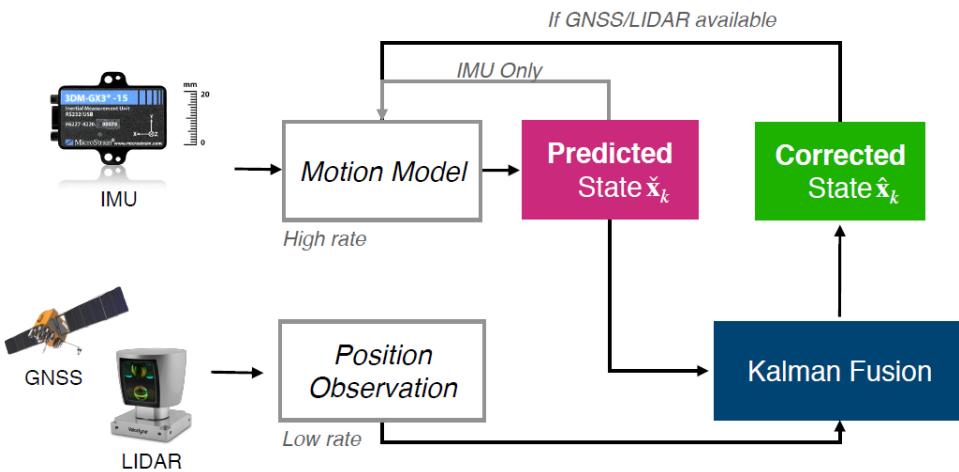
نمودار ۴-۵ تخمین مکان خودرو با ترکیب حسگرهای IMU و GNSS

#### ۲-۴-۵- ترکیب حسگر GNSS و IMU و لایدار

همانطور که در بخش قبل دیدیم همچنان امکان بهبود دقیق برای مکان‌یابی وجود دارد. همچنین نیازمند آنیم که شرایط اولیه زودتر همگرا شود تا مکان خودرو را به صورت دقیق‌تری داشته باشیم. با توجه به آنکه حسگر لایدار به تنها یک نیز دقیق خوبی دارد می‌توانیم آن را نیز در این ترکیب به کار ببریم.

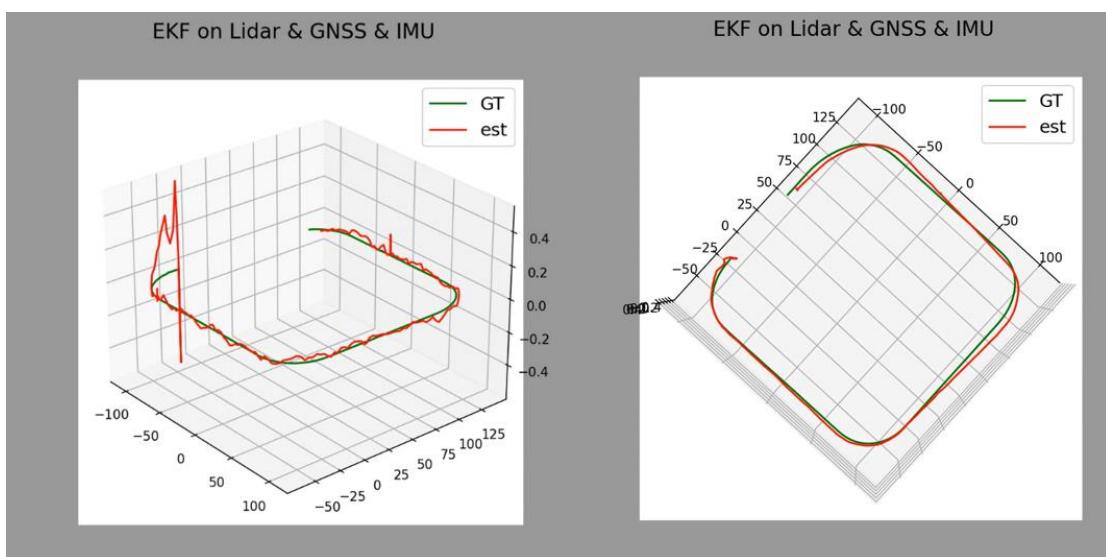
لایدار نیز مانند IMU پس از گذشت زمان دقت‌کاهش می‌یابد و به تنها یک برای مسافت طولانی کافی نمی‌باشد. همچنین در راستای  $Z$  کمی خطای خواهیم داشت، به همین دلیل عمل ترکیب آن‌ها باعث کاهش خطای خواهد شد.

ما در این بخش به کمک فیلتر کالمن توسعه یافته سعی می‌کنیم حسگرهای را ترکیب نماییم. در بخش پیش‌بینی حالت به کمک IMU می‌توان مقدار فعلی را تخمین زد و به کمک لایدار و GNSS عمل تصحیح را انجام داد. در شکل ۶-۵ نحوه این ترکیب نمایش داده شده است.



شکل ۵-۶ نحوه ترکیب حسگرهای IMU و GNSS و لایدار با فیلتر کالمان [۳۱].

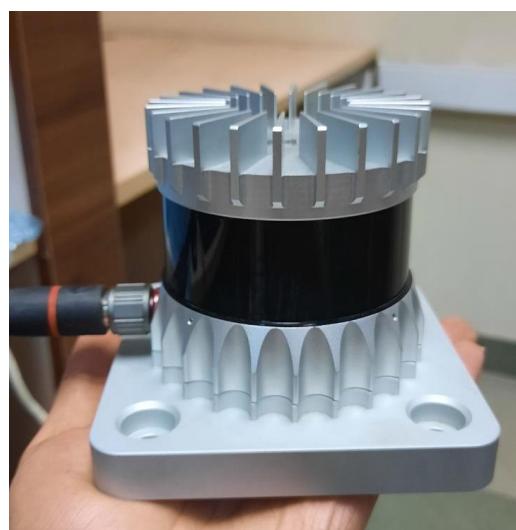
در نهایت با اعمال این روش نتیجه‌ی مطلوب ما در نمودار ۵-۵ به دست می‌آید. همانطور که دیده می‌شود نمودار ما نرم‌تر و دقیق‌تر شده است و از تمامی ویژگی‌هی مطلوب این حسگرها استفاده نموده‌ایم.



نمودار ۵-۵ تخمین مکان خودرو با ترکیب حسگرهای IMU و GNSS و لایدار.

## ۵-۵- مکانیابی با لایدار به صورت عملی

در این بخش می‌خواهیم بررسی نماییم که آیا مکان‌یابی با حسگر لایدار به صورت عملی نیز کارایی موردنظر را دارد یا خیر. برای این منظور از یک لایدار ۶۴ لایه‌ی آستر کمک گرفتیم. فرکانس این حسگر ۱۰ هرتز و برد آن ۱۰۰ متر می‌باشد که از جمله حسگرهای لایدار با کیفیت تولید شده می‌باشد. نمای این حسگر مورد استفاده را در شکل ۷-۵ می‌توانید مشاهده نمایید.



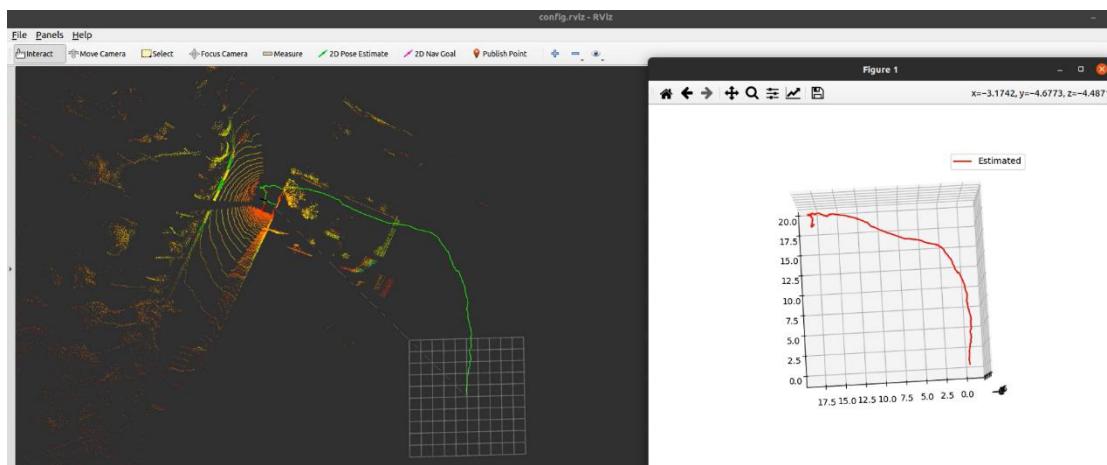
شکل ۷-۵ حسگر لایدار آستر مورد استفاده.

در گام بعد جسم متحرک مورد استفاده یک گاری می‌باشد که حسگر و لپتاپ متصل به آن را بر روی آن قرار دادیم. اتصالات با توجه بخش‌های قبلی با کمک راهنمای استفاده از حسگر انجام شده است. تصویری از مدار بسته و گاری در شکل ۸-۵ قرار گرفته است.



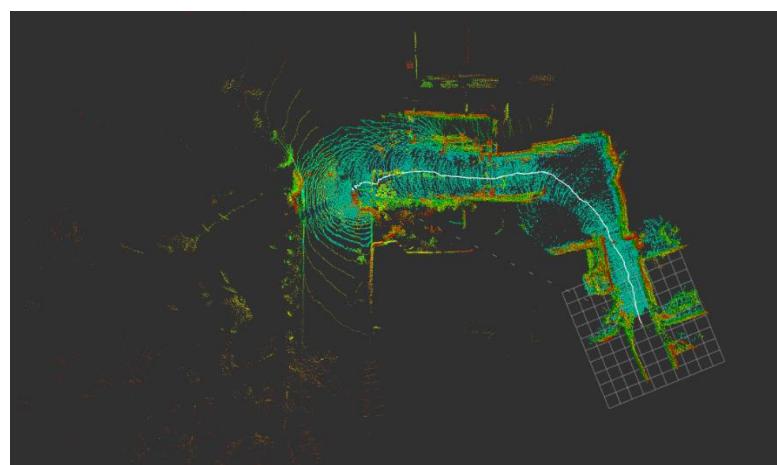
شکل ۸-۵ حسگر لایدار آستر مورد استفاده.

این آزمایش را در طبقه‌ی همکف دانشکده‌ی مهندسی کامپیوتر دانشگاه امیرکبیر انجام دادیم. در نهایت داده‌ها را در یک فایل Rosbag جمع آوری کرده و مکان‌یابی را به صورت بلادرنگ انجام دادیم. الگوریتم‌های مورد استفاده همان الگوریتم‌هایی است که در بخش شبیه‌سازی توسعه دادیم. نتیجه در نمودار ۶-۵ قابل مشاهده است. مسیر به دست آمده در یک پلات و هم در ابزار Rviz نمایش داده شده است.

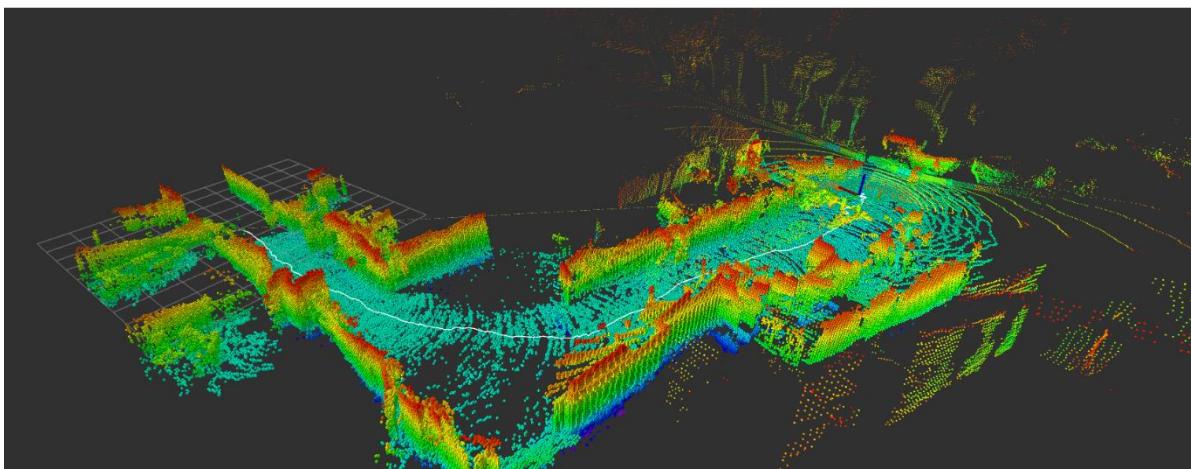


نمودار ۶-۵ مکان‌یابی با لایدار در دانشکده‌ی کامپیوتر دانشگاه امیرکبیر.

حال با توجه به آنکه عمل مکان‌یابی را با موفقیت انجام دادیم، می‌توانیم نقشه‌ی سه بعدی طبقه‌ی همکف را نیز تولید نماییم. برای این امر با کمک ROS و همچنین استفاده از پکیج Octomap توانستیم یک نقشه‌ی نمونه‌برداری شده را به دست آوریم. نقشه‌ی به دست آمده در شکل ۹-۵ و شکل ۱۰-۵ قابل مشاهده است.

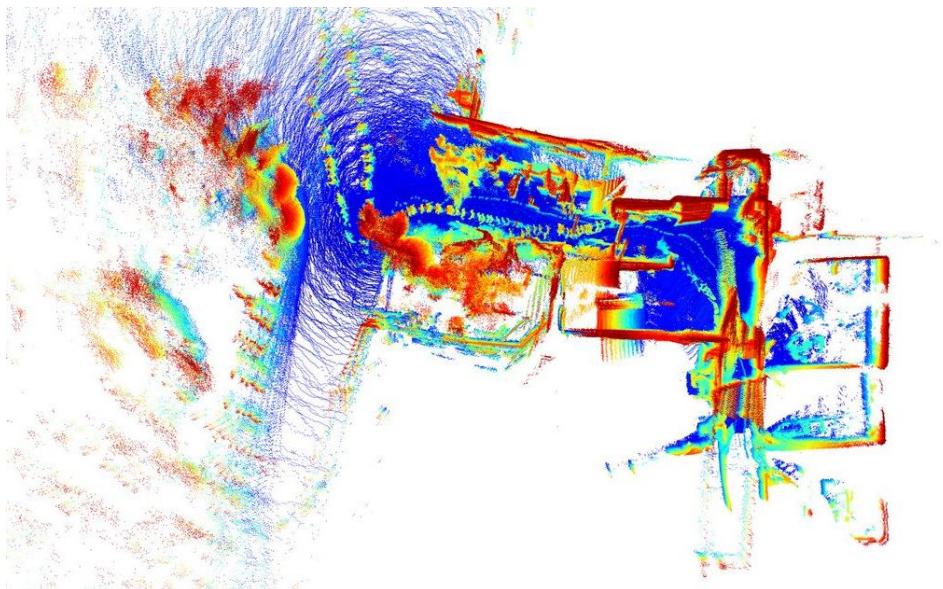


شکل ۹-۵ نمای بالای نقشه‌ی نمونه‌برداری شده از دانشکده کامپیوتر.

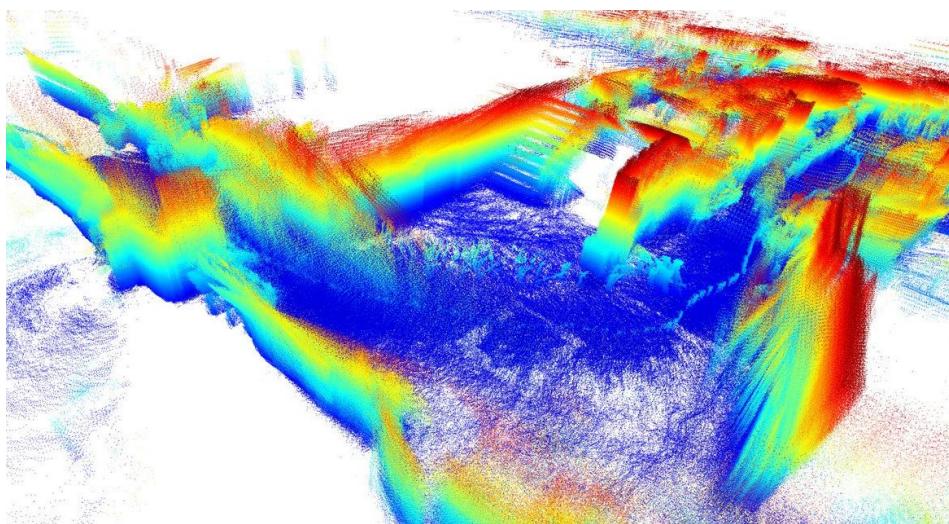


شکل ۵-۱۰ نمای ایزومتریک نقشه‌ی نمونه برداری شده از دانشکده کامپیووتر.

نقشه‌های ساخته شده‌ی قبل با توجه به الگوریتم مورد استفاده در Octomap به نوعی بهینه و نمونه برداری شده از اسکن‌ها می‌باشند. حالت دیگر آن است که با توجه به ماتریس‌های تبدیل به دست آمده از طریق مکان‌یابی به صورت مستقیم همه‌ی نقاط را بدون نمونه برداری کنار یکدیگر قرار دهیم که در این حالت به علت وجود نویزها و همچنین خطای مکان‌یابی، نقشه‌ی مطلوبی به دست نمی‌آید. برای مثال به نقشه‌ی به دست آمده در شکل ۱۱-۵ و ۱۲-۵ توجه نمایید.



شکل ۵-۱۱ نمای بالای نقشه‌ی بدون نمونه برداری از دانشکده کامپیووتر.



شکل ۱۲-۵ نمای ایزومتریک نقشه‌ی بدون نمونه برداری از دانشکده کامپیووتر.

داده‌های جمع‌آوری شده برای این بخش در قالب Rosag از طریق لینک<sup>۱</sup> قابل دسترس است.

---

<sup>1</sup> [https://drive.google.com/file/d/1MuvQYVi\\_-BfOLBubADjmwZoaPKmUMGoe/view?usp=sharing](https://drive.google.com/file/d/1MuvQYVi_-BfOLBubADjmwZoaPKmUMGoe/view?usp=sharing)

## فصل ششم

### جمع‌بندی و پیشنهادات

## جمع‌بندی

باتوجه به پیشرفت چشمگیر صنعت خودروسازی و تمایل شرکتها برای خودران‌سازی خودروها، افزایش دقت در بسیاری از قابلیت‌ها اهمیت می‌یابد. یکی از مهم‌ترین قابلیت‌ها در اتومبیل‌های خودران، تعیین مکان خودرو می‌باشد. چنانچه مکان دقیق خودرو برای مدتی در دسترس نباشد، می‌تواند موجب خسارات جبران ناپذیری شود.

در این پژوهش به دنبال یافتن راهی برای افزایش دقت مکان‌یابی و کمینه کردن احتمال از دست رفتن مکان دقیق خودرو می‌باشیم.

در مکان‌یابی نخستین حسگری که به ذهنمان می‌رسد، حسگر GNSS می‌باشد. این حسگر نرخ به روزرسانی پایینی دارد و در داخل تونل‌های شهری و پارکینگ‌های زیرزمینی امکان محاسبه‌ی مکان خودرو را ندارد. همچنین دقت آن را در مکان‌یابی در شبیه‌ساز کارلا مشاهده نمودیم که به علت اختشاشات می‌تواند در بعضی موارد دقت مطلوب را به ما ندهد. نکته‌ی مثبت آن، این است که می‌تواند به صورت جهانی موقعیت را با خطای قابل ملاحظه‌ای در اختیار قراردهد.

یکی از حسگرهای مهم IMU می‌باشد که شامل شتاب سنج و ژیروسکوپ می‌باشد. با اندازه‌گیری سرعت زاویه‌ای و شتاب خطی و همچنین در اختیار داشتن مدل دینامیکی حرکت خودرو، می‌توان مکان خودرو را به صورت محلی تعیین نمود. نکته‌ی مثبت آن فرکانس بالای به روزرسانی و وابسته نبودن به منابع خارجی می‌باشد. اما همانطور که نتیجه را در شبیه‌ساز دیدیم، بعد از مدت اندکی به علت اثر جمع شوندگی خطای می‌تواند خطای بسیار زیادی در تعیین مکان داشته باشد.

در مرحله‌ی بعد تصمیم گرفتیم تا این دو حسگر را باهم ترکیب نموده تا هم مشکل فرکانس پایین GPS به نوعی حل شود و هم شرایط اولیه‌ی مدل دینامیکی به مرور آپدیت شود و از جمع شوندگی خطای جلوگیری شود. همانطور که در خروجی‌ها مشاهده نمودیم این روش خطای کمتری نسبت به استفاده‌ی تنها از هریک از حسگرها دارد ولی به صورت کلی، اندکی خطای دارد که می‌توان آن را نیز کمتر نمود.

حسگر لایدار که با کمک پرتاب پرتوهای لیزر به محیط و دریافت آن‌ها می‌تواند نقشه‌ی محیط را درک کند، برای تشخیص موائع و نقشه‌برداری کاربرد دارد. حسگر لایدار مورد استفاده در این پژوهش فرکانس ۱۰ هرتز دارد و خروجی این حسگر در هر اسکن یک ابرنقطاط از محیط می‌باشد.

با دریافت این ابرنقطاً و اجرای الگوریتم ثبت نقاط می‌توان جابه‌جایی نسبی را محاسبه کرد و به کمک روش کیلومترشماری با لایدار مکان خودرو را به صورت محلی محاسبه نمود. همانطور که در شبیه‌ساز مشاهده کردیم این روش نیز می‌تواند دچار جمع شوندگی خطا شود اما این مقدار حدود ۲۰ سانتی‌متر پس از طی مسافت ۱۰۰ متر می‌باشد که خطای قابل قبولی است. علاوه بر آن، این حسگر وابستگی به منابع خارجی و میزان روشنایی محیط ندارد و فرکанс ۱۰ برابری نسبت به GPS دارد.

برای آنکه کارایی مکان‌یابی با حسگر لایدار به تنها‌یی را مشاهده نماییم، تصمیم به آزمایش عملی این روش گرفتیم. برای این منظور به جمع آوری داده‌های ابرنقطاً در طبقه‌ی همکف دانشکده‌ی کامپیوتر دانشگاه امیرکبیر، پرداختیم و الگوریتم مکان‌یابی را بر روی آن اجرا کردیم که نتایج دقت نسبتاً بالایی داشت. همچنین با توجه به موفقیت آمیز بودن مکان‌یابی، توانستیم نقشه‌ای سه بعدی از مسیر طی شده به دست آوریم.

در گام نهایی این پژوهش، به ترکیب هر سه حسگر نامبرده در شبیه‌ساز کارلا پرداختیم و مشاهده کردیم که دقت مکان‌یابی افزایش یافته و همچنین تا حد مطلوبی عیوبی که هریک از روش‌ها به تنها‌یی داشتند را بپوشانیم.

## پیشنهادات

باتوجه به آنکه روش‌های مختلفی برای ثبت ابرنقطاً وجود دارد می‌توان الگوریتم‌های مختلفی را برای این امر آزمایش کرد و روشی بهینه‌تر را برای ثبت نقاط برگزید. در این آزمایش روش‌های مختلف الگوریتم ICP بررسی شد که در این بین الگوریتم نقطه به صفحه دقت و سرعت بیشتری را داشت. در پژوهش‌های آتی می‌توان به روش NDT و یا روش‌های مبتنی بر هوش مصنوعی و یادگیر ماشین پرداخت.

نکته‌ی دیگر آن است که می‌دانیم حسگرهای مختلفی از جمله دوربین‌ها در خودروها کاربرد زیادی دارند و می‌توان از آن‌ها نیز برای افزایش دقت مکان‌یابی بهره گرفت. بنابراین مکان خودرو را به روش کیلومتر شماری و محاسبه‌ی جابه‌جایی نسبی با کمک عکس‌های متوالی از دوربین‌ها محاسبه نمود و مکان خودرو را به صورت محلی به دست آورد و با داده‌های سایر حسگرها، به کمک روش‌های ترکیب حسگرها، ترکیب نمود.

از جمله نکات دیگری که بردقت مکان‌یابی به کمک چندحسگر، تاثیر دارد؛ روش ترکیب حسگرهای به کارگرفته شده می‌باشد. ما در این پژوهش از روش فیلتر کالمن توسعه یافته که یکی از روش‌های بسیار خوب برای ترکیب حسگرها می‌باشد استفاده نمودیم؛ اما می‌توان روش‌های مبتنی بر هوش مصنوعی برای بالا بردن دقیق بودن برداشت این روش را در اینجا معرفی نماییم.

در این پژوهش از شبیه‌ساز کارلا بر روی یک سیستم کامپیوتروی متوسط استفاده شد. شبیه‌سازهای دیگری نیز در حوزه‌ی خودروهای خودران توسعه یافته‌اند که می‌توان از آن‌ها نیز کمک گرفت. به هرجهت هرچه دقیق‌تر موتور فیزیکی به کارگرفته شده برای شبیه‌سازی و همچنین قدرت سخت‌افزار به کار گرفته شده، بیشتر باشد، دقیق‌تر در نتایج به دست آمده نیز بیشتر خواهد شد.

## منابع و مراجع

- [1] Se, S., D. Lowe, and J. Little. *Local and global localization for mobile robots using visual landmarks.* in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180).* 2001.
- [2] Meng, X., H. Wang, and B. Liu, *A robust vehicle localization approach based on GNSS/IMU/DMI/LiDAR sensor fusion for autonomous vehicles.* Sensors, 2017. **17**: p. 2140.
- [3] Zhang, J. and S. Singh, *LOAM : Lidar Odometry and Mapping in real-time.* Robotics: Science and Systems Conference (RSS), 2014: p. 109-111.
- [4] Gu, Y., L.-T. Hsu, and S. Kamijo, *Towards lane-level traffic monitoring in urban environment using precise probe vehicle data derived from three-dimensional map aided differential GNSS.* IATSS Research, 2018. **42**.
- [5] Zheng, X. and J. Zhu, *Efficient LiDAR Odometry for Autonomous Driving.* 2021.
- [6] Graeter, J., A. Wilczynski, and M. Lauer, *LIMO: Lidar-Monocular Visual Odometry.* 2018.
- [7] Zhang, J., M. Kaess, and S. Singh, *Real-time Depth Enhanced Monocular Odometry.* 2014.
- [8] Scherer, S., et al., *River mapping from a flying robot: state estimation, river detection, and obstacle mapping.* Autonomous Robots, 2012. **33**(1): p. 189-214.
- [9] Liu, Y., et al. *Real-Time Lidar Odometry and Mapping with Loop Closure.* Sensors, 2022. **22**, DOI: 10.3390/s22124373.
- [10] Zhou, B., et al. *A LiDAR Odometry for Outdoor Mobile Robots Using NDT Based Scan Matching in GPS-denied environments.* in *2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER).* 2017.
- [11] Chen, S., et al., *NDT-LOAM: A Real-Time Lidar Odometry and Mapping With Weighted NDT and LFA.* IEEE Sensors Journal, 2022. **22**(4): p. 3660-3671.
- [12] Montemerlo, M., S. Thrun, D. Koller, and B. Wegbreit, *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem.* 2002.
- [13] CARLA. *Introduction.* Available from: [https://carla.readthedocs.io/en/latest/start\\_introduction/](https://carla.readthedocs.io/en/latest/start_introduction/).
- [14] Iranros. *What is ROS?* ; Available from: <http://iranros.com/why-ros/what-is-ros/>.

- [15] ROS. *Understanding nodes.* Available from: <https://docs.ros.org/en/foxy/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Nodes/Understanding-ROS2-Nodes.html>.
- [16] Leibs, J. *From the Evolution of Rosbag to the Future of AI Tooling.* 2023; Available from: <https://www.rerun.io/blog/rosbag>.
- [17] Ouster, *Software User Manual.* 2021.
- [18] UNAM, B.L. *Lessons on Mobile Robot Localization and Kalman Filters.* 2023; Available from: [https://github.com/RobotJustina/MRS\\_EKF\\_MatLab](https://github.com/RobotJustina/MRS_EKF_MatLab).
- [19] Auat Cheein, F.A., et al., *SLAM algorithm applied to robotics assistance for navigation in unknown environments.* Journal of NeuroEngineering and Rehabilitation, 2010. **7**(1): p. 10.
- [20] flyability. *What is SLAM?* ; Available from: <https://www.flyability.com/simultaneous-localization-and-mapping>.
- [21] MathWorks. *SLAM.* Available from: <https://www.mathworks.com/discovery/slam.html>.
- [22] Wong, K., Y. Gu, and S. Kamijo, *Mapping for Autonomous Driving: Opportunities and Challenges.* IEEE Intelligent Transportation Systems Magazine, 2020. **PP**.
- [23] Khedkar, A. *Mapping Techniques in Artificial Intelligence and Robotics.* 2021; Available from: <https://www.geeksforgeeks.org/mapping-techniques-in-artificial-intelligence-and-robotics/>.
- [24] Revopoint. *Point Cloud and 3D Image.* 2020; Available from: <https://www.revopoint3d.com/point-cloud-and-3d-image/>.
- [25] Gu, X., X. Wang, and Y. Guo, *A Review of Research on Point Cloud Registration Methods.* IOP Conference Series: Materials Science and Engineering, 2020. **782**: p. 022070.
- [26] Think\_autonomous. *Point Cloud Registration: Beyond the Iterative Closest Point Algorithm.* 2023; Available from: <https://www.thinkautonomous.ai/blog/point-cloud-registration/>.
- [27] Besl, P. and H.D. McKay, *A method for registration of 3-D shapes.* IEEE Trans Pattern Anal Mach Intell. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1992. **14**: p. 239-256.
- [28] Biber, P. and W. Straßer, *The Normal Distributions Transform: A New Approach to Laser Scan Matching.* Vol. 3. 2003. 2743-2748 vol.3.
- [29] Pang, S., et al. *3D Scan Registration Based Localization for Autonomous Vehicles - A Comparison of NDT and ICP under Realistic Conditions.* in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall).* 2018.
- [30] Galar, D. and U. Kumar, *Chapter 1 - Sensors and Data Acquisition,* in *eMaintenance*, D. Galar and U. Kumar, Editors. 2017, Academic Press. p. 1-72.

- [31] Jonathan Kelly , S.W., *State Estimation and Localization for Self-Driving Cars.* Coursera.
- [32] National.Ocean.Service. *The Global Positioning System.* Available from: [https://oceanservice.noaa.gov/education/tutorial\\_geodesy/geo09\\_gps.html#](https://oceanservice.noaa.gov/education/tutorial_geodesy/geo09_gps.html#).

## پیوست ها

در ادامه تصاویری از بخش عملی این پژوهش قرار گرفته است.



شکل پ-۱ سیستم آماده شده برای داده برداری.



شکل پ-۲ آماده سازی Rosbag برای ضبط داده ها.



شکل پ-۳ جمع‌آوری داده از داخل طبقه همکف دانشکده کامپیوتر.



شکل پ-۴ جمع‌آوری داده بیرون از محوطه دانشکده.

---

**Abstract**

---

## **Abstract**

In the automotive and robotics industries, one of the most important capabilities of a mobile entity is its ability to locate itself accurately. If the location of the entity is not accurately determined in the environment, there is a possibility of incorrect navigation and accidents. Considering that autonomous vehicles can travel at high speeds, the rate of updating the entity's location must also be high so that we are constantly informed of its position.

In traditional methods, the location of a mobile entity was estimated using GNSS sensors. However, due to their low update rates and dependence on external satellite signals, they have always had their share of problems. In some research, IMU sensors with higher update rates have been used as a complement to address the shortcomings of GNSS.

In this research, we intend to use LiDAR, which is a widely used sensor in the autonomous industry, to improve the accuracy of location and address the shortcomings of two other sensors. For this purpose, the performance of this sensor for location estimation is demonstrated in both simulation and practical aspects. Then, by combining the outputs of these three mentioned sensors using an Extended Kalman filter, we achieved a more precise location estimation for the autonomous vehicle.

**Key Words:**

Localization, LiDAR, Autonomous Vehicle, Kalman Filter



**Amirkabir University of Technology  
(Tehran Polytechnic)**

**Computer Engineering Department**

**BSc Thesis**

**Lidar based self-localization in autonomous vehicles.**

**By  
Mahdi Rahmani**

**Supervisor  
Dr. Mahdi Javanmardi**

**September 2023**