Autonomous Navigation on Urban Sidewalks Under Winter Conditions

A THESIS SUBMITTED TO THE FACULTY OF THE
UNIVERSITY OF MINNESOTA
BY

Reed Austin Johnson

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Advisor: Dr. Max Donath

April, 2020

© Reed Johnson 2020

ALL RIGHTS RESERVED

Acknowledgments

I would like to first thank my committee chair, Professor Max Donath, whose guidance and expertise was crucial in allowing this work to be a success. I would also like to thank my committee members, Professor Rajesh Rajamani, Professor Timothy Kowalewski, and Professor Raphael Stern for their time and feedback based on their review of the thesis and the thesis defense. In addition, I would like to express my gratitude to Brian Davis and Chen-Fu Liao for their help and input from the beginning. Finally, I would like to thank Professor Brendan Englot, Tixiao Shao and the rest of the researchers in the Robust Field Autonomy Lab at Stevens Institute of Technology for their time and effort in providing us with the LeGO-LOAM algorithms upon which we have based our effort.

Dedication

This thesis is dedicated to my parents Michael and Chelle Johnson. My late father inspired me from a young age to be curious and always supported my creative imagination. I know he would be proud of me today.

Abstract

We describe a multi-step approach to facilitate autonomous navigation in snow by small vehicles in urban environments, allowing travel only on sidewalks and paved paths. Our objective is to have a vehicle autonomously navigate from point A on one urban block to point B on another block, crossing from one block to another only at curb-cuts, and stopping when pedestrians get in the way. A small mobile platform is first manually driven along the sidewalks to continuously record LIDAR and Global Navigation Satellite System (GNSS) data when little to no snow is on the ground. Our algorithm automatically post processes the data to generate a labeled traversability map. During this automated process, areas such as grass, sidewalks, stationary obstacles, roads and curb-cuts are identified. By differentiating between these areas using only LIDAR, the vehicle is later able to create a path for travel on only sidewalks or roads and not in other areas.

Our localization approach uses an Extended Kalman Filter to fuse the Lightweight and Ground-Optimized LIDAR Odometry and Mapping (LeGO-LOAM) approach with high accuracy GNSS where available, to allow for accurate localization even in areas with poor GNSS, which is often the case in cities and areas covered by tree canopy. This localization approach is used during the data capture stage, prior to the post-processing stage when labeled segmentation is performed, and again during real time autonomous navigation, carried out using the ROS navigation stack.

By using LIDAR odometry combined with GNSS, the robot is able to localize under many different weather conditions, including snow and rain, where other algorithms (e.g. AMCL) will likely fail. We were able to successfully have the vehicle autonomously plan and navigate a 1.6km path in an urban snow-covered neighborhood. Our methodology facilitates autonomous navigation functionality under most weather conditions including autonomous wheelchair navigation.

Table of Contents

List of Tables

List of Figures

1. Introduction

There has been much research and progress in the field of unmanned ground vehicles (UGVs). From the increasing popularity of autonomous cars to package delivering robots, the potential applications seem limitless. However, as much as the developers of UGVs would like you to believe, they are far from perfect. Many of these systems begin to fail in inclement weather, which is why they are primarily tested in locations with good weather year-round.

Accurate localization for UGVs in all environments and weather types is a research area important in northern climates. High accuracy Global Navigation Satellite System (GNSS) can provide centimeter-level position estimation in good quality reception areas. However, localization via GNSS is often not enough for most environments, such as urban areas, where tree canopy and buildings hinder the quality of reception. To solve this, GNSS is often combined with inertial measurement units (INS) [1], [2] or wheel encoders [3], [4] to supplement the localization in low quality GNSS areas. However, both INS and wheel encoders are susceptible to drift or slip and the localization scheme will fail if the vehicle travels too far between receiving accurate GNSS positions.

Simultaneous localization and mapping (SLAM) [5] is commonly used for real-time 6 degree-of-freedom pose estimation. SLAM is either performed using vision based systems [6] or LIDAR based systems [7]. Both SLAM and adaptive Monte Carlo localization (AMCL) match the current perception of the environment, via LIDAR or vision, to a given environment model or map. No matter the system, both SLAM and AMCL algorithms, are prone to limitations. Since SLAM and AMCL are based on identifying landmarks in the environment, the accuracy of the pose estimation is heavily dependent on these landmarks remaining static. If the environment changes, due to snow cover or construction, SLAM and AMCL will likely fail. To make localization more robust, SLAM has been fused with other sensors such as GNSS [8], [9] or INS [10]. However, these do not solve the underlying constraint of needing a fixed environment, which is not the case when there are various levels of snow on the ground.

LIDAR Odometry takes into account that there is a certain amount of sensor information overlap between consecutive LIDAR scans allowing for the pose transformation of the UGV to be estimated by matching the adjacent frames. Common LIDAR odometry methods can produce adequate pose estimation results [11], [12], However, these are computationally expensive and run at lower frame rates or require high powered computers. Shan *et al* proposed a lightweight and ground-optimized LIDAR and mapping method (LeGO-LOAM) [13] that is capable of real-time 6 degrees-of-freedom pose estimation on low power embedded systems. Due to its performance and low power requirements, LeGO-LOAM is used as our localization scheme. Wang *et al* [14] used a convolution neural network (CNN) to fuse vision, GNSS, and IMU to create a localization system for UGVs. Despite promising results, our work will consider deep learning methods out of scope.

Localization aside, mapping and understanding the environment of the UGV is important because the type of terrain the vehicle is moving on impacts its mobility [15]. When traveling from pt. A to pt. B, knowing where features such as roads, sidewalks, grass, and other obstacles are located is crucial in making smart decisions and planning a path between the two points. Moras *et al* [16] proposed using LIDAR to create a map of drivable space in real time, however, they do not classify the different segments of the ground. Smadja *et al* [17] use both LIDAR and vision systems to segment the ground and label the road as well as identify street signs. The localization schemes in both of these rely solely on GNSS and will fail in low quality reception or GNSS denied areas. Ai *et al* [18] performs automated and detailed sidewalk assessment using both LIDAR and vision. However, their system relies heavily on vision whose performance is highly susceptible to lighting and they do not identify other areas of the ground such as roads and grass. Shan *et al* [19] focused on autonomous terrain traversability mapping, including both the mapping and classification of the ground whether the vehicle can or cannot travel on it. This is not always desired because there are many situations where a vehicle has the physical capability to travel somewhere but it shouldn't.

Deep learning is becoming an increasingly popular solution for object detection and semantic environment segmentation [20] as well as road and road sign detection [21]. However, these solutions are typically vision based (as opposed to LIDAR) and perform poorly in weather conditions such as rain and snow.

In this work, we propose a multi-step approach to facilitate autonomous navigation by small vehicles in urban environments, allowing travel only on sidewalks and paved paths. Our objective is to have a vehicle autonomously navigate from point A on one urban block to point B on another block, crossing from one block to another only at curb-cuts, and stopping when pedestrians get in the way. A small mobile platform is first manually driven along the sidewalks to continuously record LIDAR and GNSS data when little to no snow is on the ground. Our algorithm post processes the data to generate a labeled traversability map. During this automated process, areas such as grass, sidewalks, stationary obstacles, roads and curb-cuts are identified. The ability to classify the ground in an environment, including sidewalks, facilitates appropriate decisions during navigation such as giving the robot information where it is acceptable to travel. By differentiating between these areas using only LIDAR, the vehicle is later able to create a path for travel on only sidewalks or roads and not in other areas, including locations that are covered in snow.

Our system uses an Extended Kalman Filter to fuse the Lightweight and Ground-Optimized LIDAR Odometry and Mapping (LeGO-LOAM) approach with high accuracy GNSS where available, to allow for accurate localization even in areas with poor GNSS, which is often the case in cities and areas covered by tree canopy. This localization approach is used during the data capture stage, prior to the post-processing stage when labeled segmentation is performed, and again during real time autonomous navigation, carried out using the ROS navigation stack.

There is a gap in previous research on robust localization and navigation; it has not been applied to poor weather conditions. Dynamic environments, such as snow, construction, growing vegetation, cause problems for traditional scan matching localization systems such as AMCL [22]. By using LeGO-LOAM combined with GNSS, our robot is able to localize under many different weather conditions, including snow and rain, where other algorithms (e.g. AMCL) will likely fail. We will describe in this work how we successfully created a system that allows the vehicle to autonomously plan and navigate several kilometer-long paths in urban snow covered neighborhoods. A potential application is autonomous wheelchair navigation that could be functional under most weather conditions.

1.2 Thesis Organization

In Section 2 we describe the vehicle platform and hardware used in this research. In Section 3, an overview of the neighborhood used as the testing environment is described. The localization technique used in this work is outlined in Section 4. In Section 5, the mapping and navigation procedure steps are described and in Section 6 our results are shown. Finally, Section 7 discusses our conclusions and potential future work.

2. Vehicle Platform and Hardware

This research uses the Husky UGV platform from Clearpath Robotics as a base vehicle for mounting hardware and sensors. The Husky comes equipped with four 330 mm diameter wheels with rotary encoders. The vehicle has external dimensions of 990 x 670 x 390 mm and has a max payload of 75 Kg with a top speed of 1 m/s. In the center of the vehicle is a weatherproof storage area where the electronics are stored, such as the computer and WIFI router. Dimensions of the Husky vehicle can be seen in Figure 1.

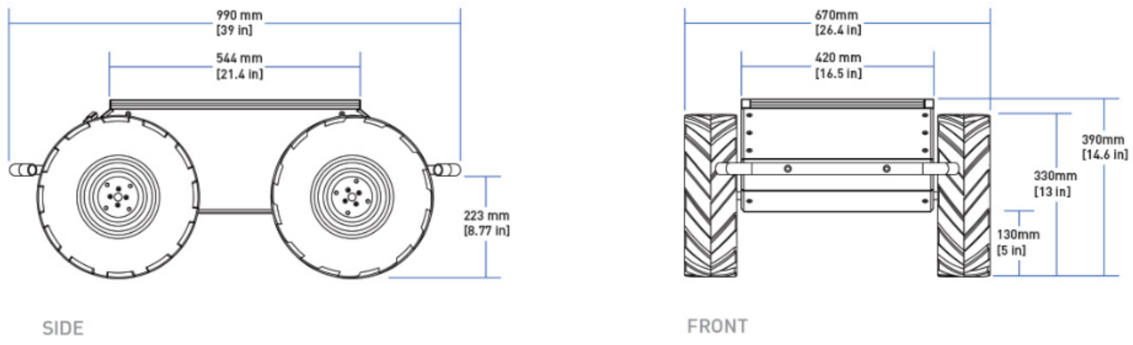

Figure 1 - Top and Side Husky dimensions [23]

The on-board computer in the Husky is a Mini-ITX single board computer with 3.6 GHz Intel i7-770 processor, 16 GB DDR4-2400 RAM, and a 512 GB SSD. The sensors mounted on the vehicle for this research are a high accuracy Trimble GNSS antenna and RTK receiver, a Velodyne VLP-16 LIDAR, and a Phidgets IMU. The Real-time Kinematic (RTK) GNSS receiver and antenna exhibits sub-centimeter accuracy in GNSS areas with good reception [24]. The Velodyne LIDAR (VLP-16) has 16 channels with a vertical field of view of -15° to +15° with each channel separated by 2° horizontally. There are 1800 LIDAR points returned for each channel in the LIDAR which has a maximum measurement range of 100 m. The LIDAR returns up to 300,000 points/second [25]. The Phidgets IMU has a 3-axis accelerometer, gyroscope, and compass [26].

A cell modem provides internet access to the vehicle and allows for the GNSS system to wirelessly receive position corrections. An E-Stop receiver is mounted on the outside of the vehicle and allows for us to remotely pause the system if needed.

The hardware mounted on the Husky platform can be seen in Figure 2.

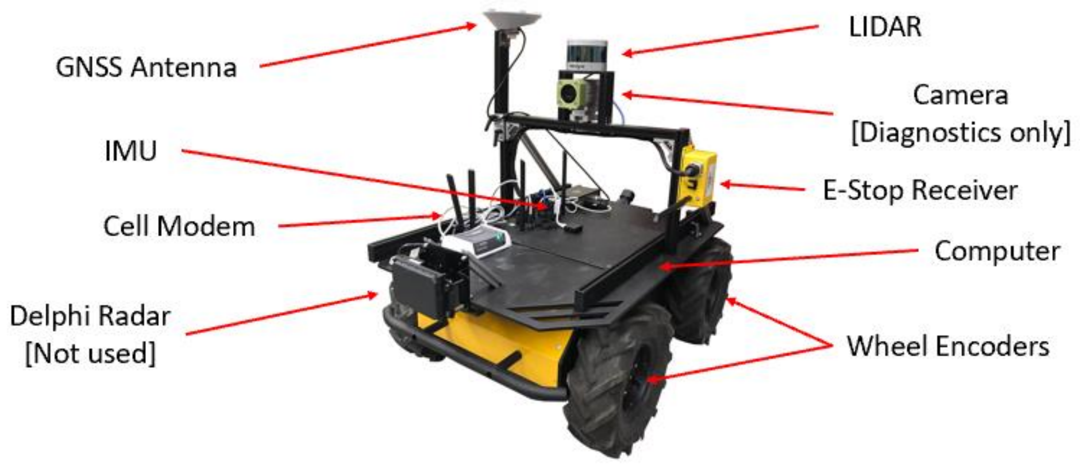

Figure 2 – The hardware and sensors mounted on the Husky platform that are used for this research.

The Husky UGV is a differential drive platform, meaning the wheels on the right side of the vehicle rotate in the same direction and at the same velocity as each other and the same can be stated about the wheels on the left side. The set of wheels on one side spin independently of the wheels on the other side of the vehicle. Differential drive kinematics allows the vehicle to rotate about its center without having to travel forward or backward. This varies from Ackerman steering in which one axle of the vehicle allows the wheels to rotate and on the other axle the wheels are fixed, which is common in many vehicles today [27].

The vehicle computer runs the Robot Operating System (ROS) for fusion of multiple sensor readings and facilitates implementation of custom control and software algorithms. ROS [28] is a "collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms". ROS is a powerful tool that allows researchers around the world to collaborate and build off previous work in an easy and efficient manner. Clearpath uses ROS to control all of its vehicles and supports its drivers through ROS. ROS allows us to fuse our sensors together (GNSS, LIDAR, IMU) in order to localize and create a labeled map, as well as send velocity commands to the motors while traveling.

3. Testing Environment

Since the goal of this research is to generate a labeled map of an urban sidewalk environment and autonomously localize and travel between two points, a neighborhood was desired that had many different characteristics that a vehicle might encounter in the urban world. A neighborhood near the University of Minnesota – Twin Cities campus was selected because it satisfied many of the desired characteristics which we desired for our research. A table of the desired characteristics can be seen in Table 1 and an aerial view of the location obtained from Google Maps can be seen in Figure 3. Images showing some unique aspects of the neighborhood can be seen in Figures 4, 5, and 6.

Characteristic	Present
Sidewalks with a boulevard of trees between it and the curb	**Yes**
Sidewalks with no boulevard between it and the curb	**Yes**
Sidewalks with curbcuts or no curbcuts at the intersections	**Yes**
Sidewalks that include driveways	**Yes**
Sidewalk curbcuts that are perpendicular to the street	**Yes**
Sidewalks located on non—orthogonal streets	**Yes**
Sidewalk curbcuts at the intersection corners that are wide enough for entry into either of the two adjacent streets	**Yes**
Sidewalks with objects located in the sidewalk, for example parking meters, trees, bike racks, which are permanent	**Yes**, street signs
Poor quality sidewalks	**No**
Sidewalks with nonpermanent objects, such as bicycles obstructing forward motion. This may also include a barrier due to a construction zone ahead or a pile of snow during the winter.	**No**
Sidewalks that just end with grass ahead	**No**
Sidewalks with grass or bushes on either side or grass/bushes only on one side	**Yes**
Sidewalks include walkways or stairs to the front door of adjacent homes	**Yes**
Sidewalks extend directly from a building (e.g. a store front) to the curb	**No**
Alley with no sidewalks	**Yes**

Table 1 – List of desired characteristics wanted in a neighborhood for conducting our research. This list was compiled to include most scenarios that might be present in urban neighborhoods.

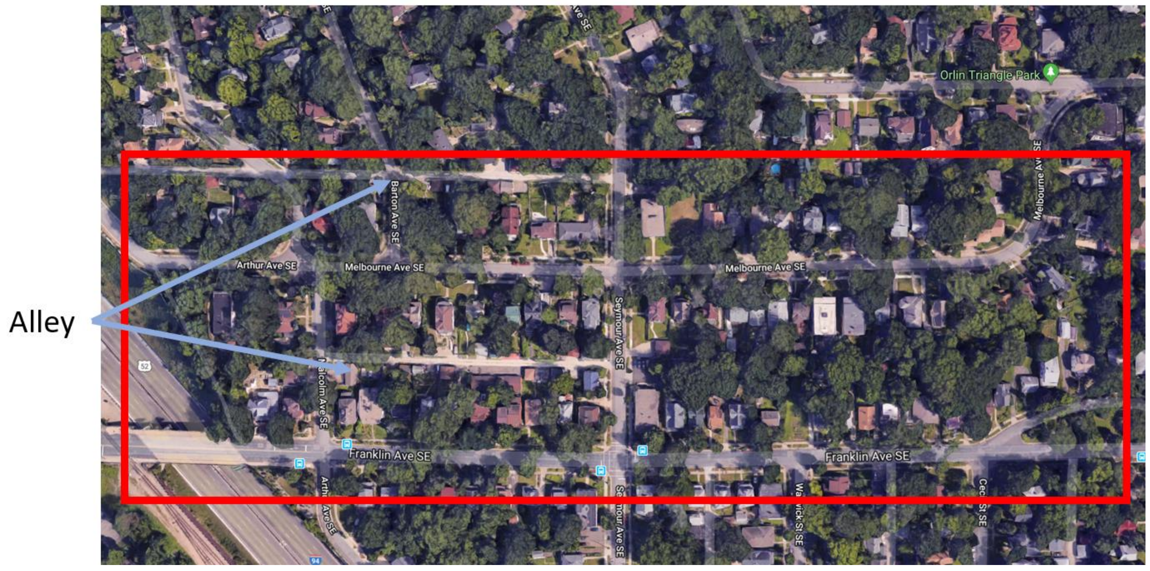

Figure 3 – Bird's-eye view from Google Maps of test neighborhood. The bounds of the test area are indicated in red and the alley ways are indicated by blue arrows.

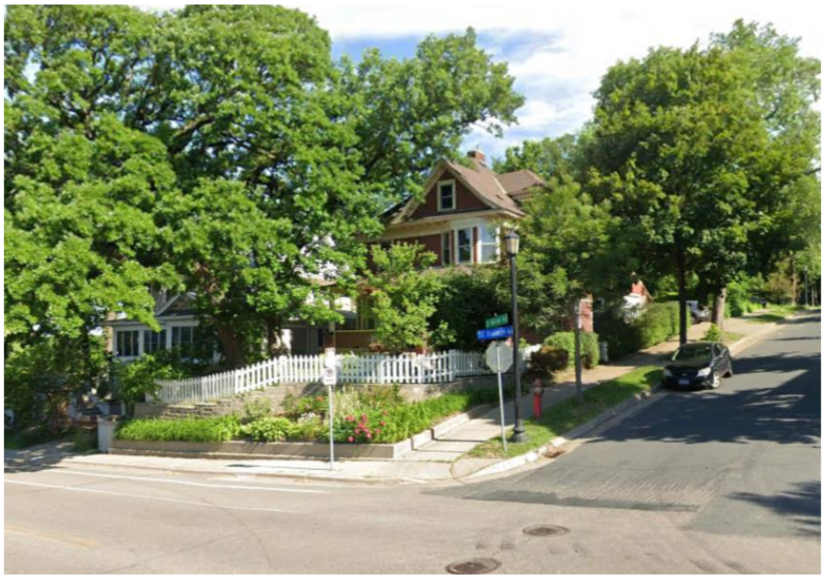

Figure 4 –Intersection of a block with signs placed in the sidewalk path and a grass median shown on the right.

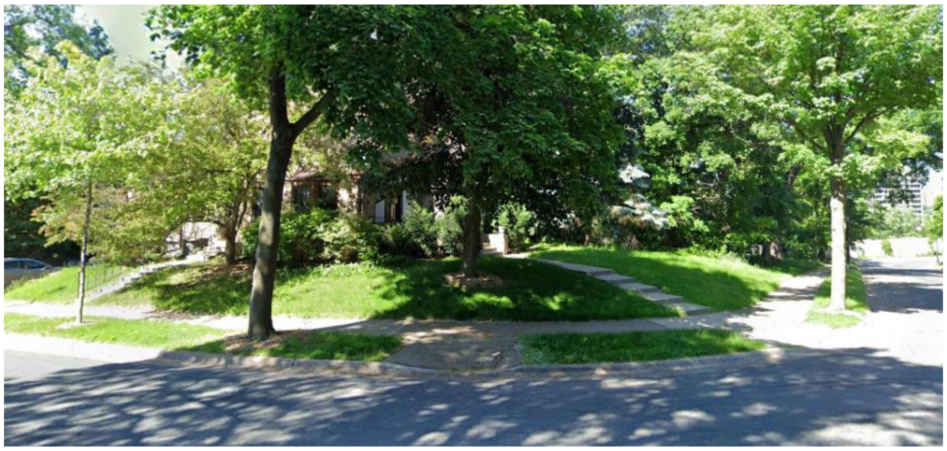

Figure 5- Rounded sidewalk corner in the testing environment.

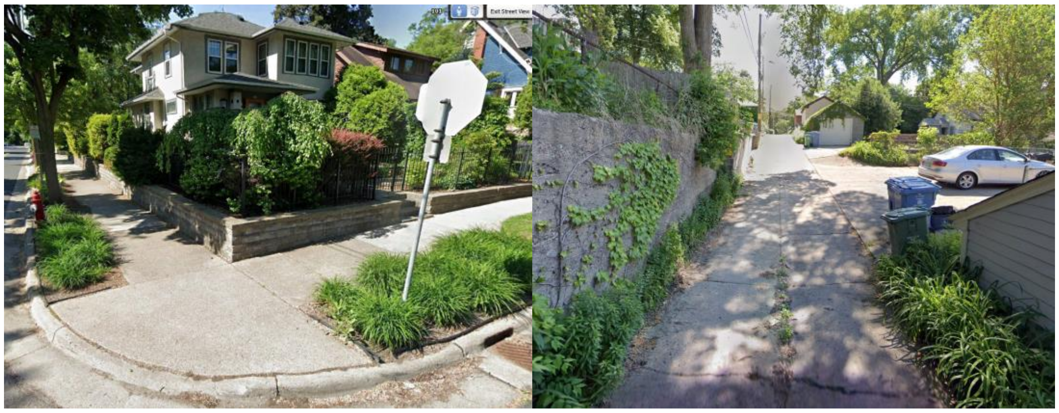

Figure 6- Sidewalk with retaining wall on one side (left). Alley in testing environment (right).

Since a major focus of this work is the ability to travel in snow covered terrain, images showing the testing environment in the winter are shown below in Figure 7 and Figure 8. Although the vehicle's ability to travel and follow a path was tested in snow, it is important to note that no mapping was conducted under snowy conditions. All mapping was done under good weather conditions when no snow is on the ground.

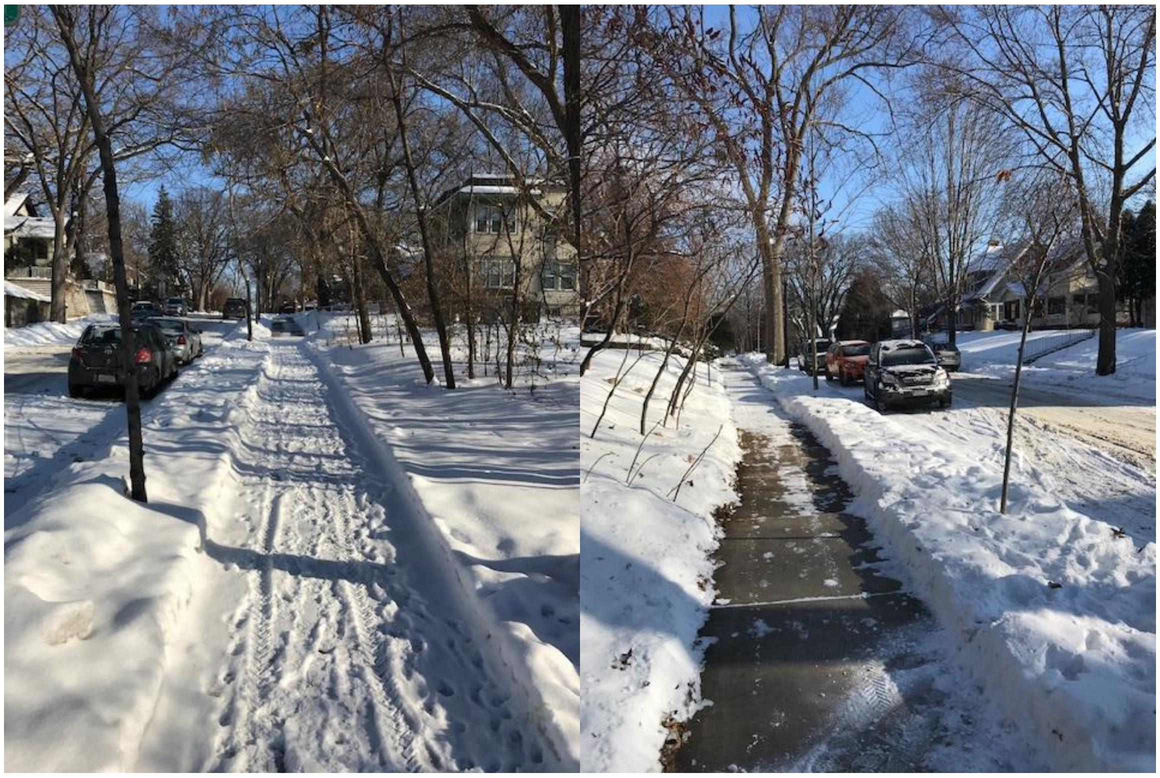
Figure 7- Sidewalks in the testing environment during winter. Complete snow-covered sidewalk (left). Mix of concrete and snow (right).

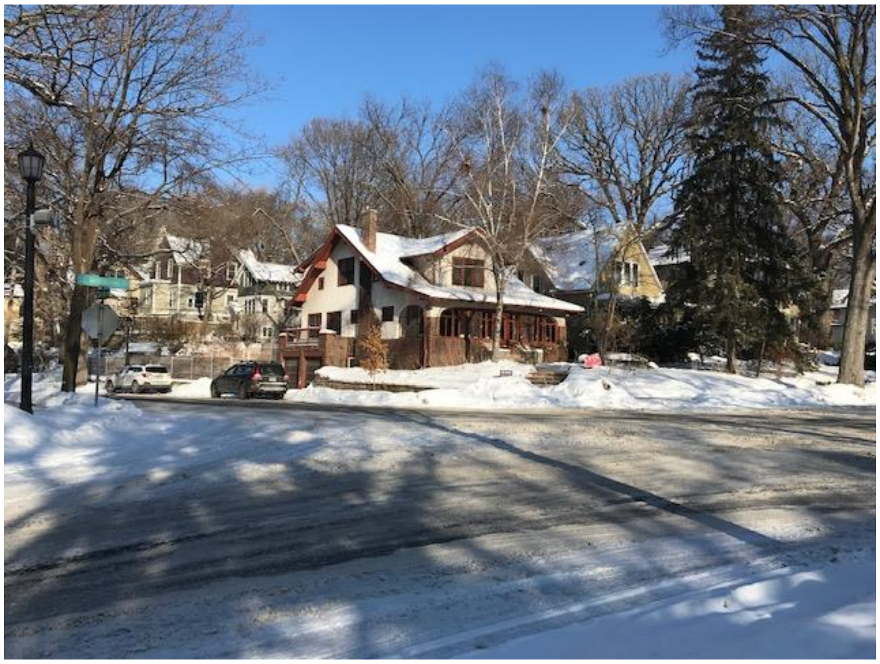
Figure 8- Intersection in the testing environment in the winter.

4. **Localization Technique**

In this paper, for simplicity, we assume a planar environment. This means that the vehicles pose can be satisfied by knowing its X, Y position as well as its yaw (heading). In order to autonomously navigate between two points, the vehicle must be able to accurately localize in the environment. The localization scheme proposed in this research uses an Extended Kalman Filter to fuse together high accuracy GNSS, LIDAR Odometry, and IMU data. The GNSS system allows for accurate global localization when there is good enough reception, such as in road intersections and clearings in the tree canopy. Global localization allows the vehicle to know where it is in the world, in terms of latitude and longitude. This is important when plotting a path between two global points in the world as it allows us to define positions in terms of latitude and longitude.

LIDAR Odometry (LeGO-LOAM) provides dead-reckoning localization in low quality GNSS areas. These areas include locations under tree canopy, indoors, and areas surrounded by buildings. The LIDAR odometry provides localization relative to the most recent accurate GNSS position. The IMU provides an initial yaw measurement of the vehicle upon startup.

The extended Kalman filter (EKF) used is provided by the ROS package *robot_localization*. *Robot_localization* is an open source collection of state estimation nodes which implement nonlinear state estimation for vehicles. The EKF [29] uses an "omnidirectional motion model to project the state forward in time, and corrects that projected estimate using perceived sensor data". The inputs and outputs of the EKF, which are constrained to a planar environment, can be seen in Figure 9.

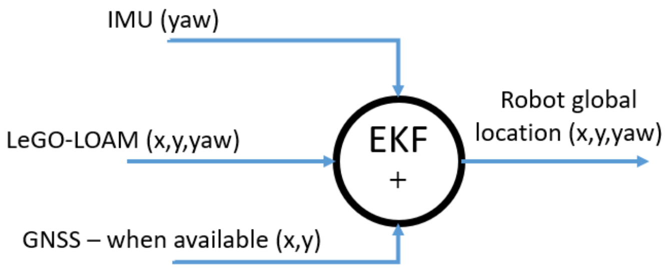

Figure 9- EKF inputs and outputs.

The EKF process provided is a nonlinear dynamic system, with

$$x_k = f(x_{k-1}) + w_{k-1}$$

Where x_k is the robot's state at time k, f is the nonlinear state transition function, and w_{k-1} is the normally distributed process noise. Without any assumptions, the state vector x is 12-dimensional which consists of the 3D pose and orientation of the vehicle, as well as the respective velocities of the pose and orientation. It is important to note that in this work we assume a planar environment, which reduces the number of dimensions down to six. When the EKF receives a measurement, it is in the form

$$z_k = h(x_k) + v_k$$

Where z_k is the input measurement at time k, h is the nonlinear sensor model mapping the state into measurement space, and v_k is the measurement noise. The initial step of the EKF is called a prediction step that projects the current state of the vehicle and its respective error covariance forward in time:

$$\hat{x}_k = f(x_{k-1})$$

$$\hat{P}_k = FP_{k-1}F^T + Q$$

Where f is a 3D kinematic model derived from Newtonian mechanics, P is the estimate error covariance, F is the Jacobian of f, and Q is the process noise covariance. Finally, a correction step is carried out to update the state vector and covariance matrix [30].

4.1 GNSS Specifics

ROS converts the latitude and longitude coordinates from the receiver to X and Y coordinates in the vehicle frame via a provided navsat_transform node in the ROS navigation stack [31]. The navsat_transform node allows for quick conversions from GNSS coordinates to cartesian coordinates in the vehicle frame. It is assumed that the vehicle starts in a high accuracy GNSS location to ensure the vehicle gets a good initial global position estimate. The first high accuracy GNSS location the vehicle receives is set as the (0,0) XY position of the vehicle. Any subsequent high accuracy GNSS receptions positions are considered relative to this initial start point. A covariance matrix is calculated automatically in ROS based on the quality of GNSS reception. Before feeding the GNSS data into the EKF, coordinates above a certain covariance threshold are filtered out. This stops low accuracy GNSS positions from negatively affecting the localization in the EKF. Tests were performed with only low accuracy GNSS and the results will be discussed in Section 6.2.

4.2 LIDAR Odometry Specifics

LIDAR odometry provides localization estimates relative to a specific pose. That is, ΔX, ΔY, and Δyaw are estimated from the starting pose of the vehicle given by the GNSS and IMU. For simplicity, a static covariance of 0.01 is set for each pose estimation for LIDAR Odometry before it is fed into the EKF. The LIDAR odometry in an urban environment can travel 500 meters before the drift reaches above 0.5 meters. LIDAR odometry needs many distinct features for an accurate localization estimate, the fewer the features, the faster the drift will accumulate.

4.3 IMU Specifics

The heading of the vehicle is needed in order to fully define its pose in a planar environment. Since the initial pose X and Y is given from the GNSS, an IMU with a magnetometer is used get the initial yaw of the vehicle. LIDAR Odometry is then used to estimate the Δyaw from the initial start position.

5. Mapping and Navigation Procedure

This research proposes a multistep approach to mapping and classifying the environment that the vehicle is traveling in. These steps include data collection, automated post-processing of data, and autonomous navigation between points while traveling only on sidewalks and intersections. It is important to note that steps 5.1 and 5.2, data collection and post-processing, are only done once while step 5.3, autonomous navigation, can be done repeatedly without have to redo the previous steps. An overview of the process flow can be seen below in Figure 10. The post-processing step classifies LIDAR data into regions such as grass, sidewalks, roads, and obstacles and generates a labeled segmented map.

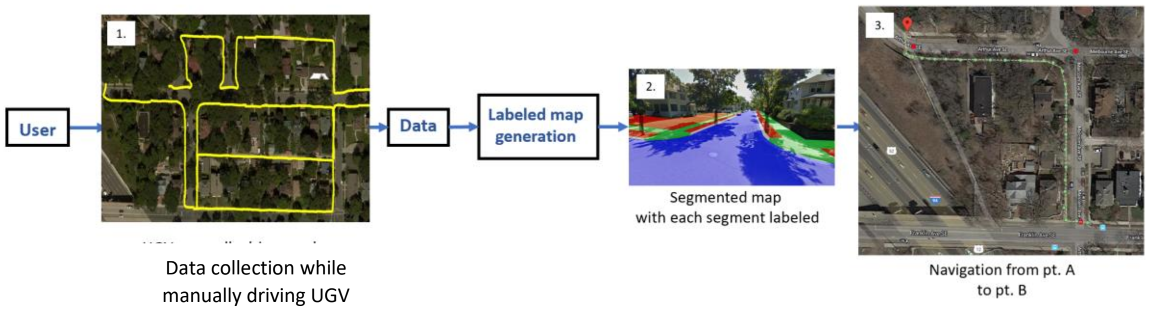
User
1.
Data collection while manually driving UGV
Data
Labeled map generation
2.
Segmented map with each segment labeled
3.
Navigation from pt. A to pt. B

Figure 10- Process flow for environment mapping and navigation.

The details for these steps are described below.

5.1 Data Collection While Manually Steering Vehicle

The first step towards generating a classified map of the region of interest is collecting the data that will be processed to generate such a map. The user steers the vehicle manually via commands from a joystick controller or arrows on a keyboard. The user must steer the robot along all the sidewalks and alleys that they want the vehicle to be able to travel on autonomously in the future. It is important to drive through all the potential destinations to which one may want the vehicle to navigate in the future, such as relevant building entrances or bus stops. This data collection is also done when there is no snow on the ground. This is so we can better segment the environment in section 5.2, since snow can easily cover many distinctive features of objects.

The data collected in this step comes from the LIDAR, GNSS, and IMU. The LIDAR data is three-dimensional information about the environment around the vehicle. The field-of-view (FOV) of the LIDAR can be seen in Figure 11 and an example of the 3-D point-cloud returned by the LIDAR can be seen in Figure 12.

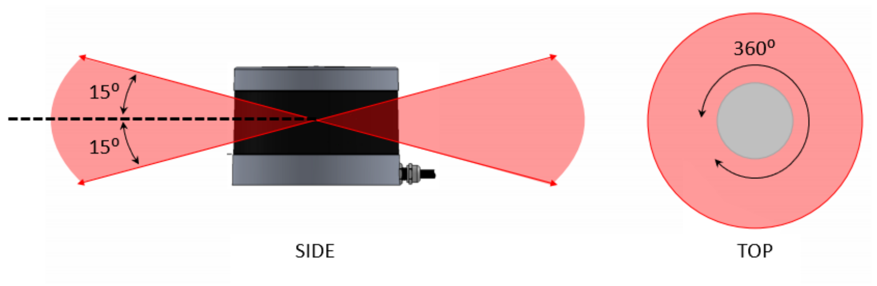

Figure 11- Side and top views of the field-of-view (FOV) of the VLP-16 LIDAR. Objects within in the FOV are returned from the LIDAR in the form of a 3-D pointcloud.

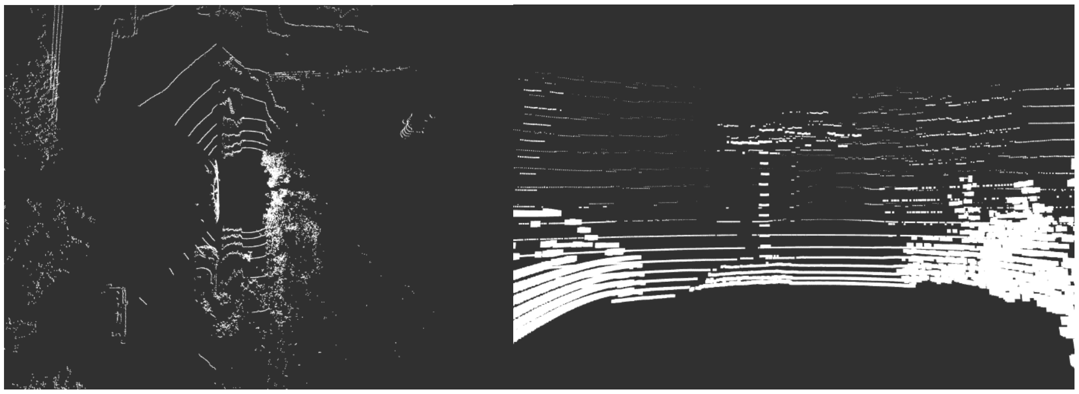

Figure 12- Top down view (left) and side view (right) of one complete LIDAR return scan. This scan consists of approximately 30,000 3-D points of the environment around the vehicle.

The GNSS data is also collected in order to identify where GNSS high-accuracy reception is available in the neighborhood. This will be important further down the pipeline when generating a path. The IMU data allows for the initial yaw pose estimation. During the mapping procedure, the vehicle pose is identified using the localization scheme described in section 4.

There has been a fair amount of research done regarding autonomous exploration of unknown environments. Wang *et al* [32] propose a method of autonomously exploring an unknown environment with the intent of generating an efficient feature-based map. Li *et al* [33] developed a system using ROS for autonomous wheel chair exploration in an indoor environment. Despite these achievements, there is a lack of research in autonomous exploration in an urban environment that adheres to socially acceptable means of exploration, such as only using sidewalks and avoiding driving over grass or other people's lawns. In addition to this, autonomously exploring a neighborhood such as the one proposed in this paper would take much longer if it was explored without the assistance of a user steering the vehicle. As a result, this work does not attempt to use autonomous exploration and instead we opt for manually steering the vehicle to collect the data.

5.2 Post-Processing of Data

The second step of the procedure is to automatically analyze the data and create a segmented and labeled map of the area of interest from the LIDAR pointcloud. The LIDAR outputs approximately 30,000 points per frame and runs at 10 Hz which results in 300,000 3-D points per second. Each of these points represent the distance to a point on the surface of objects in the environment that the vehicle is in. The LIDAR data points are sent through a custom algorithm that allows for the classification of the points into sidewalks, roads, grass, curbs and curb-cuts, and obstacles. Obstacles are defined as trees, bike racks, stairs, etc. We broadly classify these types of objects together because when traveling between two points, the vehicle just needs to know enough to avoid these objects and doesn't need to know explicitly the type of object it is. For example, knowing the distinction between grass and sidewalk may be more important than knowing the difference between a lamp-post and a tree. For our application, it is important for the robot to drive on sidewalks and not on grass so distinguishing between grass and sidewalk is important. However, both the tree and lamp-post represent an object that must be avoided; distinguishing between a tree and lamp-post is not as important as their shape. An outline of how the vehicle classifies the environment can be seen in Algorithm 1.

In Algorithm 1 described below, the slope is taken between two column-wise points. In terms of the LIDAR pointcloud, two-column wise points will be in separate channels of the lidar in the vertical direction. The grade of the road, in reference to the road curvature, is defined as the magnitude of the slope as it rises and falls along its width. Objects such as stairs and retaining walls will fall into the category of obstacle in this algorithm.

Algorithm 1: LIDAR Classification into grass, road, sidewalk, and obstacles

```
Result: Segmentation and Classification of Environment using only LiDAR
Loop through all LIDAR input points;
while valid point received do
    Calculate slope between two column-wise (vertical) points;
    $\Delta X = X1 - X0$
    $\Delta Y = Y1 - Y0$
    $\Delta Z = Z1 - Z0$
    $\theta = atan2(\Delta Z, \sqrt{\Delta X^2 + \Delta Y^2})$
    if $\theta < \theta_{thresh}$ then
        Calculate std. dev. (roughness) of n consecutive LIDAR points;
        s = $\sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \overline{x})^2}$
        if $s > s_{thresh}$ then
            The LIDAR point is Grass;
        else
            Point may be road or sidewalk;
            Calculate curvature of consecutive points;
            $k = |\frac{dT}{dS}|$
            Calculate presence of curbs (sharp elevation changes);
            $curb = \Delta Z > z_{thresh}$
            if curvature meets expected grade AND presence of curb then
                The LIDAR point is Road;
            end
            if no curvature then
                The LIDAR point is Sidewalk;
            end
        end
    else
        The LIDAR point is an obstacle;
    end
end
```

5.2.1 Sidewalk Segmentation

Determining where the sidewalk is in the environment is the most crucial aspect of this research. It is needed to give the vehicle the ability to travel using the sidewalk and not on the grass. An important characteristic about sidewalks is that a quality sidewalk (meaning without major holes, cracks, or tree roots) is smooth and flat. This means the LIDAR points falling on the sidewalk should return data that has similar elevation with little variation between consecutive points. Sidewalks are also typically surrounded by grass, curbs, walls, or vegetation which cause a discrete jump in the LIDAR scan. Taking these characteristics into consideration, the sidewalk can be extracted from the raw LIDAR scan. An example of the sidewalk LIDAR classification result can be seen in Figure 13.

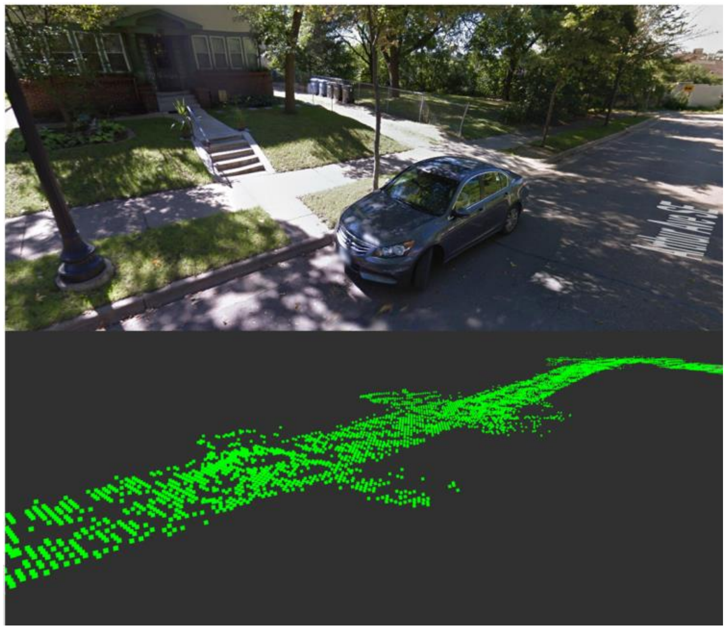

Figure 13- Image of a sidewalk (top) and the extracted sidewalk (green) and mapped using LIDAR (bottom image). The shadows in the image are shown to highlight the difficulty of using vision in neighborhoods with tree canopy.

5.2.2 Road Segmentation

Roads in urban environments are similar to sidewalks in that they are typically smooth. However, to differentiate them from sidewalks, roads also have a distinct cross slope to them [34]. The cross slope is designed into roads so that the highest point of the road is in the center which causes water to drain from the road surface to the street gutters. Urban roads are also often surrounded by curbs, meaning the LIDAR scan (when the robot is on a sidewalk) will see an elevation jump from the sidewalk down to the road, the road back up to the sidewalk, or both. This can be seen in Figure 14.

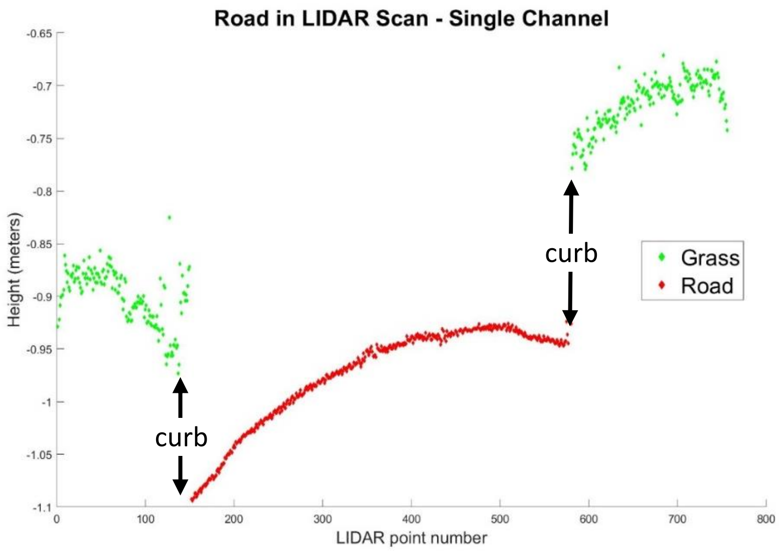

Figure 14- Example of LIDAR data from a cross-section of road in the testing environment. The distinct curvature of the road LIDAR points can be seen (red) as well as the presence of the curbs as indicated by the discrete height jumps in the LIDAR scan.

5.2.3 Grass Segmentation

Unlike sidewalks and roads, LIDAR points that fall on grass are noisy. The blades of grass create a high standard deviation of consecutive points that allow for easy classification in comparison to roads and sidewalks. An example of the road LIDAR classification can be seen in Figure 15.

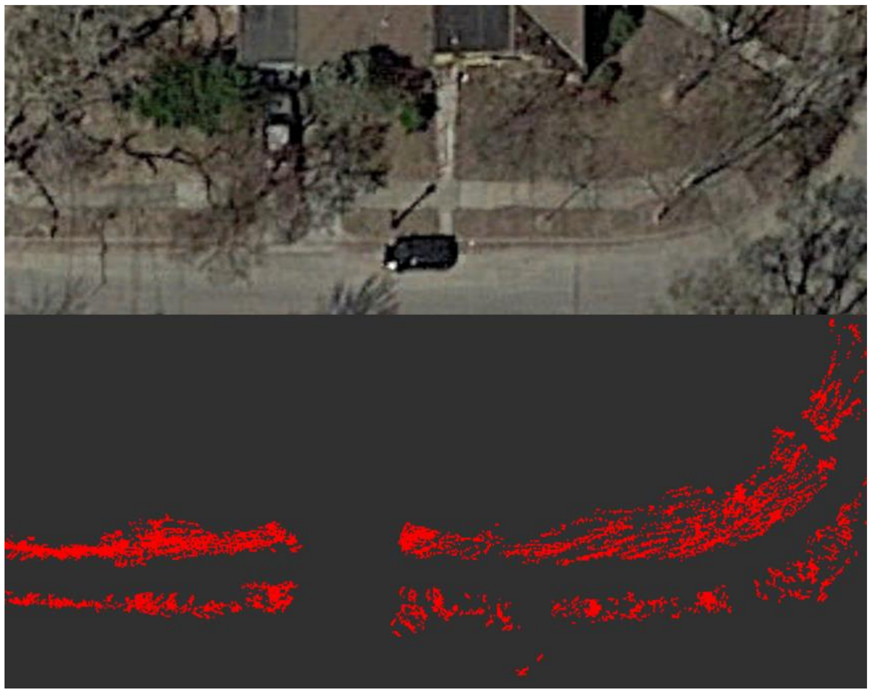

Figure 15- Image of grass (top) and the extracted grass (red) and mapped using automated processing of the LIDAR pointcloud (bottom).

It is important to note that the top section of grass in Figure 15 shows a thinner strip of grass than is apparent in the aerial image. This is because there is a steep slope in these yards that is not shown in the aerial image, as a result the algorithm does not classify the LIDAR points as grass when the slope is too steep.

5.2.4 Curb and Curb-cut Segmentation

Curbs are associated with the break in elevation between the road and sidewalk or grass median. In a typical urban neighborhood, sidewalks are positioned higher in elevation than roads. There are two common scenarios, the first one is where there is a sidewalk, a curb, then a road. The second being where there is sidewalk then a grass median, then the curb, then the road. Curb cuts are detected when the sidewalk merges with the road.

5.2.5 Sidewalk GNSS Centerline Generation

For the robot to travel only on sidewalks, the centerlines of the sidewalks must be known in order to generate a path using the sidewalks. More specifically, the GNSS coordinates of the sidewalk centerline are desired. The first step to find the sidewalk centerline is to extract the sidewalk directly in front of the robot from the first (or the lowest in elevation) channel of the LIDAR, as shown in Figure 16.

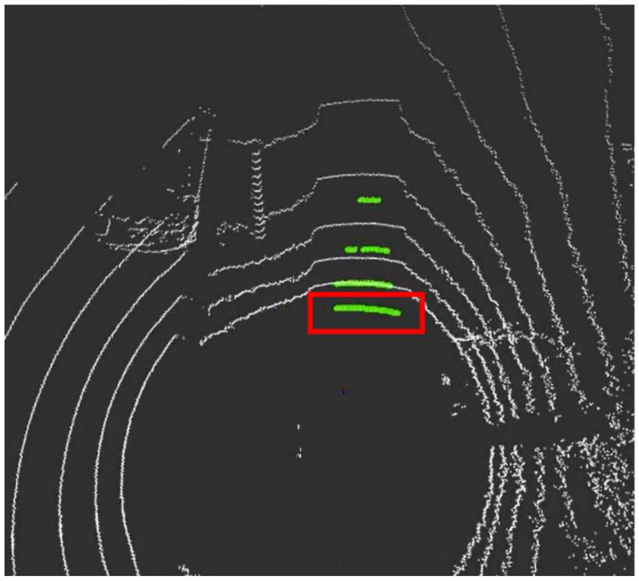

Figure 16- Top down view of a LIDAR scan and section of the scan used for the centerline generation shown in red. The noisy LIDAR points on the bottom right side of the figure is due to the LIDAR sensing vegetation in a garden.

The midpoint of this sidewalk scan is archived, and a new centerline point is found as the robot moves forward. Every 1 meter, all the centerline points that were archived for the past 1 meter are all averaged and the X and Y points (which are the output of the EKF), which are relative to the starting pose, are saved as the centerline midpoints for that section of sidewalk. This midpoint is then converted using the ROS navigation stack to a latitude and longitude coordinate. A visual of this process can be seen in Figure 17.

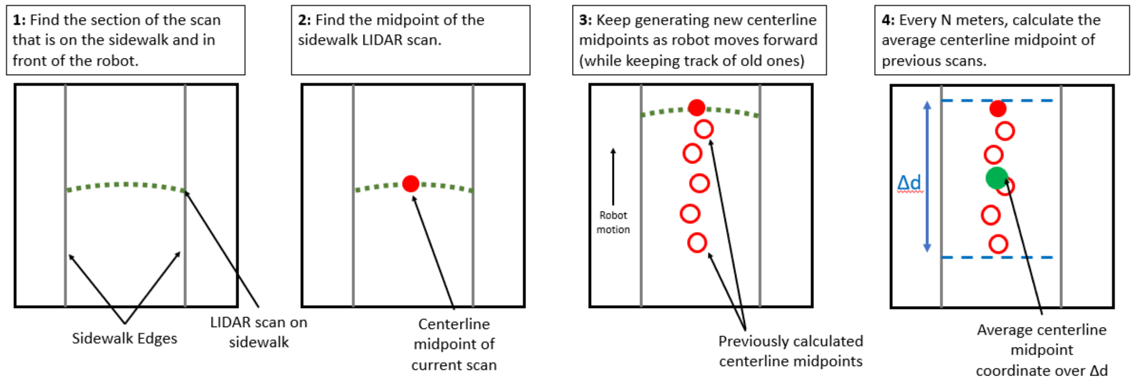
1: Find the section of the scan that is on the sidewalk and in front of the robot.
Sidewalk Edges
LIDAR scan on sidewalk
2: Find the midpoint of the sidewalk LIDAR scan.
Centerline midpoint of current scan
3: Keep generating new centerline midpoints as robot moves forward (while keeping track of old ones)
Robot motion
Previously calculated centerline midpoints
4: Every N meters, calculate the average centerline midpoint of previous scans.
Δd
Average centerline midpoint coordinate over Δd

Figure 17- Process to calculate the sidewalk centerlines.

5.2.6 Locations with High Accuracy GNSS

Areas with good GNSS reception in the neighborhood are recorded and mapped during the initial data collection. This is important information for several reasons. If the vehicle localization using LIDAR odometry were to drift too much while it is traveling, one can divert to the nearest area with good GNSS reception in order to zero it's error. During path planning, a path can also be generated so that if the vehicle knows it will be traveling for too long in an area relying solely on LIDAR Odometry and an unacceptable amount of drift will likely occur, the path can be modified to navigate to a known area with good GNSS reception to 'check-in' and reduce the localization error, and then continue on the rest of the path to the final destination. However, this functionality is not used in this research because in the testing environment all paths currently generated happen to cross through a high accuracy GNSS zone, such as those located at identified intersections.

5.3 Path Planning and Navigation

Now that the environment has been mapped, the system knows where there is grass, sidewalks, roads, curb-cuts, and obstacles. This information is used to create a path from pt. A to pt. B using only sidewalks and curb-cuts to cross intersections. The sidewalk centerline points and Google Maps API [35] are then used to plan the path in an efficient way.

The Google Maps API is used to plot an initial GNSS coordinate path between the origin and destination positions. An advantage of using the Google Maps API is the capability to use landmarks such as street names or addresses directly instead of knowing exact coordinates of the locations to which you want the vehicle to travel. However, Google Maps API will plot a GNSS 'breadcrumb' path that lies on the road centerlines and not on the sidewalks. A visual of an example path plotted via Google Maps can be seen in Figure 18.

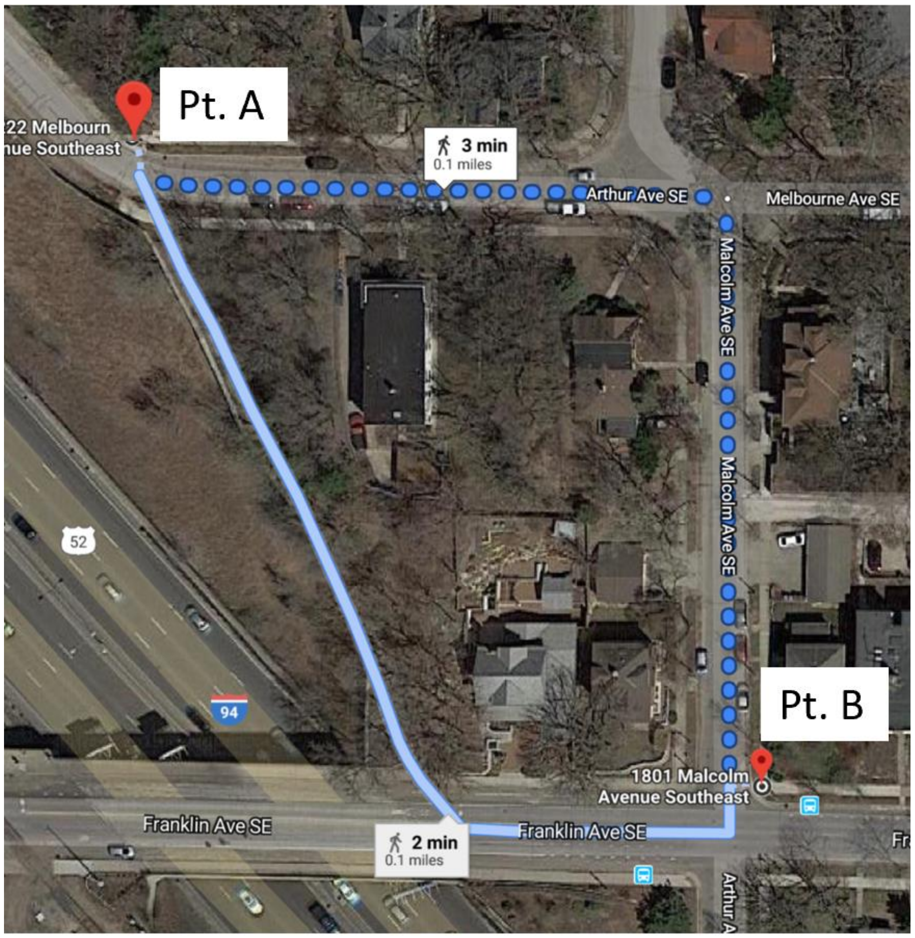

Figure 18- Example of 2 Google Maps path alternatives between origin and destination points.

The road centerline GNSS points can be modified to handle sidewalks and can be shifted over to the appropriate sidewalk centerlines. To apply corrections to the initial path, generated by the

Google Maps API, the path is overlaid onto the previously generated segmented map. The GNSS coordinates from the road are shifted to the sidewalk centerlines (Section 5.2.5) and curbcuts. Algorithm 2 (described below) will find the closest sidewalk / curb-cut entry / departure points to the initial path determined with the road centerlines.

```
Algorithm 2: Sidewalk Path Generation
  Result: Path from pt. A to pt. B using only sidewalks and intersections
  Get initial path between points from Google Maps API;
  while All points along path not refined do
      Overlay Google Maps path onto segmented map;
      Plot straight line between pt. A and pt. B;
      Calculate closest distance from center-line to all sidewalk points ;
      Calculate sum of distances for all strings of sidewalk points;
      if Sidewalk center-line GNSS points closer to line then
          Favorable sidewalk GNSS points for path;
      else
          Reject sidewalk GNSS point for path;
      end
      Correct and shift Google Maps GNSS coordinates from road to sidewalks ;
  end
```

Given a destination coordinate, the vehicle will likely have several different valid path options to take to travel to the destination. The goal of our path planner is to take the shortest route possible. In order to reduce the number of paths down to one, a straight line is projected from pt. A to pt. B as seen in

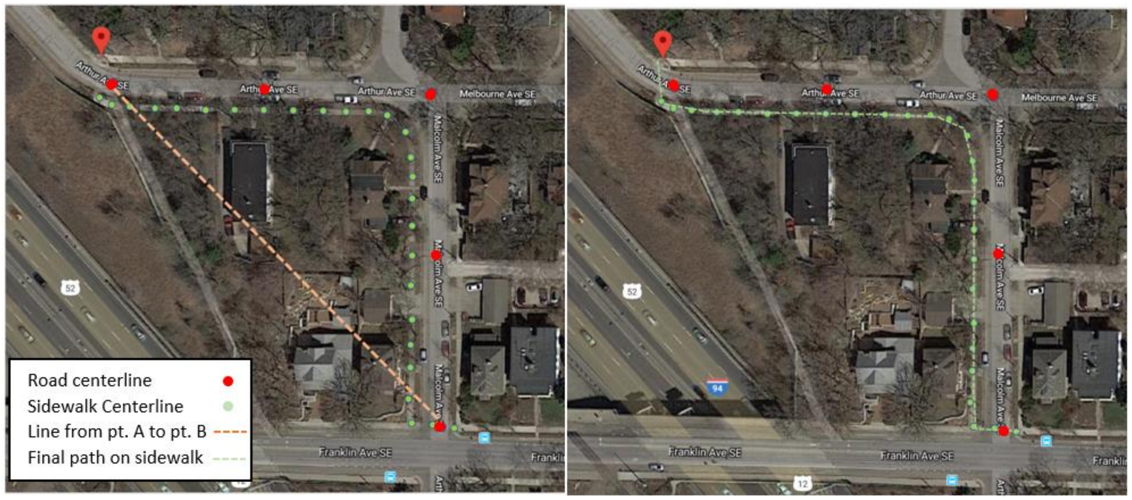

Figure *19*. This line helps us pick one side of the street to favor over the other (if needed). The distances between all the possible sidewalk centerline nodes and the line drawn between pt. A and pt. B are calculated and the sum of these distances for the different path options are found. The set of sidewalk centerlines that are closest to the line provide a single solution for the path between the points. This path is shown in

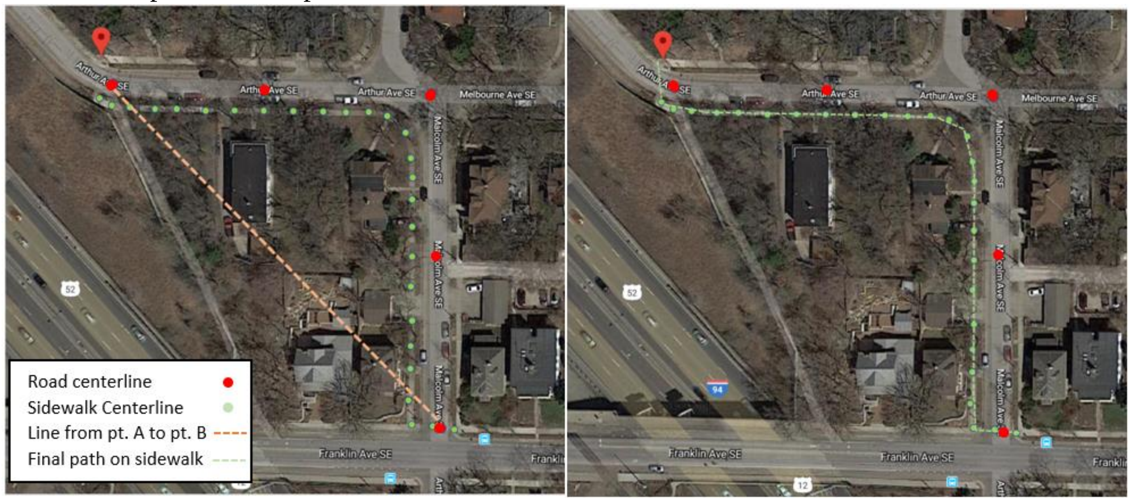

Figure *19*.

Figure 19- Line drawn between pt. A and pt. B assisting in path generation (left). The GNSS road centerlines (from Google Maps API) are shifted to the sidewalk (right).

Now that a path has been generated between two points using only sidewalks and curb-cuts at intersections, the vehicle has the ability to autonomously navigate the path using the ROS navigation stack. The ROS Navigation stack allows for path following by generating the velocity and heading needed for the robot in order to follow and stay on the path. The velocity and heading are then converted to motor commands to control the wheels on the Husky via a ROS package [36] provided by Clearpath Robotics. The ROS Navigation stack also provides functionalities such as obstacle avoidance using LIDAR scans to avoid people and pets while traveling.

While the robot is following the ‘breadcrumb’ path, the vehicle’s pose is identified using the localization scheme of combining high accuracy GNSS and LIDAR Odometry described earlier in section 4. In its current state, the path planner only allows the vehicle to cross the road at intersections, however, with minor modifications the vehicle could cross the road using other curbcuts such as driveways.

6. Results

In order to quantify the accuracy of the system and see how well it performed, the results are divided into to two sections: the mapping results and the localization results.

6.1 Evaluation of Segmented Map

The automatically generated segmented map of the environment, described in section 5, needs to be compared to a 'ground truth' to quantify the accuracy of the map. Since no map already exists, a 'ground truth' map was created by using an aerial orthorectified image of the neighborhood [37] and manually edited to add the proper labels. The raw orthorectified image can be seen in Figure 20 and the manually labeled image can be seen in Figure 21. The segmented map automatically generated in this project was converted from a 3D pointcloud down to a 2D image and then overlaid and compared to the orthorectified image to generate a measure of accuracy. The 2D pointcloud image and the image overlay can be seen in Figure 22 and Figure 23, respectively. Since sidewalks are the most important aspect of the segmentation, only accuracy results for the sidewalks will be discussed below.

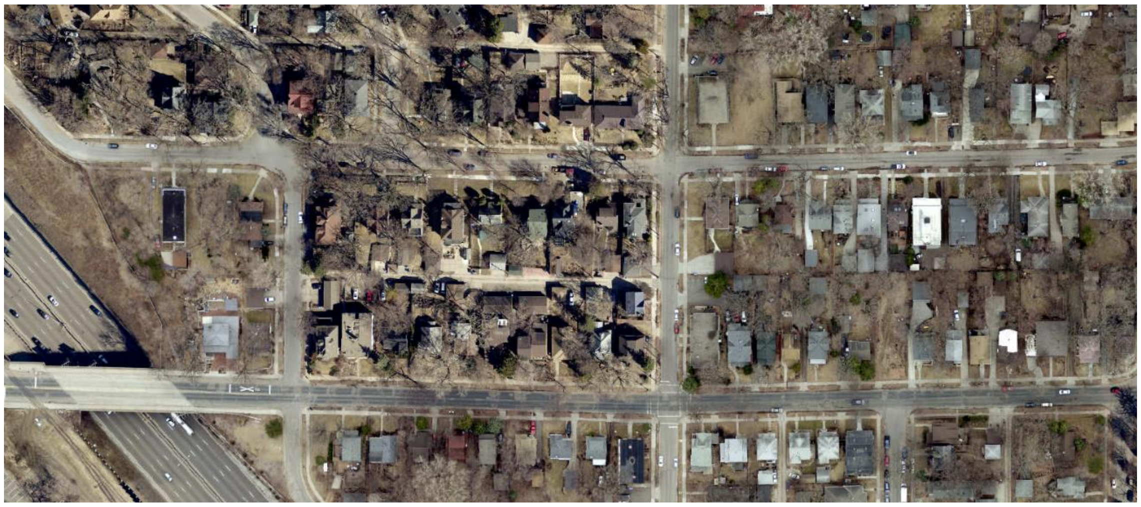
Figure 20- Orthorectified aerial image.

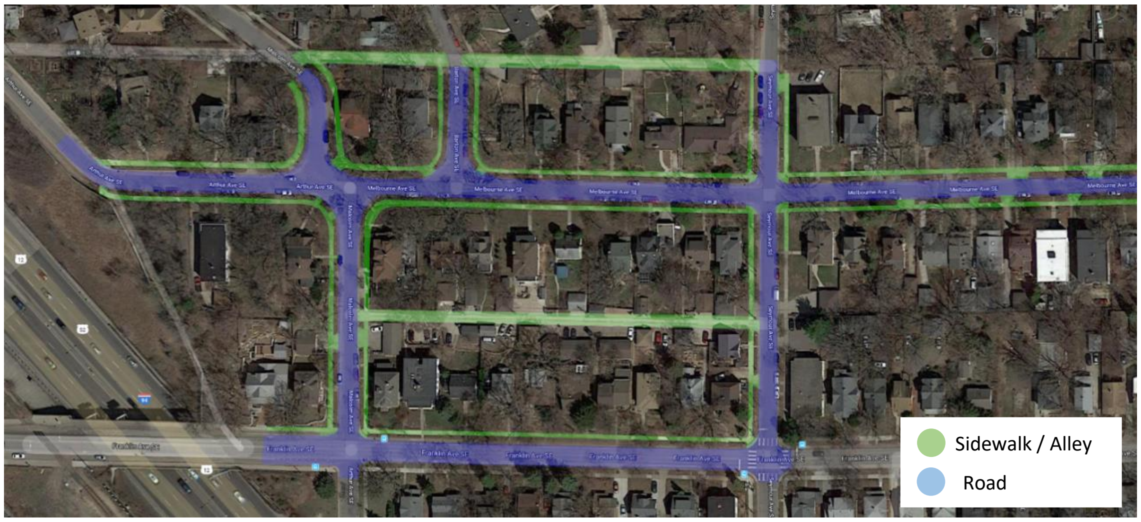

Figure 21- Manually labeled orthorectified image. Sidewalks, alleys, driveways, and curbcuts are all labeled sidewalk.

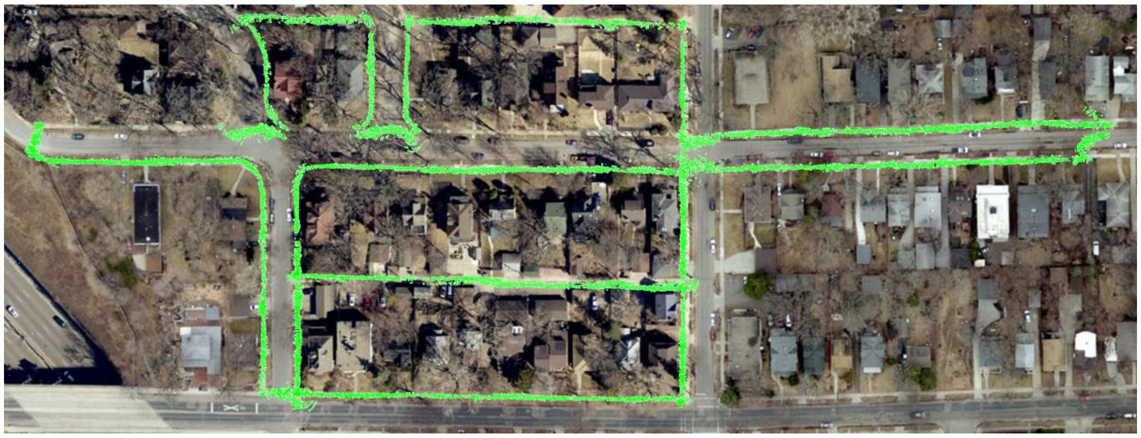

Figure 22- 2D pointcloud of segmented sidewalk overlaid on the orthorectified image.

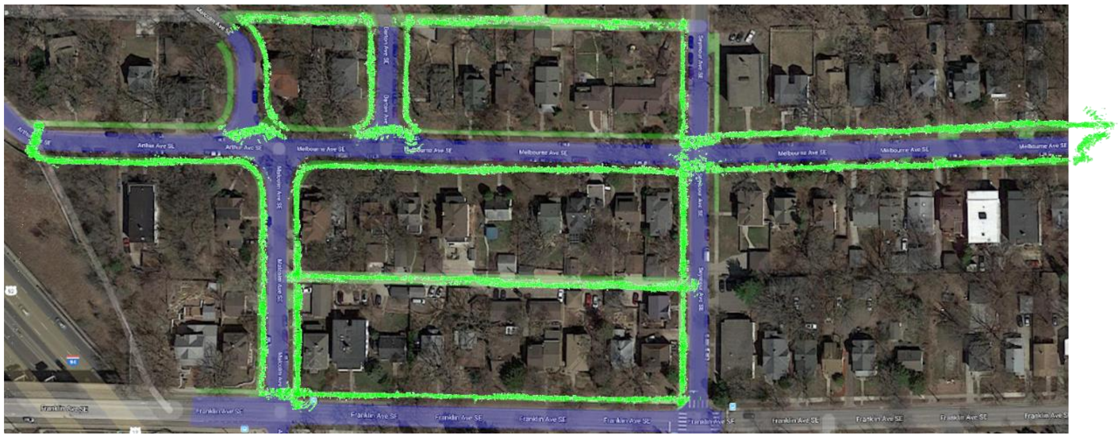

Figure 23- Sidewalk 2D pointcloud overlaid on manually labeled orthorectified image. These two images are stored as separate layers for evaluation purposes.

In Figure 22 and Figure 23, one can see that at intersections there are green pointcloud data indicating a sidewalk where there is actually a road. The reason for this is that while crossing the intersection, our requirements for a road (curb, drop in elevation) fail when the vehicle is actually on the road. As a result, the system is programmed to classify these points as a “sidewalk” even though that is not the case. However, this is not an issue because in most cases it is acceptable for the vehicle to consider intersections and crosswalks as “sidewalks”. As a result, in the sidewalk classification results below, these ‘sidewalk’ points are not considered for pixel comparison. It is important to note that at this time the vehicle is not programmed to actively take into account traffic when crossing an intersection and will attempt to cross the street regardless of cars and pedestrians that may be on the road. Instead, the user supervising pauses the vehicle when there is traffic on the road.

To evaluate the accuracy, the true positive (TP) and false positive (FP) for the sidewalk is calculated. This was done by a pixel comparison between the two images using MATLAB. The pixels that are classified as a sidewalk by the LIDAR are checked with the manually labeled image to see if it is a correct classification or a false classification. The results can be seen in Table 2.

Classification	TP	FP
Sidewalk	91.46%	8.54%

Table 2- Table showing accuracy of automatic classification of the sidewalks.

The sidewalk was 91.46% accurately classified and 8.54% falsely classified as sidewalk. Leaves and grass growing through the cracks in sidewalks accounted for much of the false classification.

6.2 Localization Results

To test the accuracy of the localization system, a 'ground truth' is needed. The high accuracy GNSS system described in section 4.1 was used to survey areas in the neighborhood where high accuracy results were available. This can only be done in GNSS areas with good reception, and since a majority of the area is covered in tree canopy or overshadowed by buildings, there were only 7 consistent locations in the neighborhood that were identified that could be accurately surveyed. These locations are indicated on the map in Figure 24.

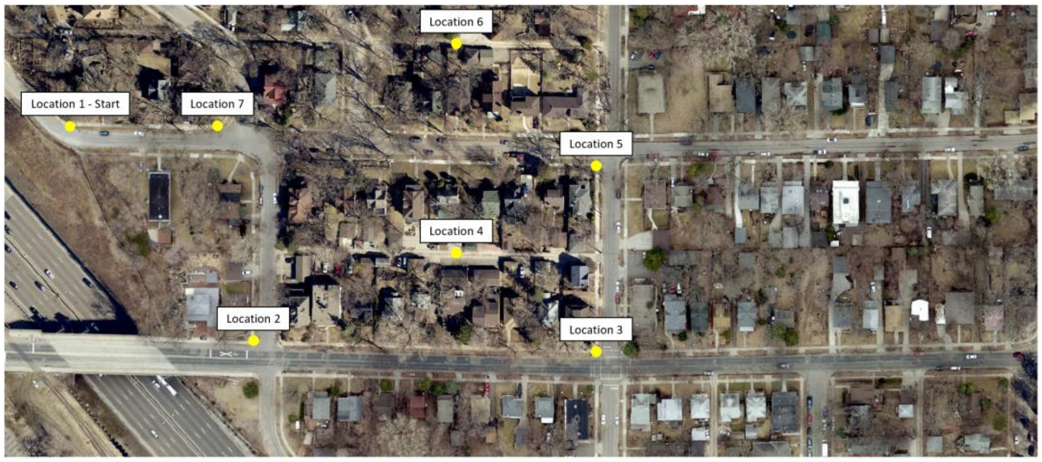

Figure 24- The areas of good reception GNSS (yellow) that were surveyed indicated on an image of the neighborhood.

To validate the accuracy in the areas with good GNSS reception, the vehicle was driven to each location with known high quality GNSS reception (clearings, intersections) and parked. GNSS coordinates were recorded over a one minute span. One minute was experimentally found to be enough time to get an accurate estimation of the true latitude and longitude. The GNSS coordinates were then converted to Universal Transverse Mercator (UTM) coordinates, which is in meters, and averaged over that period of time to generate a 'ground truth' latitude and longitude. The accuracy of the GNSS position at these locations was found by calculating the circular error probable (CEP). The CEP is a measure of the median error radius of all the location points recorded and allows us to quantify the quality of the GNSS signal at these locations. A scatter plot showing the results of the GNSS latitude and longitude readings over a one minute period and the CEP error circle for location 1 is shown in Figure 25.

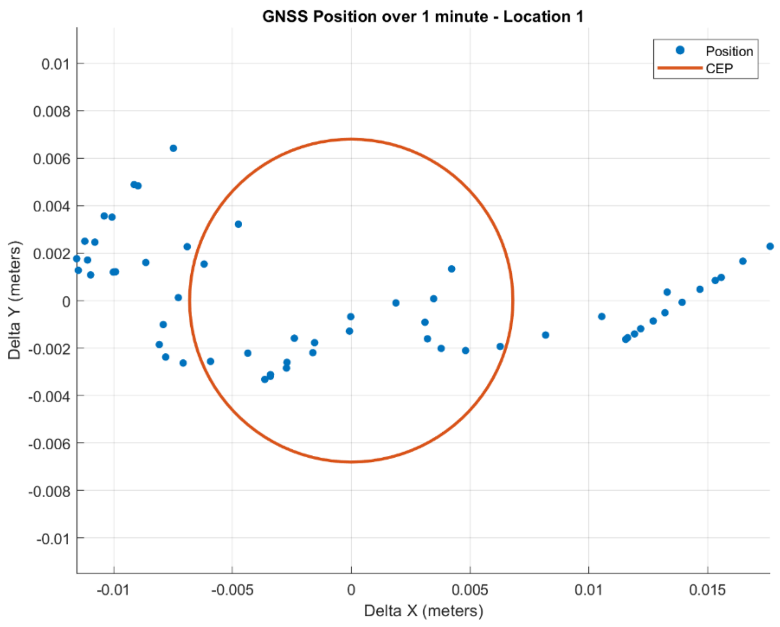

Figure 25- Plot of GNSS coordinates converted to Universal Transverse Mercator (UTM) coordinates in meters to show how the absolute position estimate varies over a one minute period of time in a GNSS zone with good RTK reception. The red circle shows the GNSS circle error probable (CEP) which is 0.0068 meters.

As the robot travels between two points and passes through these surveyed areas, we compare the estimated position of the vehicle from the localization scheme EKF to the true value as it passes through the surveyed area. The localization results were recorded for two runs. The first run was conducted during the summertime under clear sky conditions and no snow on the ground while the second run was conducted during winter under cloudy conditions with fresh snow cover on the ground.

A table showing the CEP GNSS quality values for all 7 locations as well as the total distance from the start (location 1) that the vehicle traveled as it reached the respective location and the recorded error for both the summer and winter runs can be seen in Table 3. The vehicle was given these 7 waypoints as consecutive destinations to which to navigate. The vehicle was also given 5 other intermediate waypoints to travel through, shown as red circles in Figure 26. These additional waypoints were needed in order to have the vehicle travel on most of the sidewalks in the neighborhood. The system automatically generated a path between the 7 high accuracy GNSS locations and 5 waypoints using the path generator described in Section 5.3 and autonomously traveled starting at location 1 through all subsequent locations until location 7 was reached. This explains why the path seems convoluted. Our objective was to travel between all these locations and create a longer path within the neighborhood for testing and evaluation. Once a location is reached, the error is calculated by finding the distance between where the vehicle thinks it is in the world, via the localization scheme described in Section 4, and the true location that was surveyed. The vehicle then travels to the next subsequent location. For example, once the vehicle reaches location 2, the error is calculated between actual and expected coordinates, then the vehicle continues to location 3 (adding on to the previous path between location 1 to location 2). This is continued until location 7 is reached. The paths taken between several example location can be seen in Figure 26.

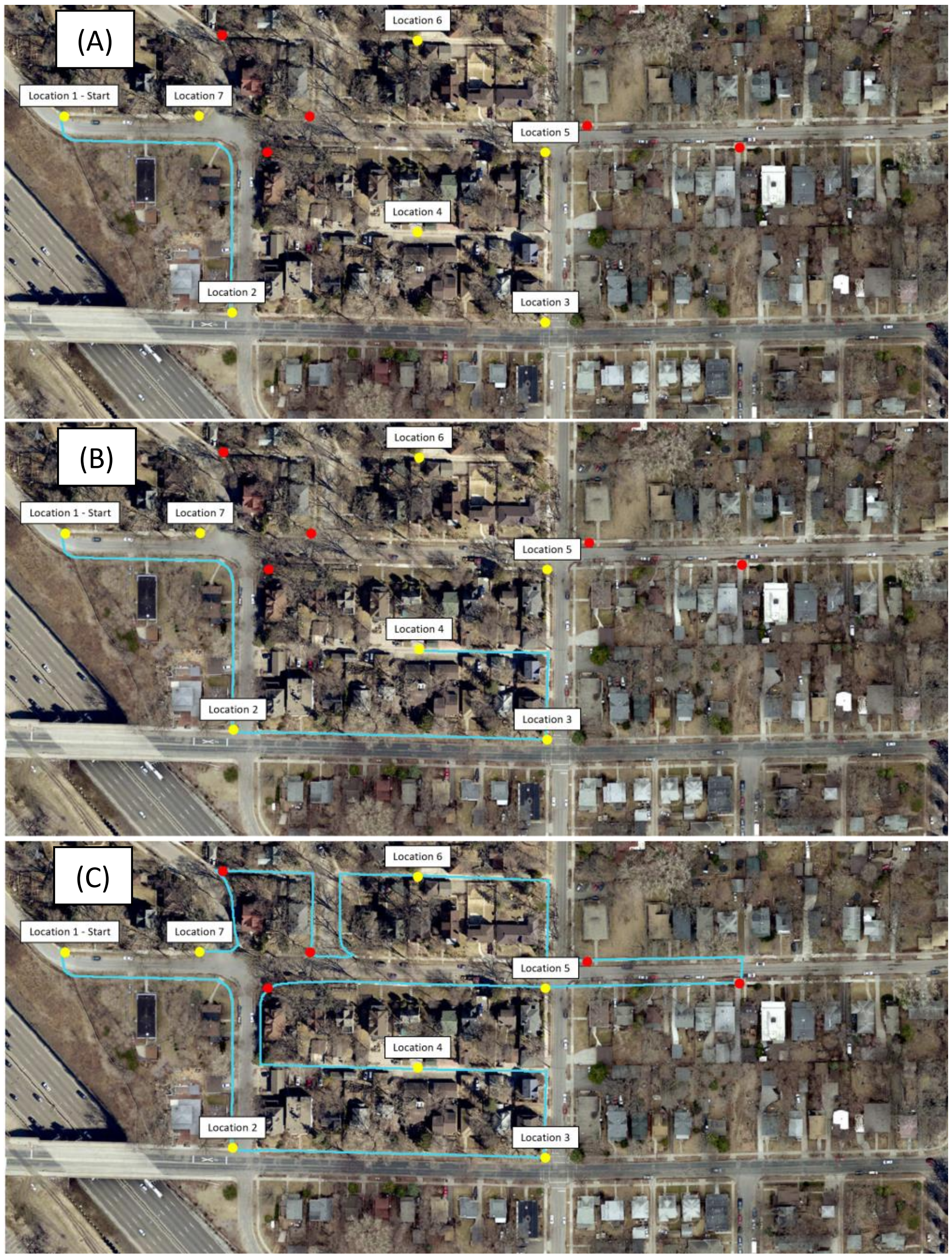

Figure 26- The paths taken by the vehicle between indicated locations shown in light blue. High accuracy GNSS locations shown as yellow circles and intermediate waypoints shown as red circles. (A) The path taken between location 1 and location 2. (B) The path taken between location 1 and location 4 (traveling through locations 2 and 3). (C) The path taken between location 1 and location 7.

Location	GNSS CEP Accuracy [m]	Distance Traveled [m]	Summer Error [m]	Winter Error [m]
1	0.0068	0	0.004	0.012
2	0.0045	186	0.051	0.056
3	0.0044	352	0.032	0.102
4	0.0063	470	0.059	0.062
5	0.0041	832	0.080	0.077

6	0.0036	1333	0.125	0.089
7	0.0049	1615	0.041	0.113

Table 3- Table showing the accuracy of the localization results for 6 different paths. There is no path associated with location 1.

The average error of the localization system is 0.056 m in the summer and 0.073 m in the winter with snow on the ground. This metric was chosen to evaluate the localization system as opposed to the Success weighted by Path Length (SPL) measure proposed by Anderson *et al* [38] because in our case the shortest path between two points is not always the best path. In the above Table 3, in order to calculate the error, the true latitude and longitude of each respective point is converted to X Y coordinates in the vehicle frame. As the vehicle passes through the location, the estimated X Y position from the EKF is compared to the true X Y location, and the error is calculated as the difference. Additionally, the distance traveled is the total distance the vehicle has moved from the start of the path (location 1). The locations in Table 3 are all locations with good GNSS reception, and as a result, it is likely that the vehicle will correct its position estimate as it travels through these points. The error of the position estimate before and after the vehicle travels through the respective points, both in summer and winter, can be seen in Table 4.

Location	**Summer – Error Before [m]**	**Summer – Error After [m]**	**Winter – Error Before [m]**	**Winter – Error After [m]**
1	-	-	-	-
2	0.051	0.016	0.056	0.017
3	0.032	0.012	0.102	0.032
4	0.059	0.018	0.062	0.013
5	0.080	0.021	0.077	0.015
6	0.125	0.029	0.089	0.022
7	0.041	0.012	0.113	0.029

Table 4- Table showing the position estimate correction as the vehicle travels through the high accuracy GNSS locations. Location 1 is the starting point.

To demonstrate the system traveling a longer distance without feeding it waypoints or correction locations, such as in Figure 26 above, the vehicle was given a destination point 500 meters away to navigate to in the winter. The locations of the start and destination locations as well as the path taken between the locations can be seen in Figure 27. On the particular day of this experiment the end location of the path turned out to be in a high accuracy GNSS location, which allowed for the localization error accumulated while traveling the path to be estimated based off where the localization scheme believed the vehicle to be and where the GNSS receiver indicated that it was. This localization error was found to be 0.016 meters. For this particular path, the vehicle did travel through locations 6 and 7 (Figure 24) so corrections based on the high accuracy GNSS were likely made in the localization scheme, however this path was not deliberate. The vehicle crossed two streets while traveling this path and each time the operator supervising the vehicle had to make sure the streets were safe to cross, as the vehicle was not monitoring for traffic.

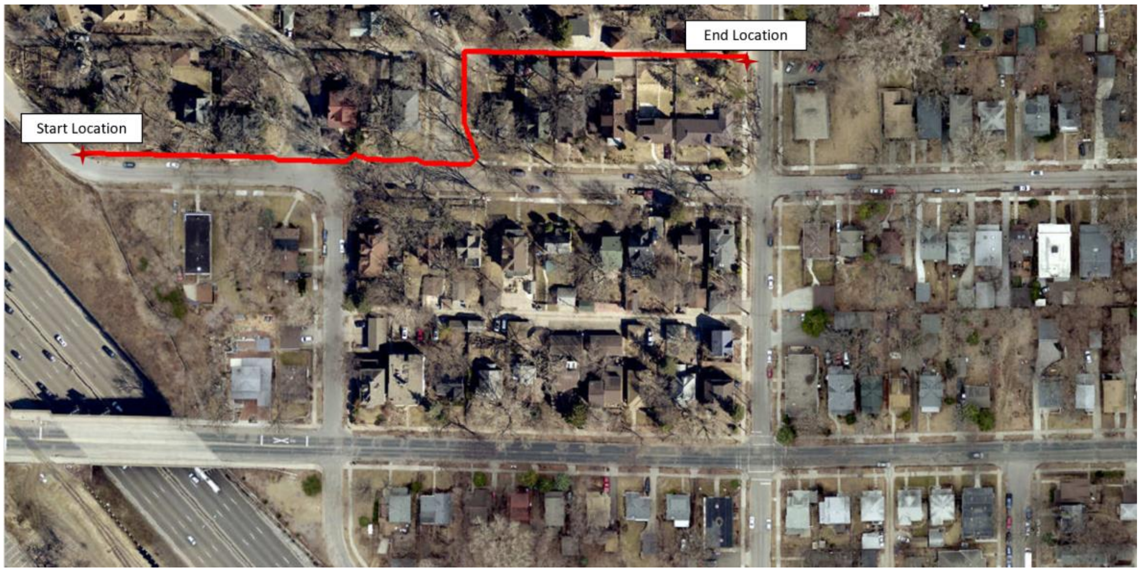

Figure 27- Path of the vehicle (red) traveling between two locations in the neighborhood. For labeled streets and sidewalks refer to Figure 21.

As a check to determine if the proposed localization scheme performs better than just by using the LIDAR odometry alone, the vehicle was driven from location 1 to location 3 (Figure 24) without any GNSS data fed into the EKF (aside from using GNSS for initial localization) during the summer time. Since there is no GNSS information available to the EKF, there will be no GNSS corrections and the localization scheme will rely solely on the LIDAR odometry while traveling between the two points. The path taken by the vehicle can be seen in Figure 28.

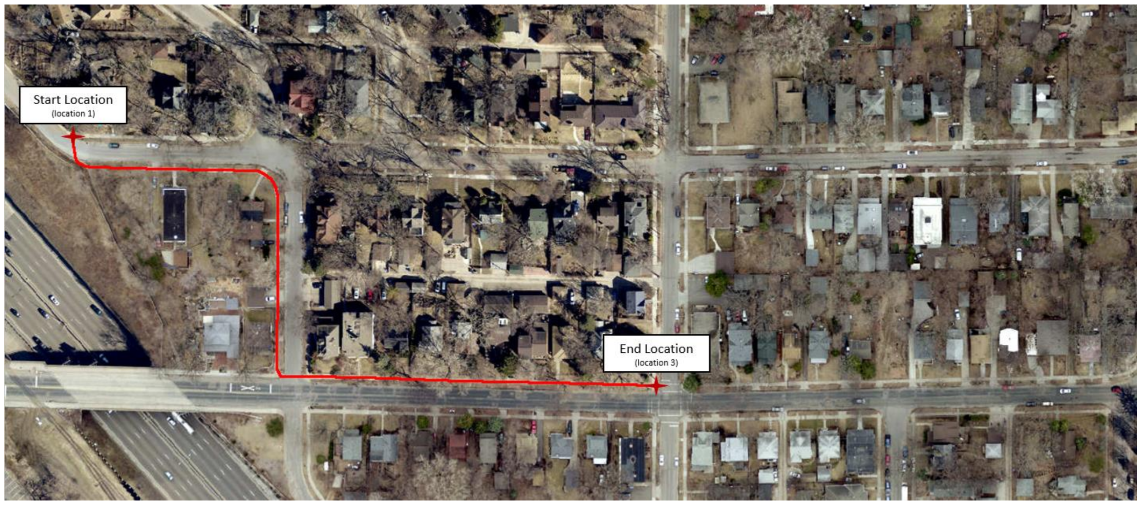

Figure 28- Path of the vehicle (red) traveling between two locations in the neighborhood without any active GNSS input to the EKF.

The error at the end of this 350 meter path was 0.153 meters. This is worse than the 0.032 meter error the vehicle incurred while traveling through location 3 during the summer time with GNSS actively being fed into the EKF (Table 3). This gives us confidence that fusing together the high accuracy GNSS together with the LIDAR odometry does in fact provide us with better localization accuracy. It is important to note that the accuracy of the LIDAR odometry is a function of the number of distinctive features visible to the LIDAR in the nearby environment.

This neighborhood provides many distinct features for the LIDAR to process, however, some parts of the neighborhood may provide more distinctive features than others and we can expect the accuracy of the LIDAR odometry to vary slightly because of this.

6.3 Assumptions and Limitations

A major assumption in this research is that the vehicle starts in a high accuracy GNSS location. This constraint can also be resolved by creating high quality LIDAR based landmarks which allow the robot to localize accurately instead of relying on only GNSS. Also, it is assumed that the sidewalks are of reasonably good quality and are not covered in grass or leaves when collecting the initial data.

Some scenarios where the proposed system may encounter problems are during heavy snowfall or rainfall. The system relies on LIDAR when no GNSS is available and LIDAR data quality deteriorates under these weather conditions. The LIDAR odometry may also drift significantly due to lack of features in the area (as would be the case next to an open field) or finding too many similar features between the high accuracy GNSS locations. This latter situation of too many “similar” features was identified when we were operating on the university campus next to a building with many identical window frames adjacent to the robots path. As a result, the drift may cause the robot to veer off the sidewalk before it can correct for the error with GNSS. This scenario is highly unlikely in a residential neighborhood.

The system does not actively monitor traffic at intersections and as a result is not aware of oncoming vehicles when crossing the intersection. The reason for this was that we had to limit the scope of our project given time constraints. To add the functionality of monitoring traffic at intersections, the LIDAR on the vehicle can be programmed to detect vehicles, bikes, and other pedestrians on the roads. An algorithm could be designed to identify the flow of the traffic in the street and determine if the intersection is clear of any oncoming vehicles. This information could be used to autonomously pause the robot while crossing intersections to avoid the risk of collisions.

Generating localization results in a snowy neighborhood proved challenging. The issues encountered in the winter may have nothing to do with the sensors or the vehicle, but rather can involve piles of snow blocking sidewalks and leaving no viable path for the vehicle to travel. An example of a pile of snow blocking a curbcut that would normally allow the vehicle to get onto a sidewalk is shown in Figure 29.

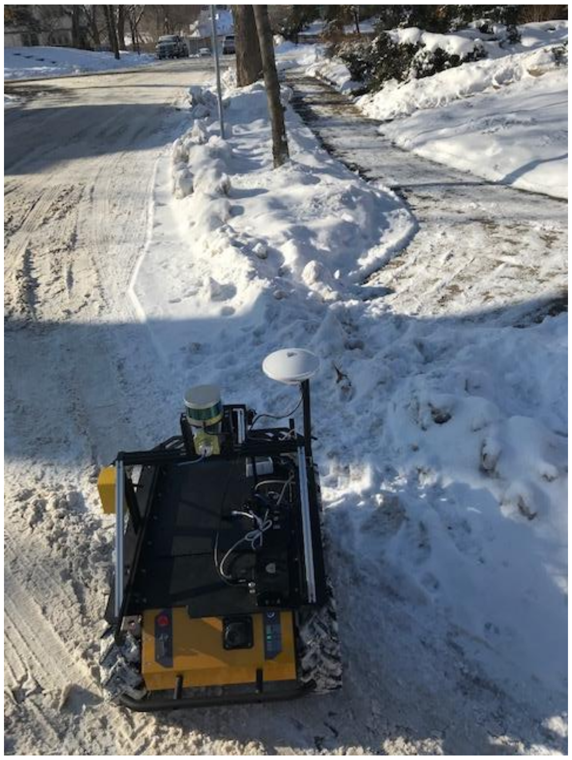

Figure 29- The vehicle encountering a pile of snow it is not able to travel through on a curbcut. The shadows show the difficulty traditional computer vision would encounter in urban neighborhoods.

The quality of snow removal on sidewalks and streets varies highly among neighborhoods and households. This is due to the fact that many households are responsible for the removal of snow on the sidewalks in front of their house. Some people do a good job removing the snow while others may not do it at all. This, in addition to snow plows creating snow piles that block many curbcuts, provides for a challenging environment when it come to sidewalk path planning and navigation in snowy environments. Clearly the local jurisdiction or neighbors need to ensure that the curbcuts are also cleared of snow and not just the sidewalks.

7. Conclusions and Future Work

In conclusion, we developed a system that can automatically localize by fusing high accuracy GNSS and LIDAR Odometry and created an algorithm that can automatically detect and label important ground features such as sidewalks, roads, grass, curb-cuts when there is little to no snow on the ground. By using this information, the robot can travel from pt. A to pt. B using only sidewalks and curb-cuts. This work utilizes Google Maps API to provide the advantage of being able to use street names. The API automatically generates a path along road centerlines between the two points, which then can be modified for sidewalks. We have shown that the system can be used to travel under most weather conditions even when the sidewalks are covered in snow (as long as the snow does not block travel). We have not tested the system in heavy snowfall; this will likely be a problem. Vision is intentionally not being used in this research, thus our system is not typically affected by snow cover.

We achieved an accuracy rating of 91.46% true positives for mapping sidewalks and an average error for the localization scheme of 0.056 meters in the summer and 0.073 meters in the winter for paths as long as 1.5 Km. We showed that the proposed localization scheme provided better accuracy than relying solely on LIDAR odometry by achieving 0.032 meters of error compared to 0.153 meters moving between locations 1 and 3. Although there is less tree canopy above the vehicle in the winter, likely resulting in better GNSS reception throughout the path, the average error was slightly worse than in the summer. This may be attributed to the snow covering some features that LIDAR odometry uses for localization. Better sidewalk classification can be achieved by further research into understanding how to filter out vegetation in LIDAR scans, however, the accuracy achieved in our project has proven to be sufficient for sidewalk navigation of UGVs in urban environments with or without snow cover. The level of localization accuracy achieved in this project is necessary for navigation on sidewalks, especially in the case of autonomous wheelchair navigation. If the accuracy were any worse, the vehicle would likely begin traveling on grass or vegetation next to the sidewalk and miss the curb cuts. For accurate navigation of wheelchairs on sidewalks, RTK GNSS is currently needed as uncorrected GNSS will not provide the accuracy required to stay on the sidewalk and reach its destination address. However, with new GNSS satellites being launched every year, high accuracy GNSS will likely be available without the need for RTK within the next few years.

Table 5 below summarizes the capabilities of the system developed here.

The system is able to:
• Automatically localize using LIDAR odometry fused with GNSS even when the availability of high accuracy GNSS is limited to a few areas
• Automatically detect and label important ground features, such as: - Sidewalks - Roads - Grass - Curb-cuts
• Automatically develop a map of sidewalk centerlines by manually steering vehicle along sidewalks
• Autonomously travel from point A to point B in an urban neighborhood using only sidewalks, paved alleys and curb cuts under most weather conditions
• Take advantage of street names and paths using Google Maps API
• Allow for more rigorous sidewalk assessment, as needed
• Handle heavy shadow due to bright sunlight in the summer or winter

Table 5- Summary of system capabilities.

Future work should explore the introduction of LIDAR landmarks for localization estimation in case the LIDAR odometry drifts in-between high accuracy GNSS locations. LIDAR landmarks could also remove the limitation of needing high accuracy GNSS for initial position estimate as the landmarks could provide this information instead.

The automatically generated segmented map can also be used for in-depth sidewalk assessment of parameters such as pavement quality, grade and width. This can also be updated over time as new data is collected. Introducing a high accuracy IMU into the localization scheme and actively feeding its data into the EKF (as opposed to just using it for initial heading as we do) may improve the localization accuracy and as such is desired. We chose not to actively feed the IMU data into the EKF because we achieved sufficiently good results without it. Implementing the hardware on an actual wheelchair would be a major milestone for this technology.

For longer term consideration, developing a system that would allow travel on the side of the road where there are no sidewalks would facilitate wheelchair travel in the suburbs where there are fewer sidewalks present. Furthermore, although this project has been focused on urban sidewalks, the methods described here (minus the use of the Google Maps API) can just as well be adapted to applications requiring autonomous travel on a network of unpaved pathways in a park or forest, as long as the destination can be uniquely identified and located.

Finally, another application of this technology is the autonomous removal of snow from sidewalks. Most urban centers expect that homeowners remove snow from the sidewalks in front of their properties. Many do not comply (as can be seen from examples described earlier). The result is that persons with disabilities are often stuck at home and cannot make their way to nearby stores or to public transit. Furthermore, cities located in northern climates often do not have a maintenance budget sufficient to clear snow from both the roads and the sidewalks. Snowplowing of the sidewalks by small autonomous vehicles may be the answer.

And let us not forget that in addition to sidewalks, this technology can be expanded to clearing out snow from larger areas, such as a plaza on a university campus or a city center.

Bibliography

[1] H. Qi, J. B. Moore, and I. Fellow, "A Direct Kalman Filtering Approach for GPS/INS Integration," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 2, pp. 687–693, 2002.

[2] W. Ding, J. Wang, C. Rizos, and D. Kinlyside, "Improving adaptive kalman estimation in GPS/INS integration," *J. Navig.*, vol. 60, no. 3, pp. 517–529, 2007.

[3] J. Gao, M. G. Petovello, and M. E. Cannon, "Development of precise GPS/INS/Wheel Speed Sensor/Yaw Rate Sensor integrated vehicular positioning system," *Proc. Inst. Navig. Natl. Tech. Meet.*, vol. 2, no. January, pp. 780–792, 2006.

[4] S. Supakwong, A. Manikas, and A. Constantinides, "Resolving Manifold Ambiguities in Direction Finding Systems," *15th Eur. Signal Process. Conf.*, pp. 95–99, 2007.

[5] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[6] D. H. Won, S. Chun, S. Sung, T. Kang, and Y. J. Lee, "Improving mobile robot navigation performance using vision based SLAM and distributed filters," *2008 Int. Conf. Control. Autom. Syst. ICCAS 2008*, pp. 186–191, 2008.

[7] J. Li, J. Zhao, Y. Kang, X. He, C. Ye, and L. Sun, "DL-SLAM: Direct 2.5D LiDAR SLAM for Autonomous Driving," *2019 IEEE Intell. Veh. Symp.*, no. 4, pp. 1205–1210, 2019.

[8] Y. Deng, Y. Shan, Z. Gong, and L. Chen, "Large-Scale Navigation Method for Autonomous Mobile Robot Based on Fusion of GPS and Lidar SLAM," *Proc. 2018 Chinese Autom. Congr. CAC 2018*, pp. 3145–3148, 2019.

[9] Y. Gu, Y. Wada, L. T. Hsu, and S. Kamijo, "SLAM with 3Dimensional-GNSS," *Proc. IEEE/ION Position, Locat. Navig. Symp. PLANS 2016*, pp. 190–197, 2016.

[10] U. Qayyum, Q. Ahsan, and Z. Mahmood, "IMU aided RGB-D SLAM," *Int. Bhurban Conf. Appl. Sci. Technol.*, pp. 337–341, 2017.

[11] B. Zhou, Z. Tang, K. Qian, F. Fang, and X. Ma, "A LiDAR Odometry for Outdoor Mobile Robots Using NDT Based Scan Matching in GPS-denied environments," *2017 IEEE 7th Annu. Int. Conf. CYBER Technol. Autom. Control. Intell. Syst. CYBER 2017*, pp. 1230–1235, 2018.

[12] H. M. Cho, H. Jo, S. Lee, and E. Kim, "Odometry Estimation via CNN using Sparse LiDAR Data," *2019 16th Int. Conf. Ubiquitous Robot.*, pp. 124–127, 2019.

[13] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 4758–4765, 2018.

[14] P. Wang, R. Yang, B. Cao, W. Xu, and Y. Lin, "DeLS-3D: Deep Localization and Segmentation with a 3D Semantic Map," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 5860–5869, 2018.

[15] R. R. Murphy, *Introduction to AI Robotics*, 2nd ed. MIT Press, 2019.

[16] J. Moras, F. S. A. Rodríguez, V. Drevelle, G. Dherbomez, V. Cherfaoui, and P. Bonnifait, "Drivable space characterization using automotive lidar and georeferenced map information," *IEEE Intell. Veh. Symp. Proc.*, pp. 778–783, 2012.

[17] L. Smadja, M. De Technopole, and L. De Vinci, "Road Extraction and Environment Interpretation From Lidar Sensors," *Computer (Long. Beach. Calif).*, vol. XXXVIII, pp. 281–286, 2010.

[18] A. Chenbo and T. Yichang, "AN AUTOMATED SIDEWALK ASSESSMENT METHOD FOR THE AMERICANS WITH DISABILITIES ACT COMPLIANCE USING 3-D MOBILE LIDAR," *J. Transp. Res. Rec.*, vol. 2542, pp. 25–32, 2015.

[19] T. Shan, J. Wang, B. Englot, and K. Doherty, “Bayesian Generalized Kernel Inference for Terrain Traversability Mapping,” *Proc. 2nd Conf. Robot Learn.*, vol. 87, no. CoRL, pp. 829–838, 2018.

[20] D. Feng *et al.*, “Deep Multi-modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges,” *arXiv Prepr. arXiv1902.07830*, 2019.

[21] V. Swaminathan, S. Arora, R. Bansal, and R. Rajalakshmi, “Autonomous driving system with road sign recognition using convolutional neural networks,” *ICCIDS 2019 - 2nd Int. Conf. Comput. Intell. Data Sci. Proc.*, pp. 1–4, 2019.

[22] D. Neumeister and D. Pape, “Automated Vehicles and Adverse Weather,” no. June, p. 82, 2019.

[23] “Husky UGV - Outdoor Field Research Robot by Clearpath.” [Online]. Available: https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/. [Accessed: 17-Nov-2019].

[24] “Trimble BX992.” [Online]. Available: http://trl.trimble.com/docushare/dsweb/Get/Document-873448/. [Accessed: 17-Nov-2019].

[25] “Puck^{TM}.” [Online]. Available: https://velodynelidar.com/vlp-16.html. [Accessed: 17-Nov-2019].

[26] “PhidgetSpatial Precision 3/3/3 High Resolution - 1044_0 at Phidgets.” [Online]. Available: https://www.phidgets.com/?&prodid=32. [Accessed: 17-Nov-2019].

[27] R. Rajamani, *Vehicle Dynamics and Control*, 2nd ed. Springer, 2012.

[28] “ROS.org | Powering the world’s robots.” [Online]. Available: https://www.ros.org/. [Accessed: 17-Nov-2019].

[29] “robot_localization wiki — robot_localization 2.4.8 documentation.” [Online]. Available: http://docs.ros.org/kinetic/api/robot_localization/html/index.html. [Accessed: 17-Nov-2019].

[30] T. Moore and D. Stouch, “A generalized extended Kalman filter implementation for the robot operating system,” in *Advances in Intelligent Systems and Computing*, 2016, vol. 302, pp. 335–348.

[31] “Integrating GPS Data — robot_localization 2.3.1 documentation.” [Online]. Available: http://docs.ros.org/jade/api/robot_localization/html/integrating_gps.html. [Accessed: 17-Nov-2019].

[32] J. Wang and B. Englot, “Autonomous Exploration with Expectation Maximization,” *Int. Symp. Robot. Res.*, pp. 1–16, 2017.

[33] Z. Li, Y. Xiong, and L. Zhou, “ROS-based indoor autonomous exploration and navigation wheelchair,” *Proc. - 2017 10th Int. Symp. Comput. Intell. Des. Isc. 2017*, vol. 2, pp. 132–135, 2018.

[34] “Mitigation Strategies For Design Exceptions - Safety | Federal Highway Administration.” [Online]. Available: https://safety.fhwa.dot.gov/geometric/pubs/mitigationstrategies/chapter3/3_crossslope.cfm. [Accessed: 21-Nov-2019].

[35] “Google Maps Platform | Google Developers.” [Online]. Available: https://developers.google.com/maps/documentation. [Accessed: 21-Nov-2019].

[36] “Robots/Husky - ROS Wiki.” [Online]. Available: http://wiki.ros.org/Robots/Husky. [Accessed: 29-Apr-2019].

[37] “GIS Open Data Aerial Imagery | Hennepin County.” [Online]. Available: https://gis.hennepin.us/OpenData/imagery/index.html. [Accessed: 22-Nov-2019].

[38] P. Anderson *et al.*, “On Evaluation of Embodied Navigation Agents,” 2018.