Auton Robot
DOI 10.1007/s10514-016-9549-1

A smart wheelchair ecosystem for autonomous navigation in urban environments

Dylan Schwesinger[1] · **Armon Shariati**[1] · **Corey Montella**[1] · **John Spletzer**[1]

Received: 30 March 2015 / Accepted: 7 February 2016

Abstract In this paper, we present a system level approach to smart wheelchair system (SWS) navigation in urban environments. The proposed SWS ecosystem has two primary components: a mapping service which generates large-scale landmark maps, and the SWS vehicle itself, which is a client of the mapping service. The SWS prototype integrates 3D LIDAR/imaging systems which provide robust perception in unstructured, outdoor environments. It also leverages these same sensors for map-based localization. In demonstrating the efficacy of the approach, the SWS navigated autonomously over a distance of more than 12 km in a representative urban environment without once losing localization, and without the use of GPS.

Keywords Service robots · Smart wheelchair system · Large scale mapping · Navigating urban environments

1 Introduction

Smart wheelchair systems (SWS) have been an active research area for over 30 years (Simpson 2005). The spectrum of work has ranged from component level safety sensors, to assistive controllers for steering, to completely autonomous solutions. This research interest is driven in part by their potential for improving the independence and quality-of-life of persons with disabilities and the elderly. Enabling mobility can allow individuals to participate more fully in basic activities such as employment, education, recreation, worship, commerce and other activities of community life that most people take for granted. An additional positive side-effect of increased independence is significant cost savings in healthcare and long-term care (Perry 2002). While the large majority of research to date has focused on indoor operations, there is growing interest in extending SWS capabilities to outdoor environments (Irie and Tomono 2012; Yokozuka et al. 2012).

Our own approach to outdoor navigation for SWS was inspired by our work with autonomous automobiles (Bohren et al. 2008). By using a route network associated with a geographic coordinate system (e.g., latitude and longitude, UTM, etc.), users merely specify a desired goal location and the automobile navigates there autonomously. This is made possible in large-part through the availability of accurate pose estimates (typically from high-performance GPS/INS systems), and exteroceptive sensors which provide rich three-dimensional (3D) data (e.g., Velodyne HDL-64E LIDAR) for robust perception in unstructured environments. Leveraging these sensor technologies, vehicles such as Google's driverless car have driven 100,000 s of miles on public roads demonstrating performance at least as good as their human counterparts (Simonite 2013).

In comparison, the progress in SWS navigation in outdoor, unstructured environments has been surprisingly slow. Part of this can be attributed to significantly less commercial interest. The electric powered wheelchair (EPW) market is far smaller, and insurance providers (e.g., Medicare) are pri-

This is one of several papers published in *Autonomous Robots* comprising the "Special Issue on Assistive and Rehabilitation Robotics".

✉ Dylan Schwesinger
dts211@lehigh.edu

Armon Shariati
ars215@lehigh.edu

Corey Montella
cmontella@lehigh.edu

John Spletzer
josa@lehigh.edu

[1] Lehigh University, Bethlehem, PA, USA

Published online: 23 February 2016

Fig. 1 The Lehigh University smart wheelchair system (SWS) prototype

marily concerned with providing mobility to users indoors. However, another significant hurdle is the relative vehicle cost. Automobiles are an order of magnitude more expensive than EPW systems. While integrating a high-performance GPS/INS system or LIDAR costing $10,000 s USD each might be possible on a luxury automobile, such systems are completely infeasible for use on a commercial SWS. We instead take a different approach (Fig. 1).

First, our emphasis is on navigation in urban environments. This is motivated by the observation that over 80 % of the U.S. population resides in urban areas (United Nations Department of Economic and Social Affairs 2015). While the availability of GPS measurements in urban areas can typically be assumed, multi-path errors from buildings, trees, etc., can significantly compromise its accuracy. However, we further observe that these same structures can be used as landmark features to yield highly accurate relative position estimates. This is illustrated in a simple experiment shown at Fig 2. In this example, the SWS was manually driven around an 80 m triangular-shaped sidewalk loop while logging GPS data. Portions of the path were lined with large trees. As a result, a large section of the path had significant position estimate errors due to multi-path. This is shown in the left satellite image. During the experiment, an extended Kalman filter (EKF) based SLAM algorithm was also executed. It took its initial pose from the GPS, but all subsequent pose estimates were derived from pole-like features (in this case, the trees) that were segmented by a LIDAR system. The resulting pose estimates are dramatically improved, as shown in the right satellite image. This motivates the use of feature-based SLAM algorithms to compensate for GPS errors. Furthermore, as with indoor paradigms, a priori maps

Fig. 2 A demonstration of the ineffectiveness of GPS as method of sidewalk level localization. The GPS signal is overlaid on satellite imagery while driving around the *triangular-shaped* path (*left*). The path derived from SLAM using trees as landmarks (*right*)

embedding accurate landmark locations have the potential to significantly improve localization performance and robustness in urban environments.

To this end, we employ a holistic approach to SWS navigation in urban environments. The proposed *SWS Ecosystem* has two primary components. The first is a mapping service which generates accurate, large-scale landmark maps and an associated route network that are made available through a cloud interface. The second is the SWS itself, which is a client of the mapping service. The SWS prototype integrates 3D LIDAR/imaging systems which provide robust perception in unstructured, outdoor environments. It also leverages these same sensors to perform map-based localization with a demonstrated accuracy at the decimeter level. The net result is an SWS platform with perception and localization systems suitable for autonomous navigation in urban environments. Furthermore, these capabilities are achieved at a monetary cost not prohibitive for the EPW consumer space.

The results presented herein extend our previous work in three significant ways. First, we developed and integrated a 3D LIDAR into the SWS prototype. This improved the safety and effectiveness of the system by enabling robust detection of both 3D landmarks and obstacles. Second, we developed a new vehicle for mapping at the sidewalk level. Using this, we created higher resolution and larger scale maps (three times what was completed in the past), and also addressed the significant issue of sidewalk occlusion that we observed previously. Finally—and in part due to these first two improvements—we have demonstrated autonomy over distances exceeding 12 km. This represents an order of magnitude improvement over our previous work.

2 Related work

The proposed SWS Ecosystem has strong ties to research in smart wheelchair systems, large-scale urban mapping, and autonomous navigation of service robots in urban environ-

ments. While a complete survey of these topics is outside the scope of this document, we have tried to highlight at least the most relevant and recent works in each of these areas.

SWS have been an active research area since the early 1980s. Our own work in the SWS space spans over a decade, and includes (Gao et al. 2010, 2008, 2007; Montella et al. 2014). A survey of the field (as of August 2005) can be found in (Simpson 2005). More recent projects of note include the MIT Intelligent Wheelchair Project (Doshi and Roy 2007), the goal of which is to develop a voice-commanded autonomous wheelchair intended for use in indoor environments. The home, lift, position, and rehabilitation (HLPR) Chair (Bostelman and Albus 2008) developed by NIST is a special-purpose assistive mobility device to provide independent patient mobility for indoor tasks, such as moving to and placing a person on a chair or bed. HLPR has demonstrated obstacle detection and navigation indoors with promising results. The Personal Mobility and Manipulation Appliance (PerMMA) (Cooper et al. 2012) is being developed at the University of Pittsburgh and Carnegie Mellon University with the objective of combining manipulation and mobility assistance in support of complete independence for its users. The system employs two robotic arms, and has demonstrated object manipulation tasks such as retrieving a drink from a refrigerator.

In contrast to these other efforts, our work has been centered around outdoor systems. Our most current research, and the focus of this paper, emphasizes robust navigation in unstructured outdoor environments. Developing robust robotics solutions suitable for use outdoors is a significant challenge compared to indoor environments. The scale is much larger, illumination levels vary from strong sunlight to near complete darkness, the environment is far less structured, environmental conditions can change quickly and dramatically, and simplifying assumptions such as a level ground plane are not reliable. Furthermore, operations at the sidewalk level require localization performance beyond the bounds of what traditional GPS can provide. Since 2012, several other research groups have turned their sights towards outdoor wheelchair systems as well. Yokozuka et al. (2012) employ an approach similar to ours in that they actuate a 2D LIDAR to produce 3D point clouds of the environment. However, they use a 3D voxel grid as the map representation whereas we use a feature-based map to operate in similar outdoor urban environments. They report that their system has autonomously operated for at least two missions of over a kilometer during the Tsukuba Challenge whereas our system has performed autonomously for over 10 km. Also, their approach to obstacle avoidance uses 2D information (the localization component uses the 3D data) which is not as robust as the 3D information used in our approach.

The work by Irie and Tomono (2012) also has a wheelchair navigating an urban environment. Their approach uses maps that are already available and annotated for human use, say from Google maps, and use those to localize in urban environments. Their approach assumes that a grid map is available where grid cells have been labeled as belonging to a roadway, sidewalk, or building and localization is done by detecting the boundaries between these types of regions using stereo vision. This is similar to our approach in that 3D perception is used and that common urban features are used for localization. Key differences are in the chosen sensing modality and map representation. We deliberately ruled out a vision-based approach due to robustness concerns. They demonstrated successful position tracking on a 150 m sidewalk course whereas our system demonstrated successful autonomous navigation (which assumes successful position tracking) of an order of magnitude more distance. Also, their assumption is that annotated maps of the environment will someday exist for easy consumption, but in this work every map had to be hand labeled. In this respect, they still require a mapping procedure.

One inspiration for our mapping approach is the Google mapping trike (Anguelov et al. 2010), which is used in areas where their mapping cars cannot traverse. Their platform has a high-end sensor suite with multiple sensing modalities whereas ours is a relatively low-cost platform. Our mapping approach reduces a 3D representation of the environment to a 2D planar map of features for the SWS client. Chong et al. (2013) use a method whereby a 3D point cloud of the environment is created using a push broom mounting of a 2D LIDAR. They then project points of interest from the 3D point cloud to a 2D plane to create a synthetic 2D LIDAR scan that is invariant to roll and pitch of the mobile platform. They report successful localization on a 1.5 km route with quantitative performance similar to ours. Their localization approach uses a 2D occupancy grid as a map representation whereas we use a feature based representation. They also demonstrate two successful autonomous missions on two shorter sub-routes. We have demonstrated both short term and long term autonomy in fifteen autonomous missions.

Mapping at the sidewalk level requires a sufficiently accurate six degree of freedom pose estimate in a global coordinate frame in GPS compromised areas. Baldwin and Newman (2012) use a push-broom style 2D LIDAR with the goal of providing an accurate pose estimate by using probabilistic methods on small swaths of aggregated push-broom scans. They also report better accuracy with their method in a GPS compromised outdoor environment than a reference high-performance GPS/INS system. Their localization approach uses previously seen swathes of the environment represented as point clouds. During operation of the localization system, the currently observed swathe is matched to the most likely swathe observed in a prior experience and the pose is estimated based on that match. They report successful localization results on 26 km of roadway, but have not demon-

strated autonomous navigation. We have not autonomously operated that amount of distance, but emphasize that we are operating at human scale, rather than road vehicle scale.

Lategahn et al. (2013) map urban environments using predominately a stereo camera and IMU. Although, they use a GPS measurement to initialize the pose estimate. Similar to our work, they use 3D features as landmarks. However, their localization approach uses only a monocular camera and IMU in conjunction with the landmark map to provide an accurate 6 degree of freedom pose estimation. Our client system operates under a 3 degree of freedom pose assumption. Their system demonstrated a mean localization performance accuracy of 10 cm on a 10 km mapped urban environment. Our mapped environment is smaller, but our localization results are similar. Furthermore, while their work focuses on localization, our SWS Ecosystem offers a more complete solution for autonomous navigation.

Finally, an important aspect to navigation at the sidewalk level is the ability to handle dynamic obstacles in the environment such as pedestrians. Kummerle et al. (2013) present a tour guide robot that autonomously navigated through a crowded 3.3 km urban route. They use a grid map representation of the environment and a particle filter based approach to localization. They demonstrate qualitatively successful localization results whereas we provide a quantitative analysis of localization accuracy. Their approach relies on 2D information for localization and dynamic obstacle avoidance. While we are also concerned with dynamic obstacles, the emphasis of our work is in localization and mapping. Robustly handling dynamic obstacles is outside the scope of this work. However, our system utilizes 3D information for obstacle avoidance and can potentially perform better when there are low lying dynamic obstacles, such as dogs, which their system had trouble with.

3 SWS ecosystem overview

As alluded to earlier, the key for our SWS to navigate reliably in an urban environment is having an accurate landmark map and route network. The landmark map consists of the absolute locations of landmark features (in this case, pole-like features) and is used for localization purposes. The route network is used to indicate wheelchair accessible paths with respect to the landmarks and is used for path planning.

We approach the problem from a client service standpoint. Hence, our ecosystem is composed of two major systems: the SWS acting as a consumer of landmark maps and route networks, and a service platform used to generate landmark maps and route networks, and to make them available to SWS clients. The envisioned approach is to use a sidewalk-level mapping vehicle with high quality sensing equipment to enable a high fidelity 3D reconstruction of the environment. Then, the 3D reconstruction could be reduced to salient landmark features and their absolute locations with respect to a global coordinate frame. Route networks could then be generated via the service mapping platform or by manually driving the client SWS on safe paths in a learning phase before autonomous operation as in Gao et al. (2010). These landmark maps would then reside in the cloud, and the SWS would download the landmark map and route network based on its location and the desired goal locations chosen by the operator.

In this paper, we describe a proof of concept implementation of the envisioned system. The mapping service platform and the map generation process is outlined in Sect. 4. The client SWS platform and methods it uses to perceive and navigate the environment are described in Sect. 5. Following the specifics of the service and client components, results of using the system in practice are discussed in Sect. 6.

4 Server side map generation

This section details the service component tasked with generating and providing the landmark map and route network. The map representation used by the system is feature-based, which was motivated by the need for the SWS to localize in a GPS compromised environment utilizing a relatively low cost sensor suite. The global feature map was generated by capturing and synthesizing a dense point cloud representation of the environment. Urban environments offer a plethora of features for tracking. In this work, we focused on pole-like features—herein referred to simply as *pole features*—such as parking meters, lamp posts, trees, etc. Examples of typical pole features in our map are shown in Fig. 3. In our previous work (Montella et al. 2014), we employed a street level mapping platform (a car) that had the advantage of utilizing a high-end OXTS RT-3050 GPS/INS system which provided accurate positional information. However, a shortcoming with mapping from the street was that features at the sidewalk level could be occluded by obstacles such as parked cars. This led to the construction of a sidewalk level mapping platform.

Fig. 3 Some example pole features. From *left* to *right* a street sign, parking meter, lamp post, and fire hydrant

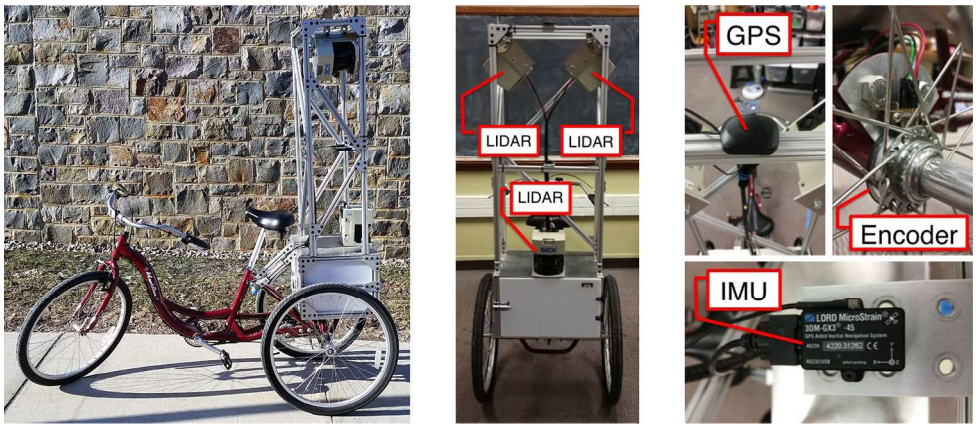


Fig. 4 The Mapping Trike platform (*left*) . A rear view of the Mapping Trike with mounting positions of the LIDARs indicated (*center*). Close up pictures of the GPS antenna, wheel encoder, and the Microstrain 3DM-GX3 IMU (*right*)

4.1 The Mapping Trike platform

Taking inspiration from Google's mapping efforts (Anguelov et al. 2010), the Mapping Trike platform was built upon a commercial tricycle augmented with a sensor suite, depicted in Fig. 4. A Microstrain 3DM-GX3-45 inertial measurement unit (IMU) with GPS antenna and two 4096 cycles per revolution (CPR) resolution encoders were used to provide pose estimation. Point data from two side-facing LIDARS (SICK LMS291-S14) and one rear-facing LIDAR (SICK LMS291-S05) were used to detect landmark features. The Mapping Trike was driven manually around an area of interest while logging data from all of its sensors. Data were then post-processed to synthesize the landmark map.

Synthesizing the map takes place in two main stages. The initialization stage creates a high fidelity 3D reconstruction of the environment via the side-facing LIDARS. This reconstruction is then reduced to landmark features that have an associated position and covariance. These landmark statistics serve as an initial estimate of the landmark map, which is later refined to correct positional errors of the landmarks.

4.2 Mapping stage 1: initialization

The initial map is generated in three main stages: pose estimation, landmark segmentation, and map synthesis. A data-flow diagram for the map initialization phase is depicted above the dashed line in Fig. 5.

4.2.1 Pose estimation

The localization module (Sect. 5.4) of the SWS works under the assumption that the landmark map is a 2D plane. Hence, the mapping goal is to register the landmarks to a consistent 2D global coordinate frame that is sufficiently planar in local neighborhoods. In our case, we use the Universal Transverse Mercator (UTM) coordinate system, which maps positions

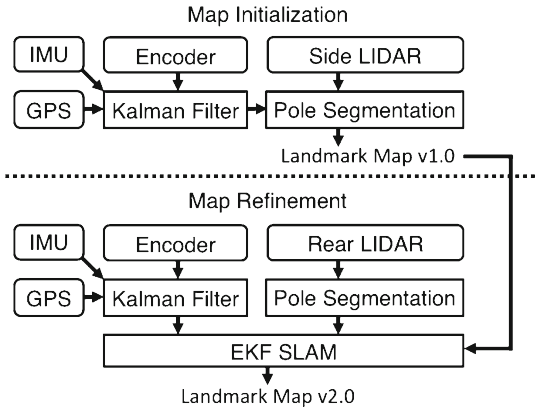


Fig. 5 A block diagram representing the data flow of the mapping process. The section *above the dashed line* represents the map initialization phase and the *section below the dashed line* represents the map refinement phase

on the globe to a 2D Cartesian coordinate frame. To achieve this goal, the estimated pose of the trike with respect to the UTM frame must be accurate to within the operating tolerances of the SWS. The 3DM-GX3-45 has an integrated an on-board GPS, but its accuracy was insufficient to accurately register the trike to the UTM frame even when fused with inertial measurements from the IMU. Our solution was to integrate feedback from the wheel encoders and incorporate the kinematics of the trike into an EKF. Thus, the trike pose was estimated by using the vehicle kinematics as the predictive step, and fusing the GPS and inertial measurements for the corrective step. We used a predictive motion model of the form:

$$\mathbf{x}_{k+1}^{-} = \begin{pmatrix} x \\ y \\ z \\ \alpha \\ \beta \end{pmatrix}_{k+1} = \begin{pmatrix} x \\ y \\ z \\ \alpha \\ \beta \end{pmatrix}_{k} + \begin{pmatrix} \cos\alpha\cos\beta & 0 \\ \sin\alpha\cos\beta & 0 \\ \sin\beta & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta_l \\ \Delta_a \end{pmatrix}, \tag{1}$$

where $\mathbf{x}$ denotes the state vector consisting of the Cartesian coordinates (x, y, z) and the yaw and pitch angles (α, β), Δ_l and Δ_a were the linear and angular displacements over some small time step derived from the encoders. The k subscript denotes a time step and the superscript $^-$ denotes the prediction of the state and covariance after the motion update. The roll angle was not tracked because the effect of roll was minimal on flat sidewalks, but tracking the pitch was crucial as small changes in pitch could distort the projection of the rear-facing LIDAR data. For example, the map in Sect. 6 has hills with a grade of approximately 8.5 %.

For the purposes of the EKF, the covariance P was updated as:

$$P_{k+1}^- = J_s P_k J_s^T + J_m Q J_m^T, \tag{2}$$

where J_s was the Jacobian with respect to the state defined as:

$$J_s = \begin{pmatrix} 1 & 0 & 0 & -\Delta_l \sin\alpha\cos\beta & -\Delta_l \cos\alpha\sin\beta \\ 0 & 1 & 0 & \Delta_l \cos\alpha\cos\beta & -\Delta_l \sin\alpha\sin\beta \\ 0 & 0 & 1 & 0 & \Delta_l \cos\beta \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3}$$

J_m was the Jacobian with respect to the linear and angular displacement defined as:

$$J_m = \begin{pmatrix} \cos\alpha\cos\beta & 0 \\ \sin\alpha\cos\beta & 0 \\ \sin\beta & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}, \tag{4}$$

and Q is additive Gaussian noise in the linear and angular displacements.

The 3DM-GX3-45 streams the GPS data and IMU data at different rates, 4 and 100 Hz respectively. So, in our EKF formulation each of these data streams has a separate measurement update. The GPS measurement update is computed with the following sequence of equations:

$$\begin{aligned} K_{g,k} &= P_{k+1}^- H_g^T (H_g P_{k+1}^- H_g^T + R_g(k))^{-1} \\ \mathbf{x}_{k+1} &= \mathbf{x}_{k+1}^- + K_{g,k}(\mathbf{z}_g - H_g \mathbf{x}_{k+1}^-) \\ P_{k+1} &= (\mathbf{I} - K_g H_g) P_{k+1}^-, \end{aligned} \tag{5}$$

where $\mathbf{z}_g$ is the GPS measurement represented as a UTM coordinate, $\mathbf{x}$ is the state vector, and H_g is a projection matrix to extract the x and y Cartesian coordinates from the state vector defined as:

$$H_g = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}. \tag{6}$$

$R_g(k)$ is a covariance matrix that represents the uncertainty of the GPS measurement at time step k, $\mathbf{I}$ is the identity matrix, $\mathbf{x}_{k+1}$ and P_{k+1} are the corrected state and covariance respectively. The IMU itself does some on-board filtering and the covariance of the GPS measurement can be requested. This is the value used for $R_g(k)$.

Similarly, the gyro measurement update was computed with the sequence of equations:

$$\begin{aligned} K_{a,k} &= P_{k+1}^- H_a^T (H_a P_{k+1}^- H_a^T + R_a(k))^{-1} \\ \mathbf{x}_{k+1} &= \mathbf{x}_{k+1}^- + K_{a,k}(\mathbf{z}_a - H_a \mathbf{x}_{k+1}^-) \\ P_{k+1} &= (\mathbf{I} - K_{a,k} H_a) P_{k+1}^-, \end{aligned} \tag{7}$$

where $\mathbf{z}_a$ is a gyro measurement of the form $[\alpha, \beta]^T$, and H_a is a projection matrix to extract the yaw and pitch from the state vector defined as:

$$H_a = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{8}$$

$R_a(k)$ is a covariance matrix that represents the uncertainty of the gyro measurement at time k, $\mathbf{I}$ is the identity matrix and $\mathbf{x}_{k+1}$ and P_{k+1} are the corrected state and covariance respectively.

4.2.2 Landmark segmentation

The EKF described above provided an acceptable pose estimate for the mapping trike. The next step was to register the LIDAR scans to a common coordinate frame (in our case, the Standard UTM frame), as we are relying upon the motion of the trike to build the 3D reconstruction of the sidewalk scene via 2D laser scans. Because of this, in our data collection we aimed for a speed of approximately 1 m/s. In conjunction with the LMS291 LIDAR scan rate of 75 Hz, this gave us a vertical scan for each 1–2 cm of distance traveled.

Once we had a sufficiently dense set of LIDAR scans registered to a common frame, we segmented the pole features by aggregating subsequent 2D scans into a 3D point cloud. The scans were aggregated in a sliding window fashion. The first n scans made up the first point cloud and subsequent point clouds began at the middle scan of the previous point cloud. The reason for this overlap was that a landmark could be missed if it straddles the boundary of the two windows. Additional bookkeeping was done to eliminate the duplicate landmarks in the overlapping regions. For each window, the points corresponding to the ground plane were removed using a RANSAC procedure. The remaining points were then clustered based on the Euclidean distance to neighboring points. The maximal intra-cluster distance was set to 10 cm. An

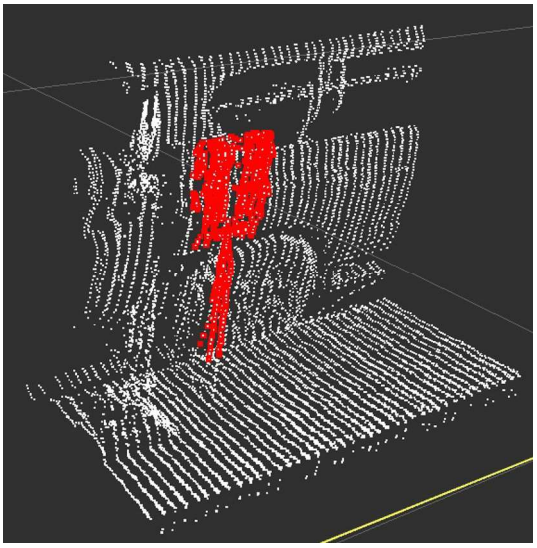


Fig. 6 Pole segmentation of a parking meter from the dense 3D reconstruction of a side facing SICK LIDAR

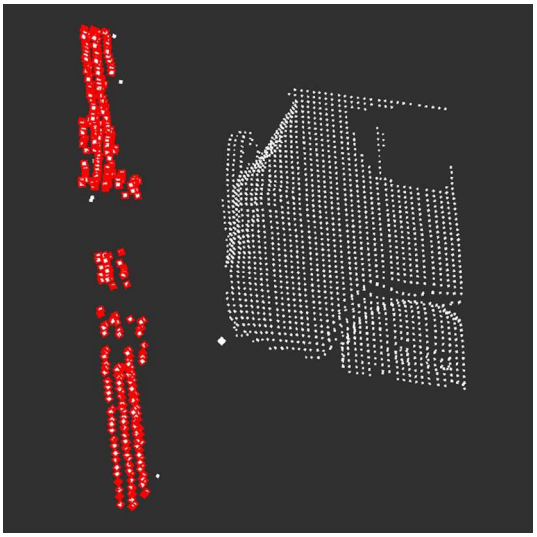


Fig. 7 An example of a lamp post that would be clustered into two components due to a section of invalid measurements

example of the segmentation of a parking meter from a window containing 100 scans is shown in Fig. 6.

Unfortunately, some features were clustered into several disjoint clusters using this approach. This was most likely to occur when observing lamp posts due to the low albedo surface. An example of a lamp post that would be improperly segmented is shown in Fig. 7. Based on the observation that vertical features are the focus of our landmark segmentation, we performed a cluster merging step to combine clusters that were likely to belong to the same vertical object. The centroids of each cluster were projected to the $x-y$ plane. If the Euclidean distance between two centroids was less than 20 cm, the clusters were merged into a single cluster. This merging phase significantly increased the accuracy of the entire segmentation process.

The next step was to validate that a cluster was a pole feature by passing some validation gates. The approach of fitting a cylindrical model, while intuitive was not very effective. This was especially true for features like street signs and parking meters which lacked uniformity. That being the case, we validated a cluster's status as a pole by first performing a spectral decomposition of the covariance matrix of its data points. This gave us information that was invariant to the coordinate system in which the data was measured. The eigenvector associated with the largest eigenvalue λ_1 was compared against the vector $[0, 0, 1]^T$ (corresponding to the gravity vector). If the angle between them was outside some tolerance, the cluster was rejected as a landmark; a cluster passing this validation gate was assumed to be a tall vertical object. We further examined the aspect ratio of the cluster. Since the eigenvalues correspond to the variances in the principal directions and the largest direction was already determined to be vertical, several ratios of the eigenvalues were tested: $\lambda_2/\lambda_3 \approx 1$ to validate that the width and depth were similar and $\lambda_1/\min(\lambda_2, \lambda_3) > 2$ to validate the object was at least twice as tall as it was wide. Only after passing these gates was a cluster considered a candidate landmark.

4.2.3 Landmark map v1.0

Ultimately, the landmarks associated with each cluster needed to be transformed to 2D UTM coordinates. The reason for this is that the client SWS is not assumed to have the sensing capability to track its position in $\mathbb{R}^3$. As such, the trike pose and centroid of each cluster were transformed to UTM coordinates and each landmark cluster was reduced to a feature vector of the form $[\rho, \phi, s]^T$ where ρ and ϕ were the range and bearing to the transformed landmark's centroid with respect to the trike's transformed pose at which the landmark was detected and s was a *signature* for the landmark (in our case the radius of the pole). The signature value aids in data association in the map refinement step (Sect. 4.3) and was computed by finding the bounding box of the landmark cluster's data points and returning $min(width, depth)/2$ as the radius.

For our landmark map, we wish to embed both the mean position and covariance of each landmark in the UTM frame. Using the 2D projection of the trike's pose estimate, denoted as $\mathbf{x} = [x, y, \theta]^T$, we converted the landmarks to the UTM frame as follows:

$$\mu_L = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = \begin{pmatrix} x + \rho\cos(\phi + \theta) \\ y + \rho\sin(\phi + \theta) \end{pmatrix}, \tag{9}$$

where μ_L denotes the mean location of a landmark. To compute the covariance we need to transform the noise model of

the LIDAR to Cartesian coordinates. Note that the variance of the signature is not considered here as it does not depend on the position. The noise model is denoted as:

$$Q = \begin{pmatrix} \sigma_\rho & 0 \\ 0 & \sigma_\phi \end{pmatrix}, \tag{10}$$

and is linearized with the Jacobian of μ_L with respect to ρ and ϕ, defined as:

$$H_Q = \begin{pmatrix} \cos(\phi+\theta) & -\rho\sin(\phi+\theta) \\ \sin(\phi+\theta) & \rho\cos(\phi+\theta) \end{pmatrix}. \tag{11}$$

However, we also need to propagate the uncertainty in the trike's pose into the uncertainty in the landmark's position. The linearization is performed using the Jacobian of μ_L with respect to x, y, and z, defined as:

$$H_t = \begin{pmatrix} 1 & 0 & -\rho\sin(\phi+\theta) \\ 0 & 1 & \rho\cos(\phi+\theta) \end{pmatrix}. \tag{12}$$

The covariance for a landmark's position is then

$$\Sigma_L = H_t \Sigma_t H_t^T + H_Q \Sigma_Q H_Q^T. \tag{13}$$

The initial map estimate is the set of means and covariances for each landmark feature denoted

$$M_{v1} = ((\mu_{L1}, \Sigma_{L1}), \ldots, (\mu_{LN}, \Sigma_{LN})) \tag{14}$$

where N is the number of landmarks.

4.3 Mapping stage 2: refinement

In our previous work (Gao et al. 2010), the mapping vehicle employed a high performance OXTS RT-3050 GPS/INS system for pose estimation. Despite this luxury, the residual errors in initial landmark positions were large enough to make SWS localization impossible. As a result, a second mapping stage was required to refine the landmark positions. Furthermore, the 3DM-GX3-45 used on the Mapping Trike lacks the accuracy of the OXTS RT-3050, and so a map refinement step was expected.

The map refinement stage used an EKF simultaneous localization and mapping (SLAM) approach modified to use M_{v1} as an input. While the map initialization stage was performed off-line using the trike's side-facing LIDARs, the map refinement stage was performed on-line using trike's rear-facing LIDAR. Extracting poles from the rear-facing LIDAR was done by first registering the laser scan to the trike's coordinate frame. The point data in the scan were then clustered based on the Euclidean distance of neighboring points; the maximal intra-cluster distance was set to 10 cm. This value was chosen to capture pole features at a range of approximately 8 m. Any cluster of less than four points was rejected as a possible landmark. A circle model was then fit to each remaining cluster using RANSAC (Fischler and Bolles 1981) and the model parameters were refined using a least squares fit to the inliers. Any cluster that had less than 90 % inliers (a measure of model fit) or a fitted circle radius greater than 40 cm (none of the landmarks have a radius this large) was rejected as a landmark. Each accepted landmark was then put into the $[\rho, \phi, s]^T$ form by converting the center point of the fitted circle to polar coordinates with respect to the trike. The radius of the circle was used for the signature s. Note that the center of the fitted circle was used rather than the centroid of the cluster's data as it was a better approximation of the landmark's actual location when seen from different viewing angles.

As in a typical EKF SLAM implementation, the vehicle pose and map locations were represented as a Gaussian with mean vector $\mu = [\mathbf{x}, \mathbf{m}_1, \ldots, \mathbf{m}_n]^T$ where $\mathbf{x} = (x, y, \theta)$ was the trike's pose and $\mathbf{m}_i = (x_i, y_i, s_i)$ was the i^{th} landmark's position and signature, and covariance, defined as:

$$\Sigma = \begin{pmatrix} \Sigma_{xx} & \Sigma_{xm_1} & \cdots & \Sigma_{xm_n} \\ \Sigma_{m_1x} & \Sigma_{m_1m_1} & \cdots & \Sigma_{m_1m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_nx} & \Sigma_{m_nm_1} & \cdots & \Sigma_{m_nm_n} \end{pmatrix}, \tag{15}$$

where Σ is a block matrix and each block corresponds to the covariance of the pose Σ_{xx}, the pose with respect to landmark Σ_{xm_i}, or the covariance of a landmark $\Sigma_{m_im_i}$.

The first step to our map refinement is to pre-process M_{v1} by tagging a small number of specific landmarks with accurate position information using satellite imagery. This information provides known correspondences for these features and aids in loop closure. We found this necessary, as the loops we are considering are on the order of a kilometer in length. The SLAM measurement update step computes the Mahalanobis distance from an observation to each landmark in M_{v1}, as well as to each landmark in the refined map maintained by the EKF denoted as M_{v2}. The maximum likelihood correspondence (MLC) (Thrun et al. 2005) is selected and the update has three possible outcomes:

(1) The MLC is below some threshold and the observation is discarded.
(2) The MLC is associated with a landmark in M_{v2} and the EKF is updated normally.
(3) The MLC is associated with a landmark in M_{v1} and the landmark feature is removed from M_{v1} and added to M_{v2}. Additionally, if the landmark has associated satellite data, then that position is used when adding the landmark to M_{v2}.

Fig. 8 A comparison of an initial landmark map to the refined map. The trike started on the left with an acceptable GPS position estimate and traveled to the right up a grade of 8.5 %. The *red triangles* indicate the landmarks in M_{v1} and the *yellow circles* indicate the landmarks in M_{v2}. A landmark at each corner was tagged with satellite imagery positional information to act as known correspondences (Color figure online)

The refined landmark map v2.0 is the result of the EKF SLAM procedure. Figure 8 shows the difference between one instance of the initial map compared to the refined map on Webster Street. The trike started on the left with an acceptable GPS position estimate and traveled to the right up a grade of approximately 8.5 %. The red triangles indicate the landmarks in M_{v1} and the yellow circles indicate the landmarks in M_{v2}. A single landmark at each corner was tagged with satellite imagery positional information to act as known correspondences.

4.4 Route network generation

The SWS uses a route network for path planning containing information about wheelchair accessible paths with respect to the landmark map. The route network is modeled as a graph $G(V, E)$ where waypoints correspond to the vertices $v_i \in V$ and the edges $e_{ij} \in E$ correspond to directed edges connecting pairs of waypoints. Each edge also has semantic information associated with it: a speed limit, a stop condition, and a weight. The speed limit is used to constrain the local path planner (Sect. 5.5.2) to a maximum speed in areas of the map where high speed traversal could be unsafe. The stop condition is used in areas of the map, such as street crossings, where the operator of the SWS must determine when it is safe to cross. The edge weight is used by the global planner to search for paths that minimize the expected time of arrival.

The route network was constructed by first sampling the trike poses as corrected by the map refinement step. The trike was driven on acceptable wheelchair accessible paths in order to ensure that the sampled path is valid for the SWS. After the initial route network was constructed, semantic information associated with the edges was added manually. While not fully automated, the associated workload is not too cumbersome. For example, the route network used in our experiments (Sect. 6) only had to have the six street crossings annotated by hand.

5 The smart wheelchair system (SWS) client

This section describes the operation of the SWS under the assumption that the landmark map and route network are available. The SWS software has three major components: perception, localization, and navigation. The perception component of the SWS has two primary tasks: pole feature segmentation for localization (Sect. 5.4), and populating a cost map for obstacle avoidance (Sect. 5.5.1). The localization component maintains an estimate of the SWS with respect to the landmark map by utilizing the output of the pole feature detection process. Finally, the navigation component (Sect. 6.3) uses the route network for global planning and the cost map (populated via the 3D sensors) to generate local trajectories. A detailed description of each component, as well as the SWS hardware platform, now follows.

5.1 The SWS platform

The SWS used in this work is pictured in Fig. 9. It is an Invacare M91 Pronto that integrates the FDA approved motion control module (MCM) developed under our previous work (Gao et al. 2008). The MCM provides a seamless interface for regulating SWS linear and angular velocities.

Fig. 9 The SWS prototype with the fields of view of the actuated LIDAR (*blue*) and each 3D camera (*orange*) highlighted. A profile view (*left*) and a top down view of the respective sensors' fields of view (*right*) (Color figure online)

Proprioceptive sensing includes high resolution quadrature encoders (4096 CPR) and a Microstrain 3DM-GX1 IMU. The latter is used to enhance the odometry performance by providing gyro corrections. Exteroceptive sensors include two IFM O3D200 3D cameras and an actuated Hokuyo UTM-30LX 2D LIDAR (described in detail in Sect. 5.1.2). These provide the necessary 3D sensing for robust obstacle detection and landmark segmentation. The total cost of all sensors is <$10K USD in quantities of one.

The software was developed using the Robot Operating System (ROS) (Quigley et al. 2009) framework and modularized based on the message passing paradigm used by ROS. Additionally, the Point Cloud Library (PCL) (Rusu and Cousins 2011) was leveraged for processing point cloud data from the exterioceptive sensors. The computational processing for the software components was performed by a laptop with an Intel Core i7-2760QM 2.4GHZ processor and 4 GiB of RAM.

5.1.1 IFM O3D200

The primary role of the IFM O3D200 3D cameras was to detect obstacles in the environment and propagate that information to the local map (Sect. 5.5.1). While the sensor has its limitations (an effective range of ≈6 m, a resolution of 48 × 60 pixels, a field of view of 30° × 40°, and a relatively low frame rate of ≈7 Hz), it has one critical capability. Specifically, the O3D200 can provide 3D measurements in outdoor conditions, including bright sunlight. This is something that lower-cost "Kinect-based" sensors are incapable of doing. It is also reasonably affordable, costing ≈$1500 USD. In our previous work (Montella et al. 2014) the 3D camera was crucial for robust obstacle detection as it was the only 3D sensing modality used. The current SWS integrates a second 3D sensing modality (Sect. 5.1.2), but the O3D200 is still useful for ground plane segmentation and to cover the near field and low-lying blind spots. In addition, to compensate for the limited field of view, two O3D200 cameras are used. Figure 9 shows the mounting positions and depicts the vertical and horizontal fields of view for each sensor.

5.1.2 Actuated 3D LIDAR

The SWS features an actuated Hokuyo UTM-30LX LIDAR, herein referred to as the 3D Hokuyo. Although it was also used for obstacle detection, the primary role of the 3D Hokuyo was landmark segmentation. Our motivation for actuation was due to robustness concerns in our previous work (Montella et al. 2014). Specifically, localization with the UTM-30LX was done using 2D information, but the SWS operates in a 3D environment. While this was sufficient for demonstration purposes, robust segmentation of 3D landmarks requires a 3D sensor. The 3D Hokuyo was actuated with a Dynamixel AX-12A servo which was encased in a custom 3D printed enclosure. The 3D Hokuyo is depicted in the left image of Fig. 10. The entire 3D Hokuyo system has a small footprint, and is comparable in size to the standard joystick controller on the right arm of the SWS. The actuated behavior was to simply nod up and down over a limited angular range. The Dynamixel servo is well suited for this task, as it can be controlled by explicitly setting angle commands and polled for its current angle position.

The task of the 3D Hokuyo was to aggregate the 2D laser scans from one angular set point to the next (a sweep either up or down), register each scan to a common odometric coordinate frame to account for SWS motion, and then report the aggregation as a 3D point cloud. The design target for the 3D Hokuyo was to stream aggregated 3D scans at 5 Hz, as this was the frequency of the motion planner's control loop. A constraint on this goal was that the vertical angular resolution had to be sufficiently small in order to effectively segment landmarks. The free parameters (the choice of angular set points) were empirically determined to be ±5° from the neutral position (the LIDAR scan parallel to the ground). This choice gave us a field of view of 270° × 10°.

Figure 10 shows histograms of the salient operating characteristics of the actuated LIDAR in operation. Due to factors such as inertia, the scan angles in a sweep, the number of scans in a sweep, and the amount of time a sweep takes are not deterministic. The results show that an average sweep of the actuated LIDAR contains approximately 7.5 laser scans. As the UTM-30LX scans at 40 Hz, the effective update rate of the 3D Hokuyo was 5.35 Hz, which was sufficiently close to our design target of 5 Hz. Figure 11 shows a visualization of the 3D point cloud data from one sweep.

5.2 Ground plane tracking

Reliable obstacle detection and mapping requires the local ground plane to be tracked. For the SWS, this is accomplished using the O3D200 3D cameras and employing an iterative reweighted least squares (IRLS) approach. IRLS has advantages over a traditional RANSAC approach in that it integrates both temporal filtering as well as regularization

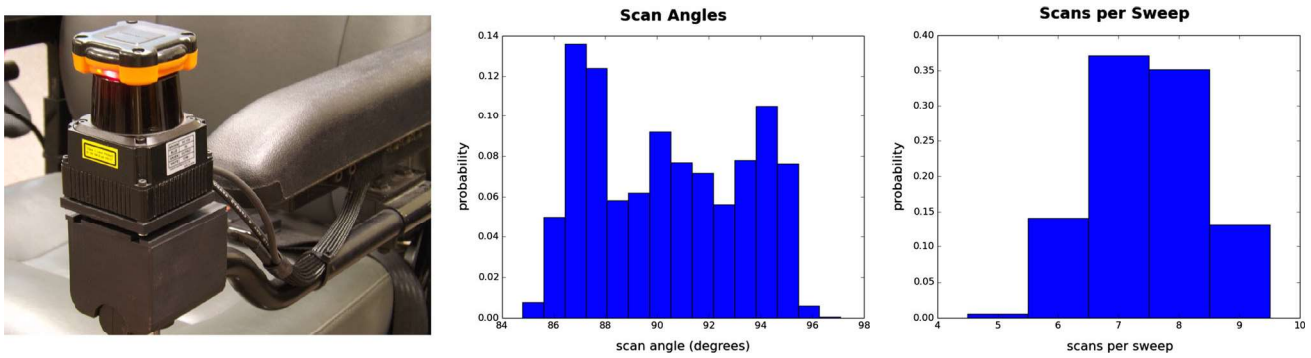


Fig. 10 The actuated LIDAR mounted on the SWS (*left*). Histogram of the representative scan angles in a sweep (*center*) and histogram of the number of 2D scans in a sweep (*right*). 3D aggregates of the scans in each sweep are streamed at ≈5 Hz

Fig. 11 A scan from the 3D Hokuyo (*top*). A photo of the corresponding scene (*bottom*). Clusters in the former from landmark poles are clearly visible

through the use of a calibrated ground plane. In (Bohren et al. 2008), we found that it outperformed RANSAC for road segmentation, and therefore employed it here as well.

IRLS starts with an estimate of an ideal ground plane derived from the extrinsic calibration parameters of the 3D cameras. We model the plane at time k as

$$\Pi_k = a_k x + b_k y + c_k z + dk = 0. \tag{16}$$

The assumption is that the ground plane orientation changes with time, but that the rate of change is small in comparison to the scan rate of the 3D camera (≈7 Hz). Let $P_k \in \mathbb{R}^{3\times n}$ denote the n points returned from a 3D camera image at time k. The normal distance from point $\mathbf{p}_k = [x_k, y_k, z_k]^T \in P_k$ on the ground plane at time k to the estimate to the ground plane at the previous time Π_{k-1} should be small in practice. This notion is formalized by solving a problem of the form:

$$\begin{aligned}\min_{a,b,c,d} \sum_{i=1}^{n} W(\mathbf{p}_{k_i}, \Pi_{k-1})(ax_{k_i} + by_{k_i} + cz_{k_i} + d)^2 \\ + \sum_{j=1}^{m} W(\mathbf{q}_j, \Pi_{k-1})(ax_j + by_j + cz_j + d)^2,\end{aligned} \tag{17}$$

where $\mathbf{p}_{k_i}$ denotes the i^{th} point of the 3D camera image at time k, $W : \mathbb{R}^3 \times \Pi \rightarrow \mathbb{R}$ is a weighting function of the form:

$$W(\mathbf{p}, \Pi) = \left[\frac{1-\alpha}{1+\beta\ \exp(\gamma\ f(\mathbf{p}, \Pi) - \delta)} + \alpha\right] g(\mathbf{p}), \tag{18}$$

where the first expression is a logistic function of the normal distance from the point $\mathbf{p}$ to the plane Π denoted as $f(\cdot)$ and the second expression is a function $g(\mathbf{p})$ that scales the measurement based on the x component of $\mathbf{p} = [x, y, z]^T$. This is to mitigate the effect of a greater density of points returned at closer ranges. The first term in the minimization problem operates on the points in the current 3D camera scan P_k. The second term is a regularization component that operates on points $Q \in \mathbb{R}^{3\times m}$ that are uniformly sampled from the ideal ground plane.

5.3 Landmark segmentation

SWS landmark segmentation follows the same general procedure described in Sect. 4.2.2, but data from the 3D Hokuyo are used as input. These data have different characteristics than the side-facing LIDAR data of the Mapping Trike. First, the ground plane does not need to be segmented as it is first

seen at approximately 9 m, and we restrict landmark segmentation to ranges of 8 m or less. Second, the vertical angular resolution is larger with the actuated LIDAR, so the clustering procedure uses a maximal intra-cluster distance of 45 cm rather than the 10 cm figure previously stated.

5.4 Localization

Localization of the SWS was performed using a Rao-Blackwellized particle filter approach similar to the FastSLAM 2.0 algorithm (Montemerlo and Thrun 2007). The FastSLAM algorithm is a solution to the simultaneous localization and mapping problem. Since our landmark map was known a priori, we were only concerned with the localization aspect. The particle filter approach was chosen because it was more robust to aspects of the environment that were not modeled by the system, namely the assumption the SWS is operating on a perfect 2D plane.

A set of p particles was maintained where each particle had the form:

$$Y_k^{[p]} = \langle x_k^{[p]}, \langle \mu_1^{[p]}, \Sigma_1^{[p]} \rangle, \ldots, \langle \mu_N^{[p]}, \Sigma_N^{[p]} \rangle \rangle, \tag{19}$$

where $x_k^{[p]}$ was the pose of the pth particle at time k defined as $[x, y, \theta]^T$ where x and y were the Cartesian coordinates and θ was the SWS's orientation with respect to the map's frame. Every particle was initialized with a map containing the mean, $\mu^{[p]}$, and covariance estimate, $\Sigma^{[p]}$, for each landmark in the map. The mean vector had the form $[x_l, y_l, s]^T$, where x_l and y_l were the landmark's position and s was the landmark's signature, which contained additional semantic information to facilitate data association. In our case, the signature value represented the radius of the pole feature. The number of particles used in our implementation was 60 and, for the purposes of planning, the mean of all the pose components was used.

The prediction phase of the filter consisted of sampling from a probabilistic motion model of a differential drive robot where the control inputs (v, ω) were corrupted with additive Gaussian noise (Thrun et al. 2005); two noise parameters were used for error in translational movement due to linear and rotational motion ($a_1 = 0.05$ and $a_2 = 0.01$) as well as two for the error in rotational movement due to linear and rotational motion ($a_3 = 0.001$) and ($a_4 = 0.1$). These parameters were determined empirically under the assumption that a diverse particle set improves localization performance. As such, the values chosen were an exaggeration of the true noise model.

Data association of segmented landmarks was done by using a maximum likelihood correspondence. During the correction phase of the filter, for a given particle p, every observation at time k was compared to each of the landmarks in p's map and a weight was computed for each observation - landmark pair that measured the likelihood that the observation corresponded to the landmark. Observations were of the form: $z = [\rho, \phi, s]^T$, where the components corresponded to the range, bearing, and signature (radius of the pole). The weight was approximated by a Gaussian with mean $(z - \hat{z}_j)$, where z is the measured observation and $\hat{z}_j$ is the predicted observation of particle p's jth landmark, and covariance matrix $Q_j = H_j \Sigma_j^{[p]} H_j^T + Q$, where H_j is the Jacobian taken with respect to the map features, $\Sigma_j^{[p]}$ is the covariance of particle p's jth landmark, and Q is the linearized measurement noise. The weight assigned to each association was the Mahalanobis distance, defined as:

$$w_j = |2\pi Q_j|^{-1/2} \exp\left(-\frac{1}{2}(z - \hat{z}_j)^T Q_j^{-1} (z - \hat{z}_j)\right). \tag{20}$$

The landmark with the maximum weight was chosen to be associated with an observation if it exceeded a minimum threshold. We found that the addition of the signature led to fewer false data associations than using location alone.

5.5 Navigation

The navigation component of the SWS is composed of two main tasks: mapping the immediate local environment from sensor data and planning a path through this local environment. At the user level, navigation only requires two inputs. The first is selecting a destination. The second is resuming the SWS when it pauses at locations where the user must determine when it is safe, such as crossing a street. The latter is necessary, as some street crossings feature high speed automobile traffic that would be detected too slowly with the on-board sensor suite.

5.5.1 Constructing the local map

The SWS employed a local map modeled as a 2D occupancy grid for the purpose of generating local plans. This local map was centered at the position of the SWS and moved with the SWS in a rolling window fashion. We leveraged ROS (Quigley et al. 2009) for populating and clearing cells in the local map via raytracing techniques. For navigation purposes, 3D points from both the filtered 3D camera data streams (Sect. 5.1.1) and the point cloud from the 3D Hokuyo (Sect. 5.1.2) were projected down to the 2D occupancy grid M where each cell in the grid that contained a projected point was classified as occupied. Each occupied cell $M(x, y)$ was given an obstacle cost value $C_{obst}(x, y) = \infty$. Nearby cells were also assigned cost values based on the proximity to occupied cells as well as the footprint of the SWS; if the SWS were to occupy a cell $M(x, y)$ and any portion of its footprint would overlap an obstacle cell where $C_{obst}(x, y) = \infty$, then that cell was also assigned a value of ∞ making it

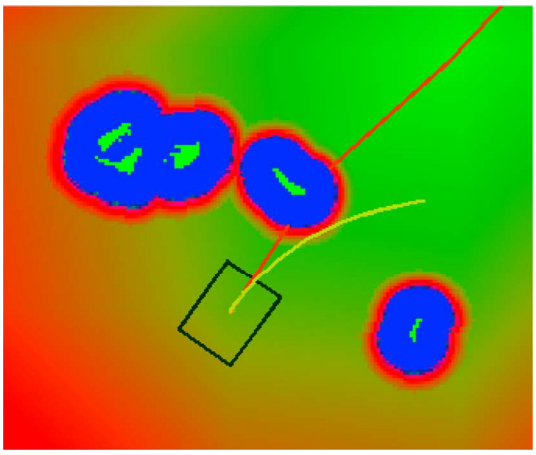


Fig. 12 Navigation visualization. The *black rectangle* is the robot footprint, the *red line* is the desired path, the *yellow line* is the lowest cost trajectory. The *bright green cells* are obstacles of maximum cost. Obstacle cells are inflated with a high cost region in *blue* (Color figure online)

untraversable by the local planner. Otherwise, obstacles were modeled by exponential potential functions.

In addition to the occupancy cost for each cell, the local map also maintained planning related costs for each cell $C_{goal}(x, y)$ and $C_{path}(x, y)$. C_{goal} was proportional to the distance from the current SWS position and the subgoal position G_i (the closest waypoint to the goal that is within the SWS's local map M) which was given a zero cost value. Similarly, C_{path} was proportional to the distance from the SWS's position to the cells along the waypoint path where path cells were determined by linear interpolation between the waypoints and assigned a cost of zero. The resulting occupancy map M and associated costs C_{obst}, C_{goal}, and C_{path} were used by the local planner for trajectory planning. Figure 12 depicts an example of a local map.

5.5.2 Planning

The global planner for the SWS is intuitive; given the waypoint network, $G(V, E)$, described in Sect. 4.4 and a desired destination $v \in V$, the path to the destination is computed via Dijkstra's algorithm. Note that the weight for each edge is the estimated traversal time rather than the distance, so the path returned from Dijkstra's algorithm minimizes the estimated traversal time and does not necessarily correspond to the shortest distance.

A sample based approach was used for local planning where the input space ranges over linear and angular velocities (v, ω) (LaValle 2006). Sampling control velocities in this way ensured that the trajectory honored the kinematics of the SWS. At the beginning of each planning cycle, a set of trajectories of the form:

$$T_i = (\mathbf{x}, v_1, \omega_1, \ldots, v_n, \omega_n), \tag{21}$$

were sampled over the range of velocities $v \in [0.1, 1.2]$ m/s and $\omega \in [-0.3, 0.3]$ rad/s where $\mathbf{x}$ is the pose of the SWS and n denotes the number of discrete time steps in the control horizon. Each sampled trajectory T_i was then evaluated with a cost function of the form:

$$C(T_i, M) = C_{obst} + k_1 C_{goal} + k_2 C_{path}, \tag{22}$$

where M is the occupancy map described in Sect. 5.5.1 and k_1 and k_2 are tunable gain parameters. C_{obst} was the maximum obstacle cost of any cell along the specified trajectory. If $C_{obst} = \infty$, the trajectory was infeasible as it passed through an obstacle and was discarded. The goal and path costs were determined by the endpoint of the trajectory (x', y') and assigned the values $C_{goal}(x', y')$ and $C_{path}(x', y')$. The optimal trajectory $T^* = \text{argmin } C(T, M)$ was selected and the associated velocity command $(v^*, \omega^*) \in T^*$ was issued to the SWS Motion Control Module (MCM).

6 Experimental results

To demonstrate the effectiveness of the two major components of the smart wheelchair ecosystem, namely the server mapping component and the client SWS, a map was constructed following a path starting at the loading dock of Packard Lab and then going across the street and circling a multi-block loop in South Bethlehem, PA. This route (shown at Fig. 13) was chosen for several reasons:

(1) The route was of significant length (1 km) and had diversity in the frequency and signature of landmarks.
(2) The route contained a large loop which allowed us to test loop closure.
(3) The route was not flat. The streets running in the north-south direction had a grade of approximately 8.5 %. This allowed us to test the effectiveness of the 2D localization and mapping approach across significant changes in elevation.

We should emphasize that the density of landmarks played no role in the selection of this route, as pole features appear to be universally pervasive in South Bethlehem.

The following sections describe the mapping results and SWS navigation results.

6.1 Server mapping performance

To test the performance of the server mapping component, we mapped a multi-block area in South Bethlehem, PA shown in Fig. 13. The route is 1 km in length, and features a loop of approximately 944 m. To validate the effectiveness of landmark segmentation by the Mapping Trike, the number of

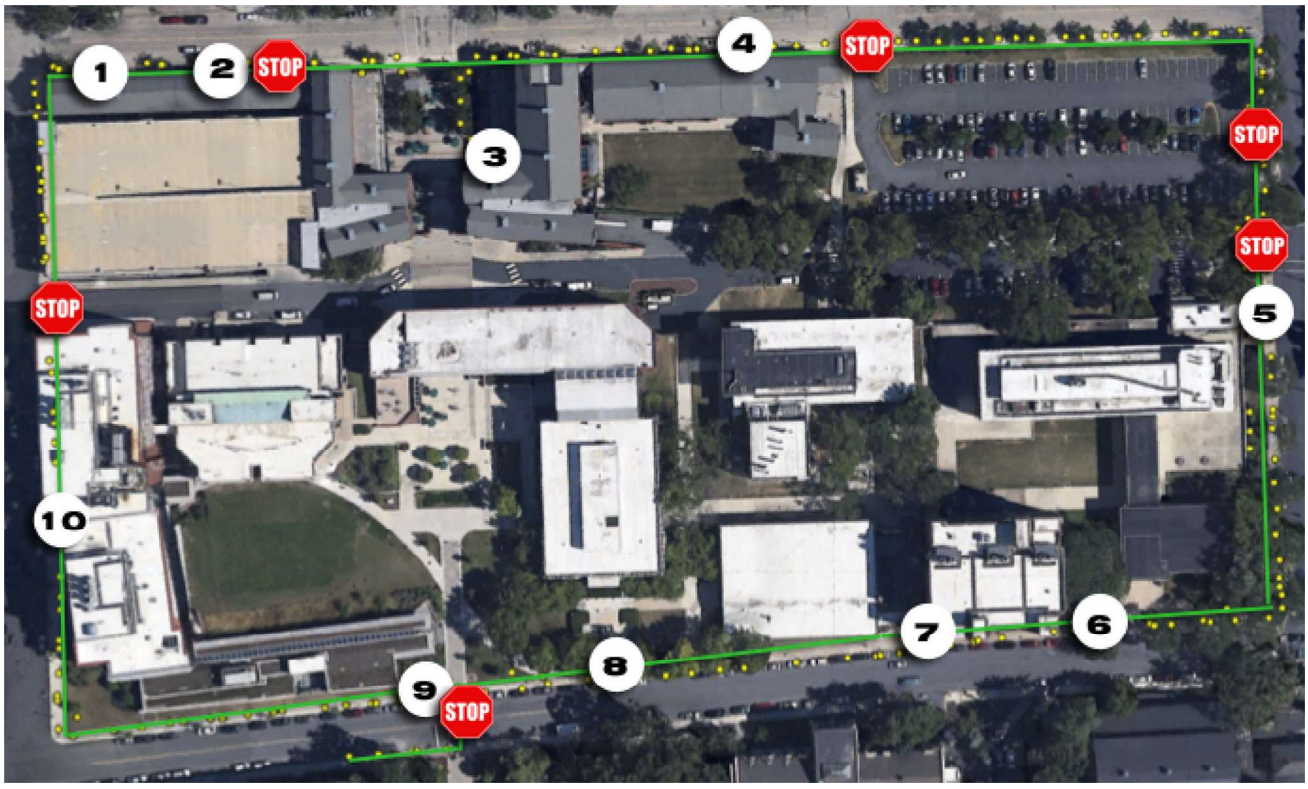


Fig. 13 Satellite view of the block route network. The route is depicted as the *green* path. The *yellow circles* indicate the landmark features. The distance around the loop is approximately 944 m. The six stop sign icons correspond to locations where the SWS automatically stops and waits for the operator to determine when it is safe to cross. The numbered locations correspond to labeled destinations for SWS navigation (Color figure online)

landmarks along this route (187) were counted by hand as a ground truth measure. This was compared against the output of the map initialization and refinement stages. The landmark segmentation procedure in the map initialization step successfully detected 186 of the ground truth poles. The sole missing landmark was a lamp post with a bike chained to it, so its omission was not unexpected. More significantly, there were 32 false positives which were associated with stationary pedestrians and the corners of buildings. The latter occurred when the sliding window only caught the very beginning of a building facade. These false positives could be readily eliminated by not accepting landmarks at the leading edge of the sliding window, as they would reappear in the center of the window at the next time-step and be easily discriminated. Eliminating false positives from pedestrians would be somewhat more complicated, likely requiring a vision system implementing people detection as a validation gate. As this was not available in the current Mapping Trike configuration, these false positives were removed manually.

By comparison, the final landmark map contained 185 landmarks after the refinement stage. The sole discrepancy between this value and the ground truth was due to a false data associations in the EKF SLAM procedure where two poles were sufficiently close to one another (<10 cm) and associated as being the same pole. As a measure of map accuracy, the final landmark map was compared to the satellite image. The coordinates of 24 clearly visible poles were obtained by clicking their positions on the satellite map and the distances to the associated poles in the landmark map were computed. The mean error in distance was 66 cm with a maximum error of 115 cm and a minimum error of 28 cm. These values are only an approximate measure of map accuracy as errors in clicking and satellite imagery are confounding factors.

To validate the generality of the server mapping component, we mapped a second neighboring block depicted in Fig. 14. The reason for this was that the parameters for the mapping procedure (e.g., the pole segmentation parameters) were tuned using data from this same loop. As a result, a test set disjoint from the training set was required. Using the same parameters as the first map, all 129 landmarks in the second map were successfully detected. However, 25 false positives were also detected. In addition to the types detected in the first map, there were also false positives from the supports of windows and glass doors, as well as from the fence posts of a chain link fence. These additional false positives are interesting cases. To elaborate, like the mapping vehicle

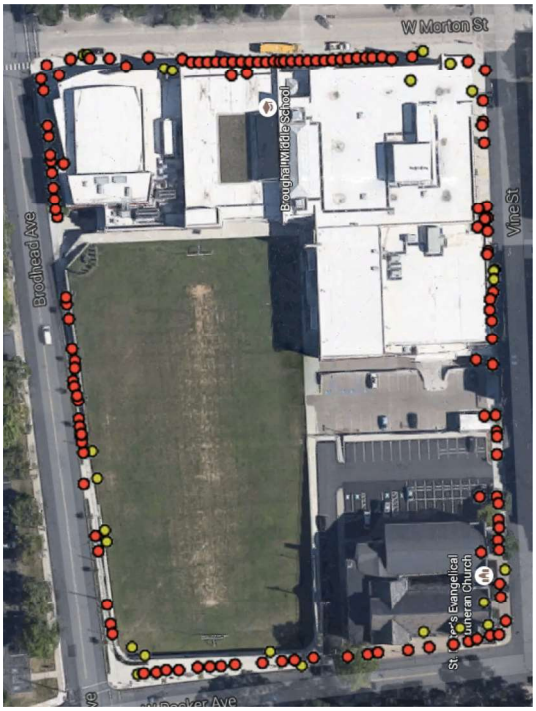


Fig. 14 Satellite view of a second block mapped with the trike using the same parameters as the first block. The *red circles* indicate the landmark features and the *yellow circles* indicate false positives. The distance around the loop is approximately 585 m. Out of 129 manually counted ground truth poles 100 % were successfully segmented. However, 25 false positives were detected (Color figure online)

our SWS segmented landmarks using 3D point cloud data. As a result, window supports or chain link fence posts likely would also be detected as landmarks by the SWS and correctly associated with those in the map. So, an argument could be made for labeling these as true positives and keeping them in the map. However, an alternate sensing modality (e.g., a camera based vision system) likely would not detect these as landmarks. Ultimately, since they did not meet our strict geometric definition for a pole feature, they were categorized as false positives.

We also note that the north-south distance of both loops was approximately 175 m along a grade of 8.5 %. Without compensating for this grade, landmark positions on the north and south ends of the map would be shifted 60–70 cm which would place many in the street. However, a review of the map indicates that these did in fact remain on the sidewalk. This validated our approach to projecting the landmark positions to the UTM frame. Since it is difficult to qualitatively discern the landmark accuracy at the scale of the map in Figs. 13 and 15 depicts a zoomed in view of a section of the map in Fig. 13. This section was chosen for two reasons: the landmarks are visible in the satellite image and it is along the east-west direction and could suffer from the previously mentioned shifting effect.

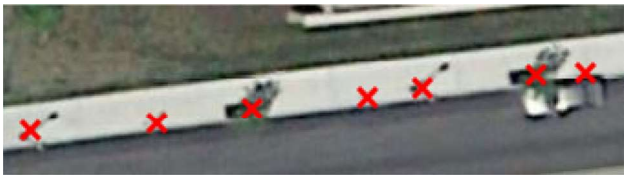


Fig. 15 A close up view of the map in Fig. 13. The landmarks are depicted as *red crosses* (Color figure online)

6.2 SWS localization accuracy

In an attempt to quantify the accuracy of the proposed map-based localization approach, the SWS was manually driven around the loop but with the localization module running. It was then stopped immediately adjacent to a reference landmark, and the normal distance from the base of the SWS to the landmark was measured manually. At the same time, this distance was also captured using the SWS's localized position. This process was repeated for a total of 30 landmarks.

This experiment was conducted a total of three times with landmark maps containing 50, 75, and 100 % of the total landmarks. The motivation for reducing the number of landmarks available to the localization module was to assess the impact of landmark occlusion, e.g., what happens if half the landmarks are occluded by pedestrians? The results of these experiments are summarized in Fig. 16 with box plots for each landmark map. These indicate that sub-decimeter level 1D accuracy could be expected when even just half the landmarks are visible. To place these results in the proper context, we note that the authors in (Baldwin and Newman 2012) evaluated their own approach in a somewhat urban environment against a "high caliber" DGPS/IMU system. They achieved a median path error of ≈0.5 m, while the DGPS/IMU error was ≈1.0 m. While theirs and our test procedures were not entirely consistent, these results support the assertion that our map-based localization approach can achieve significantly better performance than a DGPS/IMU solution in urban environments.

6.3 SWS navigation performance

To demonstrate autonomous navigation of the client SWS, the landmark map and the route network generated from Sect. 6.1 were downloaded by the SWS from the cloud to support the navigation task. Subsequently over the course of two days, the SWS drove autonomously around the loop (a distance of approximately 944 m) five times in each direction. The reason for 10 loops was to demonstrate a reasonable level of reliability. The rationale for both clockwise and counter-clockwise loops was not just scene diversity. We also wanted to investigate the impact (if any) of bias in the mapping procedure, as the map was constructed with the trike driving

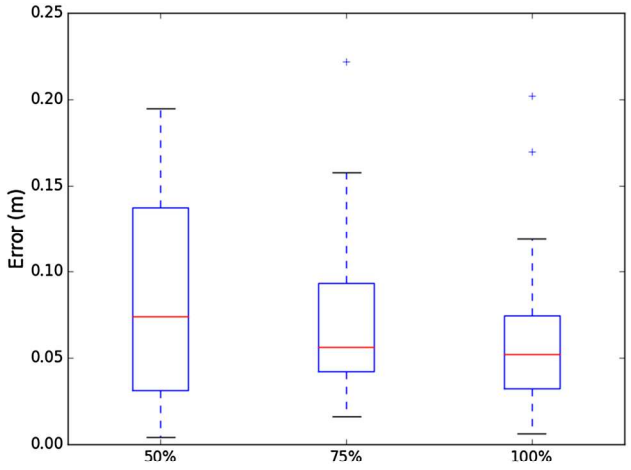


Fig. 16 A comparison of 1d localization error from 30 reference locations. Localization was performed on three landmark maps containing 50, 75, and 100 % of the landmarks. Each respective *box plot* shows the median, 25th, and 75th percentiles, with outliers plotted as individual points

in only one direction. In our test protocol, once initiated the operation of the SWS was to be completely autonomous. The only exception was when groups of pedestrians blocked the sidewalk, where it was manually paused for safety considerations. With lone pedestrians, the SWS was not hindered from performing obstacle avoidance.

Testing was run under winter conditions that were less than ideal. Figure 17 depicts some of these instances. In some areas, frost heave significantly disturbed sidewalk pavers. This caused surface deviations as much as 8 cm in height. Generous use of snow melt (i.e., rock salt) further reduced traction in these areas. Despite these challenges, the SWS motor controller performed well, although there was a noticeable decrease in velocity when traversing up steep grades. Landmarks in the map were also changed in unexpected ways. For example, a sequence of ten parking meters featured in the landmark map were covered by local law enforcement to indicate that parking was temporarily prohibited (Fig. 17). These coverings altered the signatures reported by the 3D Hokuyo which the localization relies upon. This particular stretch has two of the largest dead reckoning lengths (≈13.5 m) where the only landmarks were these parking meters. In spite of these discrepancies, accurate localization was maintained.

In total, the SWS traveled a distance of 10.3 km over an operational time of 222 min. The longest run of continuous operation was for 88 min covering 3.7 km, and was only suspended due to depletion of the laptop battery. No localization failures were observed during testing. However, during one trial manual intervention was required due to a control "failure." Specifically, a large amount of road salt in an area where sidewalk pavers were pushed up significantly from frost heave (similar to Fig. 17) resulted in the SWS drive wheels spinning in place, and the SWS could not overcome the "obstacle" on the first try. This required the operator to manually back up the SWS less than 1 m. A second attempt under autonomous control was successful and the trial continued to the end. We should emphasize that localization was not lost even with the wheel slippage as the 3D Hokuyo was able to observe landmarks during the episode.

Additional images highlighting points of interest during course navigation are shown at Fig. 18. A more informative video from testing can be viewed at http://vader.cse.lehigh.edu/videos/sws_navigation.mpeg.

An interesting question posed by one of the reviewers (for which we are grateful) was the impact on system performance if false positives in the landmark map had not been removed. We had assumed this was necessary, but was this in fact the case? To assess this, logged data from the ten trials was used to localize the SWS using the landmark map with both true and false positives in place. If a localization failure occurred during playback of the log file, the SWS was manually re-localized and the log file was continued from that point. Of the five clockwise trials, three had localization failures. Two suffered from a single failure but completed the loop upon resumption. The third had to be reset twice. Of the five counter-clockwise trials, two had localization failures. One was able to complete the loop after being manually reset, while the second had to be reset twice. Thus, the success rate for this route dropped from 100 to 50 %, motivating the need for an accurate landmark map.

Fig. 17 Changes to the environment not modeled by the system. Winter conditions raised the curb cut out by 8 cm (*left*). A parking meter covered by a bag (*center*). A bike chained to a tree (*right*)

Fig. 18 Photos of the SWS in operation, highlighting interactions with pedestrians (*left*), the narrowness and clutter of certain sidewalk areas (*center*), and automatically pausing at a crosswalk (*right*)

6.4 SWS path planning performance

The results presented in Sect. 6.2 provided insights into the accuracy of the localization system only. Uncertainty in SWS planning and control was deliberately removed from the evaluation. However, we were also interested in quantifying the performance of the planner itself. To do this, we examined the paths of the 10 trials to evaluate their consistency. This was done using the pose estimates provided by the localization module as ground truth. Since these estimates are assumed to be correct, localization uncertainty is removed from the analysis and the variance in path following can be attributed entirely to planner performance.

Figure 19 shows three 20 m sections of the route. The three sections were chosen arbitrarily except to evaluate locations with different densities of landmark features. Each section has ten paths shown (five in each direction), with the landmarks depicted as blue circles and the waypoints as black circles. Note the relative size of a landmark circle reflects its signature (its radius). Also note the radius of each waypoint (50 cm) includes the local planner's tolerance to reach the waypoint.

Qualitatively from the figure, planning performance is very good. To quantify this, we computed the 1D mean absolute error (MAE) and 1D standard deviation of the paths across a range of cross-sectional samples. These values were computed by taking 50 evenly spaced locations along the x axis. For each of these locations, the average normal was computed where it intersected the given x coordinate. Then the cross-sectional line was computed using the x coordinate and the mean y coordinate as a base point, and the average surface normal as the slope. The points of intersection of the paths along the cross-sectional line were then found and transformed to reduce the dimension to one via principal component analysis. The now 1D values were centered so that the mean value was zero. Finally, the data from each of the 50 cross sections were pooled to compute the MAE and standard deviation as it was in a compatible form. The three data sets were evaluated separately. The top figure had a MAE of 1.8 cm and a standard deviation of 2.1 cm. The middle figure had a MAE of 1.7 cm and a standard deviation of 2 cm. The bottom figure had a MAE of 1.9 cm and a standard deviation of 2.4 cm.

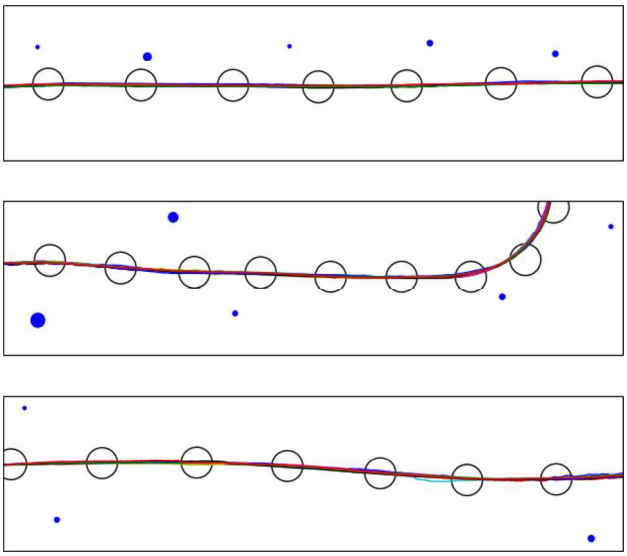


Fig. 19 Close up of ten paths (five in each direction) on 20 m sections of the block with various densities of landmarks. The landmarks are depicted in *blue*. The *black circles* indicate the tolerance for hitting a waypoint and have a radius of 50 cm (Color figure online)

When evaluated in conjunction with the localization performance reported in Sect. 6.2, we would expect the wheelchair to consistently drive the same path to a tolerance of approximately ± 10 cm. This is in fact consistent with our subjective observations made during testing.

6.5 Simulated user testing

The results presented in Sect. 6.3 are limited in the sense that they do not reflect typical user operation. Specifically, it is the same route repeated $5\times$ in two directions. Ideally, user testing with non-confederate participants would be conducted. Due

Table 1 Results of simulated user trials, where each destination was randomly generated

Path	Distance (m)
$2 \rightarrow 5 \rightarrow 8$	572
$4 \rightarrow 10 \rightarrow 7$	943
$6 \rightarrow 5 \rightarrow 8$	385
$9 \rightarrow 10 \rightarrow 6$	463
$2 \rightarrow 1 \rightarrow 4$	199

All trials were successfully completed without incident

to concerns with human-subjects use and institutional review board (IRB) requirements, we chose to simulate such a use.

First, a total of ten destinations of interest were identified in the test loop and a semantic label was attached to each. These are listed below. Note the numbers correspond to the locations highlighted in Fig. 13.

(1) Apartment 20 on Morton Street
(2) Police station on Morton Street
(3) Book store at Campus Square
(4) Apartment 14 on Morton Street
(5) Entrance to Whitaker Lab on Webster Street
(6) Entrance to Whitaker Lab on Packard Avenue
(7) Entrance to Mudd on Packard Avenue
(8) Iaccoca plaza on Packard Avenue
(9) Entrance to STEPS on Packard Avenue
(10) Entrance to STEPS on Vine Street

We then performed five trials using random sampling without replacement. First, a random starting location was chosen. This was followed by a random intermediate goal location, and then a random final goal. Table 1 shows the path locations and total distance traveled for each trial. We should note that these trials were performed during sidewalk construction on Vine Street, so the link between destinations one and ten had to be removed. As a consequence, some of the global plans (e.g, $4 \rightarrow 10 \rightarrow 7$) between locations were longer distance than usual.

All 5 trials were successfully completed without incident. These amounted to a total of 2.6 km of additional autonomous operations.

7 Discussion

In this work, we presented a system level approach to SWS navigation in urban environments. The proposed ecosystem features a mapping service which generates large-scale landmark maps, and the client SWS which integrates 3D perception for robust navigation in unstructured, outdoor environments. In our experiments, the SWS was able to reliably navigate over a distance of >12 km without losing localization. From this, we claim the proposed map-based localization approach can provide the performance of a high-end GPS/INS system, but without the cost.

We are firm believers in the future of map-based localization solutions for urban environments that leverage 3D LIDAR/camera systems. We further note that since the initial draft of this paper was submitted, our 3D Hokuyo was rendered obsolete by the release of the Velodyne VLP-16 "Puck" LIDAR. In the same window, our IFM O3D200 3D cameras were also superseded by the release of the O3D303; the smaller, more accurate, lower-cost successor has 29 times the pixels of the O3D200. These higher performance sensors would only improve the performance of our current system.

There are of course concerns with map-based solutions to the localization problem. Two obvious questions are: (1) what happens if a feature is removed (e.g., a tree is cut down), and (2) what happens if a new feature is added (e.g., a tree is planted). Fortunately, the proposed approach is robust to such small-scale changes. In the former case, the removed feature will not be detected by the SWS and would be handled no differently than when a feature is occluded (i.e., no measurement update). In the latter case, the new feature *will* be detected by the SWS. However, there will be no respective feature in the landmark map. As a result, data association will likely fail so again the result will be a "nop." One could imagine a pathological case where a feature is removed, and a new featured added at approximately (but not exactly) the same location. If the new feature had the same signature as the old one, there is the potential for it to be wrongly associated in the landmark map. However, the error tolerance would have to be comparable to the uncertainty in wheelchair position which we have shown to be quite small. As a result, we expect the impact to localization accuracy in even this pathological case to be small.

Despite being relatively stable, large-scale changes *do* occur in urban areas. We saw this first-hand when the STEPS building was recently constructed at Lehigh University, affecting a roughly 100 m stretch in our map. We also demonstrated in Sect. 5.5 that large-scale errors in the map (i.e., 15 % false positive landmarks) will result in localization failure. While there might be potential for SWS clients to update the map in crowd-sourcing fashion, in all likelihood remapping would be necessary in such an event.

While we were pleased with the navigation performance of the SWS, improvements are needed in the mapping service component. Specifically, false positives in landmark segmentation must be eliminated and an improved localization approach is necessary. However, in fairness these shortcomings are of less concern than if the issues were with the SWS itself. The Mapping Trike was fabricated from hardware on-hand, and within the constraints of our limited budget. In practice, such server vehicles would be viewed as part of the

infrastructure. So, while the price of an SWS needs to be competitive for the consumer, cost constraints would be less of a concern for the mapping vehicle. As a result, we expect shortcomings in mapping performance could be readily addressed with improvements in hardware and/or software.

There may also be questions regarding the right choice of landmark map and/or robot belief. Is a metric-based approach correct, or even necessary? Urban environments would seem to lend themselves to topological maps. In truth, we believe an ideal approach would integrate parts of both approaches. We have done preliminary work in terrain classification to automatically distinguish between sidewalk, asphalt, grass, and "other" terrain classes using remission measurements from the O3D200 3D cameras (Montella et al. 2012). It would also be straightforward to embed this additional semantic information within the edges of our route network graph. We expect this additional terrain information information will improve navigation performance over geometric information alone, and is a topic of ongoing research.

Furthermore, our initial metric-based map assumption was the impetus for utilizing satellite images to find known correspondences for the mapping procedure as these are anchored to a global coordinate frame. If we only need maps that are locally consistent, then the problem of closing the loop in our map could be automated using existing techniques. Again, both approaches have merit.

We also must acknowledge that since the focus of this work has been on SWS navigation, only limited work has been done on the human–robot interface (HRI). To date, we have investigated the use of an Emotiv EPOC brain control interface as an input device. We had hoped to identify a minimum of three inputs (turn left, turn right, start/stop toggle) to cue the SWS to the desired travel direction for semi-autonomous operation. However, to date we have only been successful in having a single input work reliably. We demonstrated this functionality during autonomous navigation as a start/stop toggle for E-Stop, and for resuming SWS operation after encountering a stop sign in the map. In part based upon this experience, we believe that at least in the short term the SWS interface should be voice-based. The quality and reliability of voice recognition systems has improved dramatically in the past decade, and we feel this would provide the best hand-free solution for the largest number of users. We are currently working to validate this hypothesis.

Finally, one significant aspect of urban environments that was ignored in this work was navigation in crowds. This is a rich research area in its own right, and one we intend to investigate with zeal in the future. One insight we are happy to report is that on the whole, pedestrians are largely considerate of the SWS and give it a wide berth. We discovered through many hours of testing that despite its sensor "warts," the SWS is perceived merely as a person operating a conventional EPW and not as a smart wheelchair system. We believe this "disguise" will be an asset when navigating crowded environments.

References

Anguelov, D., Dulong, C., Filip, D., Frueh, C., Lafon, S., Lyon, R., et al. (2010). Google street view: Capturing the world at street level. *Computer*, *6*, 32–38.

Baldwin, I., & Newman, P. (2012). Road vehicle localization with 2d push-broom lidar and 3d priors. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2611–2617).

Bohren, J., Foote, T., Keller, J., Kushleyev, A., Lee, D., Stewart, A., et al. (2008). Little ben: The ben franklin racing team's entry in the 2007 DARPA urban challenge. *Journal of Field Robotics*, *25*(9), 598–614.

Bostelman, R., & Albus, J. (2008). Sensor experiments to facilitate robot use in assistive environments. In *Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments* (p. 8) ACM

Chong, Z., Qin, B., Bandyopadhyay, T., Ang, M.H., Frazzoli, E., & Rus, D. (2013). Synthetic 2d lidar for precise vehicle localization in 3d urban environment. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 1554–1559).

Cooper, R., Grindle, G., Vazquez, J., Xu, J., Wang, H., Candiotti, J., et al. (2012). Personal mobility and manipulation appliance design, development, and initial testing. *Proceedings of the IEEE*, *100*(8), 2505–2511.

Doshi, F., & Roy, N. (2007). Efficient model learning for dialog management. In: *2nd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2007 (pp. 65–72).

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395.

Gao, C., Hoffman, I., Panzarella, T., & Spletzer, J. (2007). Automated transport and retrieval system (atrs): A technology solution to automobility for wheelchair users. In *6th Conference on Field and Service Robotics* (FSR07)

Gao, C., Miller, T., Spletzer, J. R., Hoffman, I., & Panzarella, T. (2008). Autonomous docking of a smart wheelchair for the automated transport and retrieval system (ATRS). *Journal of Field Robotics*, *25*(4–5), 203–222.

Gao, C., Sands, M., & Spletzer, J.R. (2010). Towards autonomous wheelchair systems in urban environments. In *Field and Service Robotics* (pp. 13–23). Springer

Irie, K., & Tomono, M. (2012). Localization and road boundary recognition in urban environments using digital street maps. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 4493–4499).

Kummerle, R., Ruhnke, M., Steder, B., Stachniss, C., & Burgard, W. (2013). A navigation system for robots operating in crowded urban environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2013 (pp. 3225–3232)

Lategahn, H., Schreiber, M., Ziegler, J., & Stiller, C. (2013). Urban localization with camera and inertial measurement unit. In *IEEE Intelligent Vehicles Symposium (IV)* (pp. 719–724).

LaValle, S. M. (2006). *Planning algorithms*. Cambridge: Cambridge University Press.

Montella, C., Perkins, T., Spletzer, J., & Sands, M. (2014). To the bookstore! autonomous wheelchair navigation in an urban environment. In *Field and Service Robotics* (pp. 249–263). Springer

Montella, C., Pollock, M., Schwesinger, D., & Spletzer, J.R. (2012). Stochastic classification of urban terrain for smart wheelchair

navigation. In *Proceedings of the IROS Workshop on Progress, Challenges and Future Perspectives in Navigation and Manipulation Assistance for Robotic Wheelchairs*

Montemerlo, M., & Thrun, S. (2007). *Fastslam 2.0. FastSLAM: A scalable method for the simultaneous localization and mapping problem in robotics* (pp. 63–90). Berlin: Springer.

Perry, D. (2002). *Medical never-never land: Ten reasons why america is not ready for the coming age boom* (p. 1). Washington, DC: Alliance for Aging Research.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A.Y. (2009). Ros: an open-source robot operating system. In *ICRA workshop on open source software*, vol. 3 (p. 5)

Rusu, R.B., & Cousins, S. (2011). 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 1–4).

Simonite, T. (2013). Data shows Googles robot cars are smoother, safer drivers than you or I. In *Technology Review*, Vol. 25.

Simpson, R. C. (2005). Smart wheelchairs: A literature review. *Journal of Rehabilitation Research and Development*, *42*(4), 423.

Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. Cambridge: MIT press.

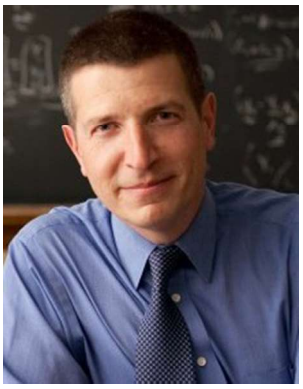
United Nations Department of Economic and Social Affairs: World Urbanization Prospects: The 2014 Revision. United Nations Publications (2015).

Yokozuka, M., Suzuki, Y., Hashimoto, N., Matsumoto, O.: Robotic wheelchair with autonomous traveling capability for transportation assistance in an urban environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 2234–2241).

Dylan Schwesinger is a PhD student working in the field of mobile robotics under Professor John Spletzer at Lehigh University's VADER Laboratory. His primary research interests are 3D perception, planning, and control. He receive both his Bachelor's degree and Master's degree in Computer Science from Kutztown University.

Armon Shariati is a PhD student working under Professor Camillo Taylor at the University of Pennsylvania's GRASP Laboratory. His research interests include robot mapping and computer vision. He received his Bachelor's degree in Computer Engineering from Lehigh University. As an undergraduate he worked as a research assistant in the VADER laboratory under the guidance of Professor John Spletzer.

Corey Montella is a PhD student at Lehigh University. He uses statistical and machine learning techniques to build robots that can reliably operate in complex and uncertain conditions. Previous projects include a robotic wheelchair that can navigate outdoor urban environments autonomously and a gliding autonomous aircraft that maps windfields and harvests energy to soar indefinitely.



John Spletzer is an Associate Professor in the Computer Science and Engineering Department at Lehigh University, where he heads up the VADER Laboratory. His research interests include intelligent vehicle systems, assistive technologies, and multi-robot systems. He teaches courses in mobile robotics and real-time image processing. He also serves as the instructor for both the Computer Science and Computer Engineering senior design courses. His research is funded by the National Science Foundation, the Pennsylvania Department of Community and Economic Development, and a range of private corporations.