2011

Pedestrian Tracking With A Low-Cost 3D LIDAR System On A Mobile Platform

Constantin Savtchenko
Lehigh University

PEDESTRIAN TRACKING WITH A LOW-COST 3D LIDAR SYSTEM ON A MOBILE PLATFORM

BY

CONSTANTIN SAVTCHENKO

A THESIS
PRESENTED TO THE GRADUATE AND RESEARCH COMMITTEE
OF LEHIGH UNIVERSITY
IN CANDIDACY FOR DEGREE OF
MASTER OF SCIENCE

IN

COMPUTER SCIENCE

LEHIGH UNIVERSITY

MAY 2011

THIS THESIS IS ACCEPTED AND APPROVED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE MASTER OF SCIENCE.

DATE

THESIS ADVISOR

CHAIRPERSON OF DEPARTMENT

Contents

List of Figures

Abstract

Three-dimensional (3D) LIDAR systems are becoming the sensor of choice for many mobile robotics applications. This can be attributed to their accuracy, robustness, and strong invariance to ambient illumination levels. In many applications, adoption is only limited by the high system cost. In this work, we investigate the use of low-cost 3D LIDAR systems in a people tracking application for a smart wheelchair system. The limited spatial resolution of these systems proved challenging in the people tracking task. To solve this problem, we employed a k-Nearest-Neighbor (k-NN) appearance classifier in conjunction with an extended Kalman filter (EKF)-based motion classifier. Preliminary experimental results indicated a successful tracking rate of over 95%.

1 INTRODUCTION

Mobile robotics has been gaining steady momentum for many years now. The year 2010 is considered by some to be an influential year for the mobile robotics field [1]. Educational robots such as iRobot's Create platform have become ubiquitous at universities, allowing students to gain experience in the field of robotics. The author of this paper began his career in robotics from one of these classes. Inspirations from robotic and pattern recognition courses culminated into enabling pedestrian detection on an autonomous robotic platform.

1.1 Motivation

The rise of three-dimensional (3D) Light Detection and Ranging (LIDAR) systems has generated many new research possibilities in the field of robotics. The case for 3D LIDARs was made convincingly in the 2007 DARPA Urban Challenge. Only 6 vehicles (of 89 original entries) completed the race; all 6 relied upon (then) new 3D LIDAR systems [2]. As LIDARs increase in popularity, their capabilities improve as their prices drop. However, LIDARs are not the only 3D systems on the market. Other low-cost 3D sensors, like Microsoft's Kinect, enable a range of indoor applications at a previously unheard of price point [3]. There is a rising tendency for autonomous robotics systems to move towards 3D data due to the many advantages it offers (Section 2).

All 3D sensors measure data in a similar format, a cloud of points in Euclidean space. Research into effectively using this data has only recently begun, offering a wide variety of new research topics. An area of focus is the successful navigation of autonomous robots through physical environments (indoors or outdoors.) A subset of this problem is the capability to track pedestrians. Urban areas and sidewalks in particular are often congested with pedestrian traffic making navigation difficult. As a result, there is significant interest in reliable detection and tracking of persons in a variety of environmental conditions. While much work has focused upon 3D LIDARs for use on automobiles, little has been done with field and service robots operating among pedestrians. This is not unexpected, as the high cost of 3D LIDARs makes them not economically viable for many small personal applications. However, the recent release of low-cost, low-resolution 3D flash LIDARs offers the potential for robust 3D perception in various environments.

1.2 Background

This work will focus on the use of a 3D LIDAR system attached to an autonomous wheelchair to determine obstacles (pedestrians) in its path. A brief introduction to LIDAR systems and pattern classification is provided to define terms in this field and give information relevant to this work.

A LIDAR uses the concept of time of flight to determine distances. Given the speed of light, the distance of a point can be determined by measuring the time it takes for light to return from the object. A laser range finder used by contractors to measure distances is an example of a LIDAR, typically one distance is returned. A 2D LIDAR takes advantages of the fact that the speed of light is fast enough for the sensor to rotate a laser along a plane and record a multitude of points in a relatively short period of time (*e.g.*, 1/60th of a second.) The pool of points is then returned as “one” measurement or *scan*. A subset of 3D LIDARs apply this concept in three dimensions using off-axis rotation with a group of lasers [4].

Another subset of 3D LIDARs activates an array of lasers in unison to take a *scan*, and returns the distance taken by each laser. These LIDARs are known as flash LIDARs [5] (analogous a digital camera returning distances x, y, z in Euclidean space instead of RGB.)

In this work, data returned by the LIDAR was examined for pedestrians. This task is known as pattern matching or classification. The first step in pattern matching is determining the organization of the data; a sensor returns many points which have no contextual information. These points are grouped and related together using a clustering or labelling algorithm. The clusters of points are then passed to an algorithm that makes a decision on the group of points as a whole, assigning the cluster of points to a *class* (*e.g.*, car, pole, person, tree.) *Features* are meaningful values that describe a property of the data or show tendencies. They are extracted from grouped data points. For example, it might be of interest to determine the distance between the two farthest points in a scan. This distance represents the width of the cluster, coupled with the assumption that a pedestrian will not have a width greater than 1 meter, a segmenting feature has been generated. Determining what useful features to extract from point clouds is an essential research area of pattern matching. In essence, pattern matching is determining what features separate one *class* of objects from all other classes of objects.

Upon classification as a person, the object was tracked. Tracking an object is the act of correlating the location of the object between *scans*. This allowed the platform in this work to determine unique objects and predict their motion given a few scans. By predicting motion, an autonomous platform can consider the path of the pedestrian and generate a navigational plan that is convenient and safe for pedestrian and robot. Tracking is key for safe and efficient operation in crowded pedestrian environments. For example, if an autonomous robot determines a pedestrian walking towards it, slightly to its left, it should:

1. Continue normal operation, *e.g.*, not stop.
2. Change its navigational plan to slightly veer right.

1.3 Objective

The objective of this work is to investigate the suitability of coarse low-cost 3D LIDAR systems for real-time people tracking on a mobile platform. The results have potentially broad applications to field and service robotics, in cases where human-robot interactions are of necessity.

2 RELATED WORK

There has been extensive work using camera systems and image analysis algorithms for pedestrian tracking, a survey of which by Geronimo *et al.* can be found in [6]. The paper identifies limitations of camera-based pedestrian tracking, many of which can be resolved with 3D data. Two difficulties with camera-based tracking are ground plane estimation and region of interest extraction. Ground plane estimation is the determination of the pixels signifying the ground in an image. Removal of these pixels, leaves only pixels that represents objects of interest in the image. Region of interest extraction is the determination of the pixels in an image that a describe a unique object. Since an image is simply many colored points, there is no underlying data or contextual information that relates points. The relation between points needs to be determined, often a computationally intensive problem. It should be noted that finding regions of interest is simpler when the ground has been removed from an image. It is common for image analysis procedures to build upon each other in such a manner.

There has also been recent work with LIDAR-based approaches. Spinello and Siegwart used a Velodyne 3D LIDAR to investigate pedestrian tracking in parts [7]. Their approach split a pedestrian into layers, and each layer voted to determine if the object is a pedestrian. Their approach is powerful in that they ignore the ground plane (*e.g.*, when dealing with multi-level ground planes, sidewalks to roads, *etc.*) and that it can leverage techniques developed and expanded upon in 2D LIDAR research. Douillard *et al.* combined a 3D lidar with a camera for urban classification [8]. Objects such as trees, cars, and pedestrians were classified using a rule-based system approach.

Navarro-Serment *et al.* used a combination of multiple 2D and 3D scanners [9]. Point clouds gathered by the 3D scanner were projected into 2D after ground plane removal. Classification was then done using thresholds set for an object's geometry, velocity, and distance. Tracking was done using a simple rule, if the object was inside the bounding box of an object from the previous frame, they were the same object. In later work [10], a similar two-dimensional approach with two linear Support Vector Machines (SVMs) was used. Linear SVMs are one of many approaches to pattern classification. Both papers use a Geometric Score (GS) and Motion Score (MS) to make classifications.

Prokhorov used a Recurrent Neural Network (RNN), yet another approach to pattern classification, to achieve a high vehicle detection rate [11]. His approach relied on the RNN to do the segmentation based on the temporal order of points received from a Velodyne LIDAR. Object detection was high with a low number of false detections. This paper stands out in its unique classification and segmentation method.

Compared to these works, the approach presented here differed in several ways. First and foremost, the focus was on people tracking in crowded environments using low-cost 3D LIDARs. Low-cost LIDARs achieve price reductions by lowering the resolution (the number of points returned per scan) and the frequency of scans. The limited resolution of such LIDARs introduced significant challenges to the classification process, and the low frequency posed challenges to the tracking process. To achieve consistent people tracking in crowded environments, a k-Nearest-Neighbor (k-NN) appearance classifier was employed in conjunction with an extended Kalman filter (EKF) motion based classifier. The former proved to be advantageous in dealing with imperfect data, while the latter served to improve tracking frequency and reduce classification of inanimate objects as pedestrians. Details on the approach now follow.

3 TECHNICAL APPROACH

3.1 Development Platform

The development platform for this project was based upon the smart wheelchair system (SWS) developed for the Automated Transport and Retrieval System (ATRS) shown at Figure 1 [12]. Odometry measurements are provided by high-resolution quadrature encoders (8,192 Counts Per Revolution). For exteroceptive sensing, the SWS integrates an IFM Effector O3D200 flash LIDAR. The IFM provides 3D measurements of the environment at reasonable cost ($<$\$1,500US). The tradeoff for the low price point is fairly coarse resolution (64×50) over a field-of-view of $40^\circ \times 30^\circ$, respectively.

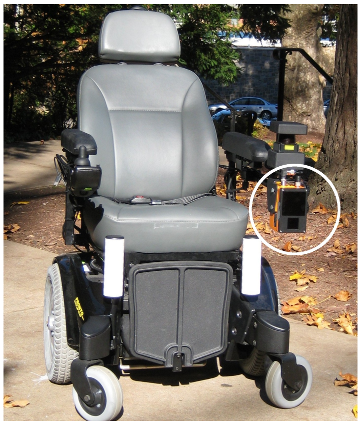

Figure 1: Prototype smart wheelchair system integrating the the flash LIDAR (encircled white).

3.2 Object Segmentation

Fundamental to the approach was the ability to easily segment objects of interest (*i.e.*, persons vs. non-persons) from the background scene. Camera-based techniques struggle with this step as their data is in color space. The advantage of 3D information is apparent here; objects cannot occupy the same physical space. Thus, it can be assumed that any clusters of points are a unique object.

The first step in determining clusters was to establish an estimate for the relative orientation of the local ground plane. This served as a reference elevation when segmenting objects of interest. Since all objects of interest (persons) will be on the ground, removing the ground plane separated the clusters by their inherent location in Euclidean space. A ground plane at time k was described using 3 parameters a, b, c and the plane equation

$$a_k x + b_k y + c_k z + d = 0 \tag{1}$$

Note that d can easily be solved for if necessarry. Taking inspiration from [13], an iterative re-weighted least squares (IRLS) approach was employed to fit a ground plane to the points taken by the flash LIDAR.

The strength of the IRLS formulation when compared to more traditional ground plane tracking approaches (*e.g.*, RANSAC [14]) is that it integrated all *a priori* knowledge of the ground plane orientation through temporal filtering and regularization. To illustrate this, let

$$\Pi_{k-1} = a_{k-1}x + b_{k-1}y + c_{k-1}z + d_{k-1} = 0 \tag{2}$$

denote our estimate for the ground plane at time $k-1$, and $P_k \in \mathbb{R}^{3\times m}$ denote the m points recovered from the LIDAR at time k. It was assumed that the ground plane orientation changes with time, but slowly when compared to the scan rate of the LIDAR. In other words, if a point $\vec{p}_k = (x_k, y_k, z_k) \in P_k$ were on the ground plane at time k, then its distance from Π_{k-1} should be small in practice. IRLS exploits this constraint by solving a problem of the form

$$\min_{a,b,c,d} \sum_{i=1}^{m} W(\vec{p}_{k_i}, \Pi_{k-1})(ax_{k_i} + by_{k_i} + cz_{k_i} + d)^2 \tag{3}$$

where $\vec{p}_{k_i}$ denotes the i^{th} of the m points from the LIDAR scan at time k, and W :

$\mathbb{R}^3 \times \Pi \to \mathbb{R}$ is a weight function based upon the normal distance of a point in the LIDAR scan to the estimated ground plane. In this work, W was a logistic function of the form

$$W(x) = \frac{1}{1 + A * e^{B(x-C)}} + D \tag{4}$$

It was empirically determined that values of $A = 0.9, B = 15, C = 0.1$, and $D = 0.02$ worked well in practice. The resulting weight function is illustrated at Figure 2. The x-axis represents the distance (in meters) from the previous ground plane, and the y-axis is the weight assigned to the point. The logistics function was chosen because it has an upper and lower bound, which are easily adjusted through the parameters. Note that points close to the previous ground plane are given a high weight, and that the weight quickly diminishes.

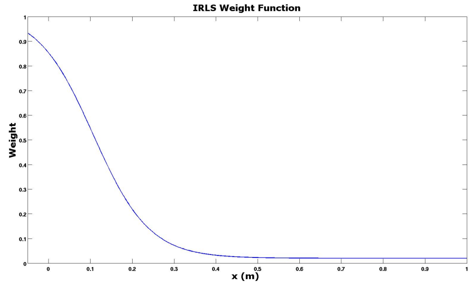

Figure 2: Weighting function used for the IRLS algorithm.

In practice, the IRLS algorithm input was constrained to use LIDAR points within a range of 5 meters. This area was densely populated with points that described the ground plane. To mitigate the effect of a greater density of points being returned from closer ranges, the input was binned into 10 cm × 10 cm cells in the LIDAR's x-z (horizontal) plane, and the point with minimum y (vertical) value was used. The choice of minimum was done to further reduce the impact of large vertical obstacles (*e.g.*, walls). Finally, a set of seeds corresponding to points on the default ground plane was also added to (3) to act as a regularization component. If the LIDAR's view of the

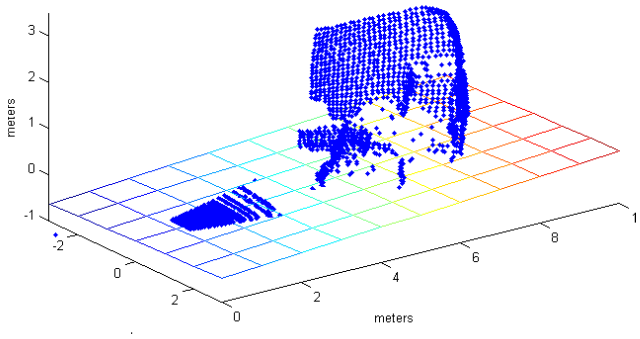

Figure 3: Ground plane tracking with the IFM. The inclined ground plane is clearly tracked.

ground plane was occluded, the regularization component would be enough to make an informed guess at the location of the ground plane. The amount of seeds had to be determined carefully. Too many seeds would prevent the correct convergence of the ground plane; while, too little seeds would not offer any regularization. It was empirically determined that a 5×7 mesh of 35 points produced good results in practice. Representative results from this process are illustrated at Figure 3, which shows the raw 3D scan from the IFM, along with the recovered ground plane.

Once Π was estimated, all points were rotated such that the new ground plane was the vertical vector $\vec{j}$. All points within a distance threshold of 5 cm ($y \leq 5$) were eliminated from the scan. The remaining points within 10 meters of the LIDAR were then converted to a $64 \times 50 \times 100$ voxel image where the x-z plane was relative to Π (as per our rotation). The voxel image was then pre-processed using connected component labeling to identify the set $\mathcal{C} = \{C_1, \ldots, C_n\}$ of objects of interest. A connected component algorithm determines which points are connected, or considered one cluster. This can be achieved in a variety of ways. This work used 26-connectivity and an algorithm described in [15]. The scan, separated into clusters, was ready to be classified.

3.3 An Appearance-based Classifier

The appearance-based classifier assigned an object of interest to one of two classes: person $\mathcal{P}$ or non-person $\bar{\mathcal{P}}$. To accomplish this, each connected component $C_k \in \mathcal{C}$ recovered from the segmentation approach described in Section 3.2 was first transformed into a compact six-dimensional feature vector: $\mathbf{x}_k = [h_k, w_k, d_k, v_k, \lambda_k, \rho_k]^T$. The first four elements correspond to the maximum y-value (height), width, depth, and volume of the bounding box of C_k, respectively. Here λ denotes the percentage of pixels above mid-height, to model the tendency of people to be top heavy. Finally, ρ represents the density of the x-y projection of C_k, which is defined as the number of pixels in the projection divided by the total number of pixels in the associated bounding box. This process is illustrated at Figure 4. Discussion on these features can be found in Section 5.1.

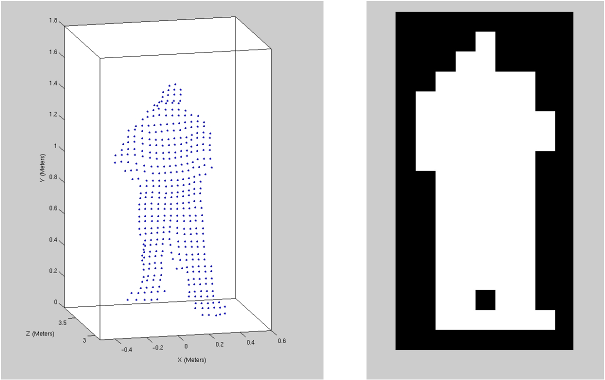

Figure 4: Three-dimensional point cloud (left) and its associated x-y projection in the voxelized image. Object density ρ is defined as the number of pixels in the projection divided by the number of pixels in its bounding box.

The objective in this work was the discovery of observations that are elements of the person class $\mathcal{P}$. Note that $\mathcal{P}$ is a multi-modal distribution: persons can be viewed from a variety of orientations, can exhibit different degrees of motion, can be partially occluded, certain clothes can affect sensor performance, *etc*. As such, simple thresholding was not sufficiently robust for the required accuracy. The application of the k-Nearest Neighbors (k-NN) classifier [16] was investigated. k-NN compares an observed vector $\mathbf{x}'$ to a set of labeled prototype vectors $\mathbf{X} = \{\mathbf{x_1}, \ldots, \mathbf{x_n}\}$ that are known *a priori*, and assigns $\mathbf{x}'$ to the class that occurs most frequently among its k closest neighbors in $\mathbf{X}$. k-NN classifiers excel at classifying multi-modal distributions, which would typically require non-linear thresholding functions to segment.

k-NN requires supervised learning to estimate these difficult non-linear thresholding functions, which in the context of the project amounted to establishing the $\mathcal{P}$ and $\bar{\mathcal{P}}$ classes. To this end, a total of 200 persons were imaged by the LIDAR from a range of distances and relative orientations while driving the wheelchair through South Bethlehem. Additionally, a total of 350 non-person prototypes were also imaged to represent the non-person class. Note that $|\bar{\mathcal{P}}| > |\mathcal{P}|$ to reflect the greater diversity of non-person objects. Each image was then synthesized to a 6-D feature vector as outlined above.

k-NN classifiers require a distance metric for defining “nearness.” Both Minkowski and Mahalanobis distance metrics were investigated. For the latter, the covariance matrices (Σ) formed by the separate classes, the pooled classes, and combinations thereof were examined. Ultimately, the Mahalanobis metric was chosen

$$d(\mathbf{x} - \mathbf{x}')^2 = \left(\mathbf{x} - \mathbf{x}'\right)^T \Sigma^{-1} \left(\mathbf{x} - \mathbf{x}'\right) \tag{5}$$

where $\Sigma = (\Sigma_{\mathcal{P}} + \Sigma_{\bar{\mathcal{P}}})/2$ was the average covariance of the two classes. This modeled equal contributions from each of the two classes, and provided the best empirical results of the alternatives that were considered. Further discussion of the distance metric can be found in Section 5.2.

Traditional k-NN classifiers assign a unit vote to each prototype in the k nearest neighbor set. This work instead employed a weighted voting scheme where each of the k nearest prototypes received a number of votes equal to the squared distance to the test sample, *i.e.*,

$$W_{\mathcal{P}} = \sum_{\mathbf{x} \in \mathcal{P}} d(\mathbf{x} - \mathbf{x}')^2, \qquad W_{\bar{\mathcal{P}}} = \sum_{\mathbf{x} \in \bar{\mathcal{P}}} d(\mathbf{x} - \mathbf{x}')^2 \tag{6}$$

The test point $\mathbf{x}'$ was then assigned to the class with the greatest weight. The ratio of these weights $Q(\mathbf{x}') = W_{\mathcal{P}}/(W_{\mathcal{P}} + W_{\bar{\mathcal{P}}}) \in [0, 1]$ was also used as a quality metric, and provided a confidence measure with respect to class assignment. $Q = 1$ indicated that all k neighbors $\in \mathcal{P}$, while $Q = 0$ is indicative of all k neighbors being from $\bar{\mathcal{P}}$. The use of Q is discussed in more detail in Section 3.5.

3.4 Tracking With An Extended Kalman Filter

Initial testing with the appearance-based classifier indicated it was capable of identifying people with a high probability. However, the frequencies of both false negatives and false positives were not insignificant. Expanding the k-NN prototype database with the misclassificiations lowered the *false positives* but also lowered the classifications of badly represented pedestrians. An example can be seen at Figure 5, where one floating torso is successfully classified and the other is not.

To improve classifications of fragmented scans, while keeping the *false positive* rate low, a second-stage motion-based classifier (MC) was added to the person detection process. As input, the MC took the set of objects $\mathbf{X}' = \{\mathbf{x}'_1, \ldots, \mathbf{x}'_j\}$ output by the appearance classifier, along with their associated confidence scores $\mathbf{Q} = \{Q(\mathbf{x}'_1), \ldots, Q(\mathbf{x}'_j)\}$. The objects in the set $\mathbf{X}'$ were rotated and translated to the world frame in the x-z plane (height and width). The rotation matrix was constructed from the encoders on the Smart Wheelchair System.

The objective of the MC was to associate tracks with acandidate person. For each $\mathbf{x}'(k) \in \mathbf{X}'(k)$, if $Q(\mathbf{x}'(k)) \geq Q_{\mathcal{P}} \implies \mathbf{x}'(k) \in \mathcal{P}$, and the MC immediately associated a track $\mathbf{t}(k) \in \mathbf{T}$ with the person $\mathbf{x}(k)$. For $Q(\mathbf{x}'(k)) < Q_{\mathcal{P}} \implies \mathbf{x}'(k) \in$

$\bar{\mathcal{P}}$, and the non-person object is subsequently ignored. Given two frames, at time k and k+1, a track was generated if an object was related using a simple threshold on the object's velocities between the two frames. The track contained an object's v and θ determined from the finite differences of the two frames.

Successful application of this model enabled the prediction of the location of an object given its velocity at the beginning of every frame.

$$x_{k+1} = x_k + v_k \cos\theta_k \Delta t y_{k+1} = y_k + v_k \sin\theta_k \Delta t \tag{7}$$

An object in frame $k + 1$ was associated with an object from k using a simple distance threshold. Upon successful relation, a new velocity, v_{k+1} was calculated and the old velocity v_k was discarded. This basic model worked well as a proof-of-concept, increasing our *true positive* rate.

Examination of the velocities, showed high variance between frames. One frame a person may have a velocity of 1.8 m/s and in the next 0.9 m/s. This variance is expected due to the strides of pedestrian and inaccuracies in the predictions and measurements. It was apparent that smoothing and filtering of noise was necessary to achieve accurate results. A Kalman filter fit this requirement perfectly. The MC evolved from using a Kalman filter to an extended Kalman filter. Details on the motivation of this evolution now follow.

3.4.1 Kalman Filtering

Kalman filtering is a mathematical method to estimate the state of a process observed over time in such a manner that it minimizes the squared error. The Kalman filter produces values closer to the true values of measurements by analyzing the uncertainty of predictions and measurements. These uncertainties are factored in when making the final decision on the values of the current state. In other words, the Kalman filter assumes that predictions are informed, and that sensors are not 100% accurate. Combining prediction and measurement increases the accuracy of the state description. A quick review of the Kalman filter process is presented, but the reader is encouraged to read [17] for a full review of the Kalman filtering process.

A Kalman filter estimates the state of a process described by a discrete *linear* stochastic equation. There are two stages, a Time Update (prediction) phase and a Measurement Update (correction) phase. In this work, the Kalman filter performed the Time Update phase in the beginning of the frame analysis, and the Measurement Update phase at the end of the frame analysis. Examination of each phase now follows.

During the Time Update phase, the Kalman filter made a prediction for the state vector in the current frame, k+1, using the state vector from the previous frame, k. Initially, the state vector in this work was described using $[x, y, \dot{x}, \dot{y}]^T$. This state differed from the state presented in the initial MC model. The state update equations were

$$\begin{aligned} x^-_{k+1} &= x_k + \dot{x_k}\Delta t \\ y^-_{k+1} &= y_k + \dot{y_k}\Delta t \\ \dot{x}^-_{k+1} &= \dot{x_k} \\ \dot{y}^-_{k+1} &= \dot{y_k} \end{aligned} \tag{8}$$

Where x, y were the coordinates of the object being track, and $\dot{x}$, $\dot{y}$ were the corresponding velocities. Δt was the time between frame k and frame k+1; it should now be apparent why the Time Update phase was done in the beginning of frame k+1. Note the presence of a “minus” sign by the variables, this denotes a prediction. These state equations were chosen due to their linear nature, simplifying the tracking calculations.

In addition to predicting the state, the Time Update phase must project the uncertainty of the prediction, $\mathbf{P}$, into the current state.

$$\mathbf{P}^-_{k+1} = \mathbf{A}\mathbf{P}_t\mathbf{A}^T + \mathbf{Q} \tag{9}$$

Where $\mathbf{A}$ was the 4×4 identity matrix I and Q was the covariance matrix associated with the process noise in Equation 8, in other words the uncertainty in prediction. Under this Kalman filter, uncertainties were linearly dependent, and thus simple to determine. Note that at every Time Update, the uncertainty matrix $\mathbf{P}$ increased by the additional process noise in $\mathbf{Q}$.

During the Measurement Update phase, the sensor generated the *measurement* vector, $\mathbf{z}_{k+1}$, which had the same inputs as the *state* vector. The purpose of this phase is to use a portion of the prediction and a portion of the measurement to update the state vector estimate. To determine how much weight to put on the sensor measurement, or how much weight to put on the prediction, the Kalman gain $\mathbf{K}_{k+1}$ was calculated.

$$\mathbf{K}_{k+1} = \mathbf{P}^{-}_{k+1}\left(\mathbf{P}^{-}_{k+1} + \mathbf{R}\right)^{-1} \tag{10}$$

Here $\mathbf{R}$ was the covariance matrix associated with the sensor and was related to the *measurement* vector. The importance of this equation can be understood when examined as a scalar fraction. If the sensor covariance $\mathbf{R}$ is low, the Kalman gain will approach 1. On the other hand, if the $\mathbf{R}$ dwarfs $\mathbf{P}^{-}_{k+1}$, then the Kalman gain will approach 0.

The Kalman gain was used to update the state vector, and the uncertainty associated with the state at time k+1. The state vector $\mathbf{x}_{k+1}$ and uncertainty $\mathbf{P}_{k+1}$ were calculated using

$$\mathbf{x}_{k+1} = \mathbf{x}^{-}_{k+1} + \mathbf{K}_{k+1}\left(\mathbf{z}_{k+1} - \mathbf{x}^{-}_{k+1}\right) \tag{11}$$

$$\mathbf{P}_{k+1} = \left(I - \mathbf{K}_{k+1}\right)\mathbf{P}^{-}_{k+1} \tag{12}$$

The prediction $\mathbf{x}^{-}_{k+1}$ was added to a *proportion* of the difference between the measurement vector $\mathbf{z}_{k+1}$ and the prediction $\mathbf{x}^{-}_{k+1}$. If the Kalman gain was 1, implying that the sensor has no uncertainty, then the full difference would be added, and the state vector would be equal to the measurement vector. However, if there was high uncertainty in the sensor, the Kalman gain would approach 0, setting the state vector equal to the prediction. To determine the uncertainty associated with the state at k+1, the Kalman gain is subtracted from the Identity Matrix. If the Kalman gain was close to 1, the covariance matrix $\mathbf{P_{k+1}}$ would approach 0, essentially stating there is no uncertainty. On the other hand a lower Kalman gain would only partially reduce the uncertainty in our state estimate. The reader should note that the uncertainty cannot increase in this stage, it can only decrease. This is in contrast to the Time Update phase, where the uncertainty can only increase.

A Kalman gain of 1 is essentially the initial basic movement model. If Kalman gain for the velocity term is 1, this implies that velocities from the previous frame are simply discarded, and the new velocities are accepted. Kalman gains lower than 1, imply that previous velocities also have an input on the decision of the current state. This achieves smoothing through *a priori* information.

During the calculation of the variances of the state vector, it became apparent that this Kalman filter would not function well. A person's velocity had a variance independent of direction; thus it could not be separated into a variance in the x-axis and a variance in the y-axis. Calculating the variance for $\dot{x}$ and $\dot{y}$ would not describe the variance of a person's velocity. The variance in $\dot{x}$ was dependent on the variance in $\dot{y}$ and vice versa. The solution was to examine a person's velocity, v, as a magnitude, and have an angle, θ, to describe the direction. The IFM did not measure v or θ, instead these values were calculated by using the finite difference between two frames in same manner as the initial model.

3.4.2 Extended Kalman Filtering

Inclusion of the v and θ terms in the state and measurement vectors changed the state transfer functions to

$$\begin{aligned} x_{k+1}^{-} &= x_k + v_k \cos\theta_k \Delta t \\ y_{k+1}^{-} &= y_k + v_k \sin\theta_k \Delta t \\ \theta_{k+1}^{-} &= \theta_k \\ v_{k+1}^{-} &= v_k \end{aligned} \tag{13}$$

These transfer functions are *nonlinear*. Uncertainties from these terms must be properly propagated between time steps. However, the original Kalman filter can only propagate uncertainties linearly. The solution was to linearize the uncertainties, this approach is known as an Extended Kalman filter.

An extended Kalman filter (EKF) follows the same processing scheme as the original Kalman filter, employing the Time Update (prediction) and Measurement Update (correction) phases. In this work, x and y were nonlinearly dependent on v and θ. Any uncertainty in v and θ had to be added to the uncertainty in x and y. Unlike the original Kalman filter, the EKF employs Jacobians to linearize the added uncertainty. There were two types of uncertainty: uncertainty due to predictions $\mathbf{Q}$ and uncertainty due to measurement $\mathbf{R}$.

In the Time Update phase, the uncertainty equation had to be changed to

$$\mathbf{P}_{k+1}^{-} = \mathbf{A}_{k+1}\mathbf{P}_k\mathbf{A}_{k+1}^T + \mathbf{W}_{k+1}\mathbf{Q}\mathbf{W}_{k+1}^T \tag{14}$$

where $\mathbf{A}$ was a Jacobian that propagates the uncertainties of v and θ from the previous frame k into the uncertainty of x and y of the current frame k+1. The Jacobian $\mathbf{W}$ linearizes the uncertainty from v and θ, which is inherent to the process functions, to be added to the uncertainty from the previous frame, k. It should be noted that the Jacobians $\mathbf{A}$ and $\mathbf{W}$ are calculated at each frame.

3.4.3 Determination of Jacobians

The extended Kalman filter requires Jacobians to linearize uncertainty. Uncertainty generated by v and θ propagated to the values of x and y in the state vector determined by Equation 13. Since x, y are predicted solely from v and θ the prediction covariance matrix $\mathbf{Q}$ was a 2×2 matrix. To calculate $\mathbf{Q}$, the initial basic tracking model (Equation 7) was employed and scans of moving persons were taken. The covariance of the v and θ in these scans was $\mathbf{Q}$. The Jacobian $\mathbf{W}_t$ was used to propagate the uncertainty related with v and θ. The derived Jacobians $\mathbf{A}$ and $\mathbf{W}$ were

$$\mathbf{A}_k = \begin{bmatrix} 1 & 0 & -v_k \sin\theta_k \Delta t & \cos\theta_k \Delta t \\ 0 & 1 & v_k \cos\theta_k \Delta t & \sin\theta_k \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{W}_k = \begin{bmatrix} -v_k \sin\theta_k \Delta t & \cos\theta_k \Delta t \\ v_k \cos\theta_k \Delta t & \sin\theta_k \Delta t \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{15}$$

To calculate the 4×4 sensor covariance $\mathbf{R}$, scans of a person rotating in place were taken. This allowed for easy correlation of the person between scans, ensuring that all variance was from the sensor. The covariance of the entire measurement vector z_t was $\mathbf{R}$. No linearization was done because the uncertainties from the sensor were added to the prediction uncertainty.

3.5 The Complete Classifier

Using the constructed extended Kalman filter, the operation of the complete Motion-based classifier is now described. In explaining its operation, the case at time k where no tracks are yet established is considered first. For each $\mathbf{x}'(k) \in \mathbf{X}'(k)$, if $Q(\mathbf{x}'(k)) \geq Q_{\mathcal{P}} \implies \mathbf{x}'(k) \in \mathcal{P}$, and the MC immediately associated a track $\mathbf{t}(k) \in \mathbf{T}$ with the person $\mathbf{x}(k)$. For $Q(\mathbf{x}'(k)) < Q_{\mathcal{P}} \implies \mathbf{x}'(k) \in \bar{\mathcal{P}}$, and the non-person object is subsequently ignored.

Time $k+1$ begins with a data association phase. If a track $\mathbf{t}(k) \in \mathbf{T}$ established at time k cannot be associated with a person $\mathbf{x}'(k+1) \in \mathcal{P}$ based upon a maximum distance threshold (like the initial motion-based classifier), the track was deleted. If a successful data association is made, an extended Kalman filter was established to facilitate future tracking and $\mathbf{t}(k)$ is moved to $\mathbf{T}_{EKF}$. Each track $\mathbf{t} \in \mathbf{T}_{EKF}$ was parameterized by a state vector $[x, y, \theta, v]^T$ corresponding to the position, bearing, and velocity of the person relative to the wheelchair. Note this implies that a track can only persist in $\mathbf{T}$ for a single step before being either promoted to $\mathbf{T}_{EKF}$ or deleted. Lastly, if no association was made for a person $\mathbf{x}'_j(k+1)$, a new track $\mathbf{t}_j(k+1)$ was added to $\mathbf{T}$.

The more interesting case occurred at time step $k+2$. First, all tracks in $\mathbf{T}_{EKF}$ are propagated in accordance with the EKF time update equations (Equation 13) A data association phase was then made using these updated track positions with persons in $\mathbf{X}'(k+2)$ again based upon a distance threshold. If this data association fails, the track was *not* immediately deleted. Instead, a lower hysteresis threshold Q_{min} was employed for established tracks, and association with every $\mathbf{x}'(k+2) \in \bar{\mathcal{P}}$, but where $Q(\mathbf{x}'(k+2)) > Q_{min}$ was attempted. This approach was particularly advantageous when persons become partially occluded, or are on the edge of the LIDAR's field-of-view, the appearance classifier will often identify these as non-persons. However, established tracks provided additional confidence to the motion classifier, and as a consequence it accepted a lower Q value as a result.

If still no association was made for a track $\mathbf{t} \in \mathbf{T}_{EKF}$, the track was deleted. Otherwise, once an association was made, the measurement update phase of the respective EKF was run where

$$\mathbf{z_k} = [x_k, y_k, \theta_k, v_k]^T \tag{16}$$

It should be noted that (16) was a simplification of the actual measurement process. While the relative x-y positions of a person were estimated directly using the centroid of the bounding box from the appearance classifier, θ_k and v were estimated using a finite difference approach from successive position estimates across two time steps. However, this simplification worked well in practice. Once all tracks in $\mathbf{T}_{EKF}$ were processed, any tracks in $\mathbf{T}$ and persons in $\mathbf{X}'(k+2)$ were processed as in the previous time step.

To summarize the motion classifier update phase:

1. Any tracks in $\mathbf{T}_{EKF}$ were processed first. They were initially associated with persons in $\mathcal{P}$ as identified by the appearance classifier, and if necessary with non-persons $\mathbf{x}' \in \bar{\mathcal{P}}$ but where $Q(\mathbf{x}') > Q_{min}$. If either association succeeds, the respective EKF was updated. Otherwise, the track was deleted.
2. Any tracks in $\mathbf{T}$ were processed second. They were associated with any persons remaining in $\mathcal{P}$. If this was successful, the respective track is moved to $\mathbf{T}_{EKF}$, and an EKF was initialized. Otherwise, the track was deleted.
3. Any persons remaining in $\mathcal{P}$ that were not associated with tracks in $\mathbf{T}_{EKF}$ or $\mathbf{T}$ were added to $\mathbf{T}$.

4 EXPERIMENTAL RESULTS

All experiments were conducted on the Lehigh University campus. For ground-truth data, both LIDAR frames and video images were simultaneously logged and timestamped for manual post-processing. The LIDAR exposure times were set to 150 ms and 2000 ms, which empirically provided the best performance outdoors. This resulted in a frame rate of approximately 8 Hz. Results now follow.

4.1 Appearance Classifier

The performance of the appearance classifier was characterized using single LIDAR frames taken while manually driving the wheelchair across Lehigh's campus. Samples in the test set were disjoint from the training set used by the k-NN classifier. The goal of the appearance classifier was to minimize the number of false positives. False negatives have the potential to be overridden by the motion classifier, as discussed in Section 4.2.

A total of 105 images of persons and 464 non-persons were acquired. Classification results with $k = 5$ and $Q_{\mathcal{P}} = 0.5$ are presented in the confusion matrix below. Of the 105 persons imaged, 97 were classified as persons, for a detection rate of 92.4%. Of the 464 non-person objects imaged, 19 were classified as persons for a false-positive rate of 4.1%.

Table 1: Confusion Matrix for k-NN Appearance Classifier

		Decided	
		nonperson	person
Actual	nonperson	445	19
	person	8	97

In truth, the true-positive rate was lower than hoped. Upon reviewing misclassifications, several pathological cases were identified. One was labeled the "floating torso" problem. Due to the relatively low power of the IFM LIDAR, detection of darker objects became problematic as range increased. In particular, when pedestrians wore dark pants, the range which they could be detected dropped dramatically. The net result was the appearance of a floating torso in space. This is illustrated at Figure 5. The frequency of these occurrences influenced the addition of several of these as prototypes in the training set. Still, misclassifications of these instances were not uncommon. It should be noted that if the person continued to approach the wheelchair and a more complete LIDAR scan is received, a correct classification resulted. A similar situation was observed in the other extreme, when a person started too close to the wheelchair, resulting in an incomplete scan. As the person distanced themselves from the wheelchair, a correct classification would result. However, beyond a certain distance, the quality of the scan would degrade. In these situtations, the motion classifier was useful in maintaining a correct classification.

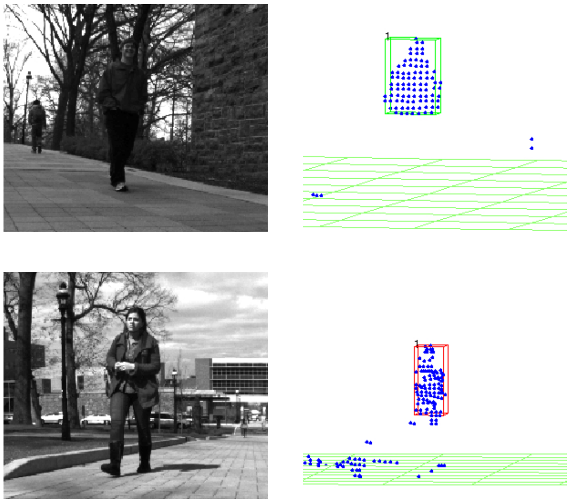

Figure 5: Floating torso distributions. The top example was correctly classified as a person, while the more fragmented bottom example was not.

4.2 Appearance + Motion-based Classifier

The second phase of testing involved evaluating the full-classifier in continuous, dynamic operations. Preliminary testing involved an approximately 5 minute run of driving the wheelchair across Lehigh's Campus during normal hours to ensure sufficient pedestrian traffic. Pedestrians were only counted once, meaning each entry was a unique person. Classification was determined successful if an EKF track was established on the pedestrian. For these tests, $k = 5$ and the Q parameters for the MC were set to $Q_{\mathcal{P}} = 0.5$ and $Q_{min} = 0$. The latter meant that if at least 1 of the 5 neighbors believed the object was a person and an EKF track was established, the MC could override the decision of the appearance classifier. Results are provide in the confusion matrix below. From these results, it can be seen that 45 of 47 persons were classified

Table 2: Confusion Matrix for Complete Classifier

		Decided	
		nonperson	person
Actual	nonperson	50	7
	person	2	45

correctly, for a detection rate of 95.7%. Examination of the two failure cases revealed that both were instances where the pedestrian never completely entered the field of view of the LIDAR. An example of this is shown at Figure 6, where the bottom half of a pedestrian appears on the edge of the LIDAR's field-of-view. If these cases are excluded, the true positive detection rate would have been 100%.

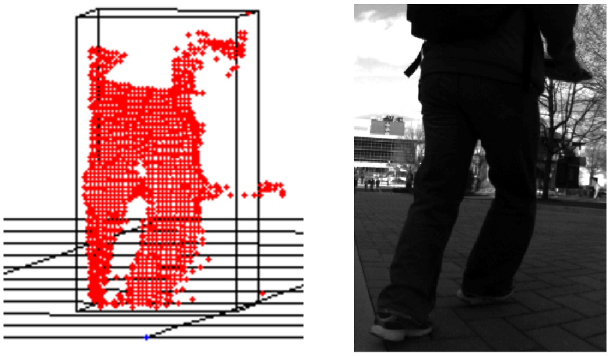

Figure 6: A situation where the appearance classifier did not get a full view of the pedestrian, deterring tracking and classification

During this experiment a false positive rate of over 12% was ascertained. The increase from the static test can be attributed to the fact that during driving operations, a single non-person object might be imaged 10s of times from a range of distances and orientations. If it was incorrectly classified as a person for a single frame, it was considered a false positive. In practice, associating conditional probabilities with such cases would likely prove useful.

A strength of the MC for maintaining tracks is illustrated by Figure 7. In this example, there are two persons in $\mathbf{T}_{EKF}$, meaning that EKF tracks have been established. The person on the right side begins to migrate out of the LIDAR's field-of-view to a point where the appearance classifier no longer associates it with the person class. Nevertheless, the MC is able to maintain a track until the person is almost entirely out

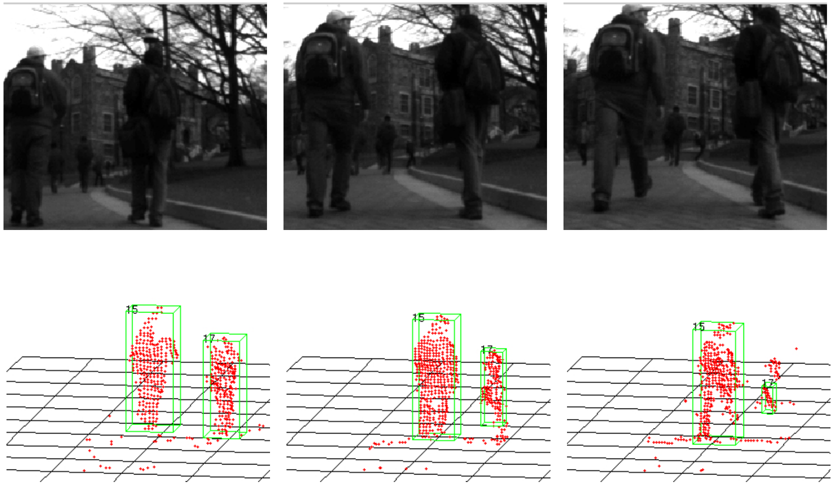

Figure 7: A sequence of frames, showing the tracking of a person even after they fail the appearance classifier.

of the LIDAR's view.

5 DISCUSSION

5.1 Appearance Classifier And Features

As stated in the introduction, the success of pattern recognition is reliant on the features extracted from the data. Many classifications methods exist, a review of which is beyond the scope of this work. Some methods attempt to determine features on their own; however, even these algorithms must be guided in the right direction.

A sizeable proportion of development time was spent on determining features that would generate classifications correctly. The simplest features (height, width, and depth) were chosen in the early stages of the algorithm and offered excellent results. There are few classes that have similar height, width, and depth, implying that pedestrians have a well defined mean (or distribution) in this feature space. To determine other features, failure cases were examined. One of the earliest failure cases to be avoided was the “floating torso” distribution (Figure 5).

Failure when presented with floating torso’s was prevented in two manners. First, the height, h_k, for cluster C_k was determined by

$$h_k = \max_y C_k \tag{17}$$

This equation offset the height of floating torsos to the height of a scan of a full person. Earlier equations used the height of the bounding box; this resulted in floating torso’s height being halved, distancing it from the person class.

The failure case was also avoided by the introduction of λ, the top-heaviness feature.

$$\lambda_k = \frac{|T|}{|C_k|} \qquad \text{where } T = \left\{ \vec{p} \quad | \quad \vec{p} \in C_k \text{ and } p_y > \frac{h_k}{2} \right\} \tag{18}$$

The top heaviness feature depended on the heuristic that a person has more surface area above their waste; more surface area equated to more pixels captured by the flash LIDAR. However, this heuristic can be shown to fail. A specific, and interesting, failure case occurred when women wore long dresses or skirts. The clothing provided enough surface area to imbalance the top-heaviness ratio.

Of note-worthy mention is the density feature, ρ. Clusters C_k were taken and flattened into the x-y plane (height and width plane.) This was achieved by ignoring the z values of every point in the cluster. An example of the density feature can be seen at Figure 4. The result was a two-dimensional, discrete binary image. Points were indexed by pixel values (elements of $\mathbb{N}$), instead of Euclidean values (elements of $\mathbb{R}$). As stated earlier, there is extensive research in using camera systems for pedestrian detection. Most color approaches generate a mask, a binary image describing pixels of interest, to examine the shape of a pedestrian. Extracting this binary image is a computationally expensive procedure when working in RGB space. For LIDARs, this procedure is computationally trivial. Upon extracting this mask, algorithms developed for analyzing geometry from a color image can be applied. These algorithms are refined in producing powerful features for classification. It is important to reiterate that only geometric features can be extracted. It is advantageous to reapply provenly powerful techniques in new fields of research.

Not every feature developed was helpful, many features were discarded as they proved to be disadvantageous. One of the earliest features discarded was the height-width ratio, r_k.

$$r_k = \frac{h_k}{w_k} \tag{19}$$

Heuristically, people are taller than they are wide, providing good grounds for use. Computationally, this ratio was not consistent. Examination revealed the variation of the ratio did not allow for proper segmentation from dissimilar classes (*e.g.*, fire hydrants).

Attempts were made in describing the distributions of the clusters, C_k. The skewness, γ, describes the assymetry of the distribution. Similar to the top-heaviness feature, it was expected that γ would favor the torso of the pedestrian.

$$\gamma_k = \frac{\mu_{(3)k}}{\sigma_k^3} \tag{20}$$

where $\mu_{(3)k}$ represents the third moment arm from the mean of C_k and σ_k was the standard deviation of the distribution.

The eigenvectors of the distribution of the clusters were also examined. Given a covariance matrix Σ_k of cluster C_k, the eigenvectors were determined using

$$\mathbf{V^{-1}\Sigma_k V = D} \tag{21}$$

where $\mathbf{V}$ is the matrix of eigenvectors that diagnolizes the covariances matrix Σ_k into the eigenvalues $\mathbf{D}$ of Σ_k. Eigenvectors describe the orientation of a distribution. It was expected that pedestrians would typically be scanned while upright, producing a major eigenvector with nearly vertical orientation. Unfortunately, most scanned clusters of other classes also had major vertical eigenvectors (*e.g.*, trees, walls, posts).

5.2 k-NN Distance Metric

Pattern classification can neatly be described as assigning an object to a class given the probabilities resultant from *a priori* information. k-NN classifiers are particularly useful when there is no information on the probability of a class. The probability distribution is estimated by a feature space populated by preclassified samples. k-NN reduces to determining the class of the closest neighbors to the object in question. The object in question is then assigned the majority class. The reader is reminded that objects are described by a feature vector, and are thus points in feature space. Each feature is a dimension in feature space. However, one can quickly infer that features do not scale the same. The top-heaviness ratio, λ_k, is bounded between $[0..1]$, while height $h_k \in \mathbb{R}$. This difference in scale affects the distance, applying differing weight to the distance of features.

Distance is a major aspect to the k-NN approach, thus a properly scaled distance metric required careful consideration. Two approaches were extensively tested to provide the most consistent and accurate results: the Minkowski Metric and the Mahalanobis Distance.

The Minkowski Metric is defined using a level l parameter.

$$d^l(\mathbf{x} - \mathbf{x}') = \left[|\mathbf{x} - \mathbf{x}'|^l \right]^{\frac{1}{l}} \tag{22}$$

The reader should note that $l = 1$ is the Manhattan distance, and $l = 2$ is the well known Euclidean distance metric. To normalize the feature spaces, such that each feature has an equal part in the distance, a scaling factor α was applied to each feature of the prototypes such that the feature's standard deviation, $\sigma = 1$.

The Mahalanobis distance was described in Equation 5. Testing was done on what training data to use for the determination of the covariance matrix Σ. The training data was classified by two classes, person $\mathcal{P}$ or non-person $\bar{\mathcal{P}}$. A covariance matrix determined using data from

1. $\mathcal{P}$
2. $\bar{\mathcal{P}}$
3. Both $\mathcal{P}$ and $\bar{\mathcal{P}}$

was tested. Futhermore, Option 3 could be calculated in a number of ways. One attempt calculated the covariance matrix of the set $\mathcal{P} \cup \bar{\mathcal{P}}$. Another approach calculated the covariance matrix of each class separately and averaged the two covariance matrices. The latter approach provided consistently accurate results.

The Minkowksi metric consistently offered a lower *false negative* rate. A *false negative* is a person classified as a nonperson. This misclassification is considered costly, and effort was taken for it to be avoided. The Minkowksi Metric also had a very high *false positive* rate, a nonperson classified as a person. Mahalanobis distance offered an overall lower rate. The *false positive* rate was much lower, while the *false negative* rate was slightly higher. It was decided that the lower overall error rate of the Mahalanobis distance was advantageous over the Minkowski's Metric lower *false negative* rate. The decision was influenced by the existance of the motion-based classifier's added classification. A lower *false positive* rate reduces the number of tracks generated by the MC, and *false negatives* are efficiently overruled by the tracking algorithm.

5.3 Evaluation of the Extended Kalman Filter

Time was spent examining the operation of the EKF, producing results worthy of discussion. A common trend of the filter was to relate the distance y of the object in question to the gain value for the velocity. As y increased, the EKF was less and less certain of the v measured, lowering the Kalman gain for the v term. This trend is understandable as measurements became inaccurate at larger distances; inaccuracies were caused by inherent noise from the sensor as well as the inconsistent quality of scans of objects at a large distance. Inconsistent scans generated differing centroids as the appearance of the object to the IFM changed.

The EKF consistently produced accurate predictions, adjusting well when a pedestrian made a sudden change in direction. Furthermore, the velocity was smoothed due to the use of the Kalman gain. This smoothing increased the error of the prediction in some frames by preventing a drastic change in velocity, even if that change was present; however, in the common case, the drastic change in velocity was attributed to noise. More importantly, the error caused by the Kalman gain was now consistent. Consistent errors are more advantageous than sporadical accuracy.

6 FUTURE WORK

The greatest limitation of the IFM LIDAR system was its relatively coarse angular resolution. This made it difficult to segment pedestrians that were walking in close proximity to one another. An example is shown at Figure 8. In this instance, all three persons were lumped into a single cluster where from the ground-truth camera image they are obviously disconnected. One potential means for handling these cases would be to refine larger connected components using a k-means clustering approach, where k could be correlated to the geometry of the bounding box. More work is required in this area.

An oversight in this work was the calculation of the density feature. Currently, the feature is generated by binning all pixels into 10 cm $\times$ 10 cm boxes (in the x-y plane), which discretized the Euclidean points. However, this approach took a coarse scan and created an even coarser image as seen at Figure 4. The IFM LDAR returned scans in three 64×50 matrices (x, y, and z). Using the discretization provided by the original scan may prove advantageous. However, this would require a restructuring of the prototype database. Currently, the database is populated by clusters of points

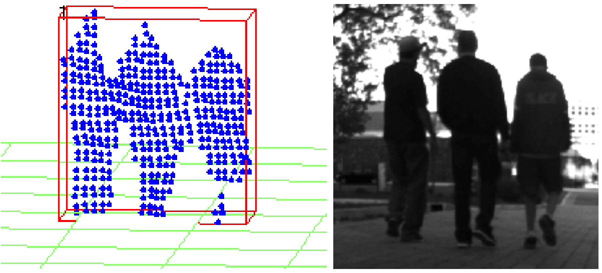

Figure 8: Where the CCL was unable to seprate the objects of interest.

(where each cluster represents one prototype.) The pattern classifier converts each cluster into a feature vector at intialization. The strength of the current structure is the ability to generate new features without manipulating the database. However, the current structure removes the contextual information provided by the organization of the original scan. More research needs to be done into a method to discretize the points in the database into a binary image.

The appearance classifier is still a work in progress. As discussed, there exist many well developed features that can be extracted from binary images. Currently, research on and application of this subset of features has not be undertaken. The search for features is one that can be described as having no end. There are always properties which can be discovered to further separate classes, creating more accurate classifiers. However, even the most advanced features will be useless if the underlying knowledge base is not populated. There is always room for more prototypes in a k-NN database, a position taken by many pattern recognition researchers. The case for this position can be made by examining Figure 9. The person class is not well enveloped in the presented figure. Due to the bad distribution of prototypes (black dots), an object is misclassified (red) as the closest neighbors to it are in the Person Class. More features may be helpful in this particular instance, however a similar *hole* can be found in another area of the feature space. In the perfect world, there would be a infinite number of prototypes,

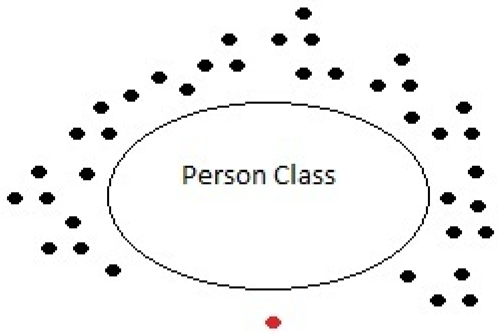

Figure 9: An instance in feature space where the person class is not well enveloped, causing a misclassification (red). The black dots represent prototypes that are not part of the Person Class.

thus generating the perfect distribution. It should be noted that increasing the size of the database increases the complexity of the computation, as more distances need to be determined. This dilemma is inherent to the k-NN classifier. A possible solution is to construct an algorithm that spaces all prototypes in the database, filling in the holes as misclassifications occur and pruning overpopulated areas. This algorithm can be run offline as the database is indepedent of the classifier.

To this point, only single values for the parameters $Q_{\mathcal{P}}$ and Q_{min} have been examined. A sensitivity analysis to tune these parameters may in fact improve performance.

7 CONCLUSION

In this work, the potential of low-cost 3D LIDARs to be applied to a people tracking task was demonstrated. While providing significantly lower angular resolution than their more expensive cousins, these systems still provide the accurate distance estimates and illumination invariance associated with LIDAR systems. A k-Nearest Neighbour classifier was constructed to determine objects to track based on appearance. An extended Kalman filter was established to aid the appearance classifier with fragmented scans. Preliminary results to date indicated a successful tracking rate of over 95% during dynamic operations. Nevertheless, there is significant room for improvement.

References

[1] J. Dietsch, “2010: When mobile robots reached the tipping point,” *IEEE Robotics and Automation Magazine*, Dec 2010.

[2] M. Buehler, K. Iagnemma, and S. Singh, Eds., *Journal of Field Robotics: Special Issue on the 2007 DARPA Urban Challenge*, vol. 25, Wiley Periodicals, 2008.

[3] Microsoft, “Xbox Kinect,” `http://www.xbox.com/en-US/kinect`.

[4] Inc. Velodyne Lidar, “Hdl-64e,” `http://www.velodyne.com/lidar/hdlproducts/hdl64e.aspx`.

[5] IFM Efector, “3D Sensor Wins Design News Magazine’s 2009 Best Product of the Year - Sensors and Vision Category,” `http://www.ifm.com/ifmus/web/prod_tip_3d.htm`.

[6] David Geronimo, Antonio M. Lopez, Angel D. Sappa, and Thorsten Graf, “Survey of pedestrian detection for advanced driver assistance systems,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1239–1258, 2010.

[7] L. Spinello, K. O. Arras, R. Triebel, and R. Siegwart, “A layered approach to people detection in 3d range data.,” in *Proc. of The AAAI Conference on Artificial Intelligence: Physically Grounded AI Track (AAAI)*, 2010.

[8] Bertrand Douillard, Alex Brooks, Fabio Ramos, and Hugh Durrant-Whyte, “Combining laser and vision for 3d urban classification,” in *NIPS Workshop on Learning from Multiple Sources with Applications to Robotics*, 2009.

[9] Luis Ernesto Navarro-Serment, Christoph Mertz, Nicolas Vandapel, and Martial Hebert, “Ladar-based pedestrian detection and tracking,” in *Proc. 1st. Workshop on Human Detection from Mobile Robot Platforms, IEEE ICRA 2008*. May 2008, IEEE.

[10] Luis Ernesto Navarro-Serment, Christoph Mertz, and Martial Hebert, “Pedestrian detection and tracking using three-dimensional ladar data,” in *Proc. of The 7th Int. Conf. on Field and Service Robotics*, July 2009.

[11] D.V. Prokhorov, “Object recognition in 3d lidar data with recurrent neural network,” in *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, june 2009, pp. 9 –15.

[12] C. Gao, I. Hoffman, T. Miller, T. Panzarella, and J. Spletzer, “Autonomous docking of a smart wheelchair for the automated transport and retrieval system (atrs),” *Journal of Field Robotics*, vol. 25, no. 4-5, pp. 203–222, 2008.

[13] Jon Bohren, Tully Foote, Jim Keller, Alex Kushleyev, Daniel Lee, Alex Stewart, Paul Vernaza, Jason Derenick, John Spletzer, and Brian Satterfield, “Little Ben: The Ben Franklin Racing Team’s entry to the DARPA Urban Challenge,” *Journal of Field Robotics*, vol. 25, no. 9, pp. 598–614, Sep 2008.

[14] M. Fischler and R. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” in *Communications of the ACM*, 1981.

[15] Qingmao Hu, Guoyo Qian, and Wieslaw L. Nowinski, “Fast connected-component labelling in three-dimensional binary images based on iterative recursion,” *Computer Vision and Image Understanding*, vol. 99, pp. 414–434, April 2005.

[16] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, John Wiley and Sons, 2001.

[17] Greg Welch and Gary Bishop, “An introduction to the kalman filter,” 1995.

8 VITA

Constantin Savtchenko was born on May 20, 1987 in Sofia, Bulgaria. He is the son of Andrey Savtchenko and Magdalena Anguelova. He attended undergraduate school at Lehigh University in Bethlehem, PA and was awarded a B.A. degree majoring in Computer Science and Cognitive Science with a minor in Business in May 2010. He achieved Magna Cum Laude with his senior thesis on Reinforcement Learning. He also did his graduate work at Lehigh University and was awarded a M.S. in Computer Science in May 2011. His masters research was performed in Lehigh's VADER (Vision, Assistive Devices, and Experimental Robotics) Lab under Professor John R. Spletzer.