People Matching for Transportation Planning Using Texel Camera Data for Sequential Estimation

Scott E. Budge, *Senior Member, IEEE*, John A. Sallay, *Student Member, IEEE*, Ziang Wang, and Jacob H. Gunther, *Senior Member, IEEE*

***Abstract*—This paper addresses automatic people matching in the dynamic setting of public transportation, such as a bus, as people enter and then at some later time exit from a doorway. Matching a person entering to the same person exiting at a later time provides accurate information about individual riders such as how long a person is on a bus and the associated stops the person uses. At a higher level, matching exits to previous entry events provides information about the distribution of traffic flow across the whole transportation system. The proposed techniques may be applied at any gateway where the flow of human traffic is to be analyzed.**

For the purpose of associating entry and exit events, a trellis optimization algorithm is used for sequence estimation, based on multiple texel camera measurements. Since the number of states in the trellis grows exponentially with the number of persons currently on the bus, a beam search pruning technique is employed to manage the computational and memory load. Experimental results using real texel camera measurements show 96% matching accuracy for 68 people exiting a bus in a randomized order. In a bus route simulation where a true traffic flow distribution is used to randomly draw entry and exit events for simulated riders, the proposed sequence estimation algorithm produces an estimated traffic flow distribution which provides an excellent match to the true distribution.

***Index Terms*—People Matching, Texel Camera, Trellis Optimization, Multiple Measurements.**

I. Introduction

TRANSIT authorities use statistical information about their ridership in order to plan bus routes. Currently, many transit systems employ Automatic People Counters (APCs) to obtain a count of the people on the bus. An APC also provides a count of the number of people who enter and exit the bus at each stop. Many counting technologies currently exist, such as Traf-sys [1] and Acorel [2].

A people counting system associates each stop with the number of people entering and exiting at that stop. There is, however, other information that would be of value in route planning that is not available in current people counting systems. For example, it would be useful to know at which stops each person enters and exits the bus. This would associate each person with two stops. Transit authorities could use this information to plan routes based on the flow of human traffic instead of making decisions based on knowing only the number of people who utilize each stop. Therefore, it is desirable to match an exiting person to one who previously entered in order to obtain this information.

This paper compares two approaches to associating entry and exit events: traditional classification methods and sequence estimation using a trellis search algorithm. In the former approach, when a person enters, a classifier for that person is created. When a person exits, the set of exiting measurements is compared against all available classifiers. This sort of dynamic classification has received little attention in the literature, but a few cases exist. Bagui et al. [3] use a modification of the nearest-neighbor rule on all of the measurements collected. Shakhnarovich [4] creates a distribution from the measurements and uses the Kullback-Leibler divergence for matching. Roy and Khattree [5] use a modified Linear Discriminant Analysis (LDA) for binary classification problems. To reduce the number of unknown parameters, they assumed compound symmetry and first order autoregressive structured covariance matrices.

A second approach for people matching builds memory into the matching process so that an optimal ordering of exit events is obtained. In this paper, a sequence estimation technique is explored for the ordering task, based on a Viterbi-like trellis search algorithm [6]. Tanaka *et al* [7] use the Viterbi algorithm and a modified trellis for the recognition of distorted patterns. Dietterich [8] reviews the statistical learning problems involving sequential data, mentioning the use of Viterbi algorithm for computational efficiency. Grafmuller [9] gives the brief introduction to a trellis-based classification method, and demonstrates it using an optical character recognition (OCR) example. In the present work, he number of calculations and the amount of memory needed for the trellis search turns out to be intractable. Thus, a beam search approximation is explored [10], [11] as well.

Although it is a common figure of merit, the matching (or classification) accuracy is not the primary measure of interest in understanding traffic flow. Of primary interest is the distribution of entrances and exits. For example, one might be interested in a probability distribution of how long persons ride the bus, or one may be interested in the distribution describing how long persons entering at a particular stop ride. The proper figure of merit in these cases is based on how well the probability distribution is estimated. The proposed trellis-based sequence estimation algorithm provides the information needed to estimate this traffic flow distribution. Since classification accuracy is commonly used, this paper reports results for both classification accuracy and distribution estimation accuracy.

Scott Budge, Ziang Wang, and Jacob Gunther are with the Department of Electrical and Computer Engineering, Utah State University, Logan, UT, 84321. E-mail: {scott.budge@ece.usu.edu, jacob.gunther@usu.edu}.

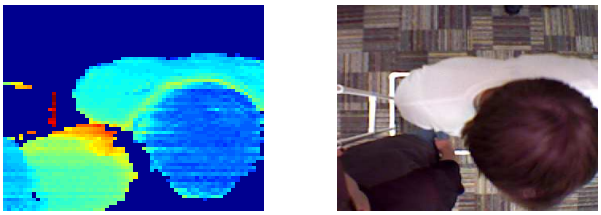
John Sallay is with Zeta Associates, Fairfax, VA, 22030.

This paper is organized as follows. Section II explains the technology used for data gathering. Section III presents the people matching algorithm and its components. Test results are compared and summarized in Section IV. In Section V, performance of the estimation techniques in a simulated bus route scenario is given. Lastly, conclusions are presented in Section VI.

II. Texel Cameras and Configuration

Data are gathered using a special device known as a texel camera, which is the combination of a conventional digital color camera with a Light Detection and Ranging (LIDAR) sensor [12]. LIDAR is a technology that measures the distance between the sensor and object in the field of view, allowing the points from the sensor to be connected into a 3D triangulated wire-frame surface. A detailed description of the LIDAR sensor used in this paper is found in [13], [14]. The digital camera and LIDAR sensor are co-boresighted and aligned so that the digital image and LIDAR data are fused together to form a 3D textured surface [15]. The texel camera thus captures both distance and color information. A pixel in a texel image is a 4-tuple (r, g, b, d) where d stands for distance.

In a practical system, a texel camera would be mounted above each doorway in a bus. The results in this paper were obtained using a simulated bus environment, where a texel camera was mounted above a door frame in a laboratory. People walked through the door frame in order to simulate getting on and off a bus. A person is detected and tracked from one edge of the field of view to the other as they walk underneath the camera (using tracking algorithms not reported here), with their head and shoulders identified and tracked separately. An example of the fused range and color images acquired simultaneously from the texel camera are given in Fig. 1. Two people are visible. The full head and one shoulder of one person can be seen in the middle of the image. A small portion of the head and one shoulder of a second person appears at the lower-left corner of the image. Situations where one or more people are present in the doorway of a bus are typical as people enter and exit.



(a) (b)

Fig. 1. Sample images simultaneously acquired from a texel camera in a simulated bus doorway. (a) Depth image pseudocolored from closer (blue) to farther (red) with the floor thresholded to dark blue, (b) Color image.

III. People Matching Algorithm

The method used to match people exiting a bus to a pool of people who are currently on the bus is given in Fig. 2. The algorithm consists of three parts. First, features are extracted from the texel camera video frames of persons exiting at a bus stop. These features are then used to create feature vectors. The vectors are compared to feature vectors stored in a database of people already on the bus, and a cost measure is computed for each possible match. The costs are then entered into a trellis-based optimization step which makes the best decision on the sequence of persons exiting the bus. The decisions are made only when the data support a decision, and not at every exit event.

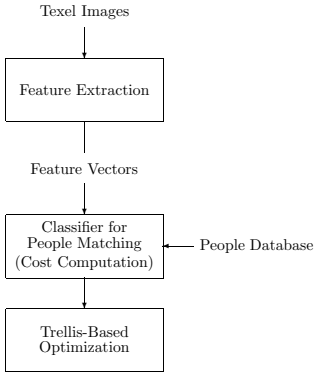


Fig. 2. People matching algorithm.

A. Feature Extraction

In order to correctly associate entry and exit events, identifying information needs to be collected about each person. This is done by segmenting the images into regions of pixels corresponding to the head and shoulders of each person in the field of view. This segmentation step is necessary to distinguish individuals when there are several persons in the field of view, and is also used to track and count individuals as they enter or exit the bus. (The process used for this segmentation, and the association of head regions to shoulder regions, will be described in a separate publication.) Features describing the head and shoulders of each person are then computed from the segmented regions.

The features collected can be broken down into three categories: depth features, color features calculated using chromaticity value, and color features calculated from tristimulus value. The depth features used are head and shoulder height. The color features used are found in the RGB (Red, Green and Blue) and HSI (Hue, Saturation, and Intensity) color systems. The intensity in the HSI color systems is not used because the intensity of a scene can vary greatly between the time when a person enters and when they exit. For example, a person may enter a bus in direct sunlight and then leave in a shadow. The hue and saturation, however, should not change significantly if the illumination changes.

The separation of a person into head and shoulder regions provides a natural division of features. All of the features that are calculated for a head region are also calculated for the shoulder regions. A feature vector is created by stacking all of the features associated with a person into a vector. In the following discussion, features will be referred to as belonging

to a region. This region could be either a head region or a shoulder region. Sometimes, sufficient information as to whether a region is a head or shoulder is not available in a single frame; thus, a region is not decided to be a person's head or shoulder until the person completely exits the camera's field of view. At that time, the association is made and an entire feature vector is created, using data from both the head and shoulder regions, for every frame in which the person appears.

During the time a person enters or exits the bus, multiple frames from the texel camera are obtained, creating a 3D video sequence. Assuming the texel camera has a framerate of about 15 frames per second, and that a person takes 1.5 seconds to pass through the field of view, this provides about 22 observations per entry and exit event. This number may vary depending on the speed of the passenger.

B. Supervised Classification Techniques

When a person enters the bus, a sequence of feature vectors is collected by processing the output of the texel camera as described above. These vectors are used as a training set to construct a classifier for the person. When a person leaves the bus, another sequence of feature vectors is collected. This set is compared to all of the classifiers that currently exist and the best match is found. This section explores different criteria for training classifiers and finding a best match.

A wide variety of classification techniques exist [16]–[18]. This section explores three: Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest Neighbor (KNN). The first two of these were selected because they fit easily into the maximum-likelihood sequential estimation framework proposed here (See Section III-C2). The KNN classifier was selected for the simplicity of implementation needed for a practical, real-time, bus application. It also allowed a non-statistical point of comparison to the other methods.

The problem to be explored is different than usual application of these techniques. Normally, one measurement is classified at a time. In this particular case, several measurements are collected as a person walks underneath the texel camera, and then the measurements are combined together. The aforementioned classification techniques have been extended to account for classification using multiple measurements known to be from the same class.

1) Linear Discriminant Analysis: In LDA, it is assumed that all measurements are independent draws from a set of Gaussian distributions with different means, $\boldsymbol{\mu}_i$, for each class. In the context of the problem at hand, each class corresponds to a distinct individual on the bus. It is further assumed that all of the Gaussian distributions have the same covariance matrix, R. When only one measurement is considered, the maximum *a posterori* (MAP) decision is given by

$$\hat{k} = \arg\max_k P(G = k | \mathbf{X} = \mathbf{x}), \tag{1}$$

where k is the class considered and $\mathbf{x}$ is the measured feature vector for one frame in an exit event.

Using Bayes Rule,

$$P(k|\mathbf{x}) = \frac{P(\mathbf{x}|k)P(k)}{P(\mathbf{x})}, \tag{2}$$

where

$$P(\mathbf{x}|k) = \frac{1}{C \cdot |\hat{R}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\hat{\boldsymbol{\mu}}_k)\hat{R}^{-1}(\mathbf{x}-\hat{\boldsymbol{\mu}}_k)^T}, \tag{3}$$

$$\hat{\boldsymbol{\mu}}_k = \frac{1}{V_k} \sum_{g_i = k} \mathbf{y}_i, \tag{4}$$

$$\hat{R} = \frac{1}{V - K} \sum_{k=1}^{K} \sum_{g_i = k} (\mathbf{y}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{y}_i - \hat{\boldsymbol{\mu}}_k)^T, \tag{5}$$

$$P(k) = \hat{\pi}_k = \frac{V_k}{V}. \tag{6}$$

In (4)–(6), V is the total number of vectors in the training sequence, obtained from all previous frames of all persons entering the bus, V_k is the total number in the sequence corresponding to class k, K is the total number of classes, and $\mathbf{y}_i$ is the ith training vector.

If terms common to all k are ignored and equal priors are assumed, then the problem can be simplified to finding the class k that minimizes

$$\delta_k^l(\mathbf{x}) = (\mathbf{x} - \hat{\boldsymbol{\mu}}_k)^T \hat{R}^{-1} (\mathbf{x} - \hat{\boldsymbol{\mu}}_k), \tag{7}$$

which is the log likelihood (or Mahalanobis distance between $\mathbf{x}$ and $\hat{\boldsymbol{\mu}}_k$). This is often expressed as

$$\delta_k^l(\mathbf{x}) = (\mathbf{x} - \frac{1}{2}\hat{\boldsymbol{\mu}}_k)^T \hat{R}^{-1} \hat{\boldsymbol{\mu}}_k, \tag{8}$$

where the $\mathbf{x}\hat{R}^{-1}\mathbf{x}$ term is removed because it is common to all classes.

2) Linear Discriminant Analysis with Multiple Measurements: There are several feature vectors collected as a person exits a bus, one from each frame the person is present in. A better classification can be performed by using all of these measurements for classification.

LDA can easily be extended to exploit multiple measurements. In the case of $m = 2$ measurements, repeated application of Bayes Rule yields

$$P(k|\mathbf{x}_1, \mathbf{x}_2) = \frac{P(\mathbf{x}_1|k, \mathbf{x}_2)P(k|\mathbf{x}_2)}{P(\mathbf{x}_1|\mathbf{x}_2)}, \tag{9}$$

$$= \frac{P(\mathbf{x}_1|k, \mathbf{x}_2)P(\mathbf{x}_2|k)P(k)}{P(\mathbf{x}_1|\mathbf{x}_2)P(\mathbf{x}_2)}. \tag{10}$$

If the measurements are independent, this simplifies to

$$P(k|\mathbf{x}_1, \mathbf{x}_2) = \frac{P(\mathbf{x}_1|k)P(\mathbf{x}_2|k)P(k)}{P(\mathbf{x}_1)P(\mathbf{x}_2)}. \tag{11}$$

This process can be extended to m independent measurements as

$$P(k|\mathbf{x}_1, \cdots, \mathbf{x}_m) = P(k) \prod_{i=1}^{m} \frac{P(\mathbf{x}_i|k)}{P(\mathbf{x}_i)}. \tag{12}$$

As in LDA with one measurement, computation can be simplified by minimizing the $\log$ of (12) and removing common terms. This produces

$$\delta_k^l(x) = \log \hat{\pi}_k - \frac{1}{2} \sum_{i=1}^{m} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)^T \hat{R}^{-1} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k), \tag{13}$$

where the x in $\delta_k^l(x)$ represents the set of measurements, $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m\}$. If all of the classes are equally likely, this can again be simplified as

$$\delta_k^l(x) = \sum_{i=1}^{m} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)^T \hat{R}^{-1} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k), \tag{14}$$

The most likely class is now the one that minimizes this sum of Mahalanobis distances.

As in the case of LDA with one measurement, this can be reduced to minimizing

$$\delta_k^l(x) = (\bar{\mathbf{x}} - \frac{1}{2}\hat{\boldsymbol{\mu}}_k)^T \hat{R}^{-1} \hat{\boldsymbol{\mu}}_k, \tag{15}$$

where $\bar{\mathbf{x}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}_i$. This form of $\delta_k^l(x)$ yields a significant reduction in the number of required computations.

3) Quadratic Discriminant Analysis: QDA makes the same assumptions as LDA, except that it does not assume that each of the classes have a common covariance matrix. Thus, a covariance matrix needs to be computed and stored for each class as given by

$$\hat{R}_k = \frac{1}{V_k - 1} \sum_{g_i = k} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)^T. \tag{16}$$

This produces similar discriminant functions to those described in Section III-B1. The $\frac{1}{|\hat{R}_k|^{\frac{1}{2}}}$ term from the original distribution can no longer be ignored. In the case of one measurement this becomes

$$\delta_k^q(\mathbf{x}) = \log \hat{\pi}_k - \frac{1}{2}\log|\hat{R}_k| - \frac{1}{2}(\mathbf{x} - \hat{\boldsymbol{\mu}}_k)^T \hat{R}_k^{-1} (\mathbf{x} - \hat{\boldsymbol{\mu}}_k). \tag{17}$$

When multiple measurements are available the discriminant function becomes

$$\delta_k^q(x) = \log \hat{\pi}_k - \frac{m}{2}\log|\hat{R}_k| - \frac{1}{2}\sum_{i=1}^{m} (\bar{\mathbf{x}}_i - \hat{\boldsymbol{\mu}}_k)^T \hat{R}_k^{-1} (\bar{\mathbf{x}}_i - \hat{\boldsymbol{\mu}}_k). \tag{18}$$

4) K-Nearest Neighbor: In KNN, feature vectors are classified according to the vectors from the training set that they are closest to in Euclidean distance. When people enter a bus, the corresponding set of feature vectors is stored in a database. When a person exits the bus, the feature vectors for the exit event are classified based upon a K-nearest neighbor decision rule, and a majority voting rule is used to make the final decision: an exiting person is assigned to the class to which most of their feature vectors were classified.

C. Sequence Estimation

Marginal accuracy characterizes the error rate for single exit events, and is defined as the probability of correct association when one person is removed from the set of persons on a partially full bus. In real bus environments, however, sequences of exits occur and a decision error made at an early event can influence decisions made at later events. Therefore, the problem now turns into one of sequence estimation. This section describes a few methods for incorporating previous information into future decisions for sequences of events.

1) A Simple Technique: A simple approach is to make a hard decision as soon as a person exits the bus. The class associated with the exiting person is removed from further consideration. In this manner, everyone on the bus will eventually be assigned to an exiting event.

The hard decisions are the only information that is incorporated into the later decisions. Any decision reduces the number of people to be considered at a later time. If the decision is correct, this increases the probability of a correct decision at a later time. However, if the decision is incorrect, the error will propagate through to other decisions. One decision error can cascade into several.

2) A Maximum-Likelihood Approach: The simple approach of Section III-C1 leaves much room for improvement. This section describes an optimal method of performing sequence estimation in the maximum likelihood sense.

The likelihood of a sequence of n sets of feature vectors is

$$P(x_1, x_2, \cdots, x_n | k_1, k_2, \cdots, k_n), \tag{19}$$

where the observation x_i is the set of m feature vectors collected as the ith person exits a bus, $x_i = \{\mathbf{x}_{1i}, \mathbf{x}_{2i}, \cdots, \mathbf{x}_{mi}\}$, and k_i is the index of the person that leaves the bus on the ith exit event. The likelihood in (19) is the probability of measured data given that certain people exited the bus in a specific sequence.

The sequence estimation problem may be expressed as

$$\hat{\mathbf{k}} = \arg\max_{\mathbf{k}} P(x_1, x_2, \cdots, x_n | k_1, k_2, \cdots, k_n), \tag{20}$$

where $\mathbf{k}$ is a vector whose ordered entries are $k_1, k_2, \cdots, k_n$.

Each observation, x_i, only depends on the class, k_i, from which it came. That is to say that each x_i is conditionally independent given k_i,

$$P(x_1, x_2, \cdots, x_i | k_1, k_2, \cdots, k_n) = \prod_{i=1}^{n} P(x_i | k_i). \tag{21}$$

LDA is used for classification, thus each x_i is assumed to be a set of observations from a Gaussian random variable with mean, $\hat{\boldsymbol{\mu}}_{k_i}$, and covariance, $\hat{R}$. The log of (21) can be used to simplify computations,

$$\log \prod_{i=1}^{n} P(x_i | k_i) = \sum_{i=1}^{n} \log P(x_i | k_i), \tag{22}$$

where

$$\begin{aligned} \log P(x_i | k_i) = &- \log C - \frac{1}{2}\log|\hat{R}| \\ &- \frac{1}{2}\sum_{j=1}^{m} (\mathbf{x}_{ji} - \hat{\boldsymbol{\mu}}_{k_i})^T \hat{R}^{-1} (\mathbf{x}_{ji} - \hat{\boldsymbol{\mu}}_{k_i}). \end{aligned} \tag{23}$$

This can be simplified in the same manner as was used in Section III-B2. The end result is $\delta_{k_i}^l(x_i)$ from (15). The new problem is to find

$$\hat{\mathbf{k}} = \arg\min_{\mathbf{k}} \sum_{i=1}^{n} \delta_{k_i}^l(x_i). \tag{24}$$

An optimal solution would be to evaluate this sum for every possible value of $\mathbf{k}$ and choose the set that minimizes the sum.

This would be prohibitive, even for a relatively small number of people on a bus, since the number of possible orderings for N people is $N!$.

The Viterbi algorithm reduces the number of orderings to be tested without sacrificing optimality [6]. It uses a trellis to search through all of the possible paths. Figure 3 shows an example trellis with four people on a bus. The description of the approach starts with the introduction of trellis components for the problem:

- State – Stores the indices of people currently on the bus, which is represented using black dots.
- Edge – Indicates the index of people exiting the bus, linking two states in two steps, the person index is labeled on the edge.
- Step – Stores the states after a person's getting off the bus. A real bus stop may have multiple steps since there may be more than one person getting off the bus. In this example, one person exits at each stop. The S_i labels at the bottom of the graph represent the different steps. A step is an exiting or entering event.
- Pool – People that have not been decided by the algorithm yet, who are either on the bus or have exited the bus. When people get on the bus, they are put into the pool for possible decision. Once decided by the algorithm, they are moved out of the pool. The behavior of people getting off the bus will not necessarily result in the removal from the pool since they may not be decided by the algorithm the moment they get off.

The first state on the left hand side of Fig. 3 is {a,b,c,d}, which is to say that all four people are on the bus. One person exits at S_1. There are four possible new states. Each of these are produced by removing one person from the initial state, {a,b,c,d}.

The trellis can also be used to accommodate people entering a bus. Figure 4 gives an example where there are originally three people on the bus. Two people leave (one per stop), and then another person enters. There are no labels on the edges corresponding to person d entering the bus. The edge labels only correspond to exiting people.

It should be noted that in practice, multiple people may enter or exit the bus at each stop. In such a situation, each step of the trellis represents an event, which is either a person entering or a person exiting. In the case of people entering, all of the people can be added as a group to the states containing people on the bus as a single event, and the trellis is extended as in Fig. 4. People exiting must be processed one at a time, as single events, before the entering persons are added.

When each step is added to the trellis, each edge extending forward from the last step is weighted with the cost of that decision, as given in (15). The total cost of each path backward from a state in the new step is computed by adding the costs for each edge in the path. The trellis is then pruned by removing all edges from each new state extending backward to a state in the previous step, except the edge with lowest cost. At this point, only one unique path backward survives from each new state.

Any paths that (after initial pruning) have no surviving edge extension to the next step must also be pruned back, starting from the current step and pruning edges and states recursively backwards to the step where the number of edges forward from a state after pruning is at least one.

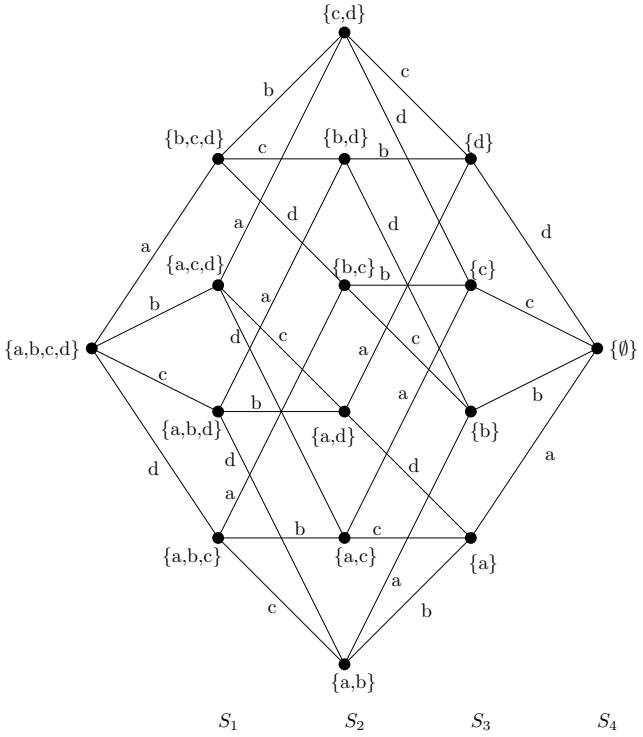


Fig. 3. A simple trellis with four people.

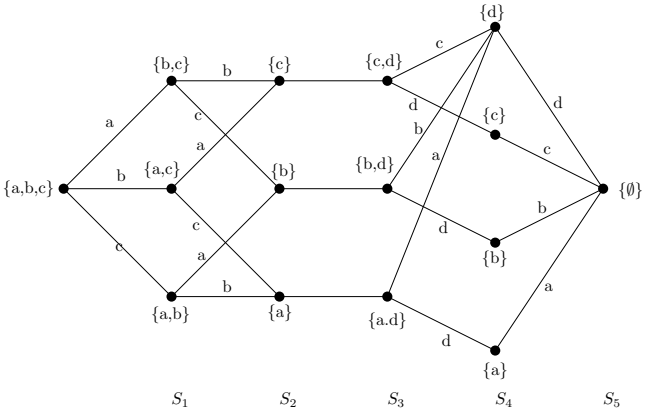


Fig. 4. Trellis representing an entering person.

If at any time there exists only one path backward from a step, the sequence of decisions represented by that path are made, and the trellis is pruned to start at that step.

Finally, in order to minimize the number of surviving paths in the trellis at each step, the minimum cost associated with each person exiting in a group at a stop could be sorted and the steps ordered from lowest cost to highest. This allows the less ambiguous matches to be processed first, possibly eliminating later ambiguities.

3) A Beam Search Approximation: The trellis optimization algorithm significantly reduces the number of paths that need to be explored in order to arrive at an optimal solution in the maximum likelihood sense. However, the number of paths can

still be prohibitive. The number of states at a given time step depends on the number of people currently on the bus, the past history of persons entering and exiting, and the number of people that will be considered as a possible match. For example, assuming N people are initially on the bus, no hard decisions are made, and one person exits at each stop, the number of states after k stops is

$$\binom{N}{k}. \tag{25}$$

The number of states to be considered can grow quite large, even for relatively small N. It is not feasible to store or to compute paths for such a large number of states. A common approach used in such a situation is a beam search [10], [11]. A beam width, W, is selected. At each step of the trellis, only the W minimum cost paths are stored. At the next step, edge extensions are made from these paths, the path costs are sorted, and then the number of edge extensions is pruned to the W paths of smallest cost. Optimality is sacrificed in order to decrease memory and computational requirements.

The number of states to be stored at each step is constant, however, the number of edge extensions changes for each step depending on several factors. It is bounded by

$$E_t^+ N_t \leq E_{t+1}^- \leq \binom{N_t^p}{N_t - 1}, \tag{26}$$

where E_t^- is the number of edge extensions before pruning at step t, E_t^+ is the number of edge extensions after pruning (W or less), N_t is the current count of people on the bus, and N_t^p is the current number of people under consideration (those for whom a hard decision has not been made).

It is not necessary to make all E_{t+1}^- edge extensions. At step t, all of the paths in the trellis are in sorted order. There are N_t people on the bus at this time. For each person, there is a cost associated with the event of that specific person exiting at time t. These costs are also sorted. Using sorted data reduces the number of edge extensions that need to be computed and stored.

It is not necessary to store the full state at each time step either. The necessary information can be found by only storing the person corresponding to the edge labels as in Fig. 3. The set of exiting people can be found by back tracing through the trellis.

4) Trellis Pruning and Decision Making using a Beam Search: The optimal trellis search maintains a unique backward path for each state in the latest step. When using a beam search, there may be some states that have no edge extensions to the new step since only the W minimum cost states are saved. These must be pruned as described in Section III-C2. As this process is applied to every step, the number of surviving states in each previous step reduces from the initial W.

There are four types of decisions that can be made with the addition of each new step:

1) Decide the current person exiting and delete the added step – When a step is added, the edges linking previous step and the new step may all have the same labels. This is because the cost of the match between the person's entering feature vectors and leaving feature vectors is so good that the cost of other decisions will not be lower within the lowest cost W states. No new step is added, and the person getting off the bus is immediately decided.
2) Delete an inner step – After pruning the states within a step, there is a possibility that edges linking two steps within the trellis all have the same label. This condition means that the best choice for that step is the unique label on the edges. After recording the decision, the step is pruned and the edges from the previous step and the next step are linked together.
3) Trace an inner path – This is a special case of the previous type of decision. After pruning the states within a step, there is a possibility that only one edge is linking the step to the previous step within the trellis. Because all paths from previous steps converge to this edge, all people in the path to the edge will be decided by tracing back from the edge.
4) Trace the final path – This happens when all people on the bus have gotten off. If the number of persons on the bus are correctly counted, the trellis will always end with a single edge. At this time, all people in the trellis that haven't been decided by the previous three methods will be decided by tracing back from the last edge along the path with the minimum accumulated cost.

IV. Sequence Estimation Performance

The previous section outlined three sequence estimation techniques. The simple sequential estimation and beam search approximation to the maximum-likelihood estimate were tested. The marginal accuracy of LDA is also included for comparison.

A. Experimental Design

Data was collected for 68 distinct individuals entering and exiting a simulated bus environment. In a real-life scenario the direction in which a person travels determines whether a person is entering or exiting. However, in the simulated bus environment, this distinction between entering and exiting is not necessary. There are two sets of feature vectors collected for each person: one is taken as the entry set and the other is taken as the exit set. The decision as to which is entry and which exit is arbitrary. In the tests outlined in this section, the association is done randomly. That is to say that for some people in an experiment, the first set of feature vectors will correspond to those people entering and for the rest of the people in the experiment, the second set of feature vectors will correspond to those people entering. This increases the randomness of the experiments performed. It is important to note that this does not double the number of people in the experiment; it randomly selects which data will be used for training and which data will be used for testing.

The texel camera used for data acquisition uses a cold mirror to separate the incoming light from the scene and the returning light from the laser. Unfortunately, some of the light from the laser passes through to the color camera sensor. The LIDAR being used emits laser light at 785 nm wavelength, close to the

visible red band. As the silicon sensor used for the color image acquisition is sensitive to this wavelength, the laser tends to saturate the red values, especially for tall people. Therefore, it was necessary to preprocess the red component in each image before extracting features. Two approaches were used. The first sets the mean of the red component to be the average of the means of green and blue components for the same pixel since they are both affected much less than the red component by the additional LADAR light. The second is to set the red component to be the same value in every pixel.

Table I shows the features that were used. There are 12 features that are collected for the head and shoulder regions, or 24 features per person. Note that the same features are used for both heads and shoulders. The number of feature vectors extracted from the 3D video sequence ranged from less than nine to over 20 per individual.

TABLE I
LIST OF FEATURES USED FOR CLASSIFICATION.

Image used	**Feature extracted**
Depth images	Height
Color images with the mean of red shifted by the average of means of green and blue	Chromaticity mean (green, blue) Chromaticity standard deviation (red, green and blue) Hue mean, Saturation mean
Color images with the red of all pixels set to a constant value	Green mean, Blue mean Hue mean, Saturation mean

In considering the accuracy of the proposed people matching algorithms, the number of errors is dependent on the order in which people exit and the total number of people on the bus. In order to communicate this information as accurately as possible, Monte Carlo analysis is used. This analysis is done for differing numbers of initial people on the bus, ranging from 1 to 68. In all cases, a graph will be shown that displays the accuracy as a function of the initial number of people on the bus.

In the experiments performed, a bus size, N, is chosen. N people are randomly selected from the 68 available people. One of the two sets of feature vectors available for each person is chosen at random to be the training data and a classifier is constructed. In Section IV-B, it is shown that LDA performed better than the other classifiers, and thus it will be used as a baseline in all of the experiments performed. The mean is found for each person and an covariance matrix for all of the persons is found.

One person is removed from the bus at a time, and the discriminant function value is found for each possible association. People are removed from the bus until the bus is empty. The number of errors, E, is counted, and the error percentage is computed as $\frac{E}{N}$. There exist several possible error measures for this system. For the purposes of this paper an error is an incorrect association of an exit event with an entrance event.

Each experiment is repeated so that the total number of exit events is approximately constant. The total number was chosen to be 6,800. For example, when two people are on the bus, the experiment described above will be performed 3,400 times, and when 68 people are on the bus the experiment is performed 100 times. In this manner the accuracy percentages are calculated using approximately the same number of samples in each case.

B. Performance of Classification Techniques

A simple test was performed in order to compare the three classification techniques, described in Section III-B, prior to selecting one for use in the sequence estimation experiments.

The marginal accuracy of a classifier is used to compare the discrimination power of a classifier. It is more difficult to match people correctly if 100 people are on the bus than if there are only 2 or 3. Thus, the marginal accuracy is associated with a given number of people on the bus.

In all of the comparisons, the marginal accuracy was computed with 68 people on the bus. For each person, one of the two sets described above is selected at random to be the entering vectors, and the other to be the exiting vectors. The total number of possible selections is therefore $O(2^{68})$.

The number of possibilities is far too large to test every possible combination of people. Monte Carlo analysis was used to estimate the accuracy of each classifier. The experiment of removing one person from the bus was repeated 6,800 times and the number of correct associations was counted. Table II summarizes the performance of each technique.

TABLE II
MARGINAL ACCURACY OF CLASSIFIERS WITH 68 CLASSES.

Technique	**Accuracy**
LDA	90.4%
QDA	56.8%
KNN	55.2%

It is assumed that the head and shoulder features are independent. This produces a block diagonal covariance matrix that was used in LDA and QDA,

$$R = \begin{bmatrix} R_H & 0 \\ 0 & R_S \end{bmatrix}. \tag{27}$$

LDA performed significantly better than the other classifiers. The reason for this is the sparsity of data. In QDA each person has 24 features that are tracked in two 12x12 correlation matrices. There is a minimum of twelve measurements needed in order for the matrices to be invertible. In many cases, there are fewer than nine measurements available for a person. A regularizer is used, but it causes significant distortion when there are only a few measurements. QDA fails to perform well simply because the correlation matrices are not very accurate for some of the people in the data set.

In KNN a similar problem a occurs. Due to the sparsity of data, a measurement may be relatively close to the true mean, but close to only a few of the measurements from the correct person to be associated with.

In LDA, the overall covariance matrix is constructed from all of the training measurements. The main cause of variation

from the mean value of the features collected is measurement noise. This noise is independent of the person being matched. LDA is able to produce a more stable covariance estimate than QDA by combining all of the data available. LDA was used for the remainder of the experiments in Section IV.

Other classification techniques could have been tested. For example, there are many distribution matching techniques. These would suffer from the same problems as QDA and KNN. The sparsity of the data would limit the accuracy of such techniques.

C. Marginal Accuracy for Different Size Sets

The marginal accuracy of several classifiers was compared in Table II for a set size of 68 people. Although marginal accuracy provides performance for a single event, and not for a sequence of events, it is included for purposes of comparison.

Figure 5 shows the marginal accuracy for differing set sizes using LDA, without sequence estimation.

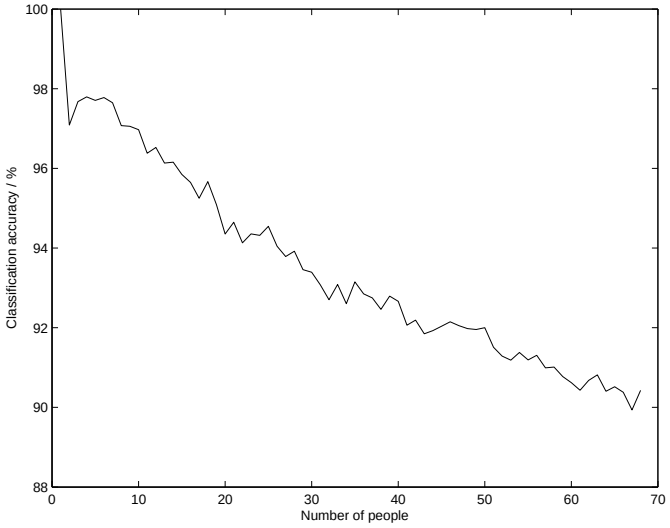


Fig. 5. Marginal accuracy for different numbers of persons in the classification set.

D. A Trellis-Based Beam Search Approach

The trellis-based approach was described in Section III-C2. The full trellis is far too large to store for even small N; thus, a beam search is used where only the W best paths are kept at a given time. There is a trade-off between beam-width and the amount of memory and computations required. Large beam widths are more accurate but the amount of time necessary to use them may be prohibitive.

Perhaps the easiest classification test is to always choose the best match at every event. This person is then removed from further consideration. This simple approach is equivalent to a trellis-based estimation with a beam width of 1.

Beam widths of 1, 10, 100, 500 and 1000 were tested. Figure 6 shows the accuracy percentage for a sequence of exit events as a function of the number of people on the bus at the start of the exit events, and the beam width. Accuracy is defined as the number of correct matches divided by the total number of events. The accuracy increases with the beam width. This is to be expected because a larger beam width is a better approximation of the full trellis. A beam width of 1000 is a significant improvement over a beam width of 1, with an increase in accuracy of almost 15% with 68 people on the bus.

The dip in performance with two persons on the bus is related to features used in these experiments. Optimal feature selection will be addressed in a future publication, and preliminary results indicate this effect is reduced by better feature selection.

It can also be seen that for small numbers of people on the bus, an increase in the beam width has little effect on the accuracy. This occurs because the full trellis can be stored for very small N. Increasing the beam width beyond the maximum width of the trellis has no effect on the outcome.

If the best match at a given event is not the correct match, it is possible, in some cases, to correct this using the trellis. This is, however, highly dependent on the ordering of the data. As an example, number the people according to the order in which they exit. If person 2 is matched to person 1, this error may be corrected at the next event when person 2 exits. If person 68 is matched to person 1, however, the error cannot be corrected until person 68 exits, 67 events later.

In the trellis, any path that does not have any successor edges at the current step is pruned. When, at a given step, there is only one path that has not been pruned a hard decision is made. The trellis delays those decisions and the beam width controls how long the delay is. For a small beam width, the delay is also small, which reduces the error correction capabilities of the trellis.

The beam search requires significantly more memory and computations than the other proposed methods. However, the amount required is still reasonable. There is very little overhead, and each state in the trellis requires only 8 bytes of storage in the current implementation in C++. Even with a beam width of 1000, each step requires approximately 8 kB of storage. Thus, the program would only need a maximum of about 1 MB of memory for 125 steps with a beam

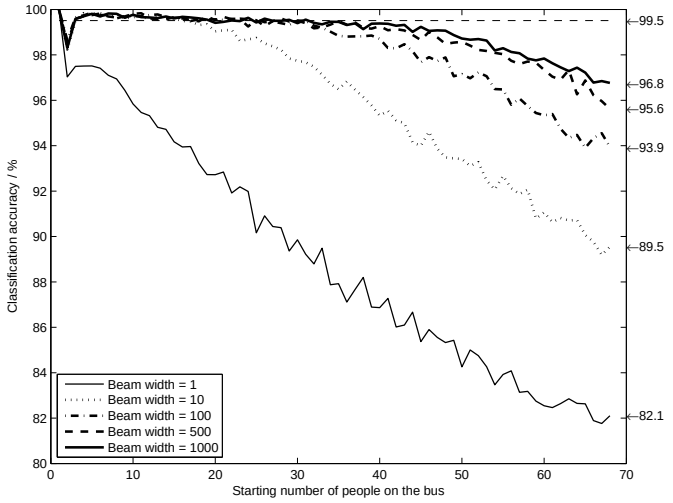


Fig. 6. Accuracy of sequential estimation using a beam search.

width of 1000. The computations are lengthy but may not be prohibitive. Using a beam width of 1000, it takes about 0.15 seconds to process each step, with 68 people on the bus on a machine with a 2.67 GHz dual-core processor and 8 GB of RAM. The computations for the matching algorithm can be done in between the stops that a bus makes. In most cases this would still allow time to process all of the data before the next bus stop occurs.

V. Simulation of a Real Bus Route to Estimate Ridership Statistics

The purpose of this simulation is to use the sequential estimation method to estimate the probability distribution of the riding length when people get on the bus at different stops. The simulation attempts to accurately represent how people use a transit system, by randomly assigning more people to enter and exit at popular stops, and simulate the cyclical nature of bus routes.

A. Simulation Setup

The total number of people in the database is divided into groups entering the bus at each stop according to a known distribution. This distribution is input to the simulation based on information about the bus route to be simulated, such as stops that are more popular or less popular for entering passengers. For example, if ten persons are in the database and there are three stops in the bus route, an average of 40% (four) might be selected for the first stop, an average of 30% (three) for the second stop, and an average of 30% (three) for the third stop. The exact number of people getting on the bus at each stop is generated randomly using a Poisson distribution with the mean set to these averages. Once the number of persons entering at each stop is known, the persons selected to enter at each stop are then randomly chosen from the database.

The probability distribution of the number of stops that people ride is input into the simulation for each stop. These distributions are also determined from information about the bus route. For example, if Stop 2 is more popular as an exit stop then Stop 1 or Stop 3, the distribution for Stop 1 will have more weight for exiting at Stop 2 than for Stop 3. Similarly, the distribution for Stop 3 will have more weight for exiting at Stop 2 than for Stop 1. This can be represented as a probability matrix, with each row representing an entrance stop, and each column representing an exit stop. Thus, a person getting on at Stop 1 will be assigned to get off at a stop determined by drawing a random number according to the distribution for that stop.

In reality, bus routes are often circular, which means that the bus will return to its starting point when it finishes the route, and repeat the route. For the simulation, the bus will run multiple circuits. Also, because of the cyclical nature of the route, people getting on the bus at a stop may take a whole circuit riding the bus to the stop that is prior to the stop that he or she gets on the bus. As a result, when the bus is about to finish its simulated schedule, it will take one additional circuit to let people get off the bus who get on during the last circuit of the bus, and no passengers will get on the bus during the additional circuit. By doing this, the probability distribution of each circuit in the schedule will be the same.

From the description above, information that is tagged to a person has four parts:

- The stop number that the person gets on.
- The stop number that the person gets off.
- The circuit number that the person gets on.
- The circuit number that the person gets off.

The trellis starts empty on the zero-th step, and the pool is empty since no one is on the bus. At the first stop, people get on the bus, and no people get off the bus. The pool is filled with people from the first stop. From the second stop on, there will be people getting off the bus, causing the trellis to grow, while the pool is filled by people getting on the bus.

When a person is assigned to get off, the person is compared to all the people in the pool, by computing the Mahalanobis distance of the leaving feature vectors of the person getting off using the classifiers for the people in the pool according to (15). These costs are then entered into the trellis. The classification decision is postponed until one of the four possible decisions, described in Section III-C4, can be made. It should be noted that the enlarging of the pool is done after growing the trellis, since there is no need to compare the people getting off with those just getting on the bus. When a person is decided, the stop that he gets off the bus is recorded.

Using the information of the stop number that person gets on and off the bus, as well as the number of stops of the route, the riding length for the person can be determined. If the stop number that a person gets on the bus at is smaller than that at which the person gets off the bus, then the two stops are within the same circuit. If the stop number that a person gets on the bus at is larger than that at which the person gets off the bus, then the two stops are in different circuits. When information of all people is accumulated, the probability distribution of the riding length is computed.

B. Simulation Results

The bus route used in the simulation is Cache Valley Transit District (CVTD) Route 3, located in Logan, Utah, USA. The map of Route 3 is shown in Figure 7. Data were gathered using 10,000 Monte Carlo trials of the route simulation. The comparison between the estimate and the input distribution of the ridership for the entire route and one of the stops (No.23) are shown in Figure 8 and Figure 9, respectively. Note that for Stop 23, peaks in the rider lengths occur four stops later and 12 stops after that. These correspond to persons exiting at the CVTD Transit Center and at Stop 12 (located by a park and a church), and represent a realistic model of the route. These results are typical of all 26 stops on the route. Using a beam width of 200, the matching accuracy of the simulation is 99.5%. This is reasonable since at most of the stops the total number of people on the bus is less than 40.

There are two ways of describing the error of the simulation. The first is to use the matching accuracy, and the second is to use the error between the estimated probability distribution of the riding length and the truth. It can be seen that some matching error will not be reflected in the error in estimating

the probability distribution of the riding length, because the mismatch may happen within a stop, giving correct statistics for the riding length even though there are incorrect matching results.

Fig. 7. Map of CVTD Route 3. (Source: http://www.cvtdbus.org)

VI. Summary and Conclusions

Three classification techniques (LDA, QDA and KNN) were extended for the case of multiple independent measurements, and LDA was found to perform the best in the bus scenario. Two sequence estimation techniques (simple and trellis-based sequence estimation) were also developed and tested on actual texel image data using Monte Carlo methods. Based on this empirical study, the trellis-based beam search significantly outperformed the simple technique, with the accuracy improving with the width of the beam. The main limitation with the beam search is processing time, but it was found to be reasonable for widths up to $W = 1000$. The system is able to perform very well in the situations tested, achieving 96% accuracy with 68 people on a bus.

The estimates for rider length were very good, with the probability mass function estimates very close to the truth (all estimates less than 0.14% in error). This suggests that a very accurate understanding of the route usage statistics can be found using this technique. With this information, transit systems can better design efficient bus routes and rate schedules, far more effectively than using counting information alone. These tests show the system has the potential to work very accurately in an actual bus environment.

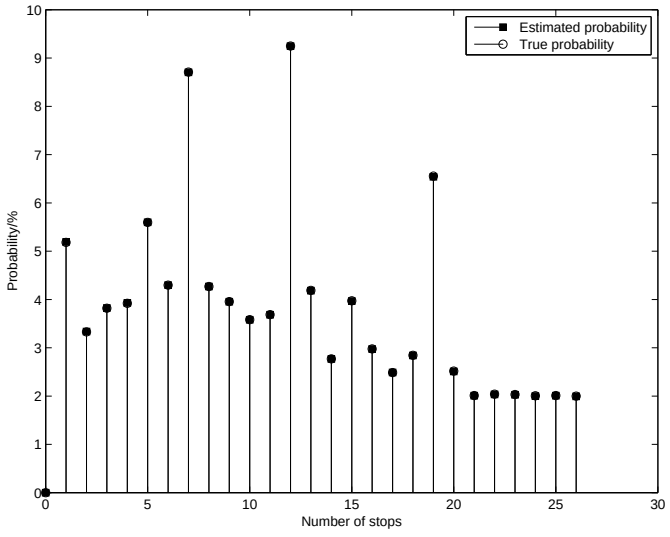


Fig. 8. Distribution of the number of stops that passengers take for the whole route.

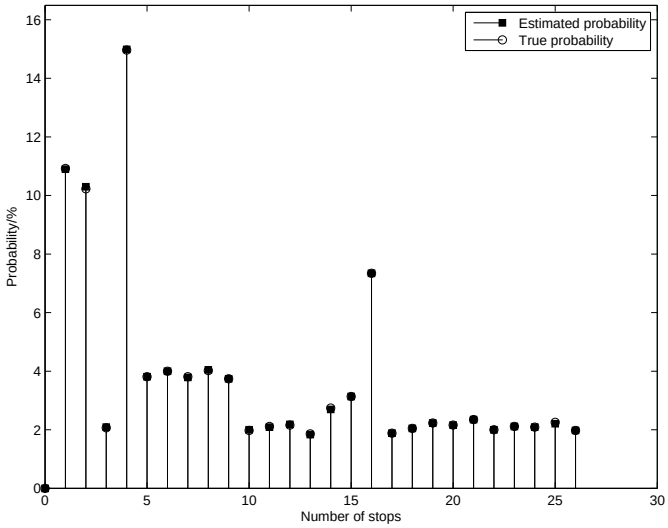


Fig. 9. Distribution of the number of stops that passengers take when getting on the bus at Stop 23.

References

[1] TrafSYS, "People counters," [http://www.trafsys.com/people-counters.aspx], 2009.

[2] ACOREL, "ACOREL Onboard Counter," [http://www.acorel.com/en/ACOREL%20Onboard%20Counter.pdf], 2009.

[3] S. Bagui, S. Bagui, A. Chatterjee, and K. Mehra, "Classification with multiple independent measurements under a separate sampling scheme," *Statistical Methodology*, vol. 3, no. 3, pp. 234–251, 2006.

[4] G. Shakhnarovich, J. W. Fisher, III, and T. Darrell, "Face recognition from long-term observations," in *Proc. 7th European Conf. Computer Vision-Part III*. London: Springer-Verlag, pp. 851 – 868, 2002.

[5] A. Roy and R. Khattree, "Discrimination and classification with repeated measures data under different covariance structures," *Communications in Statistics - Simulation and Computation*, vol. 34, no. 3, pp. 167–178, 2005.

[6] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 260–269, Apr. 1967.

[7] H. Tanaka, Y. Hirakawa, and S. Kaneku, "Recognition of distorted patterns using the viterbi algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-4, no. 1, pp. 18–25, Jan. 1982.

[8] T. G. Dietterich, "Machine learning for sequential data: A review," *Structural, Syntactic, and Statistical Pattern Recognition*, pp. 15–30, 2002.

[9] M. Grafmuller, "A first approach to trellis-based classification," in *Proceedings of the 2009 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vison and Fusion Laboratory*, 2009, pp. 175–185.

[10] P. Gupta, D. Doermann, and D. DeMenthon, "Beam search for feature selection in automatic SVM defect classification," in *Proc. 16th Int. Conf. Pattern Recognition*, vol. 2, 2002, pp. 212–215.

[11] H. Ney and S. Ortmanns, "Dynamic programming search for continuous speech recognition," *IEEE Signal Process. Mag.*, vol. 16, no. 5, pp. 64–83, Sep. 1999.

[12] R. T. Pack, P. Israelsen, and K. Sealy, "A co-boresighted synchronized ladar/EO imager for creating 3D images of dynamic scenes," in *Laser Radar Technology and Applications X*, G. W. Kamerman, Ed., vol. 5791. SPIE, May 2005, pp. 42–50.

[13] Canesta, "Canesta 101: Introduction to 3D vision in CMOS," [http://www.canesta.com/assets/pdf/technicalpapers/Canesta101.pdf], 2008.

[14] S. Göktürk, H. Yalcin, and C. Bamji, "A time-of-flight depth sensor - system description, issues and solutions," in *Computer Vision and Pattern Recognition Workshop*, Jun. 2004, pp. 35–35.
[15] B. M. Boldt, S. E. Budge, R. T. Pack, and P. D. Israelsen, "A handheld texel camera for acquiring near-instantaneous 3D images," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Nov. 2007.
[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer-Verlag, 2008.
[17] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer Verlag, 1996.
[18] G. Wahba, Y. Lin, and H. Zhang, *Advanes in Large Margin Classifiers*. Cambridge, MA: MIT Press, 2000, ch. GACV for support vector machines, pp. 297–311.

PLACE PHOTO HERE

Scott E. Budge received the B.S., M.S., and Ph.D. degrees in Electrical Engineering from Brigham Young University in 1984, 1985, and 1990, respectively.

Scott joined the faculty at Utah State University in January of 1989, where he has been involved in research into image data compression algorithms for transmission systems and space-based observation platforms. His current work involves research on methods for exploitation of full-waveform LADAR and EO images for 3D image applications. He is also involved in the development of high-performance image processing algorithms designed for implementation in hardware and VLSI systems.

PLACE PHOTO HERE

John A. Sallay concurrently received the B.S and M.S degrees in Electrical Engineering in 2009 from Utah State University. He is currently with Zeta Associates in Fairfax, VA.

PLACE PHOTO HERE

Ziang Wang received the M.S. degree in Electrical Engineering in 2012 from Utah State University. He is currently with Wavetronix in Provo, UT.

PLACE PHOTO HERE

Jacob H. Gunther received the B.S., M.S., and Ph.D. degrees in electrical engineering from Brigham Young University (BYU), Provo, UT, in 1994, 1994, and 1998, respectively. From 1992 to 1994, he was a Research Assistant with the Microwave Earth Remote Sensing Laboratory, BYU. From 1994 to 1995, he was with Lockheed-Martin, Manassas, VA, where he worked on target detection algorithms, sonar systems, and satellite digital communication systems. From 1998 to 2000, he worked at Merasoft, Inc., Provo, where he worked on speech recognition, speaker identification, acoustic echo, and noise cancellation using microphone arrays. He joined the faculty of the Department of Electrical and Computer Engineering, Utah State University, Logan, in 2000. His research interests include array signal processing, wireless communications, blind deconvolution and source separation, system identification, and machine learning.