5-4-2012

People Matching for Transportation Planning Using Optimized Features and Texel Camera Data for Sequential Estimation

Ziang Wang
Utah State University

PEOPLE MATCHING FOR TRANSPORTATION PLANNING USING OPTIMIZED FEATURES AND TEXEL CAMERA DATA FOR SEQUENTIAL ESTIMATION

by

Ziang Wang

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Electrical Engineering

Approved:

Dr. Scott E. Budge Major Professor	Dr. Jacob H. Gunther Committee Member
Dr. Edmund A. Spencer Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2012

Abstract

People Matching for Transportation Planning Using Optimized Features and Texel Camera Data for Sequential Estimation

by

Ziang Wang, Master of Science

Utah State University, 2012

Major Professor: Dr. Scott E. Budge
Department: Electrical and Computer Engineering

This thesis explores pattern recognition in the dynamic setting of public transportation, such as a bus, as people enter and later exit from a doorway. Matching the entrance and exit of each individual provides accurate information about individual riders such as how long a person is on a bus and which stops the person uses. At a higher level, matching exits to entries provides information about the distribution of traffic flow across the whole transportation system.

A texel camera is implemented and multiple measures of people are made where the depth and color data are generated. A large number of features are generated and the sequential floating forward selection (SFFS) algorithm is used for selecting the optimized features. Criterion functions using marginal accuracy and maximization of minimum normalized Mahalanobis distance are designed and compared.

Because of the particular case of the bus environment, which is a sequential estimation problem, a trellis optimization algorithm is designed based on a sequence of measurements from the texel camera. Since the number of states in the trellis grows exponentially with the number of people 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 good results for 68 people exiting from an initially full 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 sequential estimation algorithm produces an estimated traffic flow distribution which provides an excellent match to the true distribution.

(81 pages)

Public Abstract

People Matching for Transportation Planning Using Optimized Features and Texel Camera Data for Sequential Estimation

by

Ziang Wang, Master of Science

Utah State University, 2012

Major Professor: Dr. Scott E. Budge
Department: Electrical and Computer Engineering

This thesis explores pattern recognition in the dynamic setting of public transportation, such as a bus, as people enter and later exit from a doorway. Matching the entrance and exit of each individual provides information about the distribution of traffic flow across the whole transportation system. A texel camera is implemented and repeated depth and color measures of people are made. A large number of features are generated using the depth and color data for classification.

Because of the particular case of the bus environment, a trellis optimization algorithm is designed based on a sequence of measurements from the texel camera. Experimental results using real texel camera measurements show good results for 68 people exiting from an initially full 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 sequential estimation algorithm produces an estimated traffic flow distribution which provides an excellent match to the true distribution.

Acknowledgments

I give many thanks to my major professor, Dr. Scott Budge, who provides me with the excellent opportunity of working on the project and shares with me much experiences and ideas. I thank Dr. Jacob Gunther and Dr. Edmund Spencer for their time spent on this paper. I thank the Utah Transit Center for the sponsorship for this project. John Sallay gave a good start to the project and should be appreciated. Finally, I give my thanks to my wonderful wife, Meng, for her much support in completing the thesis.

Ziang Wang

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Transit systems are used every day and are an essential part of many people's lives. Transit authorities are very concerned with how to improve the quality of service while keeping the expense within control. Statistics are thus required for that purpose, such as the number of people that enter the bus at each stop. Traditionally, it is gathered through a manual process, either by the driver counting the number of people when they enter the bus, or by people using identification systems such as passenger cards for the riding record. Those methods are either labor-consuming or expensive for the transit authorities. Automatic People Counters (APC) are designed to fulfill the job and provide more accurate statistics while requiring little manual operation. There are some APC system producers, such as EZ People Counter [1] and Acorel [2], that uses infrared as the detection and counting media, and Sensource [3], that uses video cameras for counting.

An APC system could give statistics about the number of people that enter and exit the bus at each stop. With these stastistics, the ridership for a given time and route can be derived. This is very useful for the transit authority to make future service changes, but not specific enough because only the number of people riding the bus can be known, while any further information, such as when those people entered and exited the bus, is unknown. Knowledge of when people enter and exit is useful for the transit authorities since they can plan the route based on the flow of traffic instead of doing so only based on the number of people that use the stops. The traditional APC systems have difficulty deriving these statistics, since data from only the infrared or video cameras will not provide features to allow matching a person getting on the bus to the person getting off.

This thesis proposes an idea for the solution of the above situation by using a texel camera to derive features for matching. The texel camera is a combination of a time-of-

flight (ToF) camera and a color camera. The texel camera will capture much more data than infrared or video cameras. This data is used to generate the features of a person when he enters or exits the bus. After implementing pattern recognition algorithms, we can determine the length of time that each person rides on the bus as well as the number of the stops at which they enter and exit. The system will report many statistics in detail about individual riders and the specific route, helping the transit authorities to make more effective and optimized service plans.

1.1 Time-of-Flight Technology

A time-of-flight camera is a range imaging system based on the propagation of light, resolving the distance from the sensor to the interested objects by measuring the time that the emitted light propagates from the transmitting sensor to the receiving sensor. There are three types of ToF cameras developed [4].

- Pulsed light source with digital time counters
- RF-modulated light sources with phase detectors
- Range gated imagers

The first type uses a pulsed laser and spreads the laser pulse to the target with the transmitted optics. The receiving sensor pixels will capture the reflected pulse and compute the round trip time for each pulse. The second type works by modulating the outgoing beam with a carrier, then measuring the phase shift of the carrier when receiving it. The third type has a shutter in front of the image sensor, which blocks the reflected pulse according to its time of arrival and computes the distance using the camera range determined by the round trip of the light pulse, as well as the amount of pulse that is received and blocked.

The ToF camera, which is used for this thesis, is provided by Canesta Inc. [5], and belongs to the second type of the cameras. Rather than emitting pulsed light, the camera has a modulated light source to give the emitted light a modulation envelope. To measure the distance, the phase of the modulation envelope of the transmitted light is measured

when it reaches the object and returns to the detecting sensor. The distance between the sensor and the object can be calculated using the phase shift of the transmitted and received modulated light.

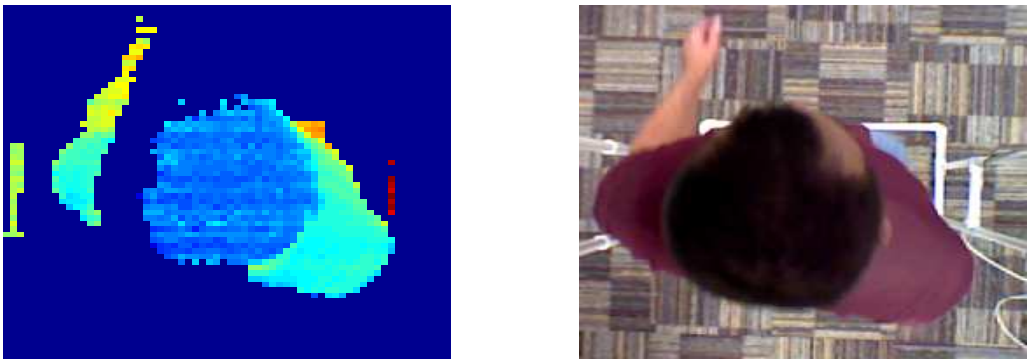
The ToF camera used in this thesis relies on continuous modulation rather than laser pulse compared with the other two types of systems. It makes the system less dependent on the detection accuracy of the reflected pulse, but gives constraint to the unambiguous range of measurement due to the periodicity of the emitted signal and the cyclic nature of phase. By using a light source that illuminates the entire scene and an array of detectors, it makes the measurements of distance complete with one shot, which is suitable for real-time applications.

1.2 Texel Camera and Image

A texel camera is the combination of a ToF camera and a electro-optical (EO) camera. They are mounted together. For the camera used for this thesis, the cameras are mounted so that they are 90 ° from each other on a supportive structure, and a cold mirror is used to separate the incoming light into two bands, which are at 45 ° between the two cameras. The cold mirror is reflective for the visible wavelengths while transparent to the infrared wavelengths. Thus it is capable of separating incoming wavelengths, directing the visible wavelengths into the lens of the color camera and the infrared wavelengths into the lens of the ToF camera. Further information on the configuration of the texel camera can be found in Boldt [6].

By co-boresighting and synchronizing the two cameras, both depth and color data can be retrieved from the texel camera when a shot is made. An example of the fused depth and color images from a single shot of the texel camera are given in Figure 1.1. Because the ToF camera is able to gather depth data of a scene at the same time, as described in Section 1.1, all pixels and points are captured simultaneously. This makes the data collection easy, since any post processing to fuse the raw data is avoided.

The texel camera used for this thesis is well designed, but it has its weak points. The first are low frame rate and resolution. The frame rate of the ToF camera from Canesta



(a) (b)

Fig. 1.1: 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.

Inc. is about 8 frames per second (fps), which is not high enough to make it suitable for real-time jobs in a frame rate of 25 - 30 fps [7]. The depth image resolution is 64x64, which is much smaller than the color images. The second is that the characteristics of the cold mirror is not perfect enough to reflect all visible wavelengths, affecting the color images with the ToF illumination light.

The solution to the first weak point is to build the texel camera using a new ToF camera. The new ToF cameras are delivered with higher image resolution and higher frame rate, such as 40 fps at 200x200 resolution by PMD Technologies Inc. [8]. The second weak point can be improved by using postprocessing algorithms on the color images, which will be described in Chapter 2. These improvements are not available at the writing of this thesis.

1.3 Previous Work

1.3.1 People Counting

Automatic people counting has become a very active research area and there have been many methods proposed. Chen et al. [7] use a zenithal camera in the bus for capturing the

passengers bi-directionally and get a 92% counting accuracy. Yang et al. [9] use a single-camera-based system for a highly crowded bus environment and an online clustering-based people counting network, which gives 96.5% counting accuracy.

An automatic people counting system using a texel camera is explored by Sallay [10] and is not the subject to be explored in this thesis. However, the people matching procedure proposed here will be based on Sallay's people counting results, and feature vectors will be generated from data that comes from the counting system.

1.3.2 Feature Selection

Feature selection is an essential part of pattern recognition. A large amount of data about objects, events, and phenomena are generated and stored because of the improved capability of data processing and acquisition as well as the decreasing cost of data storage. Data are used to represent interested objects in the form of features. However, not all of the features generated will be helpful for pattern recognition. They may be irrelevant to each other or redundant within the feature set. When used in pattern recognition algorithms, unnecessary features will increase the processing time and computational burden because of the extra size of the search space, but the performance of the algorithm may not increase. Even worse, due to the curse of dimensionality, increasing the size of the feature set will sometimes degrade the performance. Consequently, feature selection is widely used in pattern recognition, machine learning, and data mining, to improve performance by enhancing processing speed, alleviating the curse of dimensionality and increasing generalization capability. Feature selection can also help people understand which features are important and how they are related.

Numerous feature selection algorithms have been proposed, and these can be put into two categories: filter and wrapper [11]. Filter feature selection algorithms are independent of classifiers, by ranking the features using certain statistical criteria of the data's intrinsic properties. Features with highest ranking values are selected. Methods such as t-test, mutual information, and principal component analysis are referred to as the filter methods. Filter methods execute faster but are not efficient enough to solve the irrelevance and

redundancy among all features, because every feature is treated separately. It is also difficult to know the exact number of features for the applications to be used, because the methods relate to no classifier, making it unable to give a quantitative threshold for the optimal number of features.

Different from the filter feature selection algorithms, the wrapper feature selection algorithms use a classifier, or a criterion function, to test the feature subsets. During the execution of the algorithms, the subset in the feature set is searched and statistical information such as accuracy is generated by inputting a feature subset into the criterion function. By using a subset, the relevance and dependency of the features is taken into account and the redundancy is lowered. The wrapper feature selection algorithms thus provides better and more effective selection results than those of the filter counterpart. Since a search of the subset within the feature set is required, the wrapper feature selection algorithms can be classified into two categories according to the search methods: randomized and deterministic.

The randomized wrapper algorithms, such as the ant colony optimization (ACO) [12] and genetic algorithm (GA) [13], are developed to solve large scale combinatorial problems, searching the feature subset in a randomized sense. These algorithms are able to perform effectively considering the irrelevance and redundancy without making any assumption on behavior of the criterion functions. But the downside of these algorithms is that they are expensive to compute. As a result, the deterministic wrapper feature selection algorithms that are relatively less computationally expensive will be used for the feature selection task of this thesis.

1.3.3 Classification Algorithms and Multiple Measurements

A classifier is used to generate a pattern label prediction for an incoming unknown feature vector $\boldsymbol{l}$ together with the z feature vectors $\{\boldsymbol{l}_0, \boldsymbol{l}_1, \cdots, \boldsymbol{l}_m, \cdots, \boldsymbol{l}_z\}$ which have previously learned labels. To minimize the classification error, the unknown feature vector $\boldsymbol{l}$ should be labeled using one label of the feature vectors $\boldsymbol{l}_m$ already learned, with which the posterior probability $P(\boldsymbol{l}_m \mid \boldsymbol{l})$ is the largest among others. The classification process is re-

ferred to as Maximum a posteriori (MAP) and the computation of the posterior probability can be done using Bayes Theorem

$$\begin{aligned} P_{MAP} &= \max_{m\in\{1,\ldots,z\}} P(\boldsymbol{l}_m \mid \boldsymbol{l}), \\ &= \max_{m\in\{1,\ldots,z\}} \frac{P(\boldsymbol{l} \mid \boldsymbol{l}_m)P(\boldsymbol{l}_m)}{P(\boldsymbol{l})}, \end{aligned} \tag{1.1}$$

where P_{MAP} is the MAP probability, $P(\boldsymbol{l} \mid \boldsymbol{l}_m)$ is the likelihood of the unknown feature vector, $P(\boldsymbol{l}_m)$ is the prior probability of the known feature vectors, and $P(\boldsymbol{l})$ acts as a normalization constant. The $P(\boldsymbol{l}_m)$ is usually calculated using the relative frequency of the known feature vectors within the data set, and $P(\boldsymbol{l})$ is often omitted or approximated according to different methods. Estimating the $P(\boldsymbol{l} \mid \boldsymbol{l}_m)$ is a difficult task, and different attempts to estimate $P(\boldsymbol{l} \mid \boldsymbol{l}_m)$ will provide the problem with different solution methods. Examples are linear discriminate analysis (LDA), quadratic discriminate analysis (QDA), k nearest neighbor (KNN), and multilayer perception classifier (MLP) [14].

For this thesis, repeated and independent measures of a person going through the field of view (FOV) are made, which can be used together to perform much more robustly than estimation with single measurement. Lix and Sajobi [15] give a review of the discriminant analysis procedures for univariate and multivariate repeatedly measured data. Roy and Leiva [16] develop classification rules for multivariate repeatedly measured data with structured correlations on repeated measures on both spatial and temporal data. Krzysko and Skorzybut [17] propose new classifiers under the assumptions of multivariate normality for multivariate repeatedly measured data with Kronecker product covariance structures. The LDA classifier with multivariate repeated measures data will be discussed in Chapter 2.

1.3.4 Sequential Data, Viterbi Algorithm, and Trellis

Ability to deal with noise or other distortion is always a criteria for judging the capability of a classifier, especially when sequential data need to be classified. The Viterbi algorithm [18] is used for computing the probability of a sequence of observed events, by finding the most likely sequence of hidden states that forms a sequence of observed events.

The Viterbi algorithm is widely used in the field of decoding trellis-coded modulation, automatic speech recognition and decoding convolutional codes. Tanaka at el. [19] use the Viterbi algorithm and a modified trellis on the recognition of distorted patterns. Dietterich [20] reviews the statistical learning problems involving sequential data, mentioning the use of Viterbi algorithm for computational efficiency. Grafmuller [21] gives the brief introduction to the trellis-based classification method, and demonstrates it using an optical character recognition (OCR) example.

A trellis diagram is used to implement the Viterbi algorithm's dynamic programming. However, searching the entire trellis is computationally intensive and prohibitive as the trellis expands. Thus, a beam search method [22] will be used together with the trellis, balancing the performance and computational burden.

1.4 Purpose of Thesis

The purpose of this thesis is to develop algorithms to generate feature data from the existing people counting system using the texel camera, to process the feature data using feature selection algorithms, and to execute automatic people matching. The matching process will explore the sequential estimation of people exiting a bus, and improve the classification accuracy using the trellis optimization algorithm.

After people matching, the results will be used to generate statistics about individual riders and traffic flow. This will show that the automatic people matching algorithm is capable of classifying people accurately and will provide more useful and detailed information than would be generated by using the people counting system alone.

1.5 Overview

The feature selection algorithm and the automatic people matching algorithm are two separate parts and are not executed at the same time. Features are generated and stored by the people counting system when people in the training data enter and exit the bus. The optimal features are selected once from the training data, and used in the matching from then on. A data set using the optimized features will be generated and classification

will be done using the automatic people matching system. The block diagram of the people matching system is shown in Figure 1.2.

Chapter 2 explores the full details about the people matching system, from the design to the implementation as well as performance analysis. Chapter 3 describes the feature selection algorithms, in which the design of the algorithm used will be mentioned and the advantages and disadvantages will be analyzed and compared. Chapter 4 does simulations on the algorithms and results are discussed and analyzed. Chapter 5 gives conclusions and ideas for future research.

1.6 Thesis Contribution

A list of the contributions from this thesis is given below.

- Feature generation using the depth and color data from the texel camera.
- Implementation of the feature selection algorithm for selecting the optimized feature subset, with comparisons between the traditional and new methods.

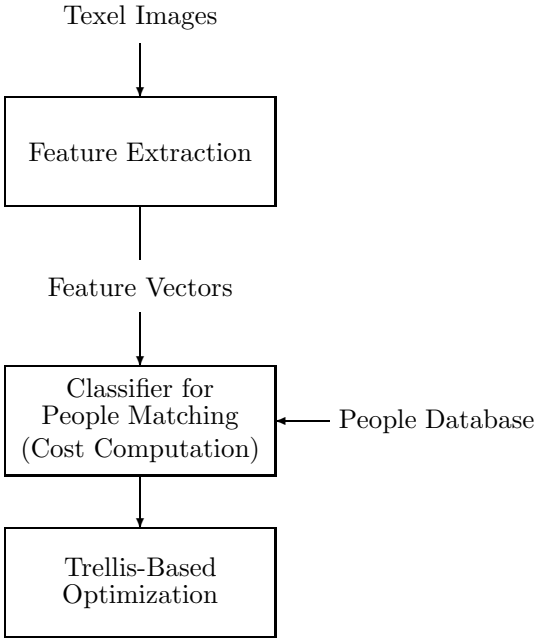


Fig. 1.2: Block diagram of system.

- Design and implementation of the trellis optimization algorithm for people matching, as well as an in-depth analysis of it.
- Generation of the traffic flow distribution, which provides more information for transit authorities than that from the current people counting systems.

Chapter 2

Sequential Estimation and Trellis Optimization Algorithm

After feature selection, the optimized features will be used to form feature vectors for the classification task in the bus environment. Because of the particular characteristics of the bus environment, the classification is considered to be a sequential estimation problem and a trellis optimization algorithm is used. Section 2.1 explains the situation. Section 2.2 explores the trellis optimization algorithm and its implementation. Section 2.3 introduces the beam search for the trellis to balance the performance and computation. Section 2.4 gives a detailed example on the execution of the trellis. Section 2.5 discusses the programming of the algorithm.

2.1 Problem Description

In a real bus environment, the classifier tries to correctly associate each exiting person with one who had entered previously. However, because of existence of noise and the difficulty in designing a perfect classifier, classification errors will always exist. If a classification decision is made immediately after the classification process is completed and the decision is incorrect, there will certainly be matching errors later on. For example, if there are only two people to be matched, an incorrect decision will certainly cause the other decision to be incorrect.

This problem can be mitigated using a sequential estimation technique, by considering the relationship between the current matching decision and the future matching results. As mentioned above, decisions made previously can have impacts on future decisions, which makes it necessary to incorporate previous decision making information into the future decision making process. If a decision is certain enough, then it should be made immediately. If a decision is unsure, then decision making should be postponed until more information

can be gathered or other strong decisions have been made.

2.2 Design of Trellis-Based Classification Algorithm

Sallay [10] derives the LDA, QDA, and KNN classifier using repeated measures and makes a comparison. Because there are only a few frames of each person, there is not sufficient data for computation of the covariance matrices of individual people. Therefore, the entire person database will be used to compute the covariance matrix. Thus, the LDA classifier will be used for the feature selection task, which is

$$\delta_{\hat{\boldsymbol{\mu}}_i}(\mathbf{f}) = \sum_{m=1}^{z_i} (\mathbf{f_m} - \hat{\boldsymbol{\mu}}_i)^T \hat{\mathbf{R}}^{-1} (\mathbf{f_m} - \hat{\boldsymbol{\mu}}_i), \tag{2.1}$$

after derivation and simplification, where z_i is the number of the ith person's feature vectors from exiting frames, $\boldsymbol{f}$ is the person's feature vector set from exiting frames and $\boldsymbol{f}_m$ is each one of the feature vectors in the set, $\hat{\boldsymbol{R}}$ is the covariance matrix computed using entering feature vectors for all persons in the database, and $\hat{\boldsymbol{\mu}}_i$ is the mean feature vector computed for the i_{th} person who entered the bus. As a result, the $\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f})$ is the sum of Mahalanobis distance of all feature vectors of a person's exiting feature vector set with the mean feature vectors of the ith person's entering feature vector set. The computed distance value will be sorted, and the index i of the mean vector that minimizes $\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f})$ will be assigned to the person who exits the bus.

To incorporate previous information into the future decision making process, the problem can be expressed as

$$\arg\max_{\boldsymbol{a}} P(F_1 = \boldsymbol{f}_1, \cdots, F_i = \boldsymbol{f}_i, \cdots, F_n = \boldsymbol{f}_n \mid A_1 = a_1, \cdots, A_i = a_i, \cdots, A_n = a_n), \tag{2.2}$$

where n is the number of people, i is the index of people, a_i is a person, $\boldsymbol{f}_i$ is the set of feature vector of people a_i, $A_i = a_i$ stands for the event that the ith person to leave the bus is person a_i, $F_i = \boldsymbol{f}_i$ stands for the event that the ith person's feature vectors $\boldsymbol{f}_i$ are collected, and $\boldsymbol{a}$ is a vector with n people. To simplify the expression, $F_i = \boldsymbol{f}_i$ is expressed

to be $\boldsymbol{f}_i$ and $A_i = a_i$ is expressed to be a_i, making the problem's expression to be

$$\arg\max_{\boldsymbol{a}} P(\boldsymbol{f}_1, \cdots, \boldsymbol{f}_i, \cdots, \boldsymbol{f}_n \mid a_1, \cdots, a_i, \cdots a_n). \tag{2.3}$$

As each observation $\boldsymbol{f}_i$ is independent of each others and only depends on the class it comes from, the equation can be simplified to be

$$\arg\max_{\boldsymbol{a}} P(\boldsymbol{f}_1, \cdots, \boldsymbol{f}_i, \cdots, \boldsymbol{f}_n \mid a_1, \cdots, a_i, \cdots a_n) = \arg\max_{\boldsymbol{a}} \prod_{i=1}^{n} P(\boldsymbol{f}_i \mid a_i). \tag{2.4}$$

Because of sparsity of the data, LDA will be used for the task. Each $\boldsymbol{f}_i$ will be considered from the Gaussian random distribution with mean $\hat{\boldsymbol{\mu}}_{a_i}$, and covariance matrix $\hat{\boldsymbol{R}}$. The covariance matrix is computed using the entering feature vectors of all people. Using logarithm in (2.3),

$$\arg\max_{\boldsymbol{a}} log \prod_{i=1}^{n} P(\boldsymbol{f}_i \mid a_i) = \arg\max_{\boldsymbol{a}} \sum_{i=1}^{n} logP(\boldsymbol{f}_i \mid a_i), \tag{2.5}$$

and using the expression for the Gaussian distribution

$$logP(\boldsymbol{f}_i \mid a_i) = -logC - \frac{1}{2} log|\hat{\boldsymbol{R}}| - \frac{1}{2} \sum_{m=1}^{z_i} (\boldsymbol{f}_{i,m} - \hat{\boldsymbol{\mu}}_{a_i})^T \hat{\boldsymbol{R}}^{-1} (\boldsymbol{f}_{i,m} - \hat{\boldsymbol{\mu}}_{a_i}), \tag{2.6}$$

where C is the constant value, m is the index of feature vectors, z_i is the number of feature vectors for person i, $\boldsymbol{f}_{i,m}$ is the mth feature vector of person i. In this way, the solution of the sequential estimation problem is associated with the classifier using repeated measurements. Equation (2.5) can be simplified by omitting the constant value and scale factor,

$$\arg\min_{\boldsymbol{a}} \sum_{i=1}^{n} \delta_{\hat{\boldsymbol{\mu}}_{a_i}}(\boldsymbol{f}_i), \tag{2.7}$$

where $\delta_{\hat{\boldsymbol{\mu}}_{a_i}}(\boldsymbol{f}_i)$ is given by (2.1).

Equation (2.7) can be interpreted to mean that the sum of the Mahalanobis distance of the feature vectors of exiting of all people with the mean feature vectors of entering of all

people is to be minimized. The best option for doing this is to calculate every combination of the elements in a and make comparisons to select the one that minimizes the sum of distance value. However, this is impossible because of the number of possible combinations if $\boldsymbol{a}$ is large enough to be prohibitive, which is $n!$ for n people.

In order to reduce the computational cost while still being optimal, the trellis optimization algorithm is used. All possible ordering combinations are searched through the trellis. Before using the trellis, it is introduced and illustrated using an example.

In Figure 2.1, there are four people on the bus, and the black dots, lines and labels in the figure stand for different components of the trellis.

- State: Stores the indices of people that are on the bus. The states are represented by black dots and the indices of people are beside each state showing which person is on the bus.

- Step: Stores all the states where there is a person exiting the bus and one or more people entering the bus. A real bus stop may have multiple steps since there may be more than one person getting off the bus. In the figure, there is only one person exiting the bus, and none entering. The S_i labels at the bottom of the figure indicate the step indices. In the algorithm, a person exiting will generate a step, while one or more people entering will also generate a step, since people's entering the bus will not cause any computation or classification.

- Edge: Indicates the index of people exiting the bus, and links two steps next to each other. The edges are represented using straight lines in the trellis, with the numbers showing the indices of people that exit. Whenever a person exits the bus, an edge with an edge label will be created. When people enter the bus, an edge without label will be created.

- Path: Multiple edges linking different steps from the start to the end of the trellis form a path. A path is used for decision making, since when all people exit the bus,

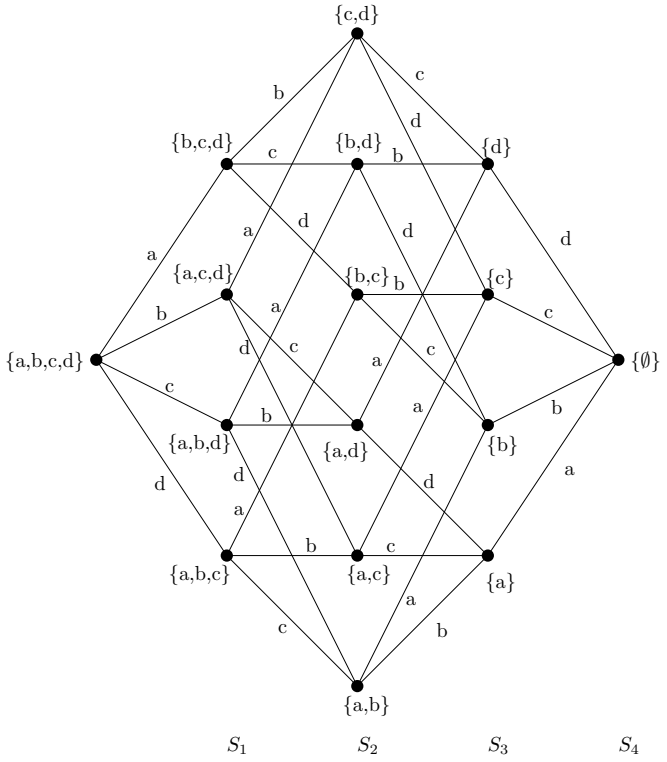


Fig. 2.1: A simple trellis with four people.

people in the path with the minimum accumulated cost will be the winning path and all states in the path will be decided.

- Distance: Whenever a person exits the bus, the feature vector set from exiting frames will be compared with all feature vector sets from entering frames. The comparisons are implemented using (2.1) using multiple measurements and a distance is the value returned from (2.1).

- Cost: Every path has a related cost value, which is accumulated with the distance from a person exiting and a person on the bus when the trellis extends by a step. For example, if there are two people on the bus and a person exits, there will be two edges extending from the path by a step. The two edges will both have a distance, which will be added to the cost of the path respectively. After extension, there will be two paths existing and each has an accumulated cost. An accumulated cost is used for decision making when the trellis stops extending.

Besides the components above, there is another component that is not shown in the figure. A pool is used to store the indices of people that have not been decided using the trellis. When there are people entering the bus, they will be put in the pool for classification. When a person is exiting, his exiting feature vectors will be used to compute the Mahalanobis distances with the feature vectors from entering frames which belong to the people in the pool. The people in the pool can either stay on the bus or can exit the bus. This is how the trellis incorporates the previous information into the current decision making process. In a step, a decision is not immediately made, although the person may have exited the bus. By saving the people that are not confidently decided, the incorrect decision making can be postponed and possibly corrected in the future. If further correct matches are strong and confident, the incorrect previous matches will be remedied since the pool will be shrinking and the incorrect matching pairs may not exist.

The trellis is also able to deal with people who enter the bus. Figure 2.2 shows an example of people exiting and entering the bus. In the trellis, people entering and exiting are regarded as events; a person exiting and one or more people entering are two types of events. This is because the trellis will extend whenever a person exits the bus, but will not extend when multiple people enter the bus, since only the pool expands.

There are also some constraints for the trellis. In a path, there will be no identical edge label for two different edges. The reason is that the same person can not exit the bus twice. Another constraint is that two paths can not have the same set of edge labels with different orderings. For example, if a path has three edges with labels $\{1, 2, 3\}$, there should not be another path which has edge labels $\{1, 3, 2\}$. This can be checked by examining whether two paths end at the same state of a step. If two paths converge to the same state, they will have the same set of edge labels. Whenever the trellis extends by adding a new step, the paths will be checked and compared. If two paths have the same set of edge labels, the path cost will be compared and the one with larger cost will be deleted. This means that there will only be one path extending back from any state.

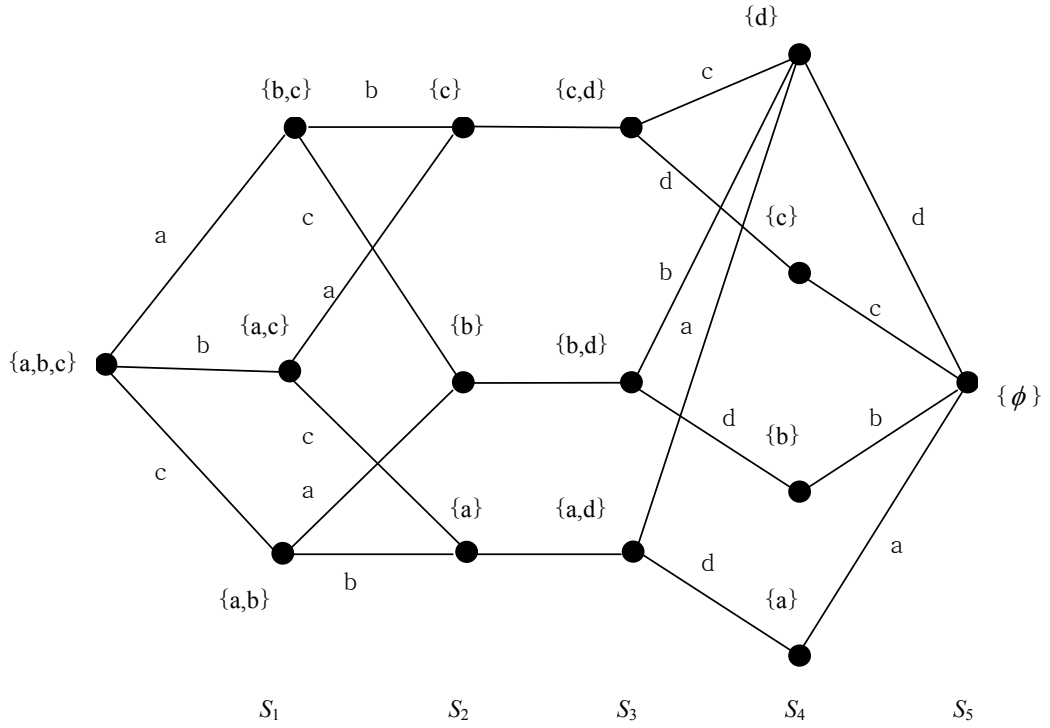


Fig. 2.2: Trellis representing an entering person.

2.3 Beam Search

Although the trellis helps reduce the computational cost by reducing the search space, the computational cost is still very high if the number of people in the pool is large. This situation can happen if the classifier used is not sufficient to give confident decisions or if the data is distorted by noise, resulting in a growing pool size as people enter the bus. Because the extension of the trellis depends on the size of the pool and the number of people exiting, if the size of the pool is large then more edges will be created from a state, and this will generate more paths.

In order to further lower the computational cost while maintaining the performance and accuracy, a beam search is used by adding a beam width to constrain the number of states in a step. As mentioned by Jang and Lin [23], there are three kinds of beam search categories.

- Rank based: At each step, the top W paths compared using path costs are kept while others are deleted.

- Cost based: At each step, the paths whose path costs are smaller than a threshold T are kept.

- Combination of the two above: At each step the paths which are smaller than a threshold T and have costs smaller than the top W are kept.

Here the rank-based beam search method is used. A beam with width W will be applied to the number of states in each step. The beam search method is used with the sorting of the paths according to the path costs. When the trellis is extended by adding a new step, the newly generated paths will be sorted by the accumulated costs and the number of paths will be counted. If the number of paths is smaller than W, the paths will be kept. If the number exceeds W, the top W paths with smaller costs will be kept and others will be deleted. During the next extension of the trellis, only the ending states of the W remaining paths will be used for extension. The process is repeated until all people exit the bus and the trellis stops extending. The beam search method provides a tradeoff between accuracy and computation, and the performance with different beam widths will be explored in Chapter 4.

2.4 Decision Making Process with Beam Search

Besides causing a decrease in computational cost, the beam search method also provides chances to prune the trellis, which will allow more ways of making decisions during the classification process.

The pruning of the trellis is accomplished by using a variable E stored in the data structure of states, indicating how many states in the next step are linked to the states in the current step, or how many edges are extended from the state. When a new step is added, the E for each state in current step is filled. However, because of the beam width, when some paths are not added during the beam search process, some states in the current step will not have states in the new step linked to them. This will make E equal 0 for those states in the current step, which will be deleted. Then by finding the states in the previous step linked to the states in the current step with E equal to 0, the E for these states in the

previous step can be decreased. Recursive processing is carried out on all states in every step of the trellis and the number of states in each step can be lowered. Thus, whenever a new step is added to the trellis, all previous steps will be pruned.

After trellis pruning when a new step is added, the trellis is checked to see whether one of the following decision making criteria is met.

- Delete a new added step: This takes place when a step is added, and all the edges linking a previous step and a new step have the same labels. This is because the Mahalanobis distance between a person's entering feature vectors and leaving feature vectors is so small that the cost of other people's feature vectors will not surpass it, causing them not to be added because of the beam width. By deleting the new step added, the E for each state of current step is cleared and the person getting off the bus is immediately decided and the corresponding person is removed from the pool.

- Delete an inner step: After pruning the states within a step, there is possibility that edges linking two steps within the trellis have the same label, resulting in a decision for a person in the pool. By choosing the person, the step is deleted and the E of the previous steps is assigned according to the linking between the previous step and the next step which connect the inner step being deleted.

- Trace an inner path: This is the subset case of deleting an inner step, with different processing. After pruning the states within a step, there is a possibility that only one edge will link the two steps within the trellis. Because all paths from previous steps converge to this edge, all people in the path to this remaining edge will be decided by tracing back from the edge.

- Trace the final path: This happens when all people on the bus have exited. When people are correctly counted, the trellis will always end with a single edge. At this time, all people in the trellis that have not yet been decided by the previous three ways of decision making will be finally decided, by tracing back from the last edge along the path with the minimum accumulated cost.

The four ways of decision making will be executed under different conditions and contribute differently towards the accuracy of the trellis-based classification algorithm, which will be discussed in Chapter 4.

2.5 Trellis Example

In this section, an step-by-step example about the execution of the trellis is shown. The example is produced from a actual simulation on five people in the initial pool and a beam width of six is used. In this example, the extension of a trellis, trellis pruning because of the beam width, decision by deleting a new added step, and decision by tracing the final path are illustrated. The trellis starts with the initial pool with five people, and the state which has five people is in step 0.

2.5.1 Extension of Step 1

Five people with indices $\{10, 21, 41, 44, 51\}$ are drawn randomly from the 68 people database. From Figure 2.3, when a person exits the bus, the Mahalanobis distance with all people in the initial pool will be computed, which will extend the trellis. A number in the parentheses beneath each state (which is represented using black dots) is the number of edges extended from the state, which is E as mentioned in Section 2.4. For example, there are five edges extended from the state in the 0_{th} step and currently no edge from the states in the 1_{st} step. The cost $\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f})$ for each edge is not shown.

2.5.2 Extension of Step 2

In step S_1, each state has four people. Therefore, each state will have four edges extended from it. However, because of the constraint mentioned in Section 2.2, paths with the same set of edge labels but different order will be compared. For example, from the first state in S_1, there will be an edge extended with label 41, and one with label 51 from the second state in S_1, as shown in Figure 2.4. In this case, the two paths has the same set of edge labels (51 and 41), but different orders ($\{51, 41\}$) and ($\{41, 51\}$),ending at the same state. Thus, the two paths will be compared and the one with larger accumulated cost will

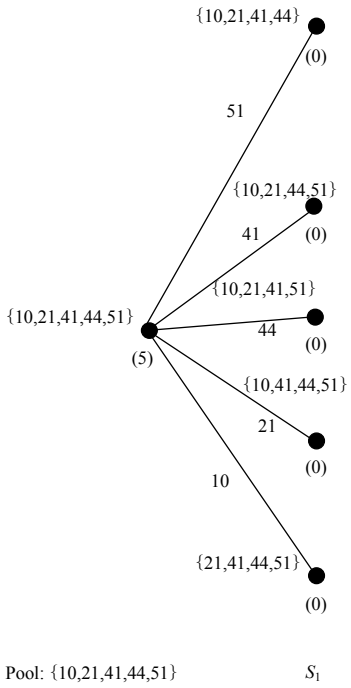


Fig. 2.3: The first step of the trellis.

be deleted. This is shown as a dotted line in Figure 2.4 for each state.

Also, because at most, four edges will be extended from the five states in S_1, the total number of edges from S_1 will be at most 20. This is larger than the beam width, which is 6 in the example. Therefore, the six edges which produce paths with smaller accumulated cost will be retained, while others are deleted.

From Figure 2.4(a), the last two states in S_1 do not have edges extended from them, since $E = 0$ for both when the trellis finishes extension to S_2. Therefore, the last two states in S_1 are deleted, as shown in Figure 2.4(b). Also, note that the E of the states left in step S_1 is updated according to the number of edges extended.

2.5.3 Extension of Step 3

From Figure 2.5(a), it can be observed that the constraints of beam width and path label mentioned before make all edges extended from S_2 to S_3 have the same label 10. Therefore, the decision criterion for deleting a new added step will be met. The step S_3 will be deleted and the decision will be recorded, which is that person 10 is the third to exit the

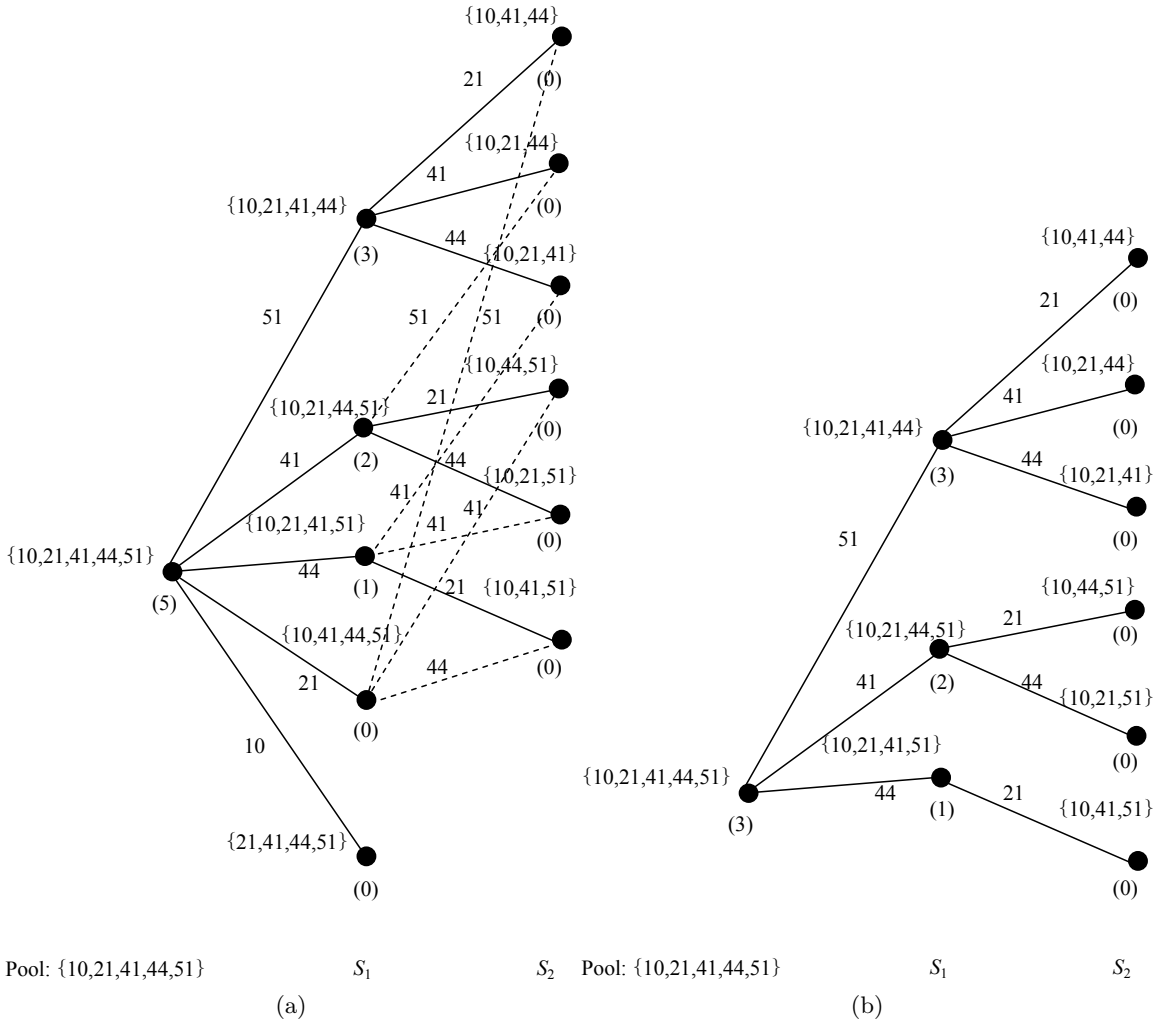


Fig. 2.4: Processing on Step 2. (a) Before pruning (higher cost extensions shown with dotted line), (b) After pruning.

bus.

After deleting S_3, the E for states in S_2 will be updated, and the initial pool is updated by deleting the person 10. Figure 2.5(b) shows the trellis after making a decision and deleting a new added step.

2.5.4 Extension of Step 4

The processing of S_4 is the same as that of S_2. The trellis before and after pruning is shown in Figure 2.6.

2.5.5 Extension of Step 5

At S_5, all people have exited the bus, and the trellis will collapse to the last state, as shown in Figure 2.7. Since all paths extends to the same state, the accumulated cost of all the paths will be compared and the path with the smallest cost will be kept for decision making. As mentioned in the decision criterion, the criterion for tracing the final path is met. Then the decisions along the final path are made and recorded, combining the decision made previously in S_3. The decisions made by using the trellis are $\{51, 21, 10, 41, 44\}$.

2.6 Programming the Trellis

In the big picture, the trellis consists of paths, and an accumulated path cost is related to a path. There is no need to store all the costs within the path, and only the accumulated cost vector is stored. The new costs are added to it whenever a new step is added to the trellis and the paths are extended. All the paths' costs are sorted so that when the beam width W is reached, the paths with larger costs can be deleted. The sorting method used is an insertion sort, since the extension of the paths to the new states are processed one at a time. When a new path is created, it will be compared with the existing paths for the edge label set. If two paths have the same set of labels, the path with the larger cost will be deleted. The unique path will then be added to the proper position in the trellis, by comparing the path cost with all path costs stored, starting from the last one. If the beam width has been reached and the cost is larger than that of the last path, the new path will

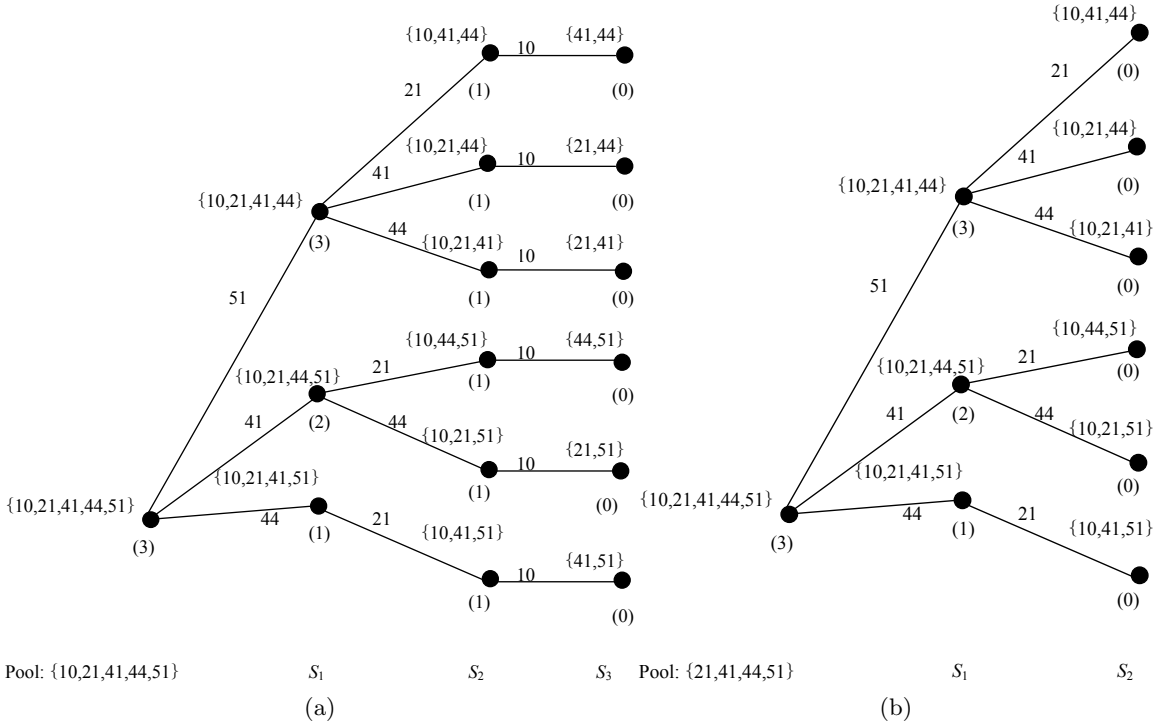


Fig. 2.5: Processing on Step 3. (a) Before pruning, (b) After pruning.

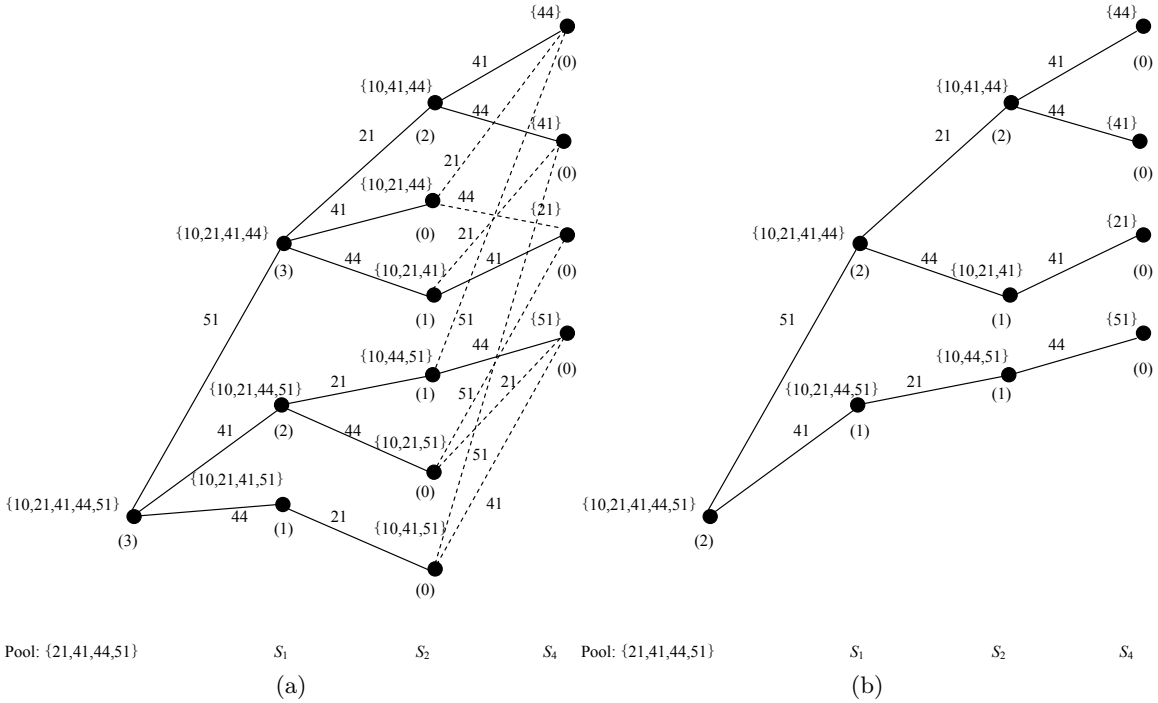


Fig. 2.6: Processing on Step 4. (a) Before pruning (higher cost extensions shown with dotted line), (b) After pruning.

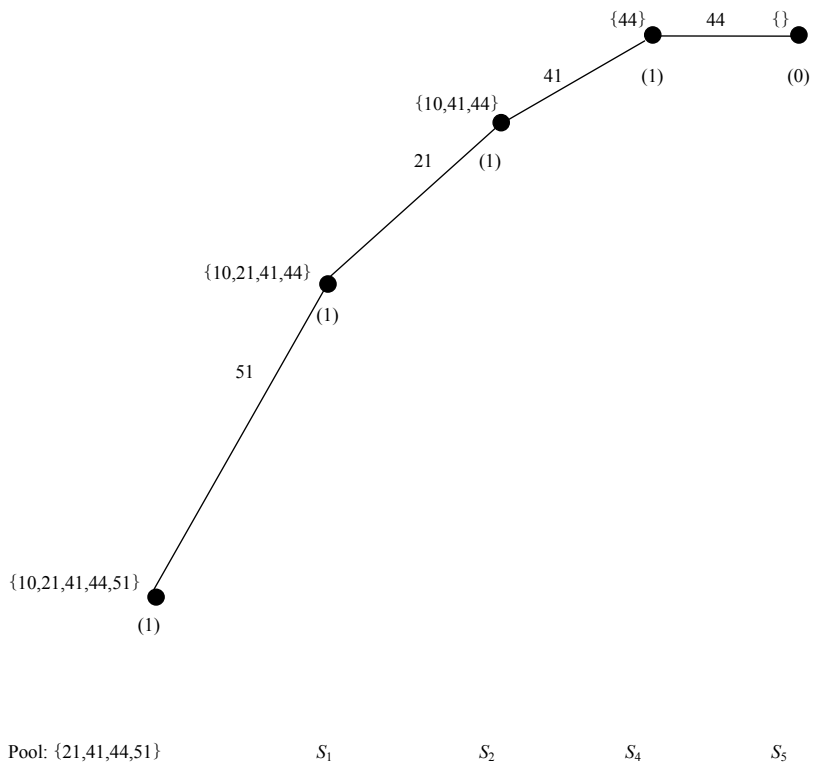


Fig. 2.7: The fifth step of trellis and final path.

not be added. If the beam width has been reached and the cost is smaller than that of the last path, it will be added and the last stored path will be deleted.

The steps compose the trellis; thus, the steps will be the main data structure for the trellis. The steps are connected by way of a linked list such that they are growing in one direction, and can be deleted so that the two steps beside it can be linked. An example is that three steps $\{1, 2, 3\}$ are in a path. When step 2 is deleted, steps 1 and 3 can be linked.

A step contains states, and all states that belong to the step will be stored. The number of states is updated within the step. If the number exceeds the beam width W, the path cost will be compared and paths with larger accumulated cost ranking greater than W will be excluded. There will be no need to store the states that each state links forward to, but the one state that it links back to will be stored as well as the label of the edge that links the two states. In this way, the set of edge labels of a path in the trellis can be maintained by tracing back the path recursively, without storing the edge labels in a separate piece of memory.

Also mentioned in Section 2.4 are the number of edges from a state that are stored in a variable E. By updating E whenever a new step is added to the trellis, the trellis is able to be pruned by checking whether there are any states that have $E = 0$. By using the variable, the pruning of the trellis is easily implemented.

Chapter 3

Feature Selection

This chapter describes the feature selection algorithms for selecting the optimized features generated by the people counting system. Section 3.1 gives a brief description about the simulation setup and data acquisition. Section 3.2 illustrates all features to be selected and the generation. Section 3.3 shows the design of the algorithms being used and describes the implementation. Section 3.4 explains the criterion functions used. Section 3.5 gives the feature selection results and analysis.

3.1 Simulation Setup and Data Acquisition

The texel camera is mounted on top of a portable frame built to simulate the bus entrance, which is about 96 inches high as measured from a 2007 Gillig ski bus. Mounting the texel camera in this way guarantees that in real bus environment, people will not be disturbed by the illumination of the texel camera, and that people can be measured repeatedly as they walk through the FOV of the texel camera. The real bus entrance has no step, making it easy for the simulation since the frame can be put on any flat ground. The camera is located near the middle of the ceiling, so that when people go through, the relative distance between the texel camera and the people will stay the same within the FOV. Sample images are given in Figure 1.1.

The ToF camera and color camera are mounted on the structure so that they can measure the same scene at the same time. However, the frame size of the two cameras are not the same, since the Canesta ToF camera was one of the first ToF cameras manufactured and the 64x64 dimensions of the depth images are much smaller than those of the color images which are 1280x1024. In this case, the color images are downsampled to a resolution of 256x256 so that higher resolution can be attained while computation cost can be lowered.

When people go through the FOV of the texel camera, the direction of people's movement is predetermined. In the simulation, moving from the top to the bottom of the FOV is considered to be entering the bus, while moving from the bottom to the top of the FOV is considered to be exiting the bus.

As mentioned in Section 1.2, the configuration of the ToF camera makes the frame rate too low for real-time applications. The low frame rate will cause the images to blur as people go through the FOV using their normal walking speed. In order to solve the problem, the people in the simulation walk at $\frac{1}{3}$ of a normal speed, in order to simulate the real-time frame rate of a next-generation texel camera.

After setting up the simulation environment, people walk through the frame at the specified speed. This thesis does not explore the counting algorithm, therefore there is only one person going through the frame at a time. The person will enter the bus by moving from the top of the FOV to the bottom, and then exit the bus by moving from the bottom to the top of the FOV. When the texel camera takes a shot, three images will be generated: one of depth, one of brightness, and one of color. The depth and brightness images are from the ToF camera and the color image is from the color camera.

These three types of images will be input into the people counting system. Further and detailed information about the people counting system is given by Sallay [10]. In this thesis, the database of 68 people who have valid data for feature selection is used.

3.2 Feature Generation

Feature generation is part of the people counting system, and features can be considered to be the byproduct of the people counting system. As a person walks through the FOV of the texel camera, his body regions in the images are divided into two parts: head and shoulder. The separation of a person into head and shoulder regions provides a natural division of features. Features are computed for both regions. A feature vector is created by stacking all of the features associated with a region into a vector. In the following discussion, features will be referred to as belonging to a region. This region could be either a head or 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 determined to be a head or shoulder region until the person completely exits the FOV. At that time, the association and designation of heads and shoulders are made, using data from both regions in image frames where the person appears. After feature generation, each region will be represented using a feature vector. As repeated measures will be made when people go through the FOV, for each person there will be more than one feature vector for the head region and more than one feature vector for the shoulder region.

For the feature generation process, there are two assumptions for the people counting system. The first is that the counting of people should always be acurate, with no double counting or miscounting. Therefore, when a person walks through the frame, all feature vectors generated are assumed to belong to the person. The second assumption is that the association and designation of head and shoulder should always be correct, where no head feature vector is assigned to be shoulder and vice versa.

A cold mirror is used for separating the incoming light into two bands, for the ToF camera and color camera, respectively. However, because of the characteristics of the cold mirror, some of the infrared wavelengths next to the visible wavelengths will be reflected. The ToF camera has an illumination set to emit near-infrared light, whose frequency is about 785 nm [24]. Some of this will be reflected into the color camera lens. The extra ToF illumination directed into the color camera lens will be regarded as red by the sensor. Consequently, the color images will contain more red than they normally do when only the color camera is used.

There are two ways of post-processing the images with excessive red. The first is to abandon the red component, and use only green and blue components and the corresponding hue and saturation. The second is to shift the red by some amount. In this case, the red is shifted using the average of the means of the green and blue components, because they are not affected by the excessive red coming into the sensor. After shifting, the color magnitude is ignored by using the chromaticity value [25]. This is because for the bus environment, the illumination when a person enters and exits the bus may be different by a great amount.

Thus, the intensity of the scene may change greatly; however, the hue and saturation will not change significantly. By setting the intensity to a constant value, the chromaticity will provide color information that does not change based on illumination, helping to clasifiy images accurately.

There are 86 features for the head and 82 features for the shoulder. For each region, there is one depth ($D1$) image, one red ($R1$) image, two green ($G1$ and $G2$) and blue ($B1$ and $B2$) images, and two hue ($H1$ and $H2$) and saturation ($S1$ and $S2$) images. The images and translations of them are shown in Figure 3.1.

The features to be generated are put into three categories: depth-related features, color-related features, and texture-related features. The depth related features are generated from the depth data, and the color related features, as well as texture related features are computed from the color images. The detailed features to be generated are listed by category in Table 3.1.

Besides mean value and standard deviation, the skewness and kurtosis are used, which are the third and fourth central moments. Skewness is a measure of the asymmetry of the data around the sample mean. Kurtosis is a measure of how outlier-prone a distribution is.

The texture-related features used contain energy, contrast, entropy and homogeneity [26]. These kinds of values are computed using the LxL co-occurrence matrix $\boldsymbol{G}$. The co-occurrence is computed by counting the number of times a pixel pair with value z_i and z_j occur in an image in a defined position pattern. An example of the position pattern is "one pixel immediately to the right," which is used in the example below and this thesis. Using $\boldsymbol{G}$, the relative positions of pixels in an image can be attained and by choosing the proper position pattern, the texture can be detected.

A example of the computation of the co-occurrence matrix is shown below, where

$$\boldsymbol{I} = \begin{bmatrix} 1 & 2 & 0 \\ 1 & 1 & 1 \\ 2 & 0 & 0 \end{bmatrix}, \boldsymbol{G} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 2 & 0 & 0 \end{bmatrix}. \tag{3.1}$$

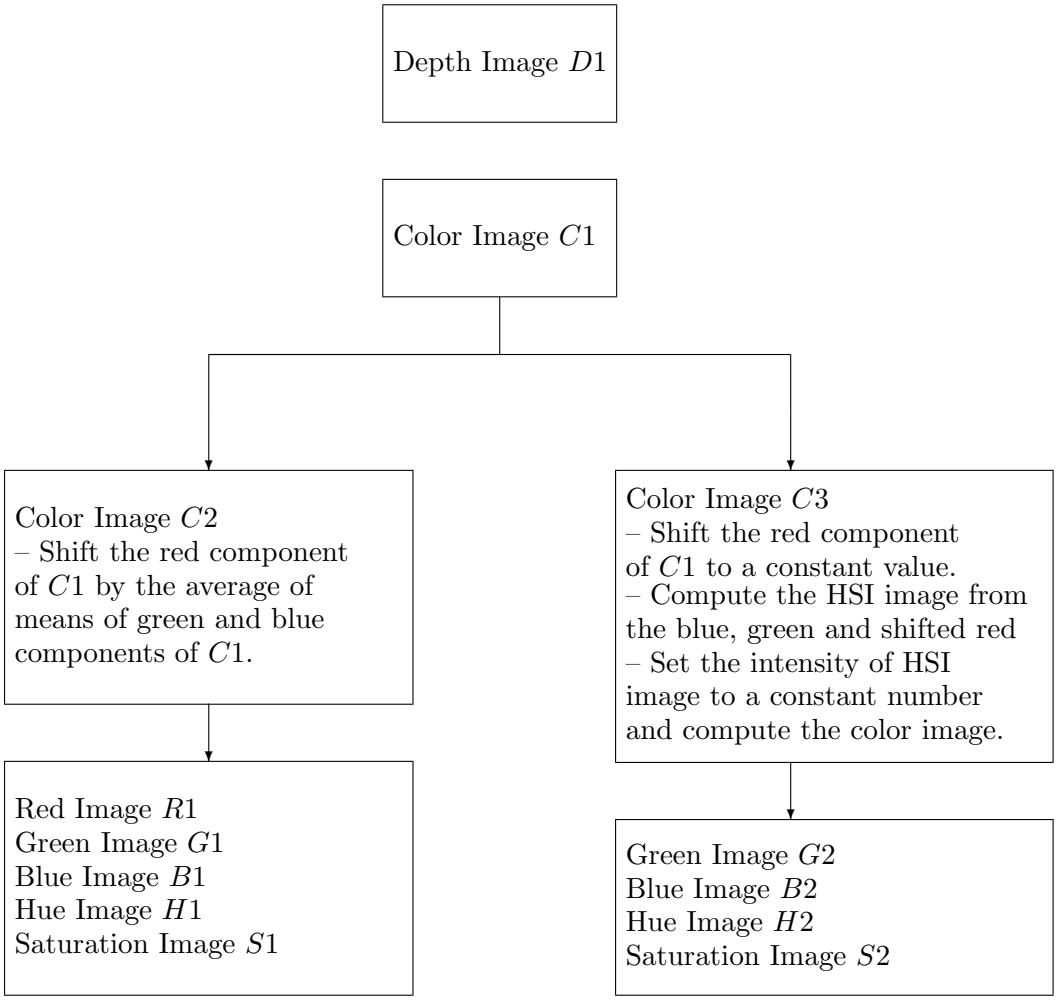


Fig. 3.1: Translation of images used for feature generation.

Table 3.1: List of features before feature selection.

Using head or shoulder data	
Image used	Feature extracted
Depth images ($D1$) (Height value is used for $D1$)	Height Mean Gaussian curvature mean Mean curvature mean Standard deviation Skewness, Kurtosis Energy, Contrast Entropy, homogeneity
Red images ($R1$) Green images ($G1$) Blue images ($B1$) (Pixel value is used for $R1$, $G1$ and $B1$) Hue images ($H1$) Saturation images ($S1$)	Mean Standard deviation Skewness, Kurtosis Energy, Contrast Entropy, homogeneity
Green images ($G2$) Blue images ($B2$) (Pixel chromaticity value is used for $G2$ and $B2$) Hue images ($H2$) Saturation images ($S2$)	Mean Standard deviation Skewness, Kurtosis Energy, Contrast Entropy, homogeneity
Using head and shoulder data	
Depth images ($D1$)	Head-to-shoulder difference Head-to-shoulder ratio Head-to-shoulder difference to head ratio Head-to-shoulder difference to shoulder ratio

$\boldsymbol{I}$ is the input matrix and $\boldsymbol{G}$ is the co-occurence matrix. An example of computation is that at positions $(0,0)$ and $(0,1)$ in I, the value is 1 and 2, thus at position $(1,2)$ in $\boldsymbol{G}$, the value will increase by 1. But since it is the only pair in $\boldsymbol{I}$ and in the designated position pattern that have value 1 and 2, the value at position $(1,2)$ in $\boldsymbol{G}$ will remain 1.

The formulas for the features are described in Table 3.2, where x is the pixel value, μ is the mean of a region, and σ is the standard deviation of a region. The value g_{ij} is the entry of $\boldsymbol{G}$ normalized by the sum of all entries in $\boldsymbol{G}$ and L is the size of $\boldsymbol{G}$. L is 128 in this thesis, which means that the value of each input image will be scaled to 128 gray-levels.

The mean, standard deviation, skewness, kurtosis, energy, contrast, entropy, and homogeneity are applied on all images. However, there are some features which only use the depth images. These are the Gaussian curvature mean, mean curvature mean and head-to-shoulder difference related features. The Gaussian curvature and mean curvature [27] of a depth image pixel are computed using coefficients of the first and second fundamental form of the surface. E_{f_1}, F_{f_1}, and G_{f_1} that are not boldfaced are coefficients of the first fundamental form, and L_{f_2}, M_{f_2}, and N_{f_2} that are not boldfaced are coefficients of the second fundamental form. Subscripts f_1 and f_2 stand for "first fundamental form" and "second fundamental form." The first fundamental form of a surface $\boldsymbol{z} = \boldsymbol{z}(u,v)$ at a point of the surface associates the tangent vectors at that point. It is computed using the inner product of any vector $\boldsymbol{t} = \alpha \boldsymbol{z}_u + \beta \boldsymbol{z}_v$, which is the linear combination of the tangent vectors $\boldsymbol{z}_u$ and $\boldsymbol{z}_v$. α and β are real numbers. $\boldsymbol{z}_u$ and $\boldsymbol{z}_v$ are linearly independent for any (u,v) in the domain of z and span the tangent plane. The traditional expression of the first fundamental form is computed using the inner product

$$\begin{aligned} < \boldsymbol{t}, \boldsymbol{t} > &= \alpha^2 < \boldsymbol{z}_u, \boldsymbol{z}_u > + 2\alpha\beta < \boldsymbol{z}_u, \boldsymbol{z}_v > + \beta^2 < \boldsymbol{z}_v, \boldsymbol{z}_v >, \\ &= E_{f_1}\alpha^2 + 2F_{f_1}\alpha\beta + G_{f_1}\beta^2. \end{aligned} \tag{3.2}$$

In (3.2), $\boldsymbol{z}_u$ and $\boldsymbol{z}_v$ are the partial derivatives of $\boldsymbol{z}$ with respect to u and v, $E_{f_1} =< \boldsymbol{z}_u, \boldsymbol{z}_u >$, $F_{f_1} =< \boldsymbol{z}_u, \boldsymbol{z}_v >$ and $G_{f_1} =< \boldsymbol{z}_v, \boldsymbol{z}_v >$.

An example of the computation of the first fundamental form is given as follows. For

Table 3.2: Formulas for some features.

Name	Formula
Skewness	$\frac{E(x-\mu)^3}{\sigma^3}$
Kurtosis	$\frac{E(x-\mu)^4}{\sigma^4}$
Energy	$\sum_{i=1}^{L}\sum_{j=1}^{L}(i-j)^2 g_{ij}$
Contrast	$\sum_{i=1}^{L}\sum_{j=1}^{L} g_{ij}^2$
Entropy	$\sum_{i=1}^{L}\sum_{j=1}^{L}\frac{g_{ij}}{1+\lvert i-j\rvert}$
Homogeneity	$-\sum_{i=1}^{L}\sum_{j=1}^{L} g_{ij} log_2 g_{ij}$
Gaussian curvature	$\frac{L_{f_2}N_{f_2}-M_{f_2}^2}{E_{f_1}G_{f_1}-F_{f_1}^2}$
Mean curvature	$\frac{L_{f_2}G_{f_1}-2M_{f_2}F_{f_1}+N_{f_2}E_{f_1}}{2(E_{f_1}G_{f_1}-F_{f_1}^2)}$
Head-to-shoulder difference	$h_1 - h_2$
Head-to-shoulder ratio	$\frac{h_1}{h_2}$
Head-to-shoulder difference to head ratio	$\frac{(h_1-h_2)}{h_1}$
Head-to-shoulder difference to shoulder ratio	$\frac{(h_1-h_2)}{h_2}$

a plane $\boldsymbol{z}(u,v) = u\boldsymbol{p} + v\boldsymbol{q}$, where $\boldsymbol{p}$ and $\boldsymbol{q}$ are perpendicular unit vectors. Then $\boldsymbol{z}_u = \boldsymbol{p}$ and $\boldsymbol{z}_v = \boldsymbol{q}$. So $E_{f_1} =< \boldsymbol{z}_u, \boldsymbol{z}_u >= 1$, $F_{f_1} =< \boldsymbol{z}_u, \boldsymbol{z}_v >= 0$ and $G_{f_1} =< \boldsymbol{z}_v, \boldsymbol{z}_v >= 1$. The first fundamental form is then $\alpha^2 + \beta^2$.

The second fundamental form is a quadratic form on the tangent plane of a surface $\boldsymbol{z} = \boldsymbol{z}(u,v)$ in the three-dimensional Euclidean space. The formula of the second fundamental form is similar to that of the first fundamental form,

$$L_{f_2}\alpha^2 + 2M_{f_2}\alpha\beta + N_{f_2}\beta^2. \tag{3.3}$$

In (3.3), L_{f_2}, M_{f_2}, and N_{f_2} are computed using the second partial derivatives $\boldsymbol{z}_{uu}$, $\boldsymbol{z}_{uv}$ and $\boldsymbol{z}_{vv}$ of $\boldsymbol{z}$ according to u and v, as well as the normal vector $\boldsymbol{b}$ which is

$$b = \frac{\boldsymbol{z}_u \times \boldsymbol{z}_v}{|\boldsymbol{z}_u \times \boldsymbol{z}_v|}, \tag{3.4}$$

where $\boldsymbol{z}_u \times \boldsymbol{z}_v$ is the cross product of $\boldsymbol{z}_u$ and $\boldsymbol{z}_v$. Then $L_{f_2} =< \boldsymbol{z}_{uu}, \boldsymbol{b} >$, $M_{f_2} =< \boldsymbol{z}_{uv}, \boldsymbol{b} >$ and $N_{f_2} =< \boldsymbol{z}_{vv}, \boldsymbol{b} >$.

The formulas for computing the Gaussian and mean curvatures are shown in Table 3.2, where E_{f_1}, F_{f_1}, and G_{f_1} are coefficients in (3.2), and L_{f_2}, N_{f_2}, and M_{f_2} are coefficients in (3.3). The edges of the surfaces are not used for curvature computation since the current people counting system does not adequately preserve the edges.

The head-to-shoulder difference related features are computed using the height of head h_1 and the height of shoulder h_2 for each frame.

3.3 Feature Selection Algorithm

The deterministic wrapper feature selection algorithms use less computation than other wrapper feature selection algorithms. Sequential backward selection (SBS) [28] and sequential forward selection (SFS) [29] are the two most commonly used wrapper feature selection algorithms. The SBS starts from all features in the set and progressively excludes the worst one, stopping when the removing of a feature causes the performance to drop. The SFS

starts empty and adds one best feature a time from the set, stopping when there is no further improvement in accuracy. The problem with these two algorithms is that they are easily trapped into local maxima. In SBS, once a feature is removed, it is removed permanently. In SFS, when a feature is added, it will never be removed.

The plus l - take away r algorithm which is also referred to as (l, r) algorithm, can partially overcome the problem of SBS and SFS, by combining the SBS and SFS within the algorithm. The SFS is applied l times followed by applying SBS r times, repeating the process until the demanded number of features is reached. The weak point with the (l, r) algorithm is that it is difficult to determine the proper number of l and r.

Pudil et al. [30] suggest sequential floating forward selection (SFFS) and sequential floating backward selection (SFBS) algorithms, so that the single-track search problem can be solved. For SFFS, the backward search will be applied as long as the corresponding subsets performs better than the previously evaluated and saved ones with the same number of features. This is the same for SBFS. Due to dynamic back-searching, there is no parameter required for the algorithm. For practical programming, the SFS is applied at the beginning of SFFS several times to give an initial selection of SFFS, after which the SFFS is executed based on the number of features already selected.

The SFFS was chosen for the feature selection task. The algorithm is described in Algorithm 3.1. The F function used in Algorithm 3.1 is the criterion function and will be described in Section 3.4.

3.4 Criterion Function

The SFFS algorithm can be described as a greedy search of the optimal subset that gives a more optimal performance than other subsets. To evaluate the performance of subsets with the same number of features, a criterion function is used. A criterion function assesses the performance of each subset, which is used by the SFFS for comparison. For this feature selection task, two criterion functions are explored and compared: the marginal accuracy for people classification and the maximization of the minimum normalized Mahalanobis distance of incorrect matching.

Algorithm 3.1 SFFS Algorithm

Input:

Total number of features, Z
Feature Set, $\boldsymbol{S} = \{s_t \mid t = 1, \cdots, Z\}$, where s_t is the tth feature
Number of Features Required, E

Output:

Optimal Feature Subset, $\boldsymbol{S}_E$

Initialization:

Empty subset, $\boldsymbol{S}_E = \varnothing$
Current Number of Features, $e = 0$

Begin

While $e \neq E$
/* Inclusion */
(1) $s_{plus} = \arg \max_{s_t \in \boldsymbol{S} - \boldsymbol{S}_e} F(\boldsymbol{S}_e + s_t)$

/* Add the most significant feature with respect to $\boldsymbol{S}_e$ */
$\boldsymbol{S}_{e+1} = \boldsymbol{S}_e + s_{plus}$
$e = e + 1$

/* Conditional Exclusion */
(2) $s_{minus} = \arg \max_{s_t \in \boldsymbol{S}_e} F(\boldsymbol{S}_e - s_t)$

/* Find the least significant feature within S_e */
If $F\ (\boldsymbol{S}_e - s_{minus}) > F(\boldsymbol{S}_{e-1})$ **then**
$\boldsymbol{S}_{e-1} = \boldsymbol{S}_e - s_{minus}$
$e = e - 1$
Goto (2)
Else
Goto (1)
End

End

In the following, "entering features" refer to features acquired from a person entering the bus, and "exiting features" refer to features acquired from a person exiting the bus. Whether the person is entering or exiting the bus is decided before acquisition of data.

3.4.1 Marginal Accuracy

The marginal accuracy is the figure of merit in simulations of this section. Each person is classified by comparing with all people in the pool. When a decision is made by finding the minimum distance value, the corresponding person in the pool is not removed. The accuracy is calculated by dividing the number of correct classifications r by the total number of people n in the pool used for classifications.

Although a person's captured images will be divided into shoulder and head regions, when it comes to the marginal accuracy calculation, the feature vectors of head and shoulder in an image frame are stacked together to form the feature vectors for the person. This is because the head-to-shoulder related features are used, which requires the association of the head and shoulder feature vectors within a same image frame. This is done by associating the image regions in each frame that go with the head and shoulder feature vectors. If there are feature vectors of the head or shoulder regions that do not have associated shoulder or head counterparts within a same image frame, they will be discarded. The reason is that in the database there are two persons whose head and shoulder do not match in a same frame because of the noise which affects the people counting system. In order to use the head-to-shoulder difference related features, the two persons have to be removed from the simulations, which makes $n = 66$ in the simulations.

The 66 people in the pool is fixed in the simulations and marginal accuracy is computed based on it. However, this is a small set of people and may not provide robust results. As a result, more people are used for the simulations. As mentioned in Chapter 2, the classifications are implemented by computing the Mahalanobis distance of two people using (2.1). Also from the acquisition of feature vectors of each person, there are a set of feature vectors from the entering frames and another from the exiting frames. The ordering of people is not considered for the classification, but the selection of a person's feature vectors

from either the entering or exiting frames as $\boldsymbol{f}$ in (2.1) is considered to enlarge n by a multiple of 66. Here $n = 66\text{x}3 = 198$. An example is used to show that setting different feature vectors as $\boldsymbol{f}$ will give different results. If there are two persons in the database $\{a_1, a_2\}$ and the feature vectors are $\{\boldsymbol{f}_{1,e}, \boldsymbol{f}_{1,l}, \boldsymbol{f}_{2,e}, \boldsymbol{f}_{2,l}\}$. $\{\boldsymbol{f}_{1,e}, \boldsymbol{f}_{1,l}\}$ are the entering and exiting feature vector sets of person 1 and $\{\boldsymbol{f}_{2,e}, \boldsymbol{f}_{2,l}\}$ are the entering and exiting feature vector sets of person 2. If $\{\boldsymbol{f}_{1,e}, \boldsymbol{f}_{2,e}\}$ are selected as $\hat{\boldsymbol{\mu}}_i$ and $\{\boldsymbol{f}_{1,l}, \boldsymbol{f}_{2,l}\}$ as $\mathbf{f}$ for one trial, while $\{\boldsymbol{f}_{1,l}, \boldsymbol{f}_{2,e}\}$ are selected as $\hat{\boldsymbol{\mu}}_i$ and $\{\boldsymbol{f}_{1,e}, \boldsymbol{f}_{2,l}\}$ as $\mathbf{f}$ for another trial, different probability of correct classifications may be produced. This is because it is possible that $\delta_{\boldsymbol{f}_{1,e}}(\boldsymbol{f}_{1,l}) > \delta_{\boldsymbol{f}_{2,e}}(\boldsymbol{f}_{1,l})$ which gives an incorrect classification, while $\delta_{\boldsymbol{f}_{1,l}}(\boldsymbol{f}_{1,e}) < \delta_{\boldsymbol{f}_{2,e}}(\boldsymbol{f}_{1,e})$ which gives a correct classification.

The enlarging of n is done in the same way as the example above. When the first 66 people finish classifications and create r_1 correct classifications, the selection of a person's feature vectors from either the entering or exiting frames as $\boldsymbol{f}$ in (2.1) will be changed among the 66 people randomly. The classification will go on with the "renewed" 66 people in the pool to create r_2 correct classifications. The process will continue to form a third group of 66 people, creating r_3 correct classification. The marginal accuracy will be computed using $\frac{r_1+r_2+r_3}{66\text{x}3}$ and will be used as F in Algorithm 3.1.

The criterion function F used for SFFS will balance the execution time and the performance. However, it is not robust enough because it does not cover enough situations for the selection of entering or exiting feature vector set of a person as $\hat{\boldsymbol{\mu}}_i$ and $\mathbf{f}$ in (2.1). Thus a further processing on the SFFS results is required, by inputting the feature vectors selected into the classifier F_0 which enlarges n to 100 times of 66 and computes the probability of correct classifications, which is used in Algorithm 3.2. The classifier F_0 has no relationship with F since F_0 is used for postprocessing the results from the SFFS using F. The feature subsets selected from SFFS will be used to be input of F_0 and the accuracy will be computed. The accuracy of $\boldsymbol{S}_e$ from F_0 is represented using V_e and will be used for the decision of which feature subset is to be selected. The feature subset with the largest probability of correct classifications from F_0 will be selected.

Algorithm 3.2 Feature Selection

Input:

Feature Set, $\boldsymbol{S} = \{s_t \mid t = 1, \cdots, Z\}$

Output:

Best Feature Subset, $\boldsymbol{S}_{best}$

Initialization:

Set of temporary feature subsets from SFFS, $\boldsymbol{S}_{tmp} = \{\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_Z\}$

Empty Marginal Accuracy, $\boldsymbol{V} = \{V_t \mid t = 1, \cdots, Z\}$

Begin

For $e = 1$ to Z

Run the SFFS in Algorithm 3.1 and get the best feature subset $\boldsymbol{S}_e$ of e features with criterion function F

End

For $e = 1$ to Z

/* Use a robust classifier F_0 with more runs */

$V_e = F_0(\boldsymbol{S}_e)$

End

$\boldsymbol{S}_{best} = \arg \max_{\boldsymbol{S}_e \in \boldsymbol{S}_{tmp}} V_e$

End

3.4.2 Maximization of the Minimum Distance

Using the marginal classification accuracy to be the criterion function is a reasonable method of assessing the performance of each feature subset. However, the method is not effective when the trellis optimization algorithm is used (described in Chapter 2). The reason is that, to increase the classification accuracy in the bus environment, which is a sequential estimation problem, the trellis is implemented in a maximum-likelihood sense so that when classification for a person is finished, the classification decision is not made immediately but is postponed until a robust case for decision making is met. This is carried out by accumulating the distance along the paths in the trellis and selecting the path with the smallest accumulated distance. The marginal accuracy does not take the specific distance value into account, but only the classification accuracy represented by the comparison of distances. Because of the averaging effect, the computation of the marginal accuracy will ignore the existence of outliers and produce a higher classification accuracy, which will make improper selections of the feature subset. Outliers are incorrect classification of people and

are caused by improper feature usage. When an improper feature set is used, the distance value of the correct classification and the minimum distance value of incorrect classifications may be close to each other. Since the difference between distance value of the correct and incorrect classifications is small, either may be kept in the path with minimum accumulated cost in the trellis. If the distance value of incorrect classifications are input into the trellis, it will be more difficult for the trellis to keep a path with minimum accumulated cost that contains correct classifications.

Because the specific distance value will be used for the trellis, a new criterion function for feature selection should be developed. The new method used here takes the distance value into account and returns the minimum distance value of the incorrect classification pair among all people in the database. Then by finding the feature subset that maximizes the returned minimum distance value of different feature subsets with the same number of features, the subset with the largest minimum distance value is selected. The idea of this criterion function is that, since the distance value will affect the accumulated distance of a path in the trellis, the distance of an incorrect classification pair should be as large as possible and the distance of a correct classification pair should be as small as possible. As a result, when an incorrect classification happens, the later correct classification will remedy the situation with fewer steps, keeping the path with minimum accumulated cost as the one with correct classification pairs.

Based on the description above, the criterion function has the formula

$$F(\boldsymbol{S}_e) = \min_{i,j \in n, i \neq j} (\delta_{\hat{\boldsymbol{\mu}}_1}(\boldsymbol{f}_j), \delta_{\hat{\boldsymbol{\mu}}_2}(\boldsymbol{f}_j), \cdots, \delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}_j), \cdots), \tag{3.5}$$

where e is the number of features, i and j are indices of people, $\boldsymbol{S}_e$ is the input feature subset, n is the number of people in the database, $\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}'_j)$ is the Mahalanobis distance between the entering and exiting feature vector set of different persons, $\boldsymbol{f}_j$ is the entering (or exiting) feature vector set of the jth person, $\hat{\boldsymbol{\mu}}_i$ is the mean exiting (or entering) feature vector set of the ith person. Note that the function is applied to all people in the database.

When it comes to implementation, the criterion function is modified by normalizing the value of $\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}_j)$ by the value of $\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}_i)$,

$$F(\boldsymbol{S}_e) = \min_{i,j \in n, i \neq j} \left(\frac{\delta_{\hat{\boldsymbol{\mu}}_1}(\boldsymbol{f}_j)}{\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}_i)}, \frac{\delta_{\hat{\boldsymbol{\mu}}_2}(\boldsymbol{f}_j)}{\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}_i)}, \cdots, \frac{\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}_j)}{\delta_{\hat{\boldsymbol{\mu}}_i}(\boldsymbol{f}_i)}, \cdots\right). \tag{3.6}$$

This is because when using different feature subset $\boldsymbol{S}_e$, the scale of the $F(\boldsymbol{S}_e)$ is different such that they are not comparable. By normalization, the value returned from the criterion function can be compared and used by the SFFS algorithm. For a certain number of features, the SFFS algorithm will find the subset that maximizes the $F(\boldsymbol{S}_e)$ for e features. The optimal number of features is selected by drawing the curve of $F(\boldsymbol{S}_e)$ as a function of e. If the peak of the curve is smaller than 1, then the number of features and corresponding feature subset with the largest normalized distance value is selected. If the peak of the curve is larger than 1, then the smallest number of features and corresponding feature subset among those whose normalized distance value is larger than 1 is selected.

3.5 Feature Selection Result

The performance of the feature selection algorithm using marginal accuracy and maximization of minimum distance will be discussed in this section.

3.5.1 SFFS with Marginal Accuracy as Criterion Function

The SFFS with the marginal accuracy as the criterion function is analyzed first. The feature selection results are shown in Figure 3.2. From the figure it can be seen that using the criterion function of marginal accuracy as F, which is mentioned in Algorithm 3.1, the marginal accuracy increases according to the increase of the number of features selected at first, and then stays almost unchanged as the number of features selected increases further. When more than about 120 features are selected, the marginal accuracy starts to decrease. After training the features, the selected features are tested using F_0. The feature set of 136 features is selected to be the best among others.

After the selection, the features will be used for classification, which is mentioned in

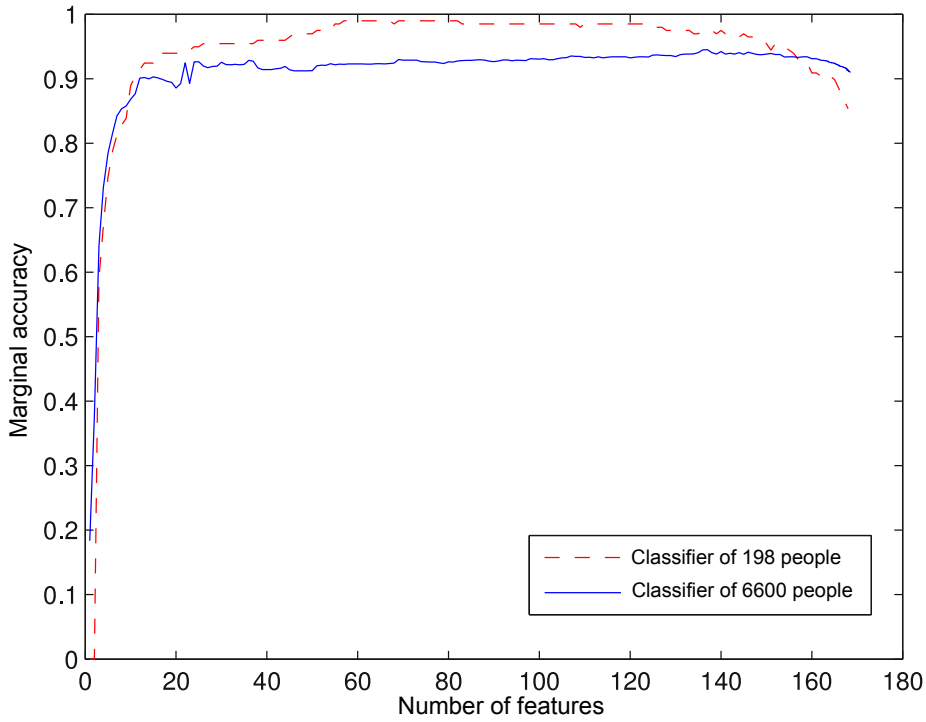


Fig. 3.2: Feature selection results using SFFS and marginal accuracy on a database of 66 people.

Chapter 2 and will be further described in Chapter 4. Here the classification results using the 136 features are shown in Figure 3.3, where the beam width of the trellis is 1000. It can be seen that the accuracy is good when a large number of people get off the bus at a given stop. But when there are fewer people getting off the bus, there is a huge classification accuracy drop. This is because the marginal accuracy does not take into account the Mahalanobis distance value, but the average. Consequently, there will be some outliers that are very small in Mahalanobis distance, which are difficult to identify using marginal accuracy, but can greatly affect the trellis optimization performance. The feature selection algorithm using the marginal accuracy as the criterion function is not good when used without considering the trellis optimization.

It also needs to be mentioned that in the simulations, the head-to-shoulder difference related features are not selected until a large number of features are considered, when the marginal accuracy drops significantly. Thus, those features will not be used for the feature selection simulations using the maximization of the minimum normalized distance

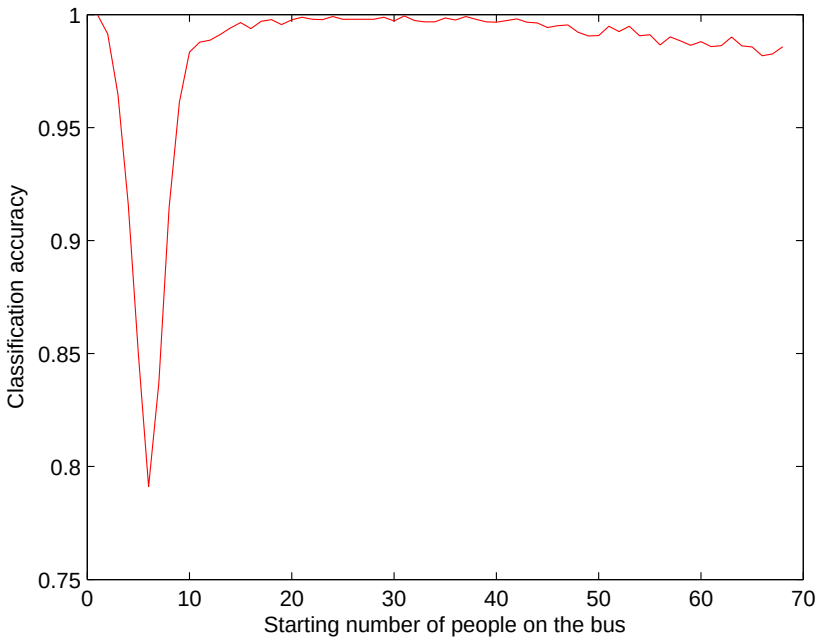


Fig. 3.3: Trellis performance using 136 features from SFFS and marginal accuracy.

as the criterion function. The reason for not using those features is that if the averaging effect of the marginal accuracy computation can not produce a higher accuracy, there are chances that outliers still exist and more people will be mixed with each other than when the average accuracy is high. Consequently, the maximization of minimum distance, which emphasizes the specific distance value, will definately have a lower accuracy. Without using those features computed using data from both head and shoulder, computation of the Mahalanobis distance can be implemented by assuming the independence of head and shoulder and ignoring the covariance between them.

3.5.2 Maximization of Minimum Distance as Criterion Function

The new criterion function for the SFFS trying to maximize the minimum normalized Mahalanobis distance of incorrect matching is an improvement over the marginal accuracy and is suitable to be used together with the trellis optimization algorithm. The feature selection results are shown in Figure 3.4. In the figure, the minimum normalized Mahalanobis distance of the incorrect matching first increases along with the increase of the number of

features selected, then reaches the peak, after which it decreases with the growth of the number of selected features. This illustrates the phenomenon of curse of dimensionality.

It can also be observed that there is a range of feature subsets that can cause perfect classification without the trellis optimization, which is from 52 features to 75 features. A minimum normalized Mahalanobis distance of incorrect matching which is above 1 guarantees that the correct matching pair will have a smaller distance than all other incorrect matching. It is clear that the criterion function using the maximization of minimum distance over using the marginal accuracy is superior, since the latter will not find the feature subset that is always perfect among the people used. Also, in Chapter 4, simulations are done to show that there is no accuracy drop for the first few people disembarking the bus.

Since the new criterion function is proven to be effective through simulations, its robustness should be considered to indicate if the selected features can also be effective on other data sets. The robustness is tested by implementing the feature selection process, using subsets of all people in the data set, and testing the performance using their complements. There are 50 people and 34 people who are used as the training set respectively, while the complementary 18 people and 34 people are used as the testing set, running the feature selection algorithm 55 and 26 times.

Figure 3.5 shows the number of features that are the same among different trials against the number of features selected. For example, if training features using 50 people, when 120 features are selected, there will be about 20 features that are always the same among different trials. Different from what might be expected, there are very few features that are always selected compared to the number of features selected in all runs for both tests. This means that as different people are selected as the training set, the features in different selected feature subsets are mostly different. The figure also shows that if more people are selected for the training set, the features selected will be more robust. In Figures 3.6 and 3.7, the individual runs can be seen and there are significant differences between the performance of the training set and that of the testing set. Figure 3.8 shows the average of the difference between the minimum normalized Mahalanobis distance of incorrect matching

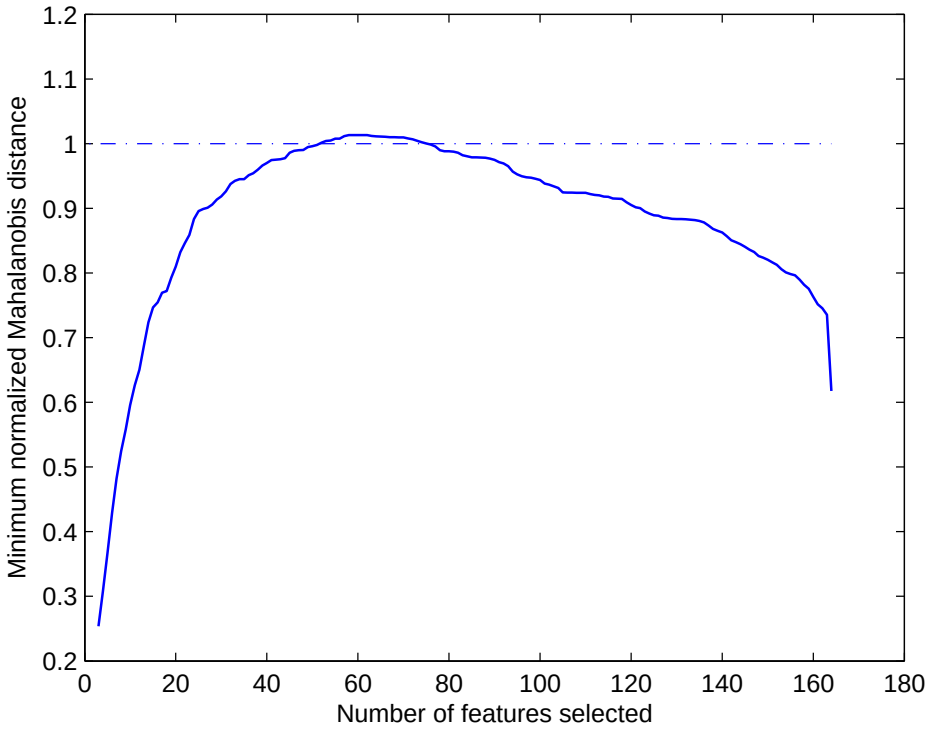


Fig. 3.4: Feature selection results using maximization of minimum distance as criterion function.

between the training and testing set, which also indicate the large difference in performance.

Although these tests give negative indications about the robustness of the feature selection algorithm using the new criterion function, further testing is required to confirm this. This is because the data set used for testing the robustness is small, with only 68 people. There is high probability that the training set does not cover a large variety of people such that the features that dominate in the testing set are considered while training the optimized features. For example, in the training set, people may wear shirts that are simple in color while in the testing set people's shirts may have a variety of colors or textures. In that case, the features selected in the training set will not be adequate when used on the testing set. More people-related data should be gathered and categorized in the future so that feature selection can be accomplished on a variety of different people. Also indicated in Figure 3.5, the more people that are used as the training set the better the selected features will be. This is another reason to use more people for feature selection. Note that Figure 3.8 indicates that as the training set gets larger, the average difference gets smaller.

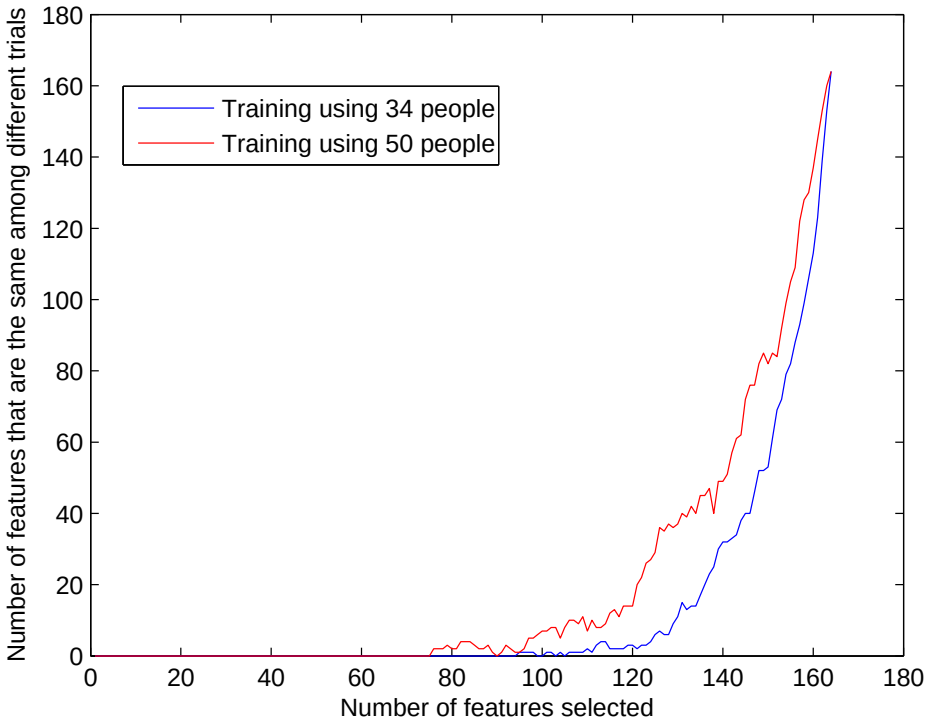


Fig. 3.5: Number of features that are robust among all trials.

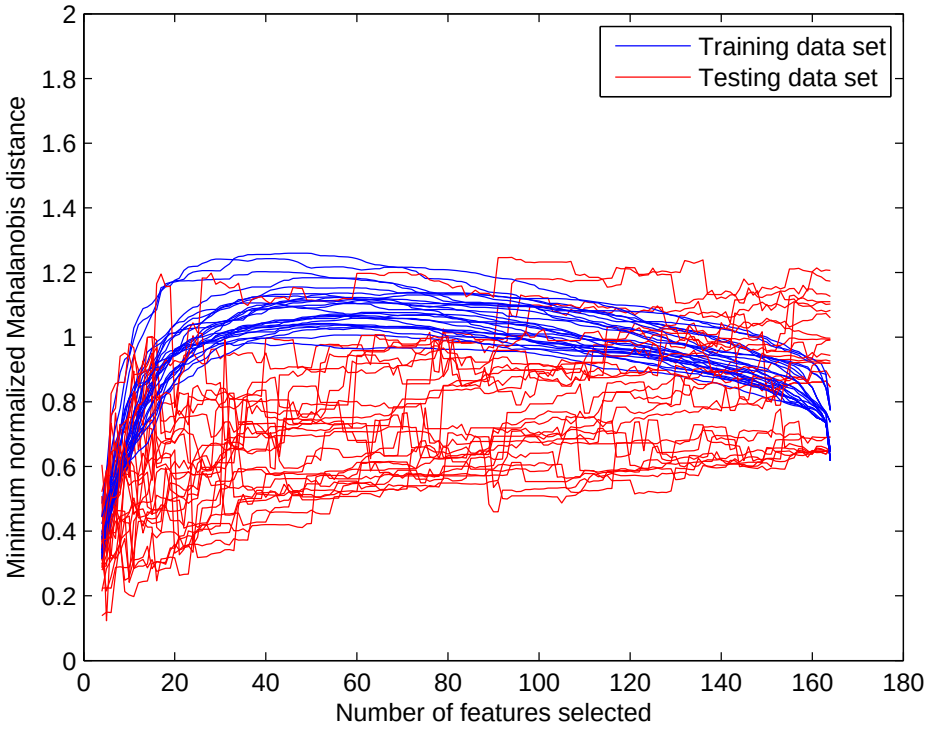


Fig. 3.6: Comparison of minimum normalized Mahalanobis distance (50 people as training set).

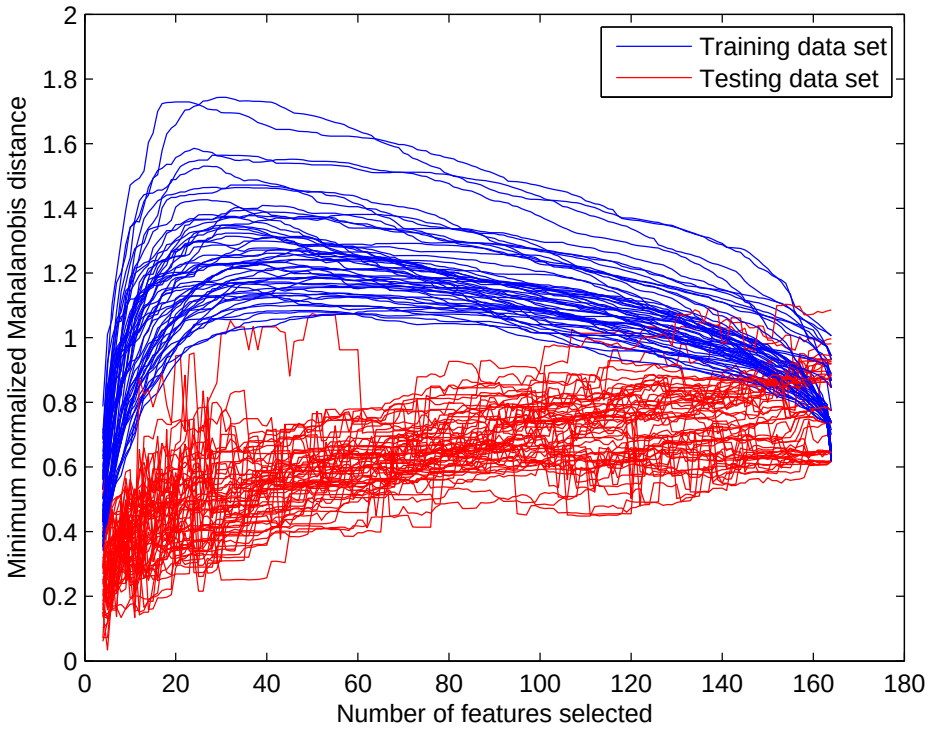


Fig. 3.7: Comparison of minimum normalized Mahalanobis distance (34 people as training set).

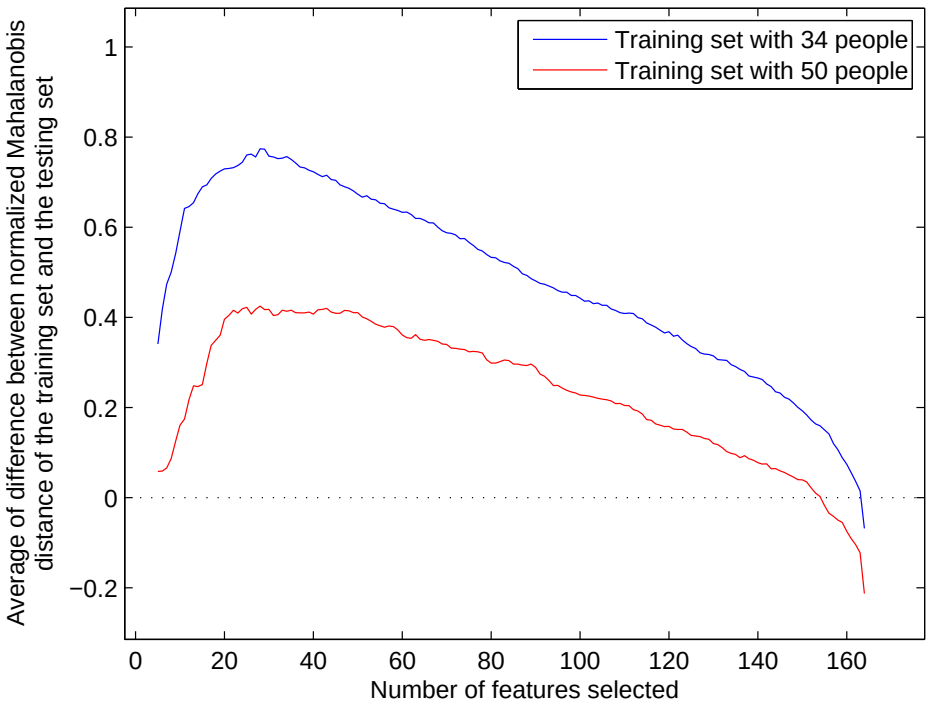


Fig. 3.8: Average of difference between normalized Mahalanobis distance of the training and testing data sets.

Chapter 4

Simulations and Performance

Simulations were done after feature selection and design of the trellis optimization algorithm. Section 4.1 reports experiments testing the trellis using simulations of people exiting the bus without entering. Section 4.2 gives the relationship between the trellis performance and the beam width as well as the normalized distance. Section 4.3 reports on the trellis behavior and shows the decision making process of the trellis. Section 4.4 shows results from a full simulation of the trellis by testing it in a real bus environment with multiple routes and retrieving statistics about individual people as well as about the route.

Discussions about the performance will be described within each section.

4.1 Simulation of Trellis Optimization Algorithm

4.1.1 Simulation Description

The performance of the trellis optimization algorithm is computed by running the classification simulations s times with n people being used for each single simulation. After all the s simulations there will be r correct matching of the people, which will generate a classification accuracy of $\frac{r}{s \cdot n}$.

There are data of 68 people generated from the counting system, from which their feature vectors are generated. For every person, a set of entering feature vectors and a set of exiting feature vectors will be computed. The feature sets will form two groups, one of which will be used for computing the mean feature vector for each person and the covariance matrix of all people since (LDA is used), while the other contains unknown people for classification.The features used for the simulations are shown in Table 4.1, and those

Table 4.1: 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

features are not from the feature selection algorithm described in Chapter 3. These features are used for initial performance evaluation, and is compared later with the performance using features that are from the feature selection algorithm in Chapter 3. Note that the head and shoulder of each person have the same features, although it is not necessary.

Because of the characteristics of the sequential estimation, which are different from the computation of marginal accuracy, the ordering of people is taken into account. There are three variables for the simulations: the number of people in the database, the number of people on the bus at the start, and the entering and exiting feature vectors for each person. The different ordering of the n people from the 68 people in the database will have $\frac{68!}{(68-n)!}$ possible permutations, and every person has two choices of using his feature vector sets, which will generate a total permutation of $\frac{68!}{(68-n)!}2^n$ different ways of using the people in the database. This is prohibitive for computing since it is impossible to use all of the combinations for the simulations. As a result, the Monte Carlo algorithm is used for simplification of the simulations to lower the computational cost.

The way of implementing the Monte Carlo algorithm is that a simulation is created randomly, by generating the ordering of people and the assigning of entering and exiting feature vector sets. In such a way, the total simulation of the algorithm is completely random, and will give a good estimate of the actual performance when using a large number

of simulations.

After randomly assigning the ordering and the selection of feature sets of people, a simulation will start with all people on the bus creating the initial pool. When the mean feature vectors of all people and the covariance matrix are computed, people begin to be removed from the bus one at a time. Whenever a person is removed, the Mahalanobis distance is computed with people in the pool and sorted.

There are two algorithms being compared. The simple sequential estimation algorithm does not put the Mahalanobis distance value into the trellis, and decisions are immediately made, while the trellis optimization algorithm will store the value and put it into the trellis to make decisions later. Each simulation will end when all people are removed from the bus.

When a simulation ends, the classification results, which is the order of exits estimated using the simple sequential estimation algorithm or trellis, will be compared with the true ordering used for the simulation. A classification error is declared if the entering feature vector set of a person is not assigned the same label as that of the exiting feature vector set of the same person after classification and association.

4.1.2 Marginal Accuracy and Simple Sequential Estimation Algorithm

The marginal accuracy is used for showing the classification capability of the classifier with different pool sizes. The marginal accuracy is computed in such a way that when a decision is made, the person who finishes classification is never removed from the pool. The starting pool size increases from 1 to 68 for each trial. The marginal accuracy of the LDA classifier against different pool sizes is shown in Figure 4.1.

The simple sequential estimation algorithm is that whenever a person is classified as exiting, the corresponding person in the pool will be removed. Decisions are made sequentially until all people exit the bus. This is simple and decisions are made immediately. However, if an incorrect decision is made, at least one error will certainly occur in the future, since the incorrect decision will never be corrected. Even worse, the error may propagate and degrade the performance even further. The performance of the simple sequential estimation

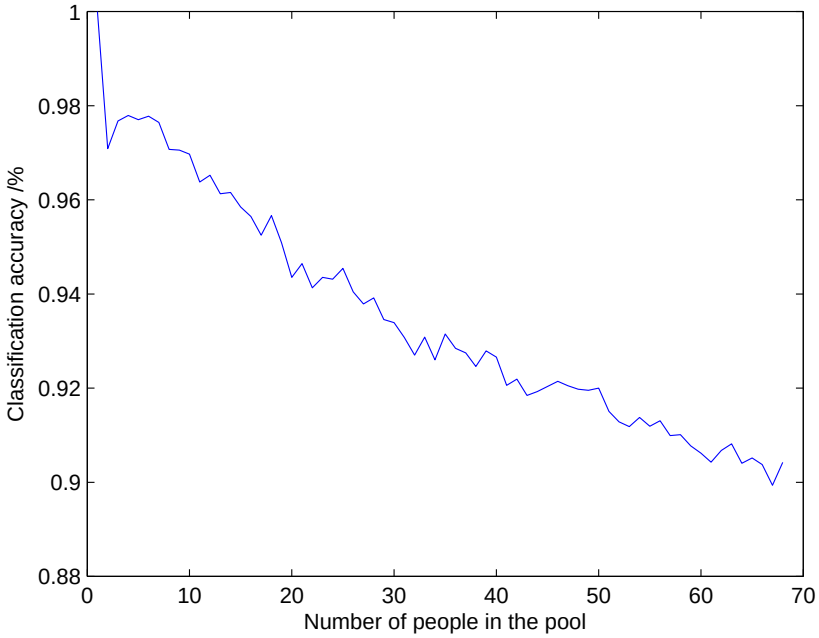


Fig. 4.1: Marginal accuracy of LDA for different pool sizes.

algorithm is shown in Figure 4.2. There are 10,000 trials of simulations implemented on the database of 68 people, thus a total of 680,000 people are tested for the starting number of 68.

4.1.3 Trellis Optimization Algorithm with Beam Search

A single simulation of the trellis optimization algorithm starts with a certain number of people on the bus in the initial pool. After random selection of exiting order as well as entering and exiting sets, one person will be removed from the bus and compared with all people in the pool, one at a time. The removed person will generate a set of Mahalanobis distances computed with the people in the pool, using which the trellis is extended by accumulating the path costs. The beam width is used to constrain the size of the step and whenever the beam width is reached, the trellis will be pruned and the four decision criterion will be checked on the trellis. If a decision is made, the corresponding person will be removed from the pool, and the decision recorded. The performance of the trellis optimization algorithm is shown in Figure 4.3 for different beam widths. There are 10,000

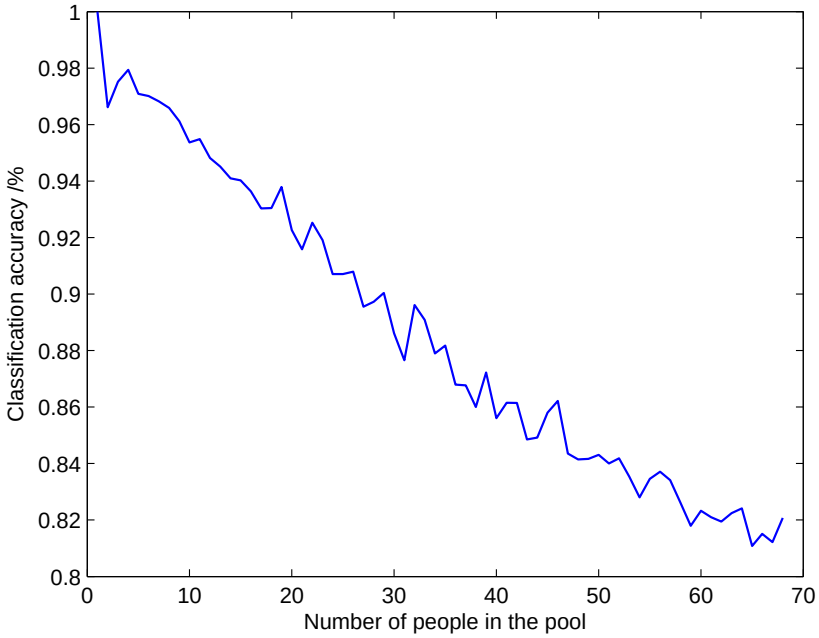


Fig. 4.2: Performance of the simple sequential estimation algorithm for different pool sizes.

trials of simulations implemented on the database of 68 people, thus in total 680,000 people are tested for the starting number of 68.

As can be seen from the figure, the performance of the classification will improve with larger beam width. This is because more paths will be kept with a larger beam width, which gives more chance of making correct decisions. Note that a beam width of 1 is the same as the simple sequential estimation algorithm.

4.1.4 Performance Using Optimized Features

Using the feature selection algorithm described in Chapter 3, optimized features can be selected which are suitable for the trellis optimization. In order to make comparisons with the performance of the features that are not selected using the feature selection algorithm, the feature selection results for 24 features are used in this subsection. The features are listed in Table 4.2, where there are 10 features for the head and 14 features for the shoulder. The features for the head and shoulder regions are not necessarily the same. The symbols $D1$, $R1$, $G1$, $G2$, $B1$, $B2$, $H1$, $H2$, $S1$, and $S2$ are from Figure 3.1.

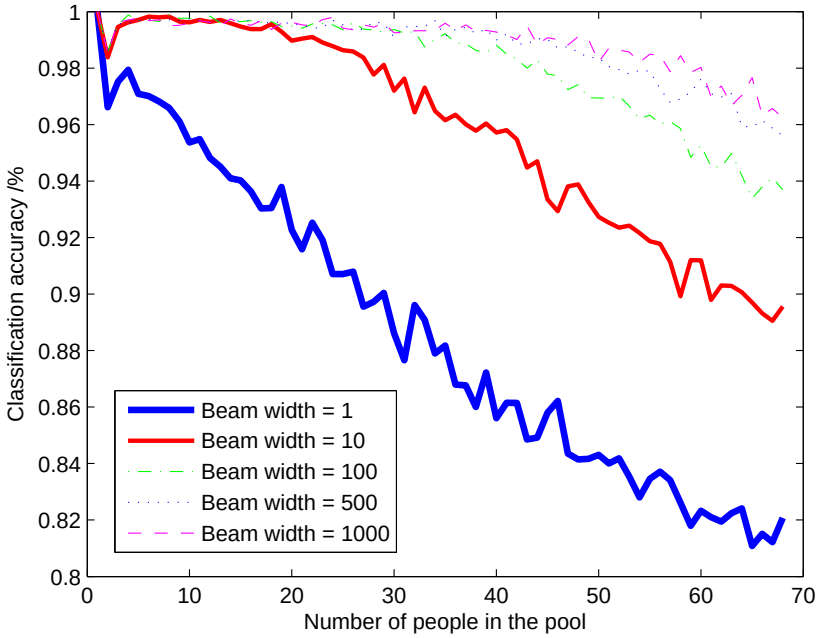


Fig. 4.3: Performance of trellis optimization algorithm with different beam width for different pool sizes.

The performance of the new features is shown in Figure 4.4, which is much improved from that of the previous features. There is no accuracy drop for the first few people, and the accuracy is much higher when trellis optimization is implemented. Thus, the feature selection algorithm is proven to be valid and effective.

4.2 Trellis Performance Using Different Beam Width and Normalized Mahalanobis Distance

The simulations in Section 4.1 show that the trellis optimization algorithm improves the classification performance, and the introduction of beam width helps to balance the performance and complexity. When thinking of the configuration of a system, one should consider how to choose a proper beam width in combination with the feature selection results. The selection of the features affects the distance between the correct and incorrect matching, and the beam width controls the size of the trellis. Both of these will have an effect on the classification accuracy. Figure 4.5 shows how the classification accuracy is

Table 4.2: List of optimized features used for classification.

Head	**Shoulder**
Height from $D1$,	Contrast from $D1$,
Standard deviation from $D1$,	Entropy from $D1$
Kurtosis from $G1$,	Kurtosis from $R1$
Entropy from $G2$	Green from $G1$
Skewness from $B1$	Skewness from $G1$
Kurtosis from $B1$	Energy from $G1$
Contrast from $B2$	Entropy from $G2$
Entropy from $H1$	Energy from $B1$
Entropy from $S1$	Homogeneity from $B2$
Homogeneity from $S1$	Contrast from $H1$
	Standard deviation from $H2$
	Contrast from $S1$
	Standard deviation from $S2$
	Homogeneity from $S2$

related to the beam width and the maximum of the minimum normalized Mahalanobis distance for incorrect matches.

The minimum normalized Mahalanobis distances are from the data shown in Figure 3.4, which is generated using all 68 people in the data set. Each normalized Mahalanobis distance corresponds to a set of features, which are used to generate the feature vectors for 68 people. The feature vectors will be used for classification using LDA and the Mahalanobis distances computed using (2.1) will be input into trellis for optimization. The simulation with each distance and beam width is done using all 68 people as the starting number of people on the bus and using 1000 runs, which has in total 68,000 people. The classification accuracy of the 1000 runs is averaged, and the error bars in the figure show the standard deviation of classification accuracy of each distance and beam width pair.

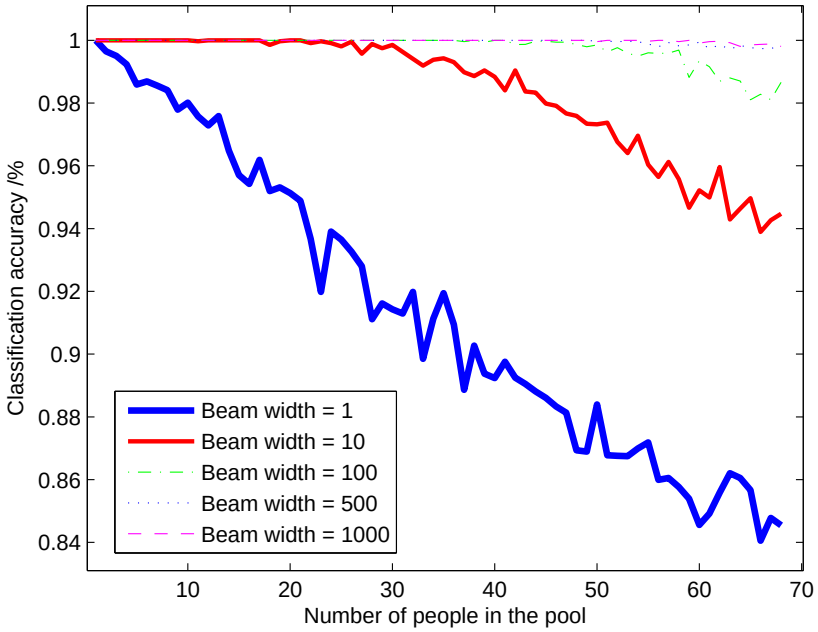


Fig. 4.4: Performance of trellis optimization algorithm with different beam width using optimized features for different pool sizes.

From Figure 4.5, the increase of the minimum normalized Mahalanobis distance will help increase the performance of the classification, and the increase of the beam width will do the same. It can also be observed that using the trellis, the minimum normalized Mahalanobis distance of incorrect matching does not have to be above 1 to have perfect classification accuracy. This is important because it shows the necessity of using trellis optimization, since even with a small beam width such as 10, the performance will have a large increase compared to not using trellis when beam width is 1.

4.3 Trellis Behavior

Described in Chapter 2, there are four criteria for making a decision: deleting an added step, deleting an inner step, tracing the inner path, and tracing the final path. They will be executed when the trellis reaches different conditions and decisions will be made. The simulations are the same as described in Section 4.1.1, which start with 68 people on the bus with one removed at a time.

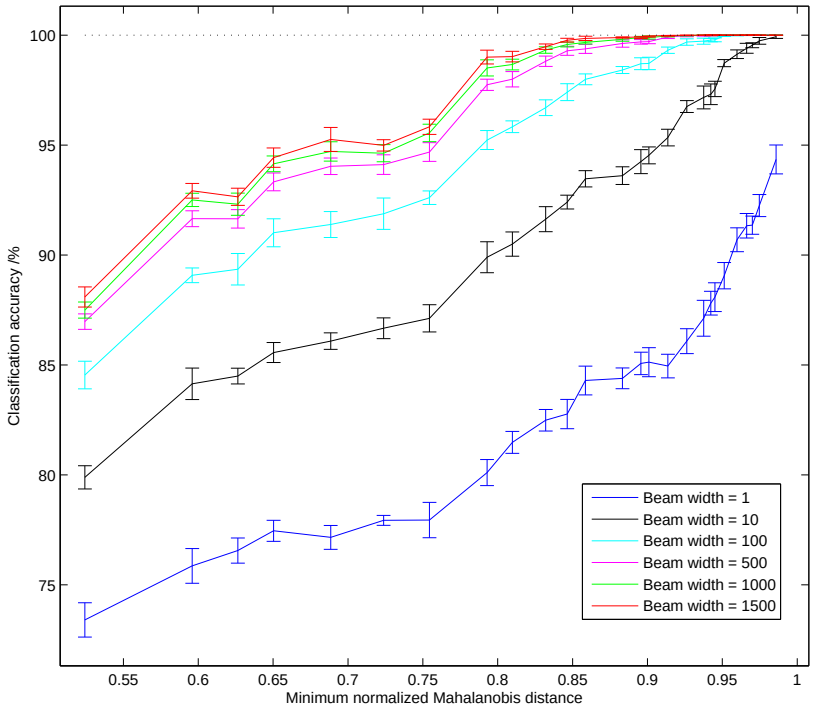


Fig. 4.5: Relation between trellis classification accuracy, beam width, and normalized Mahalanobis distance.

Figure 4.6 shows the distribution of the decision method against the number of people that have exited the bus when a person is exiting, which is the number of people who have finished computing the distance value and had them input into the trellis. For example, 9 means that 9 people have exited the bus and the 10th person is exiting the bus. From the figure, it can be seen that with the increase of the beam width, the number of decisions made by deleting an added step decreases, while the number of decisions made by tracing the final path increases. This is because the larger beam width makes it difficult for a person to dominate all the edges connecting to the new step, where there will be some incorrect matching because of noise. In this way, the steps remain undecided until the last step is added, which will cause the trellis to collapse all steps.

Figure 4.7 shows the distribution of the error source. First, it can be seen that with the increase of the beam width, the number of errors in the simulations decreases, where the simulations use 500 trials with the starting number of 68 people in the pool. The errors from the decisions made by tracing the final path comprise the majority of errors. This is

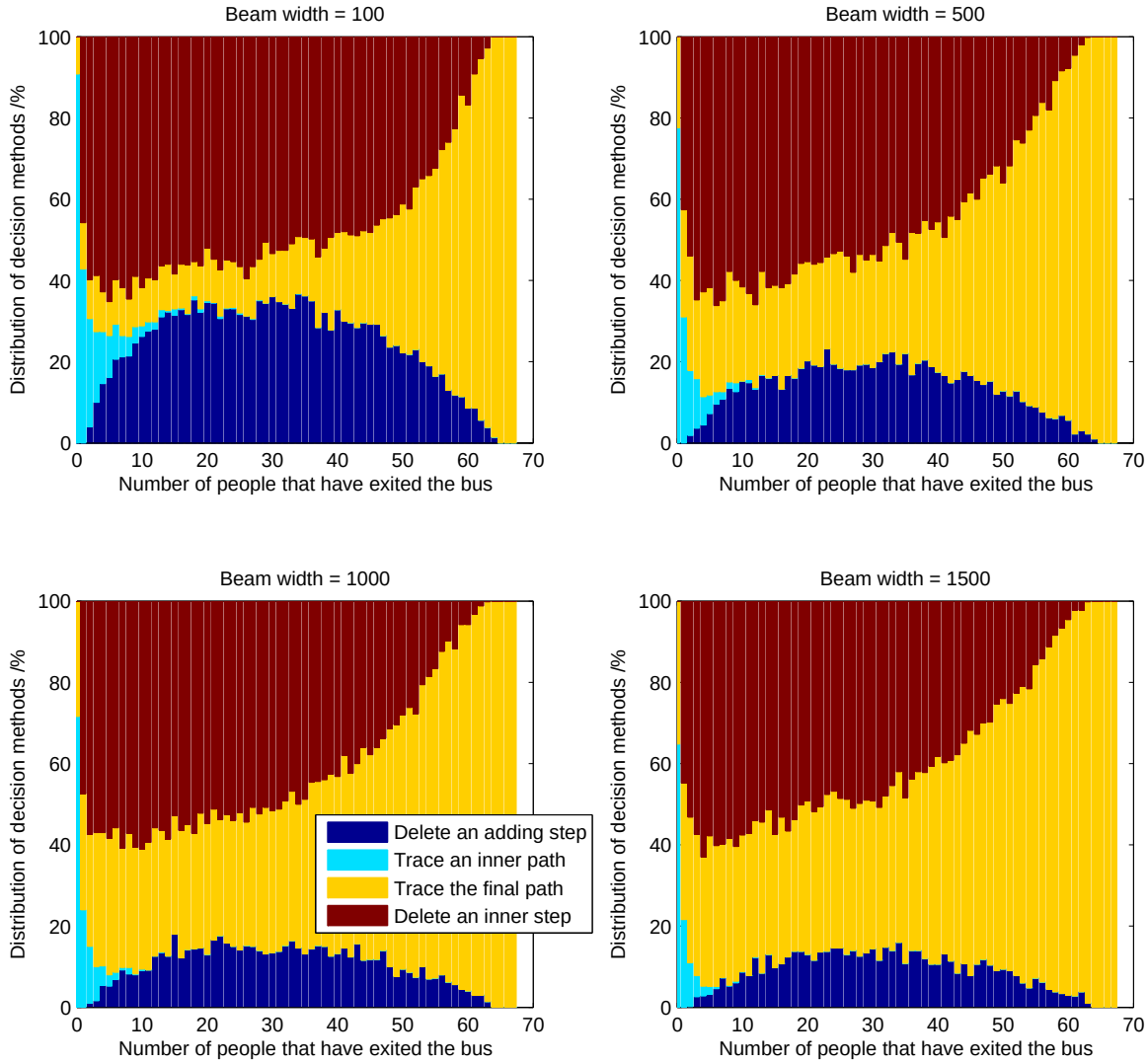


Fig. 4.6: Distribution of different decision making method.

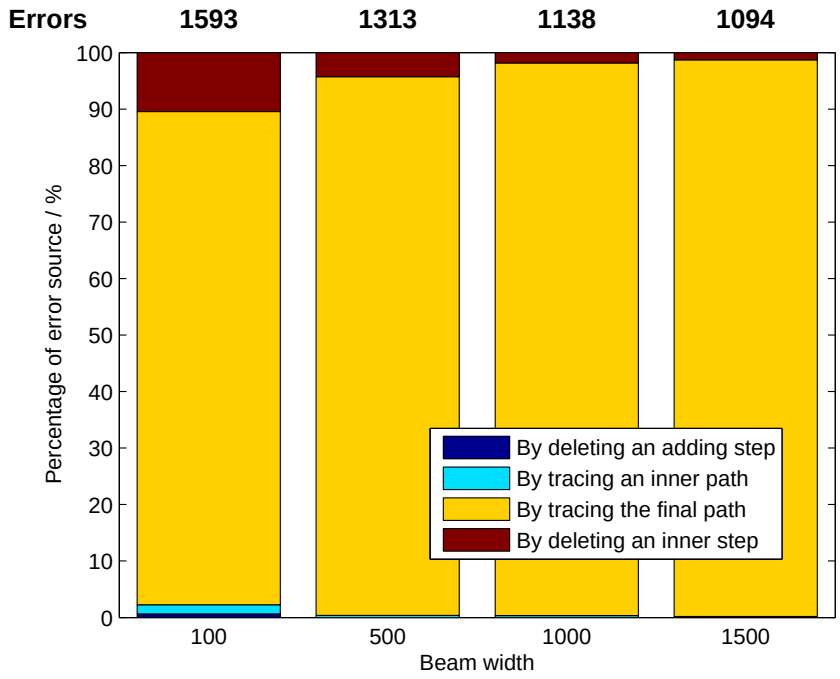


Fig. 4.7: Distribution of error source.

obvious since the steps that are difficult to decide immediately because of noise remain in the trellis for future decisions. If they stay in the trellis until the last step is added, which causes the trellis to collapse, it means that there is still uncertainty and decisions are still difficult to make even with all the other steps combined. Thus if a decision is made by tracing the final path, errors are likely to occur.

Figure 4.8 shows the ability of the trellis to make correct decisions against the number of people that have exited the bus. Based on 1,000 simulation trials with a starting number of 68 people in the pool, the percentage is computed by dividing the number of correct decisions for people who exit at the same time by 1,000. For example, among the 1,000 people who are the 10th to exit the bus and has 9 people already exited the bus, there are 982 among the 1,000 who are correctly classified, then the probability of correct decisions for people who are the 10th to exit is 98.2%. As can be seen from the figure, the later a person exits the bus, the lower the percentage of correct decision. This is because the trellis is ending when the distance value of the last few people are input into the trellis. The trellis will not maintain a path that is long enough to contain the states of correct classifications.

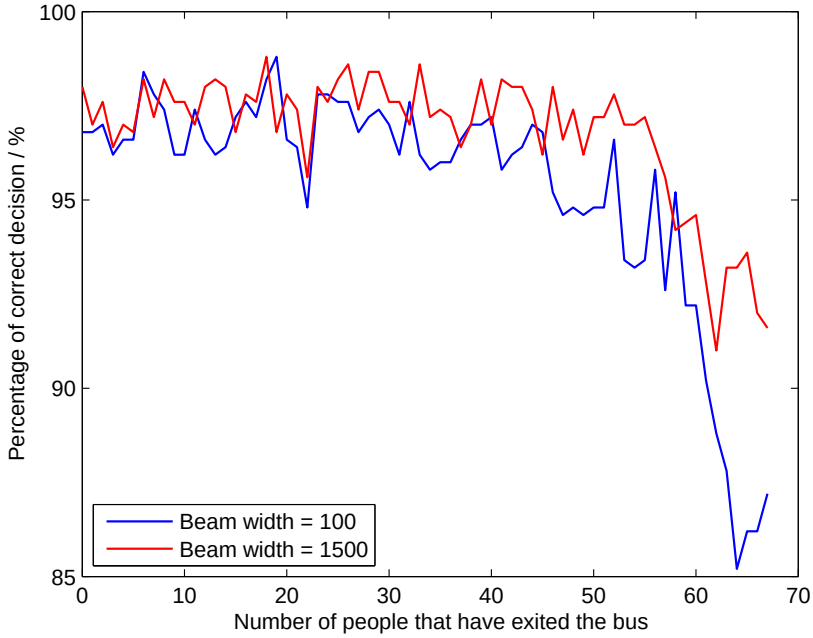


Fig. 4.8: Ability of making correct decisions according to the number of people that have exited the bus.

4.4 Simulation on the Real Route

After classification, a person will be associated with a specific entering stop and a specific exit stop. With this information, more statistics can be provided than from the people counting systems only. Although it is a common figure of merit, the 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 people ride the bus, or one may be interested in how long people remain on the bus when entering at a particular stop. These statistics give more detailed information about the route usage than the ridership because even if ridership of a section of the route is known, we have no idea where those people entered and exited the bus.

To do the simulations, the bus route must be simulated. A real bus route is usually circular, which means that the starting point and the ending point of the route will be the same stop. The bus generally runs the route several times a day, so there should be multiple

runs of the route in the simulation.

Another aspect to be considered is the last run. Because the real route is circular with multiple runs, there is the possibility that people will enter the bus at stops near the end of one run and stay on the bus through the start point of the next run. This is not a problem when the route is not in its last run. But during the last run of the route, there is no chance for anyone to exit during the next run since the bus will no longer run after the final stop, and this will make the distribution of the riding length different. To solve this, there is one more run of the route added to the simulation at the end of the last run. The difference between the added and the other runs is that there will be no people entering the bus, and people are only being removed. Designed in this way, the distribution of the riding length of each stop will be the same for all runs.

After the design of the route, the total number of people from the database will be separated according to a percentage. For example, if there are 10 people in the database and there are 3 runs for the route, then 40% (4 people) might be selected for the first run, 40% (4 people) for the second, and 20% (2 people) for the last run. The average number of people entering the bus at each stop are based on the information about the route, such as size and popularity. The exact number of people getting on the bus in each run is generated using the Poisson distribution with the average at each stop based on the prior information.

The true probability distribution of the number of stops that people ride is generated manually according to the sizes and locations of the stops from the route selected for simulation. For example, there are three stops $\{1, 2, 3\}$, if Stop 2 is more popular as an exit stop than Stop 1 and Stop 3, there will be more weight in the distribution of Stop 1 for Stop 2 than for Stop 3, as more people entering the bus at Stop 1 will exit the bus at Stop 2 than Stop 3. The distribution can be represented using a probability matrix, with each row representing an entrance stop, and each column representing an exit stop. Since the

route is circular, the matrix will be square. A example matrix is as follows,

$$\begin{bmatrix} 0 & 0.6 & 0.4 \\ 0.5 & 0 & 0.5 \\ 0.3 & 0.7 & 0 \end{bmatrix}, \tag{4.1}$$

where people entering the bus at Stop 1 will have 60% probability to exit at Stop 2, and 40% probability to exit at Stop 3. The probability matrix has its diagonal elements, which is the stop that a person enters the bus, assigned 0 since people will not ride the bus to their starting stop. The largest number of stops people ride is the number of stops of the route minus 1. A person entering the bus at a stop, which will select a row in (4.1) according to the stop number, will be assigned a stop to exit by drawing a random number and comparing it with the distribution of the stop (elements along the row). For example, if a person enter the bus at Stop 1, then a uniformly distributed number between 0 and 1 will be generated. If the number is smaller than 0.6, then the person will exit at Stop 2. If the number is larger than 0.6, the person will exit at Stop 3.

People in the database with their feature vectors are randomly assigned stops for entering, and randomly assigned stops to exit according to the number generated using the method above. In a random way using Monte Carlo analysis, the simulations will give a statistical description about the performance of the implementation of classification with trellis optimization on a real bus route.

From the description above, information that goes with a person has four parts.

- The stop number that the person enters.
- The stop number that the person exits.
- The circle number that the person enters.
- The circle number that the person exits.

The trellis starts in empty state with no step, and the pool is empty since no one is on the bus. At the first stop, people enter the bus, and no people exit the bus. The pool

is filled with people from the first stop. Starting from the second stop, there will be people exiting the bus and the trellis starts to extend, while the pool is continuously being filled by people entering the bus. When a person exits, he will be compared with all people in the pool, by computing the Mahalanobis distance which will be put into the trellis. The person will not be decided until one of the four criterion for decision making is met. It should be noted that the enlarging of the pool occurs after the extension of the trellis, since there is no need to compare the people exiting with those just entering the bus. When the person is decided, the stop that he exits the bus will also be recorded. This relies on the previous assumption that people are being counted correctly, since if there is a miscount or double count, there will be errors when deciding people's exiting stops.

Using the data from the stop numbers where a person enters and exits the bus, as well as the route numbers, the riding length for a person will be determined. If the stop number where a person enters is smaller than where he exits, the two stops are within the same circle. If the stop number where a person enters is larger than where he exits, the two stops are in different circles. When data for all the people is accumulated, the probability distribution of the riding length will be computed.

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 4.9. 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 4.10 and Figure 4.11, respectively. Note that for Stop 23, peaks in the rider lengths occur four stops later and then 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 errors of the simulations. The first is to use

Fig. 4.9: Map of CVTD route number 3.

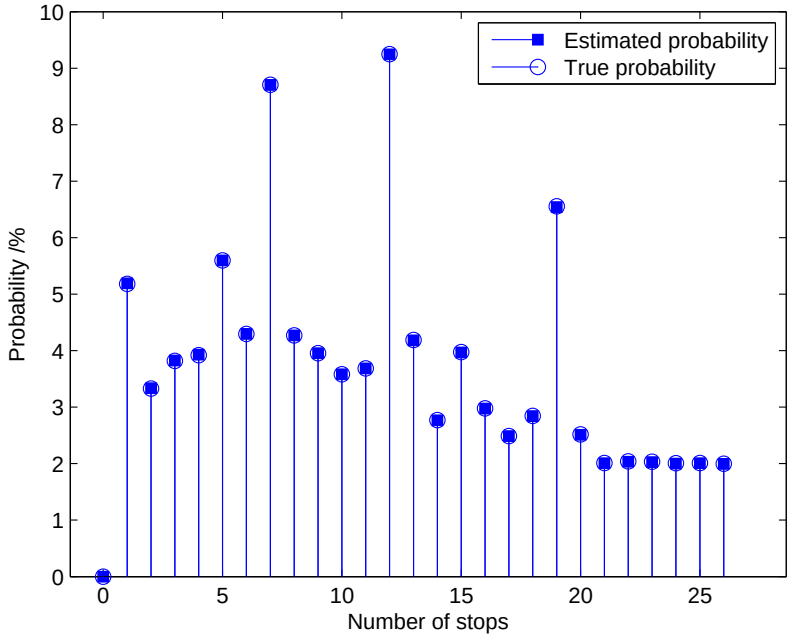


Fig. 4.10: Distribution of the number of stops that passengers take for the whole route.

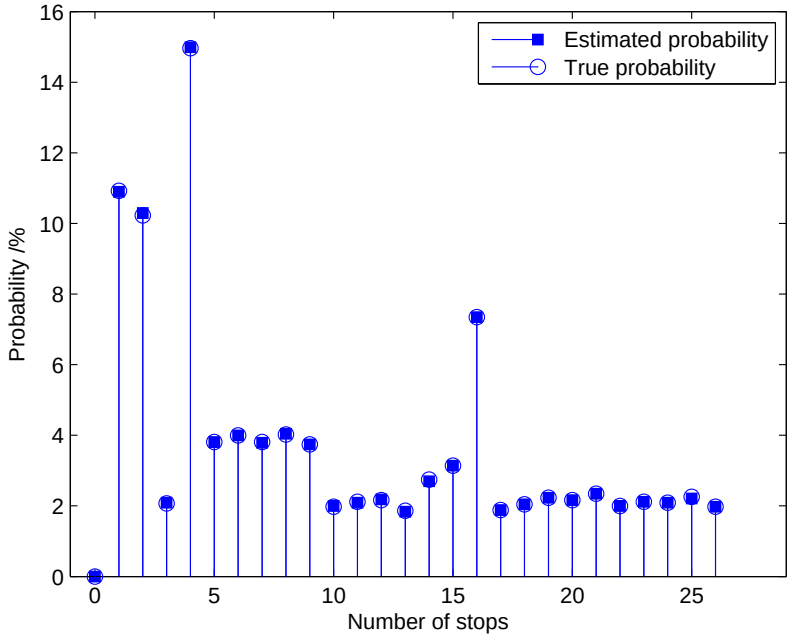


Fig. 4.11: Distribution of the number of stops that passengers take when getting on the bus at Stop 23.

the matching errors, and the second is to use the errors between the simulated probability distribution of riding length and the actual mumbers. It can be seen that some matching errors will not be reflected in the errors in estimating the probability distribution of the riding length, because the mismatch may happen within a stop, giving the correct statistics of riding length with incorrect matching results.

Chapter 5

Conclusion

This thesis explores pattern recognition applications in a bus environment. With the setup of a texel camera, more data can be gathered than traditional systems. Repeated measurements are made for each person and used for feature selection as well as further classification tasks. The SFFS feature selection algorithm with maximization of the minimum normalized Mahalanobis distance of incorrect matching as the criterion function is used for the optimized feature subset, and the modified LDA is used for classification. For the sequential estimation problems, the trellis optimization algorithm with a beam search technique is designed, implemented, and analyzed. Using the trellis, simulations of the real route can be done and more detailed statistics can be generated.

5.1 Summary of Contribution

Chapter 2 illustrates the trellis optimization algorithm in detail. After analyzing the environment and special situation of bus classification, the trellis is introduced to postpone the decision making for people exiting the bus, which will improve classification accuracy. In order to reduce the computational cost, a beam width is added to the trellis to constrain the size. The introduction of the beam width brings out four ways of making decisions for the trellis.

Chapter 3 describes the collection of data for feature vector generation and the weak points of the system as well as the solution. Many features are generated for feature selection and are grouped for convenience. The design of the SFFS algorithm is explored and the criterion functions for the SFFS algorithm are designed and compared. The criterion function using the maximization of the minimum normalized Mahalanobis distance of the incorrect matching is proven to be more effective than the marginal accuracy. The imple-

mentation of the SFFS feature selection algorithm works well on the data in hand, whereas the robustness of the algorithm is needed to be further tested.

Chapter 4 gives the simulation results of the trellis optimization algorithm first, together with the trellis behavior showing the decision making procedures and their contributions for the classification accuracy. How the classification accuracy is related to the beam width and normalized distance is also given. Then another application of the trellis is described, to estimate the probability of riding length of people entering the bus at different stops. The accurate simulation results show that the trellis optimization algorithm can be applied on the pattern recognition task and that correct classifications of people can be used for generating route statistics.

5.2 Ideas for Future Research

From the simulations, the feature selection algorithm using the maximization of the minimum normalized Mahalanobis distance is proven to be valid and effective. However, the method does not take into account the average normalized Mahalanobis distance of all training people. There might be cases where the minimum distance using the feature set is larger, but other distances using the feature set which are larger might be near the minimum distance, compared with the feature set whose minimum distance is smaller, but other distances using the feature set which are larger are much larger than the minimum one. Simulations need to be done to test how the trellis will perform under these situations. Also, the feature selection algorithm can combine the minimum and average normalized Mahalanobis distance to form a new selection algorithm.

The feature selection algorithm uses the SFFS as the wrapper algorithm with the criterion function. The SFFS is not an exhaustive search, but is still a greedy algorithm searching the optimized results. Other wrapper algorithms, such as the GA, can also be used to search the optimized results in a randomized way. Also, the SFFS only deals with the performance of adding and subtracting one feature, but for a combinatorial problem, the adding and subtracting of two or more features may also affect the search results much. Consequently, it should be explored on how to balance the accuracy and complexity of the

feature selection algorithm.

The feature selection in this thesis is based on a database of 68 people. It is believed that more people should be used for the feature selection task for a more robust performance. In addition, the people used should be diverse in height, color, and texture of clothes, so that the features selected will be much useful for classification.

References

[1] EZ People Counter, "People Countering System," [http://www.ezpeoplecounters.com/], 2011.

[2] ACOREL, "ACOREL Onboard Counter," [http://www.acorel.com/], 2011.

[3] SenSource, "People Counter," [http://www.sensourceinc.com/peoplecounters.htm], 2011.

[4] "Time-of-flight Camera," [http://en.wikipedia.org/wiki/ToF_camera/], 2011.

[5] "Canesta," [http://en.wikipedia.org/wiki/Canesta/], 2011.

[6] B. Boldt, "Point Cloud Matching with a Handheld Texel Camera," Master's thesis, Utah State University, Logan, UT, 2007.

[7] C.-H. Chen, Y.-C. Chang, T.-Y. Chen, and D.-J. Wang, "People Counting System for Getting In/Out of a Bus Based on Video Processing," *ISDA Eighth International Conference on Intelligent Systems Design and Applications*, pp. 565–569, 2008.

[8] PMD Technologies, "PMD Camcube 3.0," [http://www.pmdtec.com/products-services/pmdvisionr-cameras/pmdvisionr-camcube-30/], 2011.

[9] T. Yang, Y. Zhang, D. Shao, and Y. Li, "Clustering Method for Counting Passengers Getting in a Bus with Single Camera," *Optical Engineering*, vol. 49, 2010.

[10] J. A. Sallay, "Automatic People Counting and Matching," Master's thesis, Utah State University, Logan, UT, 2009.

[11] I. A. Gheyas and L. S. Smith, "Feature Subset Selection in Large Dimensinality Domains," *Pattern Recognition*, vol. 43, pp. 5–13, 2010.

[12] M. Dorigo and L. M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 53–66, 2002.

[13] W. Siedlecki and J.Sklansky, "A Note on Genetic Algorithm for Large-scale Feature Selection," *Pattern Recognition Letters*, vol. 10, pp. 335–347, 1989.

[14] R. G.-Osuna, "Pattern Analysis for Machine Olfaction: A Review," *IEEE Sensor Journal*, vol. 2, pp. 189–202, 2002.

[15] L. M. Lix and T. T. Sajobi, "Discriminant Analysis for Repeated Measures Data: A Review," *Frontiers in Psychology*, vol. 1, pp. 1–9, 2010.

[16] A. Roy and R. Leiva, "Classification Rules for Multivariate Repeated Measures Data with Equicorrelated Correlation Structure on Both Time and Spatial Repeated Measurements," uTSA, College of Business, June 2009.

[17] M. Krzysko and M. Skorzybut, "Discriminant Analysis of Multivariate Repeated Measures Data with a Kronecker Product Structured Covariance Matrices," *Statistical Papers*, vol. 50, pp. 817–855, 2009.

[18] A. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, vol. 13, pp. 260–269, 1967.

[19] H. Tanaka, Y. Hirakawa, and S. Kaneku, "Recognition of Distorted Patterns Using the Viterbi Algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, pp. 18–25, 1982.

[20] T. G. Dietterich, "Machine Learning for Sequential Data: A Review," *Structural, Syntactic, and Statistical Pattern Recognition*, pp. 15–30, 2002.

[21] M. Grafmuller, "A First Approach to Trellis-Based Classification," *Proceedings of the 2009 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vison and Fusion Laboratory*, pp. 175–185, 2009.

[22] X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm and System Development.* Upper Saddle River, NJ: Prentice Hall, ch. 13, pp. 585–636, 2001.

[23] J.-S. R. Jang and S.-S. Lin, "Optimization of Viterbi Beam Search in Speech Recognition," *Proceedings of International Symposium on Chinese Spoken Language Processing*, pp. 114–118, 2002.

[24] *Electronic Perception SDK Reference Manual*, Sunnyvile, CA: Canesta Inc., 2004.

[25] K. Barnard, V. Cardei, and B. Funt, "A Comparison of Computational Color Constancy Algorithms. I: Methodology and Experiments with Synthesized Data," *IEEE Transactions on Image Processing*, vol. 11, pp. 972–984, 2002.

[26] R. C. Gonzalez and R. E. Woods, *Digital Image Processing.* Upper Saddle River, NJ: Prentice Hall, ch. 11, pp. 852–854, 2007.

[27] A. Pressley, *Elementary Differential Geometry.* London, UK: Springer-Verlag, ch. 8, pp. 179–196, 2010.

[28] S. F. Cotter, K. Kreutz-Delgado, and B. D. Rao, "Backward Sequential Elimination for Sparse Vector Subset Selection," *Signal Process*, vol. 81, pp. 1849–1864, 2002.

[29] S. Colak and C. Isik, "Feature Subset Selection for Blood Pressure Classification Using Orthogonal Forward Selection," *Proceedings of IEEE 29th Annual Bioengineering Conference*, pp. 122–123, 2003.

[30] P. Pudil, J. Novovicova, and J. Kittler, "Floating Search Methods in Feature Selection," *Pattern Recognition Letters*, vol. 15, pp. 1119–1125, 1994.