2D/3D Staircase Detection and Localization

Wang Sisong

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University in partial fulfillment of the requirement for the degree of Master of Engineering

2010

Acknowledgements

I would like to express my sincere gratitude and appreciation to my supervisor, Associate Professor Wang Han for his enthusiastic encouragements, excellent guidance and sound advice throughout the duration of this project.

Appreciation is also extended to all the colleagues and students in the Autonomous Robotics Research Laboratory (ARRL) at the School of Electrical and Electronic Engineering for their support and friendship during the past 2 years.

Finally, I would like to thank the School of Electrical and Electronic Engineering at Nanyang Technological University and the university itself for offering me this opportunity to pursue a higher degree and a valuable research scholarship during the course of my study.

Table of Contents

Summary

This work presents a vision-based staircase identification system which comprises two sequential tasks: 2D staircase detection followed by 3D staircase localization. The 2D detector pre-screens the input image and the 3D localization algorithm continues the task of retrieving geometry of the staircase on the reported region in the image.

The 2D staircase detector defies conventional thinking by exploring the issue from the machine learning perspective. It implements a binary classifier based on Viola-Jones rapid object detection framework. This thesis introduces a novel set of PCA-based Haar-like features which extends the classical Haar-like features from local to global domain and are extremely efficient at rejecting non-object regions at the early stages of the cascade. To adapt the framework to the context of staircase detection, modifications have been made on the scanning scheme, multiple detections integrating scheme and the final detection evaluation metrics.

The “V-disparity” concept, originally used for ground plane estimation in autonomous navigation, is applied to detect the planar regions on the staircase surface. “V-disparity” can locate 3D planes quickly from disparity maps. However, its inherent drawbacks impede it from being used in wider applications. We have made improvements to the original “V-disparity” concept to adapt it to the application of 3D staircase localization.

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Motivation

In recent years, the topic of autonomous navigation for unmanned ground vehicles (UGV) has been a hotspot in the computer vision community. The detection of road-elevated delimiters, such as curbs, is of great importance to the estimation of the vehicle's position relative to the lane for safe driving. Curb detection and localization is a well established issue in the literature [1], [2], [3], [4]. Some researchers extend the problem of curb/step detection and localization to staircase detection and localization, mostly with range data obtained from a diverse set of sensors being used for the purpose [5], [6].

Staircases are ubiquitous man-made land features, designed to bridge large vertical distances for humans. However, the ability to detect and climb the stairs automatically is still a great challenge to robots. Intensive research on state-of-the-art stair climbing algorithms [7], [8], [9], [10] is still ongoing. Stair detection is not a matter of consideration in these researches as they assume the robots have been aligned directly heading the staircase. For a robot to be fully autonomous, automatic detection and localization of the staircases in the world environment is of equal importance to stair climbing itself. This requires the robot to gain real time information about the environment, such as the minimum robot stair distance, the dimension and orientation of the steps, the free negotiable area on the staircase surface etc.

This thesis confines itself to a vision strategy for staircase detection and localization in the sense of forgoing the aid of any other means of sensing systems. An established paradigm [11], [12], [13], [14] for vision-based staircase detection relies on the accurate extraction of linear patterns that correspond to the perspective projections of parallel lines in Euclidian space. This type of method will inevitably make lots of misclassification errors. Consider a highly structured urban setting where floors and pavements are covered with checkered tiles. The dominant features that could be extracted from the scene are a set of concurrent lines. However, none of them belongs to a staircase. Stereo vision could solve the problem at the cost of extra processing time.

This thesis does not attempt to cover every aspect on autonomous stair climbing, but focuses on the domain of (stereo) vision-based 2D/3D staircase detection and localization. The proposed algorithm consists of 2 stages. The first stage employs a fast 2D staircase detector to pre-screen every frame acquired from the camera, the second stage is triggered only when staircases are found from the previous stage. 3D operation is performed only on the region reported by the 2D staircase detector.

1.2 Objective

This project attempts to think outside the box and develop creative solutions to the old problem of staircase detection and localization. The final detector should be able to detect most of commonly seen staircases with parallel edges. Like most of the existing methods we do not consider spiral staircases.

The solution should avoid using straight lines as the only source of clues for monocular analysis, and the performance of the 2D staircase detector should not rely on the availability and distribution of linear features.

For 3D staircase localization, it must produce accurate 3D results, including the camera staircase geometry and the dimensions of each step of the staircase.

1.3 Major contribution of the thesis

This thesis presents 3 major contributions to the community:

- *Improved Viola-Jones object detection framework for staircase detection*

  Viola-Jones object detection framework [26] presents the machine with a set of labeled training samples, both positive samples which are instances of target objects and negative samples which are arbitrary images that are free of the intended targets. The machine searches a broad-based feature pool to find statistically best features that separate the positives from negatives with minimum error rate. Finally, these selected features are synthesized into a so-called “strong classifier” that classifies any valid input into either positive group or negative group.

  In the context of staircase detection, we have made the following improvements and changes to the original Viola-Jones object detection framework:

1. Enhanced Haar-like feature pool with global contrast descriptors termed "PCA-based Haar-like features", which will soon be singled out as another major contribution of the thesis (Chapter 3).
2. Redefines positive samples from the final intended targets (staircases) to a part of a staircase, in order to weaken the large intra-class uncertainties across individual staircases (Section 3.3).
3. Uses the statistically more reasonable Real AdaBoost algorithm to replace the simple Discrete AdaBoost in training. (Section 4.2).
4. A different scanning scheme employs a constant sampling rate at various scales of the cascade detector (Section 4.4).
5. Because the positive samples fed into training are parts of staircases, integration of primitive detections is necessary. The process greatly reduces the chances of occurrence of false positives. (Section 4.5)
6. A new real-valued rating scheme is introduced to assess the accuracy of the final detections instead of the Boolean classification scheme (Section 4.6.1).

- *PCA-based Haar-like features*

  This thesis proposes a novel set of Haar-like features, termed "PCA-based Haar-like features". The features extend the classical Haar-like features from merely local contrast descriptors to global filters that are extremely efficient at rejecting non-object regions in the early stages. A few PCA-based Haar-like features could

work as well as a large number of classical features. These features are manually designed by users based on the discretized PCA images of the positive data set. On top of that, PCA-based notion can be generalized to the design of target-oriented Haar-like features for any objects.

- *"V-disparity" for quick plane extraction from disparity maps*

  For 3D staircase localization, we have tried to work in the "V-disparity" space. Since the advent of the "V-disparity" concept, it has been widely used in the applications of ground plane or road surface estimation for its simplicity and high-speed capability. The "V-disparity" method is very fast at plane extraction and staircases can be modeled as a collection of planar regions. The "V-disparity" concept is used to quickly locate these planar regions on the staircase surface.

1.4 Organization of the thesis

The thesis is organized into 7 chapters with the first and last chapter introducing and summarizing the thesis respectively. Chapter 2 through 6 can be divided into 2 parts. The first part covers Chapter 2 to 4, which explores the task of 2D staircase detection. Chapter 2 reviews conventional line-based 2D staircase detection algorithms and the well-known Viola-Jones object detection framework. Chapter 3 introduces a novel set of PCA-based Haar-like features and how they were designed. Chapter 4 proposes the 2D staircase detection algorithm based on Viola-Jones framework and presents thorough

and comparative experiments. Chapter 5 and Chapter 6, discusses the problem of 3D staircase localization. Chapter 5 reviews the “V-disparity” concept and briefly illustrates the derivation of relevant formulae. Chapter 6 proposes the 3D staircase localization algorithm and gives 2 experiment scenarios.

Chapter 2 An Introduction to Viola-Jones Face Detector

2.1 Introduction to 2D staircase detection

Conventional 2D staircase detection methods are based on the recovery of the interrelated stair edges from monocular images or video sequences [11], [12], [13], [14]. The underlying principles for these methods, at the highest level of abstraction, are identical, that is to extract a group of concurrent lines that are possibly projected from a world space structure containing parallel lines. However, they can be broadly categorized into two groups in terms of their objectives. The first group of methods aims at guiding their mobile robots for autonomous stair climbing, assuming the robots were on the stairway. The heading direction and robot position with respect to the stair could be estimated with the extraction of the line segments correspond to the stair edges. These parameters are then used to steer the robot. The second group of methods aims at detecting staircases from given images by seeking for groups of lines, exhibiting certain pattern and geometric relations. This is in line with our interest of discussion here. The emphasis of this part is thus placed on the second type of the methods.

Despite the objectives diverge; the process of implementation of the conventional methods could be summarized into three stages. In the first stage, straight lines are detected. The best group of parallel lines or concurrent lines is then extracted if they were present in the subsequent stage. Stage three differs in how the lines are used. The first group of methods will use these lines to recover the robot-stair offset parameters,

while the latter seeks further support to verify the hypothesis that the pencil of lines or partial of them are projected from a stair-like structure in the space. Simply speaking, it tries to search for two groups of interleaving edges that are equally spaced in the space. The following subsections will brief each of the stage.

2.1.1 Line detection

Lines are important intermediate constructs, frequently used in high-level visual interpretation processes, such as staircase detection. Research in line detection has continued active from the landmark paper of Roberts [15] in computer vision history to the current time.

Following an initial edge extraction process, various techniques have been proposed to aggregate the local information into more global line-like structures, a difficult task in many domains. These methods include Hough transform [16], edge tracking and contour following [17], gradient orientation guided points clustering [18], curve fitting [19], graph theoretic methods[20], relaxation algorithms[21]… a long list to recall.

With the variety of methods proposed, the choice is surely application dependent. This section will not touch on the working principles of each method, but put the focus on the investigation of which should be potentially superior to the others in the context of staircase detection.

Hough transform is most widely used and often appears after Canny edge detection. This classic combination is used in [13], [14] to extract the stair edges. However, Hough

transform may end up with undefined results in cluttered edge image. Unluckily, this is often the case in staircase images. [12] has envisaged this issue, and introduced a template operation to the edges, attempting to filter out the unwanted details. The idea is rather simple: if an edge gets through or partially get through the template boundary, it may be a line within. If no edge pixels reside on the boundary of the template, the inner content can be eliminated. This filtering technique is effective for isolated pixels, but may fail if the trivial edges are ramblingly linked with each other. The authors then fit remainder regions as lines and link the collinear segments. Curve fitting makes sense since [12] assumes the robot is on the stair. [11] also perceives the inability of Hough transform and generic edge detectors, but from another perspective. It claims that shadows can break stair edges and the generic edge detectors can cause the linear characteristics completely missing in low contrast regions. The authors take a least-commitment approach in the initial edge pixels detection by omitting the thresholding processing entirely. Edge tracking and contour following method is then used to grow the line segments by iteratively adding new pixels into the existing edge and updating the line parameters. The piecewise linear segments are then linked based on some grouping criteria until no new line segments occur.

As we shall see, no matter what method is used, it seldom gets an elegant result. A robust algorithm is always facing the confrontation of large cross individual variations of stair instances, a single tuning in the full spectrum of parameterization may cause the detection results change drastically, and a lot of other unforeseen problematic situations. However, we can still draw an empirical conclusion, that edge tracking and

contour following is better than the others. Another benefit associated with edge tracking is that when we know the stair edges are near horizontal, line growing is directed towards the right direction.

2.1.2 Parallel and concurrent lines detection

If lines are the cornerstones of staircase detection, the extraction of parallel and concurrent lines establishes the skeleton of the whole work.

When the stairs are fronto-parallel to the image plane, parallelism is preserved. For the stair climbing applications [11], [12], the task could be reduced to parallel lines detection. While for generic staircase detector, it requires further consideration. It is well known that the projections of 3D parallel lines intersect at a point, called vanishing point. They are invariant features of a scene and have been employed for both qualitative and quantitative image analysis. Qualitatively, vanishing points can be used to group lines for image matching, quantitatively they have been suggested for use in camera calibration or for solving inverse perspective mappings in structured machine vision applications [22].

There are two ways to find concurrent lines. One approach is to search for vanishing points using Hough transform [23]. The vanishing points are characterized as those points where most of the supporting line segments intersect. However, Collins and Weiss [24] considered vanishing point computation as a statistical estimation problem and observed that it is not reliable when not many lines are passing through that point.

A different approach is proposed in [14] for concurrent lines detection. Starting from the lines detected in stage one, the algorithm first picks out groups of lines within an preset angle range as seeds and checks each seed for concurrency, discarding those which do not intersect well. This is done by applying a least square procedure to the seed lines to find the point which is closest to these lines. Afterwards, the residual is computed as an indicator of how well the lines intersect. If it is less than a certain threshold, the lines are considered concurrent, else they are eliminated. Now, for each resulting seed, it seeks further support from other lines to expand the group. The group with the maximum number of supporting lines is selected as the hypothesis for a staircase. Moreover, the algorithm discards short edges and those that are far away from the rest which are likely to arise from features other than a staircase.

The concurrent lines detection algorithm introduced in the last paragraph generally works well for staircase detection. Reasons are that, firstly we have narrowed down the searching domain of the line segments since we roughly know the orientation of the edges; secondly, stair edges are long and dominant linear features, this avoids consideration of short non-stair edges. Lastly, the subsequent partition stage allows certain degree of outliers to coexist with the actual stair edges.

2.1.3 Partition staircase edges

For the first group of staircase detection methods, the parallel lines will be used to estimate the robot-stair offset parameters to help the steering of the vehicle. However, for the second type of methods, an additional layer of constraints is needed to verify

that the underlying structure consists of two sets of equally spacing and interleaving parallel lines, the convex and concave stair edges. This is now a much stronger constraint for a regular staircase compared to merely searching for structures with parallel lines. In addition, the partition of the concurrent (parallel) lines has the potential of recovering missing lines and removing outliers. [14] gives three ways to partition the edges.

2.1.3.1 Cross-ratio

In mathematics, the cross-ratio is defined on an ordered 4-tuple of distinct points on a line and an ordered 4-tuple of concurrent lines in a plane. The cross-ratio of four collinear points A, B, C and D shown in Figure 2.1 is defined as:

$$(A, B; C, D) = \frac{AC \cdot BD}{BC \cdot AD} \tag{2.1}$$

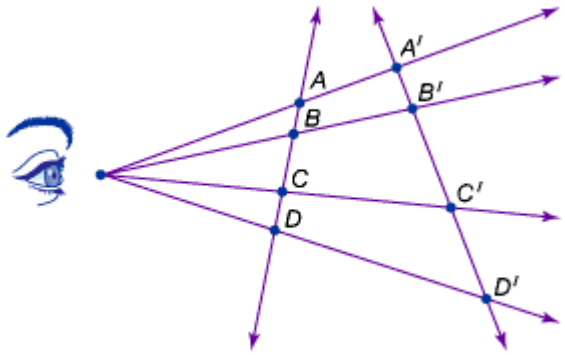

Figure 2.1: Cross-ratio

Cross-ratio is a projective invariant in a sense that it is preserved by the projective transformation. Thus, we have:

$$(A', B'; C', D') = (A, B; C, D) \tag{2.2}$$

Suppose A', B', C' and D' are the four distinct points on a line passing through four consecutive concave (convex) stair edges. For a regular staircase, we can derive the following length relation:

$$A'B' = B'C' = C'D' \tag{2.3}$$

When the points are projected onto the image plane, we have:

$$(A, B; C, D) = \frac{AC \cdot BD}{BC \cdot AD} = \frac{A'C' \cdot B'D'}{B'C' \cdot A'D'} \approx 1.33 \tag{2.4}$$

This numerical invariant is employed to gather relevant lines. The best two groups of alternate lines are then selected as the stair edge projections if they are present.

2.1.3.2 Linear homography

In computer vision, we define linear homography as a projective mapping from one line to another. The mapping of points on a line to points on a raster line of our image is an example of linear homography. It is possible to express this mapping in terms of matrix multiplication if we use homogeneous coordinates to express both the viewed point and the point on the raster line to which the viewed point is projected:

$$\begin{bmatrix} v \\ 1 \end{bmatrix} = \begin{bmatrix} a & b \\ c & 1 \end{bmatrix} \begin{bmatrix} v' \\ 1 \end{bmatrix} \tag{2.5}$$

Where v is the coordinate along the raster line and v' is the coordinate of the viewed point along the space line. If we define an arbitrary space line passing through the concave (convex) stair edges, the intersections of the line with the stair edges will be projected into collinear points in the image. If we have three of these mappings, we could determine the homography matrix explicitly. Technically, it is possible to group any three lines in the image as the projections of three consecutive stair edges which have equal spacing between each other. Thus we can use some consecutive integer values for v' (e.g. 0, 1, 2) in Equation 2.5 to calculate the coefficients in the homography matrix. We can then substitute neighboring integer values (e.g. -3, -2, -1, 3, 4, 5…) into Equation 2.5 to estimate the projections of these points and seek support in the image. This method can also recover missing edges. We will retain the ones with good support (≥ 4) since every selection will have at least three supporters. Among the good selections found, we need to choose two of them which do not overlap with each other and their union has a maximum size among all the other choices.

$$P_1 \bigcap P_2 = \phi, \qquad P_1 \bigcup P_2 \subseteq P \tag{2.6}$$

Where P_1 and P_2 represent the two partitions. If the two partitions cannot be found or their union size is far too low, that means the edges found cannot be partitioned into two sets of equally spaced lines or too many edges are missing.

2.1.3.3 Intensity variation

The above two methods use rigid geometric constraints to partition the stair edges. One of the most useful contributions made by [14] is the observation that stair edges in gray scale images usually represent intensity delimiters from dark to light or from light to dark. The intensity profile along a line drawn in Figure 2.2 (a) is given in Figure 2.2 (b).

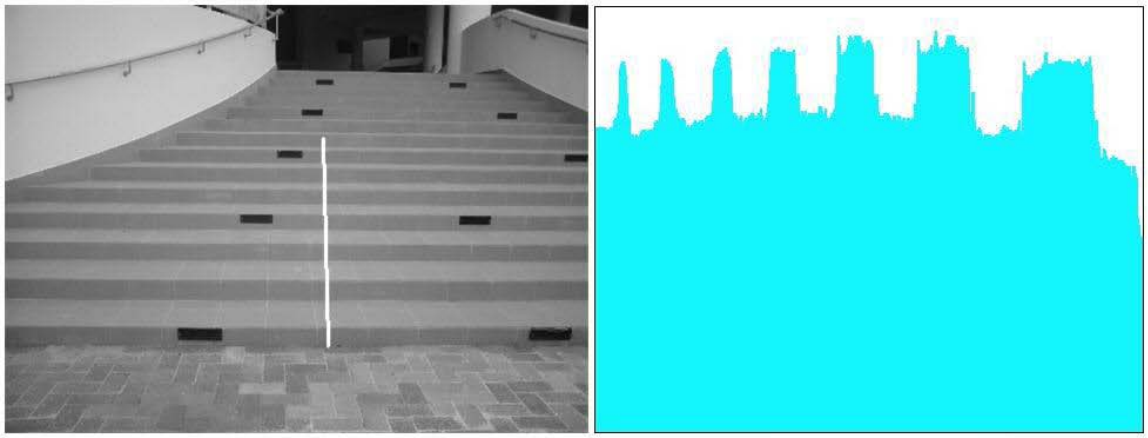

Figure 2.2: Intensity variation (a) original image with a line crossing stair edges (b) intensity profile along the line

It is clear from the profile that there exists a pattern of up and downs of the intensity, corresponding to the tread and riser of the stair steps. In order to facilitate the partition, the average intensity along that line is used to threshold the image to obtain a binary image as shown in Figure 2.3.

Figure 2.3: Binary threshold stair image

From the binary image, it is possible to quickly partition the edges and remove outliers. Geometric constraints can be applied to refine the partitions as well.

Using intensity variation is simple. However this does not always make sense.

2.1.4 Discussions

We have categorized the conventional line-based staircase detection methods into two groups and introduced the three main stages implemented in these methods. From a pragmatic perspective, it will be hard to draw a convincing conclusion that lines with certain pattern drawn from an arbitrary image can prove the existence of a staircase object if there's no reliable 3D data to back the argument, since the equal spacing constraint used in stage three are ubiquitously present in artificial objects. In view of

this limitation, [11] [12] are not addressing the issue of stair detection in the literal sense, but are taking advantage of special imaging effects of stair edges to achieve a specific engineering task of autonomous stair climbing for robots. However, they provide useful insights into solving the elementary but knotty image processing issues, like line extraction and linking through cluttered images etc.

Nevertheless, [13] is the most comprehensive piece of work that I have ever seen as far as 2D stair detection is concerned. It actually has 3 parts, and we have only introduced the second part so far, which is the short-range staircase detection.

The first part tackles the detection of distant staircases where stair edges may not be resolvable from edge images. The method treats the staircase pattern as a type of texture for detection. It convolves the image with a series of Gabor filters [25] at various frequencies and orientations, then finds a combination of a frequency and an orientation that gives the best response in that direction but poor at the orthogonal as opposed to the best orientation. The region corresponds to the best response in the convolved image is retained as the staircase region.

In the last part, the method tries to recover the staircase pose by a homography search approach. It iteratively applies an updated slope (initial guess: $45°$) to the search space to find the best vertical rotation angle such that the homograply transformation turns out a set of horizontal lines and 2 sets of equally-spaced edges in the image. The process is iterated three times before converging on the final values of the staircase slope angle

and the vertical rotation angle, which is equivalent to the robot-stair offset angle in the first category of stair detection methods.

As you may have expected, line-based staircase detection relies heavily on the robustness of the line detection algorithm. Even this problem is cleared; you still cannot convince people that you've detected a staircase not a cluster of zebra crossing.

This project discards the conventional methods used so far, and uses a machine learning based algorithm to train a binary classifier as a staircase detector. This was inspired by the success of Viola-Jones face detector [26] and its applications for the detection of various objects. Next section will give an introductory description of machine learning concept and a detailed description of one of the machine learning algorithms: AdaBoost, which is successfully used in Viola-Jones face detector. The chapter will be then concentrating on Viola-Jones face detection algorithm.

2.2 Machine Learning

This thesis tries to introduce the Machine Learning concept into staircase detection. Machine learning is a scientific discipline that is concerned with the design and development of algorithms that allow computers to learn based on data, such as from sensor data or databases. A major focus of machine learning research is to automatically learn to recognize complex patterns and make intelligent decisions based on data. In recent years, machine learning has many successful applications in computer vision,

such as optical character recognition, human face and pedestrian detection, image segmentation and a lot more [26], [27], [28].

Machine Learning techniques can be divided into a broad range of categories such as supervised learning, unsupervised learning, semi-supervised learning, analytical learning, reinforcement learning, active learning etc. Supervised learning deals with learning a function from labeled data sets. Unsupervised learning is concerned with algorithms that form clusters or groupings to learn patterns or associations from datasets that have no specified class labels. Semi-supervised learning deals with data sets that are combination of labeled and unlabeled samples. Analytical learning uses data sets that have background knowledge or domain theory instead of labeled examples. Reinforcement learning deals with algorithms that learn a control policy through reinforcement from an environment. Active learning is concerned with unlabeled data sets that can be labeled in a sequential process in the sense that the corresponding labeled examples contribute to obtain a more accurate function.

This thesis is focused on supervised learning. The task of supervised learner is to predict the value of any valid input instance after having been presented a number of labeled training samples. Common supervised learning techniques include k-nearest neighbors, neural networks, decision trees, support vector machines and boosting. The AdaBoost [29] algorithm, a member of boosting family, is used to train the staircase detector presented in this thesis.

2.2.1 Boosting

Machine learning studies techniques to train the machines to make accurate predictions based on previous observations. However, a highly accurate predication rule is extremely hard for the machines to learn based on finite samples; nevertheless, it may not such a difficulty to come up with some moderately accurate rules of thumb. The method or algorithm to find these moderately accurate rules of thumb is called "weak" or "base" learning algorithm. The question is that whether a "weak" learning algorithm that performs just slightly better than random guessing can be "boosted" into an arbitrarily accurate "strong" learning algorithm. Kearns and Valiant [30], [31] were the first to pose this question. Later, Schapire [32] proposed the first provable polynomial-time boosting algorithm to construct such a better ensemble of "weak" learners in 1989. The AdaBoost algorithm, introduced in 1995 by Freund and Schapire[33], is considered as a first path-breaking step toward practically feasible boosting algorithms.

2.2.2 AdaBoost

The AdaBoost algorithm for binary classification is the most well-known boosting algorithm. It takes as input a training set $(x_1, y_1), \ldots, (x_m, y_m)$, where each x_i belongs to some domain or instance space X, and each label y_i is in some label set Y. When applied to object detection, the training set is consisted of two groups of examples, labeled as positive samples and negative samples respectively. Positive samples are simply images of target objects, which must be carefully cropped to trim off background regions. While negative samples are randomly selected image sub-windows which do

not contain any instance or part of the target object. In the binary classification context, the label set Y can be denoted as $Y = \{-1,+1\}$.

- Given : $(x_1, y_1), \cdots, (x_m, y_m)$ where $(x_i, y_i) \in X \times \{-1,+1\}$
- Initialize $D_1(i) = 1/m$.
- For $t = 1, \cdots, T$
  - Train weak learner using distribution D_t.
  - Get weak hypothesis $h_t : X \to \{-1,+1\}$ with error:

$$\varepsilon_t = \frac{\sum_i D_t(i)\left|h_t(x_i) - y_i\right|}{2} \tag{2.7}$$

  - Choose

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \tag{2.8}$$

  - Update the weights:

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t} \tag{2.9}$$

  where Z_t is a normalization factor chosen so that D_{t+1} will be a distribution.

- Output the final hypothesis:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{2.10}$$

Table 2.1: The boosting algorithm, AdaBoost

Table 2.1 illustrates the pseudocode of AdaBoost algorithm. The main idea of AdaBoost is to maintain a distribution of weights over the training set. Samples are initially weighted depending on the application, and are often set with equal weights. This initial distribution is denoted as D_1. The weight of the distribution on training example i on round t is denoted as $D_t(i)$. AdaBoost repeatedly calls a weak or base learning algorithm in a series of rounds $t = 1, \cdots, T$. Over the rounds, weights of incorrectly classified examples are increased so that the next weak learner is forced to focus on the hard examples.

The weak learner's job is to find a weak classifier $h_t : X \to \{-1, +1\}$ appropriate for the distribution D_t such that the weighted in-sample misclassification error rate ε_t is minimized as shown in Equation 2.7.

Without loss of generality, we can assume that $\varepsilon_t \leq \frac{1}{2}$, since these weak classifiers predict outcomes just slightly better than random guessing.

Once the weak classifier has been trained, AdaBoost chooses a parameter α_t which measures the performance of h_t on the training examples. Intuitively, α_t gets larger as ε_t gets smaller. α_t is used to update the distribution such that the weights of misclassified examples are increased while decreasing the weights of correctly classified examples. Thus, the weight tends to concentrate on "hard" examples. The final ensemble is a linear combination of the T weak classifiers as shown in Equation 2.10, where α_t is the weight assigned to h_t.

AdaBoost can be extended to multi-class classification problem which is not detailed here. Practically, AdaBoost is easy to program, it has no parameters to tune. It requires no prior knowledge about the weak learner and so can be flexibly combined with any method for finding the weak classifiers. Finally, it gives a set of theoretical guarantees, given sufficient data, and a weak learner that can reliably provide only moderately accurate weak hypotheses.

2.3 Viola-Jones face detector

In 2001, Viola and Jones [26] published their illustrious cascade face detector that is the most successful application of AdaBoost algorithm. Compared with its predecessors [34], [35], Viola-Jones method is the first demonstrates real-time performance for human face detection. Its impressive performance helped to “boost” the recognition of AdaBoost algorithm against other well-known machine learning techniques in the pattern recognition community. A lot of related works, established on the basis of Viola-Jones face detection framework and focused on partial improvements [36], [37], [38] and/or extended applications for the detection and tracking of other objects, such as soccer ball [39], pedestrian[40], license plate[41] and various body features[42],[43], [44], have been reported in recent years. To the best of the author’s knowledge, no such attempt has been made in the detection of staircases. It was always our wish to apply this versatile framework to 2D staircase detection since the current methods have some inherent drawbacks.

The success of Viola-Jones method relies on a combination of four key concepts: A simple rectangular feature, called Haar-like feature; a new image representation called the "Integral Image" which allows rapid calculation of Haar-like features; the AdaBoost algorithm, which selects a small number of critical visual features and guarantees the accuracy of the detector; and finally a cascaded structure of boosted classifiers, which discards background regions quickly while concentrating on promising object-like regions. These concepts can be generally divided into two functioning groups in terms of different purposes. The Haar-like feature and integral image representation, as well as the cascaded classifiers are powerhouses for rapid detection, while the AdaBoost algorithm safeguards the detection accuracy.

2.3.1 Haar-like feature

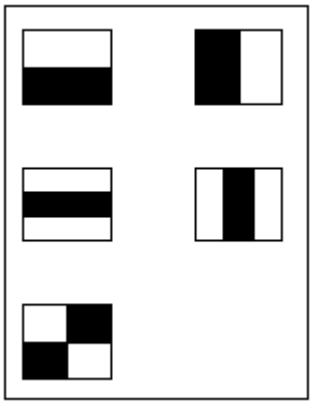

Figure 2.4: Basic Haar-like feature set

The rectangular features used by Viola-Jones are reminiscent of Haar wavelets which have been used by Papageorgiou et al [45]. The Haar wavelets are natural set basis

functions which encode differences in average intensities between adjacent regions. Papageorgiou's Haar wavelets actually operate within squares rather than arbitrary rectangles. They shift wavelets at scales of 4x4 pixels and 2x2 pixels in 19x19 sample face images and eventually generate 1734 square features. Viola-Jones redesigned the Haar wavelets into Haar-like features, which operate on arbitrary rectangles and yield a much more redundant set of around 160,000 instances at 24x24 base resolution. Examples of Viola-Jones basic Haar-like features are shown in Figure 2.4.

A Haar-like feature is defined as the difference of the weighted pixel sums of areas (black and white rectangles) inside the outmost rectangle. Each sub-rectangle is weighted such that the weighted area of positive (white) regions equals that of negative (black) regions. The values of Haar-like features indicate certain local characteristics of the target object. For example, a 2-rectangle feature can indicate where the border between a dark region and a light region lies.

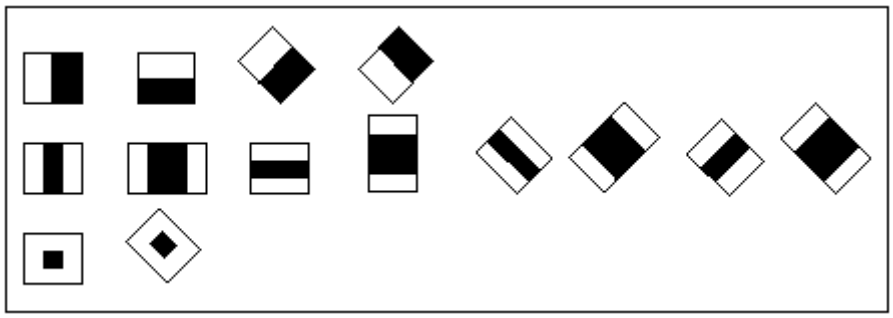

Figure 2.5: Extended Haar-like feature set

Lienhart and Maydt [46] presented an extension to the basic set with 45 rotated features as shown in Figure 2.5. They also demonstrated comparable computation

efficiency for their tilted Haar-like features. The extended set of tilted Haar-like features adds additional domain knowledge into the learning framework, which is hard to learn from the basic set.

The motivation to use features rather than raw pixels is to reduce intra-class, and increase inter-class variability. On top of that, features operate faster than directly manipulating pixels. This is also concurrent with our efforts of developing real-time object detection algorithms.

2.3.2 Integral Image

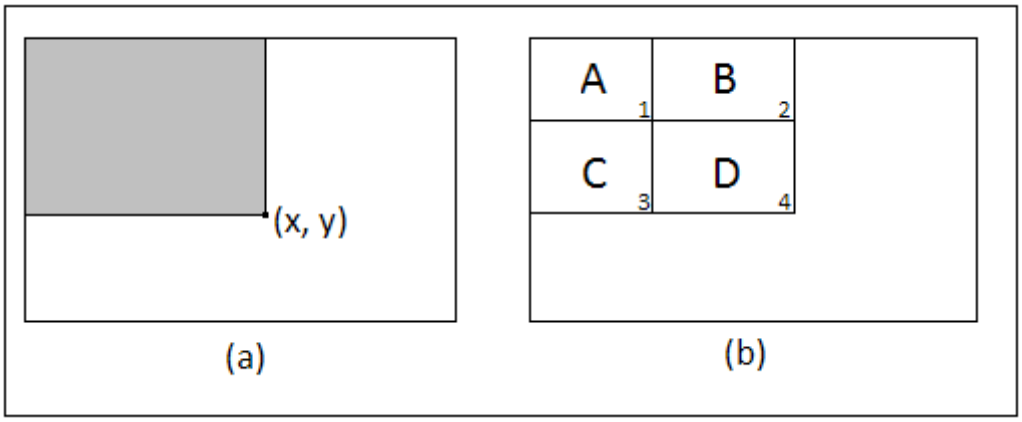

Figure 2.6: Integral image representation

Integral image is a new image representation technique introduced by Viola-Jones for the purpose of fast Haar-like feature calculation. The integral image has its roots in "summed area table" used in graphics [47].

In Figure 2.6 (a), the integral image at location (x, y) contains the sum of all pixels above and to the left of the concerned pixel (x, y) , inclusive:

$$ii(x, y) = \sum_{x' \le x,\, y' \le y} i(x', y') \tag{2.11}$$

Where, $ii(x, y)$ is the integral image and $i(x, y)$ represents the original image.

Haar-like features require the calculation of pixel sums within rectangles. The integral image fits that purpose well. In Figure 2.6 (b), A, B, C and D are the pixel sums for their respective rectangular regions, giving us the following equations:

$$\begin{aligned} ii(1) &= A \\ ii(2) &= A + B \\ ii(3) &= A + C \\ ii(4) &= A + B + C + D \end{aligned} \tag{2.12}$$

With these values available at locations $1, 2, 3$ and 4 in the integral image, we can calculate the pixel sum D within any rectangular region with just three integer operations:

$$\begin{aligned} D &= ii(4) - ii(2) - ii(3) + ii(1) \\ &= (A + B + C + D) - (A + B) - (A + C) + A \\ &= D \end{aligned} \tag{2.13}$$

The integral image works equally efficient for rectangles at any size and any location. In fact, the first step for training and running the classifier is always to convert the input raw images into integral images.

2.3.3 AdaBoost algorithm

In Viola-Jones face detector, AdaBoost is used both to select features and to train the weak classifiers. The weak learning algorithm is designed to select a Haar-like feature f_i and an optimal threshold v_i ,which best separate the positive and negative examples at current distribution D_i . A weak classifier $h_i(x)$ thus consists of a feature f_i , a threshold v_i and a parity p_i indicating the direction of the inequality sign:

$$h_i(x) = \begin{cases} 1 & if \ \ p_i f_i(x) < p_i v_i \\ -1 & otherwise \end{cases} \qquad (2.14)$$

Equation 2.14 indicates that the weak classifiers are decision trees that are only one level deep. The one level decision stump is allowed just one decision of the following form:" Is the value v of a particular feature f_i above or below some threshold v_i "; a "yes" indicates face and a "no" indicates no face.

AdaBoost then iteratively builds up a classifier as a weighted sum of these kinds of weak classifiers as illustrated in Table 2.1.

2.3.4 Cascaded classifier

A well-established face detection paradigm is to traverse a constant detector over a series of pyramid images in order to detect faces at various scales and locations. Observations reveal that an overwhelming majority of the sub-windows are non-faces

and most of the processing time is wasted on these background regions. However, a change in the mindset could lead to a reformulation of the detection problem into a cascaded rejection problem, meaning that the final detector could be composed of a cascade of rejecters. At each stage, a substantial portion of the sub-windows are rejected, and only sub-windows that pass through all previous stages are eligible for the next stage. With this design, more effort and computing power can be directed towards the later stages which concentrate on the harder regions, while keeping the early stages as simple as possible.

Viola-Jones face detector is a 32 layer cascade of classifiers which includes a total of 4297 features. The first classifier in the cascade is constructed using two features and rejects about 60% of non-faces. The next classifier has five features and rejects 80% of remaining non-faces. The next three layers are 20-feature classifier followed by two 50-feature classifier followed by five 100-feature classifier and then twenty 200-feature classifiers. Despite a large number of features involved in the cascade, they managed to run the detector at an average of 8 features per sub-window evaluated on the MIT+CMU test set, compared to a constant number of 37 features in a single layer classifier used by Papageorgiou et al [45]. It is the cascading design that particularly made the Viola-Jones detector the first real-time face detector.

2.4 Summary

This chapter reviews general 2D staircase detection methods at the beginning of the chapter. We also introduced Viola-Jones object detection framework. The framework incorporates several innovative features.

- It uses Haar-like input features which encode local brightness information.
- It introduces integral image technique which enables rapid computation of the values of rectangular regions and thus Haar-like features.
- It uses AdaBoost algorithm to select features and builds classifiers which are characterized by high detection and weak rejection.
- It organizes the strong classifiers created by AdaBoost algorithm into a cascaded structure which allows a substantial portion of the sub-windows in the input image to be rejected from the classification chain in the early stages. This design helps Viola-Jones face detector run at an unprecedented speed.

AdaBoost algorithm is the core of the Viola-Jones object detection framework. Thus, an insight into the algorithm is also given before the introduction of Viola-Jones framework.

Chapter 3 PCA-based Haar-like Features

3.1 Introduction

This chapter proposes a generalized framework for target-oriented Haar-like feature design. It tries to automate the process of summarizing the visual characteristics of target objects and goes a step beyond traditional perception-based Haar-like feature design. Within the framework, each feature designed, by nature, is an abstract representation of a principal component of positive samples. The features thus are termed “PCA-based Haar-like features”.

3.2 Target-oriented Haar-like features

As mentioned in Chapter 2, Haar-like features are good local contrast descriptors. The classical features introduced in Section 2.3.1 are applicable to objects beyond human faces. However, they are sometimes insufficient to express both the local and global characteristics unique to a particular class of objects. Training with such a degenerated feature pool will result in an unnecessarily lengthy and ponderous cascade which will inevitably prolong the detector’s processing time. In order to create simple classifiers, target-oriented Haar-like features are proposed [48], [49] to supplement the classical ones.

Islam et. al. [48] supplement two Haar-like feature bases shown in Figure 3.1 for ear detection. They claim these features are designed for curves of the helix and anti-helix. In [49], new feature bases shown in Figure 3.2 are proposed for eye detection. These features are specifically designed to capture various local characteristics of eye region.

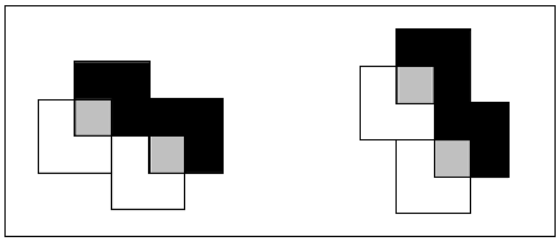

Figure 3.1: Target-oriented Haar-like features for ear detection

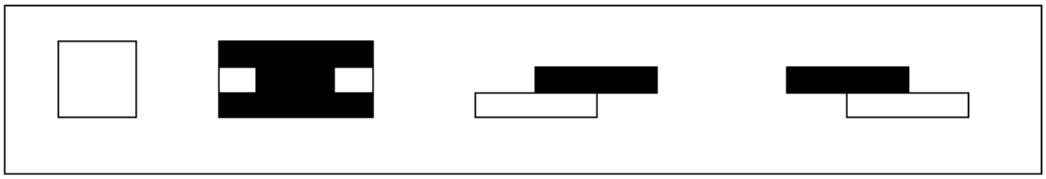

Figure 3.2: Target-oriented Haar-like features for eye detection

These target-oriented features overcome the symmetry limitation of classical Haar-like features and are capable of representing the targets with fewer features than the conventional ones. However, the features are designed based on the authors' perception of the target objects. Neither of the above methods provides direct rationale behind their Haar-like features. Following sections will introduce a generalized framework for systematic Haar-like feature design, using staircase as an example.

3.3 Positive sample collection

We need a set of positive samples to create the PCA-images. Hence, it is necessary to prepare the positive samples first. Unlike human faces, staircases are manmade objects. Their visual characteristics are highly dependent on the designers. They may have different colors, made of different materials, decorated with different patterns and consisting of different numbers of steps. Even the same staircase exhibits different visual effects under varying illumination and ambience. On top of that, we may not be able to differentiate a clear-cut boundary of the staircase from a given image, since sometimes staircases are only partially visible through the lens. To make the learning-based method practically possible on manmade objects with large intra-class variability, such as staircases, uncertainties must be weakened while stressing the commonalities among the positive samples. In line with this notion, two constraints are imposed on the preparation of a positive sample set.

The first constraint restricts the selection of candidate staircases. Only commonly-seen straight-line staircases with enough illumination are collected in the raw image collection stage. Staircases with rarely-seen materials and decorations are excluded from the training data set because their inclusion does not provide commensurate benefits, but generally degrades the performance of the final detector. By imposing this constraint, individual uncertainties are weakened.

The second constraint applies to the selection of region(s) of interest (ROI) in a staircase image. The ROI is selected as a square region that covers three successive steps. A step

consists of a tread and a riser plane. Attention must be paid to the treads since the distant treads are more likely to be occluded by the risers. As a general rule of thumb, distant steps should be avoided. Thus, our definition of a staircase object or a positive sample is a square region contains three successive and completely visible steps. Less or more than three steps are considered negative and should not be detected. This

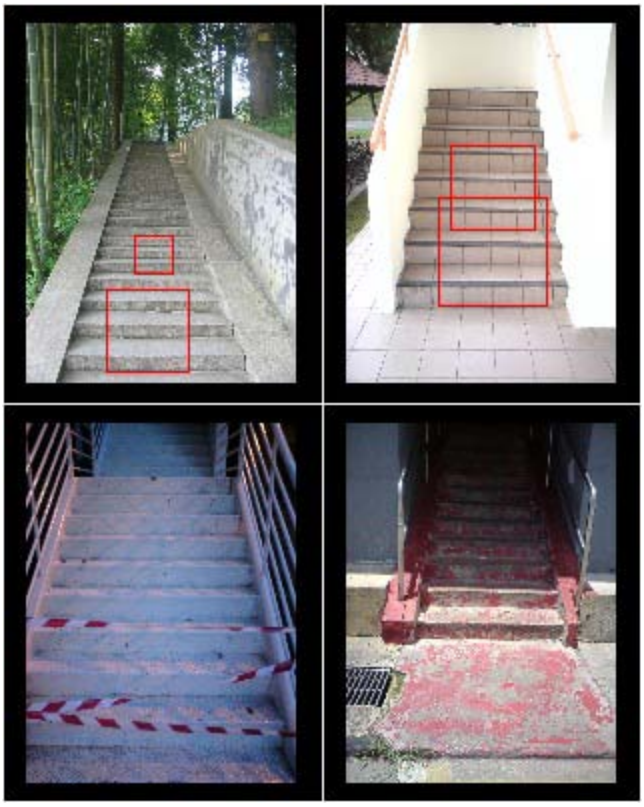

Figure 3.3: Positive samples collection

definition of staircase restricts the searching domain of the detector. On the other hand, it makes the detector sturdy and unaffected by situations such as partial visibility and unknown number of steps, since we are only interested in part of the staircase. With this constraint, positive samples now look more akin to each other with increased level

of commonalities in their structures. An organized state is achieved from an otherwise chaotic state.

Examples of accepted and rejected staircases are shown in Figure 3.3. Selected ROIs are overlaid on accepted staircases; these square regions are instances of the positive data set. The last point worth noting is that you can crop multiple regions from one staircase image as long as they do not closely duplicate each other. This is useful when you don't have enough samples for statistical analysis. In our training set, we cropped 500 positive samples from 380 staircase images.

3.4 Continuous PCA images

PCA is an important computation in statistical analysis and is widely used for data analysis in many fields. PCA transforms a number of possibly correlated variables into a smaller number of uncorrelated variables called principal components. The principal component of the largest eigenvector is the most informative axis containing the largest spread of the samples. The second most important direction corresponds to the one associated with the second largest eigenvalue, etc. One important application of PCA is data compression.

With the availability of the positive data set, we can use PCA to find the principal components, along which the positive samples are scattered. The principal components of the data can be obtained as follows:

First, we have a set of 500 24x24 staircase images. They are straightened out row by row into 576 dimensional column vectors. So the training set becomes $\{s_1, s_2, \cdots, s_{500}\}$. Then the average staircase is found: $s_{ave} = \frac{1}{500}\sum_{i=1}^{500} s_i$. Next, the average staircase is subtracted from every training image so that the resulting staircase data set has zero mean and we obtain a 576x500 matrix: $X = [s_1 - s_{ave}, s_2 - s_{ave}, \cdots, s_{500} - s_{ave}]$. Next, we compute the covariance of X : $C = \frac{1}{500} XX^T$. Since we only need a few principal components (in our experiment this number is 5), we only extract the first 5 eigenvectors corresponding to the 5 largest eigenvalues of the covariance matrix C.

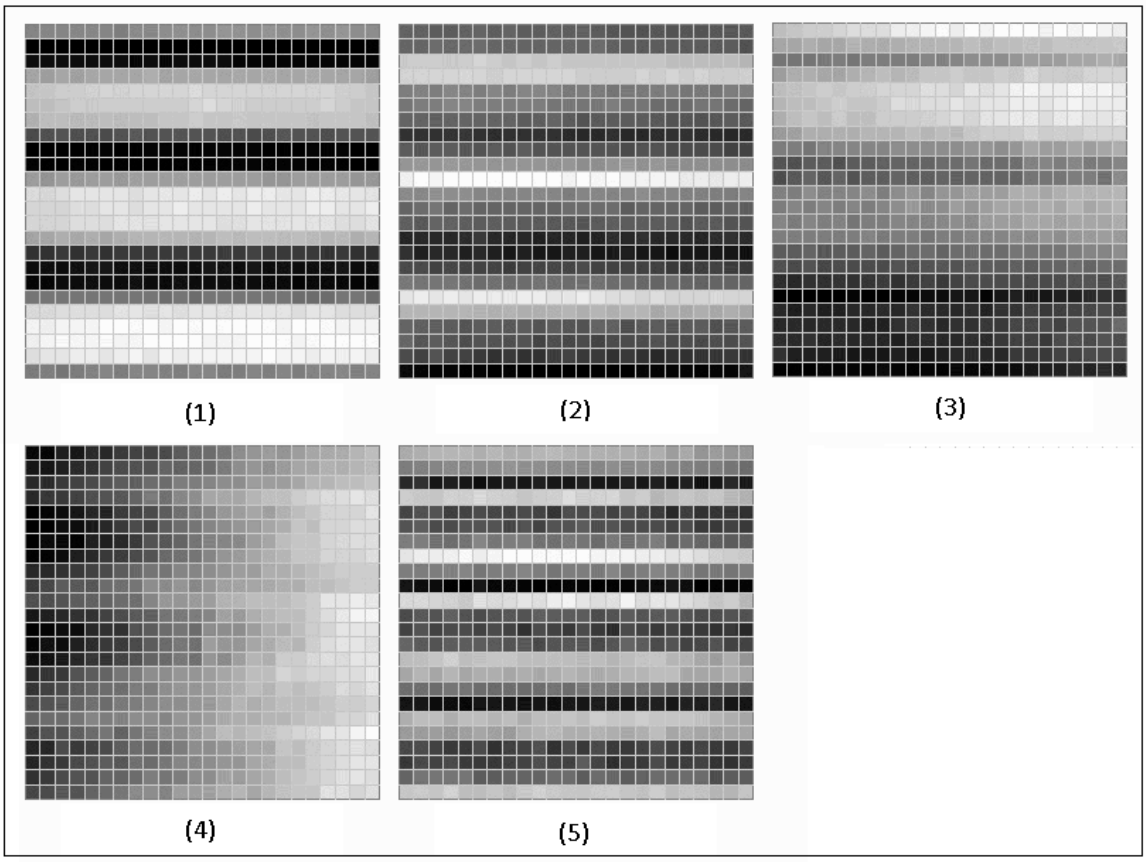

Figure 3.4: Five continuous PCA images

The 5 principal components are finally converted back into 24x24 images shown in Figure 3.4. These images are continuous in terms of real-valued pixel intensities. Note that the PCA images herein and hereafter are all normalized for display.

3.5 Discrete PCA images

A continuous PCA image can be discretized by converting it from continuous space into an equivalent discrete space. In a discrete space, the continuous pixel values are categorized into n bins. If n = 2, a binary thresholded image is obtained as shown in Figure 3.6. The discretized PCA images can be used as guidelines for Haar-like feature design.

3.6 Designing PCA-based Haar-like features

The PCA-based Haar-like features are a small set of user designed target-oriented Haar-like features that are created based on the discretized PCA images of the positive data set. Figure 3.5 illustrates the flow diagram for the creation of these features.

The design of PCA-based Haar-like features is indeed a process of converting the discrete PCA image from an n-level image into a 3-level image, with ternary representation: $\{-1,+1,0\,(don't\ care)\}$. In discrete PCA images, pixels falling into the same bin often cluster to form connected gray level regions. The design first involves

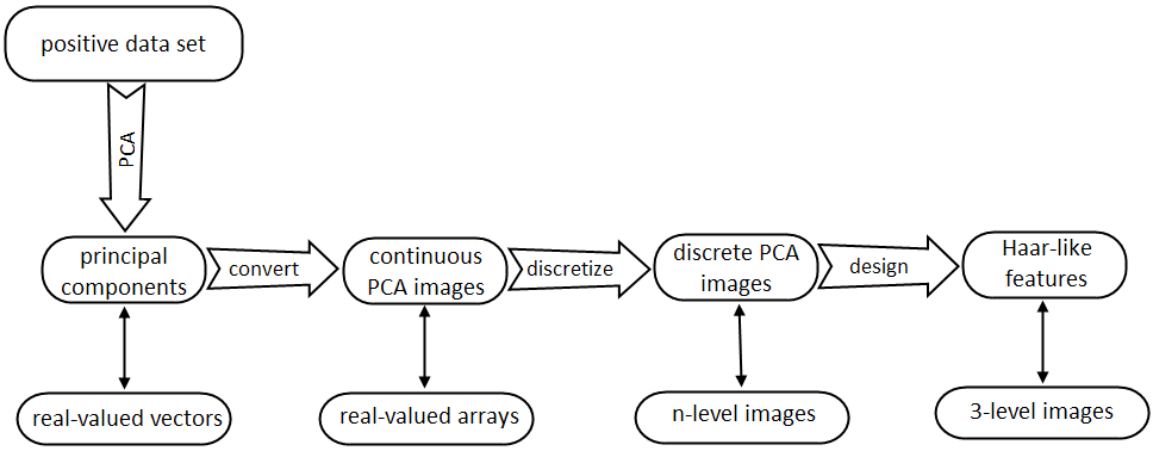

Figure 3.5: Flow diagram of PCA-based Haar-like feature design

the extraction of regions with minimal and maximal gray levels. The regions are then approximated with rectangles. It is difficult to define a clear-cut boundary for the rectangles. Hence, there is no ultimate solution in the process of PCA-based Haar-like feature design. Generally, the designing process should try to follow the principle of generating correct, accurate, symmetric and easy to program features.

We discretized the 5 continuous PCA-images with different number of bins (n), each time a set of 5 Haar-like features are produced. Figure 3.6 and 3.7 show the discrete PCA images and their corresponding Haar-like features with n = 2 and n = 4 respectively. Figure 3.8 lists all the 25 PCA-based Haar-like features fed into the training feature pool. For the features in each row, n= 2, 4, 6, 10, 20 respectively.

Classical Haar-like features are local and unable to capture global characteristics of target objects. On the contrary, features listed in Figure 3.8 are all global features. Hence, the PCA-based Haar-like features are suitable global characteristic descriptors.

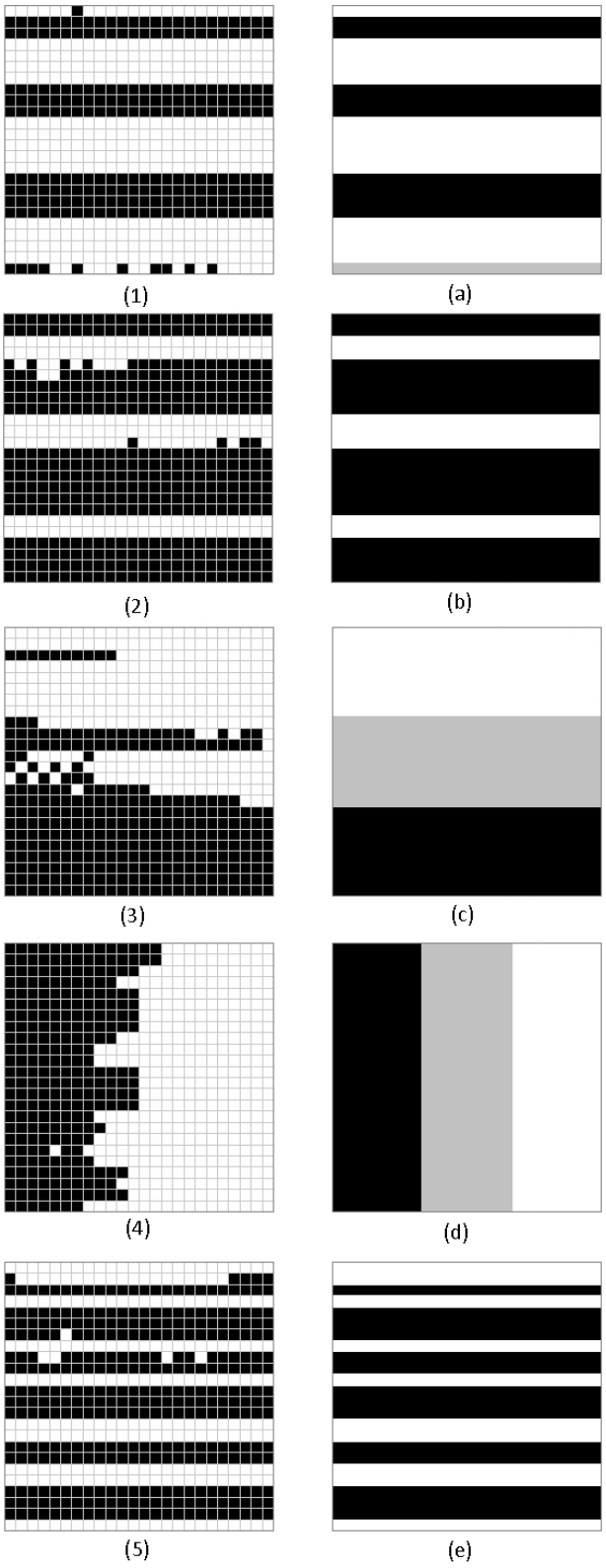

Figure 3.6: Binary discretized PCA images and their Haar-like feature counterparts

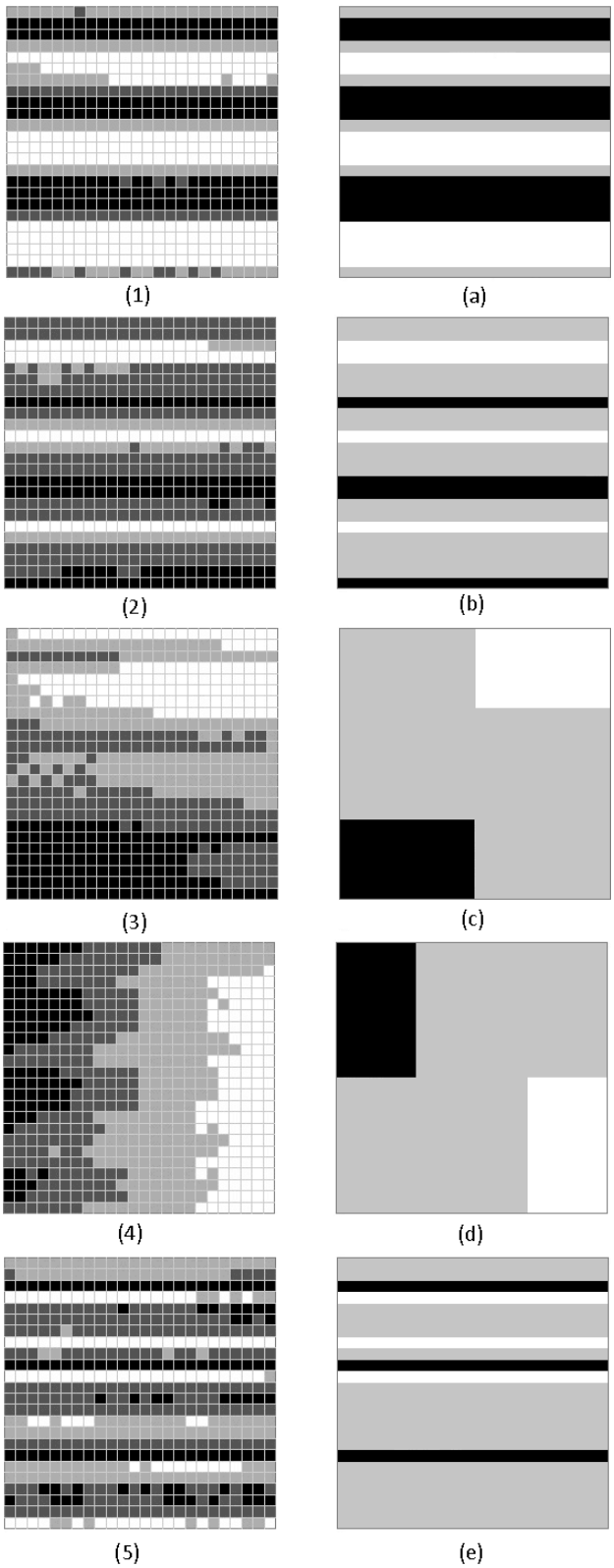

Figure 3.7: Quaternary discretized PCA images and their Haar-like feature counterparts

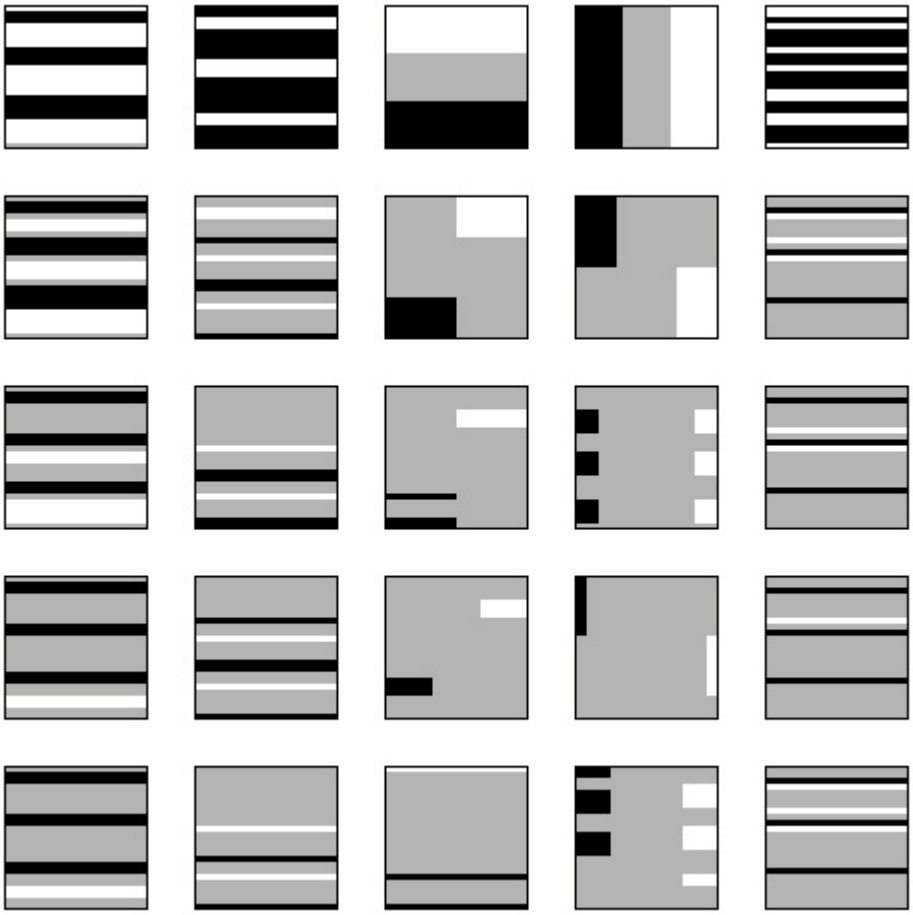

Figure 3.8: A table of all 25 PCA-based Haar-like features used in training

In [41], two global statistical features have been used as the first 2 layers in the cascade. These layers can filter out background regions at the global level, which greatly mitigate the burden on subsequent Haar-based analysis. The PCA-based features created here resemble the global statistical features [41]. However, we will let the data to determine whether and when to use these features, instead of hard coding them into the beginning of the cascade.

The target-oriented Haar-like features proposed in [48] and [49] are feature bases. When scaling and shifting the bases across the sample image, they generate tens of thousands Haar-like feature instances. Supplementing all these instances into the classical feature pool will drastically prolong the training. On the other hand, PCA-based Haar-like features have predefined scales and locations in the base image. The 25 PCA-based Haar-like features displayed in Figure 3.8 almost have no impact on the size of the feature pool, but greatly increase the Haar-like feature diversities.

The principal components can scatter the data, so do the PCA-based Haar-like features. This argument will soon be reinforced by experimental data presented in Section 4.3.3. The scattering of positive samples also makes these features highly compatible with the real AdaBoost algorithm (Section 4.2).

Last but not least, weight for each rectangle in the feature must be carefully adjusted in order to achieve an equal weighted-area balance between black and white regions.

3.7 Summary

This chapter proposes a generalized target-oriented Haar-like feature design framework. A small set of Haar-like feature instances are created systematically with the aid of discrete PCA images of the positive data set, instead of forging the feature bases based on personal perception on the target object.

The PCA-based Haar-like features extend the features from local to global domain and are effective global pattern descriptors. The PCA-based Haar-like features have their roots on principal components of positive data set. Hence, the positive samples are believed to scatter along these features. The advantages of PCA-based Haar-like features will soon emerge in the next chapter.

Chapter 4 2D Staircase Detection

4.1 Introduction

This chapter introduces a 2D staircase detection algorithm based on Viola-Jones object detection framework. The algorithm follows the infrastructure of the original framework with some improvements. PCA-based Haar-like features are added to the feature pool. Real AdaBoost algorithm is used to replace the original Discrete AdaBoost. The scanning scheme, integration scheme and evaluation metrics have been modified for compatibility with staircase detection. At the end of this chapter, experimental data is given alongside with detailed analysis. Lastly, a comparative study on the performance of the proposed algorithm with one of the start-of-the art line-based staircase detection methods [14] is presented.

4.2 Real AdaBoost

Conventional AdaBoost algorithm makes Boolean predictions: $\{-1,+1\}$, as shown in Equation 2.14, which is often known as Discrete AdaBoost. An improved version, real AdaBoost algorithm [50], however maps the sample space X to a real-valued space $\Re$. Real AdaBoost assigns confidence rather than prediction to an input sample x. This project uses real AdaBoost to replace the original Discrete AdaBoost in the training of the final detector. Details of the algorithm are shown in Table 4.1.

- Given: $(x_1, y_1), \cdots, (x_m, y_m)$ where $(x_i, y_i) \in X \times \{-1, +1\}$
- Initialize $D_1(i) = 1/m$.
- For $t = 1, \cdots, T$
  - For each weak classifier h do:
    - Partition X into several disjoint blocks $X_1, \cdots, X_n$.
    - Calculate

$$W_l^j = P(x_i \in X_j, y_i \in l) = \sum_{i::\ x_i \in X_j \wedge y_i = l} D_t(i) \tag{4.1}$$

      where $l = \pm 1$.

    - Set the output of h on each block X_j as

$$\forall x \in X_j, h(x) = \frac{1}{2} \ln\left(\frac{W_{+1}^j + \varepsilon}{W_{-1}^j + \varepsilon}\right) \tag{4.2}$$

      where ε is a small positive constant.

    - Calculate the normalization factor

$$Z = 2\sum_j \sqrt{W_{+1}^j W_{-1}^j} \tag{4.3}$$

  - Select h_t minimizing Z, ie.

$$Z_t = \min_h Z$$
$$h_t = \arg\min_h Z \tag{4.4}$$

  - Update the weights: $D_{t+1}(i) = D_t(i) \exp[-y_i h_t(x_i)]$, and normalize it into a distribution.
- The final strong classifier H is

$$H(x) = sign\left[\sum_{t=1}^{T} h_t(x) - b\right] \tag{4.5}$$

$$Conf_H(x) = \left|\sum_{t=1}^{T} h_t(x) - b\right| \tag{4.6}$$

where b is a threshold whose default is zero. Equation 3.6 gives the confidence of H

Table 4.1: Real AdaBoost algorithm

In line with the real AdaBoost partition rule, the weak learner divides the training set into n subsets for each feature f according to their values of Haar-like features $f(x)$. Examples falling into the same subset will be assigned an equal confidence value, which depends on the sample distribution within the subset as shown in Equation 4.2. A cumulative impurity factor Z , given in Equation 4.3, is computed for each feature and the feature corresponding to minimum Z is selected in the current round of feature selection.

Essentially, real AdaBoost deals with a confidence-rated weak classifier instead of a simple Boolean prediction as illustrated in Figure 4.1.

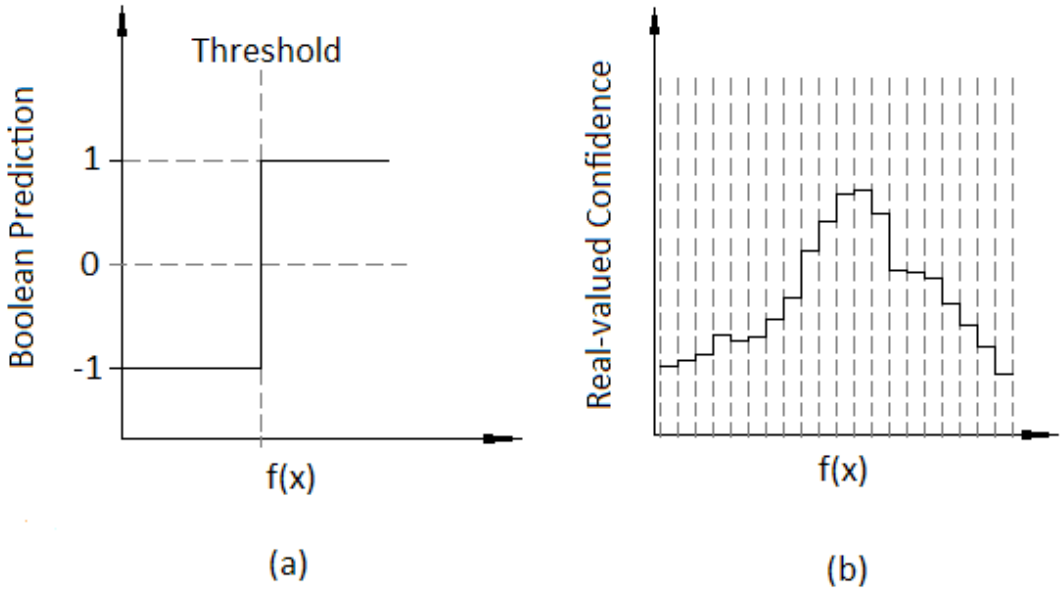

Figure 4.1: Discrete threshold and real-valued confidence

4.3 Training

Training is the most time-consuming part in Viola-Jones object detection framework, generally taking time on the order of weeks or even months. The following sections focus on the training algorithm that the project adopts and an analytical review is given to the first layer of the cascaded classifier.

4.3.1 Inter-stage sample selection

For training of binary classifiers, we need positive and negative samples. We have no direct means to obtain a complete staircase data set, except collecting samples by our own effort. Most of the positive samples are acquired from a 3.2-megapixel Sony Cyber-shot cell phone camera. We cropped 500 positive samples from 380 staircase images. Negative samples are sub-regions from 9,000 arbitrary images that do not contain staircases.

All positive samples are first converted to gray scale and resized to a 24x24 base resolution, with the variance normalized to minimize the effect of different lighting conditions.

For the first stage, all 500 positive samples and 1,500 negative samples are fed into training. Samples taken by subsequent stages must pass through all previous layers. Rejection may progressively slim down the positive data set over stages. However, the number of negative samples absorbed by each stage must maintain a proportion of 3 times that of the positive samples fed into that stage.

Positive samples are extracted from prescribed ROIs while negative samples can be selected from random regions at random scales. To broaden the negative data set in order to cover diverse background conditions, we let each stage recycle through all the 9,000 negative images. Each stage starts reading in negative samples from the last image used by the previous stage. A new cycle restarts from the first image once it reaches the end of the data set. Only one region is cropped from each image in each cycle, the cropped region is tested by the partial cascade that is currently available. Validated regions are fed into the negative data set of the stage. The process continues until the required number of negative samples is achieved. This mechanism makes the negatives change on a per-stage basis, while the "core interest" of the 500 positive samples remains unchanged throughout the training process.

4.3.2 Training a cascade of classifiers

Training is driven by a set of detection and performance goals. Without loss of generality, our detector should achieve at least 80% detection rate and an extremely low false positive rate on the order of 10^{-9} .

As mentioned in Chapter 2, each stage trains a strong classifier using the AdaBoost algorithm. Examples used to train subsequent stages must survive the test of all previous stages. The overall detection rate of the cascade can be estimated by multiplying the detection rates of all its constituent stages as:

$$D = \prod_{i=1}^{k} d_i \qquad (4.7)$$

Similarly, the overall false positive rate can be estimated by:

$$F = \prod_{i=1}^{k} f_i \qquad (4.8)$$

where k is the number of stages, and d_i, f_i are the detection rate and false positive rate of the ith stage. This implies that the users can control the overall performance by specifying the stage goals. In the experiment, we expect a 30-stage classifier with a 0.995 detection rate and a 0.5 false positive rate for each stage. The final detector thus achieves at least $0.995^{30} \approx 0.86$ overall detection rate and $0.5^{30} \approx 9.3 \times 10^{-10}$ false positive rate on the training examples.

The above design of stage classifiers is straightforward and simple to code. It is widely used and is also adopted by our program. However, it may not be the most efficient in terms of minimizing the number of features used in the cascade. Given the required overall goals, the optimization process needs to take into consideration the number of stages, the number of features for each stage and the stage threshold, etc.

When training a stage classifier, the AdaBoost algorithm continuously adds new features into the stage until the required stage goals are met. New layers are added to the cascade if the overall goals are not met yet. Our program was terminated at the 29th stage since the desired performance is achieved earlier than expected (30 stages). The training process is illustrated in Table 4.2.

- User set minimum stage detection rate d, maximum stage false positive rate f and the number of expected stages N.
- Calculate overall false positive rate $F_{target} = f^N$.
- $i = 0$
- While $F_i > F_{target}$ and $i < N$
  - Initialize weights $w_{i,j} = \frac{1}{2m_i}, \frac{1}{2l_i}$ for positives and negatives, respectively, where m_i, l_i are the number of positives and negatives fed into stage i.
  - While $f_i > f$
    - Use real AdaBoost to train a weak classifier and add it to the stage classifier.
    - Update the weights.
    - Evaluate current partial stage classifier on the training set and sort the results to obtain 2 sequences: P_{eval}, N_{eval}
    - Determine stage threshold t_i: $N_{eval}(f) \le t_i \le P_{eval}(d)$, where $N_{eval}(f)$ is the minimum threshold to guarantee the maximum accepted stage false positive rate f, $P_{eval}(d)$ is the maximum threshold to maintain the minimum detection rate d.
      - If $[N_{eval}(f), P_{eval}(d)]$ exists, decrease t_i as much as possible ($t_i = N_{eval}(f)$) to maximize the detection rate.
      - Else set $t_i = P_{eval}(d)$
  - Add the stage classifier into the cascade $(i \leftarrow i+1)$

Table 4.2: 2D staircase detector training algorithm

4.3.3 An insight into the first layer

Our final detector is a 29-layer cascade of classifiers which includes a total of 655 features. The PCA-based Haar-like features play an asymmetrically important role in the cascade in view of their small proportion in the feature pool. These features gradually lose their edges after the first 10 stages. This is meaningful as global features gradually lose competence in separating on-stage samples when training progresses.

An insight into the first layer is elaborated as follows.

The first layer selects 3 features. They are overlaid on a typical staircase image and shown in Figure 4.2.

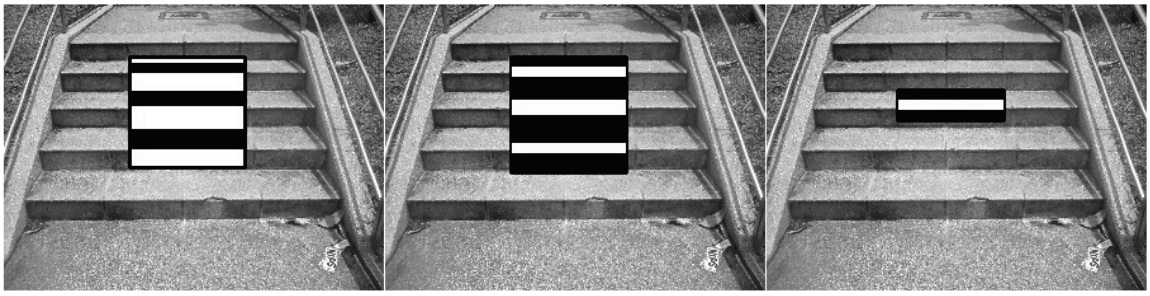

Figure 4.2: Selected features in layer 1

The first 2 features selected happen to be the first 2 PCA-based Haar-like features. They capture the global characteristics of the selected ROIs. It can be easily understood why these 2 features are first selected by the real AdaBoost algorithm. The distribution of the on-stage training samples over these 2 features is shown in Figure 4.3. Positive samples are shown in red and the negatives in blue. The abscissa axis represents the first PCA-based Haar-like feature. The distribution map shows the positives are scattered while the negatives, though in a much larger quantity, are compressed to the

central region of the map. This distribution is consistent with our prediction in the previous chapter.

The real AdaBoost partitions the samples according to their Haar values. It can easily be seen from the map that the impurity factor of lateral partitions are simply 0. That is

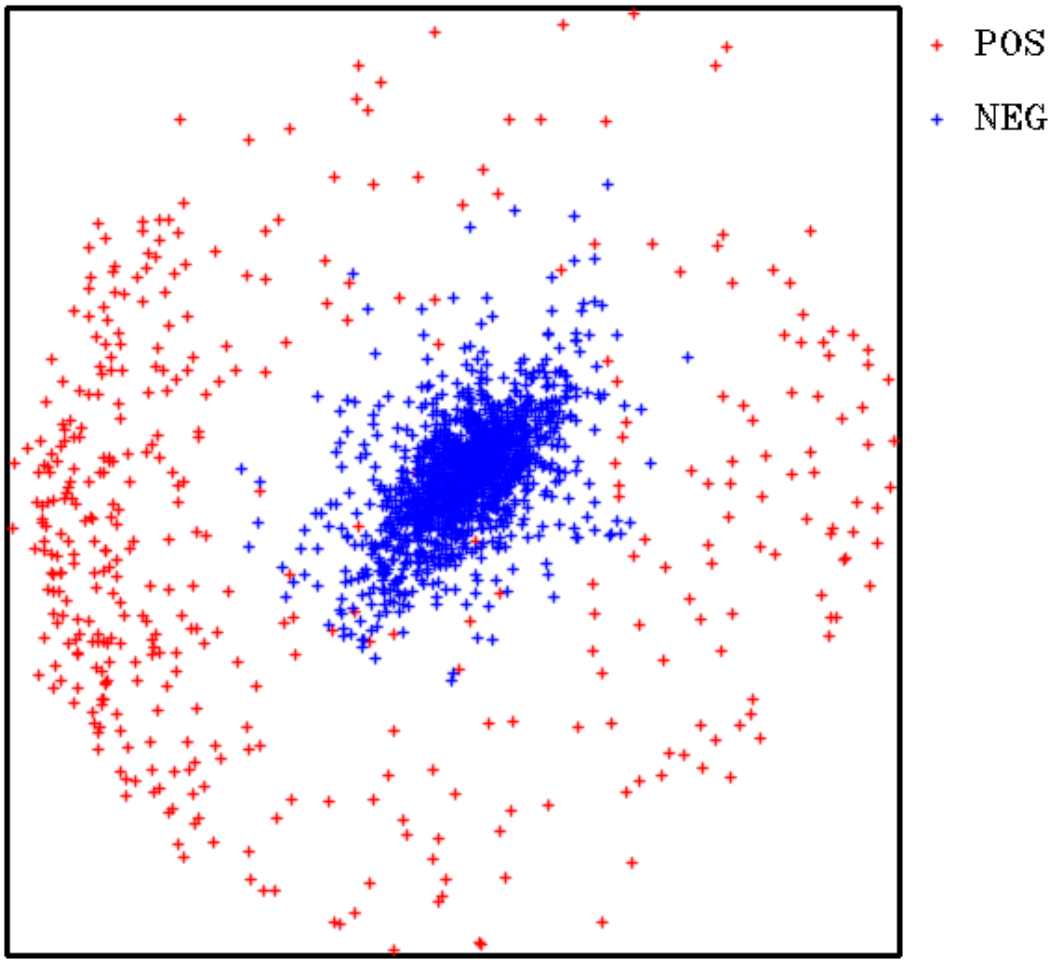

Figure 4.3: Sample distribution over the first 2 PCA-base Haar-like features

why AdaBoost selects these features first. Samples distributed like Figure 4.3 is more suitable for partition-based processing instead of simple binary thresholding. It justifies

why real AdaBoost is generally superior to the Discrete AdaBoost. The first layer rejects up to 90% of negative samples while keeping 100% of the positives. It shows the global PCA-based Haar-like features are very efficient at rejecting false windows at the very beginning of scanning. Scanning will be discussed in detail in next section.

4.4 Running the detector

After training completes, we can run the cascade detector on any images to search for staircases. The final detector scans across the image at multiple scales and locations. Scaling is achieved on the detector rather than on the image.

Unlike human faces, staircases usually take a large region on image surfaces. It is unnecessary to start scanning from the base resolution of 24x24 pixels on a 1,280x960 pixels image. Similarly, you can stop scanning earlier before reaching the maximum size. For that reason, the first two scan parameters $s_{\min}$ and $s_{\max}$, representing the starting and terminating scan window size, are defined.

The third parameter Δw is the width increment between consecutive scans, each of which materializes a scaled detector. Scaling is achieved by a constant increment as opposed to a constant scaling factor [26], since the latter down samples larger regions, where more targets may hide behind.

The detector is also scanned across locations. It is shifted some pixels to the right in the X direction $(Xshift)$ and then vertically down in Y direction $(Yshift)$ to complete one scan.

Up to now, we have introduced all the 5 scan parameters. A schematic explanation of these parameters is given in Figure 4.4.

A major difference of the proposed algorithm with [26] is that all the scan parameters are defined over the input image size. Thus, the values change when the image size changes. On the other hand, they are inter-scan constants. In short, [26] changes the parameters between scans, while the proposed scanning scheme changes the parameters with images.

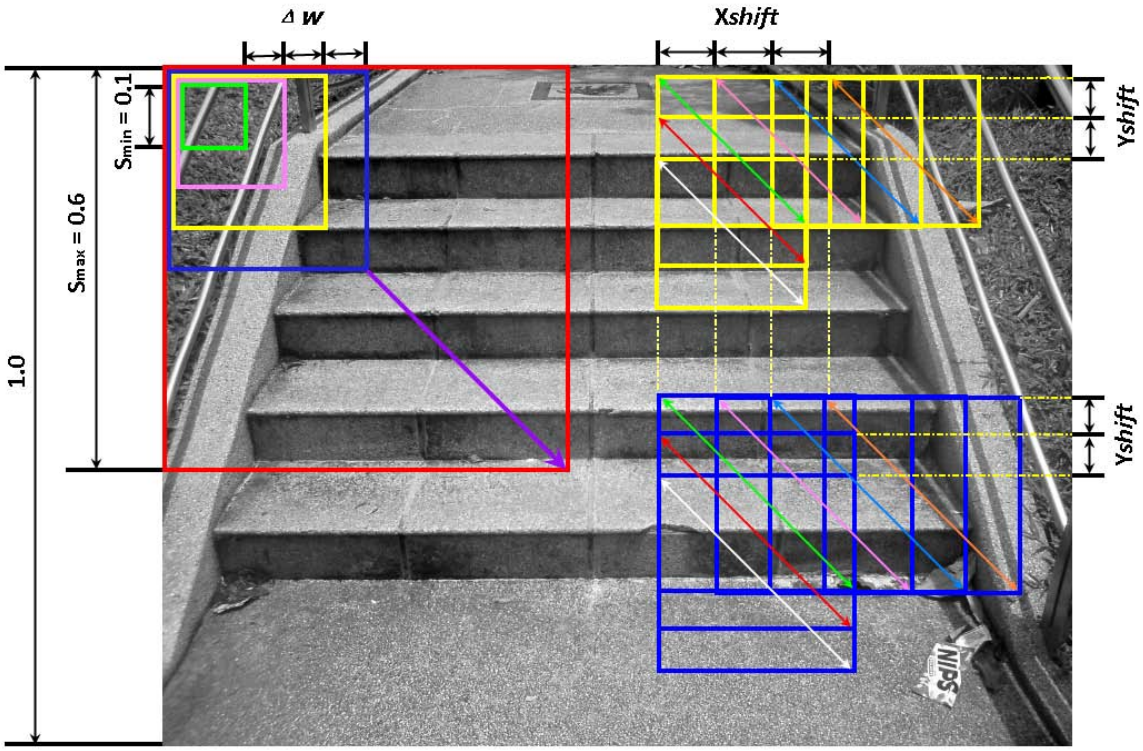

Figure 4.4 A schematic representation of scan parameters

This thesis experiments with two sets of scan parameters as summarized in Table 4.3. The absolute values are calculated on a sample size of 1,280x960 pixels. The reason for a sparse sampling in horizontal direction is that the detector has many chances to find targets if they really exist since we are dealing with frontal-looking staircases. Maintaining a much finer sampling rate in vertical direction is to avoid missing targets and safeguard detection accuracy. With this scanning scheme, the final detector will scan a constant number of sub-windows and therefore possesses certain degree of speed invariance for various image sizes.

Parameters	Scenario 1		Scenario 2	
	Factor Value	Absolute Value	Factor Value	Absolute Value
$s_{\min}$	0.100	96x96	0.100	96x96
$s_{\max}$	0.600	576x576	0.600	576x576
Δw	0.020	19	0.040	38
$Xshift$	0.020	26	0.040	40
$Yshift$	0.005	5	0.010	10

Table 4.3: Experimental scan parameters

4.5 Integrating multiple detections

Since our detector intends to search for square ROIs which cover 3 consecutive steps of a staircase, we could expect unknown number of detections within a single image. These initial detections are called primitive detections and must be combined to produce one final bounding box, indicating the meaningful staircase boundary for each staircase in the scene. The integration scheme comprises two successive steps with the second step performed on the results of the first.

Step 1 integrates primitive detections mapped from the same set of 3 consecutive steps. Deviations must be contained in order to group 2 primitive detections together, namely, the variations between the top edges and the heights of the 2 windows must be below 5% (ΔH) of the current window size. Primitive detections satisfying these constraints are termed homogeneous detections. Groups with less than a minimum number ($G_{\min} = 5$) of homogeneous detections are discarded as noises. This helps to further reduce the false alarms. It is almost impossible to find 5 false alarms around the same rows, with almost the same size in the background region. The only possibility is that individual false alarm might be incorrectly grouped into a positive sub-group, and finally has some negative impact on the determination of the final bounding box as the case shown in Figure 4.8.

Each group is then reduced to a sub-bounding box, which constitutes our secondary detections. The left and right boundaries of the sub-bounding box are the leftmost and

rightmost edges of the windows in that group, while top and bottom edges are taken as the average of its child windows respectively.

Although we mentioned above the false alarm is almost impossible in the background region, it is still quite common that wrong detections occur on foreground regions. These detections span less or more than 3 steps or have relatively large migrations from expected positions. They happened within the “right” region and may not negatively affect the integration results significantly. However, after first step of integration, these spurious groups can be removed by checking their heights. Inliers should have their heights progressively increasing from top downwards. Frequently, this reduction leads to a shrink of the final reported area. Experiments also point out this process often degrades the performance of the detector as rejected areas often come from the foreground region. In view of this, the reduction is not implemented in the experiment. Step 2 simply finds a bounding box of the remaining secondary detections.

4.6 Performance

4.6.1 Setting up test environment

Since no online resources are available, we use a relatively small test set which contains 55 staircase images to evaluate the performance of our detector. Each image contains only one staircase instance. These images are acquired using the same camera for collecting positive training samples, but are not used for training. The image sizes vary from 1,280x960 pixels to 2,048x1, 536 pixels.

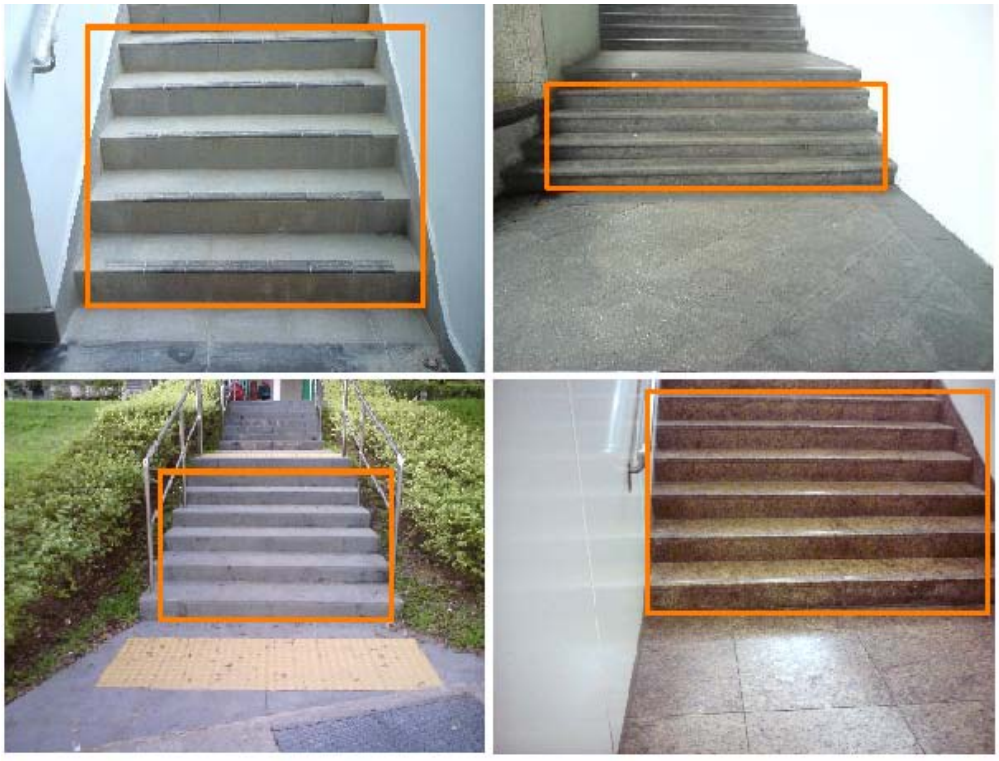

Figure 4.5: Labeling for performance evaluation

To facilitate the performance evaluation, all positive regions are pre-labeled as shown in Figure 4.5. For face detection, detection is qualified valid only if it intersects with one labeled face. Otherwise, it is classified as a false detection. This Boolean classification is too simple to be applied to evaluate the staircase detector and the parameter environment that it runs on. The reason is that staircases generally occupy a larger region than faces and modifying test parameters may displace the output window but not affect the Boolean classification result. Hence, Boolean classification is insufficient to measure the performance of a test environment. This insufficiency leads to a new evaluation metric. The new scheme measures the accuracy of detection by assigning a confidence value to it as illustrated in Figure 4.6 and Equation 4.10.

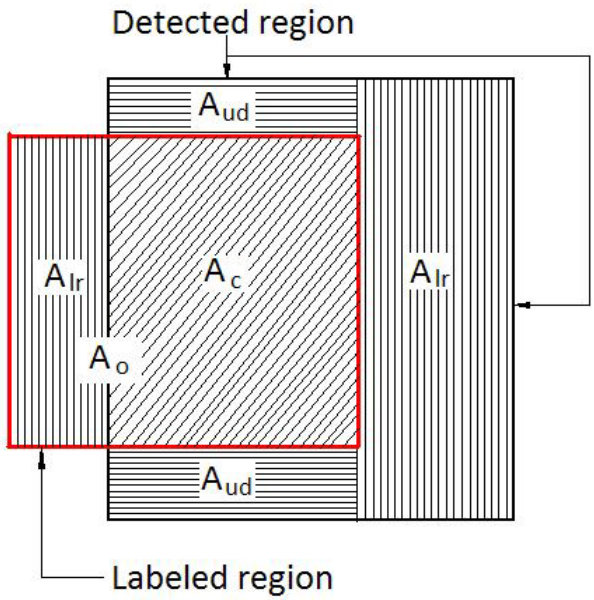

Figure 4.6: Confidence-rated evaluation metrics

$$conf_{eval} = \frac{A_c}{0.5A_o + 0.5A_c + k_1\left(e^{A_{ud}/A_o}\right)A_{ud} + k_2\left(e^{A_{lr}/A_o}\right)A_{lr}} \quad (4.10)$$

Where A_o is the labeled region area, A_c is the common area between the labeled and the detected region, A_{ud} and A_{lr} are the misaligned areas in the longitudinal and latitudinal directions, respectively as shown in Figure 4.6. Adjust k_1 and k_2 to tune penalties for misalignment in two directions. For most staircase images, it is easier to tell the left and right boundaries, but difficult in the longitudinal direction. Besides, experiments show that most of longitudinal false detections reside in the "right" region (see Section 4.5). Hence, latitudinal misalignments are more strictly prohibited. On the other hand, a certain degree of tolerance is given to longitudinal misalignments. A suggested setting is to set k_2 twice of k_1 and k_1 not less than 0.5. Our experimental setting is: $k_1 = 0.6, k_2 = 1.2$. The middle term $e^{\frac{A_{ud}}{A_o}}$ ($e^{\frac{A_{lr}}{A_o}}$) show that the penalty increases exponentially as the misaligned area increases. This is to safeguard detections at close neighborhood of the labeled region, while large discrepancies between the detected and expected boundaries (areas) will cause penalty to increase exponentially.

A complete summary of the parameters are listed in Table 4.4.

Parameter Type	Parameters	Scenario 1	Scenario 2
scan	$s_{\min}$	0.100	0.100
scan	$s_{\max}$	0.600	0.600
scan	Δw	0.020	0.040
scan	$Xshift$	0.020	0.040
scan	$Yshift$	0.005	0.010
integrate	ΔH	5%	5%
integrate	$G_{\min}$	5	5
evaluate	k_1	0.600	0.600
evaluate	k_2	1.200	1.200

Table 4.4: Summary of test parameter settings

4.6.2 Showing detection results

This section presents some detection results of the 2 test environments. Figures have a uniform layout of 2 rows and 3 columns. The first row corresponds to test environment 1, and the second row corresponds to test environment 2. The first column shows the primitive detections, while the second column displays the secondary detections after step 1 integration. The last column gives the final detection results.

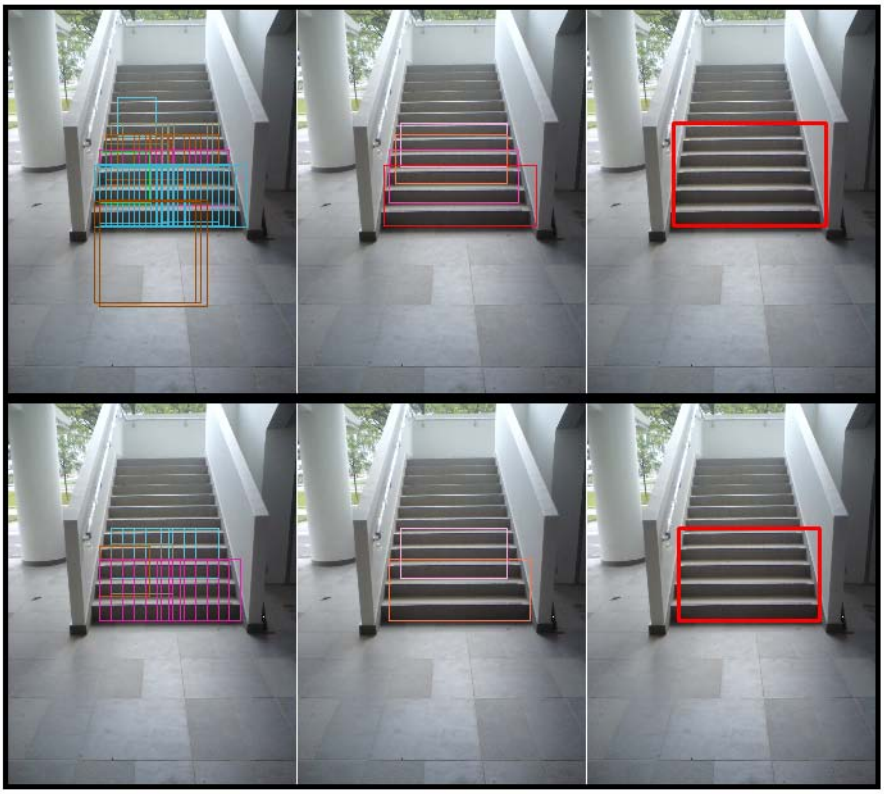

Figure 4.7 staircase detection case 1

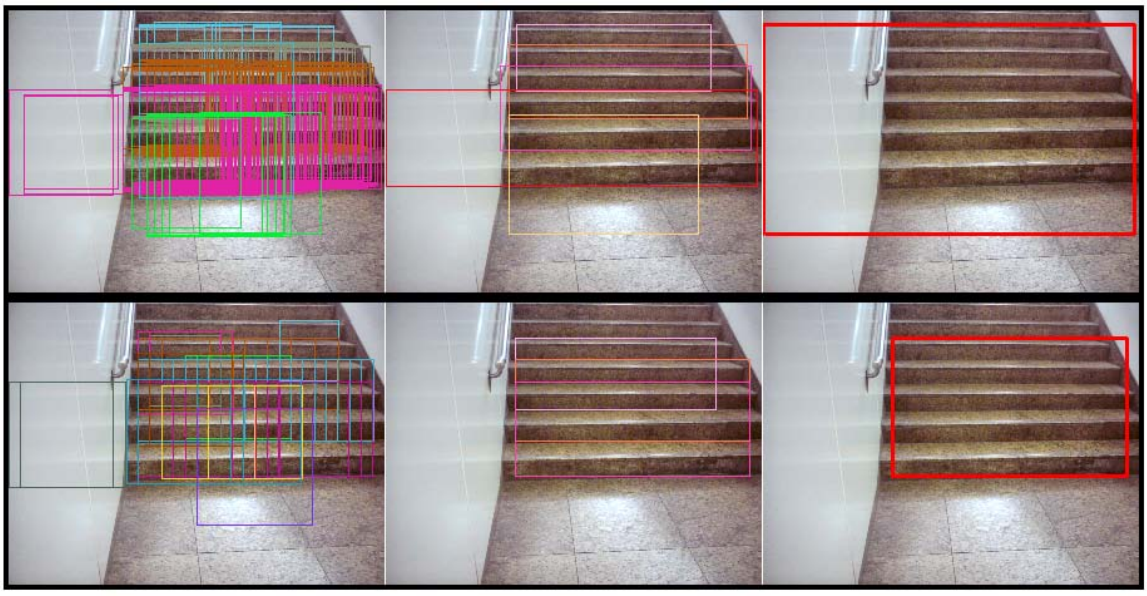

Figure 4.8 Staircase detection case 2

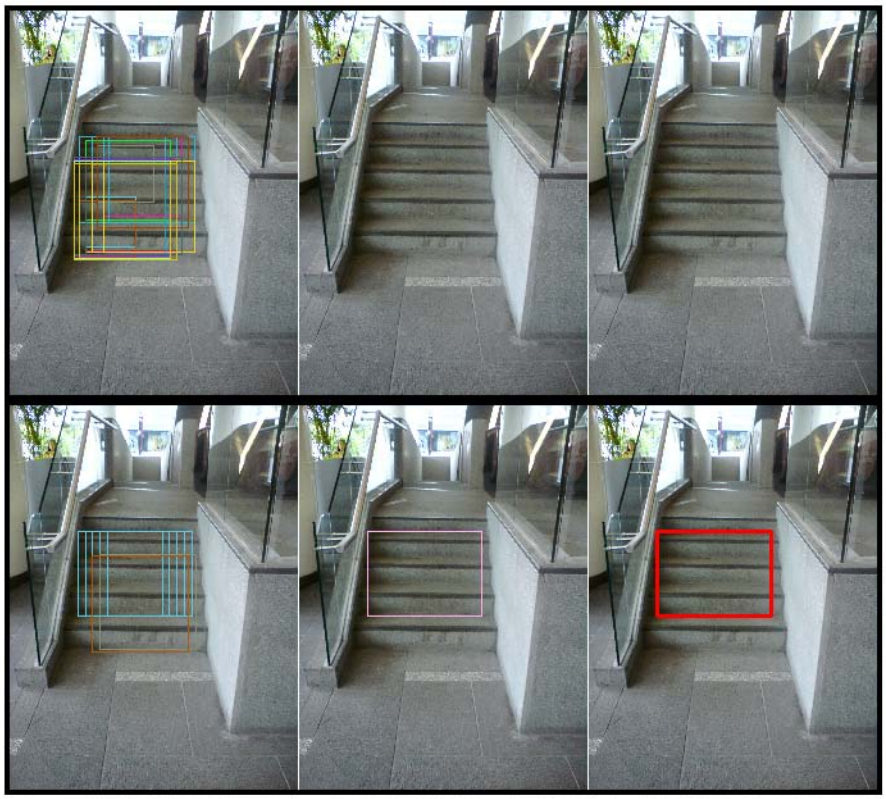

Figure 4.9 Staircase detection case 3

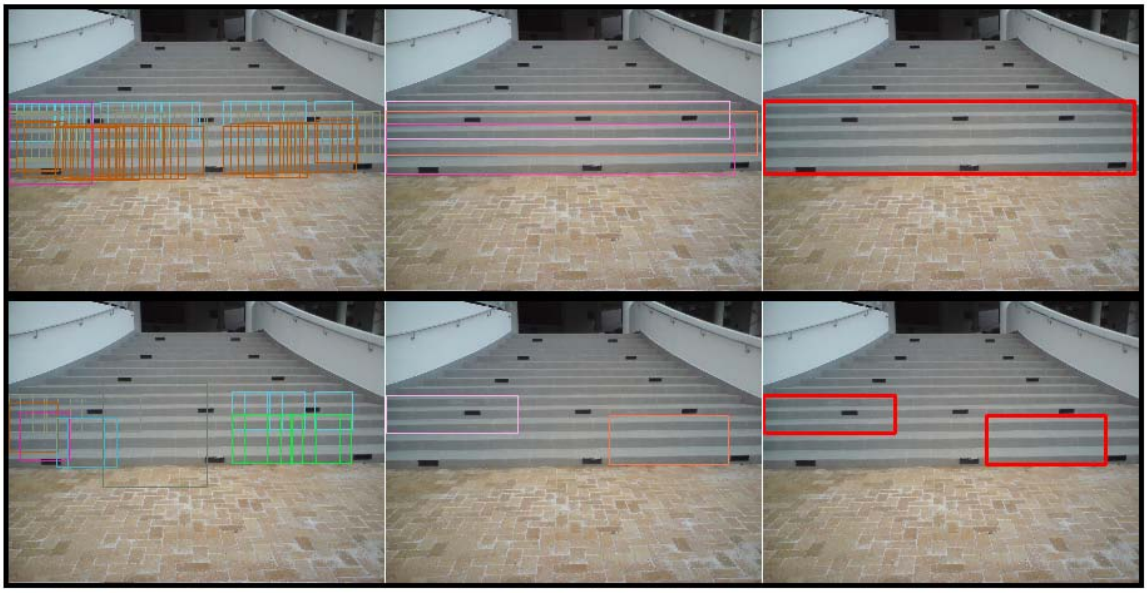

Figure 4.10 Staircase detection case 4

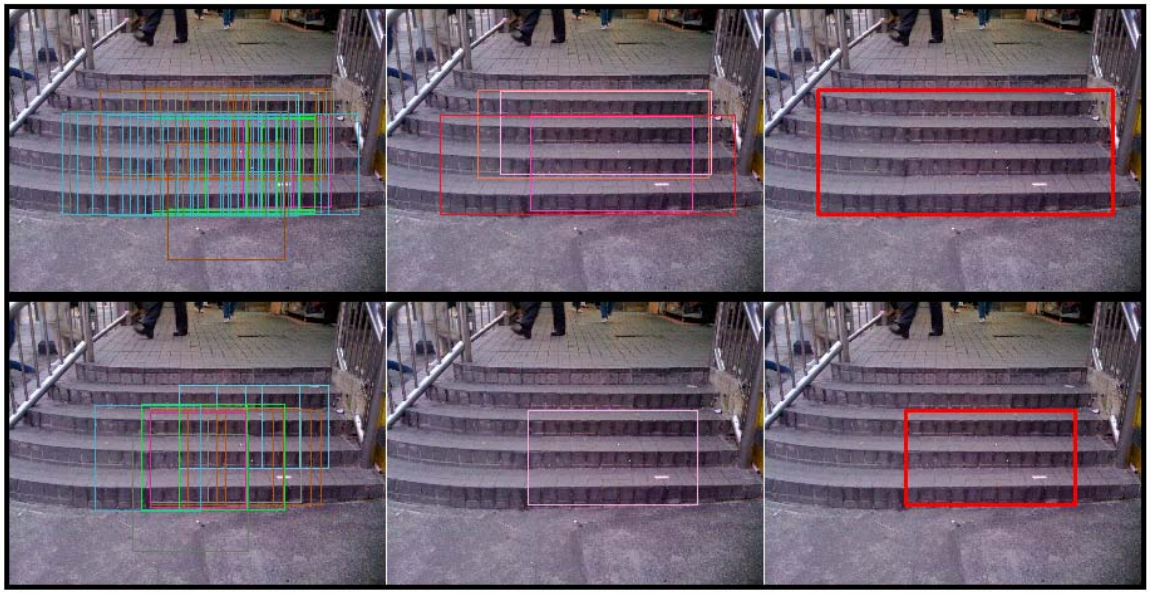

Figure 4.11 Staircase detection case 5

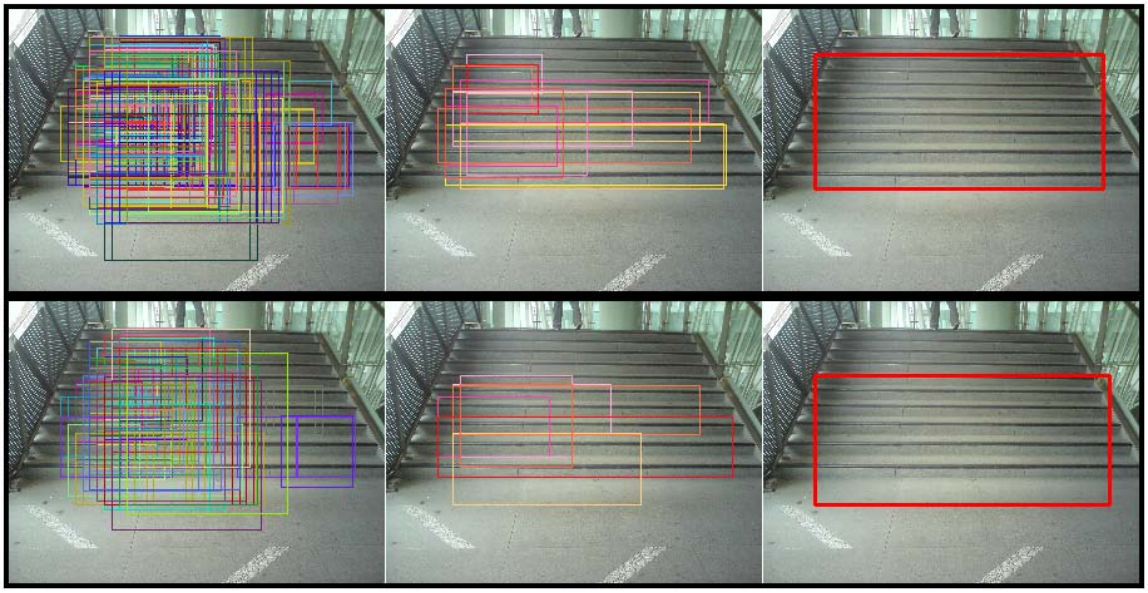

Figure 4.12 Staircase detection case 6

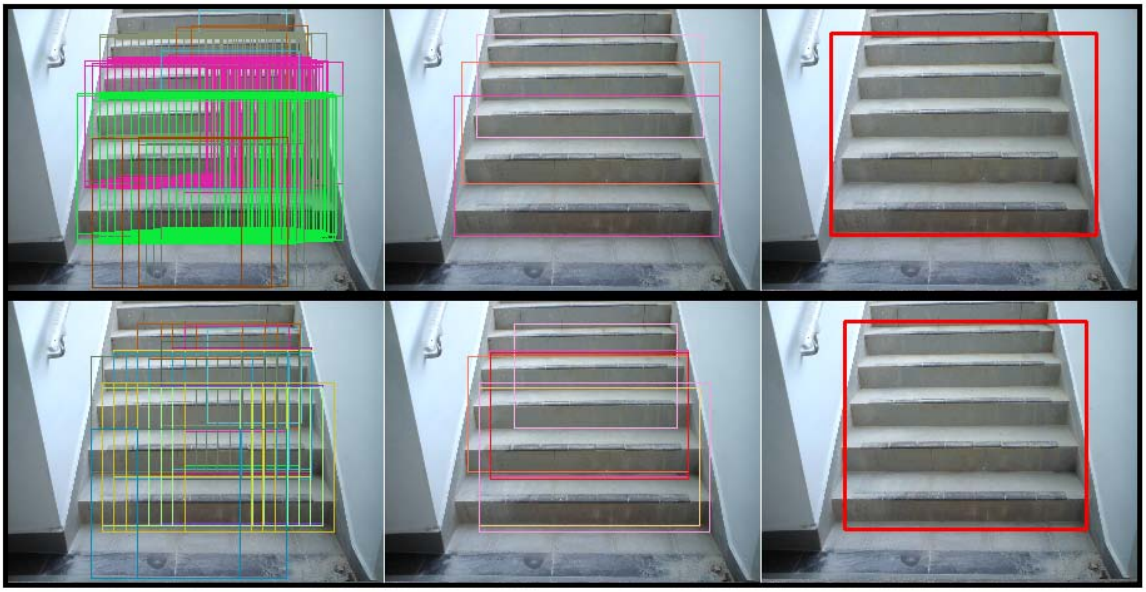

Figure 4.13 Staircase detection case 7

Figure 4.14 Staircase detection case 8

A systematic description of the performance of Viola-Jones framework on staircase detection is presented in the following sections.

4.6.3 Accuracy

Most papers use a ROC curve [26] to indicate the performance of their detectors. A ROC curve depicts how the detection rate changes with false alarm rate. It is created by continuously lowering the stage threshold and consequently, removing each layer backwards from the cascade. Thus, to create a ROC curve, we need to know the number of correct detections and the number of false detections as well as the number of labeled positives and total number of sub-windows scanned through the entire test set. Besides, a Boolean classification scheme is required to qualify the detections. As mentioned in Section 4.6.1, a confidence-rated evaluation metric is introduced to replace the Boolean scheme. Thus, instead of using a ROC curve, an accuracy curve showing the relations between accuracy and the number of stages, and a false alarm rate curve displaying false alarm rate versus stages are created.

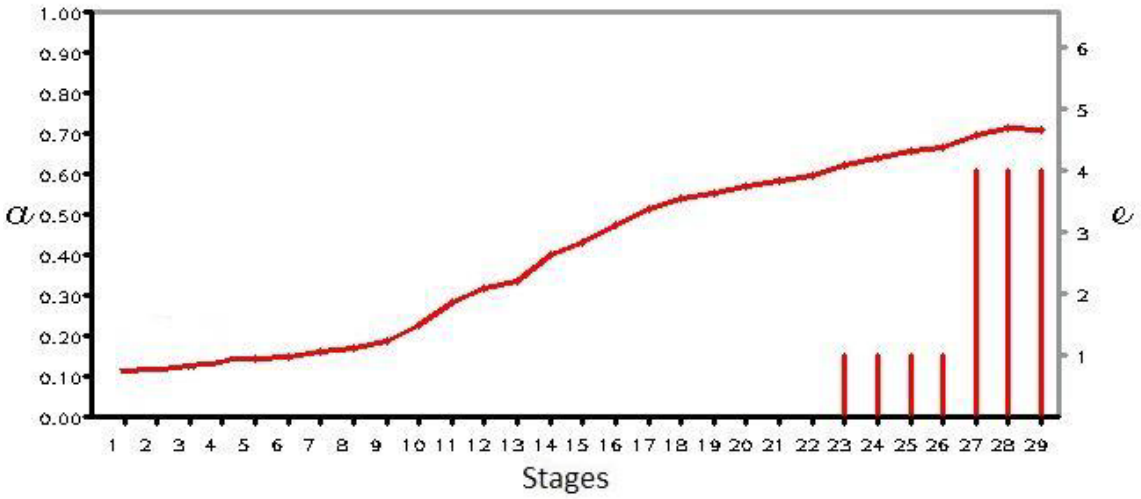

Figure 4.15: Accuracy curve for test environment 1

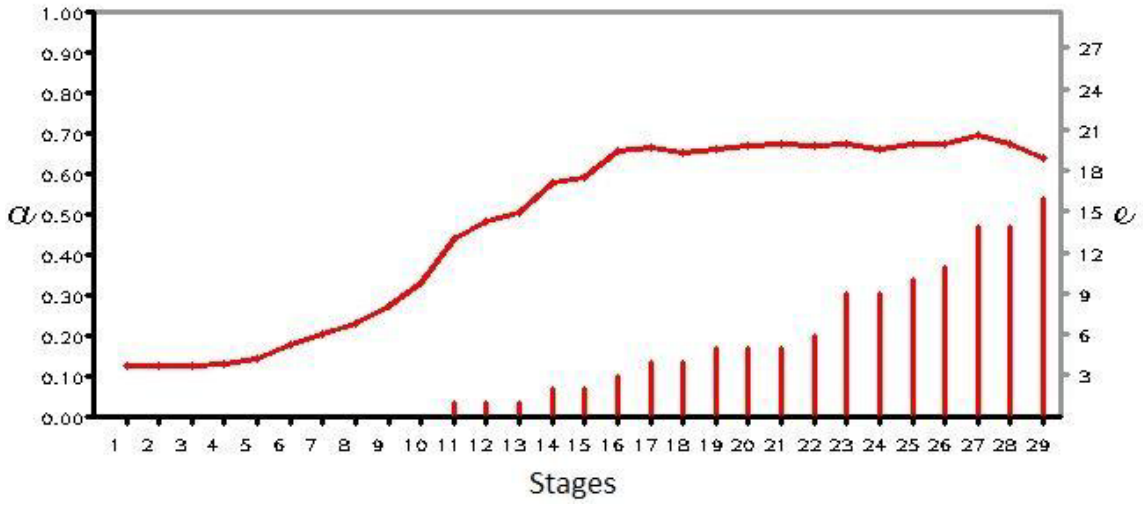

Figure 4.16: Accuracy curve for test environment 2

The accuracy curves for test environments tabulated in Table 4.4 are shown in Figure 4.15 and Figure 4.16, respectively. The two test environments only differ in the scanning scheme (dense vs. sparse).

In the figures, the left axis (α) is the average accuracy for all detected instances in the test set. The right axis (e) represents the number of undetected instances out of the 55 test images. An undetected instance is characterized by 0 detection accuracy.

Firstly, the curves give us an overall impression that the accuracy tends to increase as more layers are added to the cascade. Initially, the curves start from an equivalent accuracy level (~0.1) and then both undergo a smooth rising period (dense: stage 1 – 9, sparse: stage 1 – 5). The dense scanning curve then switches to a relatively faster increasing period (stage 9 – 29). On the other hand, the sparse scanning curve gets into an exponential rising period (stage 5 – 16) before reaching a plateau (stage 16 -29).

Surprisingly, sparse scanning does not drag down the accuracy curve; instead, accuracy is boosted above its dense counterpart. At a first glance, it seems unnecessary to employ a dense scanning scheme. However, sparse scanning leaves much more instances undetected compared with dense scanning. The numbers of undetected instances are shown as columns below the accuracy curve. Since the accuracy curve is computed only on the detected instances, negative factors that could drag down the accuracy curve have been dumped by the sparse scanning scheme.

Another byproduct of accuracy curve is that it can help us decide the optimal number of active stages to use for the current scanning scheme. For example, the dense scanning curve shown in Figure 4.6 indicates that 26 layers is the best balancing point between accuracy and false negative rate. For the sparse scanning, 16 layers will be the best. However, practical optimization process needs to consider more criteria, such as false alarm rate and detection speed, which are topics to cover in next two sections.

4.6.4 False Alarm Rate

As mentioned in Section 4.6.3, the original ROC curve has been replaced by an accuracy curve and a false alarm rate curve in the context of staircase detection. While the previous section has covered the accuracy curve, this section will continue to explore the relation between the false alarm rate and the number of active stages in the cascade.

Before we start, we shall define what a false alarm is. In face detection, a false alarm, also known as false positive, is a detection (after integration) that cannot be justified by any labeled face, while false alarm rate is obtained by dividing the number of false alarms by the total number of windows scanned. In the context of staircase detection, we use confidence rather than Boolean classification (see Section 4.6.1). Thus, we could not obtain the number of false alarms in a general sense. To produce the false alarm curve, we redefine a false alarm as a primitive detection that falls beyond the labeled region. The false alarm rate is obtained as usual by dividing the number of false alarms by the total number of scanned windows. Since it counts primitive detections, the false alarm rate would be hundredfold higher than expected.

To make the curves more readable, the curves are shown in 2 forms. Figure 4.17 –4.18 show the partial and complete false alarm rate curves for the first test environment. Figure 4.19 -4.20 show the partial and complete false alarm rate curves for the second test environment.

Under the dense scanning scheme, a total of 7,277,906 sub-windows have been scanned, while the sparse scheme scans 1,745,485 sub-windows. The false alarm rate curves for the 2 test environments are very close to each other. Generally, false alarm rate increases exponentially as the stages are removed gradually from the cascade.

When determining the optimal number of active stages in the cascade, there is no optimum point in terms of false alarm rate as long as it is below an acceptable threshold.

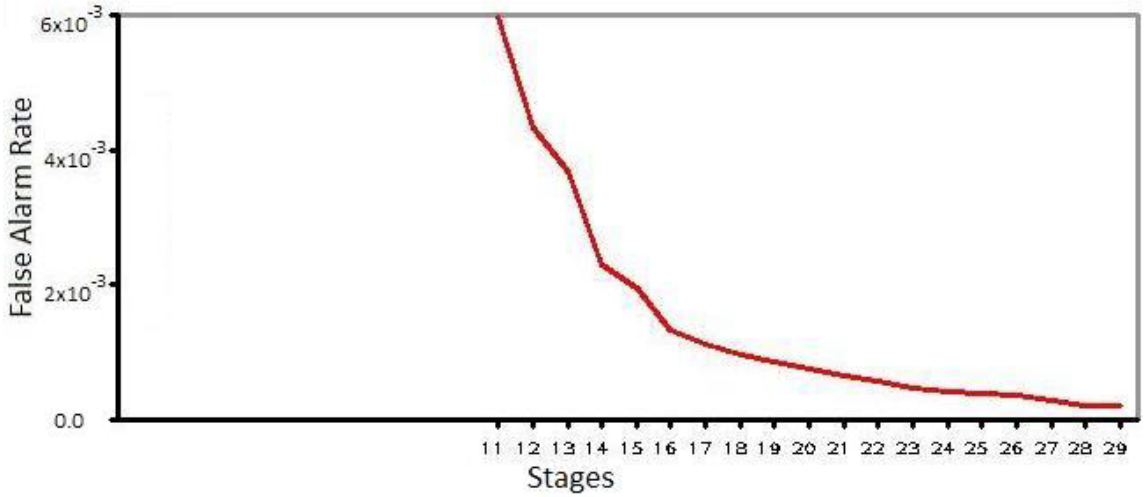

Figure 4.17: False alarm rate curve (partial) for test environment 1

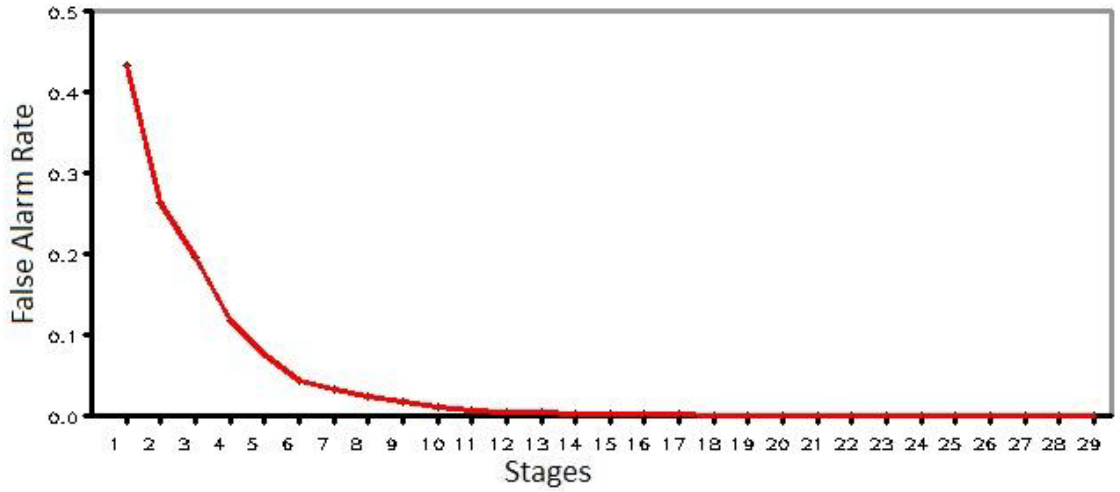

Figure 4.18: False alarm curve (complete) for test environment 1

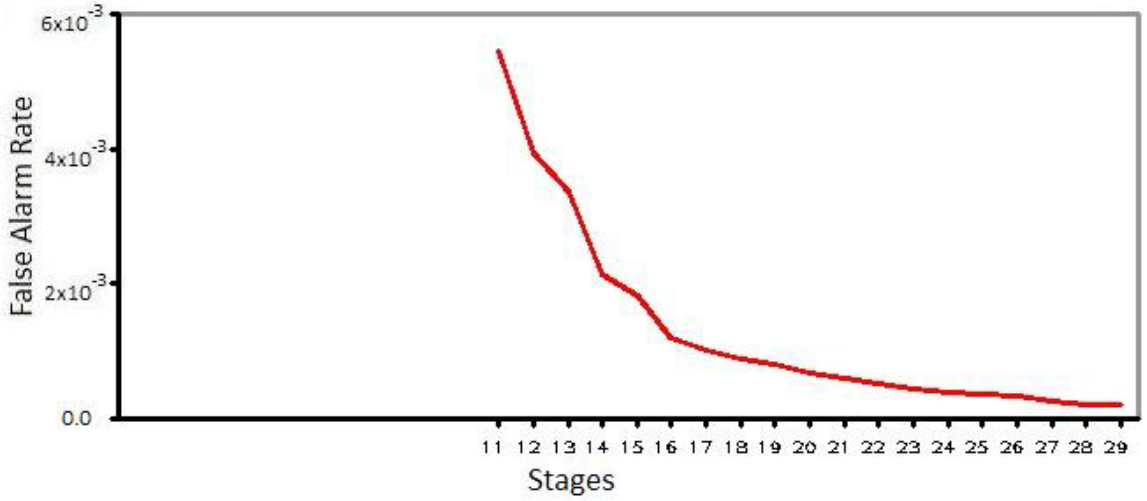

Figure 4.19: False alarm rate curve (partial) for test environment 2

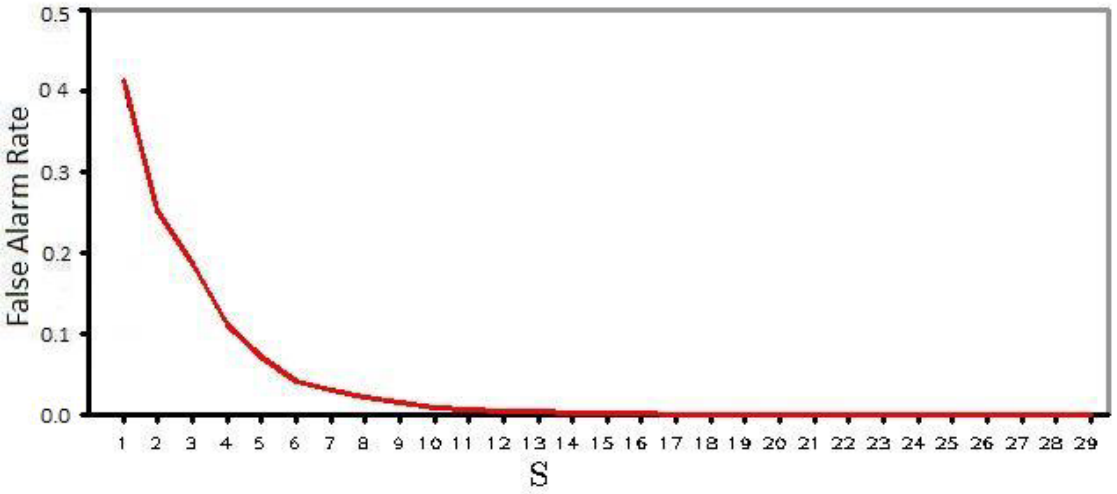

Figure 4.20: False alarm curve (complete) for test environment 2

4.6.5 Speed

The detector's speed depends on the parameter settings, especially the scanning scheme and of course the machine that it runs on. Our program was coded in C and run

on a Pentium IV, 2.66 GHz PC. A dense scanning scheme requires the detector to run through more sub-windows and consequently produces more primitive detections. Hence, a dense scanning scheme incurs higher scan time and integration time. Conversely, a sparse scanning scheme scans fewer sub-windows and desires less time but may end up with a worse performance.

The speed curves for the two scanning schemes described in Table 4.4 are shown through Figure 4.21 to Figure 4.24. The processing time consists of a scanning time and an integrating time shown in red and blue, respectively.

For readability, scanning time is separated from the overall speed curves. The scanning time can be estimated using a prototype illustrated in Table 4.5 based on 2 weak hypotheses:

- Each stage consumes equal time to scan a window. (This is weak as stages are different in terms of the number of weak classifiers they contain.)
- Each window desires equal scanning time in a stage. (Again, this is weak as windows are different in size.)

With the above weak hypotheses, we can deduce the following:

- It requires equal time for any stage to scan any window.

It must be aware that this conclusion is tenable only weakly, but it could be applied to predict the tendency of scanning time curves as follows:

Assume a is the time for a stage to scan a window, p_i is output/input ratio for stage i, ($p_i < 1$), T is the number of test images ($T = 55$), A is total sub-windows scanned (dense scanning: $A = 7{,}277{,}906$; sparse scanning: $A = 1{,}745{,}485$).

Scanning time:

Stage 1: $t_1 = Aa/T$;

Stage 2: $t_2 = t_1 + p_1 Aa/T$;

Stage 3: $t_3 = t_2 + p_1 p_2 Aa/T$;

$\vdots$

Stage n: $t_n = t_{n-1} + p_1 p_2 \cdots p_{n-1} Aa/T$; (4.11)

Table 4.5: Scanning time prototype

Equation 4.11 indicates as the number of stages increases, the scanning time increment imposed by each newly added stage will decrease. Both Figure 4.12 and Figure 4.14 show that after a transition point, the curves start to slope gently. After the point, adding new layers into the cascade will not cause violent fluctuation on the scanning time.

Figure 4.22 and 4.24 incorporate both curves. Integration time has a more complex relation with the number of active stages in the cascade. The integration time is initially thousand times of the scanning time but decreases exponentially as new stages are added. Eventually, integration takes only a fraction of the scanning time.

Compared with the integration time, scanning time does not vary significantly with the length of the cascade. When considering how the cascade length influences the

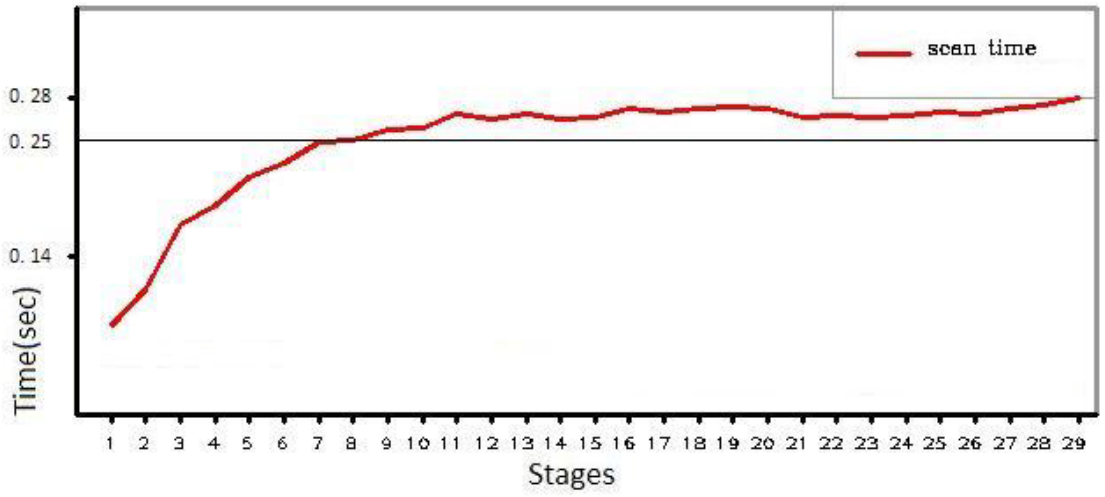

Figure 4.21: Scanning time curve for test environment 1

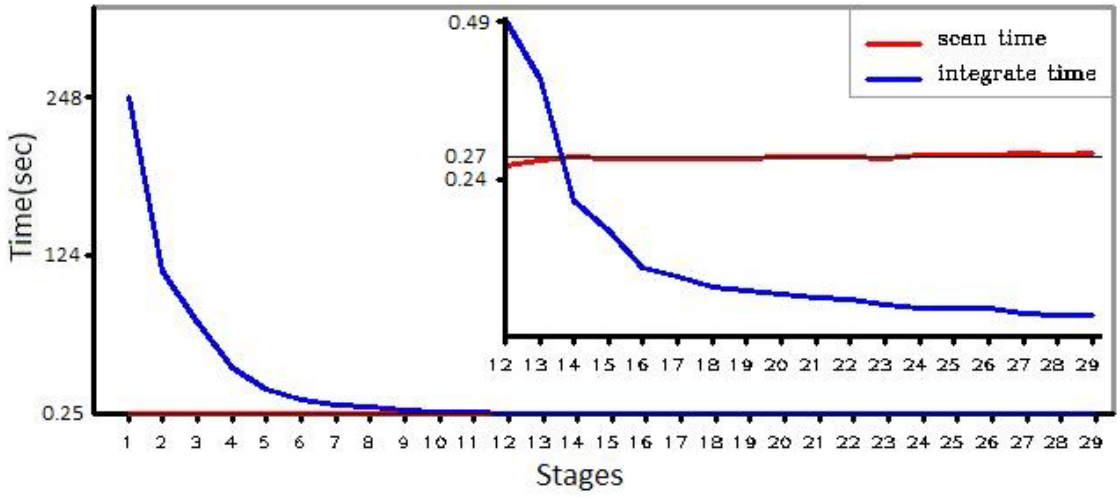

Figure 4.22: Speed curve for test environment 1

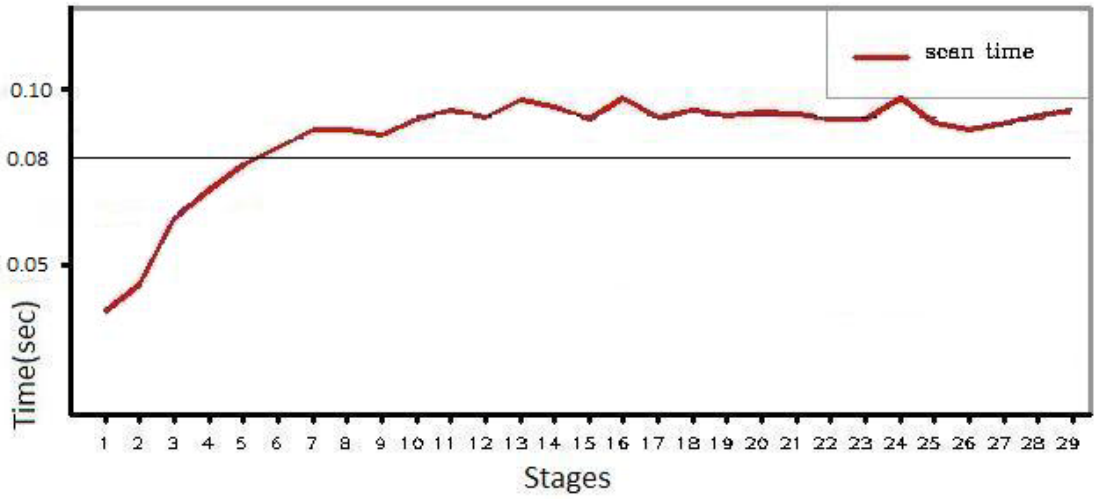

Figure 4.23: Scanning time curve for test environment 2

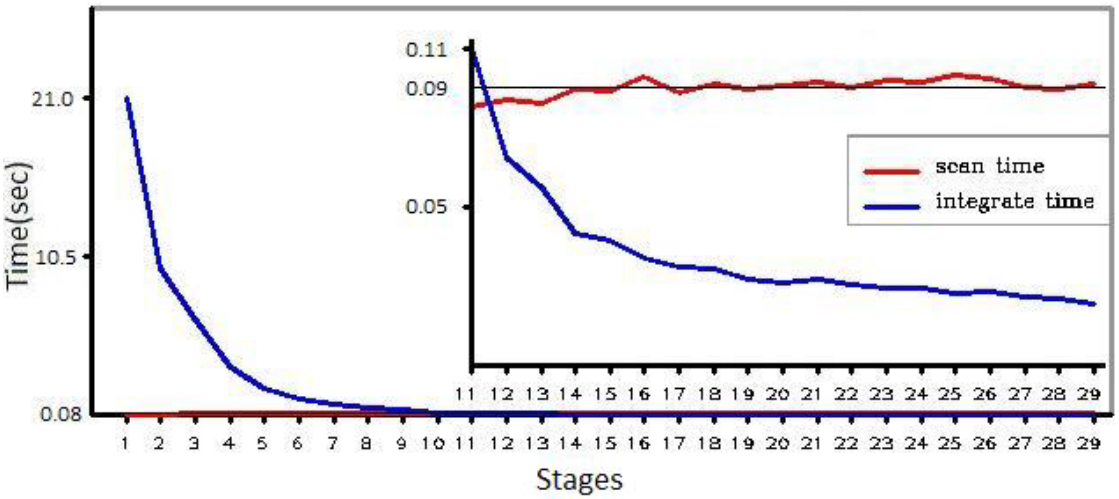

Figure 4.24: Speed curve for test environment 2

detector's speed, priority should be given to the integration time. Usually, active layers should be sufficient to bring the integration time down to the same level as the scanning time. A good practice is to choose a point to the right of the intersection of the 2 curves. For example, for the sparse scanning curves shown in Figure 4.24, there should be at

least 12 stages in the cascade to guarantee the detector's speed. The overall process time can be obtained by simply adding up the scanning time and the corresponding integration time. For a 12-stage cascade shown in Figure 4.24, it takes about 0.15 second to process a test image, while the time could be reduced to 0.11 second if all the 29 stages are used.

As mentioned in Section 4.6.3, for a given scanning scheme, determination of optimum stage number is a compromise between accuracy, false negative/positive rate and speed. For the sparse scanning scheme listed in Table 4.4, a 16-layer cascade is an optimal choice.

4.6.6 Comparing with previous work

In this section, we shall present the comparison of the proposed algorithm's performance with the state-of-the-art algorithm [14]. Besides the positive test set used in previous sections, the evaluation is also performed on the McGill man-made object data set [51], which comprises 311 images, mostly in urban settings. Note: the data set contains one side view of a stair and a distant stair, both beyond our problem domain (Line range: $[-20°,+20°]$). Therefore, these cases should not be detected.

Derived from the previous experimental statistics, we selected the sparse scanning scheme with an optimal number of 16 active stages to conduct this comparative study.

As it has been discussed in Chapter 2, Hough transform does not perform well in most circumstances for line-based stair detection. Instead, edge tracking and contour

following is used in the line detection stage. As for partition, the last two methods were attempted: linear homography and intensity variation, since cross-ratio requires at least 4 consecutive concave edges or 4 consecutive convex edges, a stricter condition than most samples can provide.

Method	Positive test set (55 samples)			Negative test set (311 samples)	
	Detected	Correct Partitioned	Avg Speed (sec)	Reported	Avg Speed (sec)
Our method (sparse)	52	NIL	0.135	0	0.010
Linear homography	30	2	0.266	45	0.069
Intensity variation	35	11	0.310	68	0.070

Table 4.6: Summary of detection results for various methods

Table 4.6 summarizes the detection results for all the 3 methods. Detailed analysis is given in following subsections.

4.6.6.1 Detection rate

According to Table 4.6, Viola-Jones framework has a much higher detection rate than the line-based methods. In terms of partition correctness, the performance of line-based methods is rather disappointing.

A selective display of the detection results is given through Figure 4.25 to Figure 4.27. The middle column corresponds to the linear homography partition method. Red and green lines represent the partitioned concave and convex stair edges.

4.6.6.2 Partition

Intuitively, you may think the linear homography partition method with rigorous mathematical argument should be reliable than the empirically plausible intensity variation method. However, the truth cannot justify this speculation. From the results, we can go a step further to predict that cross-ratio performs even worse as it will leave out more staircase instances than linear homography. This prediction consists with the one given by Se [14].

Linear homography may incorrectly group irrelevant lines but satisfying the geometric constraint into a partition. In fact, most of the partitions are incorrect. In contrast, genuine combinations may not be initiated due to problems in line detection, such as line discretization and deviation as well as completely missing the edges. Although intensity variation is purely empirical observation, and may not universally hold. It performs robustly if the observation holds for a specific case.

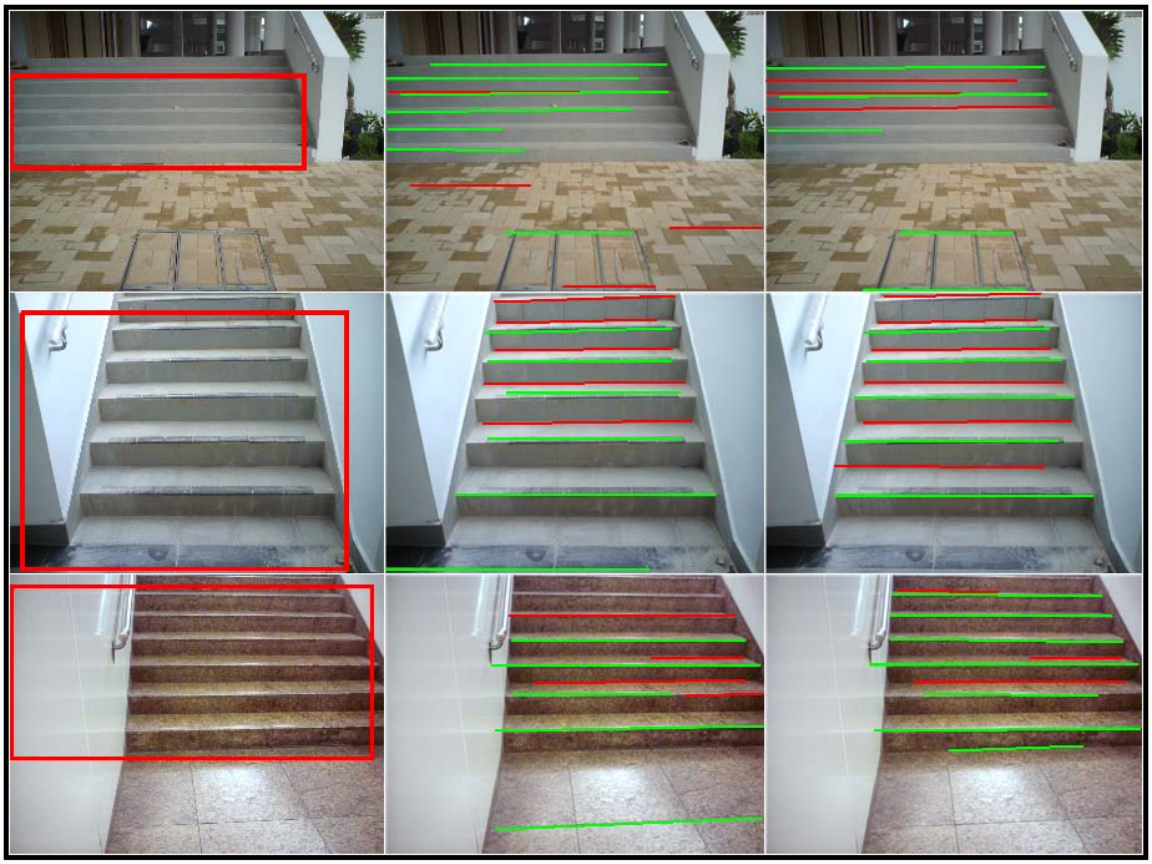

Figure 4.25 Comparative study case 1

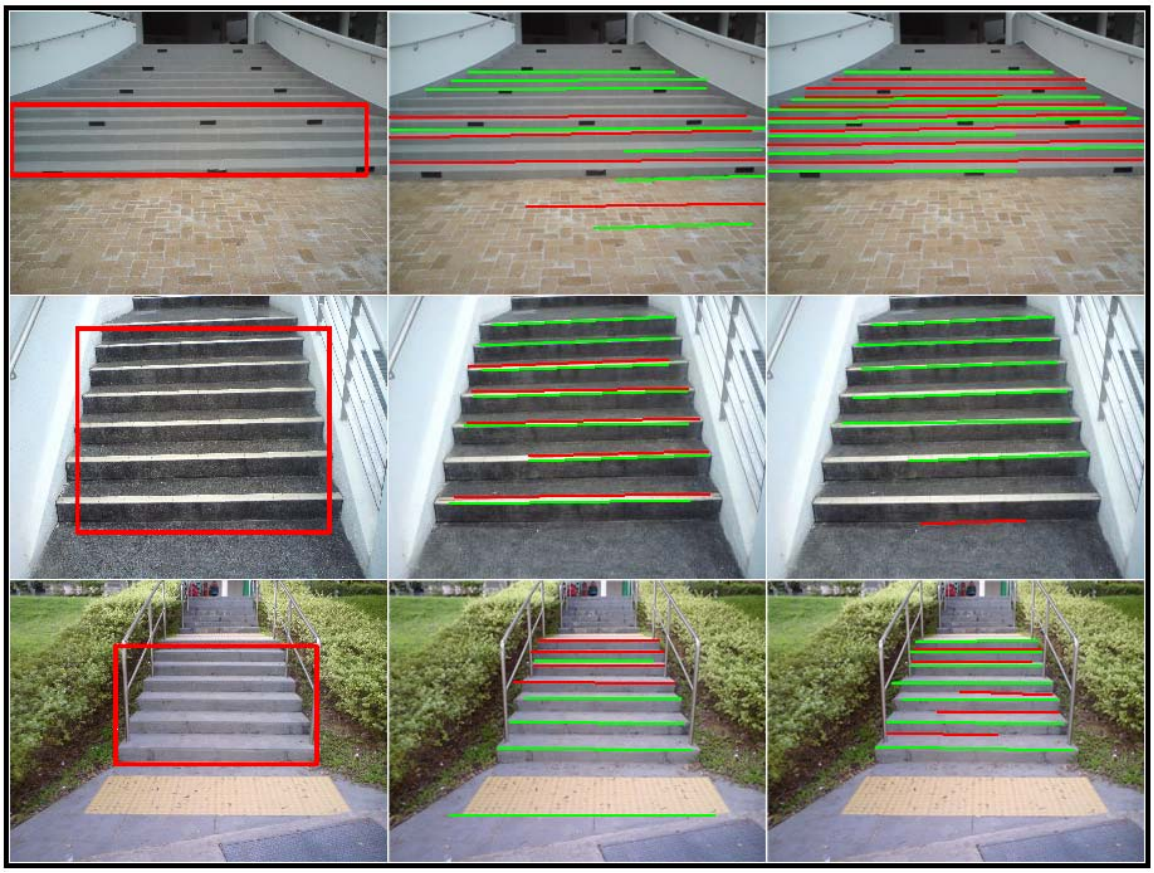

Figure 4.26 Comparative study case 2

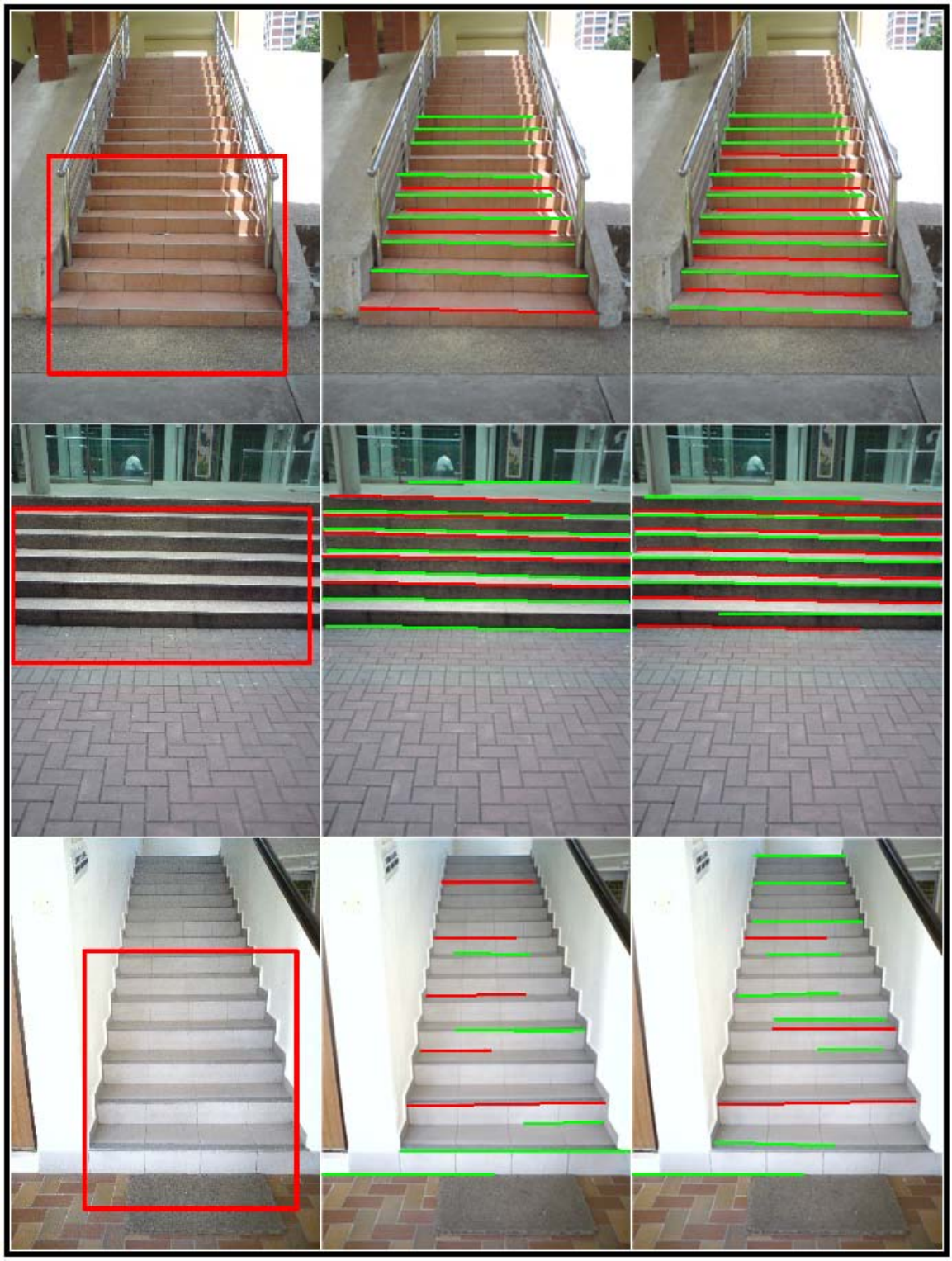

Figure 4.27 Comparative study case 3

4.6.6.3 False alarm

The results on the negative test set amply demonstrate the superiority of Viola-Jones object detection framework. It didn't raise a single alarm out of the 311 images, while line-based methods reported a lot of fake detections. In this respect, Viola-Jones framework puts the line-based methods in the shade.

Figure 4.28 and Figure 4.29 give some false detections of linear homography and intensity variation respectively. Note: to safeguard the detection rate from dropping too low, the order of contact (interleaving) constraint [14] is relaxed a bit to allow missing of edges.

As previously pointed out, this thesis deals with near-frontal looking staircase detection. The line detection has been constrained in the range of $[-20°,+20°]$. Much more false alarms will be raised from the line-based methods should we remove this restriction.

Figure 4.30 shows some occasional survivals of the Viola-Jones cascade detector. In Section 4.5, these are called primitive detections. They will eventually be discarded since no further homogeneous detections can be found to support their validity.

Figure 4.28 False detections of linear homography

Figure 4.29 False detections of intensity variation

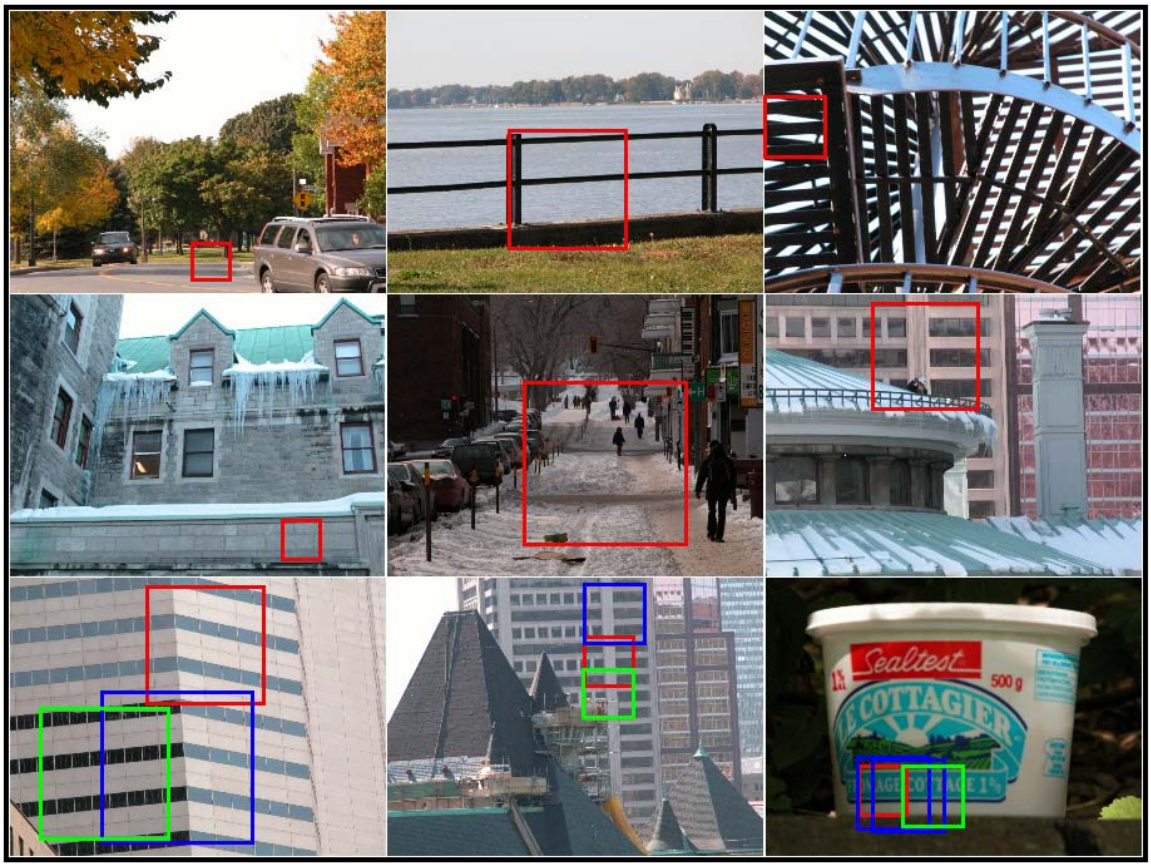

Figure 4.30 Occasional survivals (primitive detections) of cascade detector

4.6.6.4 Speed

The beauty of Viola-Jones framework consists in its swift response. It has obvious advantage over line-based methods in processing speed.

For positive samples, it takes only half of the time needed by line-based methods. The vantage even multiplies on the negative set. For positive samples, there are abundant target regions passing through the full length of active stages of the cascade. This process drags down the speed a bit. While for negative samples, an overwhelming majority of the regions can only survive the first few simple and fast stages after which they disappear. The overall process concentrates on few target-like regions.

4.7 Summary

This chapter introduces a 2D staircase detection algorithm based on Viola-Jones object detection framework. The novelty of this algorithm lies in the following:

- It redefines staircases. In the new definition, a staircase becomes a square region contains 3 consecutive visible steps. The detector thus searches for partial staircase regions rather than complete staircases in the image.
- It uses PCA-based Haar-like features. These features efficiently eliminate apparently negative regions (samples) in the early stages of the cascade.
- It adopts real AdaBoost algorithm to replace the original Discrete AdaBoost.
- It uses a new scanning scheme that makes the scanning speed irrelevant to the image size.
- It uses a specially designed multi-detection integration scheme to combine local "staircases" into global staircases.
- It presents a confidence-rated evaluation metric to measure the accuracy of detections rather than Boolean classifying them.

In addition to the above, this chapter also introduces the training algorithm and gives detailed insight into the first layer of the cascade. When analyzing the performance, detailed experimental data and discussions on detection accuracy, false positive/negative rate and speed are presented, and the method to optimize the cascade length based on these factors is also given.

Chapter 5 An Introduction to "V-Disparity"

5.1 Introduction

The previous chapter introduced a method for monocular staircase detection. However, for the fully autonomous navigation in a 3D world, visual identification of certain types of land features is far from enough. Precise 3D localization of the features is necessary to help the control system make early decisions accordingly. In the scenario of 3D staircase localization, the distance of the staircase to the robot and the dimensions of the stairs are valuable information to assist the robot in climbing the stairs. This chapter reviews previous works done in the literature pertaining to stereo vision based staircase detection and localization. A new image concept known as "V-disparity", commonly used for ground plane estimation in autonomous navigation systems, is introduced in later part of the chapter.

5.2 Previous work on staircase localization

There are limited resources related to stereo vision based staircase detection and localization in the literature. For the existing works that we are aware of, they can be broadly classified into two types - one combines both image brightness information and range data, whiles the other relies merely on range information.

Lu and Manduchi's [52] staircase detector is an enhanced curb detector. Their strategy of fusing image-based and range-based information is grounded on the observation that the quality of stereo range data is often inferior to that of the active sensors. Using stereo data alone usually does not produce acceptable results. In view of this, their method uses image brightness information to extract edges and candidate curb lines are searched over the intensity edge pixels.

A range-weighted Hough transform [53] is used to detect curb lines with the weights derived from the curvature index at each edge point. The curvature index at an edge point is expressed in terms of the mean curvature H and the Gaussian curvature K as:

$$CI = \begin{cases} \dfrac{H^2 - K}{H^2 - \varepsilon} & K \geq 0 \\ \dfrac{H^2}{H^2 - K - \varepsilon} & K < 0 \end{cases} \tag{5.1}$$

The expressions of H and K are reported in [54]:

$$\begin{aligned} K &= \frac{h_{xx}h_{yy} - h_{xy}^2}{(1 + h_x^2 + h_y^2)^2} \\ 2H &= \frac{(1 + h_x^2)h_{yy} - 2h_x h_y h_{xy} + (1 + h_y^2)h_{xx}}{(1 + h_x^2 + h_y^2)^{3/2}} \end{aligned} \tag{5.2}$$

Where h_x, h_y, h_{xx}, h_{yy} and h_{xy} are the first, second and partial derivatives of the measured depth image. A large curvature is a good indicator of ridges of the range data and a point with larger curvature is considered to be more likely to belong to a curb line. This

weighting scheme is useful at filtering out planar brightness edges while surface discontinuities are well preserved.

The detected curb lines are subjected to a 3D line regression process in order to precisely locate the lines in the 3D world. Once the 3D curb line has been regressed, some statistical techniques (Modified Selective Statistical Estimator MSSE [55]) are used to separate inliers on the line from outliers and hence determine the end points of the curb edge segments. Finally, the line segments are labeled as concave and convex based on the sign of the average mean curvature H of its points.

The above procedure is devoted to curb detection. However, it could be generalized as building blocks of a stairway location process. [52] employs the same assumption with [14] to prove the availability of staircases in the scene by examining the existence of 2 sets of parallel convex and concave edges lying on parallel planes.

The staircase localization procedure continues as follows: firstly, a variant of LMeds method is used to robustly fit a plane to the lines in each set. The plane is called a tangent plane. Then, the intersections of the tangent plane with each of its internal viewing plane that contains a detected curb line and the focal point are computed. Finally, the algorithm relies on the parallelism of stair edges in the 3D environment to reject spurious intersections that are arisen from objects other than stair edges. The localization procedure ends up with 2 set of parallel spatial line segments corresponding to the convex and concave stair edges.

Another work done by Gutmann et al [56] relies merely on range data. Plane segmentation from range data forms the basis of their algorithm. They use Jiang and Bunke's [57] scan line grouping approach to segment the range data into planar regions. Extracting planar regions from range data is a well-studied problem. For a comparative review of range image segmentation algorithms, please refer to [58], [59]. Each planar region that is approximately horizontal is examined as a tread surface candidate. Various parameters associated with the underlying tread surface are then extracted. The examination process first approximates each planar region with a convex polygon. Front and back edges as well as the width and margins of the tread surface are then computed from the polygon. The method does not make any prior assumption about the shape of the stair, so it could be used to localize spiral stairs. A test program using this method is implemented, the results show that the range image segmentation stage encounters real-time bottleneck on a general-purpose personal computer and moreover, under-segmentation often occurs during the extraction of stair regions.

Due to the instability of stereo range data and the real-time bottleneck of range image segmentation, Gutmann's method is not a good choice for the comparison with our staircase localization algorithm proposed in Chapter 6. In the experimental part of the next chapter, we'll have further discussions on [52]. Essentially, [52] could be regarded as a 3D line-based stair detection and localization algorithm.

The remaining sections of this chapter is devoted to an important image concept, known as "V-disparity" image and any peripheral knowledge necessary to reinforce

readers' understanding of the concept. Familiarity with the "V-disparity" concept is a must for understanding the staircase localization algorithm proposed in the next chapter.

5.3 V-Disparity

5.3.1 Camera placement and geometry

One basic requirement for precise 3D localization of target objects is the availability of 3D data. If an accurate 3D snapshot of the surrounding environment can be acquired, it should not be a difficult problem to identify a staircase. Various means for acquiring 3D data can be seen on a typical autonomous robot which exhibits certain degree of motion autonomy. Radar and laser rangefinder (LRF) are commonly used 3D sensors. Although these active sensing systems generally provide high accuracy measurements, they are bulky, expensive, slow and sensitive to various distractions. A stereo vision camera can be considered as an alternative. Compared with the active sensors, stereo vision sensor has the advantages of a wide field of view (FOV), excellent lateral accuracy, low cost, small dimensions and freedom from signal interference, etc. A number of stereo devices are available commercially. For this project, we have used a popular product: the Point Grey Bumblebee stereo vision camera with baseline b of 12 cm.

The Bumblebee stereo camera is pre-calibrated and does not need infield calibration. The two cameras remain coplanar during the lifespan and have the same intrinsic

parameters: a unique focal length f and equal scaling at two image directions. The stereo rig is mounted on the robot as shown in Figure 5.1 with two changing factors:

$-h$ height of the camera above the ground

$-\alpha$ pitch angle of the camera with respect to ground plane

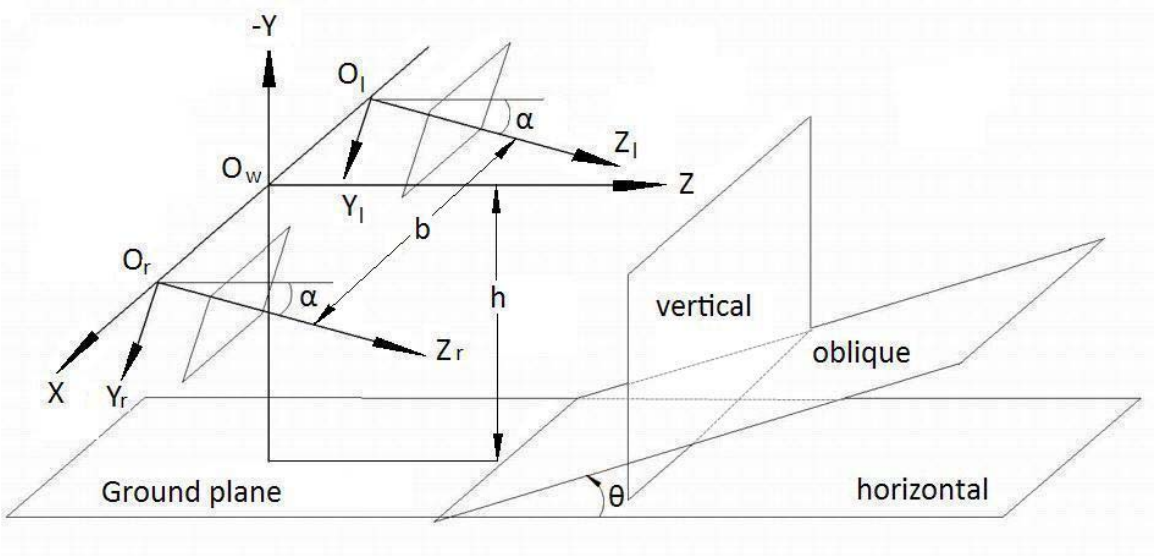

Figure 5.1: Camera geometry and three special planes

The camera geometry shown in Figure 5.1 implies the parallel epipolar constraint is fulfilled. To facilitate the declaration of the problem, we define 3 coordinate systems as shown in the figure. We put the origin of the world coordinate system WCS (X,Y,Z) to the center of the origins of two stereo camera coordinate systems: LCS (X_l,Y_l,Z_l) and RCS (X_r,Y_r,Z_r). The Z-axis of the WCS is parallel to the ground plane, while the Z-axis of the camera coordinate system is aligned with their respective optical axis. The 3 coordinate systems share a common X-axis as shown in Figure 5.1.

5.3.2 Plane projection into $\delta - v$ domain

Given the 3 coordinate systems and the camera geometry shown in Figure 5.1, the transformation from WCS to CCS is achieved by the combination of a vector translation of $\pm b/2$ and a rotation around the X -axis by an angle of $-\alpha$. Let T_l , T_r denote the translation matrices, R_l and R_r the rotation matrices from WCS to LCS and RCS respectively. The homogeneous transformation matrix is:

$$M_{lr} = R_{lr}T_{lr} = \begin{bmatrix} 1 & 0 & 0 & \pm\frac{b}{2} \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

In the image domain, the position of a point is given by its coordinates (u, v) . The image coordinate of the projection of the optical center, which is often known as principal point, will be denoted by (u_0, v_0) . Given a unit aspect ratio and focal length f , we can formulate the intrinsic matrix as:

$$K = \begin{bmatrix} f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5.4}$$

After obtaining the extrinsic and intrinsic matrices, the transformation between the WCS and image coordinate system is represented in Equation 5.5, adopting a pin-hole camera model.

$$\tilde{s}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KM_{lr}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{5.5}$$

$\tilde{s}$ is an arbitrary nonzero scale factor. From Equation 5.5, we can calculate the image coordinate of the projection of any point $P(X,Y,Z)$ defined in WCS:

$$\begin{aligned} u_{l,r} &= u_0 + f\frac{X \pm b/2}{Y\sin\alpha + Z\cos\alpha} \\ v_{l,r} &= v_0 + f\frac{Y\cos\alpha - Z\sin\alpha}{Y\sin\alpha + Z\cos\alpha} \end{aligned} \tag{5.6}$$

With epipolar constraint we have $v_l = v_r$, combined with Equation 5.6, we can derive the disparity for any point $P(X,Y,Z)$:

$$\delta = u_l - u_r = f\frac{b}{Y\sin\alpha + Z\cos\alpha} \tag{5.7}$$

With Equations 5.6 and 5.7, we could derive a linear relationship between δ and v for a special family of planes which are parallel to the X -axis.

5.3.2.1 Horizontal plane projection

The horizontal plane can be simply represented as Equation 5.8 in the WCS:

$$Y = h \tag{5.8}$$

Combing with equations 5.6 and 5.7, we can derive the following linear equation:

$$\frac{h}{b}\delta = f\sin\alpha + (v - v_0)\cos\alpha \tag{5.9}$$

5.3.2.2 Vertical plane projection

On the other hand, a vertical plane in the WCS:

$$Z = d \tag{5.10}$$

can be projected into the $\delta - v$ domain as

$$\frac{d}{b}\delta = f\cos\alpha - (v - v_0)\sin\alpha \tag{5.11}$$

5.3.2.3 Oblique plane projection

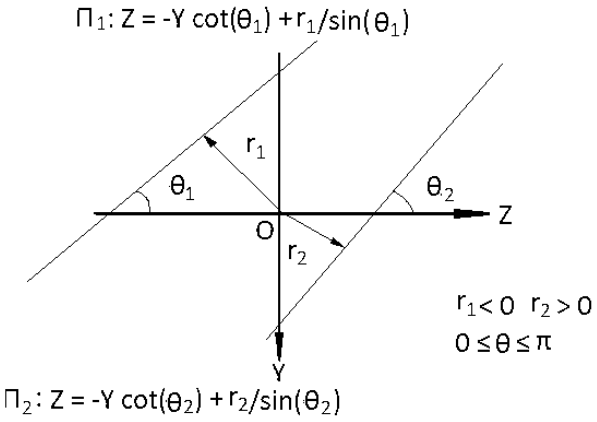

Figure 5.2: Side view of oblique planes

For general oblique planes shown in Figure 5.2, they can be represented by:

$$Z = -Y\cot\theta + \frac{r}{\sin\theta} \tag{5.12}$$

where θ is the angle between the plane and the ground (or Z -axis of WCS) in anti-clockwise direction, starting from the ground plane as shown in Figure 5.2. r is the signed distance from the origin of WCS to the oblique plane, where $r > 0$ when the perpendicular line is on the left of Y-axis of WCS (note: in Figure 5.2, Y-axis is pointing downward). $r < 0$ when the perpendicular line goes to the right of the Y-axis.

Combing Equation 5.12 with Equations 5.6 and 5.7, we can deduce the projection of a general oblique plane as:

$$\frac{r}{b}\delta = f\sin(\alpha+\theta) + (v - v_0)\cos(\alpha+\theta) \qquad (5.13)$$

The horizontal planes and the vertical planes can be viewed as special oblique planes. Thus, their projective Equations 5.9 and 5.11 should be derivable from Equation 5.11 if proper values of θ and r are substituted into the equation.

For horizontal planes, $\theta = 0°(180°), r = h(-h)$, while for vertical planes, $\theta = 90°, r = d\ (d > 0)$, in order for the plane to be visible. All planes that are eligible for projective analysis must be within the FOV of the cameras.

For the various planes discussed so far, the essential nature that characterizes each plane is their respective slanting angle θ with the ground plane. Although their projections into the $\delta - v$ domain can be boiled down to a single Equation 5.13, it still takes some time to derive it.

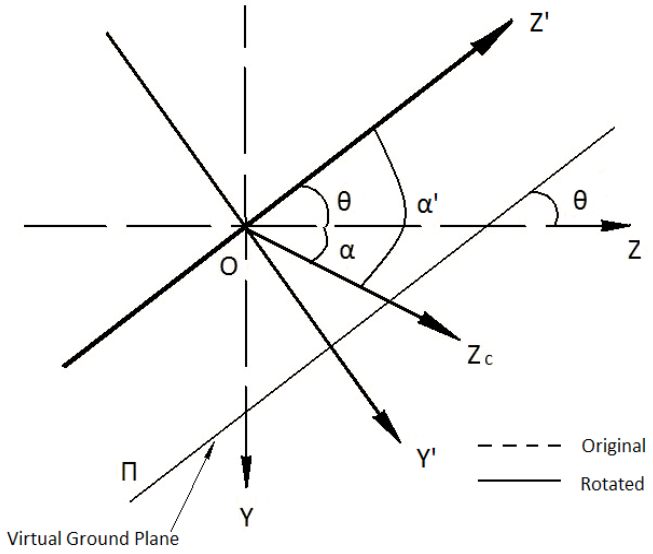

Figure 5.3: Virtual ground plane

Let us change the way of looking at the problem. Instead of viewing each plane as a changing factor, we can consider the planes constant. In the new formulation of the problem, the stereo camera maintains its pose, and the oblique plane is assumed as a virtual ground plane, while the WCS is rotated about X -axis to align its Z -axis along the virtual ground as shown in Figure 5.3. After reformulating the problem, projections for any plane could be viewed as a horizontal plane projection problem and could be derived from Equation 5.9. For a plane with slanting angle θ, we first rotate WCS by θ to obtain a new WCS, denoted as WCS'. In WCS', the plane can be reduced to:

$Y' = h' = r$, while the new pitch angle $\alpha' = \alpha + \theta$. Substituting into Equation 5.9, we can obtain Equation 5.13 effortlessly.

Equation 5.13 gives us some clues about the relations between disparity δ and the row index v for a special family of planes after projecting into the $\delta - v$ domain. These planes are parallel to the X -axis and are frequently encountered in a typical driving scene, especially on structured roads. Thus, the equation has a major application in autonomous navigation or driving scene analysis.

At the moment of image acquisition, h, f, b, α and θ are all fixed values, leaving the disparity δ to change only with row index v linearly. This finding has inspired the invention of a new image representation technique which is well-known as "V-disparity" image.

5.3.3 "V-disparity" Image

Labayrade [60] proposed the "V-disparity" concept aimed at simplifying the process of separating obstacles from road surfaces. Recently, "V-disparity" image has become popular for ground plane estimation [61], [62], [63]. The image is computed by accumulating the pixels of the same disparity along each row of the original image. Let us consider a pixel P on the "V-disparity" image with coordinates (v_p, δ_p). The intensity of P equals the number of original image pixels on row v_p that have a disparity of δ_p. In the "V-disparity" domain, the longitudinal profile of the ground plane or road surface can be projected as a straight line as proved by Equation 5.13, assuming a flat

topography. Thus a 3D plane estimation problem can be reduced to a 2D line estimation problem.

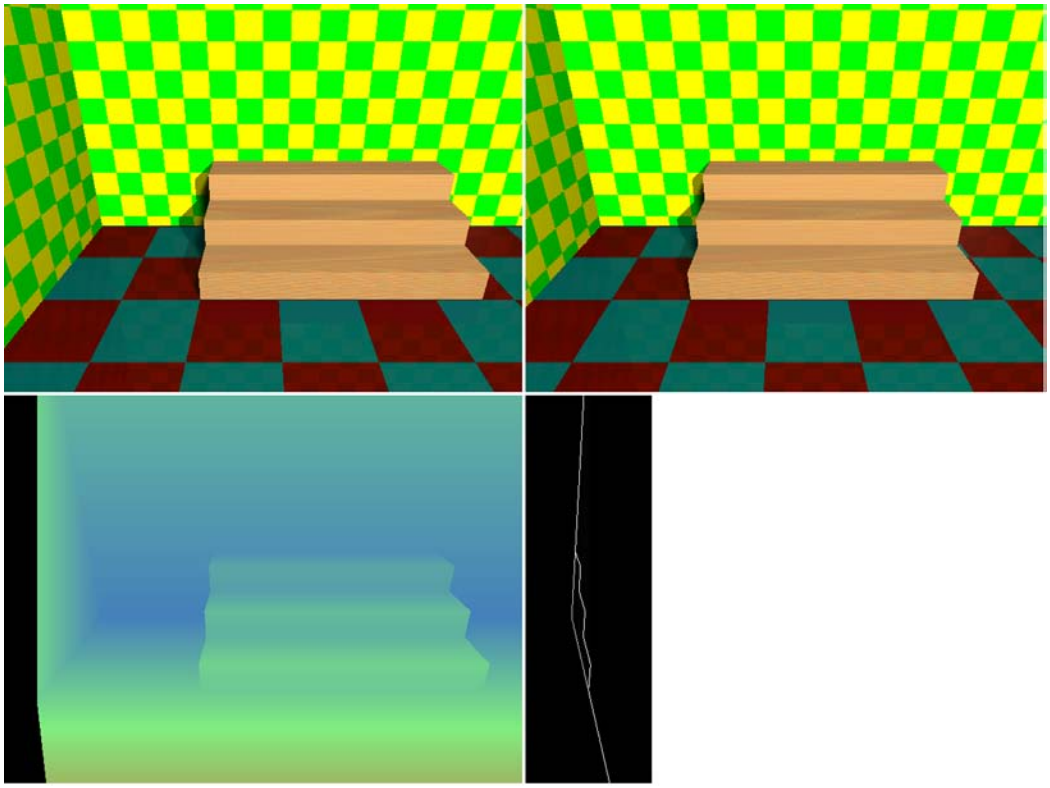

Figure 5.4: Synthetic stereo pair and “V-disparity” image

The top row of Figure 5.4 displays a pair of synthetic stereo images. The cameras are facing forward (pitch $\alpha = 0$). The scene contains a few planer surfaces: 2 vertical walls, a floor plane and a series of mutually orthogonal stair surfaces. The bottom left sub-figure gives the disparity image from where the “V-disparity” image is generated and displayed on the right. For the sake of readability, the “V-disparity “image is scaled by a factor of 2 in the horizontal direction to scatter the pixels a bit.

We can see 2 long segments intersect on the middle of the “V-disparity” image. The segments are projected from the frontal wall facing the camera and the floor plane. The zigzag profile flanked by the 2 longer segments is the projections of the stair surfaces. You may wonder where the left wall in the projection is. It is suppressed by the voting scheme that votes the “V-disparity” image pixels. In fact, the missing plane is projected into a wider band left delimited by the segment projected by the frontal wall and above the floor projection.

A common staircase can be modeled as a series of alternate horizontal and vertical planes. If the cameras take a near frontal look at such a staircase just like the one shown in Figure 5.4, it would be reasonable to expect a pencil of slanting lines corresponding to the tread planes and a set of near-vertical lines projected by risers characterizing the “V-disparity” image, as the zigzag profile exhibits in Figure 5.4. This assertion builds a solid foundation for our 3D staircase localization algorithm, a topic that will be discussed in the chapter to come.

5.3.4 “U-disparity” image

Hu extended Labayrade’s work and proposed the “U-V-disparity” concept [64]. Similarly, the “U-disparity” image is constructed by accumulating pixels of same disparity along each column of the original image. Typically, the “U-disparity” image is less popular in road scene analysis related applications as ground plane becomes hard to interpret from

the $\delta - u$ projection. However, it could be used as a subsidiary tool to extract obstacle pixels above the ground plane [63].

5.3.5 Virtual "V-disparity" image

Virtual "V-disparity" image [65] is constructed from a virtual disparity image, which is obtained by projecting a collection of 3D points derived from real disparity image to a virtual image plane. For the synthetic staircase image scenario shown in Figure 5.4, the camera is placed at point A as shown in Figure 5.5, while B is the point where the camera is intended to be moved to. Of course, the camera movement is a hypothetical situation and that is why point B is termed a virtual observing point. At the virtual observing point B, the optical axis is aligned with surface 3. From A to B, the camera undergoes a rotation and a translation, determined by α and h respectively. Therefore, the virtual disparity image can be acquired by homogeneous transformation from the real disparity image. However, some originally visible points may be cast out of the camera FOV or occluded by other points during transformation. If occlusion happens, the virtual disparity image only takes the visible points into account. The bottom left of Figure 5.6 has given the virtual disparity image after a hypothetical transformation of the camera from point A to B, alongside with the real disparity image shown above. For more information on how to construct a virtual disparity image and its applications, please refer to [65], [66], [67].

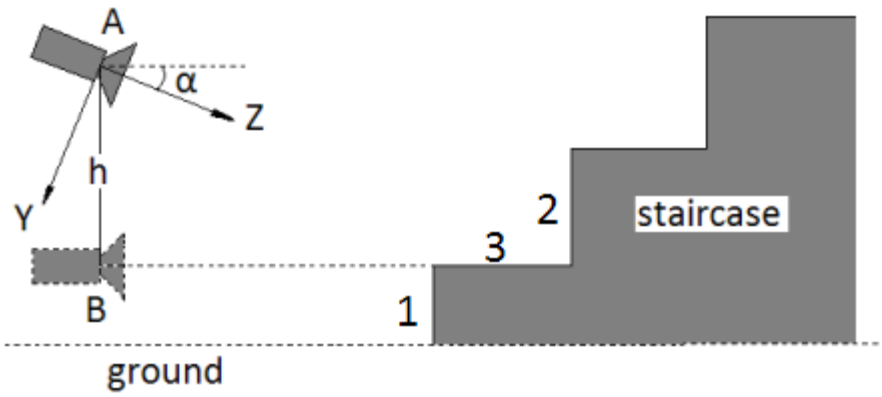

Figure 5.5: Real and virtual camera

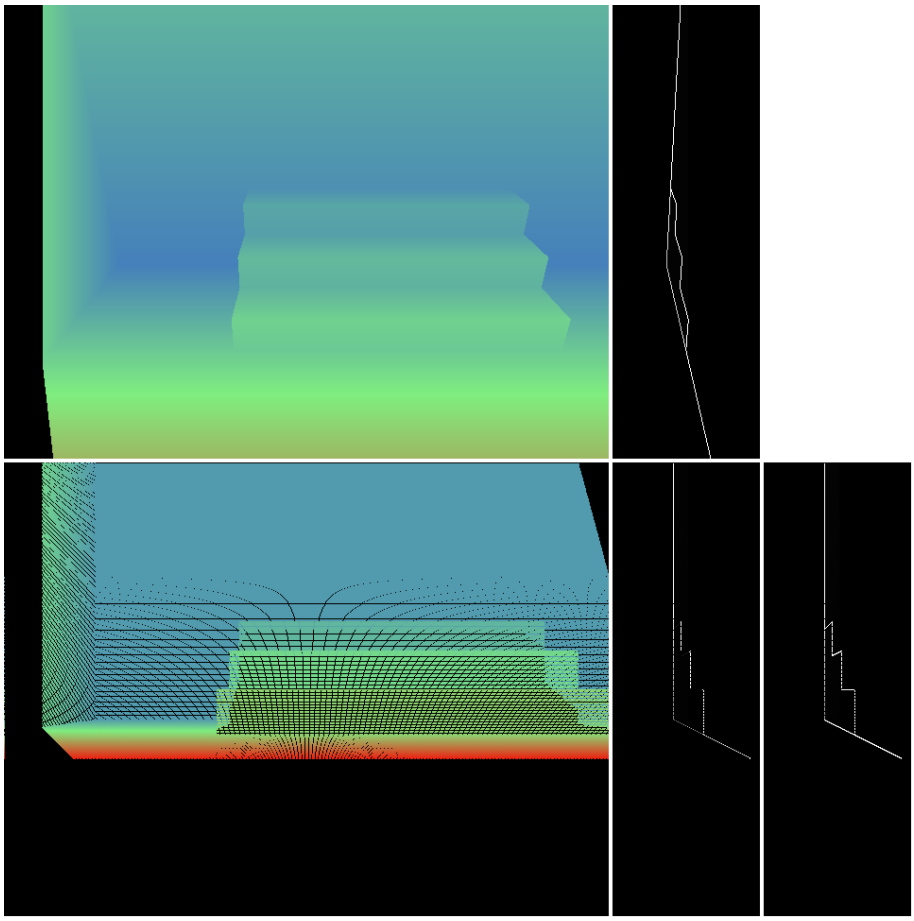

Figure 5.6 Real, virtual disparity and “V-disparity” images

The virtual disparity image could then be used to create the virtual "V-disparity" image. Because the virtual disparity image abandons the occluded points, thereby information may be lost in the formation of the "V-disparity" image. To protect information integrity, "V-disparity" images can be generated directly from real disparity image, given the homogeneous transformation matrix.

In bottom row of Figure 5.6, the middle sub-figure gives the incomplete virtual "V-disparity" image, while lost pixels are filled in the bottom right sub-figure. To distinguish between the two images, we hereafter call the first non-occluded "V-disparity" image, and the latter occluded "V-disparity" image.

5.3.6 Discussions

"V-disparity" projects 3D planes into lines and reduces the dimensionality of problem domain from 3 to 2. On the other hand, virtual "V-disparity" projects some planes into vertical lines in the $\delta - v$ domain. In other words, it has the potential of bringing down the 3D problem into 1D. However, it adds another layer of constraint on the relationship between the plane and the camera in order for it to come into full play. For general "V-disparity", it requires the plane to be parallel with the X axis of the camera coordinate system, while for plane to be projected vertically in the "V-disparity" image, the plane must be fronto-parallel with the image plane, that's a rarity in most computer vision applications. Instead of waiting for such a rarity to occur, the system actively relocates the camera virtually to achieve the effects as if the camera is fronto-parallel to the plane

under investigation. However, the transformation of the camera requires the knowledge of the camera-plane relation ($\Re_{cp}$) beforehand. Unfortunately, that relation $\Re_{cp}$ is precisely the unknown that is to know.

We now lay the doubt aside and observe the "V-disparity" images shown in Figure 5.6. It is evidently hard to extract the linear projections of all the stair surfaces from the real "V-disparity" image. Instead, we could extract the most reliable one to infer $\Re_{cp}$ or more precisely the camera pitch α, based on which a non-occluded virtual "V-disparity" image could be generated. The non-occluded "V-disparity" image has done away with all the occluded points and is much suitable for line extraction. In addition, we can narrow down the search for only vertical lines. The occluded points are subsequently added for searching of other projections. Bypassing the real "V-disparity" greatly reduces the chances of making false detections. This explains why the virtual "V-disparity" concept is necessary and important to solve our problem.

5.4 Summary

This chapter covers two topics. The first part sketches the state-of-the-art 3D stair localization algorithms. The second part intends to pave the way for the introduction of a 3D staircase localization algorithm in the next chapter. It introduces an important image concept known as “V-disparity” and gives succinct mathematical derivations. The concept states that planes with special geometrical relations to the camera can be projected as straight lines in the “V-disparity” space, which are well-suited to the problem of staircase localization. To further reduce the dimensionality of the problem domain, it then brings the virtual “V-disparity” concept into the picture. Finally, it discusses the merits of these concepts by putting them into a synthetic staircase scenario.

Chapter 6 3D Staircase Localization

6.1 Introduction

In the last chapter, we introduced a special plane-to-line projection property, which states that planes with special geometrical relations to the camera could be projected as straight lines in the $\delta - v$ domain. This chapter will be employing this feature of special planes to locate each plane on a staircase surface, rather than directly manipulating an ensemble of 3D data as its predecessors have done. To accelerate the detection speed and decrease the problem complexity, our 3D staircase detector is triggered only when a staircase is detected through monocular analysis and is operated on the reported region. This prior knowledge is a product provided by the 2D staircase detector. It is also a pre-requisite to use the "V-disparity" concept in the staircase localization stage.

Basically, our 3D staircase detector is trying to detect and locate every potential planar region within the reported domain of the scene. However, the potential planes are not randomly placed, but observe a strong geometrical relationship with each other. In another word, a staircase can be modeled as a series of mutual orthogonal tread and riser planes. This assumption should not lose generality on a typical staircase object. The algorithm that follows is based on the grounds herein assumed. Before we touch on the staircase localization problem, we first establish a formal and explicit definition of what the problem is or what the questions are that we will ascertain through investigation.

6.2 Problem definition

Figure 6.1 gives a side view and a vertical view of a synthesized staircase. The labeled parameters are the unknowns that are to be resolved at the end of the investigation. These parameters can be classified into 2 groups. The first group depicts the geometric relations between the camera and the staircase, including the camera pitch angle α, camera yaw angle β, the camera elevation above the last detected tread surface h and horizontal distance between camera and the nearest detected riser surface d. The second group of parameters defines the intrinsic geometric characters of the staircase, including the height of each detected riser surface $\{h_0, h_1, h_3, h_4\}$ and the depth of each detected tread surface $\{d_1, d_2, d_3\}$. For the yaw angle β, it is reasonable to assume it is within the range of $[-15°, +15°]$.

One thing worth noting here is the elevation of the camera above the last detected tread surface h is not the distance between the camera and the ground plane. It is due to the fact that the ground is not of interest to us, and the 2D staircase detector also excludes it from the searching domain. Concerning the camera to ground elevation, it could be estimated separately using similar techniques used here. That is beyond the scope of this thesis. Similarly d is not the actual horizontal distance between the camera and the staircase. Both h and d are subject to the results fed by the 2D staircase detector. This is acceptable as staircases are rarely completely visible. Under most circumstances, we could only perform analysis on a portion of it.

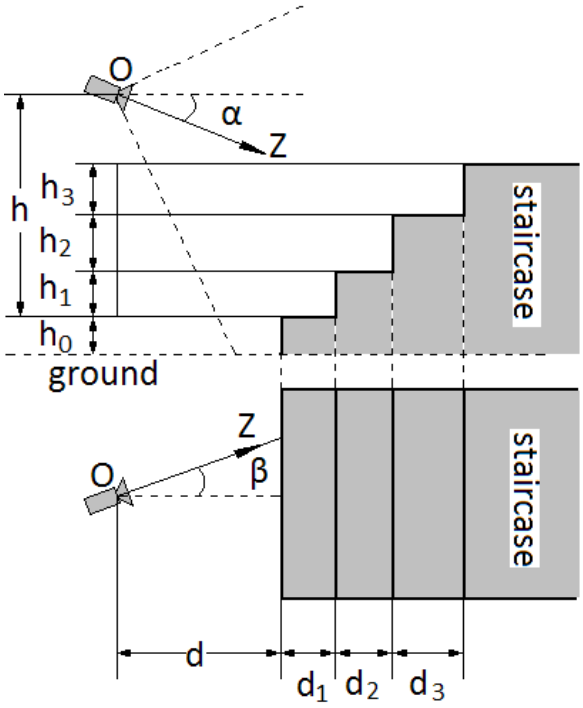

Figure 6.1: Problem definition

At the end, a 3D scene, including the camera and the detected portion of the staircase, will be reconstructed. Complete staircase model can be built through integrating results obtained from multiple views.

6.3 An overview of the proposed algorithm

The best way to describe a complex subject is to give a succinct overview of the matter before introducing the details. This section gives the reader a brief overview of the proposed algorithm. Our algorithm answers the problem posted in Section 6.2 step by step as illustrated in Figure 6.2.

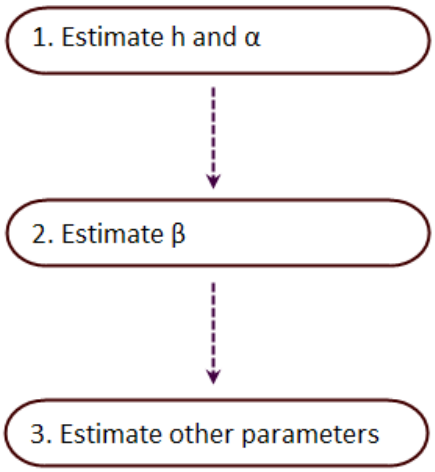

Figure 6.2: Flow of the algorithm

Firstly, the strongest line segment in the "V-disparity" image is searched. This line represents the projection of bottom tread surface since it is nearest to the camera. Based on the line, h and α are first estimated.

Secondly, the estimated h and α are then used to construct a virtual "V-disparity" image. The longest vertical line in the virtual "V-disparity" is then exploited to estimate β.

Finally, offset h, α and β construct the updated virtual "V-disparity" image. The image will then be partitioned. Each partition represents a staircase planar region. Back-project each partition into the original image to estimate their associated parameters, $\{h_i\}, \{d_i\}$ and d.

The main skeleton of the algorithm has been outlined above; more details about each step will be given in following sections.

6.4 Estimate h and α

Bottom tread plane and its projection in the "V-disparity" image play a key role to our proposed staircase localization algorithm. Precise extraction of the bottom tread projection in the "V-disparity" image provides an entry point to the solution. A few techniques have been applied in order to estimate the parameters h and α as accurate as possible.

6.4.1 Searching for bottom tread plane

The first step of our algorithm is to build a "V-disparity" image through the dense disparity map provided by Bumblebee.

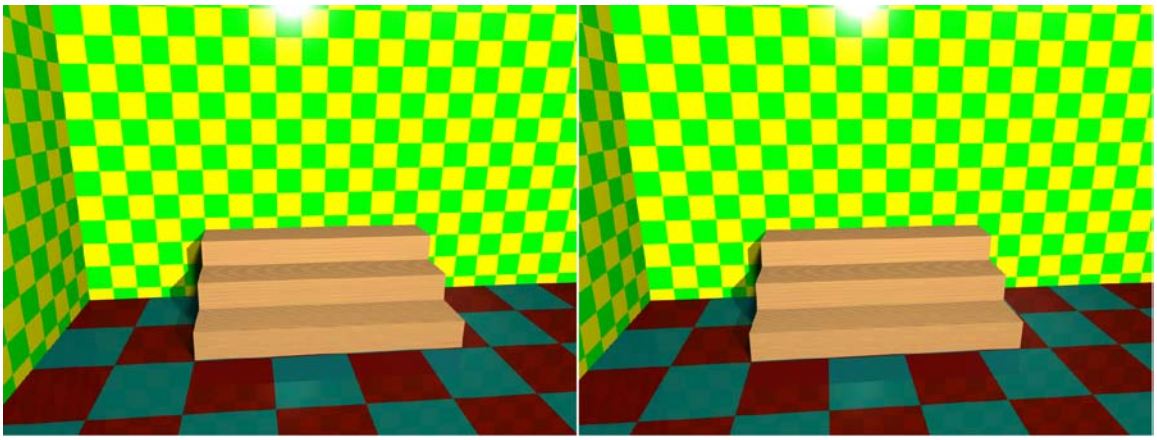

Figure 6.3: Synthetic stereo images with a staircase instance in the scene

To illustrate the proposed algorithm, a pair of synthetic stereo images shown in Figure 6.3 with the ground truth provided at Table 6.1 is referenced throughout the entire explanation.

Parameters	Value
Image size (pixel)	1024 x 768
Image center (pixel)	512 x 384
Focal length (pixel)	768
Base line (meter)	0.12
Pitch angle (degree)	15.00
Yaw angle (degree)	5.00
Camera ground distance (meter)	1.60
Step depth (meter)	0.33
Step height (meter)	0.20
Camera stair distance (meter)	2.51

Table 6.1: Ground truth table for the synthetic stereo pair

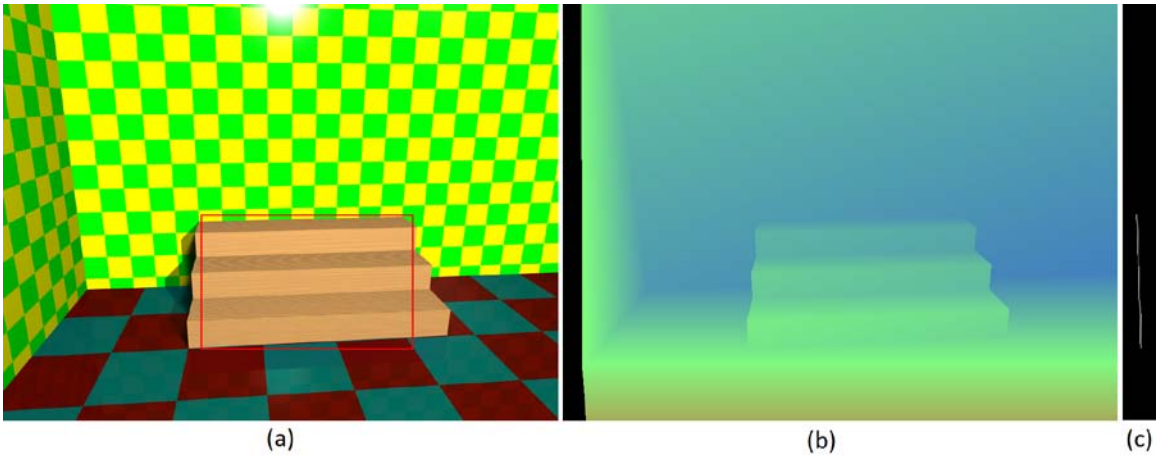

Figure 6.4: (a) detected staircase region (b) dense disparity image (c) partial "V-disparity" image

Figure 6.4(a) highlights the detected staircase region from monocular analysis on the left stereo image. A dense disparity image is shown in Figure 6.4 (b), and the non-scaled "V-disparity" image is shown in Figure 6.4 (c). Note, herein the "V-disparity" image is

constructed over a sub-region on the original image, namely, the detected staircase region marked in (a). As most V-disparity-based ground estimation algorithms have done, our algorithm also searches for the strongest line through the partial "V-disparity" image to locate the bottom tread plane. Without doubt, the line must slant to the left to indicate that it is a projection of a horizontal or near horizontal plane.

Nevertheless, when we search for bottom tread rather than ground plan from the "V-disparity" image, new problems arise.

Firstly, a tread surface usually occupies a small region in the image, while the ground plane normally takes up half of the image. Therefore, to utilize the plane-to-line projective property, we must make sure that the projection of the bottom tread must be strong enough to be sensed by generic line extraction algorithms, such as Hough Transform. In view of this limitation, we have used a relatively large image size in the experiment: 1024x768. In the first part of the thesis, we mentioned that image size shows no obvious impact on the speed of the 2D staircase detector. In addition, we also defined a lower bound for the scanning window to be 0.1 of the input image size. With this constraint, the minimum window passed for 3D investigation almost approaches 100x100, and this circumstance rarely happens.

Secondly, it's not easy to extract the projection of bottom tread plane from the original "V-disparity" image shown in Figure 6.4(c). The entire staircase surface projects in a small clustered range along the horizontal axis ($\Delta-$ axis) and fluctuating along a straight line. The zigzag profile is not resolvable and is reduced to a line. If Hough transform is

applied, this interfering line, rather than the bottom tread projection, is more likely be detected. Analytically, if the disparities are resolved to 1-pixel, it may have some difficulty in distinguishing the projections from various planes. The software package shipped with Bumblebee offers sub-pixel interpolation. Thus, we can multiply the sub-pixel disparity values by a certain constant scaling factor s_0 such that the disparities span a suitable range ψ. Thanks to the sub-pixel interpolation provided by Bumblebee, the stretched "V-disparity" image becomes more accurate rather than indistinct. The determination of scaling factor s_0 is illustrated in Table 6.2.

- Obtain the detected sub-window height ΔV and set $\psi = \Delta V$
- Create the original "V-disparity" image Iv ($s_0 = 1$)
- Find maximum intensity $I_{\max}$ in Iv
- Thresholding:

$$T = \mu I_{\max}$$

$$I(x,y) = \begin{cases} I(x,y) & if\ I(x,y) > T \\ 0 & if\ I(x,y) < T \end{cases} \qquad (6.1)$$

- Remove isolated pixels in Iv
- Find maximum disparity $\delta_{\max}$ and minimum disparity $\delta_{\min}$ in Iv
- Calculate:

$$s_0 = \frac{\psi}{\delta_{\max} - \delta_{\min}}$$

$$s_0 = MAX(1, s_0) \qquad (6.2)$$

Table 6.2: Determine s_0

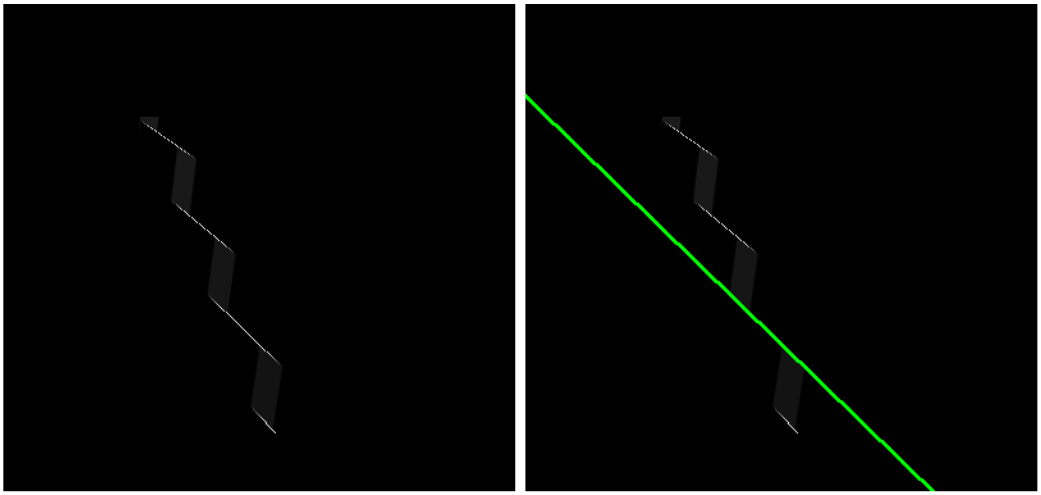

Figure 6.5: (a) Adaptive scaled "V-disparity" image (b) line extracted

Followed by the determination of s_0, the new "V-disparity" image can be immediately created as shown in Figure 6.5. Images have been clipped for better fit to the page. Line extraction algorithms then proceed on the adaptive scaled "V-disparity" image. To further reinforce the extraction accuracy, we could restrict the ranges of h and α. Generally, α is set within $[0°,45°]$, while the extent of h is subject to particular applications. More rigidly defined h and α give more accurate extraction results.

Figure 6.5(b) displays the strongest line extracted (green) using weighted Hough transform.

$$\frac{h}{bs_0}\delta = f\sin\alpha + (v - v_0)\cos\alpha \qquad (6.3)$$

Equation 6.3, which takes account of the scaling factor s_0, is a new version of Equation 5.9. With Equation 6.3 and the detected line from the "V-disparity" image, an initial estimation of h and α can be obtained.

However, the plausible result obtained from Hough transform does not necessarily correspond to the most representative one. Sappa et.al. [68] have noticed there is a nonlinear mapping between disparity and depth. It claims that less than 25% of disparity values are used for representing more than 70% of depth values, while almost half of disparity values are used for representing shorter distances, ranging about 11% of depth values. This nonlinear mapping makes the quality of the "V-disparity" image, and hence the accuracy of the line extracted, highly dependent on the distance between the camera and the target. Therefore, the initial estimation needs to be fine-tuned. Due to such inherent drawback of the "V-disparity" representation, further adjustments of h and α could be difficult if only the "V-disparity" image is used. The next section presents the optimization of h and α, performed directly in the 3D Euclidean space.

6.4.2 Optimize h and α

The optimization starts from the extracted line, denoted as l_0 in the adaptive scaled "V-disparity" image shown in Figure 6.6. The basic principle lies in a plane fitting process to a subset of 3D points projected onto l_0. Compared with "V-disparity" space analysis, plane fitting produces more accurate results but an obvious disadvantage of this sort of

algorithms is that they take much longer time. In order to reduce the fitting cost to minimum, only a subset of the point collection is used for plane fitting. Details are given below:

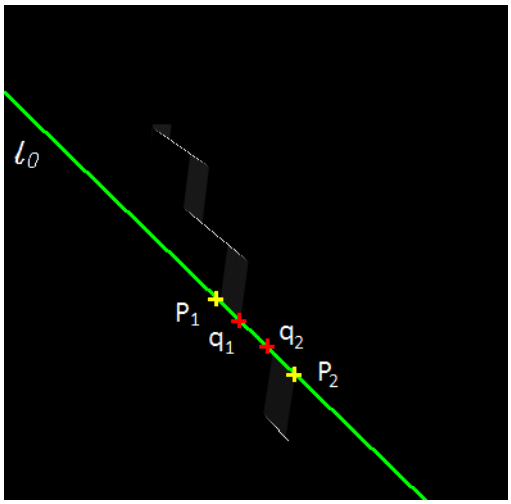

Figure 6.6: Optimize h and α

Firstly, search along line l_0 for the largest line segment, represented as p_1p_2 in Figure 6.6, where small gaps are filled. Meanwhile, the maximum intensity $I_{\max}$ along p_1p_2 is noted. Next, divide p_1p_2 into 3 equal segments and obtain segment q_1q_2 as illustrated in the figure. Lateral segments p_1q_1 and p_2q_2 are more likely to be the overlapping projective areas of neighboring tread and riser surfaces. To mitigate the influence of outliers from neighboring riser surfaces, we select the central segment q_1q_2. After that, for each row v_i lies on q_1q_2, determine an acceptable disparity range $[\delta_1, \delta_2]$ around δ_i on line l_0, while searching within the candidate range $[\delta_i - s_0, \delta_i + s_0]$ such that $I(\delta_1, v_i) \geq \mu I_{\max}$ and $I(\delta_2, v_i) \geq \mu I_{\max}$.

Once we obtain the range $[\delta_1, \delta_2]$ at v_i, add points that project to this range into a 3D point collection. Eventually, we will obtain a set of 3D points available for plane fitting. Plane fitting [69], [70] is an old but appealing topic of research, with a lot of algorithms proposed historically. It has a close relation with range data segmentation, a concept that has been mentioned in Section 5.2. In this project, a simple least square fitting technique is applied for finding the best-fitting plane to the given set of points. A better estimation of h and α could be achieved through the fitted plane.

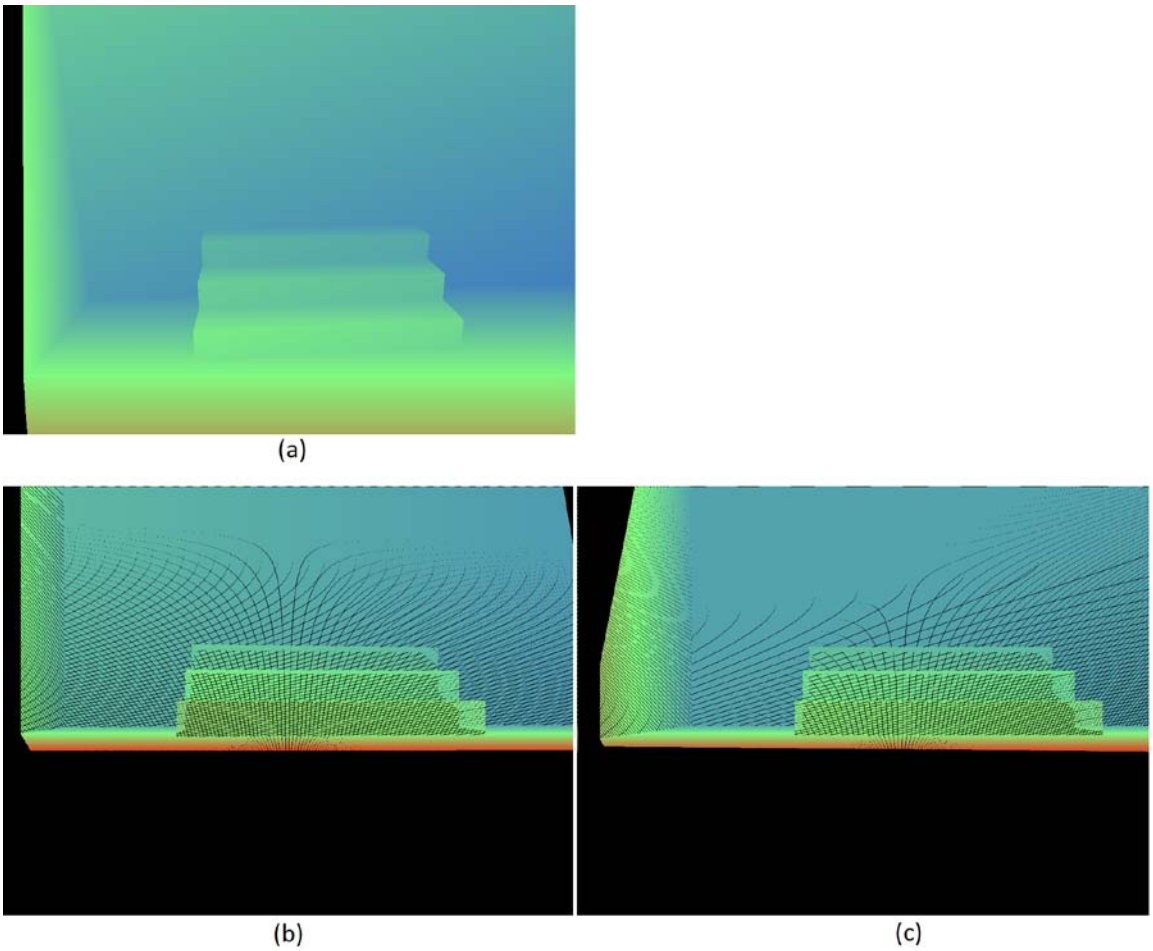

Figure 6.7: Disparity images

6.5 Estimate β

Once the parameters h and α are optimized, the algorithm continues estimating another camera pose parameter β, using the same least square plane fitting technique exploiting the "V-disparity" image. However, this time the virtual "V-disparity" image comes into play.

Given h and α, the virtual observing point would then be deduced, such that $h' = 0$ and $\alpha' = 0$, where h' is the distance between the bottom tread surface and the virtual observing point and α' is the virtual pitch angle.

After a homogeneous transformation, a virtual disparity map will be deduced as shown in Figure 6.7(b). The map shows that the camera is lying as if along the ground. That is precisely the effect we expect. Based on the virtual disparity map, a virtual non-occluded "V-disparity" image can be derived shown in Figure 6.8. Similarly, an adaptive scaling factor s_1 is applied to the disparities in order to stretch the image along the $\Delta-$axis. s_1 is determined using the same technique described in Table 6.2. However, a constant disparity stretch ($\psi = 64$) is used instread of a varying ΔV. Generally, ΔV is much larger than 64, this explains why the "V-disparity" images in Figure 6.5 are much wider than those shown in Figure 6.8. Note the images have done away the top and bottom null regions.

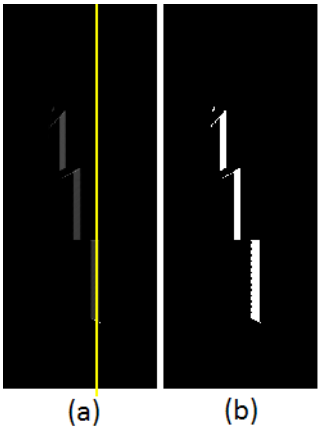

Figure 6.8: Virtual "V-disparity" images after offsetting h and α (a) vertical line extracted (b) binary thresholded virtual "V-disparity" image

Observing Figure 6.8, it shows several clusters of points, each compactly aligned along a vertical line. Apparently, the clusters represent the projections of the riser planes as treads are occluded at the virtual observing point. Under the virtual camera placement, a set of vertical line segments would dominate the "V-disparity" image if the disparities are well parsed. However, if the span ψ is not wide enough, clusters correspond to smaller disparities will conjoin each other; while larger ψ will diffuse the vertical projections and pose challenges to line detection. On the other hand, if yaw angle β is too large, it will cause severe diffusions no matter how large ψ is. However, we only consider a small yaw angle ($\beta \in [-15°,+15°]$) that does not cause a severe diffusion problem.

The interest here is not to detect all the vertical lines but the most representative one as shown in Figure 6.8 (a). Non-occluded "V-disparity" image is used to eliminate the hassle of invisible tread pixels. Generally, set $\psi = 64$ is a good trade-off from a pragmatic perspective. A standard Hough transform is used for the line extraction. Once

we have obtained the most representative vertical line, the remaining process is the same as used in the optimization of h and α. A least square plane fitting is applied to a set of points that project to a reliable segment along the detected vertical line. It would then be a simple task to determine β given the 3D representation of the riser plane in the virtual world coordinate system VCS ($X'Y'Z'$) as shown in Figure 6.9.

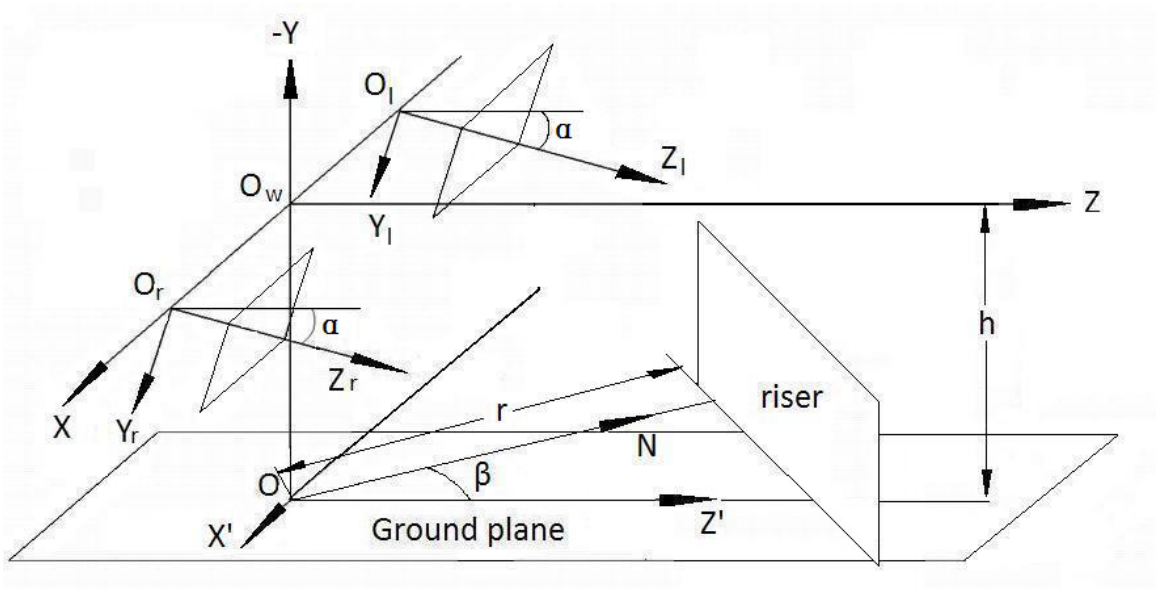

Figure 6.9: Virtual camera and riser plane

6.6 Estimate remaining parameters

After obtaining the parameters h, α and β, our journey is half done. Again, our solution continues the exploration of the virtual "V-disparity" images created with the most updated h, α and β. A virtual disparity image is shown in Figure 6.7 (c) after further offsetting the yaw angle β. The virtual non-occluded "V-disparity" image and

occluded “V-disparity” image now become better looking than ever before. They have slimmer vertical linear segments and horizontal bottom tread projections. Next, what we need to do is partitioning the occluded “V-disparity” image, and establishing a correspondence between each partition and a planar region on the staircase surface.

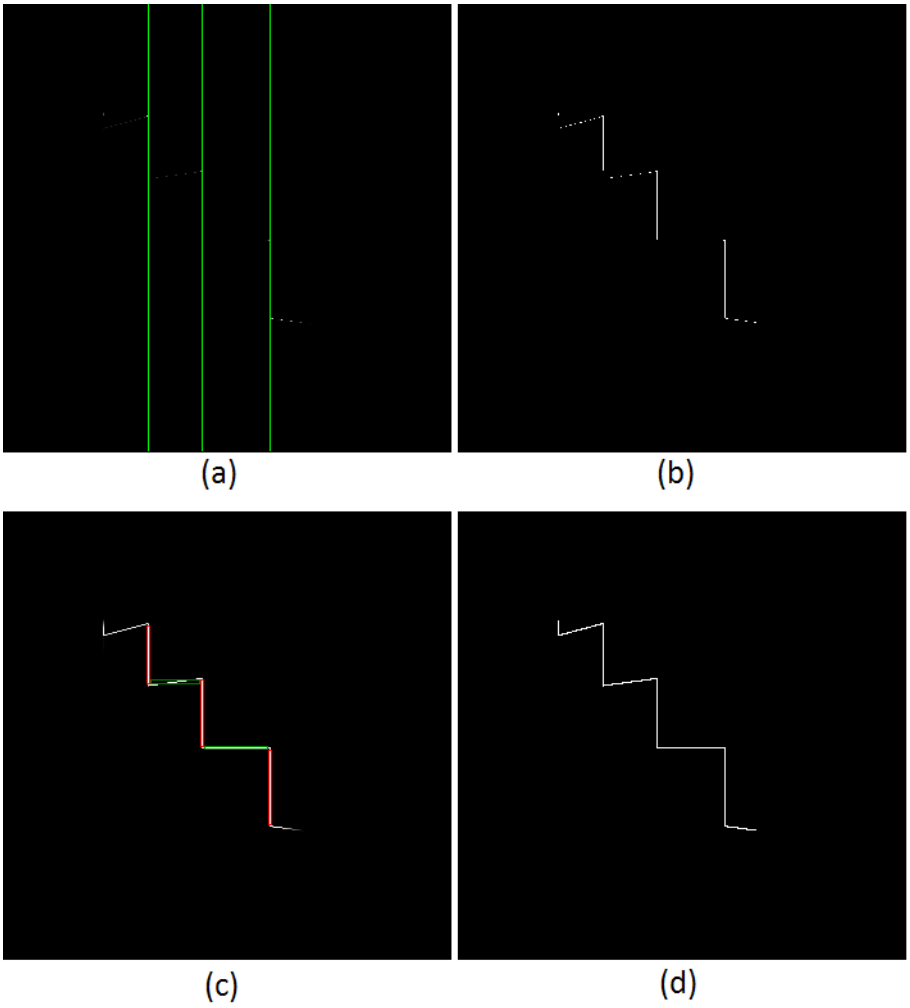

Figure 6.10: (a)Non-occluded “V-disparity” image (b) binary thresholded non-occluded “V-disparity” image (c)occluded “V-disparity” image (d) binary thresholded occluded “V-disparity” image

As usual, we need an adaptive scaling factor s_2 to spread out the disparities suited for line extraction. Then, both occluded and non-occluded "V-disparity" images are constructed at scale s_2 as shown in Figure 6.10(b), (d).

6.6.1 Partition Virtual "V-disparity" image

In the non-occluded virtual "V-disparity" image, most tread pixels are excluded, leaving a set of disconnected vertical line segments correspond to the risers. Hough transform is applied in the image for the detection of vertical lines. Outliers could be removed by checking the inter-line distance, using Equation 6.4:

$$d = \frac{bf}{\delta} \tag{6.4}$$

A suitable Euclidean distance between two vertical lines should be greater than 15cm in most of the staircases. This constraint should remove all the outliers and ensure the neighboring line segments span a reasonable distance.

Once the vertical lines projected from various risers are acquired, the invisible tread pixels are then taken into consideration in the occluded "V-disparity" image. Each vertical line is grown in either direction, until a sudden decline in the number of nonzero pixels it passes through occurs. After the outward expansion, we obtain the lateral boundaries for the projected area of each riser and 2 corners. With this information, we achieve a partition result as shown in Figure 6.11. Each partition is defined by a rectangle and the type of stair surface that it belongs to.

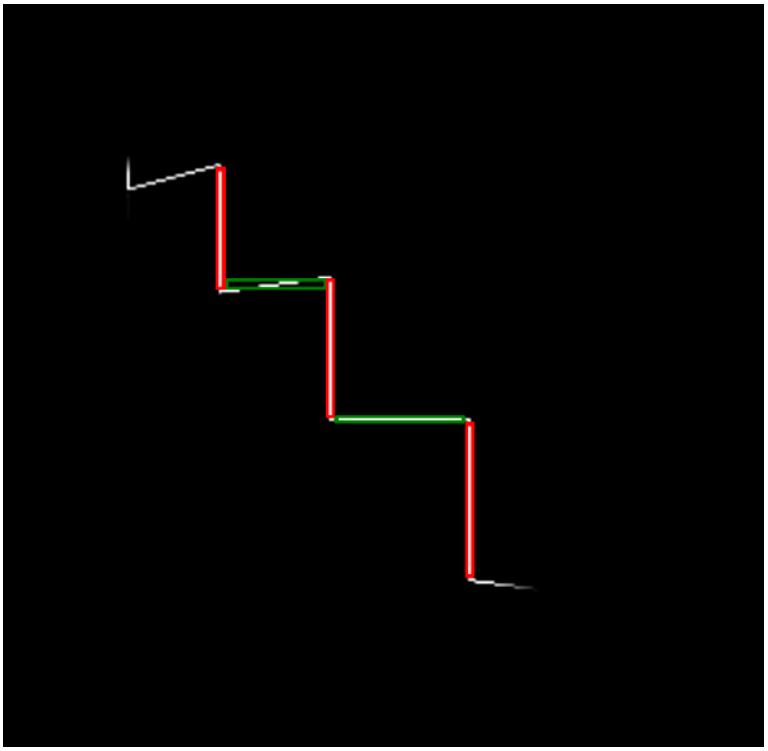

Figure 6.11: Partition the occluded “V-disparity” image

Note that some pixels are not included in either type of rectangles because they lie in a dual identity region. 3D points that projected into these regions can originate from either the tread or the riser surface. It will destabilize the system if these pixels are considered. On top of that, it’s also unnecessary to consider these pixels due to the fact that our partition is aimed at estimating the 3D location of each plane rather than a precise pixel classification on the original image as long as we have enough pixels that fall into each partition to proceed.

6.6.2 Back-projection onto original image

Each partition represents part of a planar region on the staircase surface. Next, we'll derive the geometric relation of each plane with the virtual camera, using Equation 6.4 and 6.5, which are obtained by substituting $\alpha = 0$ into Equation 5.11 and 5.9, respectively.

$$h = \frac{(v - v_0)b}{\delta} \tag{6.5}$$

Each partition π_i is back-projected onto the original image to obtain a set of valid pixels $P_i = \{p_1, p_1, \cdots, p_n\}$ that the partition represents. For a riser partition, median disparity δ'_{med} is figured out and substituted into Equation 6.4 to calculate the virtual median distance d'_{med} of the riser plane to the virtual camera point. For a tread partition, use Equation 6.5 to calculate the height at each pixel $h'_j(p_j) \in H$ and a median height h'_{med} is taken as the virtual elevation of the tread. A simple translation from the virtual observing point to the real camera position gives all the parameters: $d'_{med} \rightarrow d_i$, $h'_{med} \rightarrow h_i$.

Up to now, we have estimated all parameters as required in Section 6.2.

6.7 Experiments

This section presents experiments on 2 staircases. As space is limited, we'll not provide a step-by-step visual demonstration for every frame in each scenario. For each staircase, visual demonstration is provided for only one typical scene for illustration purpose.

6.7.1 Staircase scenario 1

For each staircase, there is an optimal test environment suited to that particular staircase. For the staircase shown in Figure 6.12, the optimal parameter settings are shown in Table 6.3.

Parameters	Value
Δw	0.02
$Xshift$	0.02
$Yshift$	0.005
Stages	16

Table 6.3: Optimal parameter setting, scenario 1

Figure 6.12 – 6.16 take down the workflow of the staircase localization algorithm. Full description will not be provided since most of these figures have been explained in previous sections. The bottom right sub-figure of Figure 6.12 is the thresholded "V-disparity" image with scaling factor s_0. The green line is the weighted Hough transform

detected line as aforementioned, while the yellow line is the projection of the fitted plane when optimizing h and α.

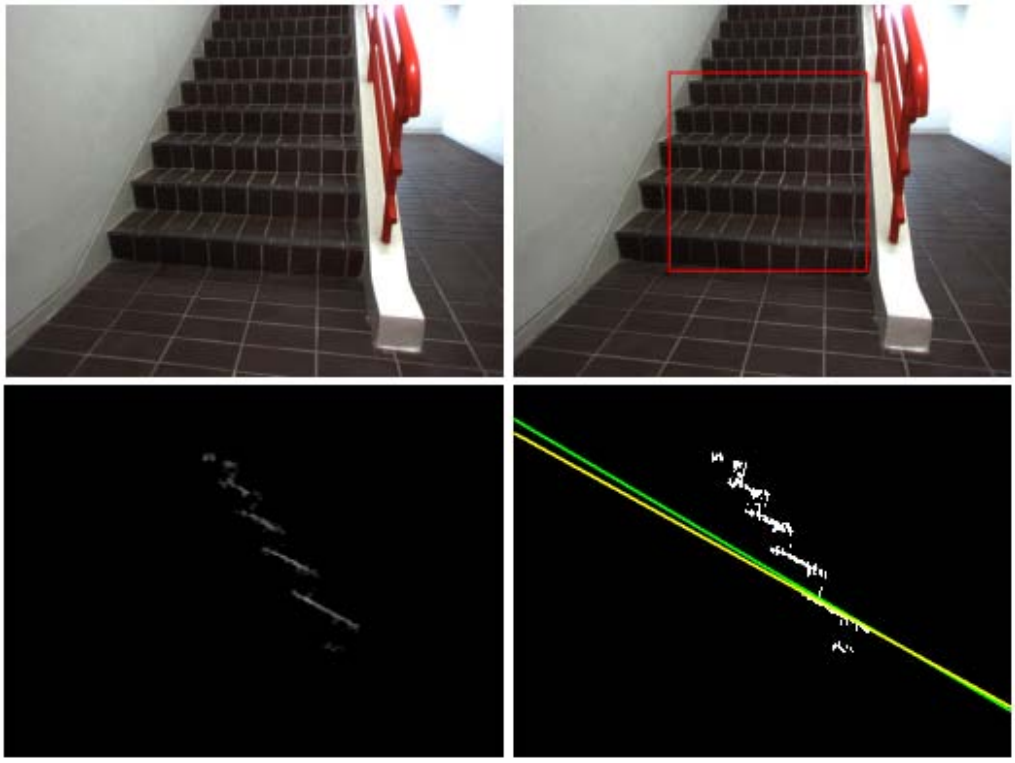

Figure 6.12: Staircase image and "V-disparity" image (s_0), scenario 1

Table 6.4 lists the final experimental data and the values of the 3 adaptive scaling factors: s_1, s_2 and s_3 used for this particular scene. The values of the scaling factors are very important to the final localization results.

Generally, the data presented in Table 6.4 are satisfactory. If the camera is well-placed (roll angle $\gamma \to 0$, $\beta \in [-15°, +15°]$) and the test parameters for the 2D scan are appropriately set (Table 6.3), the algorithm can always accurately recover the pose and dimensions of the staircase in the scene.

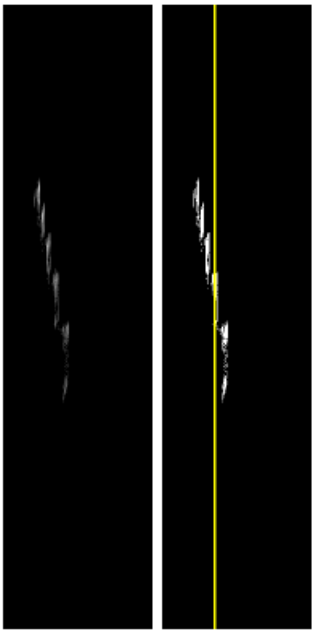

Figure 6.13: Virtual non-occluded "V-disparity" image (s_1), scenario 1

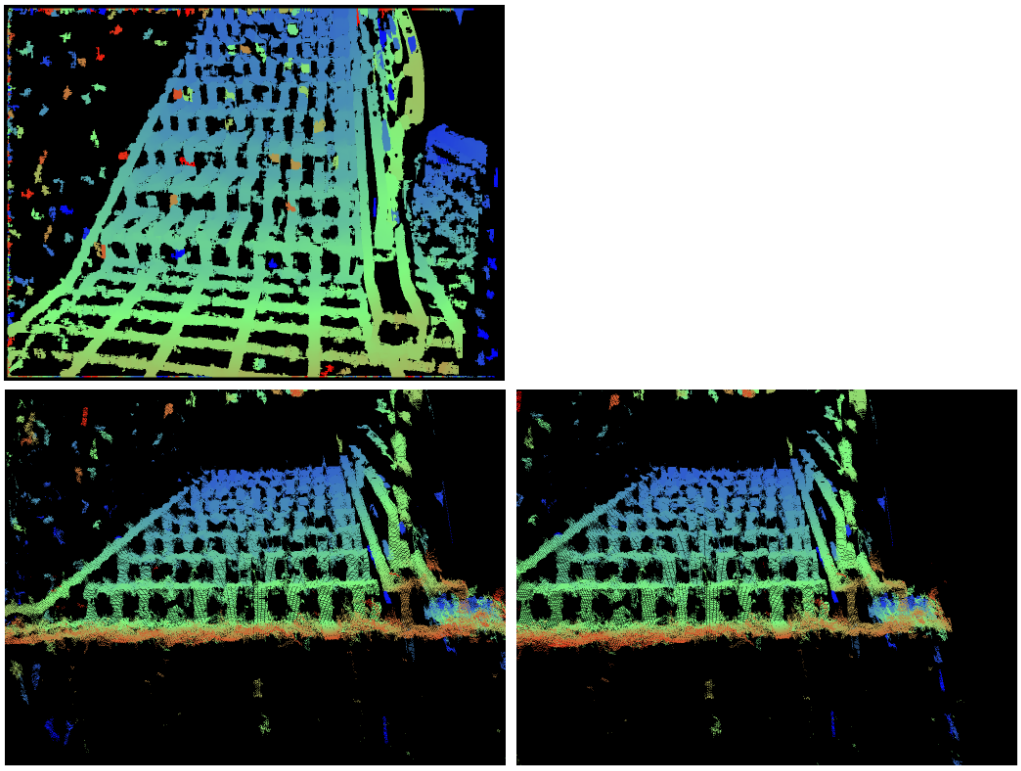

Figure 6.14: Disparity images, scenario 1

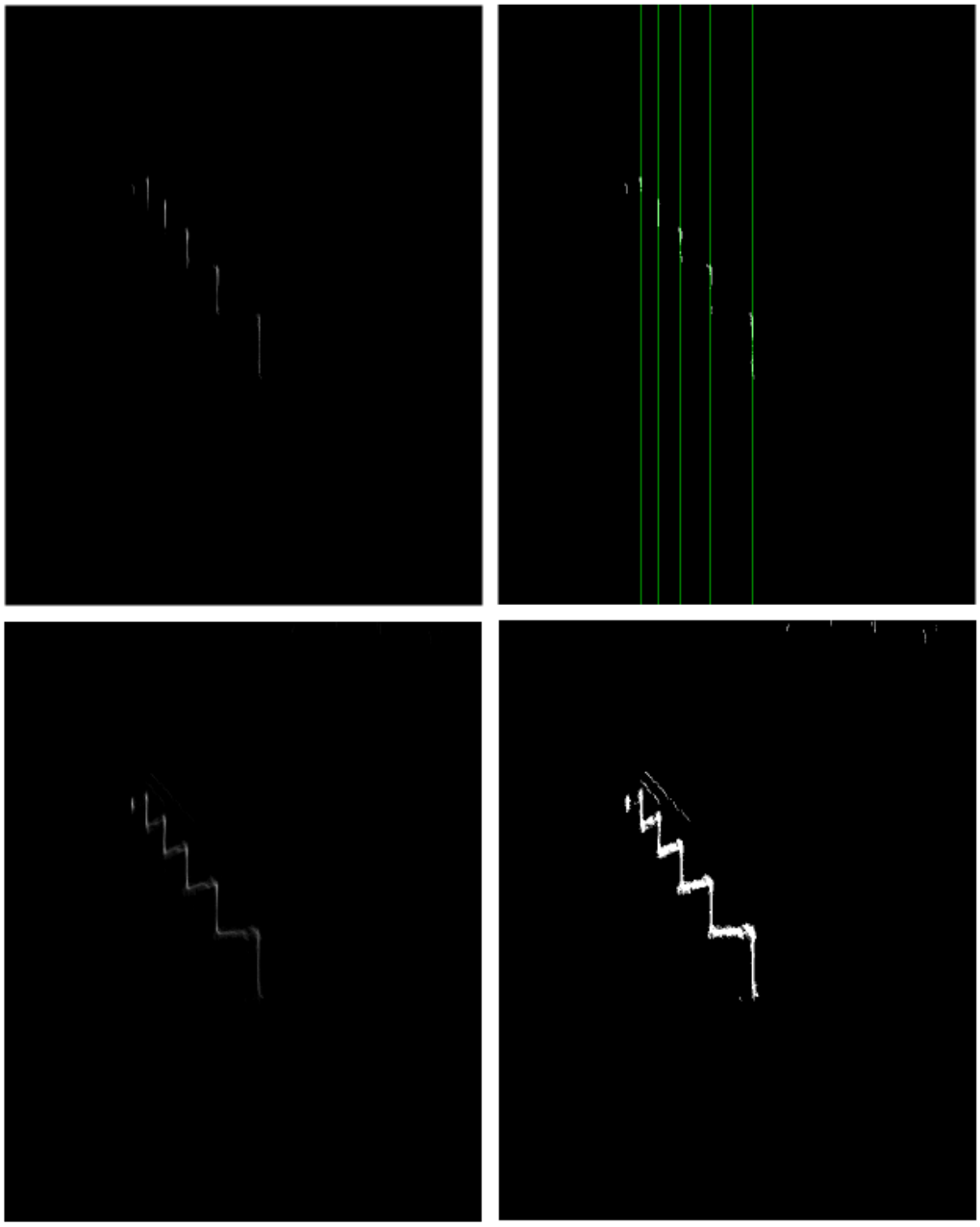

Figure 6.15: Non-occluded and occluded "V-disparity" image(s_2), scenario 1

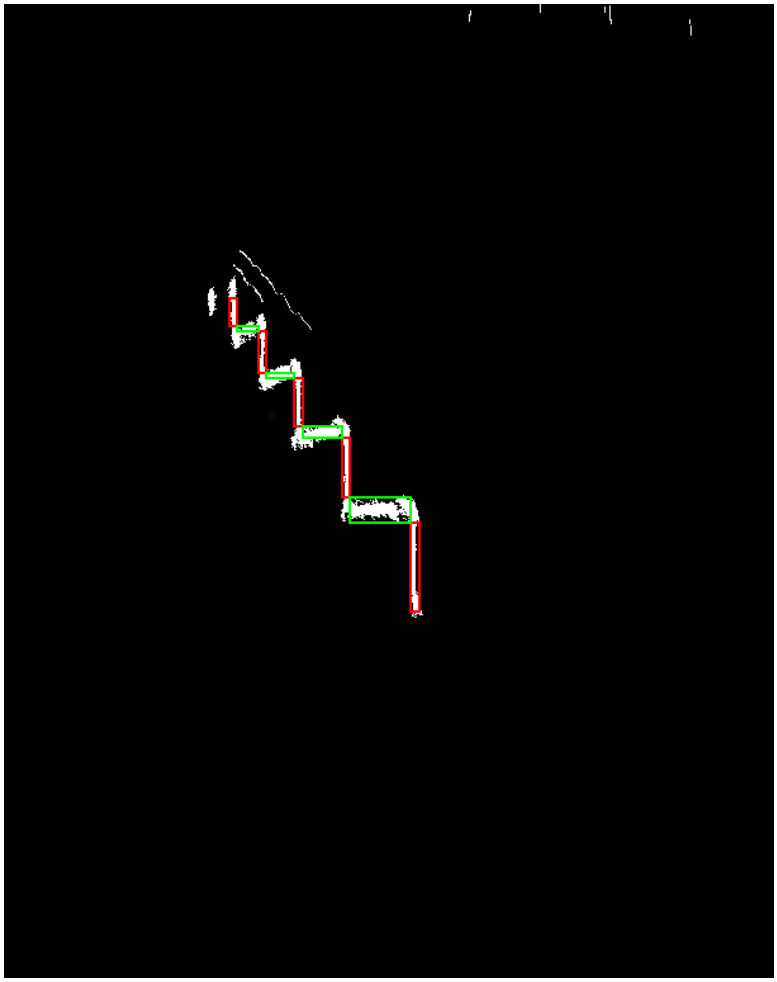

Figure 6.16: Partition on occluded "V-disparity" image, scenario 1

Scaling factor	s_0	s_1	s_2		
	16.366	2.216	15.422		
Camera geometry		h/m	$\alpha/\deg$	$\beta/\deg$	d/m
	Initial	0.834	22.85	/	/
	Final	0.798	21.01	9.05	1.487
Staircase dimensions	**Height**	h_1/m	h_2/m	h_3/m	h_4/m
		0.156	0.152	0.149	/
	Width	d_1/m	d_2/m	d_3/m	d_4/m
		0.292	0.291	0.291	0.289

Table 6.4: Experimental data, scenario 1

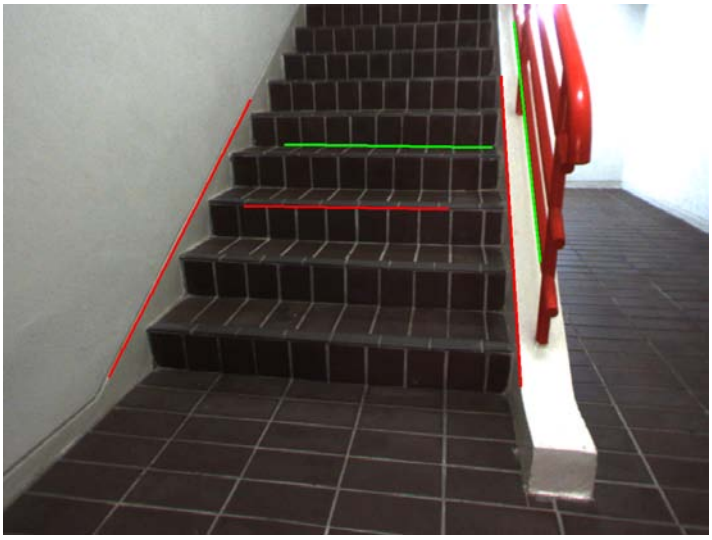

Figure 6.17: Range weighted Hough transform (RWHT) scenario 1

Figure 6.17 gives the line detection results of RWHT. Convex lines in red. The staircase is left undetected due to limited number of lines extracted in each set.

6.7.2 Staircase scenario 2

Scenario 2 tests another staircase. Micro-textures bestrewed the surface of the staircase, which helps the Bumblebee generate a much denser disparity map as shown in Figure 6.20. During the experiment, we have found that staircases which possess sufficient textures only need a sparse scanning scheme to achieve equivalent results as shown in Table 6.5.

Parameters	Value
Δw	0.04
Xshift	0.04
Yshift	0.01
Stages	16

Table 6.5: Optimal parameter setting, scenario 2

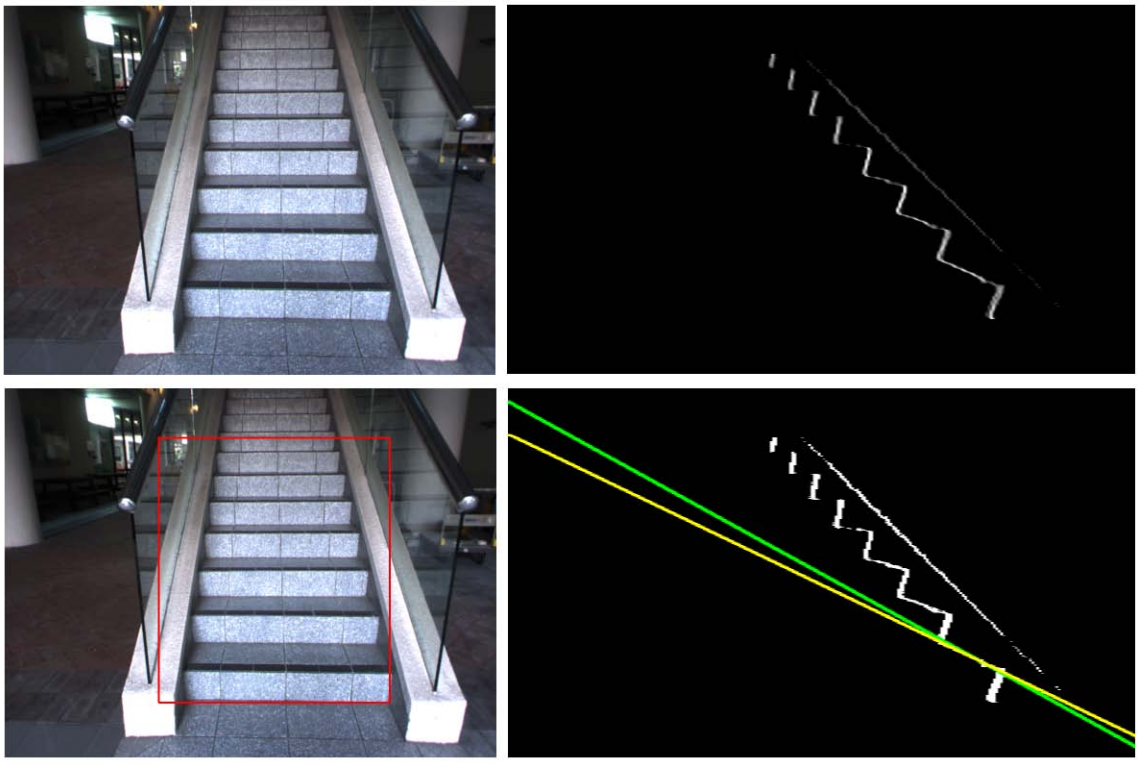

Figure 6.18: Staircase image and "V-disparity" image (s_0), scenario 2

In the “V-disparity” image shown in Figure 6.18, the longest segment is projected by the parapets’ top surface. The surfaces form an oblique plane in the WCS. If the plane is viewed as a virtual ground plane as shown in Figure 5.3, the new pitch angle can be

represented as: $\alpha' = \alpha + \theta$, where α is the real pitch angle and θ is the slanting angle between the parapet surface and the real ground plane. Normally, the slope θ of staircases approximate $45°$, which makes α' beyond the predefined range $[0°,45°]$ and leaving the longest projection undetected. Another factor that could cause the line to remain undetected is its low weights (intensities) compared with staircase planar region projections.

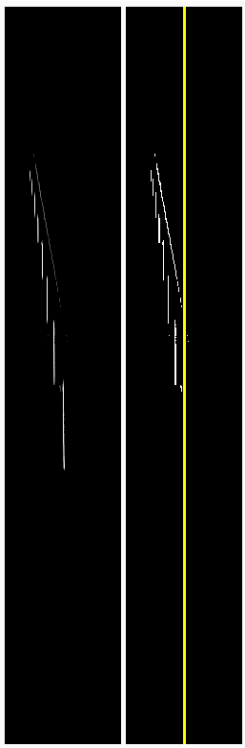

Figure 6.19: Virtual non-occluded "V-disparity" image (s_1), scenario 2

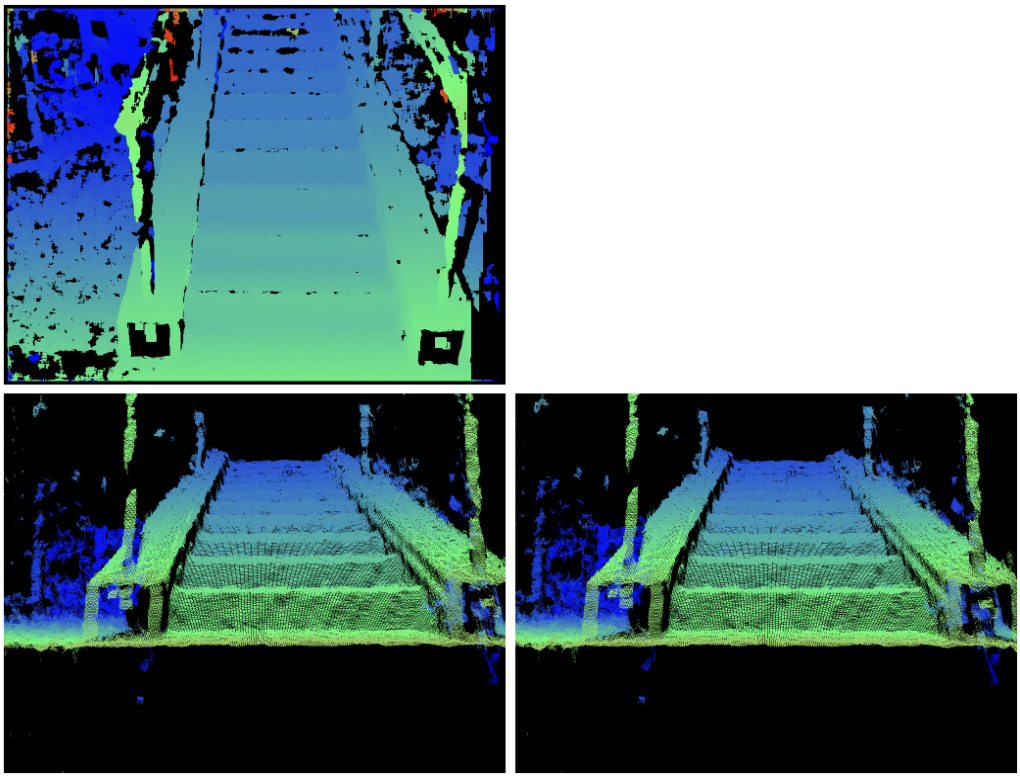

Figure 6.20: Disparity images, scenario 2

In the case of small yaw angles ($\beta \to 0$) as in the current scene, strong edges are almost parallel to the epipolar lines. It is well-known that in such a condition, window-based correlation stereo may compute incorrect disparities due to lack of gradient variations along the epipolar line. Therefore, artificial and null patches often occur on the disparity images as shown in Figure 6.14. However, this phenomenon does not occur in Figure 6.20 due to clouds of textures bestrewed on the staircase surface. With the highly accurate disparities, all the "V-disparity" images generated here are of high quality. They have slimmer projections and fewer noises. One thing worth noting is that, even though there are some incorrectly estimated disparities in the disparity map, it does not affect the result as long as majority of the disparity values are correctly estimated. This is because both "V-disparity" and Hough transform are in nature

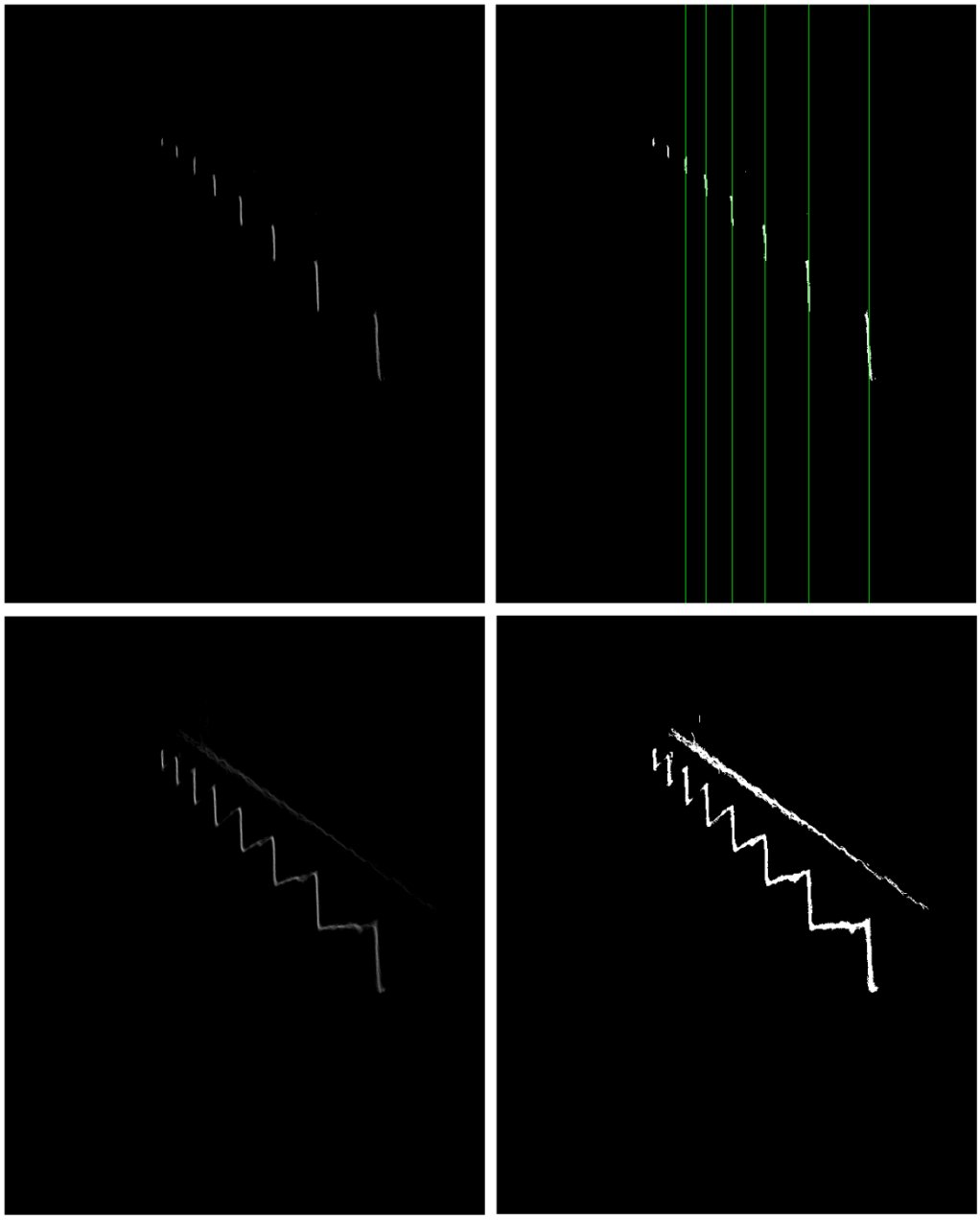

Figure 6.21: Non-occluded and occluded "V-disparity" image(s_2), scenario 2

majority voting schemes. Incorrectly estimated disparities are simply suppressed in the line extraction process.

The system runs at an average speed of 0.774 second per frame on our Pentium IV, 2.66 GHz PC with image size 1,024 x 768. Depending on the size of the reported region, the processing time roughly varies from 0.5 second to 1.0 second.

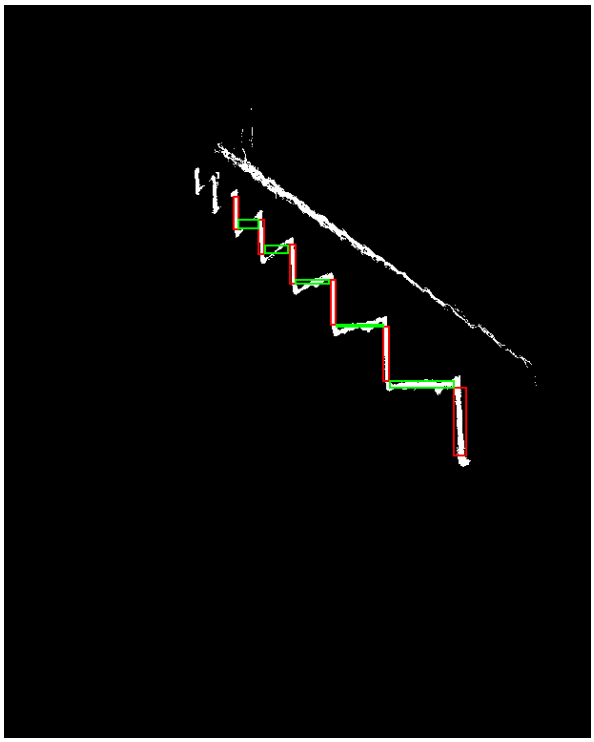

Figure 6.22: Partition on occluded "V-disparity" image, scenario 2

Scaling factor	s_0	s_1	s_2			
	22.115	1.828	15.521			
Camera geometry		h/m	$\alpha/\deg$	$\beta/\deg$	d/m	
	Initial	1.190	25.384	/	/	
	Final	1.071	21.200	1.250	1.520	
Staircase dimensions	**Height**	h_1/m	h_2/m	h_3/m	h_4/m	h_5/m
		0.162	0.162	0.163	0.169	/
	Width	d_1/m	d_2/m	d_3/m	d_4/m	d_5/m
		0.292	0.294	0.292	0.290	0.303

Table 6.6: Experimental data, scenario 2

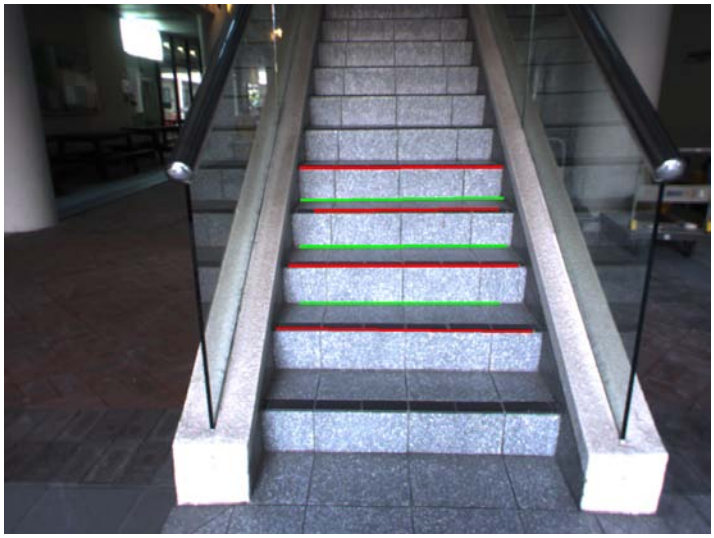

Figure 6.23: Range weighted Hough transform (RWHT) scenario 2

$$convex: \quad 0.01x - 0.64y - 0.76z + 1.674 = 0; \tag{6.6}$$

$$concave: \quad 0.01x - 0.64y - 0.77z + 1.815 = 0 \tag{6.7}$$

Figure 6.23 gives the RWHT results and the tangent planes are expressed in Equations 6.6 and 6.7 respectively for convex and concave lines, in the camera coordinate system. Figure 6.24 projects the tangent planes onto the staircase reconstructed with full parameterization given by the proposed localization algorithm. The results of both methods converge well.

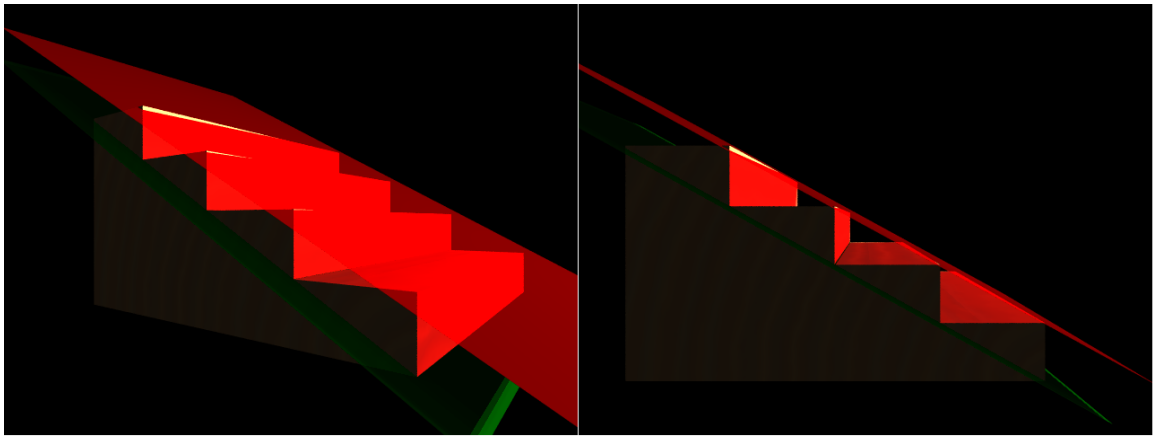

Figure 6.24: Staircase reconstruction

6.7.3 Discussions

Due to limited time and resources, this thesis only presents 2 staircase localization scenarios. Scarce of experimental data may barricade our way of making an ultimate decision about which algorithm is better, but it still shows Lu and Manduchi ‘s method [52] could not solve the staircase detection problem very well.

Lu [52] starts his solution from edge pixels of the intensity image, a common ground for all types of line-based stair detection algorithms [11], [12], [13], [14], augmented with range data. The inclusion of range information effectively removes a substantial amount of false positives, meanwhile stricter verification lows down the detection rate. Referring to the detection rate figures listed in Table 4.6, it is reasonable to deduce that [52] has a much worse detection rate compared with the proposed algorithm.

Another finding through experiment is that the curvature index expressed in Equation 5.1 often provides non-satisfactory results. It can be attributed to the quality of range data and the nonlinear mapping effect of range images (Section 6.4.1). Curvature indices at points nearer to camera are often underestimated, reducing the chance of nearer stair edges to be detected.

Concerning the detection accuracy, both methods present satisfactory results if staircase is detected. A major shortcoming of the proposed algorithm is that it works only in a range of $[-15°,+15°]$. This is a result of the boosted classifier, which will lose its competence beyond this range. The operating range figures are estimated through experiment, and may not strictly apply to all cases.

All in all, the proposed technique should represent the standout within all existing staircase detection and localization algorithms.

6.8 Summary

This chapter proposes a novel 3D staircase localization algorithm based on “V-disparity” concept. The concept applies to every planar region on the staircase surface. However, due to inherent limitations lie in the formulation of the “V-disparity” image, it is hard to identify the linear projections for every plane directly from the original “V-disparity” image. A few improvements have been made to make the “V-disparity” concept more suitable for the task of staircase localization:

- The “V-disparity” images used in the experiment are all adaptive scaled to make it more suitable for the task of line extraction.
- Virtual “V-disparity” images are used as if cameras are transformed to a virtual observing point such that all risers are fronto-parallel planes and project as vertical lines in the “V-disparity” image.
- Plane fitting is used to improve the accuracy of parameter estimation.

Experiments on 2 different staircases are presented at the end of the chapter to demonstrate the performance of the algorithm.

Chapter 7 Conclusions

7.1 Conclusions

This project presents a novel vision-based staircase identification system. The system is deliberately divided into 2 sub-tasks: 2D staircase detection followed by 3D staircase localization. Due to high computational cost of 3D data processing, the 2D staircase detector is first used to preview every input frame at a rapid rate. The 3D localization system is triggered only when a staircase is detected through 2D analysis.

The 2D detector is based on the Viola-Jones rapid object detection framework. A small set of PCA-based Haar-like features are introduced into training. The AdaBoost algorithm firstly selects these user-designed features which are extremely efficient at rejecting negative samples at early stages. The PCA-based features also set up a generalized guideline for Haar-like feature design suitable for the detection of any other objects. Scanning and integration schemes are redesigned for staircase detection. A new evaluation metric is introduced to assess the accuracy of every detection rather than Boolean classifying it. Eventually, we obtained a 29-layer cascade of classifiers, involving 655 Haar-like features. The final detector does not rely on the detection of concurrent lines such that it greatly mitigates the annoying issue of large false alarm rate for all line-base staircase detection algorithms. Experiment on 311 negative test images reports a zero false alarm. On the other hand, it managed to boost the detection rate to a higher level as opposed to traditional line-based methods.

An image concept known as “V-disparity” has been extensively used in the process of staircase localization. “V-disparity” states that special planes in Euclidean space could be projected into straight lines in the “V-disparity” space. The projection reduces the 3D plane localization problem into a 2D line extraction problem. In the experiment, staircases are modeled as a collection of orthogonal planes which project linearly in the “V-disparity” space. The locations and dimensions of the staircases in the scene could be quickly recovered from these linear projections. The 3D data are rather accurate, especially for the nearer planes, the accuracy is within 2mm.

To the best of our knowledge, this project should represent the standout in the vision-based staircase detection and localization domain and provide value insights to the community.

7.2 Limitations and recommendations

- The project offers a solution for near frontal-looking staircase detection ($\beta \in [-15°,+15°]$). In practical applications, wider viewing angle is desirable. As a constituent part of autonomous staircase climbing, the climbing-down-view should also be considered in future works.
- This algorithm gives width and height information for each step, but without length information. Actual length for each step is nearly impossible to recover but effective length could be acquired by pixel classification and boundary

growing on each planar region after the 3D data is obtained. However, this process incurs extra time. In most circumstances, the length information is unnecessary as long as the detected space is enough for robot climbing.

- Parameter settings are very important to the 2D staircase detection. For every staircase, there is an optimal setting suited for it. Hence, it's a tough nut to crack in terms of finding a universal parameter suited for all types of staircases. Currently, the problem is solved by setting a dense scanning scheme for every staircase, which may often waste time unnecessarily.

Author's Publications

[1] S.S. Wang and H. Wang, "2D Staircase Detection Using Real AdaBoost", *Proc. 7th Intl. Conf. on Information, Communications and Signal Processing ICICS 2009*.

Bibliography

[1] S. Se and M. Brady, "Vision-based Detection of Kerbs and Steps", Proc. *8th British Machine Vision Conference BMVC' 97*, pp. 410 - 419, 1997.

[2] R. Turchetto and R. Manduchi, "Visual Curb Localization for Autonomous Navigation", Proc. *2003 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems IROS 2003*, Vol. 2, pp. 1336 – 1342, 2003.

[3] W.S. Wijesoma, K.R.S. Kodagoda and A.P. Balasuriya, "A Laser and a Camera for Mobile Robot Navigation", Proc. *7th Intl. Conf. on Control, Automation, Robotics and Vision ICARCV'02*, Vol. 2, pp. 740 – 745, 2002.

[4] F. Oniga, S. Nedevschi and M.M. Meinecke, "Curb Segments Detection with Temporal Filtering for Urban Driving Scenarios", Proc. *4th Intl. Conf. on Intelligent Computer Communication and Processing ICCP 2008*, pp. 291 - 294, 2008.

[5] J.D. Martens and W.S.Newman, "Stabilization of a Mobile Robot Climbing Stairs", Proc. 1994 *IEEE Intl. Conf. on Robotics and Automation ICRA 1994*, Vol. 3, pp. 2501 – 2507, 1994.

[6] E. Mihankhah, A. Kalantari, E. Aboosaeedan, H.D. Taghirad and S.A li.A. Moosavian, "Autonomous Staircase Detection and Stair Climbing for a Tracked Mobile Robot using Fuzzy Controller", Proc. 2008 IEEE Intl. Conf. on Robotics and Biomimetics ROBIO 2008, pp. 1980 – 1985, 2009.

[7] A. I. Mourikis, N. Trawny, S.I. Roumeliotis, D.M. Helmick and L. Matthies, "Autonomous Stair Climbing for Tracked Vehicles", *Intl. Journal of Robotics Research*, Vol. 26, pp. 737 – 758, 2007.

[8] D.M. Helmick, S.I. Roumeliotis, M.C. McHenry and L. Matthies, "Multi-sensor, High Speed Autonomous Stair Climbing", Proc. *2002 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems IROS 2002*, Vol. 1, pp. 733 – 742, 2002.

[9] E.Z. Moore, D. Campbell, F.Grimminger and M. Buehler, "Reliable Stair Climbing in the Simple Hexapod 'RHex'", Proc. *2002 IEEE Intl. Conf. on Robotics and Automation ICRA 2002*, Vol. 3, pp. 2222 – 2227, 2002.

[10] M. Yim, S. Homans and K. Roufas, "Climbing with Snake-like Robots", Proc*. IFAC Workshop on Mobile Robot Technology*, 2001.

[11] Y.L. Xiong and L. Metthies, "Vision-guided Autonomous Stair Climbing", Proc. 2000 IEEE Intl. Conf. on Robotics and Automation ICRA 2000, Vol. 2, pp. 1842 – 1847, 2000.

[12] Y. Cong, X.M. Li, J. Liu and Y.D. Tang, "A Stairway Detection Algorithm based on Vision for UGV Stair Climbing", Proc. *2008 IEEE Intl. Conf. on Networking, Sensing and Control ICNSC 2008*, pp. 1806 – 1811, 2008.

[13] S. Se and M. Brady, "Vision-based Detection of Staircase", *In 4th Asian Conf. on Computer Vision ACCV 2000*, Vol. 1, pp. 535 – 540, 2000.

[14] N. Molton, S. Se, M. Brady, D. Lee and P. Probert, "Robotic Sensing for the Partially Sighted", *Robotics and Autonomous Systems*, Vol. 26, pp. 185 – 201, 1999.

[15] L. Roberts, "Machine Perception of 3D Solids", Ph.D. Dissertation, MIT Department of Electrical Engineering, Cambridge, Massachusetts, 1963.

[16] R.O. Duda and P.E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures", *Communications of the ACM*, Vol. 15, pp. 11 -15, 1972.

[17] G.Q. Lu, H.G. Xu and Y.B. Li, "Line Detection Based on Chain Code Detection", Proc. *2005 IEEE Intl. Conf. on Vehicular Electronics and Safety*, pp. 98 – 103, 2005.

[18] J.B. Burns, A.R. Hanson and E. M. Riseman, "Extracting Straight Lines", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, pp. 425 – 455, 1986.

[19] K.A. Paton, "Conic Sections in Automatic Chromosome Analysis", *Machine Intelligence,* Vol. 5, pp.39 -40, 1970.

[20] J. Montanari, "On the Optimal Detection of Curves in Noisy Pictures", *Communications of the ACM*, Vol. 14, pp. 335 – 345, 1971.

[21] S.W. Zucker, R.A. Hummel and A. Rosenfeld, "An Application of Relaxation Labelling to Line and Curve Enhancement", *IEEE Transactions on Computers*, Vol. 26, pp. 394 – 403, 1977.

[22] G.F. McLean and D. Kotturi, "Vanishing Point Detection by Line Clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 17, pp. 1090 – 1095, 1995.

[23] T. Tuytelaars, L. Van Gool and M. Proesmans and T. Moons, "The Cascaded Hough Transform as an Aid in Aerial Image Interpretation", Proc. *6th Intl. Conf. on Computer Vision*, pp. 67 – 72, 1998.

[24] R.T. Collins and R.S. Weiss, "Vanishing Point Calculation as a Statistical Inference on the Unit Sphere", Proc. *3rd Intl. Conf. on Computer Vision*, pp. 400 – 403, 1990.

[25] D. Gabor, "Theory of Communication", *Journal of the Institute of Electrical Engineers*, Vol. 93, pp. 429 – 457, 1946.

[26] P. Viola and M. Jones. "Rapid Object Detection Using a Boosted Cascade of Simple Features", Proc. *2001 IEEE Conf. on Computer Vision and Pattern Recognition CVPR 2001*, Vol. 1, pp. 511 – 518, 2001.

[27] C. Papageorgiou and T. Poggio, "A Trainable System for Object Detection", *Intl. Journal of Computer Vision*, Vol. 38, pp. 15 – 33, 2000.

[28] S. Derivaux, S. Lefevre, C. Wemmert and J. Korczak, "On Machine Learning in Watershed Segmentation", *2007 IEEE Workshop on Machine Learning for Signal Processing*, pp. 187 – 192, 2007.

[29] Y. Freund and R.E. Schapire, "A Short Introduction to Boosting", *Journal of Japanese Society for Artificial Intelligence*, Vol. 15, pp. 771 – 780, 1999.

[30] M. Kearns and L.G. Valiant, "Learning Boolean Formulae or Finite Automata is as Hard as Factoring", *Technical Report TR-14-88, Harvard University Aiken Computation Laboratory*, August 1988.

[31] M. Kearns and L.G. Valiant, "Cryptographic Limitations on Learning Boolean Formulas and Finite Automata", *Journal of the Association for Computing Machinery*, Vol. 41, pp. 67 – 95, 1994.

[32] R.E. Schapire, "The Strength of Weak Learnability", *Machine Learning*, Vol. 5, pp. 197 – 227, 1990.

[33] Y. Freund and R.E. Schapire, "A Decision-theoretic Generalization of On-line Learning and an Application to Boosting", *Journal of Computer and System Sciences*, Vol. 55, pp. 119 – 139, 1997.

[34] H.A. Rowley. "Neural Network-based Human Face Detection", PhD thesis, Carnegie Mellon University, May, 1999.

[35] H. Schneiderman and T. Kanade, "A Statistical Method of 3D Object Detection Applied to Faces and Cars", Proc. 2000 *IEEE Conf. on Computer Vision and Pattern Recognition CVPR 2000*, Vol. 1, pp. 746 – 751, 2000.

[36] R. Xiao, L. Zhu and H.J. Zhang, "Boosting Chain Learning for Object Detection", Proc. *9th IEEE Intl. Conf. on Computer Vision ICCV 2003*, Vol. 1, pp. 709 – 715, 2003.

[37] C. Liu and H.Y. Shum, "Kullback-Leibler Boosting", Proc. *2003 IEEE Conf. on Computer Vision and Pattern Recognition CVPR 2003*, Vol. 1, pp. 587 – 594, 2003.

[38] K. Masada, Q. Chen, H.Y. Wu and T. Wada, "GA Based Feature Generation for Training Cascade Object Detector", Proc. *19th Intl. Conf. on Pattern Recognition ICPR 2008*, pp. 1 - 4, 2008.

[39] A. Treptow, A. Masselli and A. Zell, "Real-time Object Tracking for Soccer-robots without Color Information", *Robotics and Autonomous Systems*, Vol. 48, pp. 41 - 48, 2004.

[40] P. Viola, M. Jones and D. Snow, "Detecting Pedestrians Using Patterns of Motion and Appearance", Proc. *9th IEEE. Int. Conf. on Computer Vision ICCV 2003*, Vol. 2, pp. 734 – 741, 2003.

[41] H.F. Zhang, W.J. Jia, X.J. He and Q. Wu, "Learning-based License Plate Detection Using Global and Local Features", Proc. *18th Intl. Conf. on Pattern Recognition ICPR 2006*, Vol. 2, pp. 1102 – 1105, 2006.

[42] Z. Niu, S. Shan, S. Yan, X. Chen and W.Gao, "2D Cascaded AdaBoost for Eye Localization", Proc. *18th Intl. Conf. on Pattern Recognition ICPR 2006*, Vol. 2, pp. 1216 - 1219, 2006.

[43] R. Lienhart, L. Liang and A. Kuranov, "A Detector Tree of Boosted Classifiers for Real-time Object Detection and Tracking", Proc. *2003 Intl. Conf. on Multimedia and Expo ICME'03*, Vol. 2 pp. 277 – 280, 2003.

[44] Q. Chen, N.D. Georganas and E.M. Petriu, "Real-time Vision-based Hand Gesture Recognition Using Haar-like Features", Proc. *2007 IEEE Instrumentation and Measurement Technology Conf. IMTC 2007*, pp. 1 - 6, 2007.

[45] C.P. Papageorgiou, M. Oren and T. Poggio, "A General Framework for Object Detection", Proc. *6th Intl. Conf. on Computer Vision ICCV 1998*, pp. 555 – 562, 1998.

[46] R. Lienhart and J. Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection", *Proc. 2002 Intl. Conf. on Image Processing ICIP 2002*, Vol. 1, pp. 900 – 903, 2002.

[47] F. C. Crow, "Summed-area Tables for Texture Mapping", Proc. *11th Annual Conf. on Computer Graphics and Interactive Techniques SIGGRAPH '84*, Vol. 18, pp. 207 – 212, 1984.

[48] S.M.S. Islam, M. Bennamoun and R. Davies, "Fast and Fully Automatic Ear Detection Using Cascade AdaBoost", IEEE Workshop on Application of Computer Vision WACV 2008, pp. 1 – 6, 2008.

[49] H.C. Lu, W. Zhang and D.L. Yang, "Eye Detection Based on Rectangle Features and Pixel-pattern-based Texture Features", *Proc. 2007 Intl. Symposium on Intelligent Signal Processing and Communication Systems ISPACS 2007*, pp, 746 – 749, 2007.

[50] R.E. Schapire and Y. Singer, "Improved Boosting Algorithm Using Confidence-rated Predictions", *Machine Learning*, Vol. 37, pp. 297 – 336, 1999.

[51] A. Olmos and F.A.A. Kingdom, "McGill Calibrated Colour Image Database, http://tabby.vision.mcgill.ca",2004.

[52] X.Y. Lu and R. Manduchi, "Detection and Localization of Curbs and Stairways Using Stereo Vision", *Proc. 2005 IEEE Intl. Conf. on Robotics and Automation ICRA 2005*, pp. 4648 – 4654, 2005.

[53] J. Forsberg, U. Larsson and A. Wernersson, "Mobile Robot Navigation Using the Range-Weighted Hough Transform", *IEEE Robotics & Automation Magazine*, Vol. 2, pp. 18 – 26, 1995.

[54] E. Trucco and A. Verri, "Introductory Techniques for 3-D Computer Vision",Prentice Hall, pp. 87, 1998.

[55]A.Bab-Hadiashar and D. Suter, "Robust Range Segmentation", *Proc. 14th Intl. Conf. on Pattern Recognition, 1998*, Vol. 2, pp. 969 – 971, 1998.

[56] J.S. Gutmann, M. Fukuchi and M. Fujita, "Stair Climbing for Humanoid Robots Using Stereo Vision", *Proc. 2004 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems IROS 2004*, Vol. 2, pp. 1407 – 1413, 2004.

[57] X.Y. Jiang and H. Bunke, "Fast Segmentation of Range Images into Planar Regions by Scan Line Grouping", *Machine Vision and Applications*, Vol. 7, pp. 115 – 122, 1994.

[58] S. Gachter, V. Nguyen and R. Siegwart, "Results on Range Image Segmentation for Service Robots", Proc*. 4th IEEE Intl. Conf. on Computer Vision Systems ICVS 2006*, pp.53 – 53, 2006.

[59] X. Jiang, K. Bowyer, Y. Morioka, S. Hiura, K. Sato, S. Inokuchi, M. Bock, C. Guerra, R.E. Loke and J.M.H. du Buf, "Some Further Results of Experimental Comparison of Range Image Segmentation Algorithms", Proc. *15th Intl. Conf. on Pattern Recognition ICPR 2000*, Vol. 4, pp. 877 – 881, 2000.

[60] R. Labayrade, D. Aubert and J.P. Tarel, "Real Time Obstacle Detection in Stereovision on non-flat Road Geometry Through 'V-disparity' Representation", *IEEE Intelligent Vehicle Symposium 2002*, Vol. 2, pp. 646 -651, 2002.

[61] J. Rebut, G. Toulminet and A. Bensrhair, "Road Obstacles Detection Using a Self-adaptive Stereo Vision Sensor: a Contribution to the ARCOS French Project", Proc. *IEEE Intelligent Vehicle Symposium 2004*, pp. 738 – 743, 2004.

[62] A. Broggi, C. Caraffi, R.I. Fedriga and P. Grisleri, "Obstacle Detection with Stereo Vision for Off-Road Vehicle Navigation", Proc. *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition CVPR 2005*, pp. 65 – 65, 2005.

[63] N. Soquet, D. Aubert and N. Hautiere, "Road Segmentation Supervised by an Extended V-Disparity Algorithm for Autonomous Navigation", Proc. *IEEE Intelligent Vehicle Symposium 2007*, pp. 160 -165, 2007.

[64] Z.C. Hu and K. Uchimura, "U-V-Disparity: An Efficient Algorithm for Stereovision Based Scene Analysis", Proc. *IEEE Intelligent Vehicle Symposium 2005*, pp. 48 -54, 2005.

[65] N. Suganuma and N. Fujiwara, "An Obstacle Extraction Method Using Virtual Disparity Image", Proc. *IEEE Intelligent Vehicle Symposium 2007*, pp. 456 – 461, 2007.

[66] N. Suganuma, M. Shimoyama and N. Fujiwara, "Obstacle Detection Using Virtual Disparity Image for non-float Road", Proc*. IEEE Intelligent Vehicles Symposium 2008*, pp. 596 – 601, 2008.

[67] N. Suganuma, M. Shimoyama and N. Fujiwara, "Obstacle Map Generation Using Virtual Disparity Image for non-flat Road", *SICE Annual Conf. 2008*, pp. 1920 – 1925, 2008.

[68] A.D. Sappa, R. Herrero, F. Dornaika, D. Geronimo and A. Lopez, "Road Approximation in Euclidean and V-disparity Space: A Comparative Study", Proc *11th Intl. Conf. on Computer Aided Systems Theory EUROCAST 2007*, pp. 1105 – 1112, 2007.

[69] C.H. Wang, H. Tanahashi, H. Hirayu, Y. Niwa and K. Yamamoto, "Comparison of Local Plane Fitting Methods for Range Data", Proc *2001 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition CVPR 2001*, Vol. 1, pp. 663 – 669, 2001.

[70] J. W. Weingarten, G. Gruener and R. Siegwart, "Probabilistic Plane Fitting in 3D and an application to robotic mapping", Proc. *2004 IEEE Intl. Conf. on Robotics and Automation ICRA 2004*, Vol. 1, pp. 927 – 932, 2004.