

**PURDUE UNIVERSITY  
GRADUATE SCHOOL  
Thesis/Dissertation Acceptance**

This is to certify that the thesis/dissertation prepared

By Satyam Srivastava

Entitled

Display Device Color Management and Visual Surveillance of Vehicles

For the degree of Doctor of Philosophy

Is approved by the final examining committee:

EDWARD J. DELP

Chair

DAVID S. EBERT

JAN P. ALLEBACH

MARY L. COMER

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): EDWARD J. DELP

Approved by: M. R. Melloch

Head of the Graduate Program

04-19-2011

Date

**PURDUE UNIVERSITY  
GRADUATE SCHOOL  
Thesis/Dissertation Acceptance**

This is to certify that the thesis/dissertation prepared

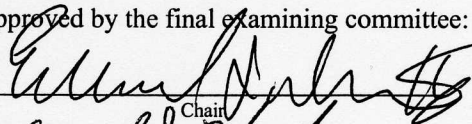

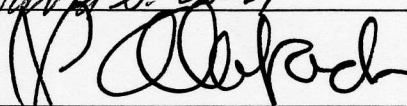
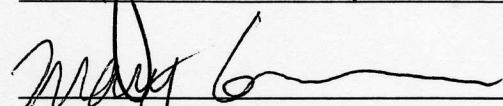
By Satyam Srivastava

Entitled

Display Device Color Management and Visual Surveillance of Vehicles

For the degree of Doctor of Philosophy

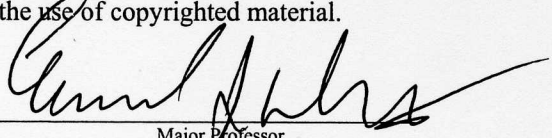
Is approved by the final examining committee:

 \_\_\_\_\_  
Chair  
 \_\_\_\_\_  
 \_\_\_\_\_  
 \_\_\_\_\_

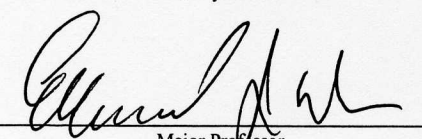
Approved by: Michael R. Mellock  
Head of the Graduate Program

4/19/11  
Date

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

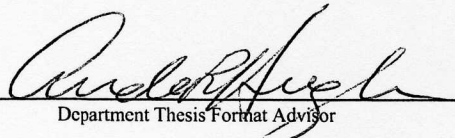
  
Major Professor

This thesis  is  
 is not to be regarded as confidential.

  
Major Professor

Format Approved by:

\_\_\_\_\_  
Chair, Final Examining Committee

or   
Department Thesis Format Advisor

**PURDUE UNIVERSITY  
GRADUATE SCHOOL**

**Research Integrity and Copyright Disclaimer**

Title of Thesis/Dissertation:

Display Device Color Management and Visual Surveillance of Vehicles

For the degree of Doctor of Philosophy

I certify that in the preparation of this thesis, I have observed the provisions of *Purdue University Executive Memorandum No. C-22*, September 6, 1991, *Policy on Integrity in Research*.\*

Further, I certify that this work is free of plagiarism and all materials appearing in this thesis/dissertation have been properly quoted and attributed.

I certify that all copyrighted material incorporated into this thesis/dissertation is in compliance with the United States' copyright law and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and save harmless Purdue University from any and all claims that may be asserted or that may arise from any copyright violation.

Satyam Srivastava

Printed Name and Signature of Candidate

04-13-2011

Date (month/day/year)

\*Located at [http://www.purdue.edu/policies/pages/teach\\_res\\_outreach/c\\_22.html](http://www.purdue.edu/policies/pages/teach_res_outreach/c_22.html)

DISPLAY DEVICE COLOR MANAGEMENT AND  
VISUAL SURVEILLANCE OF VEHICLES

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Satyam Srivastava

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

May 2011

Purdue University

West Lafayette, Indiana

Dedicated to all those who tried and failed...and tried again.

## ACKNOWLEDGMENTS

As this four year journey eases towards an end, I reminisce about the vivid memories of graduate school. The joys and frustrations paint multiple faces of those who helped me enjoy, or survive, what life tossed up. It is only fitting then, that I start my thesis by expressing gratitude to everyone of these.

It has been (and will always be) an honor to call myself a student of Professor Edward J. Delp. I am grateful to him for agreeing to be my advisor and for the opportunity to work in the Video and Image Processing Laboratory (VIPER). I also thank him for his consistent encouragement, guidance, and trust. His extensive knowledge and insight has helped keep my work progressing and on-course.

I am grateful to my advisory committee members: Professor Jan P. Allebach, Professor Mary L. Comer, and Professor David S. Ebert for their valuable suggestions, questions, and support. A special note of thanks to Professor Allebach for his guidance in my color related work and beyond. I would like to thank Purdue University, the Graduate School, and the School of Electrical and Computer Engineering for accepting me into the Doctoral program. I am thankful to Dr. Shikha Tripathi for introducing me to academic research and signal processing during my undergraduate years.

It has been a pleasure working with my wonderful colleagues: Dr. Nitin Khanna, Dr. Golnaz Abdollahian, Dr. Ying Chen Lou, Deen King-Smith, Meilin Yang, Fengqing Zhu, Kevin Lorenz, Ka Ki Ng, Marc Bosch, Aravind Mikkilineni, Albert Parra, Chang Xu, Bin Zhao, and Thanh Ha. Special thanks are due to Thanh and Ka Ki for their collaboration with my work, to Kevin for patiently solving all the IT troubles, and to Aravind for always having a solution to the otherwise unsolvable problems!

I thank all my friends, decades-long and recent, for their affection and their belief in me. They gave me the strength to weather the difficult times, which have been aplenty. I am grateful to my family for their perpetual support of my career aspirations and for their unconditional love, distances notwithstanding.

I would like to thank the Indiana 21st Century Research and Technology Fund program, the United States Department of Homeland Security, and the United States Naval Research Laboratory for funding the projects which resulted in this thesis.

## TABLE OF CONTENTS

	Page
LIST OF TABLES . . . . .	vii
LIST OF FIGURES . . . . .	viii
ABBREVIATIONS . . . . .	xi
NOMENCLATURE . . . . .	xii
ABSTRACT . . . . .	xv
1 INTRODUCTION . . . . .	1
1.1 Color Management . . . . .	1
1.1.1 Problem Formulation . . . . .	5
1.2 Visual Surveillance . . . . .	12
1.3 Contributions of this Thesis . . . . .	15
1.4 Publications Resulting from this Work . . . . .	17
2 COLOR MANAGEMENT OF DISPLAY DEVICES . . . . .	19
2.1 Color Management Using Device Models . . . . .	19
2.1.1 Measurements . . . . .	20
2.1.2 Models for prediction . . . . .	23
2.2 Color Management With 3D Look-Up Tables . . . . .	48
2.2.1 Interpolation . . . . .	52
2.2.2 Sampling . . . . .	58
3 VISUAL SURVEILLANCE OF VEHICLES . . . . .	64
3.1 Overview of the Proposed Surveillance System . . . . .	64
3.2 Image and Video Analyses . . . . .	70
3.2.1 Vehicle Detection . . . . .	70
3.2.2 Vehicle Body-Type Determination . . . . .	74
3.2.3 Tire Size Estimation . . . . .	77



	Page
3.2.4 Make Recognition . . . . .	82
3.2.5 Squat And Bounce Analysis . . . . .	85
3.2.6 Trajectory Analysis for Anomaly Detection . . . . .	87
3.2.7 Trajectory Estimation and Co-Ordinate Mapping . . . . .	89
3.2.8 Velocity Analysis . . . . .	93
3.2.9 Shape Analysis . . . . .	95
3.2.10 Color Correction for Object Tracking . . . . .	98
4 EXPERIMENTAL RESULTS . . . . .	105
4.1 Color Management . . . . .	105
4.1.1 Comparison of LUT With the Models . . . . .	105
4.1.2 Evaluation of Optimal LUT . . . . .	111
4.1.3 Comparison of Optimal LUT With ICC Profiles . . . . .	113
4.2 Surveillance of Vehicles . . . . .	116
4.2.1 Vehicle detection . . . . .	116
4.2.2 Vehicle type determination . . . . .	119
4.2.3 Tire size estimation . . . . .	121
4.2.4 Make recognition . . . . .	123
4.2.5 Squat and Bounce Analysis . . . . .	124
4.2.6 Trajectory Analysis . . . . .	126
4.2.7 Velocity Analysis . . . . .	128
4.2.8 Shape Analysis . . . . .	130
4.2.9 Color Correction for Tracking . . . . .	132
5 CONCLUSIONS AND FUTURE WORK . . . . .	136
5.1 Conclusions . . . . .	136
5.2 Future Work . . . . .	138
5.3 Publications Resulting from this Work . . . . .	139
LIST OF REFERENCES . . . . .	141
VITA . . . . .	151

## LIST OF TABLES

Table	Page
2.1 Testing error stastics of the monitor models. . . . .	29
2.2 Testing error stastics of the NLX model. . . . .	29
2.3 Testing error stastics of the LX model. . . . .	33
2.4 Results of monitor white point correction. . . . .	43
2.5 Testing error statistics of the monitor characterization models. . . . .	44
2.6 Testing error stastics of the NLXWP model. . . . .	45
2.7 Testing error stastics of the LXWP model. . . . .	47
2.8 Average LUT errors with two interpolation techniques. . . . .	56
4.1 Testing error statistics of NLXWP, LXWP, and LUT. . . . .	108
4.2 Training error of LUT with optimal sampling. . . . .	109
4.3 Testing error statistics of optimal and non-optimal LUT. . . . .	112
4.4 Testing error stastics for LUT and profile based system. . . . .	115
4.5 Execution times comparison for object detection. . . . .	119
4.6 Static effect of loading on vehicle. . . . .	125
4.7 Euclidean distance between reference and color-corrected images. . . . .	133
4.8 Bhattacharyya distance between reference and color-corrected images. . . . .	134

## LIST OF FIGURES

Figure	Page
1.1 Profile based color management. . . . .	3
1.2 Color profile support in web browsers. . . . .	3
1.3 Digital intermediate workflow. . . . .	5
1.4 Comparison of visual output of two monitors. . . . .	6
1.5 Color grading in ‘ <i>O Brother...</i> ’ . . . . .	8
1.6 Color management using LUT. . . . .	9
1.7 Color management using profiles. . . . .	10
1.8 Situation in a typical surveillance control room. . . . .	13
2.1 Photo Research PR-705. . . . .	21
2.2 A screen shot of SpectraWin. . . . .	21
2.3 A block diagram of the NLX model. . . . .	24
2.4 A block diagram of the forward monitor model. . . . .	25
2.5 A block diagram of the inverse monitor model. . . . .	27
2.6 Gamma LUTs for the monitors. . . . .	28
2.7 Distribution of error outliers of the NLX model. . . . .	30
2.8 A framework for developing the LX model. . . . .	31
2.9 Distribution of error outliers of the LX model. . . . .	32
2.10 A block diagram of the NLXWP model. . . . .	36
2.11 A block diagram of the white-balancing process. . . . .	39
2.12 Gamut mapping by straight chroma clipping. . . . .	40
2.13 Gamut leaf in the LAB space at $h^*=330$ degree. . . . .	42
2.14 Distribution of error outliers of the NLXWP model. . . . .	45
2.15 A block diagram of the LXWP model. . . . .	46
2.16 Distribution of error outliers of the LXWP model. . . . .	47

Figure	Page
2.17 Block diagram of LUTEVAL module. . . . .	49
2.18 Neighbors in 3D space. . . . .	53
2.19 Variation of lightness ( $L^*$ ) with digital red input $R$ . . . . .	60
2.20 RGB optimal sub-sampling problem. . . . .	61
3.1 A multi-sensor surveillance system. . . . .	66
3.2 A video surveillance system for vehicles. . . . .	67
3.3 Types of information extracted from video. . . . .	69
3.4 Spiral search for object detection. . . . .	73
3.5 An example template for vehicle type recognition. . . . .	75
3.6 Vehicle tires with different hubs. . . . .	77
3.7 A graphical illustration of the system's view of a tire with shiny hub. . . . .	78
3.8 The wheel-rubber edge model. . . . .	79
3.9 The second stage of tire size estimation. . . . .	80
3.10 A graphical illustration of the system's view of a tire with dark hub. . . . .	81
3.11 Examples of object detection and tracking. . . . .	90
3.12 User specified co-ordinate transformation . . . . .	92
3.13 A schematic diagram of an imaging unit. . . . .	100
3.14 A block diagram of the CCMX function. . . . .	102
3.15 Color card used for camera models. . . . .	103
4.1 Histogram of errors in LUT based prediction (against NLXWP). . . . .	106
4.2 Distribution of error outliers of the LUT-based implementation. . . . .	106
4.3 A block diagram for evaluating the three implementation models. . . . .	107
4.4 Distribution of error outliers of NLXWP, LXWP, and LUT. . . . .	108
4.5 Convergence of optimization error for the three LUT. . . . .	110
4.6 Evaluation of optimal and non-optimal LUT. . . . .	111
4.7 Comparison of LUT and profile based systems. . . . .	113
4.8 An example of simultaneous vehicle detection and tracking. . . . .	117
4.9 Examples of vehicle detection. . . . .	118

Figure	Page
4.10 Examples of object detection compared with GMM. . . . .	118
4.11 Vehicle type identification (still image). . . . .	120
4.12 Vehicle type identification (video). . . . .	121
4.13 Vehicle tire extraction (still image). . . . .	121
4.14 Vehicle tire extraction (video). . . . .	122
4.15 Extraction of tires with dark hubs. . . . .	123
4.16 Vehicle make recognition (video). . . . .	124
4.17 Visual impact of loading on vehicles. . . . .	126
4.18 Oscillation of loading and unloaded vehicle. . . . .	127
4.19 Images from the test videos and satellite maps. . . . .	128
4.20 Output of LUT-based co-ordinate transformation. . . . .	129
4.21 Anomaly detection using spatial velocity. . . . .	129
4.22 Anomaly detection using estimated velocity. . . . .	130
4.23 Results of trajectory shape analysis. . . . .	131
4.24 Result of shape analysis on a synthetic multi-turn trajectory. . . . .	131
4.25 Reduced false turn detection for curved roads. . . . .	132
4.26 Target region for color histogram computation. . . . .	134
4.27 Output images after color correction. . . . .	135

## ABBREVIATIONS

APE	Average Prediction Error
CCMX	Camara to Camera Model-based Transformation
CIE	Commission Internationale de l'éclairage
CMS	Color Management System
DI	Digital Intermediate
FVC	Front View Camera
GMM	Gaussian Mixture Models
ICC	International Color Consortium
IU	Imaging Unit
LUT	Look-Up Table
LX	Linear Transformation
LXWP	Linear Transformation with White Point correction
MABS	Motion-Assisted Background Subtraction
MMR	Make Model Recognition
NLX	Non-linear Transformation
NLXWP	Non-linear Transformation with White Point correction
PCS	Profile Connection Space
ROI	Region Of Interest
SAD	Sum of Absolute Difference
SIFT	Scale Invariant Feature Transform
SVC	Side View Camera

## NOMENCLATURE

CTLLMO-1	Display unit owned by our industry partner and first (1) device of type LMO studied in the project
CTLCSO-1	Display unit owned by our industry partner and first (1) device of type CSO studied in the project
BLRLMO-2	Display unit owned by Purdue team (BLR) and second (2) device of type LMO studied in the project
BLRCSO-2	Display unit owned by Purdue team (BLR) and second (2) device of type CSO studied in the project
Reference device	In color matching experiments, these refer to devices of type LMO, denoted by subscript <sub>1</sub> or superscript <sup>1</sup>
Viewing device	In color matching experiments, these refer to devices of type CSO, denoted by subscript <sub>2</sub> or superscript <sup>2</sup>
Artist	A generic term used to refer to photographers, cinematographers, colorists, and other individuals who participate in the creation/processing of a digital video or image
Field of view	The part of an observation environment which is visible to a camera's imaging sensors
Background model	In object detection, it is the state of the scene when no objects of interest are known to be present
Image co-ordinates	A 2D co-ordinate system in which object positions/dimensions are defined using row and column indices of an image
Front grille	A term used to describe the vehicle components including the radiator grille, the carmaker logo, and headlights. Also known as front end module
Ground co-ordinates	A 2D or 3D co-ordinate system in which true object positions/dimensions are specified. Also referred to as world co-ordinates
Operator	A generic term used to describe personnel in a remote surveillance control room. These may include observers, analysts, liaisons, and police officers
Standoff range	The approximate distance (from a subject) at which a system can reliably operate
Perspective	A generic term used to describe the effects of camera angle and the mapping from 3D to 2D plane in the imaging process

Shade	An intuitive descriptor for the color of an object as seen by a human. While the perception of color is complex, the radiometric input to the human visual system depends on the illumination and the object reflectance. Shade is associated with the reflectance and is assumed to be largely invariant to illumination.
Trajectory	The 1/2/3D position of an object as a function of time while the object is in the field of view. Typically obtained by object tracking.
Traffic flow	A qualitative descriptor for the smoothness or impededness of vehicle movement. Flow rate can be specified as number of vehicles passing a chosen region in unit time interval.
Channel	Unlike a communication system setup, here channel does not imply a medium of information transmission in the typical sense. Instead the three colors (R, G, B) in a tristimulus color model are referred to as channels. A tristimulus model of color representation is one in which different colors are generated by a weighted sum of three primary components - $m = r \cdot \mathbf{R} + g \cdot \mathbf{G} + b \cdot \mathbf{B}$ , where $\mathbf{R}, \mathbf{G}, \mathbf{B}$ represent the components and $r, g, b$ are the weighting factors.
Linear RGB	A simple form of interpreting an image in which value of each pixel is proportional to photonic energy (incident at the time of image acquisition). A digital image in linear RGB will be described by a 2-dimensional array of pixels and each pixel is described using a triplet of real numbers $(R, G, B)^{lin}$ . Depending on the context, these values may be <i>absolute</i> or <i>normalized</i> to a number between 0.0 and 1.0. Note that this form of image data may sometimes be referred to as <i>gamma uncorrected</i> to emphasize that the device (camera/display) characteristics have not been utilized in encoding the image.
Non-linear RGB	Also known as <i>gamma corrected RGB</i> , this is the common image coding format wherein pixel values are power law corrected and stored as quantized integers (8/16 bit depending on the image coding standard). Such an image will be described by a 2-dimensional array of pixels each consisting of a triplet of integers $(R, G, B)^{nonl}$ or $(R, G, B)^{lin}$ . Poynton [1] provides a detailed treatment of the topic of gamma correction.
White point	The term white point is used in many contexts but primarily refers to the color produced (measured in XYZ, xy, LAB, or any other device independent space) when a white input (say $RGB^{nonl} = (255, 255, 255)$ in an 8-bit per channel coding) is provided to a display device. It also refers to some standard illuminants like the D65 white point [2] which resembles the color of daylight.



**Gamut** In the simplest sense, gamut indicates the range of colors that a device can reproduce. Typically, no device can generate all possible colors visible to human eyes and gamut represents a subspace of the complete color space that the device can achieve. Gamut matching is the process of transforming the inputs to a display/printing device such that the colors, which would otherwise fall outside the device's gamut, stay within the gamut.

**CIE 1931 XYZ** The CIE 1931 XYZ space [3] is device independent and commonly used as an intermediate space in device to device transformations. Note that the linear RGB space and the CIE XYZ space are related by a linear transformation:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} & & \\ & M_{3 \times 3} & \\ & & \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix};$$

Where  $M$  is a non-singular  $3 \times 3$  matrix.

**Sample point** When a look-up table is used to approximate a function with a domain  $D$ , the table is constructed by evaluating the function at a subset of  $D$ . The points in this subset are called sample points or knot points.

**Regular LUT** An  $nD$  LUT will be called regular if each dimension is divided into equal parts in order to construct the set of sample points. For example, a  $9 \times 9 \times 9$  LUT is a regular  $3D$  table while a  $4 \times 5$  LUT is not a regular  $2D$  table.

**Symmetric LUT** A regular LUT will be called symmetric if sample points are collected at the same locations along each dimension. An example of a symmetric LUT is one with the sample points in the Cartesian product  $\{0, 10, 20, 30, 40\} \times \{0, 10, 20, 30, 40\} \times \{0, 10, 20, 30, 40\}$ . If this condition is not satisfied, the regular LUT will be called non-symmetric. It should be noted that the classes symmetric and non-symmetric are partitions of the set of all regular LUTs.

## ABSTRACT

Srivastava, Satyam Ph.D., Purdue University, May 2011. Display Device Color Management and Visual Surveillance of Vehicles. Major Professor: Edward J. Delp.

Digital imaging has seen an enormous growth in the last decade. Today users have numerous choices in creating, accessing, and viewing digital image/video content. Color management is important to ensure consistent visual experience across imaging systems. This is typically achieved using color profiles. In this thesis we identify the limitations of profile-based color management systems and propose an alternative system based on display device models and look-up tables (LUT). We identify techniques to design LUTs which are optimal in terms of color reproduction accuracy under resource constraints. We show that a LUT-based color management system is more accurate and memory-efficient than a comparable ICC profile-based system.

Visual surveillance is often used for security and law enforcement applications. In most cases the video data is either passively recorded for forensic applications or is remotely monitored by human operators. We propose the use of image and video analysis techniques to assist the operators. This would reduce human errors due to fatigue, boredom, and excess information. We describe a video surveillance system to observe vehicular traffic from a standoff range and detect anomalous behavior by analyzing the motion trajectories. We also extract physical information (such as make, tire size, and body type) which can help determine the “normal” behavior. The operator can also use this information to uniquely identify/describe individual vehicles. We describe low complexity techniques to perform the above analyses and show their effectiveness on real traffic videos.

## 1. INTRODUCTION

Digital image and video systems have become hugely popular with both professional and casual users. There has been a tremendous growth in the tools available to the users for creating and editing, storing/sharing/accessing, and viewing the digital content. Cellphone cameras, personal media centers, and other portable devices are now ubiquitous. Concurrent to the advances in the area of imaging devices, there has been significant interest in developing novel applications for the information-rich digital content. For example, the availability of web access, media-capable handheld devices, and advanced video coding techniques have brought video conferences from meeting rooms to cellphones.

In this dissertation, we study two topics that appear in the modern image and video processing systems – color management of devices and video surveillance. While the importance of color management has increased significantly because of the growth in imaging devices, visual surveillance has become a major application of image analysis techniques in recent years. Both topics have considerable practical value – color management in cinema and photography while surveillance in security, law enforcement, and traffic management.

### 1.1 Color Management

The quantity of visual data continues to grow as more users, both amateur and professional, create pictures using a wide range of capture devices. In particular, digital images and video have become immensely popular. Similarly, with the ease of delivery, storage and display, viewers have enormous choice of how to view the content - from the two inch screen of a mobile phone to a fifty inch high definition LCD television compatible with the ITU Rec. 709 (HDTV) specification [4] to a

movie theatre screen with Digital Cinema (DCI) Reference Projector properties [5]. Unfortunately, as a natural consequence, it is unlikely that a visual data viewed using these devices would appear the same. More specifically, a digital image or video when applied as input to such displays will not produce the same visual stimulus (measured in some objective or subjective units). For a professional video creator, however, the output visual stimulus is the most important result. In some applications, it might be acceptable to change the digital representation of the data in order to achieve the desired visual properties on a given display device. In this document, the term *feature* is used to refer to any visual data and the terms digital image and digital video are used to describe its digital representation.

Color management is the general collection of various techniques developed with a goal of achieving visual match across display/printing devices. It has become an important part of digital imaging systems. Personal computers and workstations provide support for color management at multiple levels including operating system and application levels [6]. The International Color Consortium (ICC) recommends exchange of device (capture, print and display) characteristics as color profiles [7]. A collection of profiles could then be used for adequate rendering of images by the display devices. Koren [8] provides a detailed tutorial on color management and illustrates a general color management system based on device profiles. This is reproduced in Figure 1.1.

Several image coding formats (such as PNG [9] and EPS [10]) have the capability for embedding color profiles as an optional feature. Even some web browsers (like Safari and Firefox 3.0) allow use of color profiles [11] for enhanced color viewing of downloaded images with stored profile information. Figure 1.2 shows an example of an image rendering, with (Firefox 3) and without (Firefox 2) the use of embedded color profile information. However, color management for digital videos is not so mature [12] and presents additional challenges.

A good reproduction of color and general visual appearance is very important for the cinema (and television) industry. After a feature is recorded on a film or a

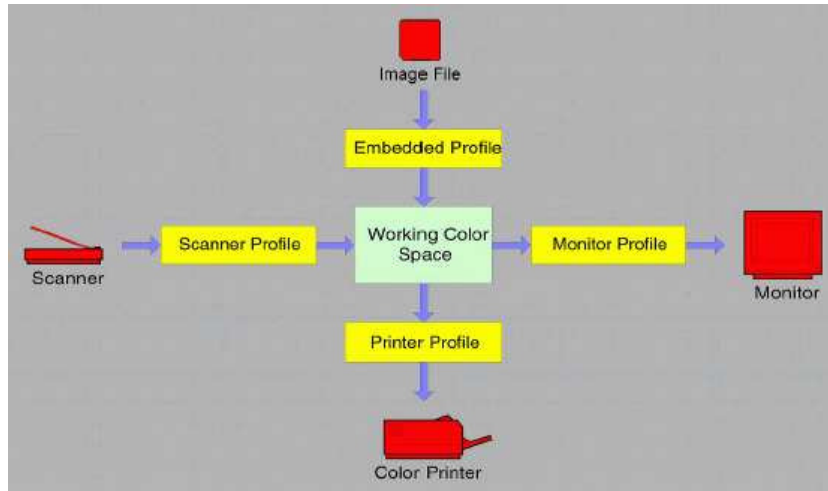


Fig. 1.1. A simplified illustration of a color-managed workflow (Courtesy: Norman Koren [8]).



Fig. 1.2. An example of color profile support in web browsers (Courtesy: Deb Richardson [11]).

digital media<sup>1</sup>, it is enhanced by artists to give a more suitable “look”. But such enhancements are performed in post-production facilities rather than actual movie

<sup>1</sup>This is called the *production* stage of a film.

theatre environments. In other words, it is not possible to observe the effects of a technique on the actual movie experience, in real time. Instead, modern facilities rely on faithful reproduction of visual data on personal viewing devices in a process known as Digital Intermediate (DI). In a DI workflow, it is assumed that the artists' personal displays can mimic a movie theatre experience and hence, all the enhancement techniques can be applied in the digital domain while observing the effects in real time. A brief description of the conventional (Telecine) workflow and the modern DI workflow is provided in [13]. Figure 1.3 shows a high level picture of the DI workflow. An important feature of this approach is that a physical film (marked as *digital intermediate* in the figure) needs to be generated after all the post-production activities are completed (on the digital or digitized video data). Hence, there is no need for multiple film prints during the process to observe the effect of various artistic techniques.

Digital Cinema Initiative, LLC (DCI) is a joint venture of several major studios with an aim of establishing an open architecture for digital production and delivery of motion pictures. A post-production process similar to the DI workflow is also applicable to features compatible with the DCI specification. In fact, for the discussions in this document, we assume that the color information is losslessly transferred from the digital representation (on which the artistic effects are applied) to the film which is distributed for screening.

It is obvious, however, that this process greatly depends on the digital (or digitized) video data displayed on the artists' viewing devices to appear the same as the actual film screened in a movie theatre. This gives us an idea for approaching the broader problem of achieving visual similarity across viewing platforms. We assume that there is a universal reference display and develop methods for matching given display devices to this universal display. With reference to the DI process, such a display would be the movie theatre projection system. Therefore, if all the display units in a DI-based workflow chain can be matched with the reference display, we

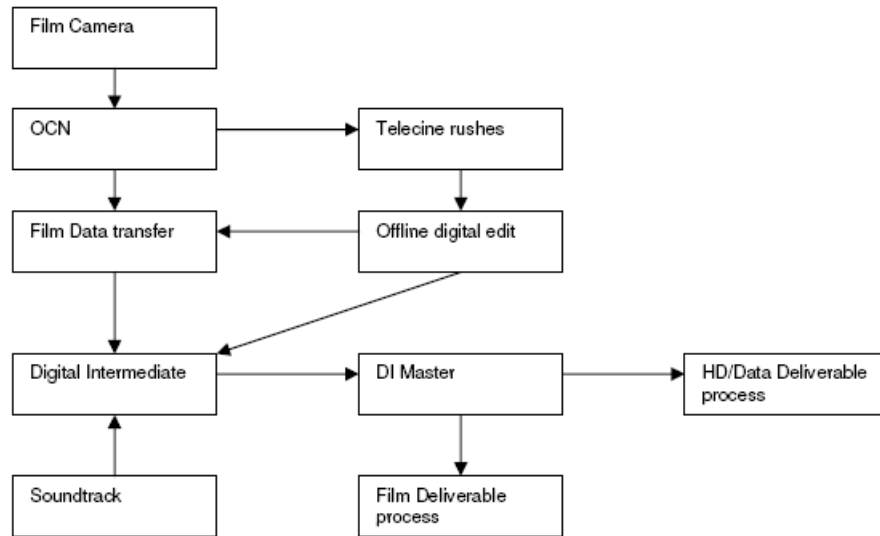


Fig. 1.3. A digital intermediate based workflow (Courtesy: Digital Praxis, UK [13]).

can expect that the feature as viewed by the artists on their personal display devices would be very similar to the movie theatre experience.

### 1.1.1 Problem Formulation

The overall objective can be defined as one of achieving visual similarity when a digital video content is viewed on any display device. While the problem itself seems arbitrarily complex, the assumption about availability of a universal reference makes it tractable. The task for any viewing device is to match the characteristics of the reference (or target) device as closely as possible.

The above problem of visual matching of a display device to a universal reference was realized using two monitors from our industry partner. belonging to different product generations by treating one display as reference (or target) and the other as viewing device. Figure 1.4 shows a photograph of the two monitors displaying a test image in a dark room, captured by a Sony DSC T5 camera. The image is a monitor/printer test image taken from [14] and includes skintones, natural scenes as well

as the Gretag Macbeth reference colors [15]. It can be seen that the reference device (right) produces warmer colors while the viewing device (left) has a richer green. It is required that a processed image displayed on the viewing monitor (based on hardware panels of type CSO $x$ ) should appear the same as the unprocessed image displayed on the reference monitor (based on LMO $x$  panels). The meaning of *processed* and *same* will be rigorously defined in the following sections.

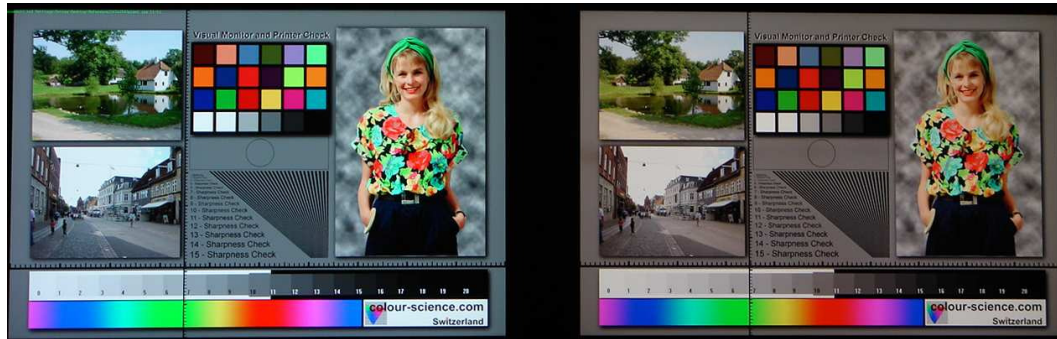


Fig. 1.4. A test image displayed on two monitors in a dark room (Test image - courtesy: Colour Science AG, Switzerland [14]).

We can further quantify the elements of the problem by formulating it in approximate mathematical terms. Let us model a display device as a transformation from the set of all digital images  $\mathfrak{S}$  to the set of all 2D visual stimuli<sup>2</sup>  $\mathfrak{R}$ . Let us denote the reference device as  $A : \mathfrak{S} \rightarrow \mathfrak{R}$  and the viewing device as  $B : \mathfrak{S} \rightarrow \mathfrak{R}$ . Then for any  $i \in \mathfrak{S}$ , it is most likely that  $A(i) \neq B(i)$ . Our goal is to develop another transformation  $G : \mathfrak{S} \rightarrow \mathfrak{S}$  such that  $B(G(i)) \approx A(i)$ , where  $\approx$  indicates visual similarity.

This can be related with the DI workflow problem by assuming that  $A$  is the cinema projector and  $B$  is an artist's display. The transformation  $G$  results in a new display device  $\tilde{B} = B \circ G$ . Now, if the matching is accurately performed, watching a feature on  $\tilde{B}$  will be visually the same as watching it on  $A$ , which is the movie theatre projection system.

<sup>2</sup>A 2D visual stimulus is precisely an *image*.



However, this would be the entry point for tackling the original problem of visual matching of different display devices. If  $B$  is any display device - computer monitor, television, digital picture frame or a cellphone on which a user can view the visual data, then a similar approach would ensure that the visual output is as close to the movie theatre experience as allowed by the device's technology. Note that in case of still images  $B$  can even be a print media and the problem of matching hardcopy and softcopy has also been actively researched [16–18].

An important observation at this point is that the term similarity is very ambiguous. We use the established definitions of color reproduction intents to formulate a more objective measure of alikeness of two displays (or equivalently, of an image as seen on two displays). This also requires a definite notion of whose opinion about an image matters the most. While it is easy to argue that such a person would be the viewer, the weakness of this argument becomes apparent as soon as we include the various media by which visual content is delivered to the viewers. When a motion picture is screened before three hundred viewers, whose opinion should matter and how should that be incorporated? Similarly, viewers watching a feature on a handheld device or even television have limited control over the appearance of the scenes (by adjusting brightness, contrast and perhaps color saturation).

This ambiguity is avoided by stating that a good benchmark for the appearance of a scene would be the intent of its creator or artist. It must be noted that motion picture and television production houses specifically employ artists and technicians (including colorists) who help create a more suitable appearance to the visual content. Even individuals sometimes express their dissatisfaction [11] with color rendering of their images in web browsers which ignore the photographer's intent. Typically, these artists have greater insight into the theme of a scene as related to the storyline and, hence, are in a better position to judge the visual appeal. Although some viewers may not appreciate the effect and ask why the grass is not green<sup>3</sup> [19] but it is the

---

<sup>3</sup>'Oh Brother, Where Art Thou?' was one of the earliest full feature films to make extensive use of digital color correction.

demand of the film's theme that the grass not be green and the artists create such an effect. Figure 1.5 compares two shots from a film where the image on the left is highly color graded to give the foliage a yellowish look.

Continuing with our model of the DI process, let us define another transformation  $\Upsilon : \mathfrak{S} \rightarrow \mathfrak{S}$  representing all the artistic effects applied to the feature during the post-production activities. Then, if  $i$  was the representation of the feature immediately after recording,  $\Upsilon(i)$  is the representation after post-production and is used in the actual distributions. Therefore, the visual output in the movie theatres would be  $A(\Upsilon(i))$  and must be exactly how the artists wanted it to be. Since  $\tilde{B} = B \circ G \approx A$ , therefore,  $\tilde{B}(\Upsilon(i)) \approx A(\Upsilon(i))$ .

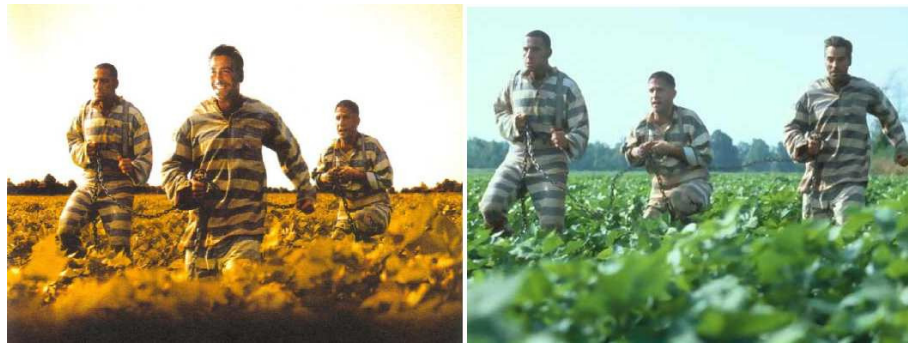


Fig. 1.5. An early example of digital color grading in films - *O Brother, Where Art Thou?* (Courtesy: Touchstone Pictures, USA [20])

Typically, however, all artists do not use the same displays and images and films are not shipped with the artists' monitors! This leads to an invalidation of the above assumption about a universal reference and exposes two more aspects of the problem. Unlike the simplified problem stated above in which both viewing stations are available for comparison, in most situations there is only one device on which a content is viewed. So, there must be some way to quantify the content creator's intent about the visual appearance or in the context of a universal reference, a way to specify the reference display. Finally, there must be a way to communicate this information to the viewing devices which would adjust itself to best replicate the same appearance.

ICC profiles provide one such mechanism of sharing display characteristics but have been reported to underperform in some applications [21–23] and are evolving.

In this thesis we propose a new approach to color management which is both accurate and hardware-friendly. Our solution is based on the assumption that a reference display device exists, and all other display devices can be matched to this reference. This consists of two steps. In the first step, we develop a method for perceptually matching a given display with a known reference display. Perceptual similarity [24, 25] is said to be achieved when two colors match in the CIE LAB space after accounting for chromatic adaptation of the human visual system [26] and ensuring a common white point. We select perceptual matching because it is more suitable for reproduction of motion content on systems with high dynamic ranges, and under different viewing conditions [25]. This first step may be completed offline. In the second step, this method is realized as a faster and simpler operation, by using a look-up table. This is illustrated in Fig. 1.6. We will show that this method provides better color reproduction because we develop a specific solution to match two display devices unlike matching a display to an unknown device using its profile (as shown in Figure 1.7)..

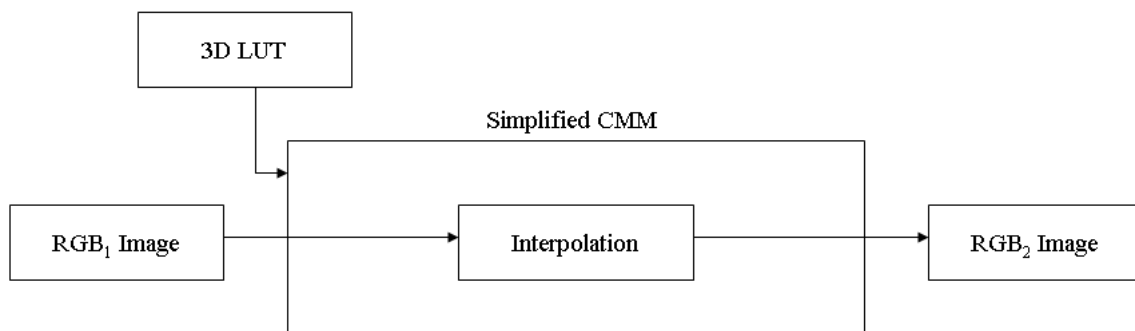


Fig. 1.6. A block diagram of our proposed LUT-based color management system.

It should be noted that the advantages of generating a direct device-to-device transformation, as we described above, have been recognized by color scientists. The ICC [25] defines a type of profile which contains the information about the transfor-

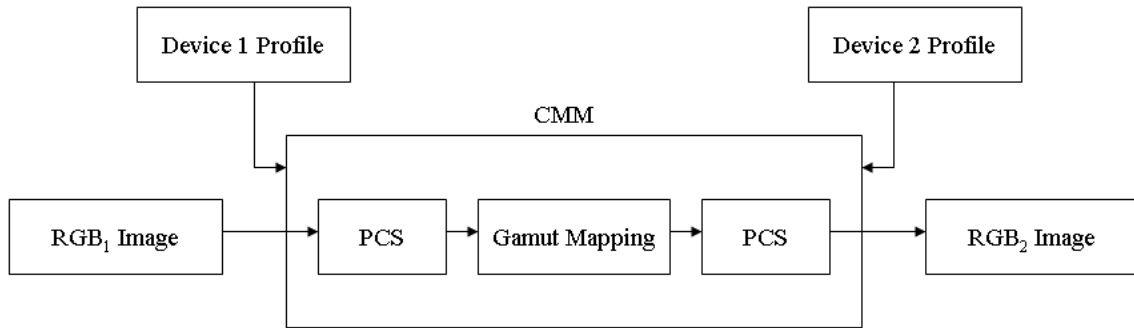


Fig. 1.7. A block diagram of typical profile-based color management systems.

mation between two known devices. Such profiles are known as *device-link profiles*. While a color management system utilizing device-link profiles promises better color accuracy and lower computational complexity, these profiles have been largely used in printing systems, but for a different reason. In high quality print systems, the transformation from one device's CMYK color space to another device's CMYK space may suffer loss of information when an intermediate space, the profile connection space, or the PCS (for example the CIE LAB space), has only three data channels. In such cases, device-link profiles can be used to bypass the PCS, and can also be used to preserve the neutral colors [27, 28] by preventing CMY components from leaking-in when only the K component may be needed.

These profiles can be created using the colorimetric data measured directly from the two devices. However, many profiling tools can generate a device-link profile indirectly, using two ICC device profiles [28–30]. Balonen-Rosen and Thornton [31] describe a method which allows a user to interactively modify the existing information in color profiles (including device-link profiles) to improve the quality of color management. This local color correction technique is useful when an image rendered on a destination device contains multiple color regions that are unacceptably different from the image rendered on a source device.

Our LUT-based approach is different from a system that uses device-link profiles in the following ways. The ICC specification does not allow a device-to-device trans-

formation (contained in a device-link profile) to be represented by *only* a look-up table. The four acceptable algorithmic models for such transformations shown in Figure 5, on page *xii* of the ICC standard [25] consist of one or more “processing elements,” including a multi-dimensional LUT. The two configurations which utilize a multi-dimensional LUT also include at least two other processing elements. Thus, a hypothetical ICC-compatible CMS that processes only device-link profiles will be more complex than our proposed CMS.

The metric of similarity can now be defined using accepted classifications of rendering intents. Hunt [24] introduced six categories of color reproduction - spectral, exact, colorimetric, equivalent, corresponding and preferred. The first five attempt to quantify accurate reproduction of an original color in colorimetric terms while the sixth allows reproduction to be independently evaluated. Similar definitions are provided by ICC which specifies four rendering intents [25] - perceptual, saturation, media-relative colorimetric and ICC-absolute colorimetric. Fairchild [32] arranges the categories in a rough hierarchical order to define five levels of color reproduction - color, pleasing color, colorimetric, color appearance and color preference. It is stated that each level implies satisfaction of the previous level too.

Secondly, our approach allows greater flexibility in the construction (and usage) of the LUT. As we describe later in this thesis, we generate a LUT by selecting a LUT size and interpolation method, and an optimal non-uniform set of sample points. The ICC standard only allows interpolation on a uniformly sampled data set. Our method allows trade-offs between accuracy, complexity, and required memory.

Thus, our end-to-end design can be more accurate and require fewer resources than a profile-based approach. Since our proposed CMS requires only table look-ups and interpolations, it is suitable for hardware and embedded implementations. For example, one could design a display device with an embedded color management capability because of the simplicity of our proposed system.

The measures of similarity used in this project are defined as follows:

- **Colorimetric matching** requires chromaticities and relative luminance (or equivalently, all of CIE XYZ values) of the color displayed on the viewing device to match that of color displayed on the reference device. Such a matching is susceptible to changes in the ambient lighting and discounts human eye's adaptability to viewing conditions. This metric is useful when the two devices to be matched are intended to be seen in identical lighting conditions. For example, when two displays are viewed side-by-side, it might be necessary to match them in a colorimetric sense.
- **Perceptual matching** is said to be achieved when two colors match in the CIE LAB space after accounting for chromatic adaptation of the human visual system [26] and ensuring a common white point. We select perceptual matching because it is more suitable for reproduction of motion content on systems with high dynamic ranges, and under different viewing conditions [25]. This intent is particularly useful when the devices to be matched have differing gamuts, maximum luminances, and contrasts.

## 1.2 Visual Surveillance

The advances in analytics systems, availability of inexpensive sensors, and the growing need for preventive technology have made surveillance a powerful tool for the law enforcement and security personnel. In particular, video surveillance is ubiquitous in office buildings, roads and intersections, and public transport systems. While image analysis techniques are used to incorporate some level of automation into such systems, most video surveillance systems simply record the video data in a passive fashion. This video data could either be monitored by a human supervisor, or be retrieved later when needed.

While human-operated surveillance systems have the advantage of utilizing the expertise and judgement of a human being, such systems have significant issues related to human error. The problems associated with these surveillance systems have been



Fig. 1.8. A typical surveillance control room has too many video feeds for the human operator(s) to monitor. Image courtesy: Springer [36].

researched and well-documented [33,34]. Two issues for which image analysis could provide solutions are loss of efficiency due to fatigue and monotony, and lack of attention due to distraction from less important information. The task of monitoring surveillance footage is monotonous primarily because for most of the time, there is no “significant” activity occurring. Similarly, an operator trying to study every subject in a video sequence may be unnecessarily fatigued. Once the task is extended to multiple video feeds, there is simply too much information for the operator (Figure 1.8). It is widely accepted that the ability to monitor video feeds drops significantly beyond 3 – 4 feeds [35]. Thus, a trespass may go undetected while the operator is randomly scanning a parking lot or an empty hallway.

Traditionally, research in computer vision has been aimed at developing methods and systems that mimic human vision [37]. However, even with the state-of-the-art techniques, a completely autonomous surveillance system that can work in any deployment scenario has not been claimed [36,38]. Dee and Velastin [38] also illustrate

situations where vision based methods can exceed human performance. Therefore, an “intelligent” surveillance system would not replace human operators but would function in a complementary manner to maximize the overall effectiveness of the system. In this thesis, we look at the problem of visual surveillance of vehicles for the purpose of detecting anomalies. The goal is to perform the analyses without obstructing the traffic flow, and to involve a human operator for making only the highest level decisions (for example, stopping a suspicious vehicle for further inspection).

Due to the increase in volume of vehicles on the road, traffic monitoring has become both important and difficult. A sound traffic monitoring system together with an adequate response process would not only make the roads safer but can also potentially disrupt criminal and/or terrorist activities. Again, most technology solutions for vehicular monitoring involve significant human supervision or are passive tools for forensic application. During the last ten years, several efforts have been made to develop vision-based vehicle surveillance systems.

Coifman et al. proposed a vision-based surveillance system with a goal of assisting traffic flow management [39]. Their system would detect and track the vehicles, and determine certain parameters deemed useful for the purpose of traffic management. As part of the Video Surveillance and Monitoring (VSAM) project [40], researchers at Carnegie Mellon University developed a system to detect, track, and classify objects as vehicles and humans. They also designed methods for detecting simple activities and interactions between human subjects. The European AVITRACK project was aimed at developing methods for monitoring activities of service vehicles and personnel on airport aprons [41]. This would enable safer and more efficient servicing of aircrafts. Li and Porikli described a method for estimating traffic conditions using Gaussian Mixture Hidden Markov Models [42]. The IBM Smart Surveillance System (S3) detects and tracks objects and has the capability for vehicle license plate recognition and detection of user-defined events [43]. In an alternative deployment scenario, Gutchess et al. propose a surveillance system to monitor vehicles and persons in a parking lot [44]. Their system can detect and track objects and study the interaction



of humans and vehicles. The system also offers nighttime operation capability by utilizing visible and near-infrared imagery. Some other interesting surveillance systems and technologies have been reported; these will be referenced in later sections.

Our work is different from these previous (or ongoing) projects because we are mainly interested in detecting anomalous events in vehicular traffic that pose potential risk to public and infrastructure safety. Therefore, we do not compute parameters related to general traffic conditions. Furthermore, we do not consider human-vehicle interaction with the exception of the vehicles' occupants. This is justified because a core requirement for our system is the ability to observe the traffic without affecting the flow. Since the observed vehicles are almost always in motion the chances of human-vehicle interaction are reduced to a minimum. Unlike most other cases where one or more cameras are used to obtain similar views of the subjects, we use a novel two orthogonal camera configuration to obtain the front and side views of the vehicles simultaneously. This allows us to extract certain information in more efficient ways, and other information (like tire size) which would otherwise not be possible. The capabilities of this system can be further extended by addition of other non-video sensors in a synergic manner.

### 1.3 Contributions of this Thesis

It is important to ensure faithful reproduction of visual data when a digital image/video is viewed on a user's media. This is particularly valuable for professional photography, cinema and television programming, and printing. In this thesis, we propose a novel approach to color management using device models and look-up tables. The main contributions are as follows:

- We studied the traditional color profile based approach to color management and identified the limitations which render it less suitable for the newer requirements in color management. We proposed a simpler color management system which would use only look-up tables which are created offline. Such a system would

be more hardware friendly and hence, more suitable for viewing content on low power devices.

- We constructed a transformation based on device models for perceptually matching two display devices.
- We approximated the above transformation using 3D LUT. We further identified methods to optimally select LUT parameters (such as sample points) in a model-based optimization framework under specified resource constraints.
- We compared the optimal LUT based color management system with an ICC profile based system in terms of color reproduction accuracy and memory requirements. It was shown that the proposed system was more accurate and memory efficient.

Surveillance systems are deployed in many countries to assist law enforcement and public safety. Image and video analysis techniques are being designed to extract information from the videos. Such automated processing would help the operators and reduce human errors. In this thesis, we describe a visual surveillance system for monitoring vehicles and detecting potential anomalies. The main contributions are as follows:

- We proposed a surveillance system with two orthogonal cameras which would image approaching vehicles from the front and the side without obstructing the traffic.
- We developed low complexity methods to extract physical information about each vehicle (such as body type and make). These would enable the system to estimate the normal ranges of dynamic measurements and help an operator identify the vehicle.
- We proposed methods for detecting anomalous behavior based on dynamic trajectory measurements (such as velocity). Toward this goal, we devised a coordinate system which compensates for the road curvatures. We developed

methods to detect unexpected changes in the velocity and to identify significant maneuvers (e.g. left turns and u-turns).

- We proposed a color correction technique which would make the color features more reliable across multiple cameras under different illuminations. This technique was shown to improve the robustness of a color based object tracker.

#### 1.4 Publications Resulting from this Work

##### Journal Papers:

1. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, “Color Management Using Optimal Three Dimensional Look-Up Tables,” *Journal of Imaging Science and Technology*, vol. 54, no. 3, May-June 2010, pp. 030402 (1-14).
2. **Satyam Srivastava** and Edward J. Delp, “Autonomous Visual Surveillance of Vehicles for the Detection of Anomalies,” *IEEE Transactions on Intelligent Transport Systems*, submitted.

##### Conference Papers:

1. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, “Co-Ordinate Mapping and Analysis of Vehicle Trajectory for Anomaly Detection,” *Proceedings of the IEEE International Conference on Multimedia and Expo*, Barcelona, Spain, July 2011 (to appear).
2. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, “Color Correction for Object Tracking Across Multiple Cameras,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, May 2011 (to appear).

3. **Satyam Srivastava** and Edward J. Delp, “Standoff Video Analysis for the Detection of Security Anomalies in Vehicles,” *Proceedings of the IEEE Applied Imagery Pattern Recognition Workshop*, Washington DC, October 2010.
4. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, “Generating Optimal Look-Up Tables to Achieve Complex Color Space Transformations,” *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 1641-1644.
5. Thanh H. Ha, **Satyam Srivastava**, Edward J. Delp, and Jan P. Allebach, “Model Based Methods for Developing Color Transformation Between Two Display Devices,” *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 3793-3796.
6. Thanh H. Ha, **Satyam Srivastava**, Edward J. Delp, and Jan P. Allebach, “Monitor Characterization Model Using Multiple Non-Square Matrices for Better Accuracy,” *Proceedings of the IS&T Color Imaging Conference*, Albuquerque, New Mexico, USA, November 2009, pp. 117-122.
7. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, “Color Management Using Device Models and Look-Up Tables,” *Proceedings of the Gjøvik Color Imaging Symposium*, Gjøvik, Norway, June 2009, pp. 54-61.
8. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, “Crowd Flow Estimation Using Multiple Visual Features for Scenes with Changing Crowd Densities,” *Proceedings of the IEEE International Conference on Advanced Video and Signal based Surveillance*, Klagenfurt, Austria, 2011, submitted.

## 2. COLOR MANAGEMENT OF DISPLAY DEVICES

Digital imaging has seen an unprecedented growth in the past five years. The enormous variety of imaging systems available to users to create and view visual data is quite large. Color management has become an important aspect of modern imaging and display systems. Color profiles have been the de facto tool for achieving faithful visual reproduction for a long time. In this chapter, we discuss issues associated with profile-based color management systems. We describe an alternative approach motivated by the problem of visually matching two known display devices. We use a model-based method to achieve this goal and propose realizing the method with simple table look-up operations. We devise a framework for designing look-up tables (LUT) which are optimal in terms of color reproduction on the displays based on resource constraints.

### 2.1 Color Management Using Device Models

This section describes the process by which we construct an analytical transformation from the color space of a reference device to the color space of the viewing device with the goal of achieving visual similarity. This consists of objective measurement of colors as displayed on the devices in order to obtain important information about the devices and utilizing this information for constructing mathematical models. These models allow us to relate the visual output of a device with the digital inputs and hence, to devise ways to adjust the visual output by controlling the digital inputs.

### 2.1.1 Measurements

It is clear that our experiments for building the device models would require measurement of colors as displayed on the monitors. Some information that might be needed may fall under the following categories:

1. The gamma values for each channel is needed to be able to convert the native (non-linear) RGB values to linear RGB values which serves as a bridge before going to a device independent space.
2. The transformation matrix  $M$ , as described in the nomenclature section, is used to transform linear RGB to CIE XYZ space.
3. The white point (in CIE XYZ) of each monitor is required for interpreting the colors in CIE LAB space.

It is obvious that accurate measurement of color displayed on the monitors is central to the process of building device models and transforming between the color spaces. The measurements require providing desired color inputs to the monitors and measuring the visual stimulus in XYZ. For better accuracy, all experiments related to color measurements were conducted in total darkness. There are many desktop color scanners that measure the image color but they have limited accuracy.

### PR-705

Photo Research PR-705 SpectraScan System is a lab-grade fast scanning spectroradiometer widely used for radiometric applications (Figure 2.1). It can accurately measure the intensity and color of light sources and display the measurements in several formats. For our experiments, we used PR-705 to measure the color of the image (a constant color patch) displayed on the above-mentioned monitors in CIE XYZ. PR-705 can be used with manual trigger and the output can be obtained into a software interface or read off from the rear LCD screen. It also allows remote mode operation by communicating with a computer over an RS 232 channel.



Fig. 2.1. Photo Research PR-705.

It is also very common to use the instrument for making multiple measurements to observe a temporal distribution or record measurements for a series of controlled time-varying inputs. PR-705 comes with an accompanying software, called SpectraWin, which allows remote measurement using a window based user interface which looks like Figure 2.2. It also allows specifying number of measurements to be taken and time interval between consecutive measurements. For the purpose of these experiments, if there was a method to display a single color image full-screen on the monitors, the self timed measurements could, at least in theory, be used.

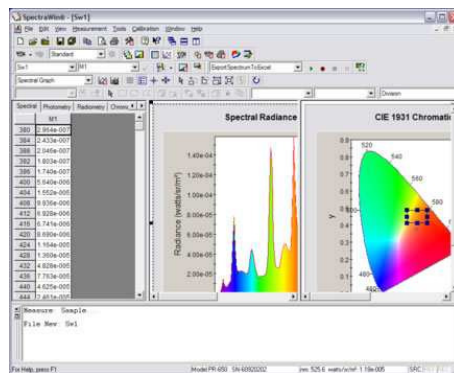


Fig. 2.2. A screen shot of SpectraWin.

Whereas PR-705 allows multiple modes of data recording - actual spectral content and measurement in XYZ/xy/Yxy - construction of device models requires only the

data measured in CIE XYZ (in candella per square meter,  $cd/m^2$ ) with a CIE 1931 2° Standard Observer [3]. It is also important to maintain a common spatial configuration throughout and across experiments which include maintaining placement of PR-705 at a fixed distance from the monitors, orienting it perpendicular to and focussed at the center of the monitors and using a constant aperture size.

### **Automatic measurement tool**

Complications arise as soon as we realize that PR-705 requires unequal amounts of time for measuring different colors and for differing light intensities. Whereas a bright display takes under five seconds, a black (monitor's flare)<sup>1</sup> screen could take as long as ten minutes! It immediately becomes obvious that the recurring measurement approach, based on constant time intervals, stated above cannot be used. If the time interval is set to too small, some colors would not get sufficient time to be measured and lead to an error. On the other hand, setting the interval very high would cause unmanagably too many measurements for brighter colors and the experiment time to blow up hugely. A good solution would be to find a means to synchronize PR-705 trigger with the change of color on the screen.

PR-705 can communicate with a computer over an RS 232 channel which is the underlying principle behind the operation of SpectraWin software. Using the instruction set, a custom software master can trigger the instrument remotely to initiate measurements or request previously recorded measurements. But unlike Spectrawin, these triggers and data requests need not be time-based. Instead these were made event-triggered for our purpose.

MATLAB's Instrument Control Toolbox was used for providing RS 232 protocol compatibility whereas a MATLAB wrapper controlled image display and PR-705 instruction selection (which would be put on the serial port by the toolbox). Full

---

<sup>1</sup>Flare is the background illumination on a display when a black input (for example (0,0,0) in native RGB) is applied to it.



screen<sup>2</sup> images were displayed using a Java-based MATLAB freeware. The MATLAB wrapper then waits for a pre-defined time to allow monitor stabilization before triggering a measurement in PR-705. Once the instrument acknowledges completion of the measurement, the recorded data is dumped into a data file. This custom measurement tool was used in all subsequent experiments.

### 2.1.2 Models for prediction

*The work presented in this section was done by the author jointly with Thanh H. Ha, see [45] for a more complete description of his work. It is included here to make the presentations easier to understand.*

Let the problem be defined as follows: We have two monitors which we call the Target Device and the Viewing Device. These monitors might differ from each other in screen gamma values, gamut, white point and other properties. We wish to develop a color transformation from the non-linear RGB space of the Target Device to the non-linear RGB space of the Viewing Device such that a visual match between them can be achieved. A correct white point rendering is also a desirable goal of this color transformation. In this section, we will describe the methods for developing such a desired color transformation. Specifically, in Section 2.1.2 and Section 2.1.2, we will describe two different models for achieving a visual match when the two monitor are viewed side by side (resulting in a colorimetric match), while in Section 2.1.2 and Section 2.1.2, we will discuss two other models for achieving a visual match when the two monitors are viewed in the different viewing conditions (matching in a perceptual sense).

---

<sup>2</sup>Following the accepted practice in the field of monitor calibration, we illuminate only 50% of the screen area with the desired color and keep the background at 10% gray (for example (25,25,25) in native RGB).

## Non-linear transformation model for colorimetric matching (NLX)

Let us assume that the two monitors are viewed side by side. We approach the problem of achieving a colorimetric match between the two monitors by matching their displayed colors in the device-independent XYZ space. This approach splits the problem into two sub-problems where one is the inverse of the other:

- Sub-problem 1: To predict the XYZ value of a displayed color when we know the non-linear RGB value which is used to drive the monitor.
- Sub-problem 2: To predict the non-linear RGB value which should be used to drive a monitor so that a color with a known XYZ value is displayed.

We solve these two sub-problems by developing two types of models. We call the model for solving sub-problem 1 the “forward characterization model” (or forward monitor model), and the one for solving sub-problem 2 the “inverse characterization model” (or inverse monitor model). If we ignore the gamut mismatch issue between the two monitors, by cascading the forward monitor model of the target device and the inverse monitor model of the viewing device, we can achieve a colorimetric match between the two monitors. Figure 2.3 depicts the overall non-linear transformation developed based on this approach. Throughout this report, we will denote this overall non-linear transformation model for a colorimetric match as NLX. In Sections 2.1.2 and 2.1.2, we will discuss the development of the forward monitor model and the inverse monitor model in more details.

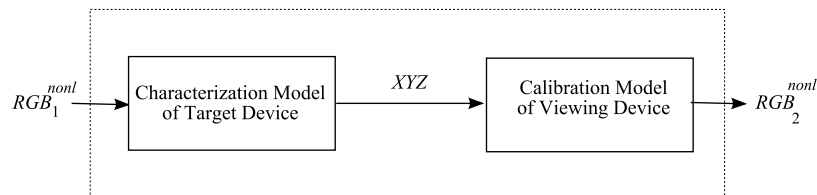


Fig. 2.3. A block diagram of the NLX model.

### The forward monitor model

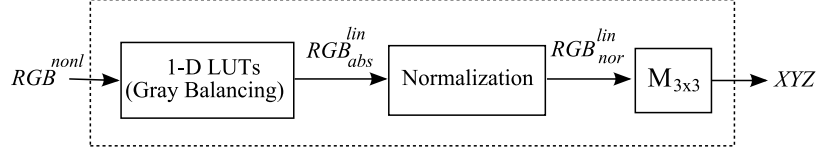


Fig. 2.4. A block diagram of the forward monitor model.

Figure 2.4 shows the structure of the forward monitor model. In the first step, an input non-linear RGB value (denoted as  $RGB^{nonl}$ ) is converted to an absolute linear RGB (denoted as  $RGB_{abs}^{lin}$ ), whose three component values are linear to the light intensity. We call this conversion the “gray balancing” process. In the second step,  $RGB_{abs}^{lin}$  is normalized with respect to the native monitor white as shown in (2.1):

$$R_{nor}^{lin} = \frac{R_{abs}^{lin}}{R_{abs}^{lin w}}; G_{nor}^{lin} = \frac{G_{abs}^{lin}}{G_{abs}^{lin w}}; B_{nor}^{lin} = \frac{B_{abs}^{lin}}{B_{abs}^{lin w}}; \quad (2.1)$$

where  $R_{abs}^{lin w}$ ,  $G_{abs}^{lin w}$ , and  $B_{abs}^{lin w}$  are three components of the absolute linear RGB value of the native monitor white.

Finally, the predicted XYZ value is computed by multiplying  $RGB_{nor}^{lin}$  by a  $3 \times 3$  non-singular transformation matrix  $\mathbf{M}$ :

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} & & \\ & \mathbf{M}_{3 \times 3} & \\ & & \end{bmatrix} \begin{bmatrix} R_{nor}^{lin} \\ G_{nor}^{lin} \\ B_{nor}^{lin} \end{bmatrix} \quad (2.2)$$

There are two main components in the forward monitor model: the gray balancing module and the  $3 \times 3$  matrix  $\mathbf{M}$  which converts  $RGB_{abs}^{lin}$  to XYZ. The gray balancing step is often implemented by fitting a single gamma curve model to the mapping between the non-linear and linear values of each component of the input RGB [46–48]. However, as stated in [47], a single curve model generally cannot fit well this mapping for the whole range of the input. Consequently, it causes some high errors in the low luminance regions. To avoid this issue, in our method, instead of using a gamma curve model, we use an 1-D Look-up table (LUT) to convert the non-linear values to

the linear values for each channel. Let us assume that the monitor has three 8-bit channels: R, G, and B. To populate a 1-D LUT for a channel, we generate a ramp of 65 levels obtained by uniformly sampling the corresponding channel, while setting all components of the other two channels to 0. We then display the ramp on the monitor which is to be characterized and measure the corresponding luminance Y values. The measured Y values contribute 65 entries to the 1-D LUT. The remaining entries of the LUT are computed by using spline-interpolation [49].

Now, we would like to discuss how to obtain the matrix  $\mathbf{M}$ . In [50], the author discusses a normalized primary matrix (NPM), whose three columns are the normalized XYZ values of the three monitor primaries, for transforming the normalized RGB to the normalized XYZ values. Although the development of the NPM is quite simple, the accuracy of the transformation using the NPM is not as good as we wish. This is because the approach is based on some assumptions such as channel independence and the primary chromaticity constancy which do not always hold true. In our research, we adopt a method similar to those in [47, 51] to obtain  $\mathbf{M}$ . Our method can be described as follows: starting with the primary matrix [50] as an initialization, we use the Patternsearch algorithm [52] to optimize nine entries of  $\mathbf{M}$  to minimize a cost function that reflects the  $\Delta E$  averaged over a training set. The training set contains the pairs of the RGB values taken from a  $6 \times 6 \times 6$  grid uniformly spanning the entire RGB cube and the corresponding XYZ values measured using a PR-705. During the optimization process, we set a constraint to ensure that the white color is transformed without error. The optimization process can be formulated as follows:

$$\mathbf{M}^{\text{opt}} = \underset{\mathbf{M}}{\text{argmin}} \sum_{i \in \Omega} \Delta E \left( LAB_i^{\text{ref}}, F_{XYZ \rightarrow LAB} \left( T_{RGB^{\text{nonl}} \rightarrow XYZ} (\mathbf{M}, RGB_i^{\text{nonl}}) \right) \right) \quad (2.3)$$

$$\text{Subject to } \begin{bmatrix} & & \\ & \mathbf{M}_{3 \times 3} & \\ & & \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} X_w^{\text{ref}} \\ Y_w^{\text{ref}} \\ Z_w^{\text{ref}} \end{bmatrix} \quad (2.4)$$

where,  $\mathbf{M}^{\text{opt}}$  is the optimal  $3 \times 3$  matrix,  $\Omega$  is the index set of the samples in the training set,  $F_{XYZ \rightarrow LAB}$  is the non-linear conversion from XYZ to LAB with respect to

the native monitor white,  $T_{RGB^{nonl} \rightarrow XYZ}$  is the forward monitor model,  $LAB_i^{ref}$  is the LAB value we obtain by converting from the measured  $XYZ_i$  value corresponding to the input  $RGB_i^{nonl}$ ,  $\Delta E(\cdot, \cdot)$  is the operator for computing the difference in  $\Delta E$  units between two LAB values. In the constraint equation, the  $XYZ_w^{ref}$  is the measured XYZ value of the native monitor white.

### The inverse monitor model

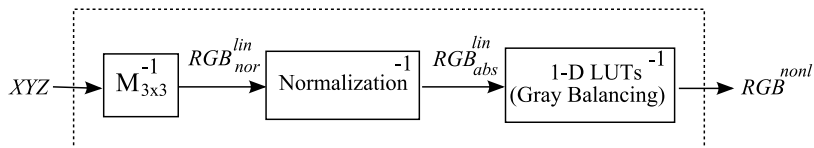


Fig. 2.5. A block diagram of the inverse monitor model.

The objective of the inverse monitor model is to predict the non-linear RGB value that should be used to drive a monitor so that a color with a given XYZ will be displayed. It is obvious that the inverse monitor model is obtained by simply inverting the forward monitor model shown in Figure 2.4. Figure 2.5 shows the block diagram of the inverse monitor model obtained in this way.

Once we have the forward monitor model of the target device and the inverse monitor model of the viewing device available, the NLX is developed by combining these two monitor models together as shown in Figure 2.3. In the next section, we will show the experimental results of the NLX model.

### Experimental results

Two different 24-inch LCD displays, CTLLMO-1 and CTLCSO-1, were used in this experiment. Among the two displays, CTLLMO-1 serves as the target device while CTLCSO-1 serves as the viewing device. First, we developed the forward monitor model for CTLLMO-1, and then the inverse monitor model for CTLCSO-1. In fact, the inverse monitor model for CTLCSO-1 was developed by first developing the forward monitor model for CTLCSO-1 and then inverting this forward model. Fi-

nally, we combined both models to construct the NLX according to the block diagram in Figure 2.3.

Figure 2.6(a) and Figure 2.6(b) show the plots of the 1-D LUTs for gray balancing of CTLLMO-1 and CTLCSO-1, respectively.

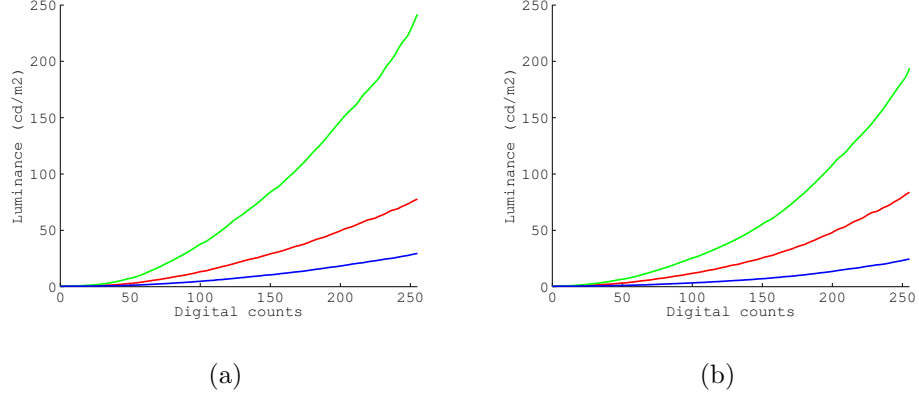


Fig. 2.6. 1D gamma LUTs for the three channels of (a) CTLLMO-1, (b) CTLCSO-1.

For each monitor, the training set for developing the matrix  $\mathbf{M}$  in the forward monitor model was generated by displaying an evenly-spaced  $6 \times 6 \times 6$  grid which spans the entire RGB space, on the monitor, and then measuring the corresponding XYZ values using the PR-705. The testing set was generated in the same way but with an evenly-spaced  $5 \times 5 \times 5$  grid. There was no overlap between the training and testing sets. Pattern Search algorithm [52] was used for the optimization problem (2.4). Table 2.1 summarizes the testing error statistics of the forward models for CTLLMO-1 and CTLCSO-1 monitors, respectively. It also shows that the matrix  $\mathbf{M}$  computed by our optimization process outperforms the matrix computed using the method in [50].

After building the NLX according to the block diagram in Figure 2.3, we tested the NLX model using the following procedure:

1. Display a  $5 \times 5 \times 5$  grid, which spans the entire RGB cube, on CTLLMO-1 and measure its corresponding XYZ values. Let us call this XYZ set “SET 1.”

Table 2.1  
Testing error statistics (in  $\Delta E$  units) of the monitor characterization models.

Statistic\Using	Monitor CTKLMO-1		Monitor CTLCSO-1	
	Primary Matrix	Optimal Matrix	Primary Matrix	Optimal Matrix
Mean	5.18	2.38	4.32	2.11
Max	9.76	6.59	7.76	5.46
Min	0.44	0.3	1.11	0.33
Median	5.21	2.25	4.42	2.15
1-Stdev	2.01	1.07	1.82	0.84

2. Use the NLX model to transform the grid in step 1 to the non-linear RGB space of CTLCSO-1.
3. Display the transformed grid on CTLCSO-1 and measure its XYZ values. Let us call this XYZ set "SET 2."
4. Finally, compute the differences in  $\Delta E$  units between SET 1 and SET 2.

Table 2.2  
Testing error statistics (in  $\Delta E$  units) of the NLX model.

	NLX
Mean	1.65
Max	3.93
Min	0.43
Median	1.48
1-stdev	0.74

The testing error statistics in  $\Delta E$  units are shown in Table 2.2. The distribution of outliers (whose errors are greater than the mean error plus one standard deviation) in the LAB space is shown in Figure 2.7. Red circles are the points in SET 1, blue crosses are the points in SET 2 and the Euclidean distance between them represents the deviation in  $\Delta E$  units. As shown in Table 2.2 and Figure 2.7, the NLX performs quite well to match between two non-linear RGB spaces of the two monitors.

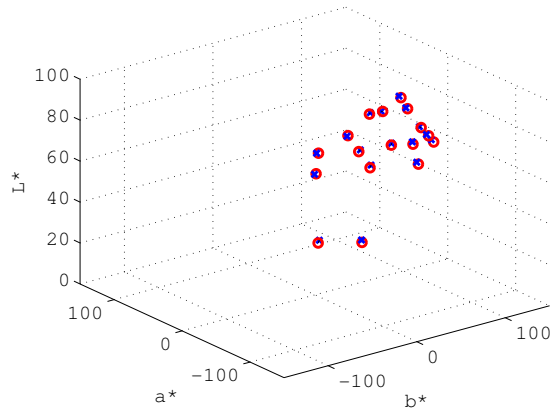


Fig. 2.7. Distribution of the  $1 - \sigma$  error outliers of the NLX model in LAB space.

### Linear transformation for colorimetric matching (LX)

Although the NLX model described in the previous section performs very well to visually match the non-linear RGB spaces of the two monitors, it is expensive to implement. The goal of this exercise is to investigate how well the NLX could be approximated by a simple linear transformation. Throughout the report, we will denote this linear transformation as LX. The equation of LX is shown in (2.5):

$$\begin{bmatrix} R_2^{nonl} \\ G_2^{nonl} \\ B_2^{nonl} \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} \begin{bmatrix} R_1^{nonl} \\ G_1^{nonl} \\ B_1^{nonl} \end{bmatrix} \quad (2.5)$$



where subscripts 1 and 2 imply the nonlinear RGB values in the target device and viewing device, respectively and  $\mathbf{M}_{LX}$  is a  $3 \times 3$  non-singular matrix.

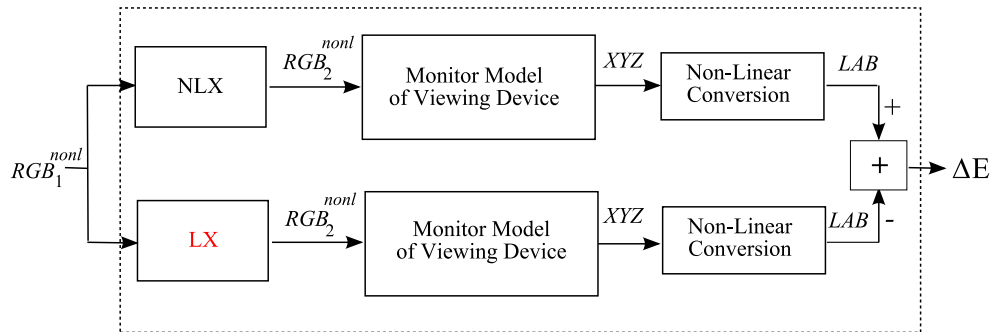


Fig. 2.8. A framework for developing the LX model.

Figure 2.8 shows the framework for developing the LX model. As shown in the figure, the NLX model serves as the reference to train the LX model. The problem now is formulated as a constrained optimization problem whose cost function is the  $\Delta E$  units averaged over a training set. The training set is a  $7 \times 7 \times 7$  grid uniformly spanning the entire RGB cube. During the optimization process, a constraint is set to ensure that the output  $RGB_2^{nonl}$  will fall within the gamut of the Viewing Device. Again, Patternsearch algorithm [52], a derivative-free algorithm, is used for the optimization process.

### Experimental results

We also tested the LX model with CTLLMO-1 and CTLCSO-1 monitors. However, the experiment now was based on the simulated data using the NLX model. In other words, instead of using the actual monitors, we used their forward monitor models for the experiment. Although this method is weaker than that in Section 2.1.2, this experiment is much simpler and the obtained results are reliable since the monitor models of CTLLMO-1 and CTLCSO-1 are reasonably accurate. The following procedure was used to test the LX model:

1. Use the forward monitor model developed for CTLLMO-1 to predict the XYZ values of the displayed colors when CTLLMO-1 was driven with a  $5 \times 5 \times 5$  grid

uniformly spanning the entire RGB cube of CTLLMO-1. Let us call the predicted XYZ values SET 1'.

2. Use the LX model to transform the grid in Step 1 to the non-linear RGB space of CTLCSO-1.
3. Use the forward monitor model for CTLCSO-1 to predict the XYZ values corresponding to the transformed grid input. Let us call these XYZ values SET 2'.
4. Finally, compute the differences in  $\Delta E$  between SET 1' and SET 2'.

Figure 2.9 shows the error distribution of the (one standard deviation) outliers of the LX model in the LAB space. Table 2.3 summarizes the error statistics of the LX model.

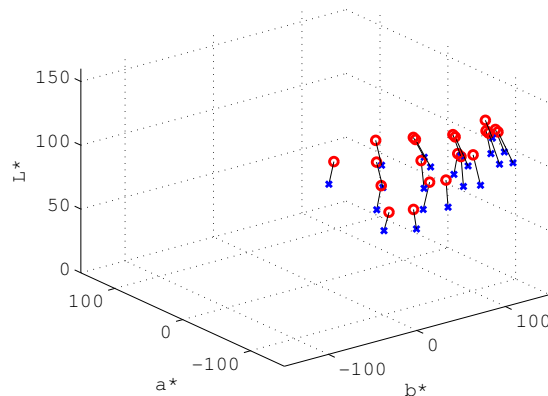


Fig. 2.9. Distribution of the  $1 - \sigma$  error outliers of the LX model in LAB space.

Comparing with Table 2.2, the LX model is far from being able to approximate the NLX model. This is understandable since the LX model is too constrained to map between two non-linear RGB spaces - a highly non-linear transformation. However, in some applications when the reproduction accuracy is not as important as the implementation simplicity, the LX model may be useful. In this situation, a solution

Table 2.3  
Testing error statistics (in  $\Delta E$  units) of the LX model.

	LX
Mean	11.44
Max	37.62
Min	1.46
Median	10.01
1-stdev	6.68

to improve the LX's performance is to incorporate some sort of weighting into the cost function so that the important colors such as memory colors or neutrals will be reproduced with greater accuracy. Also the experimental results essentially suggest that in the future we develop a piece-wise linear mapping between the two non-linear RGB spaces.

### **Overall non-linear transformation for perceptual matching with white point correction (NLXWP)**

In this section, we will consider the case when the two monitors are viewed in different viewing conditions. It is well-known that two colors with the same XYZ values will look different in different viewing conditions [32, 53]. Therefore, the NLX and LX models that yield a colorimetric match by achieving the same XYZ tristimulus will fail in this case. To achieve a visual match, we need a model which takes into account the chromatic adaptation ability of the human visual system [53]. The chromatic adaptation enables the human visual system to discount the illumination, and to approximately preserve the appearances of the objects. Once a chromatic adaptation model (CAM) is available, it is easy to extend it to a chromatic adaptation

transformation (CAT) model which predicts the necessary XYZ values such that a perceptual match is achieved across two different viewing conditions.

Let us assume that the monitors are viewed in the two different darkened rooms and thus, the human visual system completely adapts to the corresponding monitor's white point. We call the viewing condition of the target device the "Viewing Condition 1," and that of the viewing device the "Viewing Condition 2." According to [2], a CAT based on Von-Kries model [26] can be constructed as follows:

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \mathbf{H}^{-1} \begin{bmatrix} \frac{L_{w2}}{L_{w1}} & 0 & 0 \\ 0 & \frac{M_{w2}}{M_{w1}} & 0 \\ 0 & 0 & \frac{S_{w2}}{S_{w1}} \end{bmatrix} \mathbf{H} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix}; \quad (2.6)$$

where subscripts  $i = 1, 2$  imply the viewing conditions of the target device and viewing device, respectively;  $LM S_{wi}$  is the LMS cone responses [54] to the reference white  $w_i$  in Viewing Condition  $i$ ;  $\mathbf{H}$  is a non-singular  $3 \times 3$  matrix which transforms the XYZ values to the LMS cone responses:

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \mathbf{H} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (2.7)$$

Let us assume further that the reference white  $w_i$  of the two monitors have the same chromaticity coordinates denoted as  $xy_w$ . Their corresponding XYZ values are computed as follows:

$$\begin{bmatrix} X_{w1} \\ Y_{w1} \\ Z_{w1} \end{bmatrix} = Y_{w1} \begin{bmatrix} \frac{x_w}{y_w} \\ 1 \\ \frac{(1-x_w-y_w)}{y_w} \end{bmatrix} \quad (2.8)$$

$$\begin{bmatrix} X_{w2} \\ Y_{w2} \\ Z_{w2} \end{bmatrix} = Y_{w2} \begin{bmatrix} \frac{x_w}{y_w} \\ 1 \\ \frac{(1-x_w-y_w)}{y_w} \end{bmatrix}. \quad (2.9)$$

Thus,

$$\begin{bmatrix} X_{w2} \\ Y_{w2} \\ Z_{w2} \end{bmatrix} = \left( \frac{Y_{w2}}{Y_{w1}} \right) \begin{bmatrix} X_{w1} \\ Y_{w1} \\ Z_{w1} \end{bmatrix}. \quad (2.10)$$

Under this case, from (2.7) and (2.10), we can easily simplify (2.6) to (2.11):

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \left( \frac{Y_{w2}}{Y_{w1}} \right) \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix}; \quad (2.11)$$

where,  $Y_{wi}$ ,  $i = 1, 2$  is the luminance of the reference white of Viewing Condition  $i$ . Because two monitors might have different gamut, a gamut mapping module that addresses this issue is necessary. Generally, gamut mapping algorithms are often implemented in the perceptually uniform color spaces. In this research, we choose LAB as the intermediate space. The equation for the conversion from the XYZ space to the LAB space with respect to the reference white  $XYZ_w$  is shown in (2.12):

$$\begin{aligned} L^* &= 116f\left(\frac{Y}{Y_w}\right) - 16 \\ a^* &= 500\left(f\left(\frac{X}{X_w}\right) - f\left(\frac{Y}{Y_w}\right)\right) \\ b^* &= 200\left(f\left(\frac{Y}{Y_w}\right) - f\left(\frac{Z}{Z_w}\right)\right) \end{aligned} \quad (2.12)$$

where

$$f(x) = \begin{cases} x^{\frac{1}{3}} & x > 0.008856 \\ 7.787x + \frac{16}{116} & otherwise \end{cases} \quad (2.13)$$

From (2.10), (2.11), and (2.12), it is easy to show that if we convert the  $XYZ_2$  computed by (2.11) to LAB with respect to  $XYZ_{w2}$ , we will get the same LAB value as we convert the  $XYZ_1$  to LAB with respect to  $XYZ_{w1}$ :

$$LAB_2 = LAB_1. \quad (2.14)$$

We use (2.14) as the basis to build up the overall color transformation between the two monitors to achieve a perceptual match. The approach is first to correct the white points of the two monitors to a common chromaticity, making them satisfy the

aforementioned common white point assumption. In a particular case, one can choose the white point of the target device to be the desired target and correct the white point of the viewing device according to it. Accordingly, we will denote the overall non-linear transformation with white point correction as NLXWP. The NLXWP is realized by a forward transformation from the non-linear RGB space of the target device to LAB, followed by an inverse transformation from LAB to the non-linear RGB space of the viewing device. The conversion between XYZ and LAB in the forward and inverse transforms is with respect to the independently corrected white points of the target device and the viewing device, respectively. The white point is preserved through the transformation by setting the constraint on the forward and inverse transformations. Figure 2.10 shows the diagram of the NLXWP model.

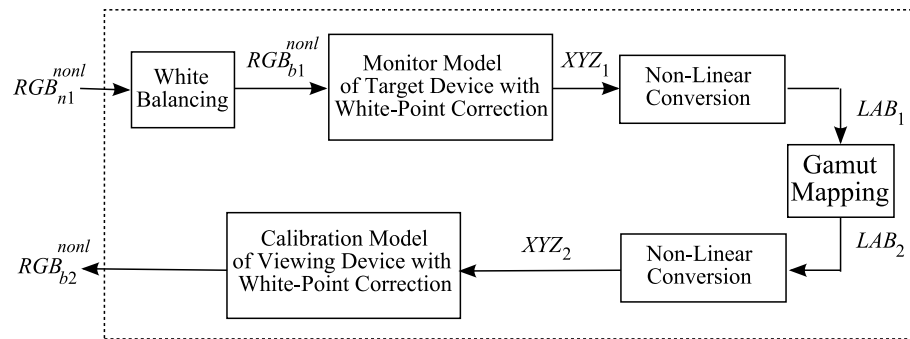


Fig. 2.10. A block diagram of the NLXWP model. Subscripts 1 and 2 imply that the values are in the viewing conditions of the Target and Viewing Devices, respectively;  $n$  and  $b$  stand for “native” and “white-balanced” spaces, respectively.

**A method for white point correction** We correct the native monitor white point to a desired chromaticity by adjusting the balance between the three color channels of the corresponding monitor. This is accomplished by two steps:

- Step 1: Determine the desirable reference white that has the desired chromaticity and the highest achievable luminance.

- Step 2: Develop a white-balancing module to transform the native monitor RGB space to a “white-balanced RGB space”. The reference white in the destination space is determined in Step 1. The goal of this white balancing process is to avoid the color cast artifact and to map the neutrals to the neutrals.

In the first step, given a desired chromaticity coordinates  $xy_{dw}$  (subscript d: desired; subscript w: white), based on the inverse monitor model described in Section 2.1.2, we can approximately compute the highest achievable luminance  $Y_{dw}$  by the monitor as follows: We first represent  $XYZ_{dw}$  in terms of  $Y_{dw}$  and  $xy_{dw}$ :

$$\begin{bmatrix} X_{dw} \\ Y_{dw} \\ Z_{dw} \end{bmatrix} = Y_{dw} \begin{bmatrix} \frac{x_{dw}}{y_{dw}} \\ 1 \\ \frac{(1-x_{dw}-y_{dw})}{y_{dw}} \end{bmatrix} = Y_{dw} \mathbf{v} \quad (2.15)$$

Also, we present the inverse matrix  $\mathbf{M}^{-1}$  which transforms XYZ to  $RGB_{nor}^{lin}$  in the following form:

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{bmatrix}; \quad (2.16)$$

where,  $\mathbf{r}_1^T$ ,  $\mathbf{r}_2^T$ , and  $\mathbf{r}_3^T$  are three row vectors of  $\mathbf{M}^{-1}$ . The three components of the normalized linear RGB value  $RGB_{nor\ dw}^{lin}$  of this desired white point is computed as follows:

$$\begin{aligned} R_{nor\ dw}^{lin} &= Y_{dw} (\mathbf{r}_1^T \mathbf{v}) = Y_{dw} k_R \\ G_{nor\ dw}^{lin} &= Y_{dw} (\mathbf{r}_2^T \mathbf{v}) = Y_{dw} k_G \\ B_{nor\ dw}^{lin} &= Y_{dw} (\mathbf{r}_3^T \mathbf{v}) = Y_{dw} k_B \end{aligned} \quad (2.17)$$

where  $k_R$ ,  $k_G$ , and  $k_B$  are the scalar values. For a given  $XYZ_{dw}$  to be achievable by the monitor, it is necessary that the three components of  $RGB_{nor\ dw}^{lin}$  be in the range  $[0, 1]$ . From (2.17) we have:

$$\begin{aligned} 0 &\leq Y_{dw} k_R \leq 1 \\ 0 &\leq Y_{dw} k_G \leq 1 \\ 0 &\leq Y_{dw} k_B \leq 1 \end{aligned} \quad (2.18)$$

Finally, the maximum achievable luminance  $Y_{dw}^{\max}$  with this fixed  $xy_{dw}$  can be determined as:

$$Y_{dw}^{\max} = \min \left( \frac{1}{k_R}; \frac{1}{k_G}; \frac{1}{k_B} \right) \quad (2.19)$$

Using the inverse model of the monitor, we can compute further the absolute linear RGB value (denoted as  $RGB_{abs\ dw}^{lin}$ ) of the desired white. Since the accuracy at this stage is limited by the accuracy of the inverse monitor model, the coarse result then is refined by a direct-search on the actual monitor. Starting with the coarse result, a software program automatically controls the PR-705 to search in the monitor color space for a color that is closest to the desired white point.

In the second step, the white balancing is implemented in the absolute linear RGB space. First, the source non-linear RGB value is converted to the absolute linear RGB space through the gray balancing process. Then, the converted absolute linear RGB value is multiplied by a diagonal matrix  $\mathbf{D}$  which is shown in (2.20):

$$\begin{bmatrix} R_b^{lin} \\ G_b^{lin} \\ B_b^{lin} \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{R_{dw}^{lin}}{R_{nw}^{lin}} & 0 & 0 \\ 0 & \frac{G_{dw}^{lin}}{G_{nw}^{lin}} & 0 \\ 0 & 0 & \frac{B_{dw}^{lin}}{B_{nw}^{lin}} \end{bmatrix}}_{\mathbf{D}} \begin{bmatrix} R_n^{lin} \\ G_n^{lin} \\ B_n^{lin} \end{bmatrix} \quad (2.20)$$

where subscript “ $b$ ” implies white-balanced; “ $dw$ ” implies desired white; “ $nw$ ” implies native monitor white and superscript “ $lin$ ” implies linear. Note that all values in (2.20) are in the absolute linear RGB space. Finally,  $RGB_b^{lin}$  is converted to  $RGB_b^{nonl}$  through the inverse of the gray balancing process. The block diagram of this white balancing process is shown in Figure 2.11. This process guarantees that the neutrals will be mapped to the neutrals in the normalized linear RGB space. The normalization for the white-balanced RGB values will be shown in (2.21) when we discuss the forward monitor model with the white point correction.

**Forward and inverse monitor models with white point correction** After the two monitors are white-point corrected, we need to re-develop the forward and inverse monitor models for them. The goal of this stage is to increase the prediction accuracy of the two models since now the input spaces are narrowed down due to the



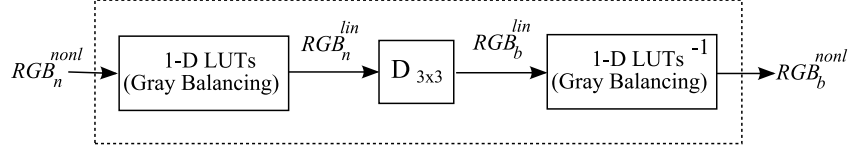


Fig. 2.11. A block diagram of the white-balancing process.

white-balancing process. The development of the forward monitor model with white point correction for a monitor is very similar to what we described in Section 2.1.2. The differences are that the inputs now are the white-balanced non-linear RGB values, and the normalization step is performed with respect to the desired white rather than the native monitor white. The normalization now is implemented using the following equations:

$$R_{nor}^{lin} = \frac{R_{abs}^{lin}}{R_{abs}^{lin dw}}; G_{nor}^{lin} = \frac{G_{abs}^{lin}}{G_{abs}^{lin dw}}; B_{nor}^{lin} = \frac{B_{abs}^{lin}}{B_{abs}^{lin dw}}; \quad (2.21)$$

where  $R_{abs}^{lin dw}$ ,  $G_{abs}^{lin dw}$ , and  $B_{abs}^{lin dw}$  are the three components of the absolute linear RGB value of the desired reference white. We also use the optimization technique as in Section 2.1.2 to develop the matrix  $\mathbf{M}$  for the forward monitor model with white point correction. However, the training set now includes the white-balanced RGB values rather than those native to the monitor. We obtain the training set by applying the white-balancing module to a  $6 \times 6 \times 6$  grid uniformly spanning the entire native RGB cube of the monitor. Once the forward monitor model with white point correction is available, the inverse model with white point correction is obtained by simply inverting the corresponding forward model.

**Gamut mapping module** An important unit of the NLXWP model is the gamut mapping module. The gamut mapping module handles the gamut mismatch issue between the two monitors. In this research, we use the straight chroma clipping technique (SCC) [55] for this module. SCC is simple and was demonstrated to perform better than some other popular clipping approaches such as closest LAB clipping, cusp clipping, node clipping [55] as well as some compression techniques [56].

In [55], SCC is implemented in the LAB space by first mapping the  $L^*$  (lightness) extremes of two gamuts to each other, and then shifting (i.e. clipping) the out-of-gamut colors to the destination gamut's boundary while still preserving  $L^*$  and  $h^*$  (hue). In our approach, the  $L^*$  extremes of the two gamuts are already equal to each other (the maximum is 100 and the minimum is 0 units). Therefore, the  $L^*$  mapping step for SCC is not necessary. This key characteristic is a good point since the sequential  $L^*$  mapping approach used in [55] has some potential problems as discussed in [57]. The SSC technique is depicted in Figure 2.12.

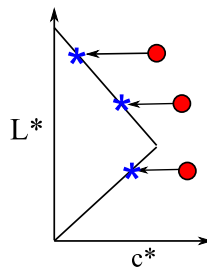


Fig. 2.12. The SSC technique for gamut mapping, red circles are out-of-gamut colors and blue stars are the mapped colors.

To clip an out-of-gamut color described by  $L^*$ ,  $h^*$ , and  $c^*$  (chroma), we fix  $L^*$  and  $h^*$ , and then use the inverse monitor model of the viewing device to determine its maximum achievable chroma (clipped  $c^*$ ). The method for determine the clipped  $c^*$  can be described in more details as follows:

For a given color described by  $L^*$ ,  $h^*$ , and  $c^*$ , the  $a^*$  and  $b^*$  coordinates in the LAB space can be computed according to (2.22):

$$\begin{aligned} a^* &= c^* \cos(h^*) \\ b^* &= c^* \sin(h^*) \end{aligned} \tag{2.22}$$

Set  $f_y = \left(\frac{L^*+16}{116}\right)$ , then the conversion from LAB to XYZ is (2.23):

$$\begin{aligned}
X &= X_w \left( \frac{a^*}{500} + f_y \right)^3 = X_w \left( \frac{c^* \cos(h^*)}{500} + f_y \right)^3 \\
Y &= Y_w f_y^3 \\
Z &= Z_w \left( f_y - \frac{b^*}{200} \right)^3 = Z_w \left( f_y - \frac{c^* \sin(h^*)}{200} \right)^3
\end{aligned} \tag{2.23}$$

where  $[X_w, Y_w, X_w]^T$  is the XYZ value of the reference white. We present the inverse matrix  $\mathbf{M}^{-1}$  in the inverse monitor model in the following form:

$$\mathbf{M}^{-1} = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix}; \tag{2.24}$$

where  $r_1^T$ ,  $r_2^T$ , and  $r_3^T$  are three row vectors of  $\mathbf{M}^{-1}$ . Then three components of the normalized linear RGB can be computed as follows:

$$\begin{aligned}
R^{lin} &= r_1^T \begin{bmatrix} X_w \left( \frac{c^* \cos(h^*)}{500} + f_y \right)^3 \\ Y_w f_y^3 \\ X_w \left( f_y - \frac{c^* \sin(h^*)}{200} \right)^3 \end{bmatrix} \\
G^{lin} &= r_2^T \begin{bmatrix} X_w \left( \frac{c^* \cos(h^*)}{500} + f_y \right)^3 \\ Y_w f_y^3 \\ X_w \left( f_y - \frac{c^* \sin(h^*)}{200} \right)^3 \end{bmatrix} \\
B^{lin} &= r_3^T \begin{bmatrix} X_w \left( \frac{c^* \cos(h^*)}{500} + f_y \right)^3 \\ Y_w f_y^3 \\ X_w \left( f_y - \frac{c^* \sin(h^*)}{200} \right)^3 \end{bmatrix}
\end{aligned} \tag{2.25}$$

For the given color to be inside the gamut of the viewing device, it is necessary that the three components of the normalized linear RGB be inside the range [0,1]:

$$0 \leq R^{lin}, G^{lin}, B^{lin} \leq 1 \tag{2.26}$$

where  $R^{lin}$ ,  $G^{lin}$ , and  $B^{lin}$  are computed by (2.25). It is obvious that deriving a closed-form equation to compute the maximum  $c^*$  satisfying (2.26) is very difficult.

Therefore in this research we use a search-based approach. The maximum  $c^*$  (or clipped  $c^*$ ) satisfying the (2.26) is determined by an optimization process as follows:

$$\begin{aligned} & \text{maximize} && f(c^*) = c^* \\ & \text{subject to} && 0 \leq R^{lin}, G^{lin}, B^{lin} \leq 1 \\ & && 0 \leq c^* \end{aligned} \tag{2.27}$$

where  $R^{lin}$ ,  $G^{lin}$ , and  $B^{lin}$  are the functions of  $c^*$  computed by (2.25). This optimization (2.27) can be easily implemented using any non-linear constrained optimization algorithm, for example the `fmincon` routine in MATLAB [58].

The  $L^*$ ,  $h^*$ , and clipped  $c^*$  will describe the clipped color. Varying  $L^*$  from 0 to 100 lightness units and  $h^*$  from 0 to 360 degree, using the approach above we can also visualize the 3-D gamut of the monitor in the LAB space. Figure 2.13 shows the gamut leaf vertically cut at  $h^*=330$  degree from the 3-D gamuts.

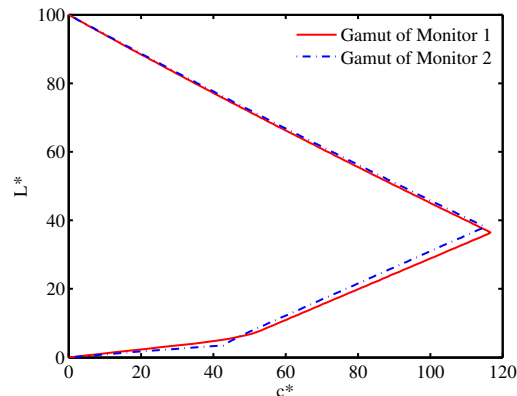


Fig. 2.13. Gamut leaf in the LAB space at  $h^*=330$  degree.

### Experimental results

The monitors used in this experiment were the two 24-inch LCD displays, BLRLMO-2 and BLRCSO-2, among which, BLRLMO-2 served as the target device; and BLRCSO-2 served as the viewing device. First, we used the method described in Section 2.1.2 to correct the white point of the two monitors to D-65 [2]. The errors in  $\Delta E$  units

between the corrected white colors of BLRLMO-2 and BLRCSO-2 and the corresponding closest colors which have D-65 chromaticity are 0.56 and 0.84, respectively. Table 2.1.2 summarizes the correction results in detail.

Table 2.4

White point correction results for the two monitors with D-65 as the target white point.

		<b>Monitor 1</b>			<b>Monitor 2</b>		
			Corrected			Corrected	
	D-65	Native	Coarse	Refined	Native	Coarse	Refined
Y		162.56	96.16	103.98	268.24	220.01	218.78
x	0.3127	0.3687	0.3052	0.3130	0.2902	0.3157	0.3127
y	0.3290	0.3761	0.3220	0.3300	0.3256	0.3346	0.3303

Then we built the white balancing module for BLRLMO-2 according to the diagram in Figure 2.11. In the next step, we developed the forward model with white point correction for BLRLMO-2 and the inverse model with white point correction for BLRCSO-2. Table 2.5 summarizes the testing errors of the characterization models with white point correction for the two monitors.

Finally, we combined these two developed models to construct the NLXWP model according to Figure 2.10. To test the NLXWP model, we used the following procedure:

1. Generate a  $5 \times 5 \times 5$  grid which uniformly spans the entire non-linear RGB cube, then use the white-balancing module for BLRLMO-2 to transform this grid to the white-balanced non-linear RGB set.
2. Use this white-balanced set to drive BLRLMO-2 and measure the corresponding XYZ set. Convert the measured XYZ set to the LAB space with respect to the corrected white point of BLRLMO-2. We call this obtained LAB set “SET 1”.

Table 2.5  
Testing error statistics (in  $\Delta E$  units) of the characterization models  
for BLRLMO-1 and BLRCSO-1 monitors.

	BLRLMO-2	BLRCSO-2
Mean	2.38	1.87
Max	5.72	5.91
Min	0.15	0.27
Median	2.67	1.68
1-stdev	0.92	0.94

3. On the other hand, we use the NLXWP model developed above to transform the  $5 \times 5 \times 5$  grid in the first step to the non-linear RGB space of BLRCSO-2.
4. Use this transformed data to drive BLRCSO-2 and measure the corresponding XYZ values. Then, convert this XYZ set to LAB with respect to the corrected white point of BLRCSO-2. We call this LAB set “SET 2”.
5. Finally, we compute the differences between SET 1 and SET 2 in  $\Delta E$  units.

Table 2.5 summarizes the testing errors of this experiment. Figure 2.14 shows the distribution of error outliers of the NLXWP model in the LAB space.

### Linear transformation for perceptual matching (LXWP)

**Model Development** We would like to see how well we can approximate the NLXWP by a single  $3 \times 3$  matrix which maps directly between the non-linear RGB spaces of the two monitors. We denote this transformation as LXWP. Equation 2.28 shows the formula of the LXWP model:

Table 2.6  
Testing error statistics (in  $\Delta E$  units) of the NLXWP model.

	NLXWP
Mean	1.94
Max	3.88
Min	0.37
Median	1.88
1-stdev	0.80

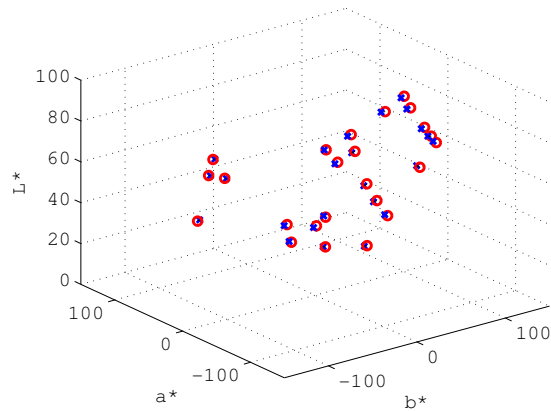


Fig. 2.14. Distribution of the  $1 - \sigma$  error outliers of the NLXWP model in LAB space.

$$\begin{bmatrix} R_2^{nonl} \\ G_2^{nonl} \\ B_2^{nonl} \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \end{bmatrix} \mathbf{M}_{LXWP} \begin{bmatrix} R_1^{nonl} \\ G_1^{nonl} \\ B_1^{nonl} \end{bmatrix} \quad (2.28)$$

Figure 2.15 shows the framework for developing the LXWP. The development for LXWP is formulated as an optimization process in which NLXWP model serves as

the reference and the entries of  $\mathbf{M}_{LXWP}$  serve as the independent variables. The cost function reflects the differences in  $\Delta E$  units between the NLXWP and the LXWP.

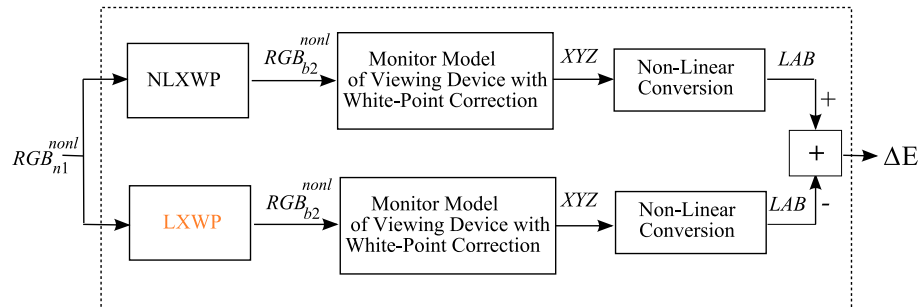


Fig. 2.15. A block diagram of the LXWP model.

### Experimental results

We tested the NXWP in the same way as we did for the NX model in Section 2.1.2 except that we used the monitor models for BLRLMO-2 and BLRCSO-2 rather than for those for CTLLMO-1 and CTLCSO-1.

1. Use the characterization model developed for BLRLMO-2 to predict the XYZ values of the displayed colors when the input is a white-balanced  $5 \times 5 \times 5$  grid spanning the RGB cube of BLRLMO-2. Let us call the predicted XYZ values SET 1'.
2. Use the LXWP model to transform the grid in Step 1 to the non-linear RGB space of BLRCSO-2.
3. Use the monitor model for BLRCSO-2 to predict the XYZ values corresponding to the transformed grid input. Let us call these XYZ values SET 2'.
4. Finally, compute the differences in  $\Delta E$  units between SET 1' and SET 2'.

Figure 2.16 shows the error distribution of the outliers of the LXWP model in the LAB space. Table 2.7 summarizes the error statistics of the LXWP model.



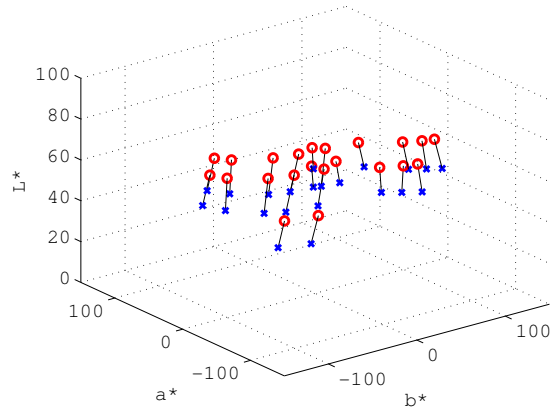


Fig. 2.16. Distribution of the  $1 - \sigma$  error outliers of the LXWP model in LAB space.

Table 2.7  
Testing error statistics (in  $\Delta E$  units) of the LXWP model.

	LXWP
Mean	10.27
Max	24.57
Min	0.79
Median	10.01
1-stdev	8.67

Similar to the case of the LX versus the NLX, the LXWP is still far from being able to approximate the NLXWP model. This is because the NLXWP model is too non-linear to approximate by a linear transformation.

## 2.2 Color Management With 3D Look-Up Tables

The NLXWP function is computationally complex and may not be easy to implement for all displays. Therefore, we use 3D look-up tables (3D LUT) to realize the function in a computationally efficient manner. Look-up tables (LUT) are a convenient tool used in many applications in image processing. Consider a function  $y = f(x)$  which maps an input  $x \in D$  to an output  $y \in R$  where  $D$  and  $R$  are the domain and the range of  $f$  respectively. The concept of a table look-up involves computing discrete values of  $f$  for a set of discrete inputs  $S_D \subset D$  and storing them as a set  $S_R \subset R$ . In order to evaluate the function at a given input  $x \in D$ , we locate points in  $S_D$  which surround  $x$ . In other words, we identify the elements of  $S_D$  which form a box (or a polytope) such that  $x$  is an internal point of the box. These points are known as neighbors of  $x$ . Finally,  $f(x)$  is obtained by interpolating between the known values of the function at its neighbors. The elements of the subset  $S_D$  for which the function is evaluated are known as the *sample* points or *knot* points.

Look-up tables have been used to realize color space transformations between different media [59]. Some of these methods were in use well before the development of standardized profile-based methods. In 1975, Pugsley [60] described a method for color correcting the output of a scanner using pre-computed values stored in the memory. In 1978, Kotera described LUT-based techniques for Bayesian color separation [61]. Researchers at Polaroid Corporation developed a LUT-based method in 1989 for managing color transformations between photographic films and printed media [62]. McCann describes 3D LUT-based methods to perform color space transformations for various applications, including matching of photographic films with printed media, and transforming between the CIE LAB space and a more uniform Munsell color space [63, 64]. These transformations were designed on a  $8 \times 8 \times 8$  hardware LUT, and achieved high levels of accuracy. The LUT-based techniques were also used for color correction in the successful efforts to create life-size replicas of fine art [65].

We previously described the applicability of a 3D LUT for achieving complex transformations [66]. We also described a method for obtaining the optimal parameters for such a system. In the following sections, we provide more detail about the approach and some motivation behind the proposed scheme for the optimal sampling of the RGB space. desired LUT is constructed using the NLXWP function and also cross-validated against it. Figure 2.17 illustrates the method of generating and evaluating the LUT using the NLXWP model. This is referred to as LUTEVAL and is used (as a black box) in our optimization setup. The block arrow between NLXWP and LUT represents the offline process of generating the table by evaluating NLXWP at desired points. The LUT generated by LUTEVAL is stored as a file and, hence, this step does not occur when the LUT is evaluated against the model.

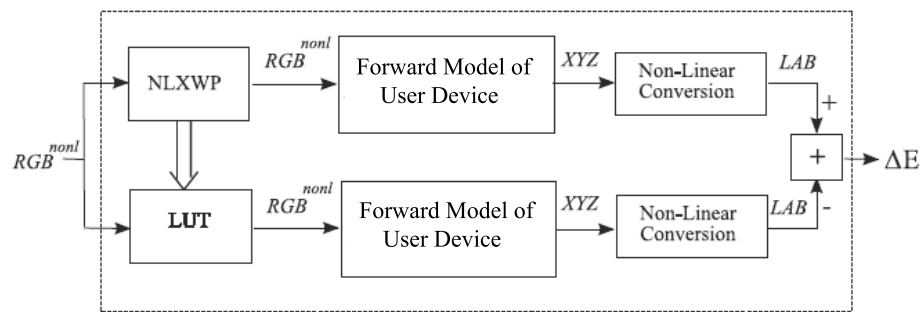


Fig. 2.17. A block diagram of the system for constructing and evaluating a 3D LUT (LUTEVAL).

Let  $f()$  represent the non-linear transformation (NLXWP) using the device models. Since our transformations map non-linear RGB space to non-linear RGB space, let  $[r, g, b]^{in}$  and  $[r, g, b]^{out}$  indicate the input and output vectors respectively. Next, we let the table look-up be described by a function  $\Phi()$ . As in the previous sections, a subscript 1 stands for the first monitor (reference/target device) and subscript 2

stands for the user device or the second monitor. Then, for the same input  $[r, g, b]^{in}$  to the two functions:

$$[r, g, b]_{LUT}^{out} = \Phi([r, g, b]^{in}), \quad (2.29)$$

and

$$[r, g, b]_{NL}^{out} = f([r, g, b]^{in}). \quad (2.30)$$

Using the forward monitor models, we can obtain the corresponding values in the LAB space and compute the difference between the colors:

$$\Delta E = \|[LAB]_{NL}^{out} - [LAB]_{LUT}^{out}\|. \quad (2.31)$$

This is illustrated in Fig. 2.17.

The transformation  $f$  requires a three dimensional (3D) LUT with a vector  $[r, g, b]$  as input. Each entry in the LUT is also a vector  $[r, g, b]$  corresponding to the color-corrected digital input for the user device. For 8-bits per channel, the input space is of size  $256 \times 256 \times 256$  but practical LUTs would be much smaller. Consider a LUT of size  $n_R \times n_G \times n_B$  and sample points selected according to the set

$$\Omega = \{r_0, r_1, \dots, r_{n_R-1}\} \times \{g_0, g_1, \dots, g_{n_G-1}\} \times \{b_0, b_1, \dots, b_{n_B-1}\}, \quad (2.32)$$

where  $\times$  denotes a Cartesian product. Then, the function  $\Phi()$  can be specified as:

$$\Phi([r, g, b]) = \begin{cases} f([r, g, b]), & \forall [r, g, b] \in \Omega, \\ \mathcal{L}([r, g, b]), & \text{otherwise.} \end{cases} \quad (2.33)$$

Here  $\mathcal{L}()$  represents an interpolation operation which is used to transform the input values not aligned with any of the table entries.

The accuracy of a LUT-based transformation is measured in terms of the deviation of the colors in  $\Delta E$  units. Let  $S_T$  be a training set of  $N$  RGB values taken randomly from the RGB space. The difference between the colors produced by the NLXWP model ( $f$ ) and the LUT ( $\Phi$ ) is given by:

$$\Delta E_i = \|[L, A, B]_{NL}^{out,i} - [L, A, B]_{LUT}^{out,i}\|. \quad (2.34)$$

An average of all the differences is referred to as the transformation error:

$$\overline{\Delta E} = \frac{1}{N} \cdot \sum_{i=0}^{N-1} \Delta E_i. \quad (2.35)$$

This average error over the selected training set serves as our yardstick for evaluating the effects of different parameters of a LUT-based system.

Having constructed a method for using a function to generate and evaluate a LUT, we next focus on the various factors which affect the accuracy with which the LUT can approximate the function. Given resource constraints, these can be broadly categorized as follows.

The size of each dimension of a multi-dimensional LUT is related to the relative importance of each dimension. In our problem, this relates to how many sample points are selected from each of R, G and B color channels ( $n_R, n_G, n_B$  respectively) such that the overall LUT complies with the overall size constraint. As an example, one can construct a  $10 \times 10 \times 10$  LUT or a  $10 \times 12 \times 8$  LUT if no more than 1000 table entries are allowed. In our experiments, we assume an equal number of sample points are collected from each color channel. Such LUTs are referred to as *regular*.

A typical (except boundary) point in the 3D space has eight nearest neighbors where the value of the function is known. An interpolation algorithm may use all the eight values or a subset of these to determine the function's value at the unknown point. It may also assign unequal weights to the known values according to the neighbor's "distance" from the point. Thus, the method of interpolation is an important aspect that affects the accuracy of a LUT system.

The sampling set  $\Omega$  contains the elements of the discrete domain of the function. This set is representative of the function and, hence, the choice of its elements plays a key role in the successful interpolation of other values. We also define a special type of sampling for a regular LUT in which  $r_i = g_i = b_i \forall i \in \{0, 1, \dots, n_R - 1\}$ . Such tables will be called *symmetric* and regular tables not meeting this condition will be known as *non-symmetric*.

### 2.2.1 Interpolation

Interpolation has been well explored and several interesting methods have been proposed in one and more dimensional spaces, and on regular and irregular shaped data grids. Bala and Klassen [67] provide a survey of the commonly used interpolation methods, in the context of color transformations. They also describe some acceleration techniques that can result in faster operation under suitable conditions. Kidner *et. al.* [68] review many interpolation and extrapolation methods as applicable to the problem of building sophisticated digital elevation models, used (among others) in terrain analysis, infrastructure design and global positioning. In the following discussion, we describe a simple interpolation scheme in three dimensions. We also present some results demonstrating the relative effectiveness of two derived schemes in our problem of approximating the NLXWP model. Note that this discussion is included primarily for the purpose of completeness, and as an illustration of the effect of choosing an interpolation technique on the performance. The focus of our optimization efforts is on the sampling of the RGB space.

In the most generalized situation, an input point  $[x, y, z]_s$  (whose output needs to be predicted) can have  $k$  neighbors  $[x, y, z]_i$  for  $i = 0, 1, \dots, k - 1$ . Let  $d(i, s)$  be some metric of distance between the points  $i$  and  $s$ . Note that each neighbor is an entry of the table and hence their outputs  $f([x, y, z])_i$  are known. Then using interpolation,

$$f([x, y, z])_s = \psi(f_i, d_i) \text{ for } i = 0, 1, \dots, k - 1, \quad (2.36)$$

where  $\psi$  is determined by the method used.

For example, a simple linear interpolation in 1D space would be

$$f_s = (f_0 \cdot d(s, 1) + f_1 \cdot d(s, 0)) / (d(s, 0) + d(s, 1)), \quad (2.37)$$

where  $d(s, i)$  is the Euclidean distance between  $s$  and its  $i$ th neighbor. A simple extension to 3D would be one where  $f_s$  is evaluated as a weighted sum of  $f_i$  for  $i = 0, 1, \dots, 7$ , with each  $f_i$  weighted by the Euclidean (3D) distance of the other neighbor (and normalized by the sum of these distances) from the required point

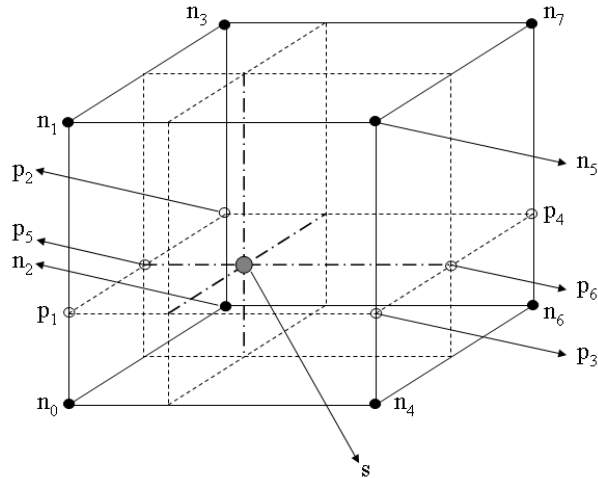


Fig. 2.18. An illustration of the neighbors of a point in 3D space.

$s$  as shown in Fig. 2.18 where the neighbors are denoted by  $n_i$ . Note that in our problem, the 3D box formed by the eight neighbors is not required to be a cube, or even a regular polytope because of non-uniform sampling. A number of schemes exist for interpolation using different neighbor sets and different distance metrics. Shepard's [69] interpolation is a general framework for interpolation on an irregularly spaced data. Another general technique known as Sequential Linear Interpolation optimizes an  $nD$  interpolation problem by sequentially reducing the dimensionality while maintaining optimal performance [70].

As noted earlier, a point in a 3D space has eight nearest neighbors, located at the vertices of a bounding box. It is possible to use a subset of these eight points for interpolation, such as in tetrahedral interpolation [71]. In our experiments for color space transformation, we employ interpolation methods which use all the eight neighbors of the unknown point. Let  $[r, g, b]_s$  be the input value for which an output is needed. We define  $[r, g, b]_i$  as the eight neighbors of the input, where  $i = 0, 1, \dots, 7$ , and the order/position is as indicated in Fig. 2.18. Observe that  $[r, g, b]_0 = [[r], [g], [b]]_s$  where  $[\ ]$  denotes the nearest lower neighbor, that is the nearest neighbor of  $s$  which can be reached by only reducing the value of the color channel. Noted that in this

context  $\lfloor \rfloor$  does not represent the usual integer flooring. Similarly,  $[r, g, b]_7$  is the nearest upper neighbor.

The interpolated value  $r^{out}$  can now be obtained in the following manner. In the following equations, the symbols  $p_i$  represent intermediate points as shown in Fig. 2.18 and  $t_i$  are the corresponding function values. The symbol  $r_i$  represents the first component of  $[r, g, b]_i$  and  $l_i$  is a metric of distance of the  $i$ th neighbor from  $s$ . The values of  $g^{out}$  and  $b^{out}$  can be similarly obtained.

$$t_1 = (r_0 \cdot l_1 + r_1 \cdot l_0)/(l_1 + l_0), \quad (2.38a)$$

$$t_2 = (r_2 \cdot l_3 + r_3 \cdot l_2)/(l_3 + l_2), \quad (2.38b)$$

$$t_3 = (r_4 \cdot l_5 + r_5 \cdot l_4)/(l_5 + l_4), \quad (2.38c)$$

$$t_4 = (r_6 \cdot l_7 + r_7 \cdot l_6)/(l_7 + l_6), \quad (2.38d)$$

$$t_5 = (t_1 \cdot l_{p_2} + t_2 \cdot l_{p_1})/(l_{p_2} + l_{p_1}), \quad (2.38e)$$

$$t_6 = (t_3 \cdot l_{p_4} + t_4 \cdot l_{p_3})/(l_{p_4} + l_{p_3}), \quad (2.38f)$$

and finally,

$$r^{out} = (t_5 \cdot l_{p_6} + t_6 \cdot l_{p_5})/(l_{p_6} + l_{p_5}). \quad (2.39)$$

In the above formulation, we can define  $l_i = d(s, i)$ . Then, the known value at a neighbor is linearly weighted by the Euclidean distance of the opposite neighbor from the unknown point. This can also be viewed as weighting the value at a neighbor with its inverse distance from the unknown point. For example, Eq. (2.38a) can be written as:

$$t_1 = (r_0 \cdot l_0^{-1} + r_1 \cdot l_1^{-1})/(l_0^{-1} + l_1^{-1}). \quad (2.40)$$

Note that this equivalence holds only when interpolating between two points. We refer to this scheme as *separable inverse distance interpolation*. We can also define  $l_i = d^2(s, i)$ . The resultant scheme is referred to as *separable squared inverse distance interpolation* in this thesis.

The choice of non-linear weighting of the function values becomes particularly significant because we are selecting the grid points for the LUT in a non-uniform



fashion, which we describe in the following section. Shepard presented a method for interpolating on an irregular grid in a two dimensional space [69]. The central idea of the method can be described by,

$$f_s = \begin{cases} f_i & \text{if } d(s, i) = 0 \text{ for any } i \in \{0, 1, \dots, P-1\}, \\ \frac{\sum_{i=0}^{P-1} f_i \cdot d^{-u}(s, i)}{\sum_{i=0}^{P-1} d^{-u}(s, i)}, & \text{otherwise,} \end{cases} \quad (2.41)$$

where  $P$  specifies the total number of neighbors used in the interpolation and the parameter  $u$  determines the weighting of grid points. Shepard goes on to describe the criteria for the selection of  $u$ . Based on empirical results, Shepard states that a value of  $u = 2$  would provide satisfactory results for surface mapping and is computationally inexpensive (compared to fractional values).

We note that by defining  $l_i = d^u(s, i)$ , we can approximate Shepard's interpolation by a sequence of pairwise interpolations, represented by Eq. (2.38) and (2.39). It was observed in our experiments that the use of squared inverse distance ( $l_i = d^2(s, i)$ ) results in lower errors when 3D LUTs were used to approximate NLXWP, than when using simple inverse distance ( $l_i = d(s, i)$ ) [66]. This is summarized in Table 2.8 where the average errors of the two schemes are presented with respect to a randomly generated testing set of 1000 RGB values. The errors were obtained using the LUTEVAL block described earlier. Here,  $\overline{\Delta E}_{inv}$  is the error corresponding to the separable inverse distance scheme and  $\overline{\Delta E}_{sq,inv}$  is the error corresponding to the separable squared inverse distance scheme.

All the LUTs were generated by uniformly sampling the RGB space in these experiments. This ensures that the difference in the errors resulted only from the choice of the interpolation techniques. Note that the impact of squared inverse distance interpolation decreases (in both absolute  $\Delta E$  units and in percentage) as the size of the LUT increases. This is expected and emphasizes the role that proper selection of the system parameters can play when the resources are constrained. We also performed this experiment with the non-separable Shepard's interpolation as given in Eq. (2.41), with  $P = 8$ . We observed that the non-separable method resulted in lower average er-

Table 2.8

Average errors ( $\Delta E$  units) obtained when approximating the NLXWP model with uniform LUTs using two interpolation methods.

<b>Size</b>	$\overline{\Delta E}_{inv}$	$\overline{\Delta E}_{sq,inv}$	$\overline{\Delta E}_{sq,inv} - \overline{\Delta E}_{inv}$
$9 \times 9 \times 9$	5.49	3.79	1.70 (30.9%)
$15 \times 15 \times 15$	3.10	2.16	0.94 (30.3%)
$21 \times 21 \times 21$	2.30	1.65	0.65 (28.3%)

rors compared to the two separable methods described above. However, the difference between the average errors becomes smaller when interpolating on lattices, optimized with respect to the corresponding interpolation schemes. It should be emphasized that the regular structure of the separable methods makes them more suitable for hardware applications particularly when interpolating in a 3D space. Therefore, in this thesis, we only provide detailed results based on the separable inverse distance interpolation and the separable squared inverse distance interpolation. In subsequent sections, the squared inverse distance interpolation is used because it produces lower errors.

We would also like to mention that the use of squared inverse distance interpolation has been selected on the basis of accuracy and the simplicity of design. We optimize the sampling of the RGB space (in the next section) with the assumption that this interpolation would be done when using the optimal LUTs. However, other interpolation schemes may be selected based on application-specific requirements such as simplicity, regularity and accuracy. While an interpolation scheme based on only linear interpolations would be simple and involve lower computational costs, an interpolation method based on higher degree polynomials, or on splines, may result in a smoother interpolated surface.

Splines are commonly used in image interpolation [49, 72] because they can produce high levels of accuracy, and can be selected to satisfy regularity requirements. Splines can be represented as piecewise polynomial functions of a specified degree. Thus, unlike higher degree polynomials, splines can be designed to avoid excessive oscillation. The theory of basis splines (B-splines) [73, 74] allows interpolation in such a way that each piecewise function is a linear combination of a set of basis functions. This results in a reduction in the complexity of spline interpolation. Two common methods for extending B-splines to a multi-dimensional space are the use of tensor product splines [75] and the use of thin plate splines [76].

We do not use spline interpolation for two reasons. The spline basis functions must be computed directly using the positions of the knot points. It is not practical

for hardware implementations to precompute these basis functions if the knot points are irregularly spaced. Secondly, the actual interpolation uses a linear combination of the basis functions and the coefficients used as weights must also be computed before interpolation. Therefore, an implementation of splines in a high dimensional irregular space would be more complex.

We summarize this discussion by stating that our optimization framework is agnostic to interpolation and a simple modification of the cost function is needed to reflect the change.

### 2.2.2 Sampling

Sub-sampling of the RGB space is a complex problem that has been discussed in the literature. Monga and Bala [77] propose a strategy for the optimization of the knot points (or nodes) based on a 3D importance function. They construct a significance function using the input distribution and the curvature of the analytical function to be approximated. Instead of trying to determine an exact solution to the problem of optimal node selection, they devise an efficient algorithm to obtain an approximate optimal lattice, in an iterative manner. They later proposed a method to jointly optimize node selection and the output values stored at the nodes [78] in order to minimize the expected interpolation error. However, a general solution for approximating 3D functions is not available, particularly when the function cannot be rigorously defined (in order, for example, to compute the curvature). Intuition suggests that a discrete domain  $\Omega$  for a LUT constructed by uniformly sampling the input space may not deliver the optimal performance. It is possible that using more samples from the dark regions of the color space for  $\Omega$  may lower the average error. This is mainly because a small change in the RGB values in the low lightness regions causes a large perceptual change in the response of the human visual system.

We experimented with different approaches to arrive at a sampling strategy that would provide an improved performance relative to uniform sampling. These are described below:

**Sampling in LAB space.** A straightforward way to achieve perceptual uniformity would be to use the color space that best models the human visual perception. The goal would be to divide the CIE LAB space into  $n$  equal sub-spaces. The boundaries of these sub-spaces could then be translated back to the RGB space using the available non-linear monitor models and a LUT could be constructed from these values. While this approach is attractive, it is intractable because the LAB space is not a regular polyhedron like the RGB space. It is quite possible that the space for a given display device is shaped like a deformed sphere. Locating the exact bounding surfaces of the LAB space might require enormous computation and it is unlikely that such a space can be divided into regularly shaped sub-spaces.

**Uniform lightness sampling.** It is tempting to try achieving at least uniform sampling in the CIE  $L^*$  dimension. One would expect that the function of lightness versus R (or G or B) would be concave with higher gradient near the dark region. Therefore, if the color channels are sampled for uniform  $L^*$ , it should result in more samples collected from the dark areas. A plot of non-linear R versus  $L^*$  is shown in Fig. 2.19. Although there is significant non-linearity near the dark end, most of the function is nearly linear. However, this is not so unexpected if we understand the relationship between the native color space and the  $L^*$  axis. There are two basic steps in going from R to  $L^*$  – first R (along with G and B values) is converted to linear RGB by a power law correction (gray balancing), and then it is transformed to XYZ by a linear transformation. Finally, lightness is obtained using another power law. The two power law corrections reduce the effect of each other to a large extent and hence a near-linear behavior is observed for most inputs.

**Optimal RGB sampling.** It is evident that no clear methodology seems to exist for achieving the best sampling map or the set  $\Omega$ . Therefore, we try to obtain the optimal value of  $\Omega$  using constrained optimization. Given a specification of the table

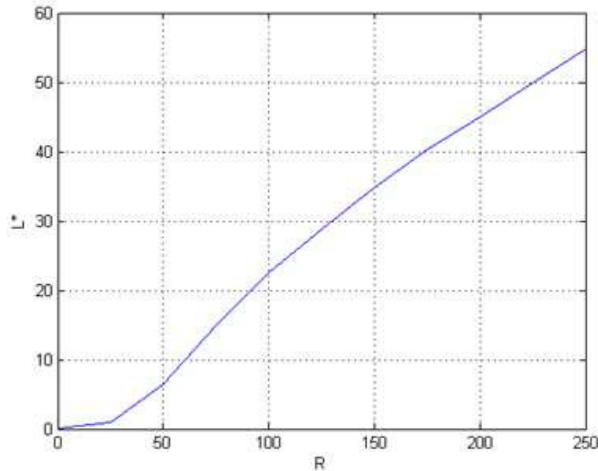


Fig. 2.19. Variation of lightness ( $L^*$ ) with digital red input  $R$ .

dimensions  $n_R \times n_G \times n_B$ , the universe of all possible sampling maps (without any constraints) would be roughly  $256^{n_R+n_G+n_B}$ . Once we apply some obvious constraints (described below), this number reduces slightly but it is still  $O(256^{n_R+n_G+n_B})$ . Considering the fact that it is unlikely that a particular knot point is selected twice, a more accurate estimate of the size would be  $[256 \times (256 - 1) \times \dots \times (256 - (n_R - 1))]^3$ . In simple terms, this is an optimization problem in a  $n_R + n_G + n_B$  dimensional space.

An optimization problem is defined in terms of a cost function. In our case, we use the average prediction error  $\overline{\Delta E}$  as the cost function given the training data set. We obtain a training data set  $S_T$  consisting of 1000 points (RGB triples) by constructing a  $(10 \times 10 \times 10)$  grid in the RGB space:

$$S_T = \{r_0, \dots, r_9\} \times \{g_0, \dots, g_9\} \times \{b_0, \dots, b_9\}, \quad (2.42)$$

where  $\times$  denotes a Cartesian product. Initially all elements of this set were chosen randomly from the RGB space, but this approach often returns a skewed data set. Therefore, we enforce order by adding the eight corners of the RGB space to the set. The remaining 992 elements were randomly selected. The size of the training set is

chosen to be significantly larger than the number of unknowns, thereby avoiding the possibilities of overfitting.

We proceed by inserting constraints which make the problem more tractable. First, we assume  $n_R = n_G = n_B$  which means that we optimize only regular LUTs. Let

$$M_R = \{r_0, r_1, \dots, r_{n_R-1}\}, \quad (2.43)$$

$$M_G = \{g_0, g_1, \dots, g_{n_G-1}\}, \quad (2.44)$$

$$M_B = \{b_0, b_1, \dots, b_{n_B-1}\}, \quad (2.45)$$

$$\text{and } \text{map} = M_R \times M_G \times M_B. \quad (2.46)$$

$$\text{Then } \Omega_{opt} = \arg \min_{\text{map}} \overline{\Delta E}. \quad (2.47)$$

where the vectors  $M_R, M_G, M_B$  are known as channel maps while  $\text{map}$  is the Cartesian product of the channel maps and contains all the knot points at which  $\Phi()$  is evaluated to construct the table.  $\Omega_{opt}$  is the value of  $\text{map}$  such that the LUT based prediction model has the smallest  $\overline{\Delta E}$  for a given training data set. The minimization is subject to the constraints that each element of  $\text{map}$  belongs to  $\{0, 1, \dots, 255\}$  and the elements of each channel map  $M_R, M_G, M_B$  are strictly increasing. For example, in the case of the red channel:  $r_0 < r_1 < \dots < r_{n_R-1}$ .

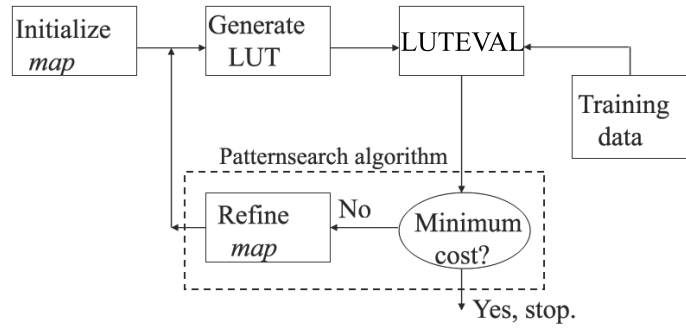


Fig. 2.20. A block diagram of the optimization problem of sub-sampling the RGB space.

The computation of  $\overline{\Delta E}$  for a given *map* is a two step process consisting of generating the LUT by evaluating the function according to *map* and then using this LUT to transform elements of the training set  $S_T$  to obtain the average prediction error. Since this process cannot be rigorously defined, many optimization techniques (such as gradient-based schemes) cannot be used. Figure 2.20 illustrates the optimization problem graphically, and the problem is solved using the pattern search algorithm.

Pattern search [52, 79] is a non-gradient based optimization technique which can locate the global minimum of a given objective function with linear constraints. It has been used in a wide variety of optimization problems. At any iteration, the algorithm evaluates the objective function at every point in a set, known as a *pattern*. The pattern is expanded or shrunk depending on these values. More specifically, the pattern is expanded if any point in the set results in a lower cost than the current minimum; otherwise the set is shrunk (usually by a factor of two). Since the method does not require computation of derivatives to update the pattern, it can be used when the objective function is such that this information is not available, or difficult to obtain. Hence, it is suitable for our optimization problem. The search for the optimum terminates when the pattern shrinks below a specified maximum size. The method has been shown to possess robust convergence properties [80]. Software implementations of the method can be obtained in the MATLAB Genetic Algorithms and Direct Search toolbox [81], in the open source OPT++ library from Sandia Corporation [82], and from the authors of the algorithm [83].

It should be noted that a symmetric sampling might allow greater control over the transformation of the neutral axis. This may be an important decision for certain applications that require pure neutral colors to be faithfully reproduced. Similarly, some applications (such as motion pictures) may place more importance on the skin tones. Our optimization setup provides an interesting method of specifying such preferences, which is by allowing the user to carefully design the training set  $S_T$ . By an intuitive crafting of  $S_T$ , one can instruct the system to acquire greater accuracy in the desired regions of the color space.



Let the optimal LUT be represented by  $\Phi : \mathfrak{S} \rightarrow \mathfrak{S}$  such that  $\Psi_{reference}(i) \approx \Psi_{user}(\Phi(i))$  for any  $i \in \mathfrak{S}$ . We would like to compare the visual similarity achieved by a LUT-based approach with that achieved by a profile-based approach. Typical ICC profiles can range from 500 bytes to over 500 kilobytes [84] and a profile-based color management system would use two such profiles as shown in Fig. 1.7. Therefore, for a fair comparison we assume that our LUT can be as large as two ICC profiles. We perform our comparative experiments using profiles of two sizes – 7 kilobytes and 66 kilobytes. The process of generating these profiles and obtaining a color transformation using them is described in the next section.

### 3. VISUAL SURVEILLANCE OF VEHICLES

The large volume of vehicles on the road has created new challenges for agencies responsible for traffic management, law enforcement, and public safety. Such agencies frequently utilize visual surveillance technology to assist monitoring of vehicles from a remote location. Such surveillance systems typically require trained human operators. Consequently, they are prone to human errors due to fatigue or diverted attention caused by excess information. Thus, a need exists for an automated system that can analyze the surveillance videos and extract important information. This information would be used to detect occurrence of “anomalous” events, in which case the human operator would be alerted.

In this chapter, we propose a visual surveillance system designed to function in the above-mentioned manner. More precisely, the system observes vehicular traffic from a standoff range and extracts information for each vehicle. This information includes vehicle type, make, tire size, and velocity. Based on the information, the system checks for anomalies in the appearance and/or motion of the vehicle. We describe methods for obtaining the vehicle information from two cameras placed in an orthogonal configuration, and for classifying the vehicles using these observations. We present the results of applying these methods on traffic videos. Our proposed system can be deployed for traffic monitoring (at intersections), or infrastructure protection (at check points). It can also be extended to incorporate other non-video sensors in a complementary fashion.

#### 3.1 Overview of the Proposed Surveillance System

The objective of this project was to determine the feasibility of a multi-sensor surveillance system to observe vehicles and detect any anomalies, particularly anoma-

lous vehicle payloads. It was required that the system should operate from a standoff range and not obstruct the traffic flow. The different sensing techniques investigated for the task are as follows:

- **Instrumented cleats** would be placed on the road with accelerometers to estimate the weight distribution along a vehicle's length.
- **Laser vibrometer** would be used to capture the acoustic signature of different parts of the vehicle (including the trunk and the door panels).
- **Acoustic arrays** would estimate the acoustic modes in the tires to determine their inflation material and/or pressure remotely.
- **Inverse synthetic aperture radar** would monitor the vertical oscillations of a vehicle and image its interior.
- **Video analysis** would be responsible for – (i) detecting vehicles and triggering other sensors, (ii) extracting miscellaneous information needed to characterize a vehicle, and (iii) performing dynamic analysis of vehicle's motion to detect anomalous maneuvers.

A schematic diagram of the surveillance system deployment is shown in Figure 3.1. Note that the position of different sensors is approximate. In this thesis we describe the video analysis system in detail. The tasks associated with the non-video sensors were studied by our project collaborators and will not be described here.

In this section, we describe our video surveillance system that can autonomously observe vehicular traffic from a standoff range of approximately 50 feet (15m) and classify the vehicles in terms of “normal” behavior. The system is primarily based on video analysis but also provides interfaces for synergic operation with other types of sensors (including rumble strips for sensing mechanical abnormalities and synthetic aperture radar). There are two premises on which this system is based. First, in most situations there is a logically acceptable range of behaviors that would be considered normal. If the response of an individual (vehicle) deviates from this range the behavior

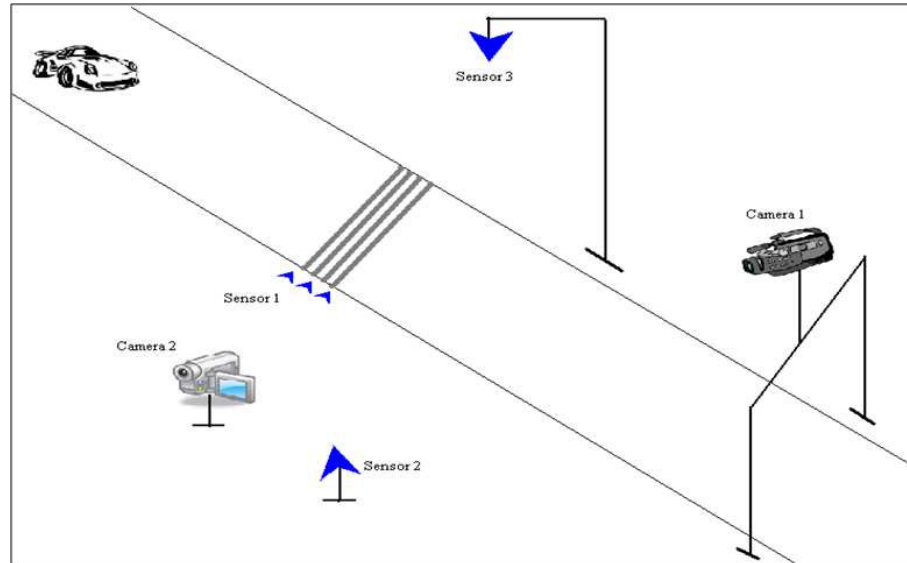


Fig. 3.1. An illustration of the multi-sensor surveillance system. The sensors 1, 2, and 3 represent cleats, acoustic sensors, and radar, respectively.

is considered anomalous. Secondly, there are many traits of anomalous behavior which can be ignored if they occur in isolation. However, the occurrence of multiple such behavior patterns is more likely to signal a potentially anomalous situation. This is described in more detail below.

We begin with examples illustrating the two hypotheses of our approach. In normal circumstances vehicles are driven within the lane boundaries and lane changes are only occasional. However, a vehicle repeatedly ignoring the lane markings may be indicative of drunk/impaired driving or of road rage. Therefore, it is possible to identify anomalous driving behavior, in this case by observing the frequency of lane departures. Similarly, presence of relatively flat tires may not convey much information alone. But, a vehicle with visibly flat tires and moving below the speed limit could be overloaded and pose a threat. We select certain traits associated with a vehicle and define normal and abnormal “observations.” These can be later used to identify anomalies.

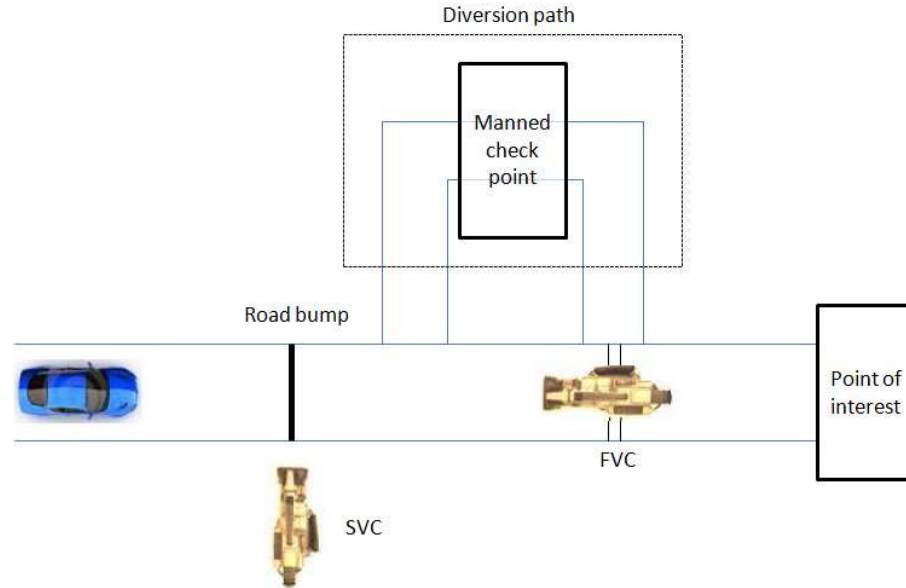


Fig. 3.2. An overview of the standoff video surveillance system deployment. The distances are not drawn to scale.

An overview of our proposed system is illustrated in Figure 3.2. The screening system can be deployed near the entrance to a point of interest (such as airport, sports venue, or a public building). The goal is to observe the vehicles without affecting the traffic flow. Our system uses two video cameras – the front view camera (FVC) and the side view camera (SVC). The actual position of these sensors is determined based on the desired field of view and camera resolution. It should be noted that the FVC is located at a fixed height above the road surface (although this is not evident in the top-view schematic) to allow sufficient clearance for vehicles.

The selection of two cameras and the particular configuration is described as follows. In our system, the purpose of multiple cameras is to generate complementary information (rather than redundant information). Thus, for a cost-effective design, the number of cameras should be small. It can be argued that a two-camera system provides considerably more information than a one camera system. However, the incremental gain with a three camera system is small. Furthermore, the front and

side views provide more important information than alternative configurations of a two camera system. For example, a rear-view (instead of front-view) allows license plate detection but eliminates the scope for driver behavior monitoring.

For any vehicle, the system would produce one of the three “actions” – **Pass**, **Monitor**, and **Stop**. The vehicle designated for ‘Stop’ would be diverted for secondary screening while the ‘Pass’ vehicles will be allowed passage unobstructed. While an ideal system should produce minimum ‘Monitor’ labels, in practice, the passage of certain vehicles may need to be determined based on the application. In certain areas, all ‘Monitor’ vehicles may be screened; in others, they may all be allowed passage. The system would also enable human operator in making such a decision by providing them with the information associated with the vehicle. The exact mechanism for vehicle diversion and secondary screening is not discussed in this thesis.

Using the two cameras we extract, using image analysis methods described in the next section, various information about the oncoming vehicles that can be placed into two categories.

**Physical Information:** The data obtained here is primarily used to characterize and/or identify a vehicle. This information includes vehicle body type, tire size, and make.

**Behavioral Information:** This set of information includes high-level analysis of the vehicle’s appearance and motion. It includes analysis of the velocity and trajectory of the vehicle. This information is directly used for assigning an action label for the vehicle.

We can organize the various types of information extracted and/or inferred from our image analysis described below into an approximate hierarchy. This is shown in Figure 3.3. Note that certain information extracted from one camera is used to analyze the images from the other camera, thus highlighting the usefulness of the orthogonal camera configuration. A typical sequence of operations when a vehicle approaches would be as follows. The name of the camera which is primarily used

for making a particular detection is provided inside parentheses. The image analysis methods used to perform these tasks are described in more detail below.

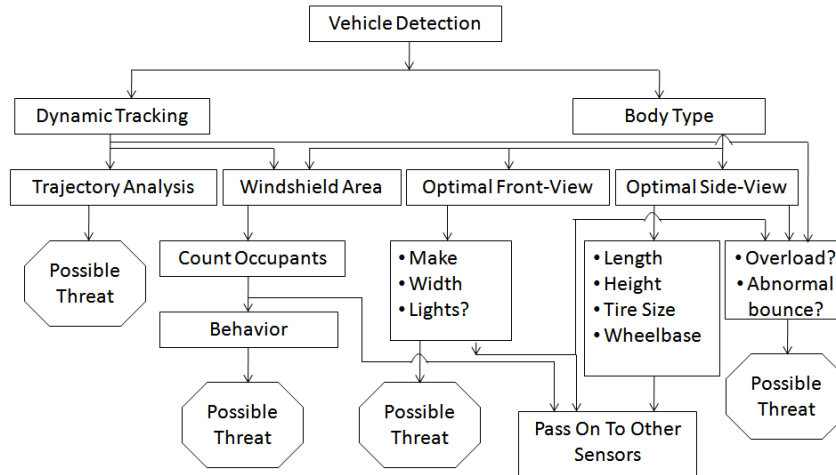


Fig. 3.3. An illustration of inter-dependencies in the information extracted from the cameras.

1. An oncoming vehicle is detected. (FVC)
2. The vehicle tracking and trajectory analysis systems are activated. An unexpected slowing/stopping may lead to the vehicle being marked for ‘Monitor’ while a sudden acceleration may result in a ‘Stop’ label. These decisions are based on the assumption that the driver may be searching for ways to evade the security systems or may be intending to crash through them resulting in these, respective, patterns in the vehicle’s approach. (FVC)
3. When the vehicle enters the SVC’s field of view, its body type is determined. (SVC)

4. The vehicle's tires are segmented and the tire size is estimated. (SVC)
5. The make of the vehicle is determined. (FVC)
6. The gap between the tires and the vehicle body (wheel well) is estimated and the front and rear values are compared. If the gap above the rear tire is significantly smaller, it can indicate a heavy load in the trunk which may lead to a 'Monitor' label. (SVC)

## 3.2 Image and Video Analyses

We now describe the different image and video analysis techniques used to extract information from the camera feeds. Our focus is on developing low complexity methods which would be more suitable for real-time operation.

### 3.2.1 Vehicle Detection

The initial steps in all intelligent surveillance systems involve detection of a subject of interest. Background subtraction is the most common approach to object detection, and several elegant algorithms have been proposed [85]. A background subtraction operation consists of comparing the current video frame with a background model with respect to some properties like pixel intensity and color. Since object detection is so important to our system, we describe it in some detail.

Let the current frame be represented (as a digital image) by

$$C = \{c(i, j) : i = 0, 1, \dots, W - 1; j = 0, 1, \dots, H - 1\}. \quad (3.1)$$

Here,  $c(i, j)$  represents the value of the chosen image property at the pixel located at  $(i, j)$ . The frame is considered to be of width  $W$  and height  $H$ . Similarly, we



can represent the background model as an image  $B = \{b(i, j)\}$ . Then, the output of background subtraction is a spatial mask  $F = \{f(i, j)\}$  defined as follows:

$$f(i, j) = \begin{cases} 1.0, & \text{if } c(i, j) \text{ is in the foreground,} \\ 0.0, & \text{if } c(i, j) \text{ is in the background.} \end{cases} \quad (3.2)$$

The decision of classifying a pixel as foreground is based on a metric to measure the difference and a decision threshold corresponding to the metric. Let the features used to specify each pixel belong to a space  $\Psi$ , and consider a function  $\delta : \Psi^2 \rightarrow \Re$  where  $\Re$  is the real number line. The value of the function  $\delta(c(i, j), b(i, j))$  quantifies the difference between the current frame pixel and the background model pixel at the location  $(i, j)$ . While the space  $\Psi$  can contain one or more of velocity vectors, acceleration vectors, intensity levels, edge orientation and trichromatic digital values (like R, G, B), we consider a simple case of grayscale images with the pixel intensity specified by a real number between 0.0 and 1.0 where 1.0 represents the maximum gray level. The corresponding difference function would be  $\delta(x, y) = |x - y|$ . We can redefine the mask  $F$  as:

$$f(i, j) = \begin{cases} 0.0, & \text{if } |c(i, j) - b(i, j)| \leq \tau, \\ 1.0, & \text{otherwise.} \end{cases} \quad (3.3)$$

Note that  $0.0 \leq \delta(x, y) \leq 1.0$ . Therefore, the decision threshold  $\tau$  would be a real number in  $[0.0, 1.0]$ .

In the basic background subtraction algorithm, the threshold  $\tau$  and the model  $B$  may be empirically chosen and remain fixed throughout the operation. Most of the improvements in the basic algorithm affect the selection and updating of these two parameters. Some algorithms allow gradual modification of the background model after each frame based on the latest classification. This gives the algorithm some flexibility to perform correct classification even when the background is not truly static. These tasks may be performed statistically [86–88] or adaptively [89, 90]. One of the key considerations for a real-time surveillance system is the complexity of the algorithm. We next describe our method for background subtraction which does not require a model update every frame and still accounts for background perturbations.

Most commonly used background subtraction methods update the background model using the results from the current frame classification. Let  $C_n$  and  $B_n$  represent the current frame and the background model at time  $n$ . Similarly, let  $F_n$  be the foreground mask corresponding to  $C_n$ . Then, the new background model can be constructed as:

$$B_{n+1} = \alpha \times C_n + (1 - \alpha) \times B_n, \quad (3.4)$$

where  $\alpha$  represents the *learning rate* of the algorithm and  $0.0 \leq \alpha \leq 1.0$ . This approach has the limitation that if a foreground object stays in the scene for an extended amount of time, it may alter the background model unnecessarily. Another issue is that for certain types of repetitive motion (tree leaves or shaky camera) the model might never be able to “catch up” and always result in incorrect classification. Moreover, updating the model for every frame is computationally expensive.

We observe that whenever the background model is changed due to small (and temporary) displacement of a background object, we can account for the change simply by comparing the current pixel with the correct background pixel. However, since the correct position of such a pixel is not known, we compare the current pixel with a neighborhood of pixels in the background model. Let  $(i, j)$  be the pixel location being investigated. Then, we construct a neighborhood of  $c(i, j)$  as:

$$N(i, j) = \{b(p, q) \in B : i - \epsilon \leq p \leq i + \epsilon, j - \epsilon \leq q \leq j + \epsilon\}, \quad (3.5)$$

where  $\epsilon$  is the search window size. Next, the foreground mask as generated by our method is given by:

$$f(i, j) = \begin{cases} 0.0, & \text{if } \exists b \in N(i, j) \text{ such that } |c(i, j) - b| \leq \tau_1 \\ 0.5, & \text{if } \tau_1 < |c(i, j) - b| \forall b \in N(i, j) \text{ and } \exists b \in N(i, j) \\ & \text{such that } |c(i, j) - b| \leq \tau_2 \\ 1.0, & \text{otherwise.} \end{cases} \quad (3.6)$$

In this formulation, an extra state of low confidence is added, represented by  $f(i, j) = 0.5$  and computed using a second threshold  $\tau_2 > \tau_1$ . For the purpose of classification, every pixel  $c(i, j)$  is considered a background pixel if  $f(i, j) \in \{0.0, 0.5\}$ . This extra

state of low confidence classification is used in background model update which we use only when a global illumination change is detected. The new threshold  $\tau_2$  is chosen as a fixed fraction of the dynamic range of the frame. We refer to our method as motion-assisted background subtraction (MABS).

It is natural to expect that in most situations, a given pixel will either be at its original position (in  $B$ ) or be displaced by a small amount compared to the size of the neighborhood characterized by  $\epsilon$ . Therefore, we examine the elements of  $N(i, j)$  in an outward spiral fashion. This is illustrated in Figure 3.4. The approach boosted the speed of MABS, particularly in sequences with small motion.

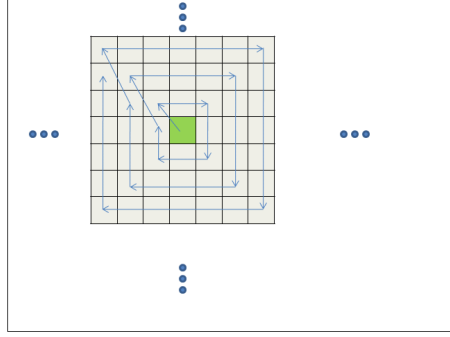


Fig. 3.4. Illustration of outward spiral search in  $B$  with  $\epsilon = 3$ .

A global illumination change is inferred when the average gray level of the current frame differs from the background model by a pre-defined measure  $\tau_i$ . This constant is empirically chosen by considering the effects of multiple foreground objects, the global gray levels, and the tolerance provided by the selection of the threshold used in Equation 3.6. In the event of a global illumination change, the background model is updated as follows:

$$b_{n+1}(i, j) = \begin{cases} c_{n+1}(i, j), & \text{if } f_n(i, j) = 0.0 \\ \alpha \times c_{n+1}(i, j) + (1 - \alpha) \times b_n(i, j), & \text{if } f_n(i, j) = 0.5 \\ b_n(i, j), & \text{if } f_n(i, j) = 1.0. \end{cases}$$

Thus, we select the learning rates for model update on the basis of confidence in the current classification.

### 3.2.2 Vehicle Body-Type Determination

The knowledge of an oncoming vehicle's body type is useful in several ways. It narrows the expected range of values corresponding to different measurements being taken. For example, we describe the tire size estimation in the next topic. This task would be much harder if the estimation algorithm is ignorant of the body type. This information also helps in locating the windshield area in the front view camera (FVC) video which can be used to count and observe the occupants. Therefore, the primary role of this information is to aid the functioning of other modules. We consider vehicles belonging to four classes – sedan, light truck, sport/utility vehicle, and hatchback.

Determination of a vehicle's type is a part of many traffic monitoring systems where it may be used to generate the temporal composition of traffic on a busy route. Some traffic monitoring systems are based on inductive loops and other non-video methods [91]. Several elegant video-based methods have been proposed for the problem. In [92], a statistical model-based approach is used to detect and classify vehicles using a two stage process. Avery [93] presents a simple method to distinguish cars from trucks based solely on the vehicle length. Other techniques extract shape and motion information to identify the type [94,95]. Lai [96] describes a traffic analysis system using a video-based virtual loop technique which also classifies vehicles.

We now describe a simple shape matching technique to determine the most likely class of a vehicle using an image taken by the side view camera (SVC). This approach is suitable because it can use the output of the background subtraction algorithm with only a small amount of processing. Recall that the output of the vehicle detection routines is a binary image or video frame where the pixels are marked as a part of the vehicle or a part of the background. Note that the third state of low confidence

described in Section 3.2.1 is treated the same as the background for classification. Thus, an intensity inversion on the foreground mask  $F$  produces a silhouette of the vehicle, on a white background.

Unlike [94], we do not extract features from the silhouette. Instead, we directly perform a silhouette-matching against the templates corresponding to each vehicle class. Let  $S = \{s(i, j)\}$  be the silhouette of an oncoming vehicle as obtained by inverting the foreground mask  $F$ . Also, let us represent the silhouettes of the template vehicles as  $T_1, T_2, T_3, T_4$ , where the class labels correspond to sedan, truck, SUV, and hatchback, respectively. These templates are obtained by thresholding carefully selected images in which the vehicles differ considerably from a neutral background. An example of such a vehicle and the resulting template is shown in Figure 3.5. After thresholding, the binary image is cropped so that the vehicle silhouette is immediately surrounded by the image boundaries. At this instant, the aspect ratio of the vehicle and the size of the cropped image are not altered in any way. Thus, the templates may mutually differ in the image dimensions.



Fig. 3.5. An example of the template extracted from a vehicle belonging to the “sedan” class.

While the templates are obtained from images with highly favorable backgrounds, the test vehicle may not be viewed against such conditions. Therefore, the silhouette  $S$  may contain white patches at places where the background subtraction algorithm made erroneous classification. This would be a problem if the image was used to extract, for instance, edge features as the patches would give rise to false edges. However, we use a direct difference-based approach and the patches only appear as

an additive constant in each vehicle class. The differencing operation is described next:

1. As in the case of the templates, the test vehicle silhouette is obtained by cropping the inverted foreground mask to the smallest size that still contains the whole vehicle. Let the silhouette  $S$  have dimensions  $w \times h$ . It should be noted that cropping eliminates the issues related to registration.
2. Each class template is scaled to a size of  $w \times h$ . These scaled templates  $\tilde{T}_k$  are used to compute the sum of absolute difference (SAD)  $\Delta_k = \sum |s(i, j) - \tilde{t}_k(i, j)|$  where the summation is taken over all the pixels.
3. The class corresponding to the smallest SAD value would be the most likely type of the test vehicle.

The primary purpose of obtaining the vehicle’s silhouette was to compute the SAD against the templates but in the process, we have also determined the vehicle’s height and length in image co-ordinates. More specifically, we note that the values  $w$  and  $h$  in the cropped templates are exactly the length and height of the vehicle in pixels. By mapping these to “world co-ordinates,” the true dimensions of the vehicle can be estimated. There are two popular methods to perform this mapping – camera calibration and use of fiducial marks.

Camera calibration techniques use spatial transformations along with some analytical models to develop the pixel-inches mapping. These can also be realized using simpler approximate functions or even look-up tables. However, in the circumstances when such calibration is not possible, the algorithms utilize the presence of objects of known dimensions in the scene for the purpose. These objects are known as fiducial marks and are, typically, simple geometric shapes like lines and points. In cases when such marks are not intrinsically present in the scene, they may be explicitly added.

### 3.2.3 Tire Size Estimation

Visual inspection of a vehicle's tires sometimes provides useful information like over/under inflation or the presence of oversized tires for better towing and off-road capability. However, much of this information is not directly of value to traffic monitoring and analysis systems. Thus, most traffic surveillance systems lack the capability to accurately segment the tires and estimate their size. Although many of the advanced object recognition systems may be trained to detect tires, they may not be suitable for a real-time and real life application because of the high complexity, and because of the variability in the tire profiles. Some examples of tires with differing hubs are shown in Figure 3.6.

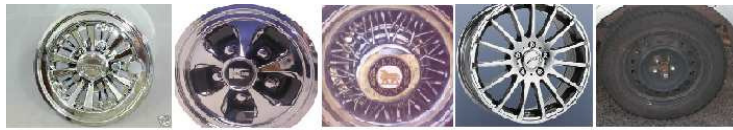


Fig. 3.6. Some examples of tire hubs including bare wheels, plastic hubcaps and alloy wheels.

Accurate location of an oncoming vehicle's tires is important to us for two main reasons. This information, along with the make (described in Section 3.2.4) and the exterior dimensions, helps in identifying the vehicle's model. Secondly, a direct analysis of the tire flatness and bounce (described in Section 3.2.5) may be used to detect overloading. We now describe a method to correctly locate the tires and determine their size. This analysis is performed using frames from the SVC's output.

There are two major challenges in accurate segmentation of a moving vehicle's tires – the lack of information about both the position and size of the tire, and the different shapes of tire hubs. We note that despite the variation in the shapes of the hub, most tires have one common feature, which is the prominent circular edge between the

wheel and the rubber. Since most vehicles on the road today have either hub caps or alloy wheels, this observation holds true for them. The notable exceptions include vehicles with older wheels and missing hubcaps and the newer black alloy wheels. Since a single approach is not universally applicable in all tires, we describe the two cases separately.

**Tires with shiny hub:** In the cases when a significant level of contrast exists between the tire's wheel and rubber, it is possible to detect the wheel-rubber edge. This situation is illustrated in Figure 3.7. Our goal is to accurately locate the circular edge because it provides us the knowledge of the tire's center and the radius of the (visible part of the) wheel. For the present discussion, consider only the front tire of the vehicle as seen during a particular video frame  $V_{SVC}$ . Let  $(x_t, y_t)$  be the true location of the front tire center in the frame and the true tire radius be  $r_t$  pixels. Further, let  $r_w$  be the true radius of the wheel as visible from a side view.

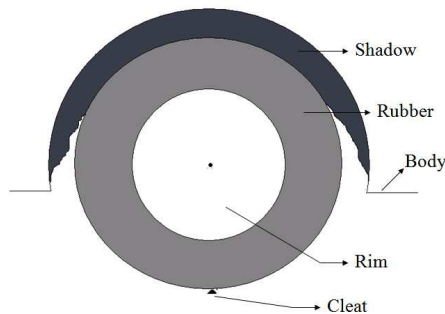


Fig. 3.7. A graphical illustration of the system's view of a tire with shiny hub.

We model the wheel-rubber edge as concentric white and black circles on a neutral background as illustrated in Figure 3.8. The thickness of the circles  $t$  is kept constant (usually 2 pixels) whereas the radius can be changed during the estimation process. Let  $r$  be the (outer) radius of the white circle. Then, at the end of the processing,  $r$  would be the algorithm's estimate of  $r_w$ . The estimation is performed by placing the edge model of a chosen radius at different positions in the frame.





Fig. 3.8. In the case of a tire with shiny hub, the wheel-rubber edge is modeled as concentric circles of chosen radii.

Let the tire model be represented by  $T_{r,p,q}$  where  $r$  is the radius of the wheel and  $(p, q)$  is the hypothesized position of the wheel center. In order to superpose the edge model on the frame, we can define the model for the same width and height as the frame:

$$T_{r,p,q}(i, j) = \begin{cases} 0.0, & \text{if } r - t < \|(p, q) - (i, j)\| \leq r \\ 1.0, & \text{if } r < \|(p, q) - (i, j)\| \leq r + t \\ 0.5, & \text{otherwise.} \end{cases} \quad (3.7)$$

Then, the error associated with this set of values for  $(r, p, q)$  is computed by

$$\tilde{\Delta}_{r,p,q} = \Sigma |V_{SVC} - T_{r,p,q}|, \quad (3.8)$$

where the summation is carried out over all pixels in the frame for which the edge model is not neutral. That is, the difference is computed only if  $t_{r,p,q}(i, j) \neq 0.5$ . Finally, the best estimate of wheel parameters is given by,

$$(r, p, q)_{optimal} = \arg \min \{ \tilde{\delta}_{r,p,q} \}. \quad (3.9)$$

It should be noted that the ranges of  $(r, p, q)$  to be searched over can be intelligently limited by using information about the vehicle's body type and the vehicle's position with respect to the speed bump shown in Figure 3.2 (and labeled as a cleat in Figure 3.7).

Once the position of the tire center is determined, the tire radius can be determined by dropping straight lines away from the center as illustrated in Figure 3.9. Multiple estimates of the radius are made by measuring the length of each such line between the center and the rubber-background edge. We drop lines only in the lower half of the tire and consider only the lines which are longer than  $r_w$ . Then, the best estimate of the tire radius is the shortest such length.

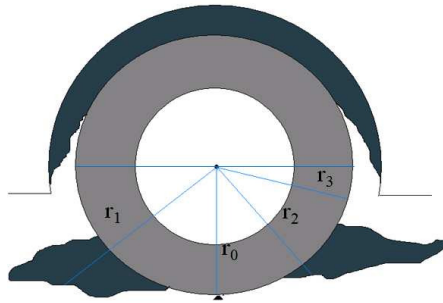


Fig. 3.9. Multiple lines are dropped from the known tire center towards the rubber-background edge to estimate the tire radius.

**Tires with dark hub:** If there is insufficient contrast between the tire rubber and the wheel (as in Figure 3.10), the strategy to look for a distinct circular edge fails. In such cases, additional information is needed by the algorithm to estimate the size of the tire. Therefore, we assume that the horizontal position of the tire's center can be determined by non-video methods. One way to accomplish this is to place pressure or switching sensors near (or inside) the speed bump to determine the exact instant when the tire crosses the bump (marked as a cleat in the figure). Since the position of the speed bump in the frame is known, the position of the lower-most point of the tire is determined.

Assuming that the tires are roughly circular in shape, we model the tire  $T_r$  as a black disk on a neutral background such that the disk has a radius  $r$  and its lower-

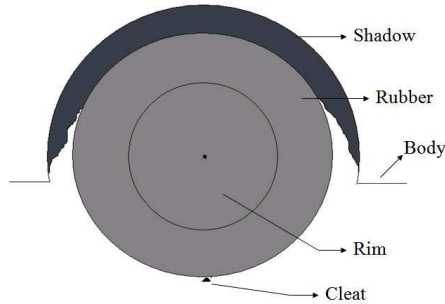


Fig. 3.10. A graphical illustration of the system's view of a tire with dark hub.

most point coincides with the speed bump. As in the case of tires with shiny hubs, we compute the error associated with a particular choice of  $r$  using,

$$\tilde{\Delta}_r = \Sigma |V_{SVC} - T_r| \quad (3.10)$$

where the summation is performed over all pixels for which the tire model is not neutral. It can be argued that if the radius of the disk is taken small enough, the error will vanish but this value could be much smaller than the actual tire radius. Thus, we define our estimate of the tire radius as:

$$r_{optimal} = \max\{r : \tilde{\Delta}_r < \epsilon\} \quad (3.11)$$

In other words, the largest disk which can be superposed on the frame and still produce no more than  $\epsilon$  error will be the best estimate of the tire. Note that, unlike in the previous method, we cannot determine the wheel radius separately. Also, the observations about efficient realizations described previously are applicable in this method.

Once the tire radius is available, the algorithm can determine the wheelbase (the distance between the centers of the front and rear tires) by locating two wheel-rubber edges or two fully dark disks of the (now) known radius, in the two cases described above, respectively. For this purpose, we would use a single video frame in which

both tires are visible. In fact, a good frame would contain the two tires at roughly the same angular distance from SVC's axis of view. The tire size estimate computed using the above methods could also be used to determine other tire parameters, such as remote tire pressure measurement [97].

### 3.2.4 Make Recognition

The problem of make and model recognition (MMR) has been actively researched in the last few years, primarily to assist law enforcement authorities in identifying errant drivers. Significant advances have been made in automatic license plate recognition, and such technologies are deployed in many highway and border surveillance systems. These systems use optical character recognition to obtain the license plate information which may be used to query a database of vehicle/driver information indexed by the license number. Recent studies have focussed on extracting the make/model information directly from the vehicle as a means to supplement and verify the information returned by the database. This is particularly useful when the license plates are missing/tampered or otherwise unintelligible to the camera.

We can broadly classify the various MMR techniques into two categories – the ones based on feature matching and the ones based on appearance matching. In [98], Petrovic and Cootes describe the use of gradient features for vehicle identification. Clady proposes a two stage method based on oriented contours which uses multiple frontal images and is claimed to be robust to partial occlusion [99]. Methods based on scale invariant feature transform (SIFT) proposed by Lowe [100] have also been applied to the MMR problem [101, 102]. Appearance based methods use pixel intensities and provide more global features. Zafar discusses the use of appearance matching using principle component analysis (PCA) and two-dimensional linear discriminant analysis (2D-LDA) [103]. This approach is inspired by the application of these techniques in face recognition systems.

The above-mentioned methods are promising solutions for the MMR problem. They can be applied in our system with the following modification. The selection of a region-of-interest (ROI) is critical to the success of any recognition algorithm. Many of the techniques above use existing license plate recognition tools to locate the license plate. This gives the algorithms adequate reference to demarcate the ROI in the image. While all vehicles have a rear license plate, in our sensor deployment scheme (Figure 3.2), the rear plate and the rear view of the vehicle cannot be used. Since, many places in the United States do not require a front license plate, there is practically no reference for an MMR algorithm to select the ROI. We select a large ROI containing the entire front grille using the vehicle body type information.

We construct an experimental make recognition system which uses the front view of a vehicle (and the body type information provided manually) with no other spatial reference objects. We work with vehicles from a small number of car makers and perform feature matching using histograms of edge orientations on the front grille. Let the video frame in which Make recognition is performed be represented by  $V_{FVC}$  and the foreground mask generated by the vehicle detection system be  $F$  for this frame. Recall that  $f(i, j) = 1$  implies that the pixel located at  $(i, j)$  belongs to the foreground object (the vehicle). Assuming that the object detection can reliably classify the pixels, we can define a bounding box around the vehicle. Let  $(i_1, j_1)$  and  $(i_2, j_2)$  be the defining vertices of a rectangular box ( $i_1 < i_2$  and  $j_1 < j_2$ ) such that,

$$f(i, j) = 1 \implies i_1 \leq i \leq i_2 \text{ and } j_1 \leq j \leq j_2$$

Then, the geometric center of the vehicle in the image is given by  $(c_i, c_j) = (\frac{i_1+i_2}{2}, \frac{j_1+j_2}{2})$ . The region of interest is specified as a rectangular region whose defining vertices are given by  $(i_1, c_j)$  and  $(i_2, c_j + \epsilon)$ . Note that this definition covers the entire width of the vehicle's front and considers the area up to  $\epsilon$  pixels below the geometric center of the vehicle. Different values of  $\epsilon$  can be used with a larger value selected when the vehicle is a truck or an SUV than when it is a sedan or a hatchback. Thus, we define the ROI as the pixels in  $V_{FVC}$  such that  $i_1 \leq i \leq i_2$  and  $c_j \leq j \leq c_j + \epsilon$ . This method

tends to over-select the region where desired features are located but we prefer it over under-selection where the key features may lie outside the ROI.

Having decided on the ROI, we construct a histogram of edge features using the Sobel operator [104]. More specifically, we evaluate the gradient at each pixel inside the ROI in terms of magnitude and orientation. Consider a pixel in the video frame at a location  $(p, q)$ , inside the ROI. We represent the Sobel operator as a 2D array or a spatial filter,

$$M = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad (3.12)$$

Then, the operation of pointwise multiplication and summation can be seen as a 2D convolution after shifting the filter to the desired location. Let  $C_x = \{c_x(i, j)\}$  such that  $c_x(i, j) = V_{FVC} * M$  be the two dimensional array containing the filter output when  $M$  is shifted so that its center is at the position  $(i, j)$ . Note that  $C_x$  contains the edge information in the horizontal direction. Similarly, we can compute the edge information in the vertical direction  $c_y(i, j) = V_{FVC} * M^T$ .

The magnitude of the edge at any point  $(i, j)$  is given by

$$\sigma = \sqrt{c_x^2(i, j) + c_y^2(i, j)}, \quad (3.13)$$

and the orientation is given by

$$\theta = \arctan c_y(i, j)/c_x(i, j), \quad (3.14)$$

where  $-\pi/2 \leq \theta \leq \pi/2$ . We compute the edge orientation at each point  $(p, q)$  inside the ROI and construct an  $N$ -point histogram of these values. Let  $h_\alpha^N$  be the  $N$ -point histogram of edge orientations for an arbitrary vehicle  $\alpha$ . We select a front view image for vehicles from each car maker and generate the histograms which are used as the signatures of the particular make,  $h_1^N, h_2^N, \dots, h_k^N$ . We further process the histograms by normalizing them with respect to the size of the ROI in each of the Make. Thus,  $\tilde{h}_1 = \frac{1}{w \times t} h_1^N$  where  $w$  and  $t$  are the width and height of the ROI in the image of car maker 1. The other histograms are similarly normalized. We then

compute the histogram for an oncoming vehicle  $h_{test}^N$  and normalize it with respect to its own ROI to obtain  $\tilde{h}_{test}$ . Finally, we compute the correlation coefficient as the measure of similarity between the test vehicle and the  $r^{th}$  make template. This is defined as

$$\rho_r = \frac{\tilde{h}_r^T \tilde{h}_{test}}{\sqrt{\tilde{h}_r^T \tilde{h}_r} \sqrt{\tilde{h}_{test}^T \tilde{h}_{test}}} \quad (3.15)$$

Then, the most likely make is given by,

$$\alpha = \arg \max_r (\rho_r), \quad (3.16)$$

where  $\alpha$  is the index number associated with each car maker.

With the information about the vehicle's make obtained above and the exterior dimensions obtained using the methods described in Section 3.2.2, we can estimate the likely model of the vehicle (or a set of most likely models). This can be achieved by querying a database of vehicle information. The make and model information is used to determine the expected range of "normal" behavior for the various measurements made by the system.

### 3.2.5 Squat And Bounce Analysis

Our goal is to detect, from the exterior appearance and motion, if a vehicle is carrying excess load which could potentially include illegal goods or persons. Even otherwise, overloading is often associated with compromised handling and braking [105], thus requiring attention and monitoring. We assume in this analysis that the vehicles are not modified for extra load-bearing, in which case the visible signs of loading would be absent.

We consider both the static and the dynamic behavior of the vehicle as seen by the SVC. We denote the measurement of lowering of the vehicle upon loading as "squat analysis" while the study of the vertical oscillations induced by a speed bump is denoted as "bounce analysis."

**Squat analysis:** Addition of a large load into a vehicle’s trunk causes two visible effects. The rear tires might appear more flat than the front tires, and the gap between the rear tire and the body (the wheel well) may appear smaller than that between the front tire and the body. While the flatness of the tires can be easily compensated by over-inflation, the lowering of the body cannot be avoided without modifying the suspension. Therefore, we focus only on measuring the gaps.

Similar to the problem of accurate tire extraction, the problem of visual determination of rear-heaviness has not been explored sufficiently. Recall Figure 3.7 where the side view of a vehicle’s tire is illustrated. The region above the tire, marked as shadow is the gap we are interested in characterizing. More specifically, we need an accurate estimate of the thickness of this gap at a point which is vertically aligned with the center of the tire. Let us denote this value by  $\delta$ . Ideally, one would like to compare this value with a predefined normal  $\delta_{norm}$  for a vehicle of the same make and model. If the measured gap is significantly smaller,  $\delta_{norm} - \delta > \tau$ , it would be deemed anomalous. On the other hand, one can also compare the  $\delta$  value for the front and rear tires. This approach is more appealing because it does not depend on the knowledge of the make and model (and the normal values associated with it), and the decision threshold can be specified directly in pixels, thus avoiding additional errors that may arise from the image co-ordinate to world co-ordinate mapping.

Our approach to determining the thickness of the gaps is by using the information about the tire centers. Consider for example the rear tire of a vehicle which has been analyzed by the tire size estimation method described earlier. Let  $(x_t, y_t)$  be the location of the tire center and  $r_t$  be the best estimate of its radius. Our goal is to find a point  $(x^*, y^*)$  such that  $x^* = x_t$  and  $y_t - y^*$  is the smallest distance from the tire center to the vehicle body directly above it. Then, the gap thickness  $\delta$  is given by  $\delta = y_t - (y^* + r_t)$  (recall that  $y$  increases from top to bottom).

An identical set of operations would be performed on the front tire and the gap value recorded as  $\delta_{front}$ . Then, the decision about rear-heaviness is made if  $\delta_{front} - \delta_{rear} > \tau$  where  $\tau$  is an experimentally determined constant. Note that, the difference



is not absolute. This is because, under normal conditions, some front engine cars may have lower front than the rear. Thus, the method may detect a false positive if absolute difference is taken.

**Bounce analysis:** The suspension of a vehicle is designed so that it can navigate uneven road surfaces in an optimal manner. Specifically, the parameters are adjusted so as to control the magnitude, frequency and decay of the oscillations induced by such an uneven surface. A speed bump is a particular kind of uneven surface which excites the oscillations during a very short duration of contact with the vehicle. Thus, the oscillations caused by a speed bump can be treated as the impulse response of the vehicle. Without delving into the physics of loaded springs, we state that this response is affected by the amount of load carried by the vehicle. Our goal is to observe and characterize this response using visual methods. The concept of overload detection by visually analyzing the vehicle's bounce is discussed by Dockstader in [106].

Our approach to observe the oscillations is to select a point ( $P$ ) on the vehicle and record its trajectory across the frames. Since we are interested in the response at the vehicle's rear, we can select the tip of the trunk, or the highest point in the vehicle directly above the rear tire's center. The methods for vehicle detection and tire location, as already described above, can be directly applied to note the position of the chosen point in every frame. This can be referred to as  $(x_n^P, y_n^P)$  where  $n$  denotes the frame number. The vertical position  $y_n^P$  describes the oscillations and can be used to determine the oscillation parameters including amplitude and the rate of decay. The parameters hence estimated would be compared with the known parameters observed when another vehicle of the same make and model was excited by the bump under loaded and unloaded conditions.

### 3.2.6 Trajectory Analysis for Anomaly Detection

Video surveillance of vehicles has emerged as the preferred technology for traffic monitoring, public safety, and law enforcement applications. In recent years, several

efforts have been made to incorporate automation into the systems using image and video analysis techniques [39, 107, 108].

The range of behavioral analysis for vehicles is limited when compared with human subjects [109] due to rigid shapes and regularity of motion. Still, timely detection of anomalies can potentially prevent mishaps and damage to life/property. Traditionally such anomaly detection is achieved using trajectory clustering and identifying deviations from the clusters [110–112]. Although the trajectory given by a tracker includes both position and velocity estimates, many approaches end up discarding the velocity changes which can be highly informative. Furthermore, path learning via clustering is affected by issues such as unequal trajectory lengths, choice of path models, and effects of distance metric.

We approach the task by hypothesizing that the velocity and shape of the trajectory can be analyzed separately and that in most situations patterns in the velocity provide enough information to detect anomalies. Basharat *et al.* describe modeling of motion patterns which does not require clustering [113]. We illustrate this with an example of vehicles approaching a check point. Medioni *et al.* use a clustering-based method to detect if a vehicle tries to evade the check point [111]. However, their approach does not detect if the vehicle slows down or stops and then proceeds toward the check point or picks up speed as it approaches indicating an intention of crashing through it. In contrast, a simple velocity differential [107] can detect these patterns.

We propose to form path models of velocities as functions of position rather than time. We also estimate the velocity models for finite sub-regions of the field of view. This is different from identifying sub-paths after clustering. In each such division, we represent the velocity with a Gaussian distribution. Note that the use of finite spatial divisions prevents the issue of unequal trajectory lengths. This allows an anomaly detection system to utilize the patterns in the velocity and also allows better modeling of the decision function than the simple thresholding used in [107]. We further observe that such velocity clusters would intrinsically have high variance because people drive at different speeds, often near the speed limits. Thus, we first scale the velocities with

the average speed estimated when the vehicle is relatively far from the observation point. These methods are described in Section 3.2.8.

Next we observe that detecting deviations from “normal” velocity would be easier if there is only one set of normal driving patterns. This is not true in general. For instance, a vehicle would slow down from its original speeds when making a turn or taking a highway exit. Thus, if a vehicle’s velocity matches a cluster corresponding to vehicles that made a turn, this vehicle should also be making a turn. We use template matching to identify significant maneuvers from the trajectory shape in Section 3.2.9. By separating the temporal and spatial analyses of the trajectory, we obtain better flexibility to define the rules of anomalous behavior.

The above analyses assume that vehicles are point objects moving in a 2D plane. In Section 3.2.7 we define a co-ordinate transformation in which the motion of the vehicle consists of two components – motion along the direction of the road and motion perpendicular to it. This transformation allows us to represent the velocity with a 1D descriptor (by ignoring the perpendicular component). It also improves the shape analysis by discarding the curvature caused by naturally curved roads.

### 3.2.7 Trajectory Estimation and Co-Ordinate Mapping

The tasks of robust object detection and tracking are central to many image analysis problems. Over the years many interesting methods have been proposed to detect moving objects and to track them. Haritaoglu *et al.* [114] and Stauffer and Grimson [86] describe popular methods for object detection and tracking. A comprehensive survey on tracking is provided by Yilmaz [115].

In this paper, we only provide some qualitative features of our tracking system and refer to appropriate publications for further details. Our system does object detection and tracking in a continuous manner. Foreground objects are detected by adaptive background subtraction as described in [116]. The method consists of adaptive computation of classification thresholds and background model updates at

variable learning rates. The output of detection (foreground blobs) is characterized by connected components analysis and is used to create targets which are tracked using a particle filter framework as described in [117]. Particle filter is a Bayesian framework which is used to estimate the “state” of a target being tracked. In our system, the object’s state includes its position and size. The filter generates several hypotheses (particles) of the new state and computes the weighted average of these hypotheses as the estimate of the new state. The weights are computed by determining the similarity of the hypotheses with respect to some chosen object features. We use color and edge orientation as the objects features. Arulampalam *et al.* provide a detailed tutorial on the application of particle filters in object tracking [118].

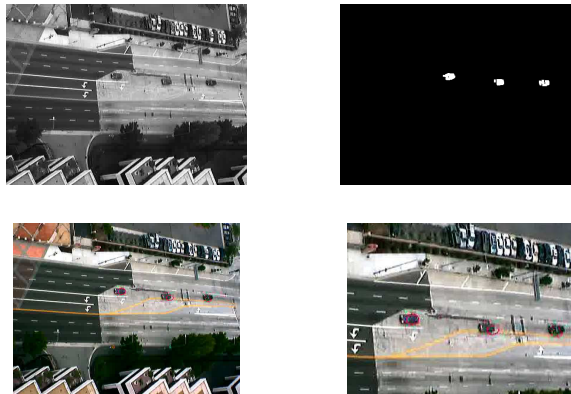


Fig. 3.11. An example of results produced by our object detection and tracking systems – from left to right: original frame and the foreground mask (top), three tracked vehicles and a close-up of the scene (bottom).

Figure 3.11 shows a snapshot from one of our test sequences. The corresponding outputs from background subtraction and object tracking are also provided. For the rest of this paper, we assume that the vehicle’s position in the video frame is available. More precisely, a vehicle’s “image trajectory” is defined as its position in the frame as a function of time (or frame number), and is provided by the tracker. Let this image trajectory be represented by

$$\Phi_c^i = \{(x_t^i, y_t^i) : 1 \leq t \leq T_c\}, \quad (3.17)$$

where the subscript  $c$  denotes a particular object. For the sake of brevity, this subscript is not used in further discussion. Similarly,  $T_c$  is the lifetime of the object in number of frames,  $t$  is the time index, and the superscript  $i$  denotes image coordinates. We follow the convention that  $x$  increases from left to right and  $y$  from top to bottom.

For any object, one would like to transform the trajectory into actual ground co-ordinates in order to remove the distortion due to camera perspective. Let this transformed trajectory be represent by  $\Phi^g$  with definition similar to  $\Phi^i$ . In our application scenario where an object can be modeled as a point, an ideal transformation would result in an aerial view of the trajectory. This can be achieved with a planar homography [119] or an approximation of the same. Our preferred method of perspective correction uses look-up tables because it is fast and requires minimal calibration. However, from our experiments we have observed that ground co-ordinates are often not the most efficient representation for trajectory analysis. For example, vehicles moving along a naturally curved road traverse a curved trajectory in ground co-ordinates whereas their motion is a “straight line” with respect to the road.

We propose analyzing vehicle trajectories in a hypothetical co-ordinate system in which the distances are arbitrarily close to the distance in ground co-ordinates but the axes are defined with respect to the shape of the roads. Thus, a vehicle moving parallel to the center median is considered as moving along a straight line. This transformation results in two immediate benefits – (1) the vehicle’s velocity can be specified with a single component (in the direction of the road) and (2) false curve/turn detection along gently curved roads can be suppressed. Again, we utilize 2D look-up tables to achieve this transformation from ground to the hypothetical co-ordinate system. This table is constructed interactively by user input. This is explained next with an example.

Consider the situation in Figure 3.12 where the parallel curves represent the boundaries of a curved road. This image is assumed to be perspective corrected. We next assume that this corresponds to a road in the hypothetical space whose

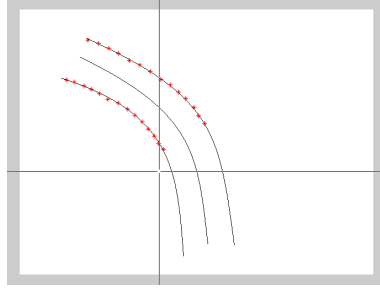


Fig. 3.12. An illustration of user specified co-ordinate transformation. The points marked with red crosses divide the “road” into equal distance segments.

length and width matches the actual section of the road but which is not curved. We establish an approximate point correspondence by asking a user to divide the two curves into equal intervals (equal in the ground co-ordinates). Depending on the number of divisions, corresponding points are located on the hypothetical road. This completes the look-up table where the input is a position in the ground co-ordinates and the output is the corresponding position in the hypothetical system. For points not aligned with the input grid, 2D Shepard interpolation [69] is used. This look-up table is later used to transform the vehicle’s trajectory into the desired co-ordinate system.

Note that a single look-up table can be created to map the object position from the image co-ordinates directly to the hypothetical co-ordinates. That is, no explicit perspective correction step is required. This is done by taking the user input (or an equivalent method) to divide the road boundaries in the image co-ordinates. This is our recommended approach and has been used in our experiments. Let the final trajectory be represented as

$$\Phi^h = \{(x_t^h, y_t^h) : 1 \leq t \leq T\}, \quad (3.18)$$

where the position  $x^h$  is along the road and  $y^h$  is perpendicular to it.

### 3.2.8 Velocity Analysis

We first introduce our proposed representation of the velocity as a function of the vehicle's position. Note first that we only consider the axial component of the velocity (parallel to the road) in this section. Let  $x(t)$  and  $v(t)$  represent the (axial) position and velocity estimates of a vehicle at time  $t$ . At this point we make the following key observations:

- Even in the absence of obstacles there is natural variation in “normal” driving speeds due to curves, turns, and inclinations/declinations. However there will not be many variations when looking at a small portion of the roadway. Therefore, patterns of normal velocities should be estimated for different portions of the road.
- A fast moving vehicle will have fewer samples in its trajectory while a slow moving vehicle will have more samples. Thus, the  $k^{th}$  sample of these vehicles' trajectories would not correspond to the same portion of the road.
- Resampling of trajectories to get the same number of samples may solve this problem of mis-registration. For instance, if a vehicle A drives twice as fast as another vehicle B we can upsample A's trajectory by a factor of 2. If the sampling time (inverse of the frame rate) is  $\tau$  then the  $k^{th}$  sample in A's resampled trajectory corresponds to time  $k\tau/2$  (since A's entry) while in B's trajectory it represents time  $k\tau$ . However, their spatial position would be very similar due to the inverse effect of velocities.

The third observation motivated us to represent the velocities as function of position  $u(x)$ . This representation removes the need for resampling because the velocities of two vehicles at same position *are* comparable. To allow easier indexing we quantize the position  $x$  by dividing the road into  $M$  smaller segments. The velocity can be specified using the index of the segment in which  $x$  lies. Let the length of the road in the field of view be  $L$  (feet/meter) where we follow the convention that  $x$  increases

from 0 to  $L$  as the vehicle gets closer to the observation point. We represent the segments by a set of points  $\{s_0, s_1, \dots, s_M\}$  such that  $s_0 = 0$ ,  $s_M = L$ , and

$$s_i - s_{i-1} = \frac{L}{M}, \quad \forall i \in \{1, 2, \dots, M\}. \quad (3.19)$$

With the length of the segments  $\Delta = \frac{L}{M}$ , the velocity  $u(x)$  can be represented using  $\bar{u}(i)$  where

$$i = \lfloor x/\Delta \rfloor \quad (3.20)$$

indicates the  $i^{\text{th}}$  segment and  $\lfloor \cdot \rfloor$  denotes integer floor operation. The length  $\Delta$  (or equivalently,  $M$ ) would be chosen small (large) enough to prevent velocity behavior changes within a segment while not over-segmenting because that would discard spatial averaging.

In the final pre-processing stage, we propose to scale the velocities with respect to each vehicle's average speed. This allows us to emphasize the *changes* in the velocity rather than the absolute speeds. Further, it also leads to smaller variances in the normal behavior distributions which we describe next. Note that our goal is to achieve real-time operation. Thus, the average speed must be estimated with only part of the observations. We estimate the average speed  $u_{avg}$  over the velocities observed when  $x < L/2$ . This was motivated by the assumption that a visible change in behavior is more likely to occur when a vehicle is near the observation point (e.g. a check point). Hence it will exhibit more "typical" behavior when in the far field of view. The final form of the velocity is given by

$$c(i) = \bar{u}(i)/u_{avg}. \quad (3.21)$$

The normal velocity behavior for each segment  $i$  is modeled with a mixture of Gaussian distributions specified by the mean  $\mu_{i,k}$  and variance  $\sigma_{i,k}^2$  of vehicle velocities for the  $k^{\text{th}}$  Gaussian component. One can use clustering to obtain these parameters. We use a simpler context-based approach in which the number of Gaussian components is related to the number of maneuvers observed in the segment. For instance, a "straight-only" lane with no curvature and/or grade will require only one component while other sections may require components for turns and straight line course.



In the testing phase, the scaled velocity  $c(i)$  of an oncoming vehicle is compared with the components in the  $i^{\text{th}}$  segment. An “anomaly” is detected if

$$|c(i) - \mu_{i,k}| > \alpha \cdot \sigma_{i,k} \quad \forall k. \quad (3.22)$$

Here  $\alpha$  is a scalar that determines the threshold. Under Gaussian assumption, a value of  $\alpha = 2$  would imply approximately 5% outliers. In the case when there exists some component such that the velocity difference is within the threshold, we validate the selection of this component by estimating the maneuver with shape analysis (described next). This ensures, for example, that if a vehicle’s velocity matches the component obtained for turning vehicles then that vehicle should also turn. A mismatch in the anticipated and observed maneuver would also result in an anomaly detection.

### 3.2.9 Shape Analysis

We now describe our methods to characterize the shape of a vehicle’s trajectory. In this exercise our goal is to develop techniques to identify significant maneuvers like turns and lane changes. We construct a library of shapes associated with different maneuvers and match the vehicle’s trajectory to these templates. Note that we perform template matching in the hypothetical co-ordinates defined earlier and both  $x^h$  and  $y^h$  are used to define the vehicle position. In order to simplify discussion (and implementation) we use a complex number representation of the trajectory in this section.

Let a spatial trajectory be defined as:

$$P = \{p_t = x_t^h + jy_t^h : 1 \leq t \leq N\}, \quad (3.23)$$

where  $N$  is the number of samples in the trajectory and  $j$  denotes the square root of  $-1$ . We obtain a trajectory template for each significant maneuver. Thus for  $k$  maneuvers we obtain the templates  $P_1, P_2, \dots, P_k$ .

## Procrustes Analysis

According to Dryden [120], “shape is all the geometrical information that remains when location, scale, and rotational effects are filtered out from an object.” For analysis purposes we represent the shape of an object by a finite number of pixels on the object’s surface. There are different methods to estimate the similarity of object shapes. Procrustes shape analysis is one such method which is invariant to scale, translation, and rotation [121]. Our use of Procrustes analysis for turn detection is inspired by a similar work by Harguess and Aggarwal [122]. Our approach is different because we do not pre-segment the trajectories resulting in a real-time analysis.

Let each object be represented by a  $N \times 1$  vector where  $N$  is the number of 2D points on the object’s surface (in our case, the number of points on the trajectory). We represent the two trajectories as  $U = (U_1, U_2, \dots, U_N)^T$  and  $V = (V_1, V_2, \dots, V_N)^T$  where  $U, V \in \mathbb{C}^N$ . That is, each 2D point is represented by a complex number as in Equation 3.23. The Procrustes distance is a minimum sum of squared distances shape metric that requires shapes with one-to-one corresponding point pairs. After filtering the location, scale, and rotational effects the minimum Procrustes distance configurations that is used in this paper is defined by:

$$d_F(U, V) = \left(1 - \frac{V^* U U^* V}{U^* U V^* V}\right)^{1/2}. \quad (3.24)$$

## Shape Matching with Sliding Windows

Our motivation for using a sliding window based method arises from the need for real-time operation. We observe that the approach used by Harguess [122] creates significant delay in the analysis because it requires the complete trajectory (or at least a large part of it) in order to detect steps, ramps, and impulses. Thus we devise a method to use template matching without a need for explicit segmentation of the trajectory.

Let us represent a window  $W$  as an index set defined by the two end points  $\tau_1$  and  $\tau_2$ :

$$W = \{\tau_1, \tau_1 + 1, \dots, \tau_2 : 1 \leq \tau_1 < \tau_2 \leq N\}. \quad (3.25)$$

We ensure sufficient samples in the windows by the constraint that  $\tau_2 - \tau_1 > n_{min}$  ( $n_{min} = 20$  in our experiments). The shape to be matched is constructed from the trajectory points located in the window. Thus,

$$\beta = \{p_t \in P : t \in W\}. \quad (3.26)$$

The shape descriptor thus obtained is a complex vector of length  $\tau_2 - \tau_1 + 1$ . In order to compute the Procrustes distance with respect to the  $k$  templates described above, two operations are needed. First, the vector is centered by subtracting the median value. Next, the centered vector must be resampled so that it contains exactly the same number of elements as the  $l^{th}$  template for each  $1 \leq l \leq k$ . The resampling is achieved using linear interpolation. Let  $\hat{\beta}$  represent the centered and length-adjusted shape vector corresponding to a given window  $W$ .

Let  $\Delta$  represent the Procrustes distances for  $\hat{\beta}$  for each of the template. That is,

$$\Delta = \{\delta_l : 1 \leq l \leq k\}, \quad (3.27)$$

where  $\delta_l$  is the distance from the  $l^{th}$  template shape. We declare the occurrence of a particular maneuver when the distance of its template is significantly smaller than the other distances. From our experiments, we observed that a 25 – 35% difference works well. We use 30% as the decision threshold in our experiments. An event is declared by marking  $p_{\tau_2}$  as the position of occurrence.

Since the only two parameters determining a window are the end points, we now explain how these points get updated during the process.

- If a turn is detected, all but the very latest parts of the current window are discarded, and a new window is created with the smallest permissible size ( $n_{min}$ ).

Thus,

$$\tau_1^{new} = \tau_2^{old} - \frac{n_{min}}{2} \text{ and } \tau_2^{new} = \tau_2^{old} + \frac{n_{min}}{2}, \quad (3.28)$$

where the superscripts *new* and *old* have been inserted for illustration purpose.

- If a turn is detected and is the same type as the last detected turn, a u-turn is inferred if the two events occurred within a chosen distance from each other. We declare a u-turn if two identical turns occur within 50 samples of each other. The end points are updated as earlier.
- If the event detected is a straight line course or no event is detected, only  $\tau_2$  is modified – increased by one if the end of the trajectory has not been reached. The analysis terminates when the end is reached.
- If the event detected is a straight line course or no event is detected, one can optionally choose to discard the oldest points in the window if the window exceeds a certain size threshold. We discard the older half of the samples if the window becomes larger than 100 samples.

### 3.2.10 Color Correction for Object Tracking

Video surveillance is a popular tool used by law enforcement and security personnel. The deployment of multiple cameras is particularly useful for remotely observing extended areas. These cameras may have different characteristics and may be operating under dissimilar illumination conditions. Thus, their outputs will exhibit considerable difference even when imaging the same object. While the white balancing algorithm in many cameras tries to adjust the colors, its function may be affected by the foreground object. This is an undesirable situation for video analysis purposes.

Many image analysis tasks (such as tracking and identification) utilize color features because they are more invariant to shape and orientation changes. However, the variability across the outputs of many cameras makes color a less reliable property when observing a subject over an extended period of time and/or region in space. For example, a parking facility may be interested in tracking a violating vehicle even

after it exits the parking area, but this task may be difficult if the lighting conditions inside the parking area are significantly different from the outside.

Some approaches attempt to construct more robust color features (histograms and correlograms) by transforming the colors to a different color space, such as HSV or LAB, instead of the devices' RGB [123]. These methods result in only marginal improvements. Porikli proposed an inter-camera calibration technique based on a correlation matrix analysis [124]. Gilbert and Bowden describe an incremental learning approach to determining the inter-camera transformation [125]. Both methods would result in a transform for every camera pair in the network. A conversion vector based method was described by Choi *et al.* and shown to produce promising results [126]. This approach is very suitable for real-time video applications because of its simplicity. In this section, we propose a systematic approach to solving the problem of color consistency using principles of colorimetry and color management for imaging and display devices [127].

Consider a public site (such as an airport or a shopping mall) which is surveilled with a network of  $k$  video cameras, possibly under  $k$  different illumination conditions. In this work, we assume that each illumination remains fairly constant over time. Thus, we can combine a camera-illumination pair into a single entity which we denote as an *imaging unit* (IU). Let the output of the  $k$  IUs at any time  $t$  be represented by  $I_0(t), I_1(t), \dots, I_{k-1}(t)$ . Note that each  $I_r(t)$  consists of pixel-wise color co-ordinates for the scene. For all practical purposes, we can safely assume that the color at a pixel is specified as an RGB value in the camera output. Further, we note that it is unrealistic to assume that an object would be imaged at the same time by all the IUs. Therefore, we drop the time index in subsequent discussions.

It is natural to expect significant variation in the images ( $I_r$ ) of an object generated by the different imaging units. If the subject entered the surveillance system from IU  $a$  and subsequently moved into the field of IU  $b$ , one would like to transform  $I_b$  such that the colors resemble those in  $I_a$ . Thus, we would need an RGB-RGB transform for every pair of imaging units in the system.

Our first step to solving this problem is to prevent the quadratic growth in the number of transforms needed ( $\binom{k}{2}$  for  $k$  IUs). We do this by modeling the image analysis system as an “observer.” Note that while the term observer is inspired by colorimetry, its interpretation here is very different from the CIE standard observer [3]. Our observer can be viewed as a standard camera operating under chosen illumination conditions – in other words, an imaging unit as defined above. Therefore, our goal is to match the output of every IU with that of the reference IU. The problem now scales linearly with the number of IUs. Without loss of generality we designate  $I_0$  as the output from the reference IU.

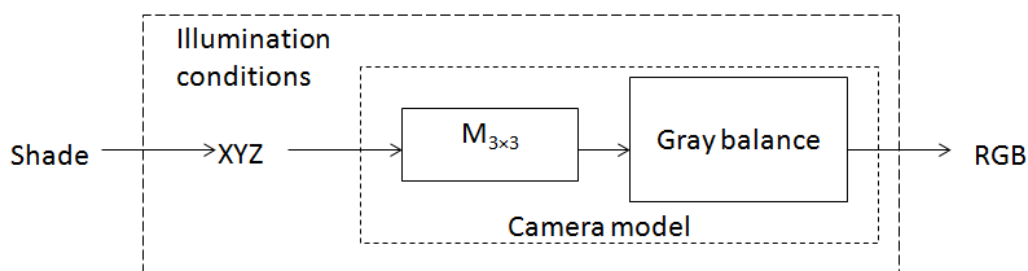


Fig. 3.13. A schematic diagram of an imaging unit.

Since the effects of illumination are abstracted into the IUs, the only invariant property of a subject would be its surface reflectance. However, to make the discussion more intuitive, we refer to this as “shade.” Shade is the identifier that a human would associate with an object with such surface reflectance and viewed under typical illumination. Thus, we require that subjects with the same shade produce similar RGB values in all IUs. The actual relation between the shade and the observed tri-stimuli (under given illumination) is difficult to establish and requires extensive radiometric measurements. Instead, we make this key assumption that the two quantities are related through the illumination white point. The resultant tri-stimuli in CIE XYZ would be the input to the camera which itself can be modeled with gray balancing and a linear transformation [128]. The complete model of an IU is illustrated in Figure 3.13.

Particle filter is popularly used for object tracking because it has been shown to be very successful for non-linear and non-Gaussian dynamic state estimation problems and is very reliable in cases like clutter and occlusions [129]. Each hypothesized state is referred to as a particle and a weighted sum of all particles gives the final estimate of the state. An observation likelihood model assigns each particle a weight according to how this particle resembles the target object. This model or rule is determined quantitatively by measuring the dissimilarity between the feature (in our case, color is used) distributions of the target  $q$  and the particle  $p$ . We use the Bhattacharyya distance as the metric of distance which is given by:

$$d[p, q] = \sqrt{1 - \sum_{u=1}^m \sqrt{p_u q_u}}, \quad (3.29)$$

where  $u$  is the bin index of the color distributions. The particle weights are approximated using a Gaussian distribution of the Bhattacharyya distances. Similar distributions result in smaller  $d[p, q]$ .

Color distributions have been widely used for tracking problems [130] because they are robust to partial occlusion, and are rotation and scale invariant. The color distribution is expressed by an  $m$ -bin histogram, whose components are normalized so that the sum of all bins equals one. For a region  $A$  in an image, given a set of  $n$  pixels in  $A$  denoted by  $B = \{x_i, i = 1, 2, \dots, n\} \in A$ , the  $m$ -bin color histogram  $T(A) = \{h_j, j = 1, 2, \dots, m\}$  can be obtained by assigning each pixel,  $x_i$  to a bin, by the following equation:

$$h_j = \frac{1}{n} \sum_{x_i \in B} \delta_j[b(x_i)], \quad (3.30)$$

where  $b(x_i)$  is the bin index in which the color component at  $x_i$  falls and  $\delta$  is the Kronecker delta function.

### Detailed Formulation

An advantage of selecting a reference is that the discussion and modeling can be build for two camera systems ( $k = 2$ ) and extended to multi-camera systems

without any changes. In this section, we first describe a model-based method to match the output of a test imaging unit ( $I_1$ ) with that of a reference IU ( $I_0$ ), and then approximate this transformation using 3D look-up tables (LUT). This two-step approach to solving a color correction problem was also proposed by Srivastava *et al.* for color management of display devices [131].

Let  $\mathfrak{S}$  represent the commonly used RGB color space. This is distinct from the linear RGB representation, where the values are proportional to the photon count. The latter are explicitly marked with a superscript (for example,  $R_2^{lin}$ ). We require a function  $\Phi : \mathfrak{S} \rightarrow \mathfrak{S}$  such that  $\Phi(I_1) = I_0$ , for the same shade. If the camera model is represented by  $S$ , then Figure 3.14 illustrates the construction of  $\Phi$ . This method is denoted by CCMX (Camera to Camera Model-Based Transformation) in this thesis. The white correction step consists of a technique to account for the different illumination conditions.

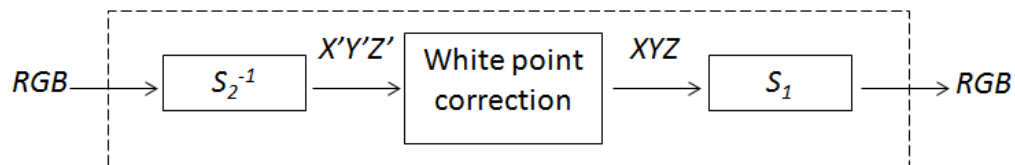


Fig. 3.14. A block diagram of the device to device transformation CCMX ( $\Phi$ ).

We choose a camera model which consists of gray balancing and a linear transform. Thus, a camera model  $S$  transforms a stimulus  $(X, Y, Z)^T$  to  $(R, G, B)^T$  as follows:

$$\begin{bmatrix} R_n^{lin} \\ G_n^{lin} \\ B_n^{lin} \end{bmatrix} = \begin{bmatrix} & & \\ & M_{3 \times 3} & \\ & & \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \quad (3.31)$$

Here  $(R, G, B)_n^{lin}$  signify the normalized linear values and  $M$  is a linear transform. Absolute linear values can be obtained by scaling with the absolute linear values of



the device white. This can also be represented using the inverse but more intuitive normalization operation,

$$R_n^{lin} = \frac{R_{abs}^{lin}}{R_{abs}^{lin w}}; G_n^{lin} = \frac{G_{abs}^{lin}}{G_{abs}^{lin w}}; B_n^{lin} = \frac{B_{abs}^{lin}}{B_{abs}^{lin w}}, \quad (3.32)$$

where  $(R, G, B)_{abs}^{lin}$  represent the absolute linear values which are proportional to the photon count at the camera sensor. Finally, the device RGB output is obtained by gamma correction or gray balancing [1]. This can be represented by a 3D function

$$f : \mathbb{R}^3 \rightarrow \{0, 1, \dots, 255\}^3. \quad (3.33)$$

In our modeling, we use a gain-gamma-offset model [132] for the function  $f$ , and obtain the transformation matrix  $M$  by linear regression. The parameters of the models are computed by measuring the RGB and CIE XYZ values for a small number of printed patches. We used only 24 colored patches (shown in Figure 3.15) and measured the XYZ values with a spectroradiometer PR-705.



Fig. 3.15. A printed sheet of colored patches used to construct the camera models. Courtesy: Norman Koren [8].

While many white point compensation techniques have been proposed in the literature [2], we use a simple rescaling based approach because it is easy to apply in real situations with very little information about the illumination available. Let the reference illumination white point be  $(X, Y, Z)_w^0$  and the test white point be  $(X, Y, Z)_w^1$ .

Then to transform an object's tristimulus  $(X', Y', Z')$  to the reference conditions, we obtain

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{X_w^0}{X_1^0} & 0 & 0 \\ 0 & \frac{Y_w^0}{Y_1^0} & 0 \\ 0 & 0 & \frac{Z_w^0}{Z_1^0} \end{bmatrix} \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix}. \quad (3.34)$$

Our goal is to efficiently color correct the output  $I_1$ . It has been shown that look-up tables (LUT) can be used to achieve complex transformations [66]. LUTs are suitable for hardware implementations and allow elaborate modeling of the system they approximate. This is because we can improve the function  $\Phi$  at the cost of added complexity. But the use of LUTs makes this additional complexity inconsequential.

Let the table look-up operation be represented by a function  $\mathcal{L} : \mathfrak{S} \rightarrow \mathfrak{S}$ . If the function  $\Phi$  has a domain  $D$  and range  $S$ , then we evaluate  $\Phi$  at certain points given by  $D_1 \subset D$ . In our application, we require a 3D LUT whose output is also a 3-tuple. Consider a LUT of size  $n_R \times n_G \times n_B$ , then

$$D_1 = \{r_0, r_1, \dots, r_{n_R-1}\} \times \{g_0, g_1, \dots, g_{n_G-1}\} \quad (3.35)$$

$$\times \{b_0, b_1, \dots, b_{n_B-1}\}, \quad (3.36)$$

where  $\times$  represents a Cartesian product. The output of the LUT is given by

$$\mathcal{L}([r, g, b]) = \begin{cases} \Phi([r, g, b]), & \forall [r, g, b] \in D_1, \\ H([r, g, b]), & \text{otherwise.} \end{cases} \quad (3.37)$$

Here  $H$  represents interpolation which is used to estimate the value of  $\Phi$  using the known values at the neighboring points in  $D_1$ .

## 4. EXPERIMENTAL RESULTS

In this chapter we present the results of testing our proposed methods on different test cases. Note that the evaluation of the methods is different for the problems of color management and visual surveillance. It can be stated that in both cases, the outputs of the methods should be evaluated by human observers. Whereas such “ground truth” can be defined in surveillance problems (such as make of a vehicle) and remains invariant across observers, the perception of color and image quality is highly subjective. Therefore, we adopt a quantitative approach to evaluating color accuracy by estimating the errors in  $\Delta E$  units. The methods for visual surveillance tasks are evaluated against the ground truth.

### 4.1 Color Management

#### 4.1.1 Comparison of LUT With the Models

In order to test the accuracy of the LUT engine against the NLXWP model (Section 2.1), a randomly generated set of 216 RGB values is constructed by first choosing any six integers between 0 and 255. Let  $\alpha = [r_0, r_1, r_2, r_3, r_4, r_5]$ . Then  $S_{test} = \alpha \times \alpha \times \alpha$  is a  $6 \times 6 \times 6$  cube of training data points. Each color in the set is transformed using the two functions -  $\Phi()$  representing the model NLXWP and  $f()$  representing a LUT-based implementation. The deviation of the two outputs is measured in  $\Delta E$  and the average prediction error  $\overline{\Delta E}$  is found to be 3.76  $\Delta E$  units. Figure 4.1 shows the histogram of errors for the testing set.

The plot of  $1 - \sigma$  outliers of the experiment in the  $x - y$  space (Figure 4.2) shows a small drift in the output color. This drift is measured by the separation between crosses (output of NXLWP) and corresponding circles (output of LUT-based

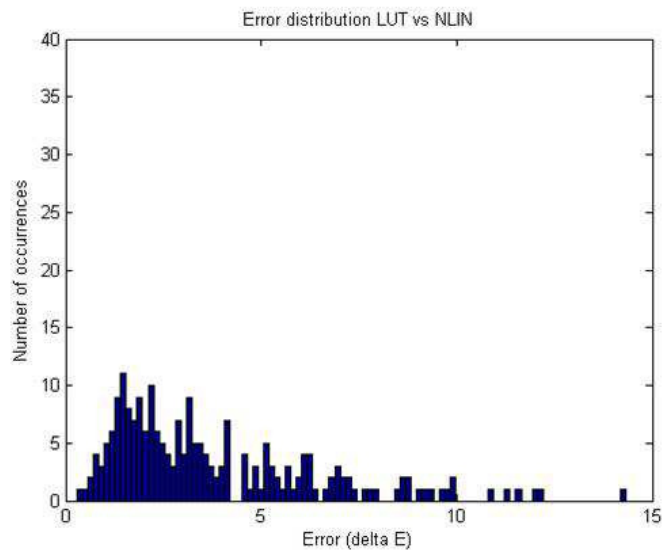


Fig. 4.1. Histogram of errors in LUT based prediction (against NLXWP).

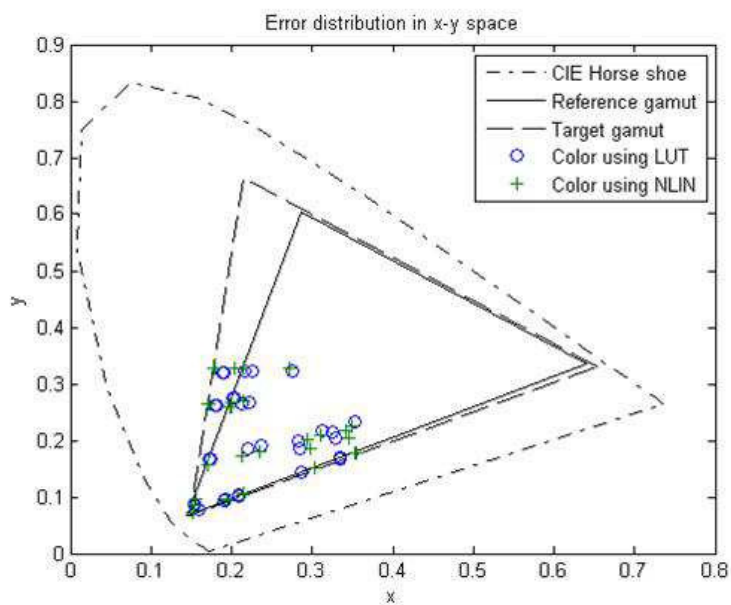


Fig. 4.2. Distribution of the  $1 - \sigma$  error outliers of the LUT-based implementation in  $xy$  space.

prediction). That concluded the first check operation and it is safely inferred that significantly accurate predictions can be obtained using a moderate sized LUT. However, the chromaticity diagram does not provide all the information about the error points. Therefore, all future results are presented in LAB space.

Another experiment for evaluating the three implementations was conducted in which a set of 122 data points (RGB triples) was displayed on the reference monitor and the color measured using PR-705. Then, the three models (NLXWP, LXWP and the LUT-based model) were used to predict the color on the viewing device. The output of each model was compared to the measured values in LAB space and their deviation computed in  $\Delta E$  units. This process is illustrated in Figure 4.3. The number 122 is chosen because we started with a  $5 \times 5 \times 5$  data set and removed the colors which transformed to a location outside of the viewing device gamut.

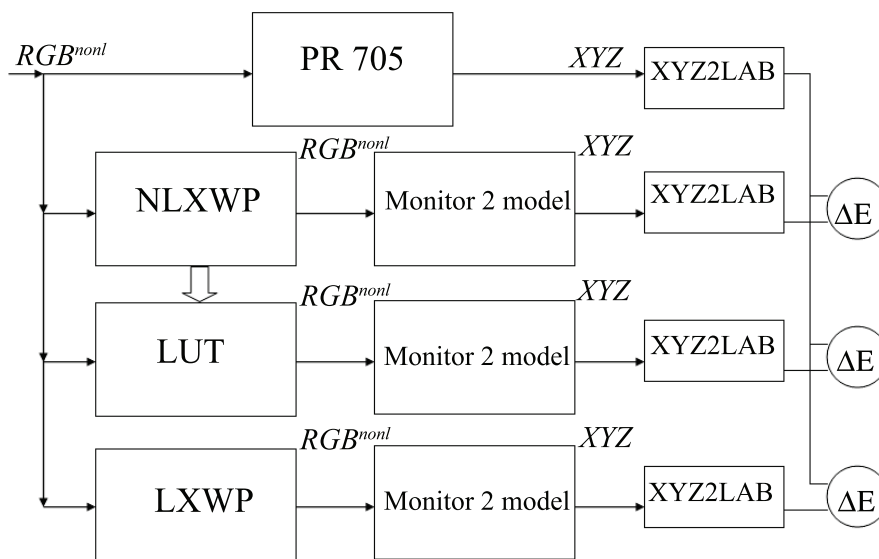


Fig. 4.3. A block diagram for evaluating the three implementation models.

The average error  $\overline{\Delta E}$  was recorded for each case and the outliers were plotted in the LAB space as shown in Figure 4.4. The red stars indicate position of the color as measured on the reference device while blue circles indicate the position of the

color as obtained on the viewing device. The distance between corresponding stars and circles gives a measure of the prediction error. The plots represent outliers for NLXWP (left), LXWP (center) and the LUT-based implementation (right). Table 4.1 shows some statistics for the errors obtained with each model.

Table 4.1  
Testing error statistics (in  $\Delta E$  units) of the three implementations.

Statistic\Model	Non-linear	Linear	Look-up table
Mean	2.9253	10.9185	3.3623
Median	2.7723	9.4240	3.1945
Standard deviation	1.4184	4.8038	1.4318

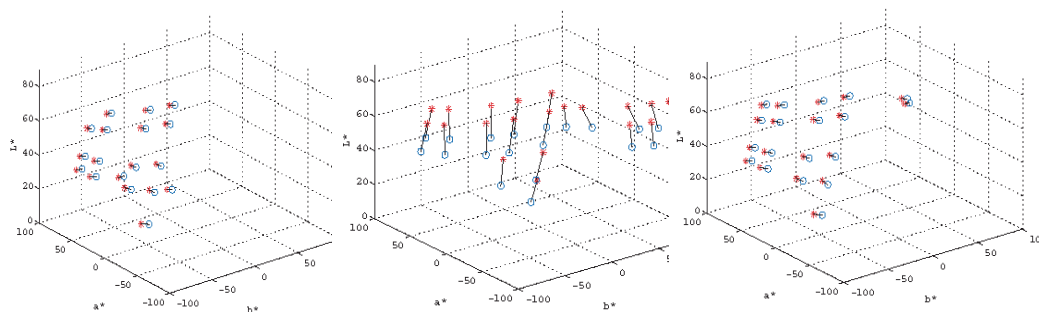


Fig. 4.4. Distribution of the  $1 - \sigma$  error outliers for each of the implementation models in LAB space.

A number of experiments are performed to study and evaluate the impact of varying different system parameters in a LUT-based prediction model. The reference for all experiments is taken as the NLXWP model although actually measuring colors using PR-705 may be more accurate.

We provide the results of the training process described in Section 2.2 for LUTs of three sizes in Table 4.2. These were obtained for a 1000 point training data set as stated earlier. Therefore, the values can be seen as the minimum *cost* at the end of

the optimization process as shown in Figure 2.20 (although there is no optimization in the case of uniform LUT, we can still view the value as the cost after a single iteration). Optimization was done for both symmetric and non-symmetric LUTs and the numbers inside the respective parentheses indicate the percentage reduction in the average prediction error as compared to a LUT of the same size but with uniformly sampled  $\Omega$ . Note that the optimization is carried out only with respect to sampling and all three types of LUTs use the same (separable squared inverse distance) interpolation scheme.

Table 4.2

Final values of the cost function ( $\Delta E$  units) evaluated at the points in the training set at the end of optimization (Numbers inside parentheses indicate percentage improvement over a uniform regular LUT of the same size).

<b>Type\Size</b>	$6 \times 6 \times 6$	$9 \times 9 \times 9$	$12 \times 12 \times 12$
Uniform regular	6.21	3.79	2.79
Symmetric optimal	5.38 (13.36%)	3.39 (10.55%)	2.38 (14.69%)
Non-symmetric optimal	4.87 (21.58%)	2.93 (22.69%)	2.19 (21.51%)

Figure 4.5 shows the progress of optimization for three LUT sizes,  $6 \times 6 \times 6$  (top),  $9 \times 9 \times 9$  (middle) and  $12 \times 12 \times 12$  (bottom), where the dashed line tracks symmetric LUTs and the solid line indicates progress of non-symmetric LUTs. It must be noted that the x-axis is not a time scale. This is because optimizing symmetric and non-symmetric tables takes unequal number of iterations and, hence, unequal time. Instead twenty uniformly spaced iterations are selected and the value of cost function (APE) at that iteration is used to track the progress of optimization process. Since the sampling *map* as shown in Figure 2.20 is initialized with uniform sampling, both optimizations start with the same APE in the first iteration.

These results validate the applicability of our optimization technique to the problem of generating optimal LUTs for color management. It can be seen that by opti-

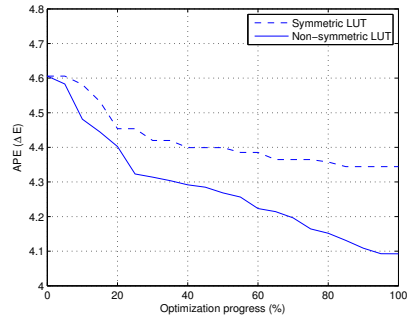
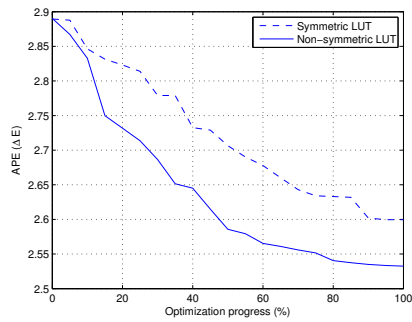
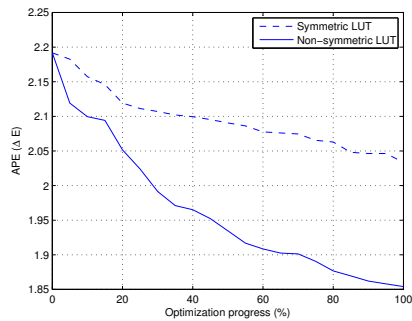
(a)  $6 \times 6 \times 6$ (b)  $9 \times 9 \times 9$ (c)  $12 \times 12 \times 12$ 

Fig. 4.5. Convergence of optimization error for the three LUT.

mizing just one variable in the LUT-based system (sampling of the RGB space), we can reduce the average error by more than 20%. This is true even when the size of the LUT is as small as  $6 \times 6 \times 6$ . We further observe that addition of constraints on the optimization problem (resulting in symmetric LUTs) still reduces the average



error by over 10% while significantly reducing the size of the problem as described in Section 2.2. In the next experiment we test the optimal LUTs against simple LUTs which do not use any optimized parameters.

#### 4.1.2 Evaluation of Optimal LUT

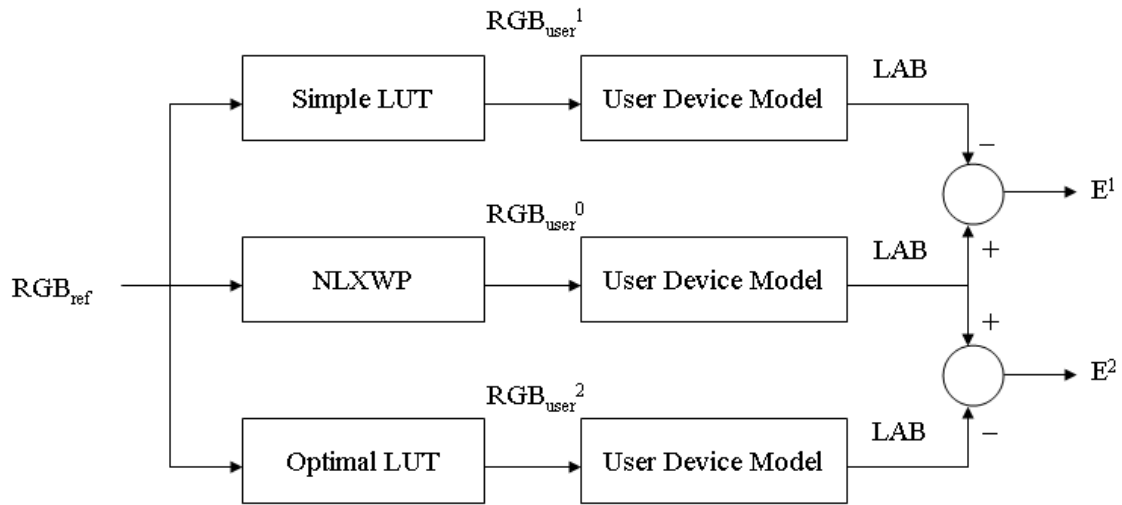


Fig. 4.6. Block diagram for evaluating the optimal and non-optimal LUTs against the NLXWP model.

Figure 4.6 shows our method for evaluating the accuracy of the optimal LUT. Similar to Fig. 2.17, the block arrows represent the offline process of constructing and training the LUTs. The optimal LUTs (which use optimal sampling and separable squared inverse distance interpolation) were tested relative to the NLXWP function using another 1000 point data set, the results are presented in Table 4.3. The statistics correspond to the errors denoted by  $E^2$  in Fig. 4.6. The function was also approximated by non-optimized LUTs (represented as “ $n \times n \times n$  Simple” in the table and producing errors denoted by  $E^1$  in the figure) which used separable inverse distance interpolations on a uniformly sampled  $\Omega$ . While the  $(R, G, B)$  values used

in the testing set were also randomly generated from the RGB space, we verified that the training and testing sets did not have any overlapping data points.

Table 4.3

Testing error statistics ( $\Delta E$  units) for LUTs evaluated relative to the NLXWP model (*Simple* LUTs use separable inverse distance interpolations and a uniform sampling of the RGB space, and numbers inside the parentheses represent percentage improvement as obtained by an optimal LUT).

Type\Statistic	Mean	Median	1-Std. Dev.
$6 \times 6 \times 6$ Simple ( $E^1$ )	8.71	7.31	6.81
$6 \times 6 \times 6$ Optimal ( $E^2$ )	4.84 (44.43%)	4.21	3.47
$9 \times 9 \times 9$ Simple ( $E^1$ )	5.49	4.69	4.72
$9 \times 9 \times 9$ Optimal ( $E^2$ )	2.92 (46.81%)	2.49	2.39
$12 \times 12 \times 12$ Simple ( $E^1$ )	3.96	3.28	3.80
$12 \times 12 \times 12$ Optimal ( $E^2$ )	2.19 (44.70%)	1.85	1.98

It can be observed from the statistics that the optimal LUTs generated by our system offers significant improvement over simple LUTs of the same size. Note that the mean values (column 2) are comparable to the cost function computed for the optimization and, hence, we compute the percentage reduction in this quantity as achieved by an optimal LUT over a simple LUT (provided inside parentheses). It is also evident that the effect of two independently selected methods (optimal sampling and interpolation) is cumulative and the percentage reduction in the average error exceeds the percentage reduction obtained in Table 4.2 where only the sampling method was being optimized. This completes our analysis of how well an optimal LUT can approximate a given function and we proceed to testing our optimal LUTs in a color management scenario.

At this point, we remind the reader that all our experiments were done on two virtual (or simulated) display devices. These devices were, essentially, highly accurate

models of two real display devices (which are shown in Fig. 1.4). Generating the output with these models (for a given input RGB value) is equivalent to measuring the output obtained on the actual displays for the same input.

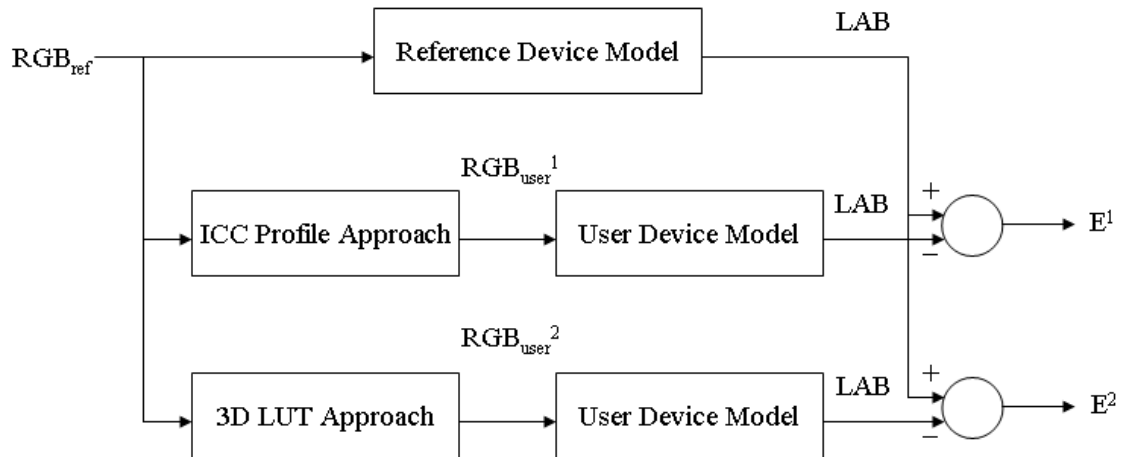


Fig. 4.7. Block diagram for the comparison of a profile-based and a LUT-based system using the output of the simulated reference device as the reference. Here  $E^1$  and  $E^2$  are the errors obtained for the profile-based and the LUT-based approaches, respectively.

#### 4.1.3 Comparison of Optimal LUT With ICC Profiles

In order to compare the LUT-based system with a profile-based system, we require a function  $g : \mathfrak{S} \rightarrow \mathfrak{S}$  which achieves perceptual similarity and is obtained from ICC-compatible device profiles. We used the open source software LPROF [133] to generate monitor profiles. The LPROF software is actively supported and is ICC version 2 compatible. It can generate a device profile based on selected parameters and a measurement sheet. The following steps are involved in obtaining the function  $g$ :

- We use the simulated ideal monitors to obtain the set of measurements required by the profiling software. This consists of computing the simulated CIE XYZ

values obtained when different RGB inputs are used. Our set of RGB inputs contains grayscale, red, green, blue and composite colors. The measurements for the two displays are recorded into corresponding “IT8” files [134]. Note that unlike a scanner calibration operation, we do not read the RGB values for an IT8 target. Instead we generate an IT8 target by passing known RGB inputs to the monitors and recording the output in CIE XYZ. For our experiments, we used 432 RGB values for creating the IT8 files. These consisted of 24 gray, 64 red, 64 green, 64 blue, and 216 mixed colors.

- LPROF uses the IT8 files as input and generates the corresponding “ICM” files. We specify the desired color LUT size and the chromatic adaptation model to be used as parameters while generating the profiles. The color reproduction intent for generating the profiles is specified as “perceptual.”
- The profiles thus generated are processed using MATLAB with the MATLAB “iccread” function, and the color transformation  $g$  is obtained using the MATLAB “makecform” function. Note that the steps involved in creating the transformation  $g$  is similar to the steps involved in generating a device-link profile, using the two device profiles as inputs. Hence, the accuracy of the color reproduction using such a device-link profile will not be significantly different from the accuracy when using the transformation  $g$ . Therefore, we did not perform separate experiments comparing our LUT-based approach with device-link profiles.

As illustrated in Fig. 4.7, we compare the performance of the profile-based transform with that of the LUT-based transform by using the output of the simulated reference device as the benchmark. Note that this is different from the evaluation of the LUT when compared with NLXWP as shown in Fig. 4.6. There the output of the user device as predicted by NLXWP was taken as the benchmark, and hence, the error statistics may be different. Table 4.1.3 summarizes the error statistics achieved by the profile-based transform and the LUT-based transform for various profile and

LUT sizes. The numbers inside parentheses represent the memory required by the two approaches in kilobytes. In the case of profiles it is the size of the respective “ICM” files, while in the case of 3D LUT it is the memory required to store the LUT, and the sampling maps  $M_R$ ,  $M_G$  and  $M_B$  as defined in Equations (2.43), (2.44), and (2.45).

Table 4.4  
Testing error statistics ( $\Delta E$  units) for LUTs and ICC profiles.

Type\Statistic	Mean	Median	1-Std. Dev.	P <sub>95</sub> <sup>1</sup>
ICC-1 <sup>2</sup> (7 KB)	10.12	8.76	5.95	22.76
ICC-2 <sup>3</sup> (66 KB)	10.24	8.91	5.90	22.82
6 × 6 × 6 Optimal (0.65 KB)	9.42	8.88	4.36	15.85
9 × 9 × 9 Optimal (2.14 KB)	8.15	8.02	3.77	12.92
12 × 12 × 12 Optimal (5 KB)	7.86	7.74	3.81	12.61

<sup>1</sup> The 95<sup>th</sup> percentile of the distribution.

<sup>2</sup> Profiles were generated using Linear Bradford [135] chromatic transformation.

<sup>3</sup> Profiles were generated using CIECAM97s [136] color adaptation models with simulated dark surroundings.

We observe that the LUT-based system provides more accurate color reproduction than a system that uses ICC profiles of much larger sizes. Furthermore, this observation is consistent in all the statistics – mean, median, 1-standard deviation and the 95<sup>th</sup> percentile. This agrees with the intuition that a one-step transformation between two known devices can be achieved with greater success than a multi-step transformation in which the devices are only characterized by their profiles. It is also

seen that a larger optimal LUT provides better color reproduction accuracy than a smaller LUT. This is also not evident for profiles. Our aim in these experiments is to demonstrate the improved color reproduction offered by a LUT-based approach compared with profile-based approaches.

## 4.2 Surveillance of Vehicles

We applied the methods described in the previous sections to actual traffic videos in our experiments. There are two important points about these experiments. First, our primary goal in this thesis is to demonstrate the feasibility of our system using off-the-shelf equipment and easily obtained video data. While a thorough testing of the system would require running the system in real time using live video feed, such experimentation is not reported in this thesis. The inter-dependencies described in Section 3.1 are realized by manually channeling the outputs of different modules. Secondly, we conducted many of our experiments using publicly available datasets. However, due to the unique camera configuration proposed in this article, all the methods cannot be tested on these datasets. Therefore, we also present the results of applying our methods on a 20 minute front and side view video collected for the purpose.

### 4.2.1 Vehicle detection

Our method for background subtraction (MABS) was applied to videos containing vehicles against an uncontrolled outdoor background. Four such videos were used to test the method – (A) the Lankershim Boulevard dataset [137] (camera 3, footage starting at 8.45), (B) the Peachtree Street dataset [137] (camera 4, footage starting at 4.00), and (C and D) traffic videos recorded by the authors. The C and D videos were taken from a much lower height (about six feet above the road) and under different weather conditions. Thus, unlike the first two datasets, the size of vehicle “blobs” changes with its position in the camera’s field of view in these videos.



Fig. 4.8. An example of vehicles detected by background subtraction. The image shows a close-up of the scene to illustrate the object ellipses.

A detection was defined as a successful hand-off from object detection method to the particle filter-based tracker. In other words, we count a true positive when a new “target” is created in the tracker following its entry into the field of view. This is depicted by the red ellipses in the Figure 4.8. Missed detections and false positives have corresponding definitions. However, note that we do not focus on the results of the tracker in this thesis. Instead, the significant visual output from the object detection stage is the foreground-background mask. Some representative frames from the different datasets and their foreground masks are provided in Figure 4.9.

During the nearly fifteen (combined) minutes of analysis, about one thousand vehicles appeared in the video. Less than 5 percent of detection failures occurred. Most of the failures were missed detections when the vehicles were highly similar to the background (the road). Failures were also encountered when the system counted two vehicles moving very close to each other as a single object.

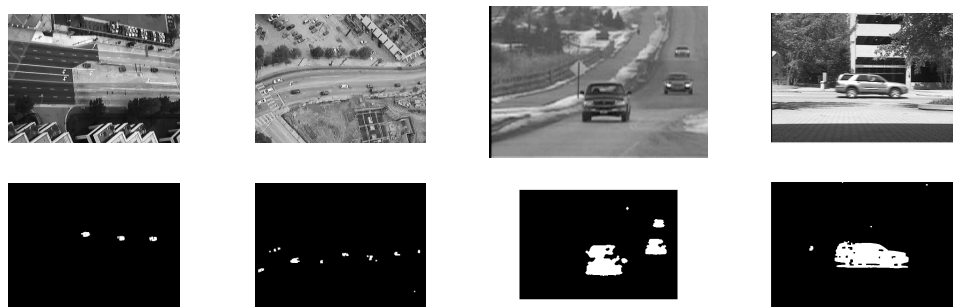


Fig. 4.9. Output of object detection applied to different video sets (A, B, C, and D). Column D shows a result of vehicle detection on the side view video.



Fig. 4.10. Comparison of MABS (bottom) with GMM (center) for object detection on three test sequences. Columns 1 and 2 show processed frames from sequences with considerable background clutter while column 3 shows a frame shortly after a sudden change of illumination.

In order to estimate the time complexity of the method, we created five video sequences with dynamic background, large foreground objects, and sudden illumination changes. Owing to the difficulty of creating such test cases with vehicles, we used human subjects for this experiments. The proposed method for background subtraction was used on these sequences. We also used a background subtraction method based on Gaussian mixture models [86] with each pixel represented by a mixture of three



Gaussian distributions. The parameters were selected such that both methods provided comparable outputs (visually determined). Figure 4.10 shows some examples from the video sequences and the outputs of the two methods. The execution time of the two methods are listed in Table 4.5. It can be observed that the proposed method is much faster than the GMM method, and hence, more suitable for our application.

Table 4.5

Running time for the two object detection methods (in seconds). The experiments were conducted in MATLAB running on a Mac OS-X machine.

Sequence	Frames	GMM	MABS
Seq 1	100	1020	102
Seq 2	100	994	104
Seq 3	180	1910	225
Seq 4	270	2150	224
Seq 5	210	2092	193

#### 4.2.2 Vehicle type determination

The template-based vehicle body type determination was tested using vehicle side view (still) images. Examples of the results are shown in Figure 4.11. Note that the  $y$ -axis in the graphs represents the normalized absolute difference. Thus, the class with the smallest difference is chosen.

Since most vehicle datasets do not contain front and side view videos we generated a testing dataset with about 20 minutes of traffic video footage. The camera was placed about one foot above the road surface and positioned such that the field of view was large enough to contain 3 – 4 typical length vehicles. Some frames from the dataset can be shown in the top row of Figure 4.12. Most vehicles were moving between 30 and 45 miles per hour. As stated earlier, we restrict the vehicle type to

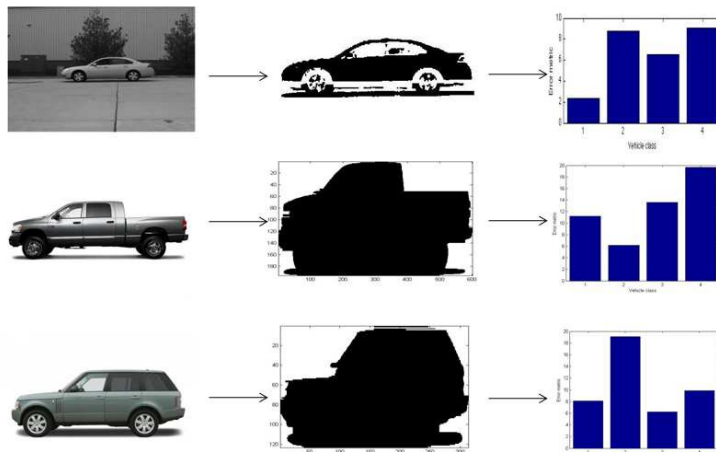


Fig. 4.11. Examples of vehicle type identification from still side-view images. The middle column shows the silhouettes obtained from vehicle detection. The graphs represent the SAD values ( $y$ -axis) for each vehicle type ( $x$ -axis).

one of the four classes – sedan, light truck, sport/utility, and hatchback. Hence, the frames used in this experiment were manually selected.

The method was applied to 85 vehicles roughly equally distributed over the four classes. While all the sedans and trucks were correctly identified, the SUV and hatchback classes were not so well separated. Nine vehicles were assigned to the wrong class, and many other (correct) classifications were achieved with a small margin. Figure 4.12 shows examples of classification with a good margin, another with a small margin, and a failure case. It also be noted that the proximity of the SUV and hatchback classes is expected since both types have similar shapes. These classes can be easily separated by using a simple threshold on the vehicle length because most utility vehicles are bigger than hatchbacks. However, this has not been incorporated in the present results.

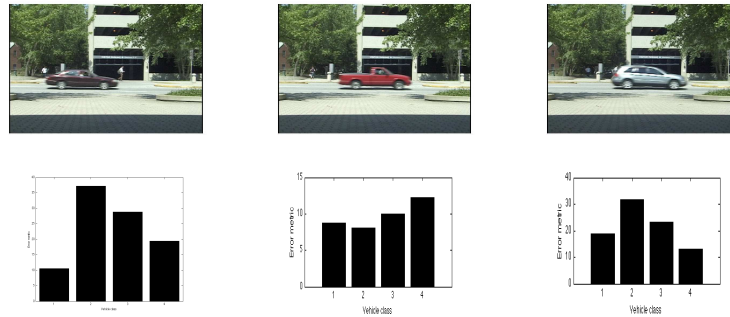


Fig. 4.12. Examples of vehicle type determination using shape matching. The three columns represent (left to right) cases of classification with high confidence, classification with low confidence, and a misclassification.

### 4.2.3 Tire size estimation

As in the case of body type determination, we first tested the tire extraction methods on side view still images. Examples of the results are shown in Figure 4.13.

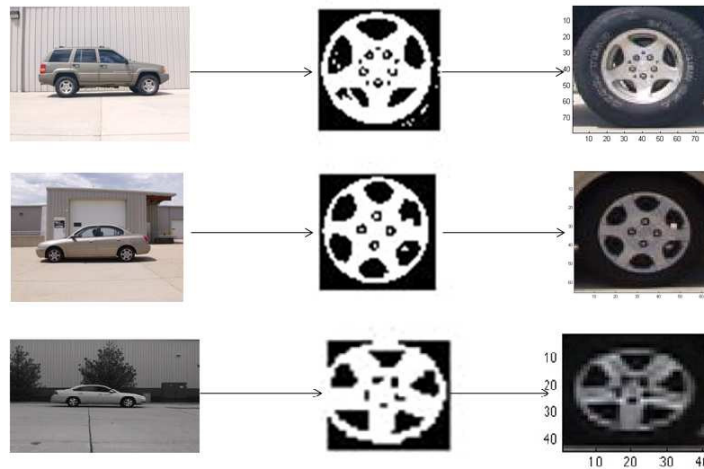


Fig. 4.13. Examples of vehicle tires extracted from still side-view images. The middle column shows intermediate results in the form of wheel positions.

Since the methods for tire extraction require a side view of the vehicles, the above-mentioned dataset (containing 20 minutes of footage) was used in these experiments. The ground truth regarding the tire sizes (measured in pixels) was obtained manually. Therefore, these methods were applied on manually selected frames and on a smaller testing set with 24 vehicles. In all but one cases, the tire diameter was correctly estimated to within one pixel tolerance. The failure case reported an incorrect position for the tire center and hence, the size estimate was irrelevant. Figure 4.14 shows some examples of the vehicles and the tire region as extracted by the system. The failure case is also shown.

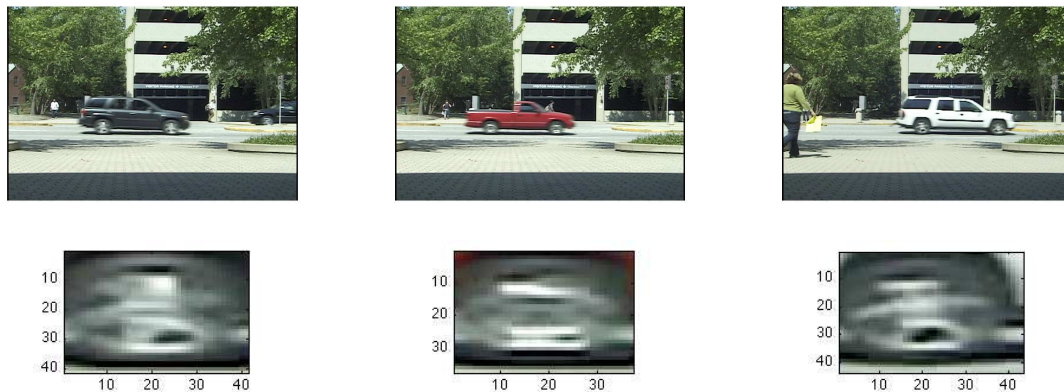


Fig. 4.14. Examples of vehicle tire extraction using the circular edge search method. In each case, the rear tire was segmented. Column 3 shows the failure case.

Note that all 24 vehicles in these experiments had a bright hub compared to the rubber. In the absence of a suitable test case, the method for estimating the size of a tire with dark hub was applied to an synthetic test case. This was done by manually coloring the tire area of a vehicle's image and is illustrated in Figure 4.15. The position of the lower-most point of the tire was provided as input to the system (corresponding to the position of the cleat as explained in Section 3.2.3). The diameter of the tire was correctly estimated. We would like to emphasize that this particular experiment (with dark tires) was only conducted as a proof of concept and does not capture rigorous evaluation of the method. On the contrary, the absence of vehicles

with such tires in the 20 minutes of traffic footage itself demonstrates the rarity of such a situation.



Fig. 4.15. Result of tire extraction on a vehicle with artificially colored dark tires. The position of the lowermost point of the tire was provided as input.

#### 4.2.4 Make recognition

Many experiments on vehicle make and model recognition are carried out using “clean” images of vehicles. Such images have high resolution and use vehicles which are stationary or near-stationary (for example, near the entrance of a parking lot or at a toll booth). Instead, we apply our methods on video frames taken in outdoor conditions and on vehicles moving as fast as 40 miles per hour. In fact, the traffic video used in this experiment was recorded at the same time and location as the video used in the previous two experiments. For some vehicles, the front grille area used to determine the make was as small as  $30 \times 15$  pixels. Some sample frames from the dataset are shown in the top row of Figure 4.16.

Vehicles from five car makers were used to test our methods. These five makes were determined by viewing the video and identifying the most frequently occurring makes. Then a total of 65 vehicles were selected as the testing set. Of these, the system correctly recognized 56 makes. As in the case of vehicle type experiment, some decisions had a large margin (or confidence) while others (both correct and incorrect) were close calls. Figure 4.16 shows some examples of the make recognition

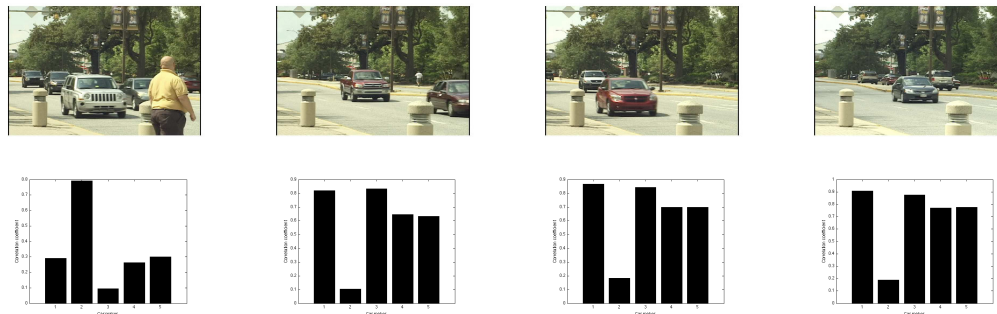


Fig. 4.16. Examples of vehicle make recognition using edge orientation histograms and automatic region selection. Column 4 shows a failure case.

output. These include both the well-resolved and the marginal cases. Note that the  $y$ -axis represents a normalized similarity metric. Thus, the class with the highest value is selected. It is evident that the classification accuracy is lower than some of the state-of-the-art make recognition methods. Even the edge orientation method of Petrovic and Cootes [98] results in about a 90 percent accuracy. This can be explained by two factors – the use of more realistic images and the automatic selection of the front region of interest. While these two factors reduce the accuracy, our experiments prove that the method is robust enough to be deployed in real-life scenarios.

#### 4.2.5 Squat and Bounce Analysis

The purpose of this particular experiment was to determine the feasibility of the proposed methods for detecting overloaded vehicles, rather than rigorous testing of the methods. The vehicles were loaded by placing 500 pounds of sand bags in the trunk. We photographed the lowering of the vehicle due to loading, as well as drove the vehicle over a bump to observe the pattern of oscillations. A checkerboard shaped fiducial marker was attached to each test vehicle to allow easier tracking.

Figure 4.17 shows 3-tuples of images corresponding to the test vehicles under unloaded and loaded conditions, and the difference image. It should be noted that

Table 4.6  
The gap above tires measured in pixels for the test vehicles.

	Unloaded		Loaded	
Vehicle	Front	Rear	Front	Rear
Vehicle 1	56	57	55	49
Vehicle 2	53	53	54	48
Vehicle 3	70	72	71	67

these images has been significantly downsized. It can be observed that for all the three test vehicles, the vehicle depresses by a visually significant amount in the rear while the front end is unaffected. In a standard resolution video frame, the lowering was between 5 and 8 pixels. The gap above the front and rear tires under loaded and unloaded conditions is listed in Table 4.6.

One of the test vehicles shown in Figure 4.17 was driven over a road bump and the vertical motion of the vehicle was recorded. This was achieved by tracking the position of the fiducial marker. Figure 4.18 shows the oscillation trajectory in which the vertical position is plotted against the horizontal position. The oscillations are plotted with the test vehicle with and without the payload of sand bags represented by the solid and the dashed plots, respectively. The vehicle traverses the bump around the horizontal position of 230. We observe that (a) the loaded vehicle always sits lower than the unloaded case, (b) the depression immediately after traversing the bump is much steeper when the vehicle is loaded, and (c) the loaded vehicle takes longer to return to the stable state. All three observations agree with the intuition and the physics of damped oscillation.



Fig. 4.17. Illustration of the visual impact of loading on test vehicles. The difference image (column 3) captures the displacement caused by loading. Columns 1 and 2 show the unloaded and loaded vehicles.

#### 4.2.6 Trajectory Analysis

The results are provided under three categories – evaluation of the co-ordinate transformation technique, evaluation of the velocity analysis method, and the evaluation of turn identification using shape analysis. Three video sequences were used in our experiments. These included the Lankershim Boulevard, California surveillance dataset generated for the NGSIM project [137] (denoted by LANK in this section) and two videos recorded by the authors (denoted by ANON1 and ANON2). A snapshot of the video sets is provided in Figure 4.19. In order to illustrate the role of the hypothetical co-ordinate system, we also provide a satellite image of the road in the field of view, using Google Maps [138]. Due to the nature of our outputs, the figures in this section are best viewed in color. For the datasets collected by the authors, the labels have been erased for this anonymous review draft.



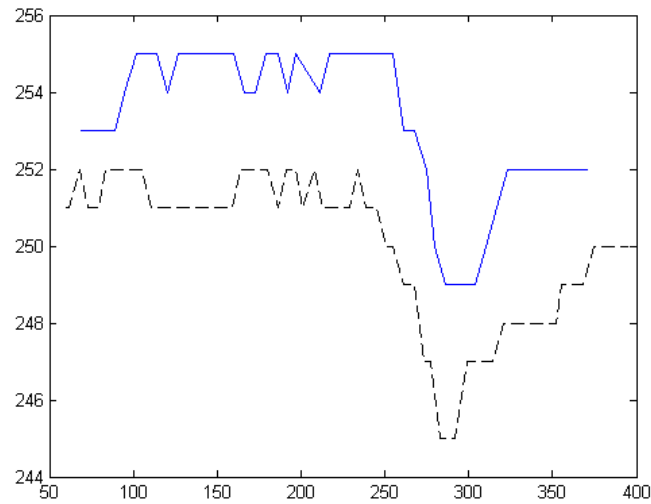


Fig. 4.18. A graphical representation of the vehicle’s bounce upon excitation by a road bump. The oscillations are larger and decay slower under loaded conditions.

We present two cases in which the proposed look-up table based method has been used on curved trajectories. These are shown in Figure 4.20. The first case is a synthetic trajectory created for the curved road used to illustrate the training process in Section 3.2.7. The second case is a trajectory from the ANON2 dataset. The linearized trajectories (in the hypothetical co-ordinate system) are shown in the second row. Note that in the second case, a single 2D LUT is used to transform from the image co-ordinates to the desired co-ordinates. The perturbation seen in the upper part of the transformed trajectory (in case 2) is due to the sensitivity of the transformation very deep in the field of view. This error can be improved by better user input or with alternative camera calibration methods although the non-linearity will always cause greater errors in the far field of view.



Fig. 4.19. Sample images from our test videos (a-c) and the corresponding map (d-f) (courtesy: Google Maps [138]). The field of view is highlighted in yellow and the labels have been erased for review.

#### 4.2.7 Velocity Analysis

Our method for analyzing the approach velocity of a vehicle was used with the ANON1 dataset. Different scenarios were realized by driving a designated vehicle in various manners. In Figure 4.21, we present the decision vectors corresponding to four cases – driving at nearly uniform velocity, unexpected slowing, unexpected sudden acceleration, and making a u-turn. Recall that we treat velocity as a one-dimensional signal (in the direction of the road) as a function of the position. Thus, the vertical axis represents the scaled velocity at different points in the roadway. The blue circles indicate normal approach while the red crosses denote (atomic) anomalies. The dashed black line represents the mean (scaled) velocity estimated from the “normal” vehicles. The anomaly flags plotted as functions of time are also provided in Figure 4.22 as described in [107]. It is clear to see that this treatment and the resultant

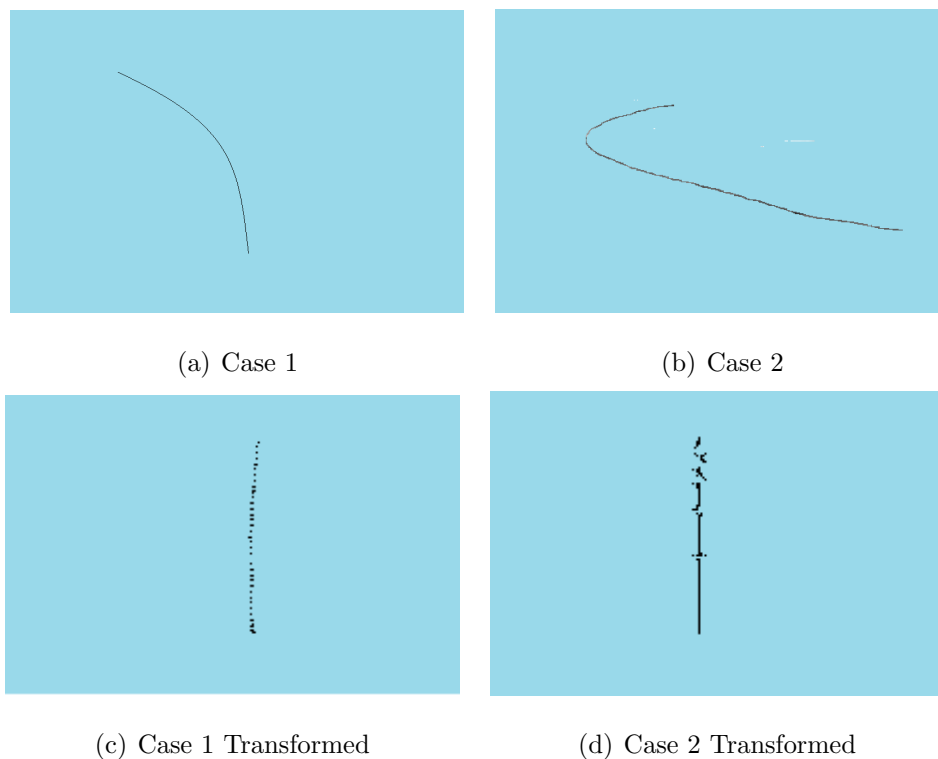


Fig. 4.20. Results of the LUT-based co-ordinate transformation to curved trajectories. Case 1 pair is a synthetic test case while the Case 2 pair is from the dataset ANON2.

method for visualization of the decisions would enable very fast decision-making by an operator.

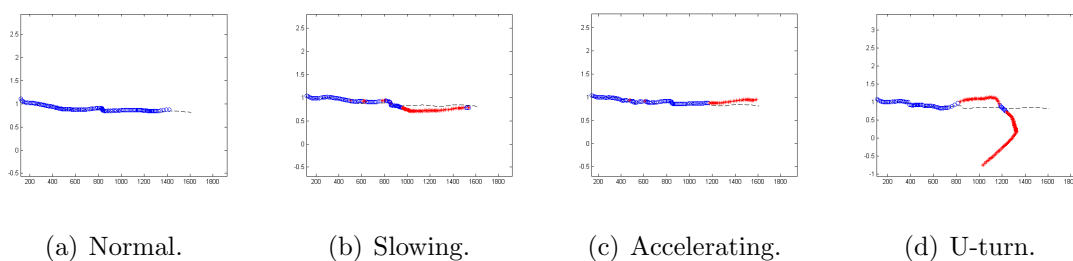


Fig. 4.21. Plot of the decision vectors corresponding to different driving behaviors. The red crosses indicate occurrence of atomic anomalies.

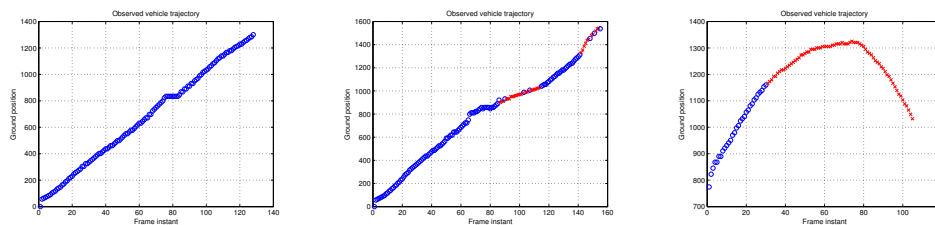


Fig. 4.22. Plot of the decision vectors corresponding to a vehicle being driven at uniform velocity, slowing and accelerating, and making a u-turn. The red crosses indicate occurrence of atomic anomalies.

#### 4.2.8 Shape Analysis

Shape analysis for detecting turns was used on both synthetic and true trajectories. Figure 4.23 shows the examples of template matching output on four types of maneuvers. These trajectories were obtained by tracking individual vehicles in the LANK dataset [137]. The black circle indicates the starting end of the trajectory (since there is no temporal information in these plots). Right turns are denoted by red circles and left turns by blue circles. The color of a u-turn marker is chosen based on the turn being left-handed or right-handed. A total of 46 vehicle trajectories were analyzed and 45 maneuvers were correctly identified. These cases included vehicles from East-West and North-South traffic and roughly equal number of right, left, and no turns. The u-turn case in Figure 4.23 was the only such maneuver in the dataset. The failure case occurred when a vehicle made a turn after prolonged delay and our shape matching method did not detect the turn.

A synthetic trajectory was generated to simulate complex multi-turn scenarios to test the effectiveness of our methods. Figure 4.24 shows the result of turn detection on the synthetic trajectory. It can be seen that our methods not only detect all the turns but also declare the occurrence of turns at a logically correct point in the trajectory.

Finally, we demonstrate the usefulness of the hypothetical co-ordinate system by using shape matching for the two test cases shown in Figure 4.20. The window method

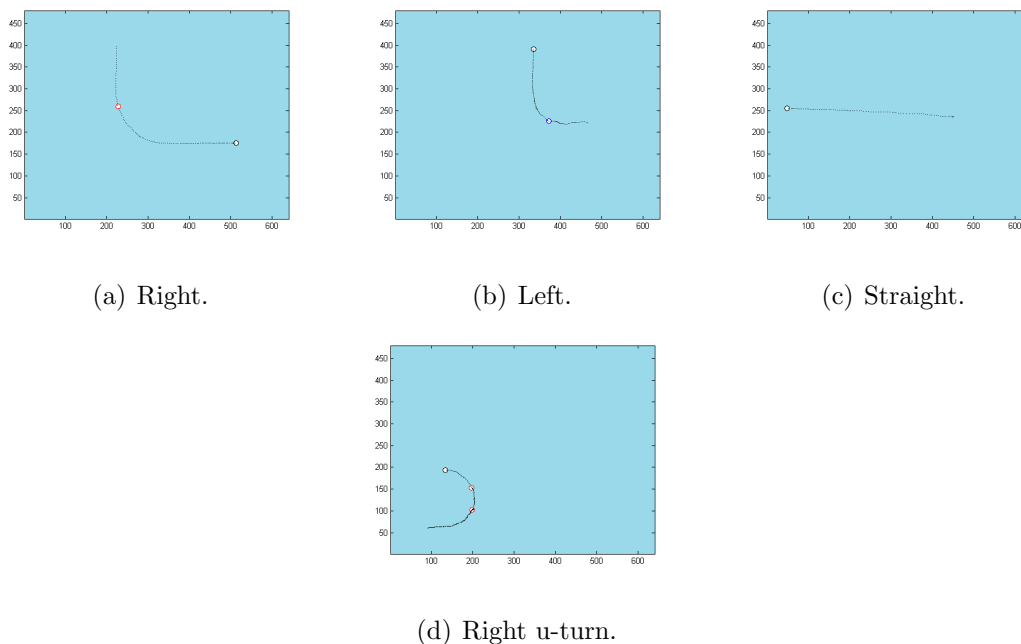


Fig. 4.23. Results of shape analysis on trajectories of different maneuvers from the LANK dataset. A black circle is used to denote the starting end of the trajectory.

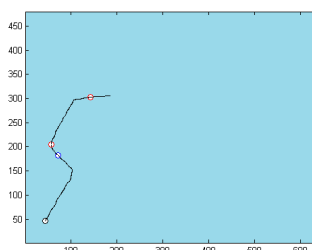


Fig. 4.24. Result of shape analysis on a synthetic multi-turn trajectory.

was used to detect turns in the original (ground) and the transformed trajectories (in the hypothetical co-ordinates). The results are shown in Figure 4.25. Clearly, the linearization enforced by the co-ordinate transformation is effective in suppressing the false turn detections.

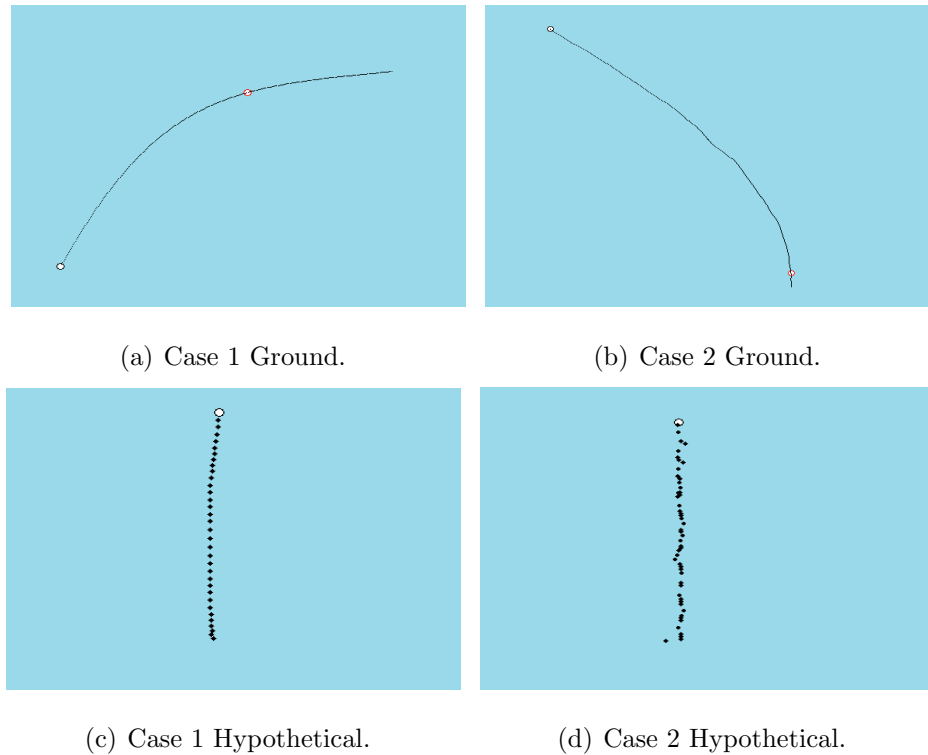


Fig. 4.25. Window method for turn detection applied to curved roads. The rows represent output of turn detection in ground (a-b) and hypothetical (c-d) co-ordinates. Note that false turns are detected only in the ground co-ordinates.

#### 4.2.9 Color Correction for Tracking

The methods developed above were tested for a system with two imaging units. We used a single video camera (Sony DCR-TRV33) to image objects under two different illumination conditions. The controlled illumination was achieved using a Gretag MacBeth light booth and the two illuminants were selected to approximately resemble the inside and outside lighting of a typical parking garage.

Since our objective is to make the corrected colors  $(\tilde{R}, \tilde{g}, \tilde{B})$  closer to the reference colors  $(R, G, B)$  in the RGB space, we use a Euclidean metric of distance given by:

$$\Delta = \|(\tilde{R}, \tilde{g}, \tilde{B}) - (R, G, B)\|. \quad (4.1)$$

Table 4.7 shows the mean error ( $\Delta$ ) values for different objects and corrected using the proposed techniques. We also corrected the colors using a constant offset in every channel as proposed by Choi [126].

Table 4.7  
Mean errors ( $\Delta$ ) between the reference image and the corrected images.

<b>Test\Method</b>	<b>None</b>	<b>Offset</b>	<b>LUT</b>	<b>CCMX</b>
Golf	22.86	23.65	16.58	10.46
DVD	27.57	27.72	19.47	15.54
Phone	37.05	20.01	18.02	10.17
Patch	15.00	25.83	9.67	5.01

In order to test the effect on color tracking, we estimate the improvement in the observation likelihood model due to color correction. Recall that the suitability of a particle is measured using the Bhattacharyya distance. Thus, we compute the Bhattacharyya distance of the object under the reference and the test conditions with different correction methods. The color features are extracted only in a small elliptical region around the object as shown in Figure 4.26. The distances for different test objects are presented in Table 4.8. As stated in Section 3.2.7 a lower distance would result in the particle being assigned a higher weight. In these experiments, the particle actually contains the target. Therefore, a higher weight (and lower distance) implies improved performance of the tracker.

There are three important observations in these results. It is evident that the color features become more robust after correction with our proposed methods. Secondly, the offset method [126] can sometimes result in more error than the uncorrected image. Finally, none of the methods appear to work when correcting a constant color patch with our chosen feature ( $8 \times 8 \times 8$  histogram). This is not entirely unexpected because of the high granularity of our histogram bins. The corrected color may fall into a different bin even when its Euclidean distance from the reference is only 5

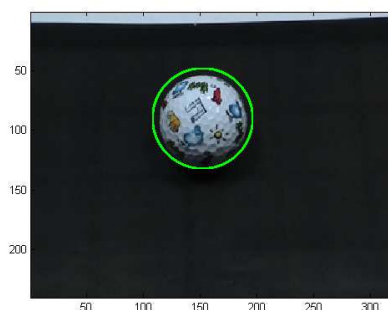


Fig. 4.26. An illustration of the target region used to compute the color histogram.

Table 4.8

Bhattacharyya distances of the candidate object in the reference and corrected images.

Test\Method	None	Offset	LUT	CCMX
Golf	0.81	0.67	0.35	0.19
DVD	0.56	0.82	0.29	0.22
Phone	0.58	0.86	0.39	0.32
Patch	1.00	1.00	1.00	1.00

units. Since the histogram in this case will be an impulse function, the Bhattacharyya coefficient would be zero. Such degeneracy can be avoided by “soft” assignment to histogram bins but this has not been used in this thesis.

The resultant images for two of the test cases are shown in Figure 4.27. Due to the nature of the results, these figures are better viewed in color. From these results we can conclude that the proposed methods offer improved color correction in both quantitative and qualitative tests. The LUT-based method slightly increases the error as compared to the CCMX method but is considerably more accurate than the offset method.



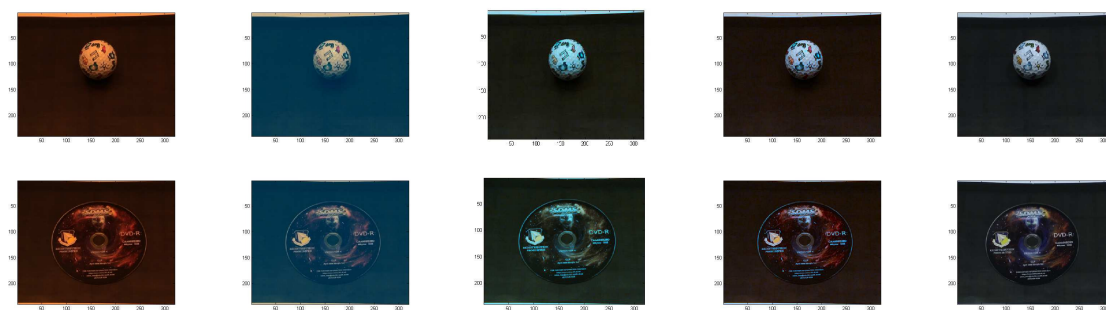


Fig. 4.27. Qualitative results using different color correction techniques. The columns represent (from left to right) (i) the test image, (ii) output of the offset method, (iii) output of the 3D LUT method, (iv) output of the CCMX method, and (v) the reference image.

## 5. CONCLUSIONS AND FUTURE WORK

In this thesis two topics of active research in image processing are described. Color management has become increasingly important because of the large quantities of visual data and the variety of image capture/viewing devices. Visual surveillance is ubiquitous and image analysis tools are being developed to make these surveillance systems more automated and efficient.

### 5.1 Conclusions

It is important to ensure faithful reproduction of visual data when a digital image/video is viewed on a user's media. This is particularly valuable for professional photography, cinema and television programming, and printing. In this thesis, we propose a novel approach to color management using device models and look-up tables. The main contributions are as follows:

- We studied the traditional color profile based approach to color management and identified the limitations which render it less suitable for the newer requirements in color management. We proposed a simpler color management system which would use only look-up tables which are created offline. Such a system would be more hardware friendly and hence, more suitable for viewing content on low power devices.
- We constructed a transformation based on device models for perceptually matching two display devices.
- We approximated the above transformation using 3D LUT. We further identified methods to optimally select LUT parameters (such as sample points) in a model-based optimization framework under specified resource constraints.

- We compared the optimal LUT based color management system with an ICC profile based system in terms of color reproduction accuracy and memory requirements. It was shown that the proposed system was more accurate and memory efficient.

Surveillance systems are deployed in many countries to assist law enforcement and public safety. Image and video analysis techniques are being designed to extract information from the videos. Such automated processing would help the operators and reduce human errors. In this thesis, we describe a visual surveillance system for monitoring vehicles and detecting potential anomalies. The main contributions are as follows:

- We proposed a surveillance system with two orthogonal cameras which would image approaching vehicles from the front and the side without obstructing the traffic.
- We developed low complexity methods to extract physical information about each vehicle (such as body type and make). These would enable the system to estimate the normal ranges of dynamic measurements and help an operator identify the vehicle.
- We proposed methods for detecting anomalous behavior based on dynamic trajectory measurements (such as velocity). Toward this goal, we devised a coordinate system which compensates for the road curvatures. We developed methods to detect unexpected changes in the velocity and to identify significant maneuvers (e.g. left turns and u-turns).
- We proposed a color correction technique which would make the color features more reliable across multiple cameras under different illuminations. This technique was shown to improve the robustness of a color based object tracker.

## 5.2 Future Work

Our look-up table based color management system can be extended in the following directions:

- The model based transformation between the display device is constructed for chosen device settings. If these settings are altered (e.g. brightness and contrast) the models and the LUT derived from them will become inaccurate. Since rebuilding the models for new settings would require computation and user effort, we would like to find ways to directly “tweak” the LUT to adjust for the new settings.
- In practical applications, there may be more than one reference displays that the user’s device should be able to match. This has two implications – (i) the digital content should have embedded information about the desired reference display and (ii) the user’s device should be able to select the appropriate LUT from a collection of LUTs.

Our proposed visual surveillance system can be extended as follows:

- More types of information can be extracted from the video by image analysis techniques. In particular, we would like to develop tools to observe and understand the behavior of vehicle occupants.
- The methods described to accomplish different tasks should be robust to changing conditions (traffic, weather, and lightning). Developing such methods would also involve potential use of external illumination including near infrared imaging for night operation.
- The tools for behavioral analysis of vehicles can be extended to detect group activities rather than just individual vehicles. This would allow detection of both synchronized activities and occurrence of events that affect multiple vehicles.

### 5.3 Publications Resulting from this Work

#### Journal Articles:

1. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, “Color Management Using Optimal Three Dimensional Look-Up Tables,” *Journal of Imaging Science and Technology*, vol. 54, no. 3, May-June 2010, pp. 030402 (1-14).
2. **Satyam Srivastava** and Edward J. Delp, “Autonomous Visual Surveillance of Vehicles for the Detection of Anomalies,” *IEEE Transactions on Intelligent Transport Systems*, submitted.

#### Conference Papers:

1. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, “Co-Ordinate Mapping and Analysis of Vehicle Trajectory for Anomaly Detection,” *Proceedings of the IEEE International Conference on Multimedia and Expo*, Barcelona, Spain, July 2011 (to appear).
2. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, “Color Correction for Object Tracking Across Multiple Cameras,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, May 2011 (to appear).
3. **Satyam Srivastava** and Edward J. Delp, “Standoff Video Analysis for the Detection of Security Anomalies in Vehicles,” *Proceedings of the IEEE Applied Imagery Pattern Recognition Workshop*, Washington DC, October 2010.
4. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, “Generating Optimal Look-Up Tables to Achieve Complex Color Space Transformations,” *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 1641-1644.

5. Thanh H. Ha, **Satyam Srivastava**, Edward J. Delp, and Jan P. Allebach, "Model Based Methods for Developing Color Transformation Between Two Display Devices," *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 3793-3796.
6. Thanh H. Ha, **Satyam Srivastava**, Edward J. Delp, and Jan P. Allebach, "Monitor Characterization Model Using Multiple Non-Square Matrices for Better Accuracy," *Proceedings of the IS&T Color Imaging Conference*, Albuquerque, New Mexico, USA, November 2009, pp. 117-122.
7. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, "Color Management Using Device Models and Look-Up Tables," *Proceedings of the Gjøvik Color Imaging Symposium*, Gjøvik, Norway, June 2009, pp. 54-61.
8. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, "Crowd Flow Estimation Using Multiple Visual Features for Scenes with Changing Crowd Densities," *Proceedings of the IEEE International Conference on Advanced Video and Signal based Surveillance*, Klagenfurt, Austria, 2011, submitted.

## LIST OF REFERENCES

## LIST OF REFERENCES

- [1] C. Poynton, *Digital Video and HDTV: Algorithms and Interfaces*. San Francisco, California: Morgan Kaufmann, 2003.
- [2] G. Sharma, Ed., *Digital Color Imaging Handbook*. Boca Raton, Florida: CRC Press, 2002.
- [3] CIE, *Commission Internationale de l'Eclairage Proceedings*. Cambridge, Massachusetts: Cambridge University Press, 1931.
- [4] ITU - Radiocommunication Sector, "Parameter values for the HDTV standards for production and international programme exchange," Recommendation BT.709-5, Geneva, Switzerland, 2008.
- [5] Digital Cinema Initiative, LLC, "Digital cinema system specification," Adopted and released document, version 1.2, Hollywood, California, 2008.
- [6] S. Upton, "Vista's new color management system: WCS," <http://www2.chromix.com/ColorNews/index.cxsa>, Accessed January 2009.
- [7] International Organization for Standardization, "Image technology colour management - architecture, profile format and data structure - part 1: Based on ICC.1:2004-10," ISO 15076-1:2005, Geneva, Switzerland, 2005.
- [8] N. Koren, "Color management and color science: Introduction," [http://www.normankoren.com/color/\\_management.html](http://www.normankoren.com/color/_management.html), Accessed January 2009.
- [9] International Organization for Standardization, "Portable network graphics (PNG) specification," ISO/IEC 15948:2003, Geneva, Switzerland, 2003.
- [10] Adobe Systems Inc., "Encapsulated PostScript - file format specification," Tech note 5002, San Jose, CA, 1992.
- [11] D. Richardson, "Firefox 3: Color profile support," <http://www.dria.org/wordpress/archives/2008/04/29/633/>, Accessed January 2009.
- [12] M. D. Fairchild, "A color scientist looks at video," *Proceedings of the International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, Scottsdale, Arizona, January 2007.
- [13] S. Shaw, "Digital intermediate: A real world guide to the DI process," <http://www.digitalpraxis.net/zippdf/di-guide.pdf>, Accessed January 2009.
- [14] Colour Science AG, "Test images for monitor and printer calibration," <http://www.colour-science.com/>, Accessed October 2008.



- [15] C. S. McCamy, H. Marcus, and J. G. Davidson, "A color-rendition chart," *Journal of Applied Photographic Engineering*, vol. 2, no. 3, 1976.
- [16] S. A. Henley and M. D. Fairchild, "Quantifying mixed adaptation in cross-media color reproduction," *Proceedings of the IS&T/SID Color Imaging Conference*, Scottsdale, Arizona, November 2000, pp. 305–310.
- [17] N. Katoh *et al.*, *Chromatic Adaptation under Mixed Illumination Condition when Comparing Softcopy and Hardcopy Images*. Vienna, Austria: Technical Committee 8 of the CIE, 2004.
- [18] Adobe Systems, Inc., "Matching RGB color from monitor to printer," Tech note 5122, San Jose, California, 1992.
- [19] B. Robertson, "The colorists," [http://features.cgsociety.org/story\\_custom.php?story\\_id=3549](http://features.cgsociety.org/story_custom.php?story_id=3549), Accessed January 2009.
- [20] The Internet Movie Database, "Touchstone pictures (US)," <http://www.imdb.com/company/co0049348/>, Accessed January 2009.
- [21] G. Gill, "What's wrong with the ICC profile format anyway?" [http://www.argyllcms.com/icc\\\_problems.html](http://www.argyllcms.com/icc\_problems.html), Accessed February 2009.
- [22] P. Green, "New developments in ICC colour management," *Proceedings of Digital Futures: International Standards around Printing and Imaging*, London, UK, October 2008.
- [23] C. Starr, "An introduction to color management," [http://aaaproduct.gsfc.nasa.gov/teas/CindyStarr\\_GST/ColorMgmt5examples/ColorMgmt5examples.PPT](http://aaaproduct.gsfc.nasa.gov/teas/CindyStarr_GST/ColorMgmt5examples/ColorMgmt5examples.PPT), Accessed February 2009.
- [24] R. W. G. Hunt, *The Reproduction of Color*. Hoboken, New Jersey: Wiley, 2004.
- [25] International Color Consortium, "File formats for color profiles," ICC.1:2001-04, Reston, Virginia, 2001.
- [26] J. V. Kries, *Chromatic Adaptation*. Fribourg, Switzerland: Festschrift der Albrecht-Ludwig-Universität, 1902.
- [27] International Color Consortium, "The role of ICC profiles in a colour reproduction system," ICC White Paper: [http://www.color.org/ICC\\\_white\\\_paper\\\_7\\\_role\\\_of\\\_ICC\\\_profiles.pdf](http://www.color.org/ICC\_white\_paper\_7\_role\_of\_ICC\_profiles.pdf), December 2004.
- [28] Heidelberger Druckmaschinen AG, "Generation and application of device link profiles," Prinect Color Solutions: [http://www.heidelber.com/www/html/en/binaries/files/prinect/device\\\_link\\\_profile\\\_pdf](http://www.heidelber.com/www/html/en/binaries/files/prinect/device\_link\_profile\_pdf), Accessed February 2010.
- [29] Microsoft Corporation, "Using device profiles with WCS," <http://msdn.microsoft.com/en-us/library/dd372213\%28VS.85\%29.aspx>, Accessed February 2010.
- [30] M. Maria, "Little CMS engine," <http://www.littlecms.com/TUTORIAL.TXT>, Accessed February 2010.

- [31] M. R. Balonen-Rosen and J. E. Thornton, "User-interactive corrective tuning of color profiles," United States Patent 6,307,961, 2001.
- [32] M. D. Fairchild, *Color Appearance Models*. Chichester, UK: Wiley, 2005.
- [33] G. J. D. Smith, "Behind the screens: Examining constructions of deviance and informal practices among CCTV control room operators in the UK," *Surveillance and Society*, vol. 2, no. 2-3, pp. 376–395, 2004.
- [34] J. Tullio *et al.*, "Experience, adjustment, and engagement: The role of video in law enforcement," *Proceedings of the ACM Conference on Human Factors in Computing Systems*, Atlanta, Georgia, April 2010, pp. 1505–1514.
- [35] E. Wallace and C. Diffley, "CCTV control room ergonomics," Technical Report 14/98, Police Scientific Development Branch, UK Home Office, 1998.
- [36] N. Haering, P. L. Venetianer, and A. Lipton, "The evolution of video surveillance: an overview," *Machine Vision and Applications*, vol. 19, no. 5-6, pp. 279–290, September 2008.
- [37] P. Meer, *Emerging Topics in Computer Vision*. Englewood Cliffs, New Jersey: Prentice Hall, 2004.
- [38] H. M. Dee and S. A. Velastin, "How close are we to solving the problem of automated visual surveillance?" *Machine Vision and Applications*, vol. 19, no. 5-6, pp. 329–343, September 2008.
- [39] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research: Part C*, vol. 6, no. 4, pp. 271–288, August 1998.
- [40] R. T. Collins *et al.*, "A system for visual surveillance and monitoring," Final Report CMU-RI-TR-00-12, Carnegie Mellon University, 2000.
- [41] J. Aguilera *et al.*, "Visual surveillance for airport monitoring applications," *Proceedings of the Computer Vision Winter Workshop*, Telc, Czech Republic, February 2006.
- [42] X. Li and F. M. Porikli, "A hidden markov model framework for traffic event detection using video features," *Proceedings of the IEEE International Conference on Image Processing*, vol. 5, Singapore, October 2004, pp. 2901–2904.
- [43] Y. Tian *et al.*, "IBM smart surveillance system (s3): event based video surveillance system with an open and extensible framework," *Machine Vision and Applications*, vol. 19, no. 5-6, pp. 315–327, September 2008.
- [44] D. Gutchess *et al.*, "Video surveillance of pedestrians and vehicles," *Proceedings of the SPIE: Acquisition, Tracking, Pointing, and Laser Systems Technologies*, vol. 6569, Orlando, Florida, April 2007.
- [45] T. H. Ha, S. Srivastava, E. J. Delp, and J. P. Allebach, "Model based methods for developing color transformation between two display devices," *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 3793–3796.

- [46] R. S. Berns, "Methods for characterizing CRT displays," *Displays*, vol. 16, 1996.
- [47] O. Arslan, Z. Pizlo, and J. P. Allebach, "CRT calibration techniques for better accuracy including low-luminance colors," *Proceedings of the SPIE/IS&T Conference on Color Imaging: Processing, Hardcopy, and Applications*, vol. 5293, San Jose, California, January 2004, pp. 18–22.
- [48] N. Katoh, T. Deguchi, and R. S. Berns, "An accurate characterization of CRT monitor (i) verification of past studies and clarifications of gamma," *Optical Review*, vol. 8, no. 5, pp. 305–314, September-October 2001.
- [49] C. D. Boor, *Guide to Splines*. New York: Springer, 2001.
- [50] G. Kennel, *Color and Mastering for Digital Cinema*. St. Louis, Missouri: Focal Press, 2006.
- [51] S. Bianco, F. Gasparini, A. Russo, and R. Schettini, "A new method for RGB to XYZ transformation based on pattern search optimization," *IEEE Transactions on Consumer Electronics*, vol. 53, no. 3, 2007.
- [52] R. M. Lewis and V. Torczon, "Pattern search methods for linearly constrained minimization," *SIAM Journal on Optimization*, vol. 10, no. 3, pp. 917–941, February-March 2000.
- [53] K. M. Braun, M. D. Fairchild, and P. J. Alessi, "Viewing techniques for cross-media image comparisons," *Color Research & Application*, vol. 21, no. 1, pp. 6–17, 1996.
- [54] R. W. G. Hunt and L. M. Winter, "Colour adaptation in picture-viewing situations," *Journal of Photographic Science*, vol. 23, pp. 112–115, 1975.
- [55] E. D. Montag and M. D. Fairchild, "Evaluation of chroma clipping techniques for three destination gamuts," *Proceedings of the IS&T/SID Color Imaging Conference: Color Science, Systems and Applications*, Scottsdale, Arizona, November 1998, pp. 57–61.
- [56] R. S. Gentile, E. Walowit, and J. P. Allebach, "A comparison of techniques for color gamut mismatch compensation," *Journal of Imaging Technology*, vol. 16, no. 5, pp. 176–181, October 1990.
- [57] J. Morovik, "To develop a universal gamut mapping algorithm," Doctoral dissertation, University of Derby, Derby, UK, 1998.
- [58] The MathWorks, "Optimization toolbox," <http://www.mathworks.com/access/helpdesk/help/toolbox/optim/index.html>, Accessed June 2008.
- [59] H. R. Kang, "3D Lookup Table With Interpolation," *Color Technology for Electronic Imaging Devices*. Bellingham, Washington: SPIE Press, January 1997.
- [60] P. C. Pugsley, "Colour correcting image reproducing methods and apparatus," United States Patent 3,893,166, 1975.
- [61] H. Kotera *et al.*, "Color separating method and apparatus using statistical techniques," United States Patent 4,090,243, 1978.

- [62] M. Abdulwahab, J. L. Burkhardt, and J. J. McCann, "Method of and apparatus for transforming color image data on the basis of an isotropic and uniform colorimetric space," United States Patent 4,839,721, 1989.
- [63] J. J. McCann, "High-resolution color photographic reproductions," *Proceedings of the SPIE: Very High Resolution and Quality Imaging II*, vol. 3025, San Jose, California, February 1997, pp. 53–59.
- [64] J. J. McCann, "Color spaces for color mapping," *Journal of Electronic Imaging*, vol. 8, no. 4, pp. 354–364, October 1999.
- [65] J. J. McCann, "Digital color transforms applied to fine art reproduction," *Proceedings of the International Conference on Imaging Science and Hardcopy*, Guilin, China, May 1995, pp. 377–379.
- [66] S. Srivastava, T. H. Ha, E. J. Delp, and J. P. Allebach, "Generating optimal look-up tables to achieve complex color space transformations," *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 1641–1644.
- [67] R. Bala and V. Klassen, "Efficient implementation of color transformations," *Digital Color Imaging Handbook*, G. Sharma, Ed. Boca Raton, FL: CRC Press, 2002.
- [68] D. Kidner, M. Dorey, and D. Smith, "What's the point? interpolation and extrapolation with a regular grid DEM," *Proceedings of the International Conference on GeoComputation*, Fredericksburg, Virginia, July 1999.
- [69] D. Shepard, "A two-dimensional interpolation function for irregularly-shaped data," *Proceedings of the ACM National Conference*, Princeton, New Jersey, January 1968, pp. 517–524.
- [70] J. Z. Chang, J. P. Allebach, and C. A. Bouman, "Sequential linear interpolation of multidimensional functions," *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1231–1245, September 1997.
- [71] J. M. Kasson, S. I. Nin, W. Plouffe, and L. James, "Performing color space conversions with three dimensional linear interpolation," *Journal of Electronic Imaging*, vol. 4, no. 3, pp. 226–250, July 1995.
- [72] H. Hou and H. Andrews, "Cubic splines for image interpolation and digital filtering," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, no. 6, pp. 508–517, December 1978.
- [73] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part I - Theory," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 821–833, February 1993.
- [74] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part II - Efficiency, design and applications," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834–848, February 1993.
- [75] I. E. Bell and W. Cowan, "Device characterization using spline smoothing and sequential linear interpolation," *Proceedings of the IS&T/SID Color Imaging Conference: Color Science, Systems, and Applications*, vol. 2, Scottsdale, Arizona, November 1994, pp. 29–32.

- [76] G. Sharma and M. Q. Shaw, "Thin-plate splines for printer data interpolation," *Proceedings of the European Signal Processing Conference*, Florence, Italy, September 2006.
- [77] V. Monga and R. Bala, "Sort-select-damp: An efficient strategy for color look-up table lattice design," *Proceedings of the IS&T/SID Color Imaging Conference: Color Science and Engineering Systems, Technologies and Applications*, Portland, Oregon, November 2008, pp. 247–253.
- [78] V. Monga and R. Bala, "Algorithms for color look-up-table (lut) design via joint optimization of node locations and output values," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Dallas, Texas, March 2010, pp. 998–1001.
- [79] R. M. Lewis and V. Torczon, "Pattern search methods for bound constrained minimization," *SIAM Journal on Optimization*, vol. 9, no. 4, pp. 1082–1099, September 1999.
- [80] V. Torczon, "On the convergence of pattern search algorithms," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 1–25, February 1997.
- [81] The Mathworks Inc., "Finding minimum of functions using pattern search," <http://www.mathworks.com/access/helpdesk/help/toolbox/gads/patternsearch.html>, Accessed February 2010.
- [82] Sandia Corporation, "OPT++: an object-oriented nonlinear optimization library," <https://software.sandia.gov/opt++/>, Accessed February 2010.
- [83] E. Dolan and V. Torczon, "C++ class-based pattern search," <http://www.cs.wm.edu/~va/software/PatternSearch/>, Accessed February 2010.
- [84] T. Newman, "Improved color for the world wide web: A case study in color management for distributed digital media," <http://www.color.org/wpaper2.xalter>, Accessed March 2009.
- [85] A. M. McIvor, "Background subtraction techniques," *Proceedings of Image and Vision Computing*, Hamilton, New Zealand, November 2000, pp. 147–153.
- [86] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, August 2000.
- [87] N. Ohta, "A statistical approach to background suppression for surveillance systems," *Proceedings of the International Conference on Computer Vision*, Vancouver, Canada, July 2001, pp. 481–486.
- [88] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh, "Background modeling and subtraction of dynamic scenes," *Proceedings of the International Conference on Computer Vision*, Nice, France, October 2003, pp. 1305–1312.
- [89] S. J. McKenna and S. Gong, "Tracking color objects using adaptive mixture models," *Image and Vision Computing*, vol. 19, pp. 225–231, 1999.
- [90] M. Seki, H. Fujiwara, and K. Sumi, "A robust background subtraction method for changing background," *Proceedings of the Workshop on Applications of Computer Vision*, Palm Springs, California, December 2000, pp. 207–213.

- [91] S. Y. Cheung, S. Coleri, B. Dunder, S. Ganesh, C. Tan, and P. Varaiya, "Traffic measurement and vehicle classification with single magnetic sensor," *Transportation Research Record*, vol. 1917, no. 19, pp. 173–181, 2005.
- [92] X. Song and R. Nevatia, "A model-based vehicle segmentation method for tracking," *Proceedings of the International Conference on Computer Vision*, vol. 2, Beijing, China, October 2005, pp. 1124–1131.
- [93] R. P. Avery, Y. Wang, and G. S. Rutherford, "Length-based vehicle classification using images from uncalibrated video cameras," *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, Washington, D.C., October 2004, pp. 737–742.
- [94] C. Huang and W. Liao, "A vision-based vehicle identification system," *Proceedings of the International Conference on Pattern Recognition*, vol. 4, Cambridge, UK, August 2004, pp. 364–367.
- [95] B. Morris and M. Trivedi, "Robust classification and tracking of vehicles in traffic video streams," *Proceedings of the IEEE Intelligent Transportation Systems Conference*, Toronto, Canada, September 2006, pp. 1078–1083.
- [96] A. H. S. Lai and N. H. C. Yung, "Vehicle type identification through automated virtual loop assignment and block-based direction-biased motion estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 2, pp. 86–97, June 2000.
- [97] R. J. Barnett, "Wireless remote tire parameter measurement method and apparatus," United States Patent 6,448,891, 2002.
- [98] V. S. Petrovic and T. F. Cootes, "Vehicle type recognition with match refinement," *Proceedings of the International Conference on Pattern Recognition*, vol. 3, Cambridge, UK, August 2004, pp. 95–98.
- [99] X. Clady, P. Negri, M. Milgram, and R. Poulencard, "Multi-class vehicle type recognition system," *Proceedings of the International Workshop on Artificial Neural Networks in Pattern Recognition*, Paris, France, July 2008, pp. 228–239.
- [100] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–100, 2004.
- [101] L. Dlagnekov, "Video-based car surveillance: License plate, make and model recognition," Masters Thesis, University of California, San Diego, 2005.
- [102] D. A. Torres, "More local structure information for make-model recognition," Project report, CSE252, University of California, San Diego, 2005.
- [103] I. Zafar, E. A. Erdisinghe, S. Acar, and H. E. Bez, "Two dimensional statistical linear discriminant analysis for real-time robust vehicle type recognition," *Proceedings of IS&T/SPIE Electronic Imaging: Real Time Image Processing*, vol. 6496, San Jose, California, January-February 2007.
- [104] I. Sobel and G. Feldman, "A  $3 \times 3$  isotropic gradient operator for image processing," Unpublished, presented at Stanford Artificial Project, 1968.

- [105] Vehicle and U. Operator Safety Agency, "Vehicle safety: the dangers of overloading," Guidance sheet: [http://www.dvtani.gov.uk/uploads/compliance/VOSA\\_VehicleSafety\\_DangersofOverloading.pdf](http://www.dvtani.gov.uk/uploads/compliance/VOSA_VehicleSafety_DangersofOverloading.pdf), Accessed: July 2010.
- [106] S. H. Dockstader, "Motion trajectory classification for visual surveillance and tracking," *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance*, Sydney, Australia, November 2006.
- [107] S. Srivastava and E. J. Delp, "Standoff video analysis for the detection of security anomalies in vehicles," *Proceedings of the IEEE Applied Imagery Pattern Recognition Workshop*, Washington, DC, October 2010.
- [108] L. Dlagnekov, "Video-based car surveillance: License plate, make and model recognition," Masters Thesis, University of California, San Diego, 2005.
- [109] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, vol. 34, no. 3, pp. 334–352, August 2004.
- [110] B. T. Morris and M. M. Trivedi, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1114–1127, August 2008.
- [111] G. Medioni, I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia, "Event detection and analysis from video streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 8, pp. 873–889, August 2001.
- [112] T. Zhang, H. Lu, and S. Li, "Learning semantic scene models by object classification and trajectory clustering," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Miami, Florida, June 2009, pp. 1940–1947.
- [113] A. Basharat, A. Gritai, and M. Shah, "Learning object motion patterns for anomaly detection and improved object detection," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008, pp. 1–8.
- [114] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: Real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809–830, August 2000.
- [115] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, December 2006.
- [116] S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Panikolopoulos, "Detection and classification of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 37–47, March 2002.
- [117] K. K. Ng and D. Edward J, "Object tracking initialization using automatic moving object detection," *Proceedings of IS&T/SPIE Electronic Imaging: Visual Information Processing and Communication*, San Jose, California, January 2010.

- [118] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, February 2002.
- [119] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. New York: Cambridge University Press, 2004.
- [120] I. L. Dryden and K. V. Mardia, *Statistical Shape Analysis*. New Jersey: Wiley, 1998.
- [121] J. C. Gower and G. B. Dijkstra, *Procrustes Problems*. Oxford, UK: Oxford University Press, 2004.
- [122] J. Harguess and J. K. Aggarwal, "Semantic labeling of track events using time series segmentation and shape analysis," *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 4317 – 4320.
- [123] L. M. Brown, "Example-based color vehicle retrieval for surveillance," *Proceedings of the IEEE International Conference on Advanced Video and Signal-Based Surveillance*, Boston, Massachusetts, August-September 2010, pp. 91–96.
- [124] F. Porikli, "Inter-camera color calibration by correlation model function," *Proceedings of the IEEE International Conference on Image Processing*, Barcelona, Spain, September 2003, pp. 133–136.
- [125] A. Gilbert and R. Bowden, "Tracking objects across cameras by incrementally learning inter-camera colour calibration and patterns of activity," *Proceedings of the European Conference on Computer Vision*, Graz, Austria, May 2006, pp. 125–136.
- [126] Y. Choi, Y. Lee, and W. Cho, "Color correction for object identification from images with different color illumination," *Proceedings of the International Conference on Networked Computing and Advanced Information Management*, Seoul, Korea, August 2009, pp. 1598–1603.
- [127] B. Fraser, C. Murphy, and F. Bunting, *Real World Color Management*. Berkeley, California: Peachpit Press, 2004.
- [128] F. M. Verdu, J. Pujol, and P. Capilla, "Characterization of a digital camera as an absolute tristimulus colorimeter," *Journal of Imaging Science and Technology*, vol. 47, no. 4, pp. 279–295, July-August 2003.
- [129] K. K. Ng and E. J. Delp, "New models for real-time tracking using particle filtering," *Proceedings of SPIE/IS&T Electronic Imaging: Visual Communications and Image Processing*, vol. 7257, San Jose, California, January 2009, pp. 72570B:1–12.
- [130] K. Nummiaro, E. Koller-Meier, and L. V. Gool, "Color features for tracking non-rigid objects," *Chinese Journal of Automation - Special Issue on Visual Surveillance*, vol. 29, no. 3, pp. 345–355, May 2003.
- [131] S. Srivastava, T. H. Ha, J. P. Allebach, and E. J. Delp, "Color management using optimal three dimensional look-up tables," *Journal of Imaging Science and Technology*, vol. 54, no. 3, pp. 402:1 – 402:14, May-June 2010.



- [132] C. Yang-Ho, I. M. Hye-Bong, and H. A. Yeong-Ho, "Inverse characterization method of alternate gain-offset-gamma model for accurate color reproduction in display devices," *Journal of Imaging Science and Technology*, vol. 50, no. 2, pp. 139–148, March-April 2006.
- [133] LPROF project team, "LPROF ICC profiler," <http://lprof.sourceforge.net/>, Accessed April 2009.
- [134] Committee for Graphics Arts Technologies Standards, "Graphic technology - color transmission target for input scanner calibration," ANSI Technical standard IT8.7/1-1993, Reston, Virginia, 2008.
- [135] K. M. Lam, "Metamerism and colour constancy," Doctoral dissertation, University of Bradford, Bradford, UK, 1985.
- [136] M. R. Luo and R. W. G. Hunt, "The structure of the CIE 1997 colour appearance model (CIECAM97s)," *Color Research and Applications*, vol. 23, no. 3, pp. 138–146, December 1998.
- [137] The Next Generation Simulation (NGSIM) Community, "Data sets," Web archive: <http://ngsim-community.org/>, Accessed: May 2010.
- [138] "Google maps," Online: <http://maps.google.com>, Accessed: August 2010.

VITA

## VITA

Satyam Srivastava was born in Rai Bareilly, Uttar Pradesh, India. He obtained his Bachelor of Engineering (Honors) degree in Electrical and Electronics Engineering (EEE) at Birla Institute of Technology and Science, Pilani, India in 2006. He was recognized as the Best Out-Going Student in the EEE Class of 2006.

Satyam joined the Direct PhD program at Purdue university, West Lafayette, Indiana in 2007. Since then, he has served as Teaching Assistant for the School of Electrical and Computer Engineering and as Research Assistant at the Video and Image Processing Laboratory (VIPER). His major advisor Professor Edward J. Delp is the Charles William Harrison Distinguished Professor of Electrical and Computer Engineering. While in the graduate program, Satyam has worked on projects sponsored by the Indiana 21<sup>st</sup> Century Research and Technology Fund, the United State Department of Homeland Security, and the United States Naval Research Laboratory. His current research interests include image and video processing, image analysis, color science, human and machine vision, and information theory.

Satyam is a student member of the IEEE and the IEEE Signal Processing Society.

Satyam Srivastava's publications from this research work include:

**Journal Articles:**

1. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, "Color Management Using Optimal Three Dimensional Look-Up Tables," *Journal of Imaging Science and Technology*, vol. 54, no. 3, May-June 2010, pp. 030402 (1-14).
2. **Satyam Srivastava** and Edward J. Delp, "Autonomous Visual Surveillance of Vehicles for the Detection of Anomalies," *IEEE Transactions on Intelligent Transport Systems*, submitted.

**Conference Papers**

1. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, "Co-Ordinate Mapping and Analysis of Vehicle Trajectory for Anomaly Detection," *Proceedings of the IEEE International Conference on Multimedia and Expo*, Barcelona, Spain, July 2011 (to appear).
2. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, "Color Correction for Object Tracking Across Multiple Cameras," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, May 2011 (to appear).
3. **Satyam Srivastava** and Edward J. Delp, "Standoff Video Analysis for the Detection of Security Anomalies in Vehicles," *Proceedings of the IEEE Applied Imagery Pattern Recognition Workshop*, Washington DC, October 2010.
4. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, "Generating Optimal Look-Up Tables to Achieve Complex Color Space Transformations," *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 1641-1644.
5. Thanh H. Ha, **Satyam Srivastava**, Edward J. Delp, and Jan P. Allebach, "Model Based Methods for Developing Color Transformation Between Two Dis-

- play Devices,” *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 3793-3796.
6. Thanh H. Ha, **Satyam Srivastava**, Edward J. Delp, and Jan P. Allebach, “Monitor Characterization Model Using Multiple Non-Square Matrices for Better Accuracy,” *Proceedings of the IS&T Color Imaging Conference*, Albuquerque, New Mexico, USA, November 2009, pp. 117-122.
  7. **Satyam Srivastava**, Thanh H. Ha, Jan P. Allebach, and Edward J. Delp, “Color Management Using Device Models and Look-Up Tables,” *Proceedings of the Gjøvik Color Imaging Symposium*, Gjøvik, Norway, June 2009, pp. 54-61.
  8. **Satyam Srivastava**, Ka Ki Ng, and Edward J. Delp, “Crowd Flow Estimation Using Multiple Visual Features for Scenes with Changing Crowd Densities,” *Proceedings of the IEEE International Conference on Advanced Video and Signal based Surveillance*, Klagenfurt, Austria, 2011, submitted.