

Segmentation of Cast Shadows from Moving Objects

Master of Science Thesis in Electrical and Electronic Engineering
(M.Sc.E.E.)

Søren Gylling Erbou

s990087

September 2004



Section for Electronics and Signal Processing, Ørsted•DTU
Technical University of Denmark (DTU)
DK-2800 Kgs. Lyngby

In cooperation with:
The Danish Defence Research Establishment (DDRE)

Supervisors:
Helge B.D. Sørensen, Ørsted•DTU
Bjarne Stage, DDRE

F-15/2004

Preface

This thesis is the result of work carried out at the section for Electronics and Signal Processing, Ørsted•DTU, Technical University of Denmark (DTU). The thesis accounts for 30 ECTS units and is a partial requirement for obtaining the degree of Master of Science in Electrical and Electronic Engineering (M.Sc.E.E.). The work has been carried out over a period of six months, in cooperation with the Danish Defence Research Establishment (DDRE) (*Forsvarets Forskningstjeneste, FOFT*).

The thesis is intended as a contribution to reducing the problems introduced by cast shadows, when detecting moving objects in systems for automated video surveillance. It is assumed that the reader has a basic knowledge within the areas of image analysis and statistics. Key flowcharts, which are referred to throughout the thesis, are additionally placed in the final appendix F, page 191, for the convenience of the reader.

Svanemøllen Kaserne, September 16, 2004.

Søren Gylling Erbou, s990087.

Acknowledgements

Several people have contributed to this thesis with encouragement and support through many fruitful discussions.

My supervisor at the DDRE, Bjarne Stage, had the ability to always ask the right questions in times of despair. I owe great debt to Christian Birkemark, who never lacked any interest in discussing minor or major aspects of methods, or in reflecting over my writing. Thomas Sams showed great interest in discussing many of the more physics-based aspects of the thesis. Erik Thiesen was an invaluable help during the data acquisition. Torben Christensen and Anders F. Johnsen were always available for discussion of statistical considerations. Additionally, I would like to thank the whole *Institut for Sensorsystemer* at the DDRE for providing a pleasant and inspiring atmosphere during my stay.

My supervisor at DTU, Helge Sørensen, was always supportive and constructive in suggestions for improvements. Furthermore, Omar Javed, University of Central Florida, clarified some of the more subtle aspects of his work.

Finally I would like to thank my family and friends for always being supportive and for proofreading. Jane, in particular, has been ever patient and encouraging during the whole period.

Abstract

This thesis describes and implements methods for segmentation of cast shadows from moving objects, detected in an outdoor surveillance application. Cast shadows reduce the general ability of robust classification, and tracking, of moving objects in such applications.

A data set, consisting of 90 different foreground objects including cast shadows, is obtained using a high resolution digital video camera, in a typical surveillance scenario. 18 of the foreground objects constitute a training set used for manually optimizing central parameters. 72 foreground objects constitute the test set, used for validation.

A state-of-the-art statistical-based method for handling cast shadows, suggested by Javed et al. [21], is implemented as a reference, and its central parameters optimized using the training set. A physics-based method for shadow removal in still images, suggested by Finlayson et al. [15] and not previously applied in a surveillance application, is examined for use in such an application, but found to be too sensitive when used with a standard dynamic range of 8 bits. Instead an enhanced method for segmentation of cast shadows is suggested, combining an improved color segmentation of regions, with the introduction of an enhanced similarity feature for classification of regions. None of the methods are, in practice, limited by spatial assumptions.

Based on the 72 examples of the test set, the enhanced method for shadow removal significantly improves the mean absolute accuracy (69.2%), and mean relative accuracy (14.9%), at a 5% significance level, compared to the reference method, whose mean absolute accuracy is 64.9%. The enhanced method tends to improve examples substantially, where the reference method fails completely. Therefore the enhanced method is also more robust than the reference method.

Resumé

I denne afhandling beskrives og implementeres metoder til segmentering af kasteskygger fra objekter i bevægelse, i et system til automatisk udendørs videoovervågning. Kasteskygger er et generelt problem i overvågningssystemer, da de har negativ indflydelse på den senere klassifikation og sporing af objekter.

Et datasæt bestående af 90 forskellige forgrundsobjekter med kasteskygge, er blevet optaget med et digitalt videokamera, i et typisk overvågningsscenarie. 18 forgrundsobjekter udgør et træningssæt, der anvendes til at optimere centrale parametre, og 72 forgrundsobjekter udgør et testsæt, der anvendes til validering.

En state-of-the-art statistik-baseret metode, foreslået af Javed et al. [21], implementeres som referencemodel, og dens ydelse optimeres i fht. centrale parametre, ud fra træningssættet. En fysik-baseret metode, til fjernelse af skygger fra enkelt-billeder og foreslået af Finlayson et al. [15], undersøges også for anvendelse i videoovervågning. Denne vurderes at være for følsom ved anvendelse af et videokamera med et standard dynamikområde på 8 bits. I stedet for foreslås en forbedret metode til segmentering af kasteskygger, som kombinerer en bedre farvesegmentering med indførelsen af en ny egenskab til klassifikation. Ingen af metoderne er i praksis begrænset af spatiale antagelser om sammensætningen af forgrundsobjekterne.

På baggrund af træningssættet viser den forbedrede metode en signifikant forbedring i absolut middel-nøjagtighed (69.2%), og i relativ middel-nøjagtighed (14.9%), sammenlignet med referencemetoden, hvis absolute middel-nøjagtighed er 64.9%. Den forbedrede metode giver en meget stor forbedring i tilfælde hvor referencemetoden fejler fuldstændigt, hvorfor den forbedrede metode derfor også er mere robust end referencemetoden.

Contents

Preface	i
Acknowledgements	ii
Abstract	iii
Resumé	iv
Contents	v
1 Introduction	1
1.1 Motivation	2
1.2 Objectives	3
1.3 System Specifications	3
1.4 Thesis Overview	4
2 Related Work	5
2.1 Computer Vision in Video Surveillance	5
2.2 W ⁴ - A System For Automated Video Surveillance	6
2.3 Shadow Removal in General	8
2.4 Statistical-Based Shadow Removal	9
2.4.1 Hsieh et al.	9
2.4.2 Javed et al.	10
2.5 Physics-Based Shadow Removal	12
2.5.1 Nadimi et al.	12
2.5.2 Finlayson et al.	13
2.6 Comparison	19
2.7 Summary	20
3 Data Acquisition	22
3.1 Camera	22
3.2 Data Sets	24
3.3 Summary	26

4	Finlayson’s Approach Using a Video Camera	27
4.1	Spectral Sensor Functions	27
4.2	Color Calibration	28
4.3	Summary	33
5	Implementation and Optimization	35
5.1	Background Modelling and Noise Reduction	35
5.2	Measuring Performance	37
5.3	Javed’s Method	39
5.4	Improving Javed’s Method	43
5.5	Finlayson’s Shadow Removal Applied for Surveillance	46
5.5.1	Detecting Edges due to Shadows	47
5.5.2	Reconstructing the RGB-image without Shadows	49
5.6	Enhanced Shadow Removal	50
5.6.1	Enhanced Similarity Feature	51
5.6.2	Applying the Enhanced Similarity Feature	52
5.7	Summary	54
6	Validation and Comparison	56
6.1	Absolute Performance	56
6.2	Relative Performance	57
6.3	Comparison Per-Example (Binomial)	59
6.4	Comparison of Means (Paired t-Test)	61
6.4.1	Absolute Means	62
6.4.2	Relative Means	63
6.5	Summary	64
7	Discussion	66
7.1	Results	66
7.2	Limitations	67
7.2.1	Data Acquisition	67
7.2.2	Javed’s method	67
7.2.3	Finlayson’s method	67
7.2.4	Enhanced method	68
7.3	Future Work	68
7.4	Perspectives	69
8	Conclusion	71
8.1	Implementation of State-of-the-Art Reference Method	71
8.2	Improving Reference Method	71
8.3	Applying Physics-Based Method	72
8.4	Final Results	73
8.5	Contributions	73

List of Figures	75
List of Tables	77
Bibliography	78
Appendices	81
A Macbeth Color Chart	81
B Data Sets - Foreground Objects to Classify	82
B.1 Training Set	82
B.2 Test Set	86
C Additional Figures	97
C.1 Detecting Shadow Edges from Illumination-Invariants	97
D Additional Results	100
D.1 Performance of Training Set	101
D.2 Performance of Test Set	107
E Matlab Routines	131
E.1 MakeFiles.m	132
E.2 BayerGR_fast.m	132
E.3 main01_SGE.m	133
E.4 ChooseFrames.m	137
E.5 Noise_Reduction_SGE.m	137
E.6 Do_Shadow.m	138
E.7 DataSets.m	142
E.8 Javed.m	144
E.9 JavedImproved.m	147
E.10 MergeRegions.m	150
E.11 Finlayson_Ill_Inv.m	151
E.12 Finlayson_FGmask.m	154
E.13 SolvePoisson.m	155
E.14 EnhancedSegmentation.m	156
E.15 DetectVariance.m	158
E.16 Performance.m	160
E.17 Compare.m	166
E.18 PlotComparison.m	167
E.19 PlotPerformance.m	171
E.20 Calibration.m	174
E.21 CropPNG.m	178
E.22 OptimizeJaved.m	178
E.23 OptimizeJavedImproved.m	181

E.24 OptimizeEnhanced.m	184
F Flowcharts	191

Chapter 1

Introduction

For several decades video cameras have been a popular means for crime solving by surveillance. Conventional surveillance applications require an operator to determine when action is needed. A single operator can only monitor a limited amount of scenes simultaneously, for a limited amount of time, because the process of manual surveillance becomes tedious. The introduction of digital video cameras, and recent advances in computer technology, make it possible to apply (semi-)automated processing steps to reduce the amount of data presented to the operator. This way the amount of trivial tasks are reduced, and the operator can focus on a correct and immediate interpretation of the activities in a scene.

In recent years the main attention has been on surveillance applications where it is necessary to take immediate action because human lives or installations of vital interest are at stake. This could e.g. be scenarios where a terrorist leaves a bag containing a bomb in a scene, or perimeter surveillance where it is crucial to detect unwanted intrusion. In such surveillance applications it is vital to ensure a consistent way of monitoring and registration of objects of interest. Automated or semi-automated video surveillance are steps in this direction, since they are capable of monitoring larger scenes over a longer period of time.

The Danish Defence Research Establishment (DDRE) is currently focusing part of it's research on implementing a system for automated video surveillance. The main objectives of the DDRE are to gain general knowledge in this area, and eventually implement an automated surveillance application that is capable of detecting, tracking and classifying moving objects of interest.

At this point the DDRE has carried out some initial studies [28, 18] in testing and implementing parts of the W^4 -system [19] for automated video surveillance. The W^4 -system effectively detects moving objects, tracks them through simple occlusions (blocking of the view), classifies them and performs an analysis of their behavior. This procedure corresponds well to the system that the DDRE would like to implement, and therefore the W^4 has been chosen as a primary reference. One limitation of W^4 is that the tracking, classification and analysis of objects fails when large parts of the moving objects are actually cast shadows.

Distinguishing between cast shadows and self shadows is crucial for the further analysis

of moving objects in a surveillance application. Self shadows occur when parts of an object are not illuminated directly, but only by diffuse lighting. Cast shadows occur when the shadow of an object is cast onto background areas, cf. figure 1.1. The latter are a major concern in today’s automated surveillance systems because they make shape-based classification of objects very difficult. Furthermore cast shadows can make objects that interact difficult to track.

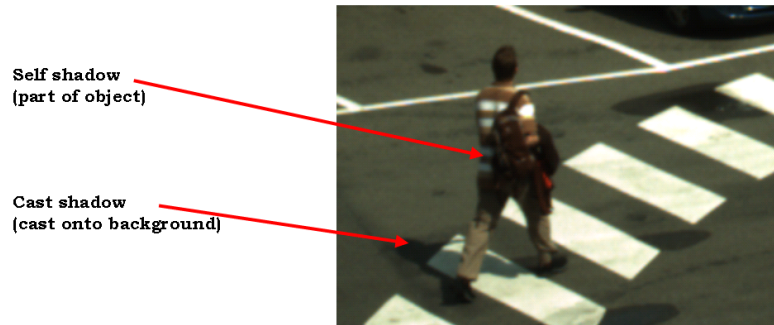


Figure 1.1: *Types of shadows. Self shadow is shadow on the object itself, a person in this case. Cast shadow is the shadow cast onto the background.*

1.1 Motivation

Cast shadows in outdoor scenarios are very likely to occur, and the problem of cast shadows in surveillance applications, is yet to be solved in general. Several approaches have been tried, but they all are limited by context dependent threshold optimized for specific applications and data sets. The DDRE surveillance application also lacks a robust shadow handling for the moving objects detected.

In [18], Hansen implements and improves upon a method for cast shadow removal based on work by Hsieh et al. [20]. The use of the method is limited to people in standing posture, because of some initial spatial assumptions of the composition of objects. For instance it often fails to segment cast shadows from vehicles. This makes the method less useful if the outdoor environment to be monitored, contains roads or parking lots, as required by the DDRE.

Javed et al. [21] use a statistical approach for segmenting foreground pixels darker than a reference image into cast shadow, self shadow and object pixels darker than the background. This method is considered state-of-the-art in surveillance applications but still faces fundamental problems concerning some very context dependent parameters.

Finlayson et al. [15] use a physics-based approach to derive an illumination invariant (therefore shadow free) gray-scale image of an RGB image. From this image the original RGB image, without shadows, is derived. Finlayson’s approach is aimed at shadow elimination in general in images obtained with a standard digital still camera. Due to assumptions in the model, and in the derivation of the shadow free RGB image, the method is far from perfect, but shadows are attenuated significantly. The method has

not been applied in a surveillance application yet.

The topic of the present thesis is therefore based on the need for a more robust way of dealing with cast shadows in surveillance applications.

1.2 Objectives

The main objective is to contribute to the design of an overall system for automated outdoor video surveillance. More specifically the focus is on methods for robust segmentation of cast shadows from moving objects.

An overview of recent methods for shadow removal is given, with emphasis on two fundamentally different approaches: A statistical approach suggested by Javed et al. [21] and a physics-based approach suggested by Finlayson et al. [15]. Both methods are studied in detail and are implemented in Matlab [23]. In order to evaluate and compare methods, a data set consisting of images typical of the environment that the DDRE wishes to monitor, is acquired.

Finlayson's approach has not previously been applied in a surveillance application or when using a digital video camera. Using such a setup, Finlayson's approach is examined to determine its applicability.

Javed's statistical approach is considered state-of-the-art and is optimized with respect to a training set and chosen as a reference (J). Then an improved version of Javed's approach (I) is suggested based on the results from the training set. Finally Finlayson's ideas are combined with Javed's improved approach in an enhanced algorithm for shadow removal (E).

The three methods (J, I and E) are then compared to each other using a test set, to determine if there are any statistically significant improvements in performance and from where such improvements might originate.

1.3 System Specifications

Several specifications for a system for shadow removal are outlined by the DDRE and the author to encompass a suitable master thesis.

The focus of the thesis is on applications using a single camera, for which reason a single digital video camera should be used to obtain the data set used to train and test the methods. The data set should represent objects that are relevant in reference to the present DDRE application, i.e. vehicles, people and bicycles. Input for the shadow removal algorithm are the moving foreground pixels detected by the algorithm implemented by Hansen [18], for the DDRE. These consist of both object pixels and cast shadow pixels (cf. figure 1.1). The segmentation of pixels should not be limited by any spatial assumptions of the object, since this would limit the object types that the method can handle. The implementation of Javed's method is used as a reference, since it is considered a state-of-the-art method for shadow removal. From an analysis of the reference method and Finlayson's ideas for shadow removal, the enhanced method for shadow removal should result in an increased performance. Finally, the data set used for comparing

the methods should be of an appropriate size to ensure statistical significance at a 5% level, when interpreting the results.

1.4 Thesis Overview

Chapter 2 gives an introduction to computer vision in automated video surveillance with the W^4 -system as the key reference. Several approaches for shadow removal are compared, with emphasis on the statistical approach by Javed et al. and the physics-based approach by Finlayson et al.

Chapter 3 describes the equipment used for data acquisition and the data sets used for training and validation of the methods. In chapter 4 Finlayson's ideas for shadow removal are examined for the digital video camera at hand, i.e. the illumination invariant image is estimated including a color calibration of the camera.

Chapter 5 gives a brief overview of the implementation of the background model and detection of foreground objects. Then performance measures are introduced leading to the implementation and optimization of Javed's shadow removal (J), which is used as a reference. Using the training set an improved version of Javed's method is suggested (I). Finally the latter is combined with some of Finlayson's ideas for shadow removal in an enhanced algorithm (E). In chapter 6 the three methods are applied on a test set and compared to each other.

Chapter 7 discusses the results obtained and how they should be interpreted. It also summarizes the work of the thesis and gives proposals for future work. Chapter 8 is the final conclusion. Appendices contain supplementary figures, results and Matlab routines. The final appendix F, page 191, contains the key flowcharts, for the convenience of the reader.

Chapter 2

Related Work

In order to make appropriate decisions on how to design a robust system for shadow handling, that corresponds to the intentions of the DDRE, a detailed study of important previous work is presented in this chapter. Both work related to video surveillance in general, and shadow removal in particular, are described.

2.1 Computer Vision in Video Surveillance

Computer Vision is a broad term covering a range of applications. When applied in surveillance tasks, it usually consists of one or more of the following parts: Object detection, tracking, classification and/or analysis. The use varies from traffic monitoring through video conference applications to use in security systems. In the relevant literature several approaches have been tried in order to obtain robust and good results in the highly complex task of interpreting video sequences. Outdoor surveillance in particular is a difficult task because of non-stationary conditions imposed by various types of weather, time of day, season, etc. Therefore the best performance is achieved for specialized systems, using a lot of *a priori* information and assumptions. The drawback being the inability to apply the methods in other situations.

Moeslund et al. [25] make a comprehensive survey (2001) of computer vision systems for human motion capture. Systems are divided into three application areas: Surveillance, control and analysis. Surveillance tasks usually take place in uncontrolled outdoor environments, requiring a high degree of robustness. As a state-of-the-art example, the W^4 system [19] is emphasized, cf. section 2.2. It uses a robust monocular 2D-approach, and deals with all of the previously mentioned aspects of surveillance. Control applications are characterized by an increasing number of assumptions, typically in indoor scenes, concerning e.g. gesture recognition etc. Complex 2D or 3D human models are often introduced, e.g. Pfinder/SPfinder by Wren et al. [39]. Analysis applications are even more specialized, typically for clinical use, and therefore of no relevance within the present framework.

Other interesting work within the area is performed by Gavrilu et al. [16] and Sidenbladh et al. [33] using complex 3D-models. 3D-models presently have the drawback of being too slow for realtime systems, and they often require an advanced camera setup

and a number of model assumptions. Siebel [34] implements a 2D-system for detection and tracking of moving objects in a London underground station. An active shape model is employed on single persons in standing posture, in order to do an analysis of their behavior. It cannot handle postures other than standing and assumes diffuse constant lighting (no shadows), making it unsuitable for outdoor use.

McKenna et al. [24] use color histograms to detect and track people, also through occlusions. Histogram methods can estimate any distribution, but require a large number of samples, which grow exponentially with the dimensionality [2]. Park et al. [27] segment and track interacting human body parts under occlusion and shadowing in a three step process. At pixel level a Gaussian mixture model is used to classify individual pixel colors. Then a Markov Random Field model is used to merge similar pixels into blobs, and finally a model of the human body is applied to handle occlusions. Certain types of occlusion are reported to still produce errors.

Background subtraction is a computationally effective, and therefore popular, way of detecting moving objects in a scene. The idea is to subtract an image of the scene without moving objects from a new frame of the scene. If the resulting image contains areas where the intensity has changed significantly, it is likely to be caused by a moving object. This works fairly well in indoor scenes, where there is little change in the background. However in outdoor environments, with changing illumination, vegetation moving due to wind etc., background subtraction is insufficient, even when using averaged background images. Usually re-initializing the model is done when large parts of an image are considered moving pixels.

In [21] Javed et al. design a robust system for outdoor surveillance, based on a statistical model (mixture of Gaussians) for adaptive background subtraction developed by Stauffer et al. [35]. Then they apply statistical modelling of shadows and use recurrent motion of objects to classify them as a single person, a group of persons or a vehicle. If a single person is found, symmetry analysis is used to detect objects carried by the person. The system performs well for an outdoor system, although it fails under sudden changes in lighting conditions and when irrelevant objects move e.g. flags waving. These drawbacks are general for state-of-the-art systems. The handling of shadows has difficulties in some cases, which will be described in section 2.4.

Elgammal et al. [11] suggest modelling the background and foreground using non-parametric kernel density estimation, thereby avoiding making any assumptions of the shape of the probability density functions (pdf's). Gaussian kernel functions are chosen and the model is reported to be able to detect moving targets against a cluttered background. It handles a background which is not completely static, as well as slow changes in illumination. Nonparametric modelling using kernel estimators requires much more computation, and therefore is difficult to implement in realtime.

2.2 W^4 - A System For Automated Video Surveillance

The W^4 developed by Haritaoglu et al. [19] (2000), is a system for real time surveillance of people and their activities. It detects moving objects using a statistical back-

ground model and classifies objects into single people, groups of people and "other" objects, using a histogram based technique for head detection. Detected objects are tracked by a second order motion model and a silhouette correlation. When a single person is tracked, an appearance model is build to handle occlusions and the different body parts and the posture are detected from the silhouettes. Finally symmetry and periodicity analysis are employed to determine whether the person is carrying an object. People in groups are segmented by detecting their heads from the silhouettes and then using a distance map. Figure 2.1 shows the system architecture of the W^4 , with an indication of which stage to implement a step for shadow removal (step 3 in red).

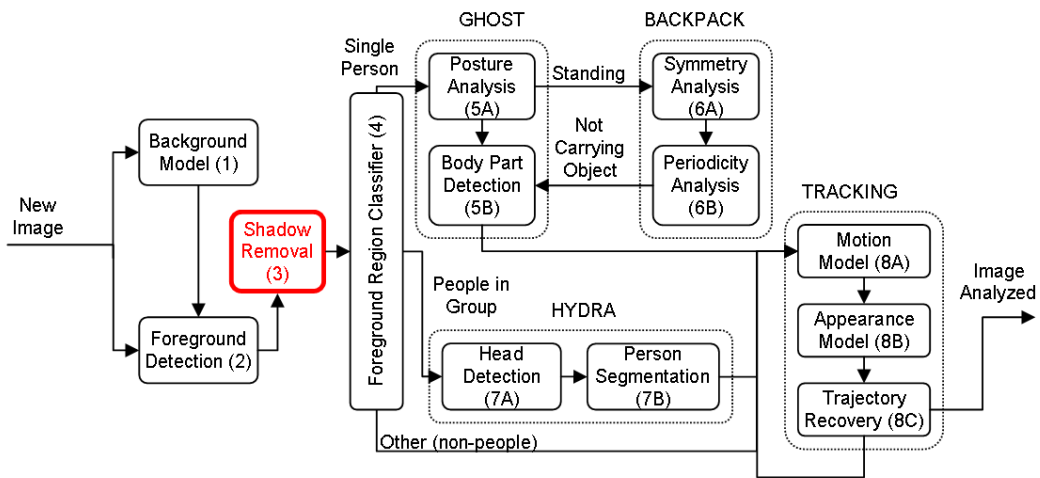


Figure 2.1: The system architecture of W^4 [19] with additional shadow removal. Step 3 (red) indicates when the shadow removal should be performed.

Originally W^4 was implemented using monocular video sources, i.e. grayscale or infrared, and runs at 25Hz for 320x240 resolution images on a 400MHz dual-Pentium II PC. Real time implementation had a high priority in designing W^4 thus a relatively simple background model was chosen, namely a statistical model of the background to subtract from each new frame. It uses the minimum and maximum intensity values and the maximum intensity difference between consecutive frames in a training period, in which there is assumed to be no moving objects present. The background model is then updated using both a pixel-based and an object-based method.

This simple model is susceptible to noise, but more robust background models have been developed in [11, 35]. Furthermore the W^4 does not take shadows into account at all. This is reported to produce significant problems in the silhouette based analysis, and is the key motivation for this thesis. While the tracking handles occlusions quite robustly, the silhouette based methods have problems detecting body parts during occlusions by stationary objects. In general W^4 performs well, but still has problems with sudden changes in illumination, a non-stationary background and shadows. Never the less it is state-of-the-art and one of the only systems that implements a surveillance application from detection through tracking and classification to an analysis of behavior. This is the

main reason for the DDRE to use W^4 as the key reference in developing a system for automated video surveillance.

Hansen [18] implemented a histogram-based method and the kernel-based method by Elgammal for the DDRE, and showed that the latter is more robust. Therefore it is Hansen's implementation, slightly changed to improve the speed in Matlab, of Elgammal's kernel-based method that will be used as the background model for detection of moving objects in this thesis.

2.3 Shadow Removal in General

There are two types of shadows, self shadow and cast shadow. Self shadow occurs when part of an object is in shadow, i.e. when part of an object is not illuminated by direct light. Cast shadow is the shadow cast onto background regions, i.e. when background regions are not illuminated by direct light because an object blocks the direct light, cf. figure 1.1. Cast shadows are a major concern in tracking and recognition tasks. A variety of approaches have been tried in search of a robust method to deal with shadows depending on the application. In [29, 30] Prati et al. give a comparative evaluation of the most important methods up until 2001. They conclude that the more general situations a system is designed to handle, the less assumptions should be made, and if the scene is noisy, a statistical approach is preferable to a deterministic model due to the uncertainty introduced in the classification. Typical features can be divided into (extension of [29][30]):

- Spectral features.
 - Gray level.
 - Color.
 - Infrared.
- Spatial features.
 - Local (pixel).
 - Regional.
- Temporal features.
 - Static.
 - Dynamic.

The vast majority of recent methods (2001 →) use color information in shadow handling because it provides extra information. The single reason for using gray levels only, is to reduce computations in realtime systems. Infrared sensors are typically used in dark environments. Some methods introduce processing at regional level as an extension of pixel-based processing. Temporal features are introduced to exploit information from analysis of previous frames. The assumptions used when applying features can be very

different. Most methods use *a priori* knowledge of how shadows affect surfaces: That the chromaticity of a pixel is largely unchanged, and that the intensity is somewhat attenuated. Some methods try to model the shadows statistically, e.g. [20, 21], making various spatial assumptions. Others suggests more physics-based approaches, e.g. [15, 26].

2.4 Statistical-Based Shadow Removal

2.4.1 Hsieh et al.

In [20], Hsieh et al. focus on removing cast shadows from pedestrians using a statistical model combined with spatial assumptions. Only situations with pedestrians in an upright posture are handled and the cast shadows are assumed to touch their feet. The shadow detection uses the output of a simple background subtraction method, and is done in two stages. From the central moments of the moving object the orientation of the object is determined. Combined with a vertical histogram difference and a silhouette curve a straight line is computed to separate the cast shadow from the rest of the object as a rough approximation. In the second stage pixels roughly detected as shadow pixels are modelled as a Gaussian using the variance-normalized intensity differences between background and foreground, and the spatial displacement of the pixel from the centroid (center-of-mass) of the roughly separated cast shadow. If the probability of a pixel still being part of the cast shadow is below a certain threshold it is not considered a cast shadow after the second stage classification.

The described method is intended for single-shadow elimination but is also implemented to handle multiple shadows. Using a vertical projection histogram and a vertical edge histogram, it distinguishes several persons from each other. It is reported to work very well with an average accuracy of 94% (detected shadow pixels / manually obtained ground truth) on images with a cluttered background. No average false alarm rate is computed in the article [20], but the examples shown reveal a false alarm rate from 1 – 10%, . As a reference model they implement a physics-based approach by Nadimi et al. [26] (cf. section 2.5) which is reported to have an average accuracy of 58.7% on the same data set.

Hansen [18] implements an improved version of Hsieh’s single shadow elimination algorithm using color information and some spatial criteria to prevent the algorithm from failing when objects are vehicles. Color information is reported not to improve performance. In the less specialized cases, as assumed in [20], the method fails in a number of cases:

- If the object is not a person in an upright posture.
- If the object is not "solid", that is if there are holes in the binary moving object.
- If the cast shadow does not touch the objects feet, e.g. when part of the cast shadow is occluded by the object itself or by other objects.

All in all, Hsieh’s method for shadow elimination is found to be too specific for use in a surveillance system for the DDRE, mainly due to the limitations of the first bullet point

mentioned above. The spatial assumptions made previous to the statistical modelling constrains the model too much.

2.4.2 Javed et al.

Javed et al. [21] use a Gaussian mixture model suggested by Stauffer et al. [35] for adaptive background subtraction. They make no spatial assumptions of posture or composition prior to the statistical shadow modelling. Instead they assume that only moving object pixels, darker than the background image, in both the R-,G- and B-channel, candidate as shadow pixels. These can belong to:

- Cast shadow.
- Self shadow.
- Part of the object, darker than the background pixel.

A flowchart of Javed’s method for shadow removal is depicted in figure 2.2, corresponding to step 3 in figure 2.1.

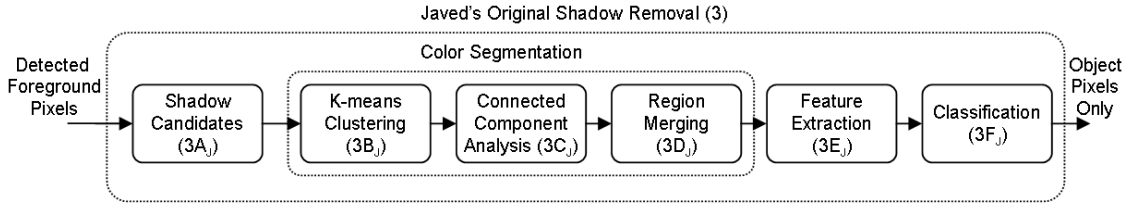


Figure 2.2: Flowchart of shadow removal as suggested by Javed. Corresponds to step 3 in figure 2.1.

When pixel-candidates are found (step $3A_J$) a K-means approximation of the EM-algorithm (*Expectation-Maximization*) [2] is used to perform unsupervised color segmentation of the pixel candidates (step $3B_J$). Each pixel candidate is assigned to one of the K existing Gaussian distributions if the Mahalanobis distance (2.2) is below a certain threshold. If above this threshold a new distribution is added with its mean equal to the pixel value. All distributions are assumed to have the same fixed covariance matrix $\Sigma = \sigma^2 \mathbf{I}$, where σ^2 is a fixed variance of the colors and \mathbf{I} is the identity matrix. After a pixel candidate is assigned to a distribution, the distribution mean is updated as follows:

$$\mu_{n+1} = \mu_n + \frac{1}{n+1}(x_{n+1} - \mu_n), \quad (2.1)$$

where x is the color vector of the pixel and μ_n is the mean of the Gaussian before the $n+1$ th pixel is added to the distribution. The Mahalanobis distance D^2 is defined as [6]:

$$D^2 = (x_{n+1} - \mu_n)^T \Sigma^{-1} (x_{n+1} - \mu_n), \quad (2.2)$$

and is a variance normalized measure of distance between a new sample and the center of a distribution.

Using a connected component analysis (step $3C_J$) the spatially disconnected segments are divided into multiple connected segments. Smaller segments are then merged with the largest neighboring segment using region merging (step $3D_J$). Then each segment is assumed to belong to one of the three classes, cast shadow, self shadow or part of the object darker than the background image. To determine which of the segments are cast shadows, the textures of the segments are compared to the texture of the corresponding background regions. Because the illumination in a cast shadow can be very different from the background the gradient direction is used. It is well suited as an illumination invariant feature (step $3E_J$):

$$\theta = \arctan \frac{f_y}{f_x}, \quad (2.3)$$

where θ is the gradient direction and f_y and f_x are the vertical and horizontal derivatives respectively. If the correlation is more than 0.75, the region is considered a cast shadow. Otherwise it is either self shadow or dark part of the object (step $3F_J$). Figure 2.3 shows examples of shadow elimination using Javed's method [21].

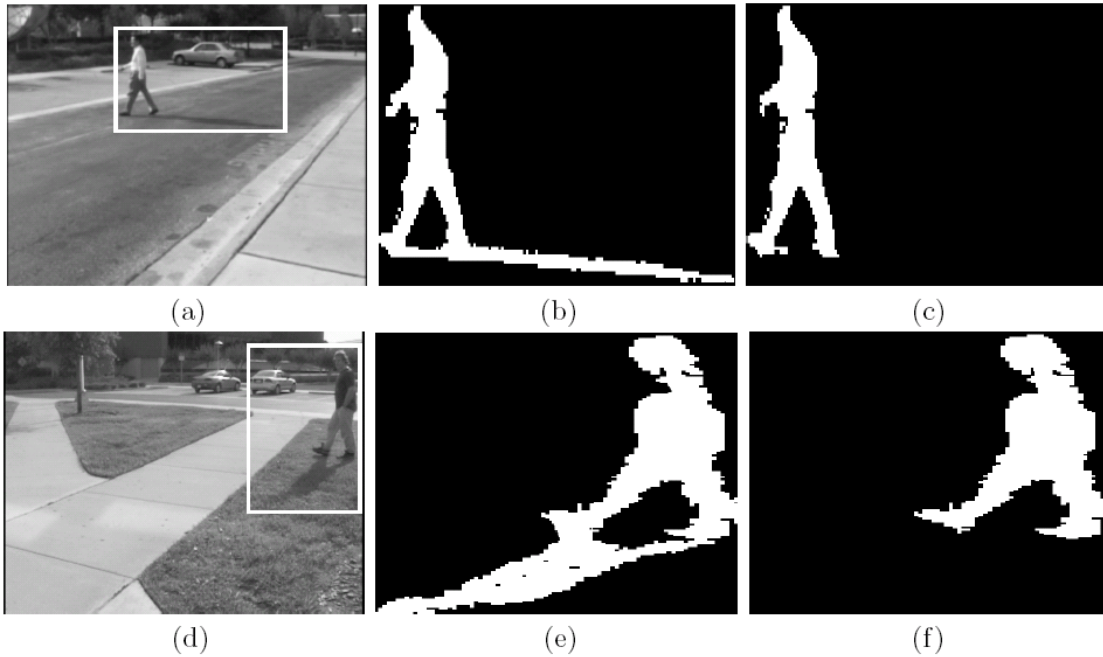


Figure 2.3: Shadow elimination using Javed's method [21]. (a) and (d) show the bounding box of the object including shadow. (b) and (e) show the background subtraction results for the bounding box. (c) and (f) show object mask after shadow elimination.

When tested, the shadow algorithm was only applied when more than 30% of an object was darker than the background image. It is reported to perform well on 70% of the frames with significant shadows in which the object was visible. In 25% of the frames it did not remove the shadows and in 5% of the frames parts of the true object were removed. The majority of errors are reported to be caused by large self shadows on objects and due to failure of the segmentation procedure to divide cast shadows and self shadows into different regions.

The major strength of Javed’s method when compared to Hsieh’s method, is that no spatial assumptions are made, making it suitable for handling objects of any shape. This is very important in reference to the interest of the DDRE. Only a few assumptions are made: That the shadows are not strong enough to completely wipe out any details of the underlying surface, and that the underlying surface is not smooth, i.e. it contains gradient features. It is not reported if the failures could be due to the texture of the object and the texture of the background being too alike. This could be a problem when using only the gradient direction to distinguish cast shadows from the other classes. Furthermore the segmentation using K-means with a fixed covariance and the correlation measure are sensitive to how thresholds are chosen. Still Javed’s method is chosen as a reference in this thesis because of its general applicability.

2.5 Physics-Based Shadow Removal

2.5.1 Nadimi et al.

In [26] Nadimi et al. use a Gaussian mixture model for background modelling. Afterwards they apply a number of steps in a physics-based shadow detection algorithm. Each step exploits various features based on the following assumptions:

- Background does not contain moving objects.
- Surface reflectance due to sky illumination is shifted toward the blue.
- Pixels in shadow regions are illuminated by the sky only.
- Inter-reflections due to nearby objects are negligible
- Cast shadow pixels have the same reflectance properties as the background.
- Shadow pixels are darker than their reference in all three channels (R,G,B).
- Background surfaces are generally matte and different from moving object surfaces.

5 steps are applied in the shadow detection:

1. *Initial shadow pixel reduction.* As in Javed’s method, only pixels darker than their background (in R,G and B) are considered shadow candidates.
2. *Blue ratio test.* Shadows are assumed to be illuminated by a blue sky only. Reflectance changes in R and G are assumed to be larger than reflectance change in B.
3. *Albedo ratio segmentation.* A measure combining ratios of differences between two neighboring pixels with ratios of differences between foreground and background pixels is defined and called the albedo ratio. It combines both spatial and temporal information and is used as a measure of similarity between two neighboring pixels. After the albedo ratio segmentation connected component analysis and region merging are done.

4. *Ambient reflection correction.* Reflection due to the sky is considered an additive component. Foreground pixel values are now subtracted from the background pixels, to remove the reflection due to ambient light.
5. *Body color segmentation.* A dichromatic reflection model is used to estimate the average body color of each surface. The total radiance of the reflected light is described as a sum of diffuse (body) and specular (surface) reflections. Applying some physical assumptions, the body color is estimated using singular value decomposition (SVD). These estimates are compared to initial estimations from step 1 and the regions within an acceptable threshold are classified as shadow regions.

No average performance measure is reported, but it is indicated that the algorithm performs very well. Four examples in the article classify 66-81% of the shadow pixels correctly, and the rest are missed. Less than 0.01% pixels are false positives. The frames used for testing include varying background and daylight. One difficult case is noted, when a part of the object in self shadow has similar color as the background. In this case the self shadow is classified as cast shadow. Exploiting the higher luminance of the self shadow compared to the cast shadow is suggested to solve this problem. The blue ratio test is sensitive to the sensor and the background color saturation. It is therefore bypassed when background is saturated or the sky is cloudy. Also the body color segmentation has problems with puddles etc. since the dichromatic reflection model does not handle highly reflective surfaces very well.

Nadimi's method also has the advantage of making no spatial assumptions. But with a number of deterministic physics-based segmentation steps several thresholds need to be optimized. Some of the steps seem more relevant than others, especially step 1, which is also used by Javed (cf. sec. 2.4). But also the albedo ratio segmentation, combining temporal and spatial features when doing the connected component analysis, could be of relevance.

2.5.2 Finlayson et al.

Finlayson et al. [13, 15] derive a grayscale illumination invariant shadow free image from a single RGB image taken with a color calibrated camera. The gradient of the shadow free image is then compared to the gradient of the original image and the edges of the shadows are removed. Re-integrating the gradient image reveals a shadow free full color image. Several physics-based assumptions are made making the method susceptible to noise.

A flowchart of how to apply Finlayson's suggested method for shadow removal is depicted in figure 2.4, corresponding to step 3 in figure 2.1. Only steps $3B_F, 3C_F$ and $3D_F$ are addressed by Finlayson [13, 15].

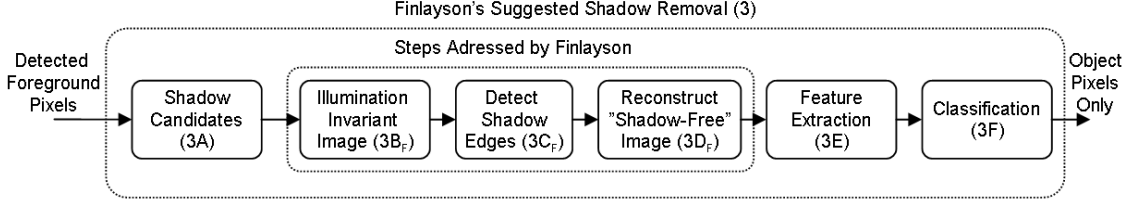


Figure 2.4: Flowchart of shadow removal as suggested by Finlayson. Corresponds to step 3 in figure 2.1.

Deriving an Illumination Invariant Image

Barrow et al. [1] originally proposed a decomposition of an image $I(x,y)$ into a product of two "intrinsic" images, a reflectance image $R(x,y)$ and an illumination image $L(x,y)$:

$$I(x, y) = R(x, y)L(x, y). \quad (2.4)$$

These intrinsic images were derived to ease typical segmentation tasks. But the derivation is an ill-posed problem with twice as many unknowns as equations. Weiss [37] suggests a statistical method for retrieving one reflectance image from an image sequence of a scene with constant reflectance and changing illumination. Finlayson et al. take the idea of intrinsic images a step further and propose a method for deriving an illumination invariant image from a single image and a color calibrated camera [13, 15].

The color of a pixel in an image depends on the illumination, the surface reflection and the camera sensors. Denoting the spectral power distribution of the illumination $E(\lambda)$, the surface spectral reflection function $S(\lambda)$, and the camera sensor sensitivity functions $Q_k(\lambda)$ ($k = R, G, B$), the RGB color ρ_k at a pixel can be described as an integral over the visible wavelengths λ :

$$\rho_k = \int E(\lambda)S(\lambda)Q_k(\lambda)d\lambda \quad , \quad k = \{R, G, B\}. \quad (2.5)$$

This description assumes no shading and distant lighting and camera placement. If the camera sensitivity functions $Q_k(\lambda)$ are furthermore assumed to be narrow-band, they can be modelled by Dirac delta functions $Q_k(\lambda) = q_k\delta(\lambda - \lambda_k)$, where q_k is the strength of the sensor. Substituting this into (2.5) reveals:

$$\rho_k = E(\lambda)S(\lambda)q_k \quad , \quad k = \{R, G, B\}. \quad (2.6)$$

Lighting is approximated using Planck's law:

$$E(\lambda, T) = Ic_1\lambda^{-5} \left(e^{\frac{c_2}{T\lambda}} - 1 \right)^{-1}, \quad (2.7)$$

where I is the intensity of the incident light, T is the color temperature, and c_1 and c_2 are equal to $3.74183 \cdot 10^{-16} Wm^2$ and $1.4388 \cdot 10^{-2} Km$ respectively. Planck's law is valid for objects with black-body radiation, also called Planckian lights. When plotted in a chromaticity diagram, Planckian lights describe a Planckian locus for varying temperatures. Daylight is very near to the Planckian locus since the sun can be described as

a black-body object (100% absorption for all λ). The illumination temperature of the sun is in the range from 2500K to 10000K (red through white to blue). For the visible spectrum (400-700nm) the exponential term of (2.7) is somewhat larger than 1. This is Wien's approximation [21]:

$$E(\lambda, T) \simeq I_{c1} \lambda^{-5} e^{-\frac{c_2}{T\lambda}}. \quad (2.8)$$

If the surface is Lambertian (perfectly diffuse reflection) shading can be modelled as the cosine of the angle between the incident light \mathbf{a} and the surface normal \mathbf{n} [9, 4]. This reveals the following narrow-band sensor response equation:

$$\rho_k = (\mathbf{a} \cdot \mathbf{n}) I_{c1} \lambda^{-5} e^{-\frac{c_2}{T\lambda}} S(\lambda) q_k, \quad k = \{R, G, B\}. \quad (2.9)$$

Defining band-ratio chromaticities r_k remove intensity and shading variables:

$$r_k = \frac{\rho_k}{\rho_G}, \quad k = \{R, B\}. \quad (2.10)$$

Taking the natural logarithm (\ln) of (2.10) isolates the temperature:

$$r'_k \equiv \ln(r_k) = \ln(s_k/s_G) + (e_k - e_G)/T, \quad k = \{R, B\}, \quad (2.11)$$

$$s_k = \lambda^{-5} S(\lambda) q_k, \quad (2.12)$$

$$e_k = -c_2/\lambda_k. \quad (2.13)$$

For every pixel the vector $(r'_R, r'_B)^T$ is formed as a constant vector plus a vector $(e_R - e_G, e_B - e_G)^T$ times the inverse color temperature. As the color temperature changes, pixel values are constrained to a straight line in 2D log-chromaticity space, since (2.11) is the equation for a line. By projecting the 2D color into the direction orthogonal to the vector $(e_R - e_G, e_B - e_G)^T$, the pixel value only depends on the surface reflectance and not temperature hence illumination (cf. figure 2.6):

$$\begin{aligned} r'_R - \frac{e_R - e_G}{e_B - e_G} r'_B &= \ln(s_R/s_G) - \frac{e_R - e_G}{e_B - e_G} \ln(s_B/s_G), \\ &= f(s_R, s_G, s_B). \end{aligned} \quad (2.14)$$

Applying (2.14) to all pixels reveals the illumination invariant image $gs(x, y)$:

$$gs(x, y) = a_1 r'_R(x, y) + a_2 r'_B(x, y), \quad (2.15)$$

where the constant vector $\mathbf{a} = (a_1, a_2)^T$ is orthogonal to $(e_R - e_G, e_B - e_G)^T$, determined by the camera sensitivity functions only (2.14)(2.13), and scaled to unit length.

$$\begin{aligned} \mathbf{a} &= \frac{\mathbf{a}'}{\|\mathbf{a}'\|}, \\ \mathbf{a}' &= \begin{pmatrix} 1 \\ -\frac{e_R - e_G}{e_B - e_G} \end{pmatrix}. \end{aligned} \quad (2.16)$$

This derivation corresponds to step $3B_F$ in figure 2.4. Figure 2.5(b) shows an example of an illumination invariant grayscale image, where edges due to shadows are not visible. Figure 2.5(a) and 2.5(c) show the original image, and the normal grayscale image.

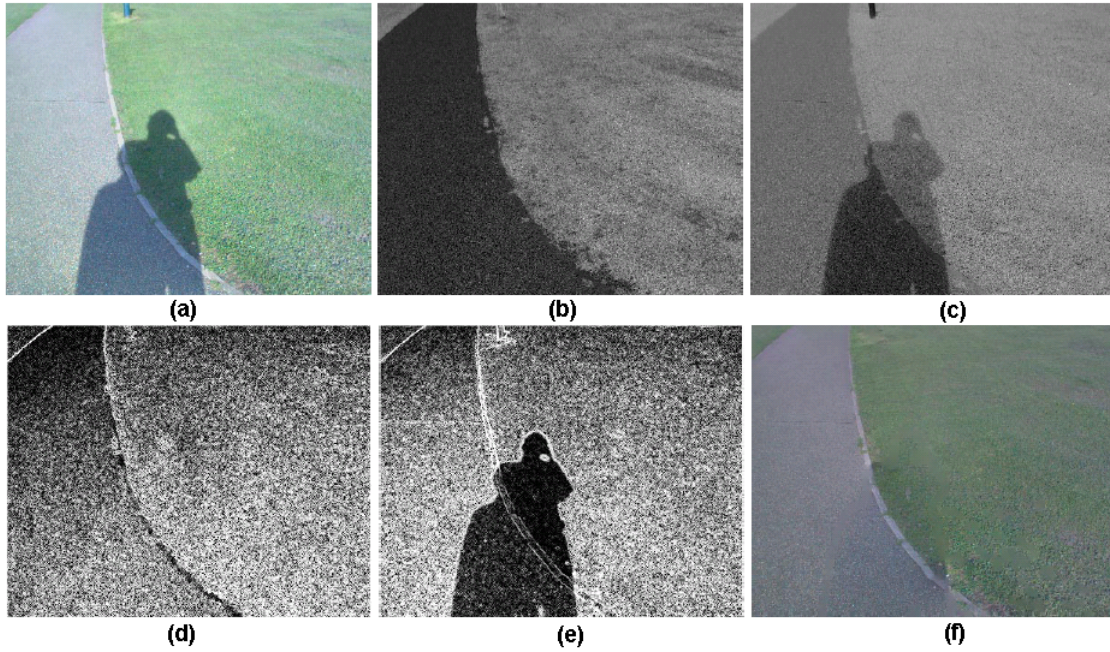


Figure 2.5: *Finlayson's approach to shadow removal [15]. (a): Original image. (b) Illumination invariant grayscale image. (c): Grayscale of original image. (d): Edge map for invariant image. (e): Edge map for non-invariant image. (f): Recovered shadow-free image.*

If the sensor functions of the camera, and thereby λ_k of (2.13), are unknown, [13] and [15] outline a procedure for camera color calibration. The invariant direction is estimated by comparing a number of images taken during the day with changing illumination, cf. figure 2.6. Daylight is assumed to be Planckian with varying temperature. Each image contains different standard color patches from the Macbeth Color Chart (cf. App. A).

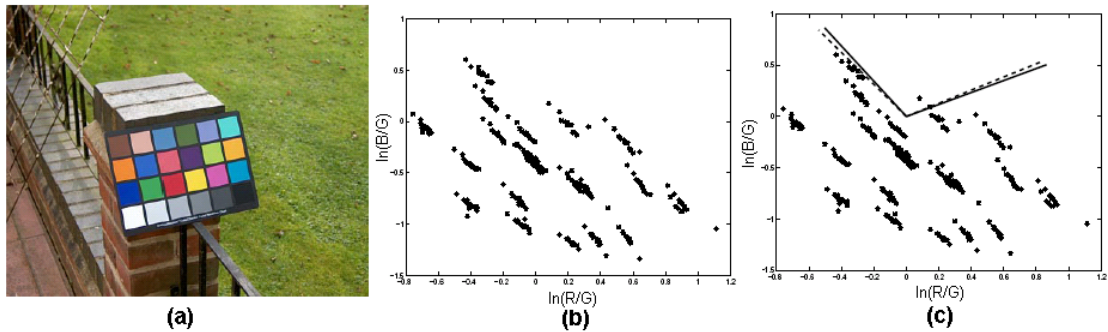


Figure 2.6: *Finlayson's color calibration of the HP912 digital still camera [15]. (a): Test image of the Macbeth Color Chart. (b): Chromaticities of 24 color patches from 14 images taken during the day. (c): Invariant directions recovered using camera sensors (solid line) and using the image sequence (dashed line).*

The various assumptions made to derive the illumination invariant image, are of course limitations to the model in way of describing realistic scenes. Each assumption affects the model in the following ways:

1. Narrowband sensors are assumed in order to be modelled by a Dirac delta function. Worthey et. al [38] show that (2.6) holds for sensitivity widths of around 100nm. If the widths exceed 300nm the model fails completely. A standard digital camera can have quite broad sensor functions, which leads to a non-linear function in the 2D color space. This leads to images being not totally illumination invariant but with shadows still more or less attenuated [15]. A discussion of how to sharpen the spectral sensitivity functions, and optimize this sharpening, can be found in [12, 8, 4, 9]. It is done by applying a linear transform of the sensor functions, resulting in an improved illumination invariant image. This way the approximation of (2.6) is improved.
2. Planckian light is assumed in order to use Planck's law (2.7). Daylight in the visible spectrum approximates Planckian light quite well [13, 15]. In [4] the approximation is studied in detail. While daylight is a good approximation of Planckian light, fluorescent light is a very poor approximation, due to highly localized emission spikes. Therefore the model will tend to decreased performance on images taken under indoor fluorescent light. Most types of light though, are placed near to the Planckian locus in the chromaticity plot, even additive combinations of Planckian light [13].
3. Lambertian surfaces are totally diffuse. This means that no matter what the angle of the viewer is, the brightness of the surface is constant. Specularities from non-diffuse surfaces can be incorrectly characterized in the illumination invariant image. In [14] Finlayson et. al outline a way of including specularities using a 4-sensor camera.

If the intensity in the shadow regions is too dark (near zero), the intensity image cannot be correctly formed. Furthermore, only for colored surfaces are the illumination edges eliminated while the reflectance edges are kept. For white, gray or black surfaces, the reflectance edges will also be eliminated, since they are all neutral in color [15, 4]. Finally it should be noted that the derivation of the illumination invariant image is only valid for linear images, that is, if e.g. gamma-correction is done (2.17), the invariant direction in the 2D log color space will change:

$$\begin{aligned} \rho_k &\rightarrow \gamma(\rho_k), \\ r'_k &\rightarrow \gamma \log(s_k/s_G) + \gamma(e_k - e_G)/T \quad , \quad k = \{R, B\}. \end{aligned} \quad (2.17)$$

If the invariant direction is found using camera calibration, it will automatically contain this change, but that is not the case if the sensor functions are used directly.

Despite these limitations and drawbacks, the illumination invariant image could be quite useful in conjunction with shadow removal, because the shadows are still attenuated in the worst case scenario.

Reconstructing an RGB-Image Without Shadows

The shadow edges are detected by comparing the gradient of each channel in the original log image, $\nabla \rho'(x, y)$, with the gradient of the illumination invariant image, $\nabla g_s(x, y)$,

cf. figure 2.5(d) and 2.5(e). The idea is that if the gradient in $\rho'(x, y)$ is high, while it is low in $gs(x, y)$, the edge is most likely to be a shadow edge. The following threshold function reveals a gradient image of the log response where gradients due to shadows are eliminated (set to zero). This is step $3C_F$ in figure 2.4:

$$S(\nabla\rho'(x, y), \nabla gs(x, y)) = \begin{cases} 0 & \text{if } \|\nabla\rho'(x, y)\| > t_1 \\ & \text{and } \|\nabla gs(x, y)\| < t_2 \\ \nabla\rho'(x, y) & \text{otherwise,} \end{cases} \quad (2.18)$$

where t_1 and t_2 are context dependent thresholds. By integrating S a log response image without shadows is recovered. This corresponds to solving the following Poisson equation (step $3D_F$ in figure 2.4):

$$\nabla^2 q'(x, y) = \nabla \cdot S(\nabla\rho'(x, y), \nabla gs(x, y)), \quad (2.19)$$

where ∇^2 is the Laplacian and q' is the log of the image without shadows. The gradient image of S equals the Laplacian of q' for each color band. Assuming Neumann boundary conditions ($\nabla q' = 0$ for boundary normals), q' can be solved uniquely up to an additive constant. When exponentiating q' to arrive at the shadow free image q the unknown constant becomes multiplicative. For the colors to appear "realistic" in each band, the mean of the top 1-percentile of pixels is mapped to maximum of the RGB image. In this way the unknown constants are fixed, and a shadow free image q is derived, cf. figure 2.5(f).

The major drawback of this method is reported to be defining the shadow edges. It turns out that using a robust edge detection algorithm (e.g. Canny or SUSAN [15]) and setting the thresholds are crucial factors. Furthermore a morphological opening is applied on the binary edge map to thicken the shadow edges and thereby improve the suppression of shadow gradients before the re-integration step.

Despite all of the assumptions and difficulties reported the method shows good results on the images shown in [13, 15, 4]. It should be noted that the gradient images and thresholds are very context dependent. However, even when the method performs poorly it still attenuates the shadows. This is often the case for shadows with diffuse edges. Therefore the method is interesting in conjunction with surveillance tasks, where the artifacts introduced by the imperfect shadow edge detection and the re-integration are not crucial. The main concern of [15] is delivering photo-quality images, not using the method in automated surveillance tasks. The Finlayson approach can be applied to surveillance tasks in several ways.

1. The shadow free RGB image could be used as described above. The drawback being robust detection of shadow edges to obtain an RGB image without artifacts.
2. The illumination invariant grayscale image can be used in the background subtraction step. This would remove/attenuate all shadows (static and dynamic) revealing a shadow free image for moving object detection. The drawback being the loss of color information.

3. The shadow free image could also be used on the foreground object only, to improve the classification of cast shadows. This way updating the background model is not affected by artifacts introduced by Finlayson’s method.

2.6 Comparison

Of the two statistical approaches examined, Javed’s method [21] is chosen as a reference method to be implemented. This is primarily based on the fact it does not make any spatial assumptions of the composition and posture of the object, contrary to Hsieh’s method [20]. This is important because the DDRE require a robust method that applies under general conditions.

Javed uses the moving object detection as origin and attempts to model the pixels statistically. This approach has the advantage of only focusing on moving object pixels that could cause problems for the later shape analysis. However the method has a number of context dependent parameters that need to be set during segmentation and classification:

- Same fixed variance and fixed Mahalanobis for all Gaussians when modelling foreground pixels using K-means, which influence the number of Gaussians needed to model all the foreground pixels. The more Gaussians, the less pixels are assigned to each region and the method is thereby more susceptible to noisy areas e.g. near edges.
- Regions less than a certain size are merged with their largest neighbor. The size criteria should be set in a way so that small noisy regions near edges are merged with, and thereby classified as, their largest neighbor without affecting the correlation measure too much. Choosing the size criteria depends on how many pixels the object consists of, since large objects tend to have larger areas of noisy pixels along the edges of the regions.
- The correlation measure is also context dependent. Setting a global threshold to determine if a region is a cast shadow or an object region is a non-trivial task. Too high a threshold results in too many cast shadow regions classified as object regions. Too low a threshold results in the opposite.

How these parameters are set is not mentioned in [21]. The author contacted Javed [22] to confirm that the parameters indeed are optimized to best performance for the specific data set. In optimization terms this corresponds to using the training error and not the test error to evaluate the performance of a specific algorithm. This is a problem when the goal is to design a system that automatically adapts to the scene.

The physics-based approach by Nadimi et al. [26] applies several deterministic segmentation steps, all of which need to be optimized. Instead focus is on the quite elegant physics-based method suggested by Finlayson [15].

Finlayson derives an RGB image invariant to illumination, i.e. a shadow free image. Since the approach has not yet been applied in a surveillance application, Finlayson only address steps $3B_F, 3C_F$ and $3D_F$ of figure 2.4, the method is chosen as an additional

method for implementation in this thesis. The limiting assumptions are that it only works for diffuse (Lambertian) surfaces, the lighting should be close to Planckian, and a narrowband camera should be used, either calibrated or with known sensitivity functions. The major drawback is that the suggested thresholding of gradients is far from robust. Even when these limitations influence the resulting image, the shadows are reported to be somewhat attenuated.

Common strengths of both Javed's and Finlayson's methods are that no spatial assumptions are made, assuring their use is not limited to certain object types, e.g. humans in upright posture. However, both methods assume that shadows are not so dark that the underlying texture is completely attenuated.

2.7 Summary

A range of dedicated computer vision systems for various surveillance tasks have been designed over the years. Systems for general outdoor surveillance face the most difficult problems because of complex non-stationary scenes due to varying weather conditions etc. The W^4 -system is chosen as a key reference by the DDRE, since it describes an implemented system that relatively successfully tracks moving objects, classifies them, and analyzes their behavior. W^4 reports problems with analyzing moving objects containing cast shadows. In [18] Hansen implements an improved version of Hsieh's method for shadow removal [20]. For instance it assumes that objects are persons in standing posture, and that the cast shadow always touches their feet. The intention of this thesis is to avoid any spatial assumptions of the composition of objects in the detection of their shadows.

Several state-of-the-art methods for shadow removal are described, with emphasis on a statistical approach by Javed [21] and an elegant physics-based approach by Finlayson [15].

The prior method applies unsupervised color segmentation of object pixels that are darker than the reference image. This is followed by a connected component analysis and region merging. Finally each region is compared to the background image where the correlation between their gradient directions is used to classify the region to either object or cast shadow. The method assumes that the shadows are significant enough to completely wipe out any gradient information, and that the underlying surface is not smooth. Several parameters are context dependent. It performs well on 70% of images with significant shadows in which the object was visible. In 25% of the cases it did not remove the shadows and in 5% of the cases parts of the true object were removed.

Finlayson's method has not previously been applied in surveillance tasks. In this thesis the method will be examined for use in such an application. The key idea of the method is the derivation of an illumination invariant grayscale image which is derived from a single RGB-image and is used to detect edges from shadows. The edge of these shadows are then set to zero in the gradient image of the original RGB-image, and an RGB-image without shadows is derived from this altered gradient image. The method requires knowledge of the camera sensor functions or alternatively a color calibration can be performed. In the derivation of the illumination invariant image, the lighting is assumed to be Planckian so

it can be modelled by Plancks law. Furthermore narrowband camera sensors and diffuse (Lambertian) surfaces are assumed. These assumptions are reportedly not very strict. The major difficulty reported is how to automatically determine which edges are due to shadows and which are not.

Neither of the two emphasized methods make use of any spatial assumptions. They do however have some drawbacks, which will be examined in detail in chapter 5.

Chapter 3

Data Acquisition

This chapter describes the digital video camera used for obtaining video sequences, and the data sets used for calibration, optimization and validation of the methods. This provides, in detail, an overview of how data is obtained and used, and is an important chapter to understanding how the results should be interpreted.

3.1 Camera

The camera which DDRE have acquired for their surveillance projects is a SVS-VISTEK 204CFCL state-of-the-art industry camera. The resolution is 1024x768 pixels 10-bit and the maximum frame rate specified is 39 frames per second (fps). It is connected to a frame grabber in a PC by a "CameraLink" connector. Each movie sequence is saved uncompressed in the *avi*-format. This makes saving data to the hard disk drive on the PC the bottleneck of the data acquisition. A maximum frame rate of 20 fps 8-bit is currently possible. No processing of data is done prior to saving the sequences ensuring linear images as required when deriving Finlayson's illumination invariant image, cf. section 2.5.2. The camera has a single CCD with an optical Bayer filter [32] in front of it to produce RGB-colors. A Bayer filter produces horizontal lines of repeating red and green pixels alternating with lines of repeating green and blue pixels as shown in figure 3.1. There are twice as many green pixels as red and blue pixels because the luminance response curve of the human eye is more sensitive to green and therefore reveals images that seem more "natural" to the human eye [32].

Using a Bayer filter requires processing the raw data to obtain an RGB-image. This is done by filtering the raw image with some simple masks depending on the pixel type and which color is to be extracted. Table 3.1 shows the masks used. This is a standard non-adaptive method for recovering a full RGB-image. Due to irregularities between the green pixels $G1$ and $G2$ in the camera used, the green color G in the green pixels $G1$ and $G2$ is averaged. According to the manufacturer these irregularities are most likely due to crosstalk. Filtering images introduces artifacts along edges where false colors may occur. Alternatively more advanced methods for obtaining RGB-images could be used [32] but these artifacts are not assessed to be critical in the present surveillance application.

Applying the masks of table 3.1 results in the spectral response curves of figure 3.2

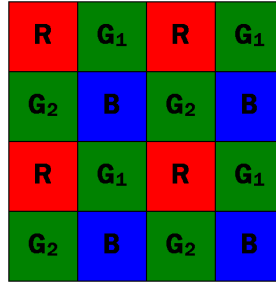


Figure 3.1: Pixel pattern produced by an optical Bayer filter [32]. Half of the pixels are green due to the human eye being more sensitive to green.

	R	G_1	G_2	B
R	1	$\frac{1}{2}$ 0 $\frac{1}{2}$	$\frac{1}{2}$ 0 $\frac{1}{2}$	$\frac{1}{4}$ 0 $\frac{1}{4}$ 0 0 0 $\frac{1}{4}$ 0 $\frac{1}{4}$
G	$\frac{0}{4}$ $\frac{1}{4}$ 0 $\frac{1}{4}$ 0 $\frac{1}{4}$ 0 $\frac{1}{4}$ 0	$\frac{1}{8}$ 0 $\frac{1}{8}$ 0 $\frac{1}{2}$ 0 $\frac{1}{8}$ 0 $\frac{1}{8}$	$\frac{1}{8}$ 0 $\frac{1}{8}$ 0 $\frac{1}{2}$ 0 $\frac{1}{8}$ 0 $\frac{1}{8}$	$\frac{0}{4}$ $\frac{1}{4}$ 0 $\frac{1}{4}$ 0 $\frac{1}{4}$ 0 $\frac{1}{4}$ 0
B	$\frac{1}{4}$ 0 $\frac{1}{4}$ 0 0 0 $\frac{1}{4}$ 0 $\frac{1}{4}$	$\frac{1}{2}$ 0 $\frac{1}{2}$	$\frac{1}{2}$ 0 $\frac{1}{2}$	1

Table 3.1: Masks applied to the raw Bayer image of figure 3.1 for obtaining an RGB-image. Rows are the R , G and B masks applied to each pixel as a function of the pixel type in the Bayer filter (columns).

as specified by the manufacturer [36]. The response is measured relative to green with center frequencies λ_k as shown in table 3.2.

k	R	G	B
$\lambda_k [nm]$	613	540	462

Table 3.2: Center frequencies in nm of spectral response curves shown in figure 3.2.

During the early stage of recording sequences with the camera there was a clear shift towards reddish colors. The lack of an optical filter suppressing the infrared part of the spectrum (IR-cut filter) was causing these false colors because of increasing CCD-sensitivity in the IR part of the spectrum ($> 700nm$) [3]. Using a standard IR-cut filter (B+W 489), cutting frequencies above 780nm, solved the problem. Figure 3.3 shows an example of the same image taken before and after the use of an IR-cut filter. The image to the right seems a bit greenish because no white balance is applied.

For all sequences the shutter speed is fixed and the aperture is set manually (F5.6-F22)

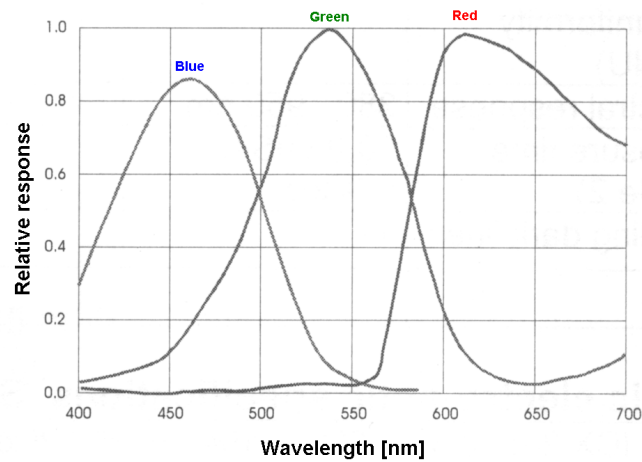


Figure 3.2: *Spectral response curves of the SVS-204CFCL digital camera [36].*



Figure 3.3: *Image acquired before (left) and after (right) use of an IR-cut filter (780nm). The CCD is sensitive in the infrared part of the spectrum revealing reddish colors.*

assuring appropriate exposure. Focus is also set manually to obtain maximum sharpness in the specific scene.

3.2 Data Sets

The data sets are carefully chosen to both resemble an actual surveillance application and to represent the typical problem of cast shadows from moving objects. Figure 3.3 shows the scene from which the majority of sequences are recorded. It contains many of the typical problems a surveillance application will face: A non-stationary background due to wind and changing illumination, occlusions, objects to be included in the background model (e.g. a parked car) and, of course, shadows. The typical moving objects are vehicles, people and bicycles.

Each sequence is converted into bitmap images (*bmp*) using the color recovery filter masks of table 3.1. Each object is used only once to avoid stochastic dependence between

samples when validating the results statistically. Therefore only every 4th frame is extracted, making it possible to focus on frames where cast shadows are an actual problem. A slightly improved version of Hansen’s implementation [18] of the Elgammal’s kernel-based background model, as described in section 5.1, is used for segmenting foreground objects. For each frame a binary foreground mask is extracted so the foreground extraction only needs to be done once. When choosing which objects to use for optimization and validation only the actual object of interest in the foreground mask is used. Segments due to noise or objects of no interest are suppressed manually. This of course would be done automatically in an online surveillance application.

Scenes illuminated by direct sunlight contain the most distinct shadows giving rise to the major cast shadows in moving objects. Because of the limited dynamic range of the camera CCD (8 bit in each color band) there is a trade-off between saturation of bright regions and completely wiping out the texture of dark regions (shadows). From a shadow removal point-of-view, experience shows that in direct sunlight it is better to saturate very bright regions to retain texture in shadows. This reasoning is not necessarily valid for other parts of a surveillance system. Therefore exploiting all 10 bits dynamic range should be examined in future work.

Appendix B contains the 90 foreground objects that constitute the total data set. Each of them are specified by a sequence name, a frame number, and an object number. The total data set is split into a training set of 18 foreground objects (20%), used for optimizing the methods (chapter 5), and a test set of 72 foreground objects (80%) used for validation (chapter 6). Only 20% of the data set is used for training, since the manual optimization is very time consuming, and because the size of the test set influences the statistical support of the results in the validation. Optimizing the parameters of the models is done manually. Numerical optimization algorithms would be even more time consuming to implement, but they could be useful to fine tune the parameters if a larger data set was available. This is beyond the scope of the present thesis. 5 of the 18 foreground objects used for training are shown in figure 3.4. Each of them are identified by the name of the video sequence, a frame number and an object number. It should be noted that all video sequences are named "Test...". This is not an indication of which data set they belong to. The training set is designed to be representative of the total data set, as shown in table 3.3. It consists of 9 vehicles, 4 persons and 5 bicycles/motorcycles, with variable object size and weather type.

Type of Foreground Object	Training Set	Test Set	Total Data Set
Vehicle	9 (50%)	37 (51%)	46
Person	4 (22%)	16 (22%)	20
Bicycle/Motorcycle	5 (28%)	19 (26%)	24
Total Size	18 (20%)	72 (80%)	90

Table 3.3: Data set split into training set and test set, cf. appendix B.

Validation of methods with statistical significance requires a large number of examples. Ideally, more than 72 examples should be obtained in order to produce statistical result

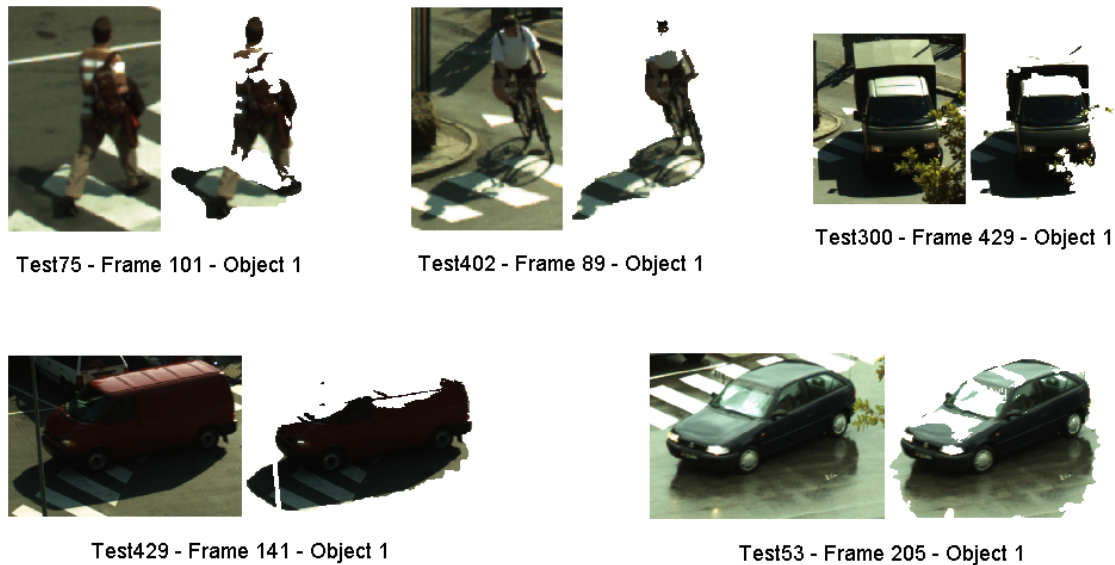


Figure 3.4: *Foreground objects used for training. Identified by the name of the video sequence, a frame number and an object number.*

that are highly significant. The size of the data set of course has natural limits due to the nature of a master thesis. Still, the 72 examples of the test set can produce reasonable statistical significance, as will be discussed in chapter 6. It can also be argued that 18 examples are far too few to train such complex methods. However, due to the very time consuming manual training, 18 examples will have to suffice in this thesis.

3.3 Summary

The camera used is a state-of-the-art industry digital video camera (SVS-204CFCL) with a resolution of 1024x768 pixels. The frame rate currently available is 20 fps., with a dynamic range of 8 bits, and with colors obtained through standard Bayer filtering. A typical scene for a surveillance application is chosen where the typical moving objects are vehicles, people and bicycles.

An improved version of Hansen's implementation [18] of a kernel-based background model is used to segment foreground objects. Only one frame of an object is used in the data set (App. B) to avoid stochastic dependence between samples. 18 foreground objects are used in a manual optimization of model parameters and 72 foreground objects are used for validation and comparison of methods.

Chapter 4

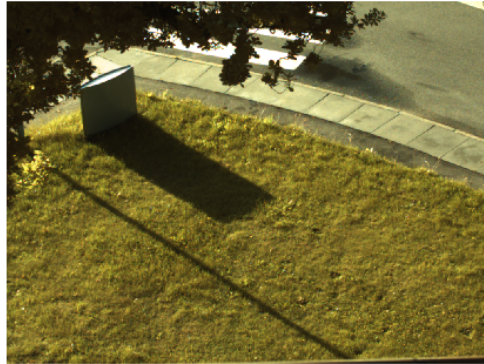
Finlayson's Approach Using a Video Camera

In order to apply Finlayson's approach for shadow removal using a digital video camera, the illumination invariant direction must be derived, and the effect of the assumptions in the model must be assessed. This is done in the present chapter, where Finlayson's ideas described in section 2.5 are examined using the SVS-204 camera described in section 3.1. In the overall framework for shadow removal, this corresponds to step $3B_F$ of figure 2.4. More specifically the illumination invariant direction in the log-chromaticity space, derived from the spectral response curves, is compared to the direction obtained from a camera calibration.

4.1 Spectral Sensor Functions

For the camera at hand the illumination invariant direction in the log-chromaticity space, described section 2.5.2, can be obtained if the center frequencies of the spectral response curves are known. In table 3.2 they are given as $[R, G, B] = [613, 540, 462]nm$. Inserting these values into equation (2.13) and (2.16) reveals the illumination invariant direction $a = [0.817, 0.576]^T = 35.2^\circ$ in the log-chromaticity space $[\ln(R/G), \ln(B/G)]^T$. The direction most variant to illumination then is -54.8° (blue line in figure 4.2). This is more or less the same general direction as the calibration of the camera in figure 2.6. Figure 4.1(a) and 4.1(b) show an image (SVS-204 camera) and its illumination invariant derived from the spectral sensor functions. The result is not optimal since there is a clear distinction between the grass region inside and outside the shadow. The texture inside the shadow has a diagonal-like pattern which is probably due to some correlated noise in the camera that degrades the signal-to-noise ratio in dark areas when the relative color (chromaticity) is computed. This does not originate from the Bayer filtering because the pattern is periodic over several pixels (≈ 10) and is dependent on the intensity, cf. figure 4.5. In the invariant image of figure 2.5(b) there is no distinction between shadow and non-shadow indicating that some of the model assumptions are not valid or that the invariant direction obtained from the spectral sensor functions of the camera is not

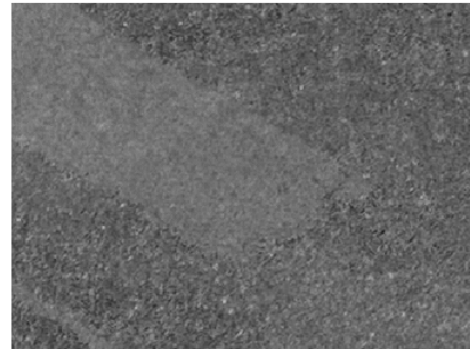
the optimal illumination invariant direction. To test if the latter is the case a camera calibration is done.



(a)



(b)



(c)

Figure 4.1: (a): Image acquired by the SVS-204 camera. (b): Illumination invariant of (a). Direction obtained using spectral sensor functions. (c): (b) zoomed.

4.2 Color Calibration

To ensure that the optimal illumination invariant direction corresponds to the direction derived in section 4.1 a color calibration is performed as described in [13][15] and section 2.5.2. The spectral sensor functions can change with ageing but that is not very likely for the present camera since it is relatively new. Still it is important to carry out a calibration in order to validate the optimal direction.

The idea is to vary only the temperature of a Planckian light source for a specific set of camera sensors and surface patches. The Macbeth Color Chart containing 24 matte color patches (Appendix A) is used in various types of daylight since it has been shown that daylight is a good approximation of Planckian light [13].

In the first experiment a number of sequences were recorded between 10am and 4pm

in all types of weather (sunny, cloudy, rainy). 24 images from these sequences were chosen in such a way that no bright color patches were saturated and the dark patches still gave a response. A Matlab routine was developed, cf. appendix E.20, to ease the annotation of the images. From each color patch, of every image, the mean of around 1000 pixels was used to approximate the RGB-values which were then plotted in the log-chromaticity plot. The principal direction of each color patch is determined as a least-squares fit of a line. Figure 4.2 shows the resulting plot.

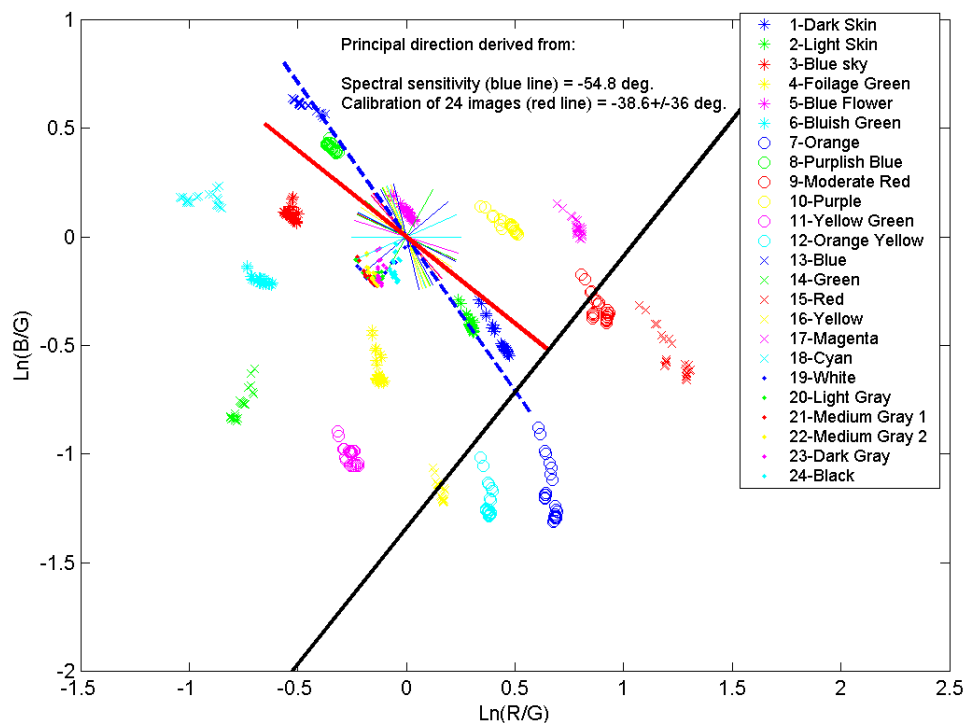


Figure 4.2: Calibration of the camera using 24 color patches from 24 images shot with varying types of daylight illumination (cloudy, sunny, rainy). Short lines are the general direction of each color patch. The blue dotted line is the most illumination-variant direction from the spectral sensors. The red and black lines are the most illumination-variant and -invariant directions derived from the calibration. Legend numbers refer to the color patches of the Macbeth Color Chart.

Each color patch has a separate marker and the principal direction of each color patch is plotted as the thin lines around the origin. The mean principal direction is the red line and the black line is the orthogonal direction (invariant direction). The blue dotted line is the direction obtained from the spectral sensor functions. The plot shows that there is hardly a principal direction supported by all color patches. The standard deviation (std.) of the mean direction is 36° . The green color (14-x) has a direction nearly orthogonal to most others and the bluish colors (3-* and 6-*) are also nearly orthogonal even though they are situated rather close in the log-chromaticity plot. The white color patch (19-•)

tends to saturate in some images since they are near to the origin. For an ideal white light source all types of colorless patches should be situated in the origin.

Since the samples are spread in a circular cloud for some of the colors, the principal direction is very sensitive to noise. This can be caused by three things: The temperature of the daylight not being varied enough as compared to random noise, the Planckian assumption not being valid for all of the images used or the narrowband assumption of the spectral sensor functions of the camera not being valid. The spectral sensor functions of the camera used in figure 2.6 [13] are not much different from those of the SVS-204 camera. The reason for the poor calibration must then lie in the Planckian assumption or the data. Several articles describe the calibration as very simple [13, 15, 4]: Just record some images during the day and estimate the principle direction. They do not discuss the effect of diffuse illumination in cloudy or rainy weather. The Planckian assumption might not be valid in these cases. In all calibration images of figure 4.2 the color chart was placed on the matte black roof of a building recording the sequences through an open window. This ensures that the major sources of illumination are direct sunlight or diffuse light from clouds.

To examine what causes the large variation of the principal directions in figure 4.2 a second experiment was performed. Again several sequences were shot from sunrise to sunset over a couple of days. In this experiment 12 images only with direct sunlight were used in the calibration procedure to eliminate possible errors due to non-Planckian diffuse light from clouds. The selection of which images to use to ensure a large variation in color temperature was done very carefully. Figure A.1 shows two calibration images with a clear difference in illumination caused by a varying color temperature. Such a large variation was not observed in the first calibration experiment. The calibration of the second experiment is presented in figure 4.3. Legends and labels are similar to those of figure 4.2.

The color patches in figure 4.3 do have a more clear principal direction. In particular two images stand out. They have a very yellowish illumination compared to the other images (cf. figure A.1, page 81). The mean principal direction from calibration is 50.6° with a std. of 6.9° . Even though the std. is small compared to the first experiment the direction derived from the spectral sensor functions is still within one std. of the mean direction of the calibration. This supports the direction found in section 4.1 and does not encourage finding a more optimal direction and thereby a "better" illumination invariant image. The std. is similar to the one reported in [13] and can largely be explained by spectral sensors not being ideal delta functions. When comparing the two experiments it shows that the variation in color temperature of the first experiment was not large enough compared to the deviations from the Planckian assumption that the daylight showed, resulting in a much larger std. Further experiments could reveal whether or not the Planckian assumption is appropriate for diffuse light, compared to direct sunlight. The narrowband assumption of the sensors can be further tested by making similar experiments with an artificial Planckian light source with variable color temperature. Under these conditions it is only the narrowband assumption that can cause deviations. These experiments are interesting for the DDRE in order to gain knowledge of their camera but

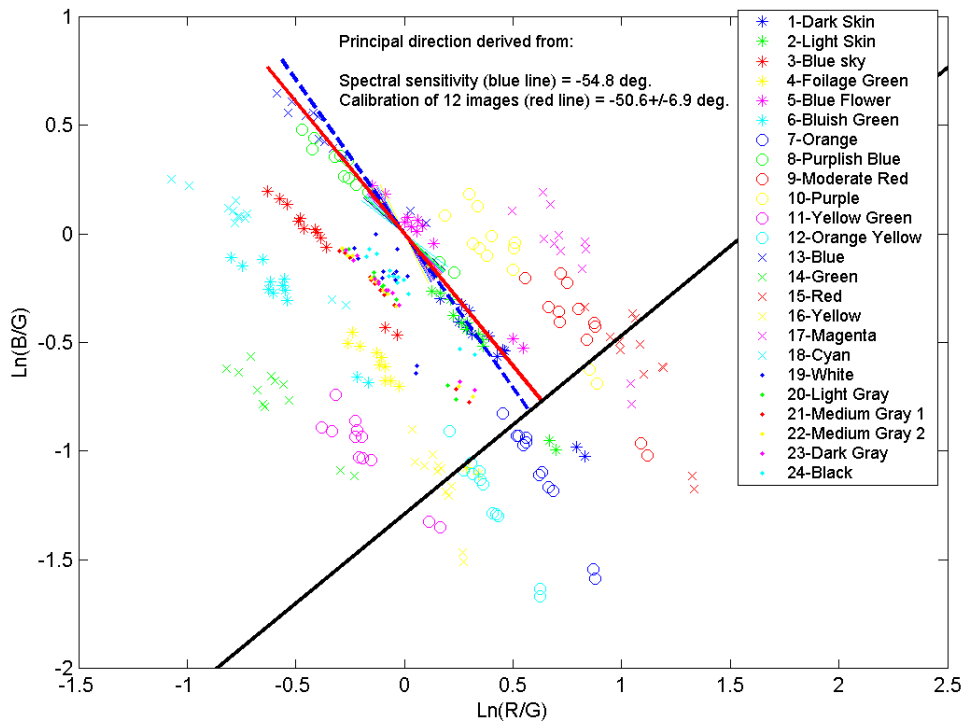


Figure 4.3: Calibration of the camera using 24 color patches from 12 images shot in direct sunlight at different times throughout the day. Short lines are the general direction of each color patch. The blue dotted line is the most illumination-variant direction from the spectral sensors. The red and black lines are the most illumination-variant and -invariant directions derived from the calibration. Legend numbers refer to the color patches of the Macbeth Color Chart.

are beyond the scope of this thesis and therefore not further pursued here.

Figure 4.4 shows the illumination invariant image of figure 4.1(a) using the projection obtained from the calibration ($-50.6^\circ + 90^\circ = 39.4^\circ$). There is no visible difference in the illumination invariant images obtained from the two projections. This is expected since the angle of projection is only changed by 4.2° . As a last experiment a number of illumination invariant images were computed by projecting from $[-90^\circ, 89^\circ]$ in steps of 1° to see how the texture of the different regions change. This supports the illumination invariant directions previously found as the optimal directions.

Even with the optimal illumination invariant projection there is a difference between regions inside and outside the shadow. The Planckian assumption has been shown to be valid for direct sunlight but it has not been possible to show that it is valid for diffuse daylight, e.g. regions in shadow, due to insignificant variation in color temperature. Since the examples in figure 2.5 and in [13, 15] are very promising it must be differences in the cameras that cause the inability to reproduce illumination invariant of the same quality as in [13, 15]. The spectral response curves shown in figure 3.2 can not be assumed delta

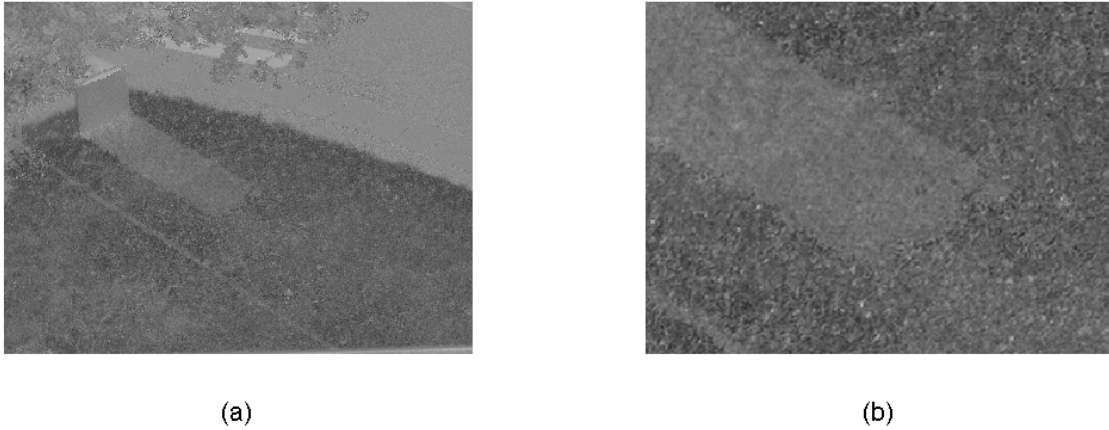


Figure 4.4: (a): *Illumination invariant of figure 4.1(a). Direction obtained from color calibration.* (b): *(a) zoomed.*

functions, but exactly how crucial this assumption is for the present camera is hard to say. In [13], Finlayson et al. show that if the response curves are sensitive in a range less than $100nm$ the assumption indeed is valid, and if it exceeds $300nm$ it is far from valid. For the SVS-204 camera the spectral response curves are sensitive within a range of approximately $150 - 200nm$ and the response of the green curve even seems to increase for wavelengths above $650nm$ (*red* \rightarrow *infrared*). This may be the reason why the illumination invariant image is of poor quality. Methods for sharpening the spectral response curves are outlined in [12, 13, 4]. These might enhance the quality of the illumination invariant image, but the methods are too extensive to be examined in this thesis.

The limited dynamic range of 8 bits can also cause poor illumination invariant images. Dark areas would be more susceptible to truncation errors because the chromaticity is a measure of relative color. For low values of green (denominator) quantization artifacts may occur. The HP912 camera used for obtaining figure 2.5(a) has the capability of producing 10- or 12 bits images in *raw*-format increasing the dynamic range substantially. [13] does not mention the dynamic range used when obtaining figure 2.5(a). It is not currently possible to exploit the full 10-bits color depth that the SVS-204 is specified for, but this should be examined by the DDRE at some point to determine the effect on the invariant images. A simple experiment was carried out where the aperture was varied in order to control the exposure. Figures 4.5(a)-(d) show zoomed images, and their illumination invariants, figures 4.5(e)-(h), with the invariant direction obtained from calibration, for different exposures.

It is evident that the exposure which reveals the best similarity between a region inside and outside of shadow is when the region outside is not saturated and the region inside is as bright as possible (fig. 4.5(c)/(g)). For images with less exposure (fig. 4.5(a)/(e)) the noisy diagonal pattern is also more evident suppressing the true texture. This pattern indicates some camera dependent artifacts in the dark areas of the illumination invariant images. As previously mentioned this could be due to correlated noise in the electronics of the camera e.g. $50Hz$ noise from the power supply.

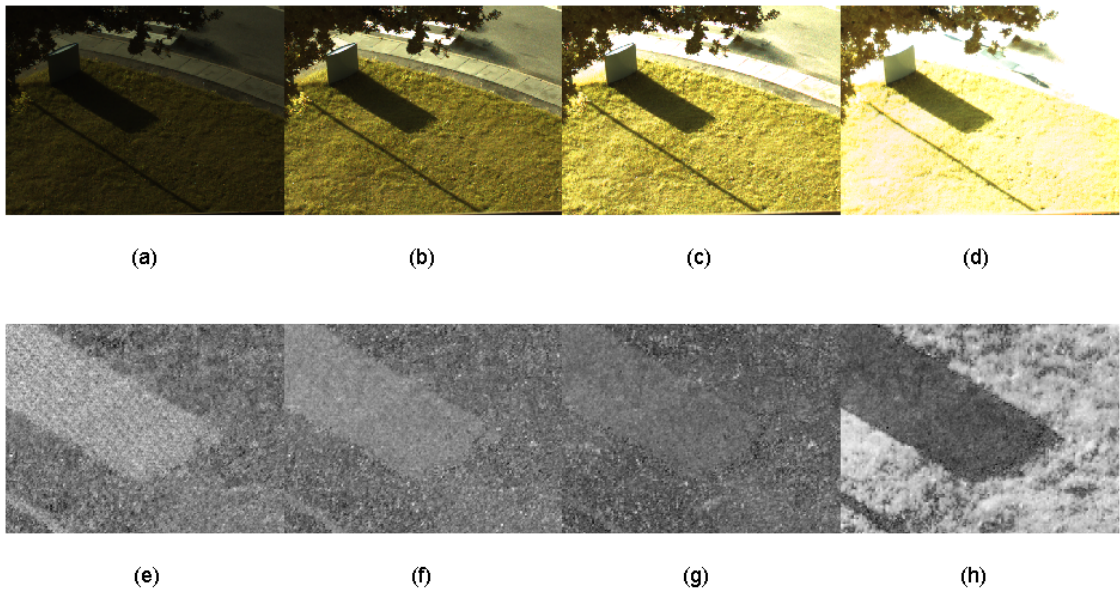


Figure 4.5: Images, (a)-(d), and their illumination invariants, (e)-(h) (zoomed), when varying exposure illustrating sensitivity to dynamic range.

Using the illumination invariant image for determining which edges are caused by shadows as suggested by Finlayson seems to be a very difficult task using the SVS-204 camera. Finlayson reports the edge detection to be very susceptible to noise even for the good-quality invariant image in figure 2.5(b). The invariant images in figures 4.1(b) and 4.4(a) contain clear edges along shadow borders. The use of the illumination invariant image is discussed in section 5.5.1.

4.3 Summary

Deriving the optimal illumination invariant direction in the log-chromaticity space is done using the spectral response curves of the SVS-204 digital video camera and by doing a color calibration with daylight of varying color temperatures. The camera calibration was found to be very sensitive to the weather type in the images used. It was not possible to do a calibration with images obtained in rainy or cloudy weather, because the color temperature did not vary enough compared to stochastic noise. Only when carefully choosing images illuminated by direct sunlight did the calibration reveal useful results. The direction derived from the spectral response curves only deviates 4.2° from the direction derived through calibration, a difference of less than 1 std. The illumination invariant images are constructed by projecting pixels onto the line with an angle of 39.4° .

Illumination invariant images obtained from the SVS-204 camera still have a clear distinction between regions inside and outside the shadow. This is not the case for the images presented in the literature (cf. figure 2.5). The assumption of daylight as Planckian light is well supported, though the effect of diffuse light in cloudy or rainy weather is not discussed [13]. Therefore it is the approximation of the camera sensors

as narrowband that does not apply. Furthermore it is found that the 8-bits color depth, currently available, severely influences the quality of the invariant images produced since the chromaticity of dark regions are influenced by noise of a certain pattern likely to originate from the camera electronics. All in all illumination invariant images using the SVS-204 camera are assessed not to be robust enough for use in an algorithm for robust detection of shadow edges.

Chapter 5

Implementation and Optimization

A brief overview of the implementation of the background model and the noise reduction used in the foreground detection is given in this chapter, corresponding to steps 1 and 2 in figure 2.1, page 7. This provides a basic understanding of how the moving foreground pixels are detected, which is the basis for the shadow handling algorithms, step 3 in figure 2.1.

Then several measures of performance are defined in a framework used for evaluation of the methods, leading towards a description of the actual implementation and optimization of Javed's method, denoted (J) (cf. figure 2.2 page 10), with respect to the training set. By analyzing the results of Javed's method on the training set, an improvement of the K-means color segmentation is suggested (denoted method I). Finally Javed's improved method is combined with some of Finlayson's ideas (cf. figure 2.4 page 14) for shadow removal in a new enhanced method for shadow removal (denoted method E).

Implementation is done in Matlab [23] and the routines can be seen in appendix E. At the present stage in designing a system for shadow handling, real-time implementation is not considered. However, real-time implementation should be possible in C++ or by the use of designated hardware. In the present Matlab implementation the segmentation of cast shadows takes from a few seconds to several minutes, on a standard 3GHz PC, depending on the size of the foreground object and the parameter values of the methods used. The high resolution of 1024x768 pixels results in severe computational demands, but can be adjusted according to a specific application. Instead of considering real-time implementation at this stage, focus is on making a comprehensive examination of state-of-the-art methods and applying, and improving, these in an application for the DDRE.

5.1 Background Modelling and Noise Reduction

Hansen [18] implemented a kernel-based background model suggested by Elgammal [11]. This implementation with minor modifications is used in this thesis. The parameters are fixed as suggested in [18] to ensure a uniform way of treating the data.

A history of 10 frames are used in the background model. For each pixel the kernel density is estimated using Gaussian kernels at each data point in the history. Kernel estimators have the advantage of being able to model non-parametric densities. This is a

convenient property for pixels that change between several modes, e.g. due to vegetation moving in the wind. The variances of the Gaussian kernels are set as a function of the median of the difference between pixels in the history. For a discussion of how these parameters are set, consult [18].

When a new frame is at hand, the probability of each pixel belonging to the background model is computed, and if below a certain threshold the pixel is considered a foreground pixel. Vegetation moving in the wind still produces foreground pixels considered as noise. The majority of these noisy foreground pixels are suppressed by doing a connected component analysis and removing segments smaller than 50 pixels. Then a morphological closing is applied to avoid small holes in the foreground objects followed by a morphological opening to further suppress noisy segments that are not actual objects. Finally an erosion is done to remove edge pixels of objects that are blurred by the Bayer-filtering. The procedure for noise reduction is based on empirical results, and is performed as a preprocessing step prior to the shadow handling.

The final mask of the foreground objects in the new frame is saved as a *bmp*-image, to be able to split up the background modelling and the shadow removal. Figure 5.1 shows an example of foreground detection using the algorithm just described. A new frame and the foreground detection before and after noise reduction are depicted. Several, but not all, noisy pixels of the moving vegetation are suppressed. General shifts in illumination due to clouds blocking the sun are also a major problem for any background subtraction scheme.

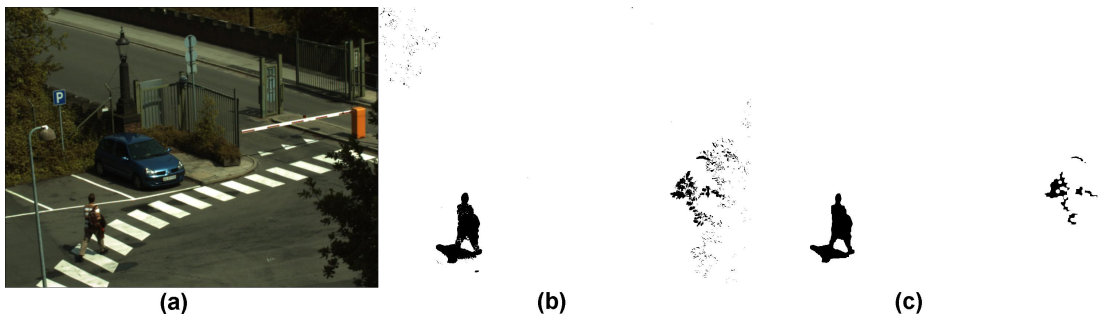


Figure 5.1: *Foreground detection using a kernel-based background model. (a): New frame. (b): Foreground pixels before noise reduction. (c): Final foreground mask.*

The foreground mask is now available for the shadow removal algorithm to remove cast shadows before a further analysis of the object is done. In this thesis only the foreground pixels connected to the object of interest in the shadow mask are used. Objects of no interest are manually suppressed in order to focus the attention on shadow removal.

A 2-class classifier scheme is used, where pixels darker than their corresponding background pixel are classified as either cast shadow pixel or object pixel. Determining which foreground pixels to classify is done by comparing the mean value of the background model for each pixel to the value of the pixel in the new frame. This corresponds to the approach in [21, 26].

5.2 Measuring Performance

All foreground pixels are manually labelled into three classes using standard paint software. These labelled images are used as "ground truth", and are to be compared to the results from the different algorithms. The three classes are "cast shadow" (dark gray), "self shadow" (light gray) and "object not in shadow" (white) (cf. fig. 5.2 and app. B). The background is black.

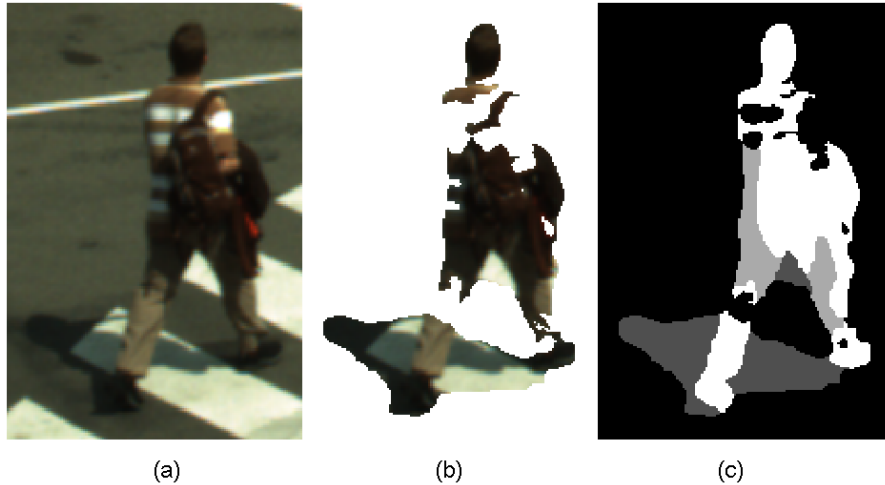


Figure 5.2: (a): Original image zoomed. (b): Foreground pixels to be classified (darker than corresponding background pixels). (c): Manual labelling of pixels. "Cast shadow" = dark gray, "Self shadow" = light gray, "object not in shadow" = white.

Once labelled, a Matlab routine (cf. app. E) is developed to automatically compute the performance of a classifier. In section 2.4 it is mentioned that the performance of Javed's classifier is influenced by the amount of self shadow pixels relative to the total amount of object pixels to be classified. This is the reason for manually labelling foreground pixels into three and not two classes, i.e. object pixels are divided into object pixels in "self shadow", and object pixels "not in shadow".

Several measures of performance are defined to evaluate the algorithms. Perfect performance occurs when a classifier outputs a labelled image identical to the manually labelled image, which is very unlikely to happen. A standard way of presenting the performance of a classifier is using a confusion matrix [17]. Table 5.1 defines the use, in this thesis, of a 2-by-2 confusion matrix:

		Predicted	
		Negative (Cast Shadow)	Positive (Object)
Actual	Negative (Cast Shadow)	A	B
	Positive (Object)	C	D

Table 5.1: 2-by-2 confusion matrix. A standard way of presenting results from a classifier [17].

- A is the number of correct predictions that a pixel is negative - cast shadow pixels.

- B is the number of incorrect predictions that a pixel is positive - cast shadow pixels classified as object pixels.
- C is the number of incorrect of predictions that a pixel is negative - object pixels classified as cast shadow pixels.
- D is the number of correct predictions that a pixel is positive - object pixels.

5 measures of performance are defined. The accuracy (AC) is the proportion of the total number of predictions that are correct:

$$AC = \frac{A + D}{A + B + C + D}. \quad (5.1)$$

The true positive rate (TP) is the proportion of positive cases that are correctly identified, i.e. the proportion of correctly classified object pixels:

$$TP = \frac{D}{C + D}. \quad (5.2)$$

The false positive rate (FP) is the proportion of negatives cases that are incorrectly classified as positive, i.e. the proportion of cast shadow pixels incorrectly classified as object pixels:

$$FP = \frac{B}{A + B}. \quad (5.3)$$

The true negative rate (TN) is defined as the proportion of negatives cases that are classified correctly, i.e. the proportion of correctly classified cast shadow pixels:

$$TN = \frac{A}{A + B}. \quad (5.4)$$

The false negative rate (FN) is the proportion of positives cases that are incorrectly classified as negative, i.e. the proportion of object pixels incorrectly classified as cast shadow pixels:

$$FN = \frac{C}{C + D}. \quad (5.5)$$

Throughout the rest of the thesis, these measures are converted to percentage points, denoted by % for a the simpler notation. Each measure provide useful insight into the strengths and weaknesses of a classifier. The accuracy is the most appropriate sole measure of how well a classifier performs, the other measures are interesting when examining and comparing specific cases and to ensure that the interpretation of the accuracy is useful. It should be noted that there is some redundancy in using all the 4 latter measures, since they pair wise sum to one. However, they are all mentioned since they ease the interpretation of the performance.

When the relative amount of pixels in the two classes are far from equal, the combination of e.g. the FP and the TP gives more insight into the classifiers limitations. This is illustrated graphically in the "Receiver operating characteristics", or ROC-curve [17], which is a useful way to determine the influence on the performance of threshold-dependent model parameters. A ROC-curve is defined as a 2D-plot where $(x, y) = (FP, TP)$. Optimum performance is at the coordinate (0%, 100%).

5.3 Javed’s Method

The implementation of Javed’s shadow removal algorithm, cf. figure 2.2, is straightforward as described in section 2.4. The K-means algorithm is initialized with a Gaussian centered at an arbitrary pixel. All pixels are sequentially assigned to an old distribution or a new distribution is added. Then the connected component analysis is done, followed by a region merging which sequentially merges regions smaller than a certain size with their largest neighbor. Many smaller regions often occur along edges due to noise. The region merging is done sequentially to avoid that these small, noisy, and connected regions are assigned to the largest region, which might not be connected to all of them. Finally, for every region, the correlation of gradient directions between foreground and background image is computed, and a correlation threshold is used in the final classification.

The aim of the optimization is to find a set of parameters that produce a relatively good performance on the training set, i.e. high AC,TP and TN, combined with a low FP and FN. Because of the limited number of examples in both training and testing, it cannot be expected that a fine tuning of the optimization, on the training set, will reveal an improved performance on the test set. Therefore a simple manual optimization is done. 3 parameters in Javed’s method are optimized:

- Fixed variance (σ^2) of Gaussians in the K-means modelling.
- Size criteria when performing region merging.
- Threshold of correlation when classifying regions.

Since all Gaussians have similar diagonal covariance matrices with the same variance in the diagonal, varying the threshold of the Mahalanobis distance (2.2) corresponds to varying the fixed variance. Therefore a fixed threshold of the Mahalanobis distance is chosen, and the fixed variance is optimized instead. Optimization is done manually selecting a set of values for each parameter to be optimized. For every combination of values the performance of the algorithm on the training set is computed. Table 5.2 show the parameter values used in the optimization. Values are chosen so as to represent a suitable part of the parameter domain where the optimum performance is expected to be found, based on initial tests. Good performance is defined as a high accuracy (AC) (5.1), or as a large true positive rate (TP) (5.2), combined with a small false positive rate (FP) (5.3). The data sets are described in section 3.2 and appendix B. Figure 3.4 show 5 of the 18

Fixed variance σ^2	25	36	49	64	81	100
Merging size threshold [<i>pixels</i>]	10	30	50	70	100	150
Correlation threshold	0.00	0.05	0.10	0.15	0.20	0.30

Table 5.2: *Parameter values used in optimization of Javed’s method for shadow removal. Values in boldface produce optimum average performance on the training examples.*

foreground objects used for training. The ROC-curve for all combinations of parameters is depicted in figure 5.3. The colors denote the correlation threshold values, which is the far most performance sensitive parameter of the three. Each color is represented 36

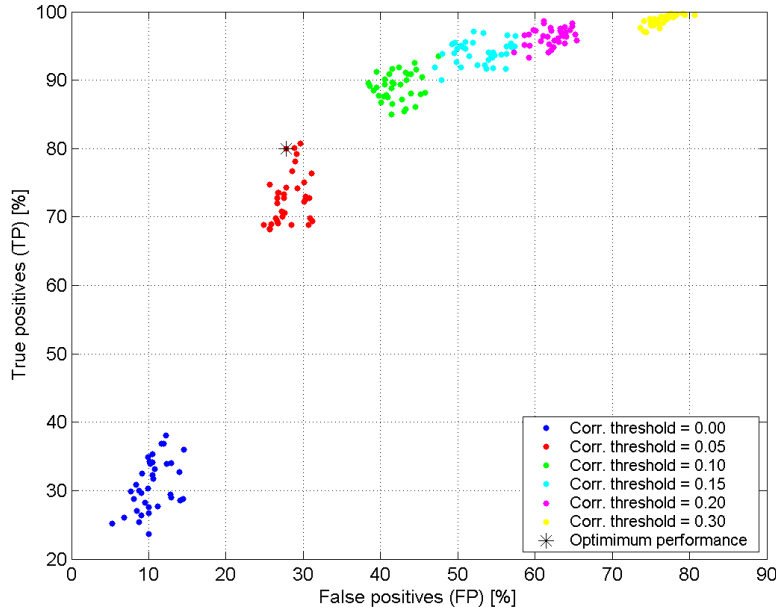


Figure 5.3: Receiver Operating Characteristics (ROC) for all combinations of parameter values of Javed’s method from table 5.2. Colors denote correlation threshold values, which is the most sensitive parameter. The performance of the optimal set of parameters is denoted by an *, at the point $(FP, TP) = (28, 80)$.

times corresponding to the combinations of the two other parameters. These parameters influence the performance much less. This is expected since only the correlation threshold is used as a classification feature. The * denote the set of parameters chosen as the optimal set (typeset in boldface in table 5.2). It is a trade-off seeking to maximize the FP while minimizing the TP. Other sets of parameters could also be argued to be optimal, but the choice made will always depend on the specific application.

The chosen correlation threshold is 0.05. This is a small value compared to the value of 0.75 reported by Javed [21]. The gradient direction as a similarity measure is quite context dependent and a correlation coefficient of 0.05 is only a very weak correlation between two regions. The variance seems to influence the performance less. An optimum value of 81 is much larger than the 32 reported by Javed [22]. The region merging threshold seems to have an optimum value around 100 pixels. Table 5.3 shows the average performance measures of Javed’s method optimized on the training set. The mean value of the AC,

	AC	TP	FP	TN	FN
Mean value [%]	77	80	28	72	20
Standard deviation [%]	13	18	27	27	18

Table 5.3: Average training performance of Javed’s method with optimal parameters.

77%, is somewhat mediocre. A larger FP than TN, and smaller FN than FP, indicate that object pixels are easier to classify correctly, than are cast shadow pixels. This is also

supported by the very high standard deviations of the FP and TN (27%). The high std. also indicate that the method is not robust enough to handle the variation of the training set. A larger data set would be likely to produce a decreased standard deviation. The large mean value and std. of the FP furthermore indicate that there are examples where large parts of cast shadow are misclassified as object.

Figure 5.4 shows performance surfaces (AC) as a function of σ^2 and the merging size threshold, for the 6 values of the correlation threshold, all averaged over the 18 examples. The * denote the optimal set of parameters chosen. Again the major influence

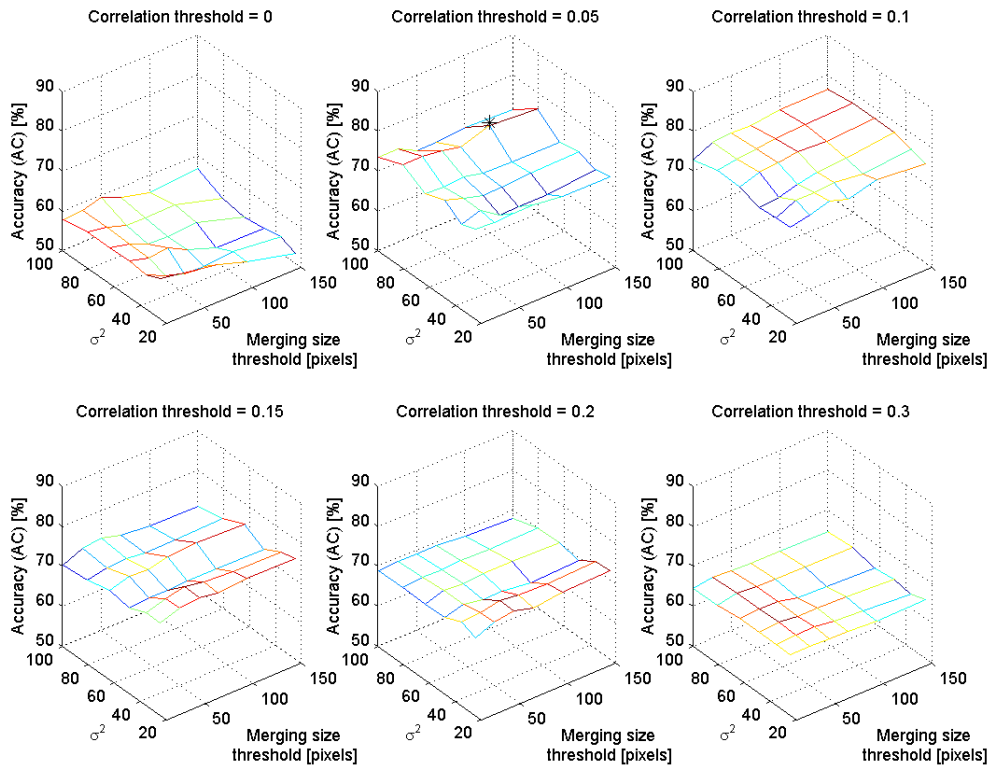


Figure 5.4: Accuracy (AC) of Javed's method as a function of fixed variance σ^2 and merging size threshold, for 6 values of the correlation threshold, averaged over the 18 training examples. The * denote the optimal set of parameters chosen.

of the correlation threshold on the performance is evident. A correlation threshold of 0.1 produce a better accuracy, but still a value of 0.05 is chosen as optimal due to a significant decrease in the FP, cf. figure 5.3. There is no obvious pattern in the way the merging size threshold and the fixed variance influence the accuracy. It depends on the specific value of the correlation threshold.

Figure 5.5 shows the result of the region merging for the subset of 5 training examples from figure 3.4, page 26, on which the classification is partly based. The arrows indicate regions consisting of parts of both the actual object and the actual cast shadow. Test300, and Test429 in particular, have large regions where this is the case. Since the classifier

cannot split up regions, the region merging limits the classifiers ability to achieve optimum performance. This is a trade-off between suppressing small noisy regions, while at the same time avoiding regions consisting of both object and cast shadow.

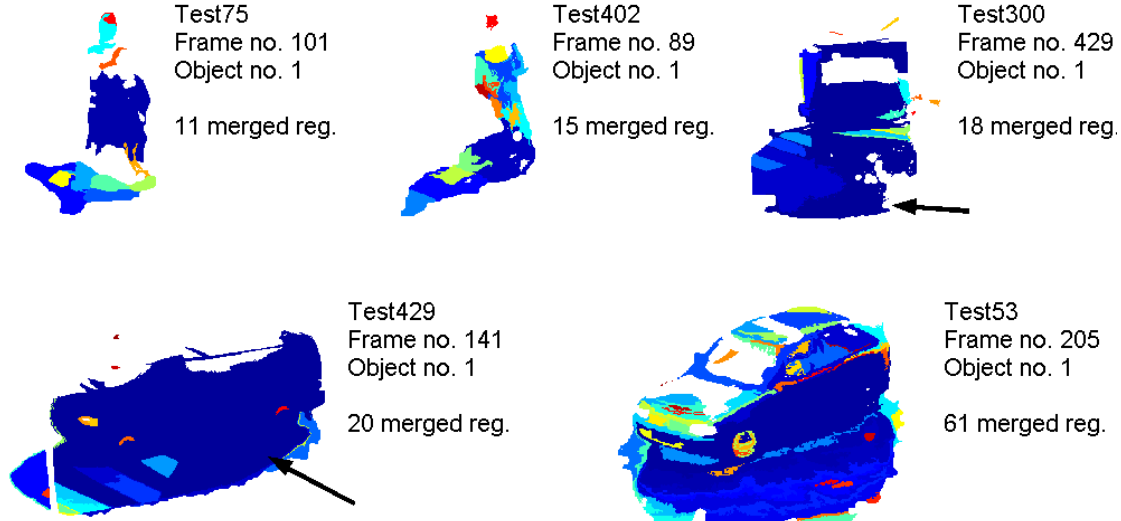


Figure 5.5: Merged regions of part of the training set using Javed's method with optimal parameters. Arrows mark regions consisting of parts of both the actual object and the actual cast shadow, leading to a decreased overall performance.

Figure 5.6 shows the individual classifications of part of the training set. [blue, yellow, red, green] denote [TN, FP, FN, TP] respectively. For example, the accuracy (AC), (5.1), is the sum of green and blue pixels relative to the total amount of pixels. Test300 and Test429 are examples of limitations of Javed's method. Appendix B shows that there are several of the training examples where the FP is very high. The training set contains foreground objects of varying size, from 5,000 to 121,000 pixels to be classified. From the few examples there is no indication of performance depending on object size in pixels. Appendix B contains more information on the training set.

Test300 and Test429 have more than 50% false positives, i.e. cast shadow pixels incorrectly classified as object pixels. This is due to the fact that some regions consist of both cast shadow and object (cf. figure 5.5), which again is due to the limited dynamic range of the camera. The darkest parts of the shadows contain very little texture, hence very little gradient information. Therefore very dark areas of both cast shadow and object (self shadow) are likely to be assigned to the same distribution in the K-means color segmentation. This drawback is also reported by Javed [21]. Other parts of the shadow are cast on areas so bright, that they are actually saturated in the background model, and therefore do not contain any gradient information either. Since the classifier classifies a region with a high correlation (similarity) as cast shadow, and regions of low correlation as object, exceeding the dynamic range will tend to reveal more cast shadow pixels classified as object pixels, i.e false positives (FP).

This is the reason for Test300 and Test429 having a notable lower accuracy (AC) than the other examples. Examining Test300 and Test429 for other sets of parameter values

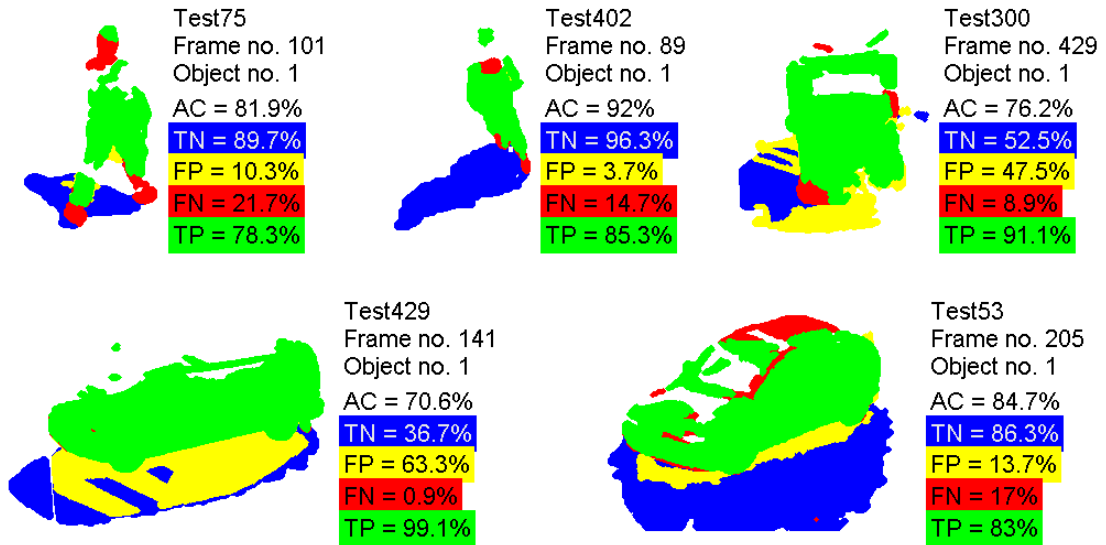


Figure 5.6: Classification of part of the training set using Javed's method with optimal parameters. [blue, yellow, red, green] denote [TN, FP, FN, TP] respectively. Test300 and Test429 have large FP's because there are regions that are saturated in the background image or are too dark to contain any gradient information, resulting in regions containing both object pixels and cast shadow pixels.

revealed a pronounced decrease in performance due to this effect. However, on average the optimization still gives the best performance, given the model and the training set.

Prior to incorporating Finlayson's ideas for shadow removal, it is found necessary to improve the K-means color segmentation due to the reasons just described. The improvement should make the segmentation more sensitive in the darkest areas. How to incorporate this change in Javed's model is the topic of section 5.4. However, in order to separate the effect of the improvement in the color segmentation, from the effect of applying Finlayson's methods, these are applied separately. Parameters of both steps are optimized with respect to the training set, and their performance are compared using the test set in chapter 6.

5.4 Improving Javed's Method

Since the performance of Javed's method on the training set is limited by the K-means color segmentation as described in the previous section, an improved color segmentation (denoted I) is suggested.

Javed suggests a K-means color segmentation using the same fixed variance for every Gaussian distribution, cf. figure 2.2. Because of the limited dynamics range of the camera at hand, dark areas of cast shadow and self shadow can be very difficult to separate, as is the case for Test300 and Test429 in figure 5.5. Testing this fixed variance for values between 25 and 100, revealed an optimum value of 81. For very dark areas this seems quite a large value because the Mahalanobis threshold is fixed at 4. A new pixel would then be assigned to a distribution if it does not differ more than $\pm\sqrt{4 \cdot 81} = \pm 18$ from

the mean.

To improve this separation the use of a variance that is a function of the mean intensity of the distribution a new pixel is compared to, is suggested. A simple piecewise linear function is suggested that assigns small variances to distributions with a low mean intensity and larger variances to distributions with a higher mean intensity. Figure 5.7 shows the variances suggested as a function of the mean intensity of a distribution.

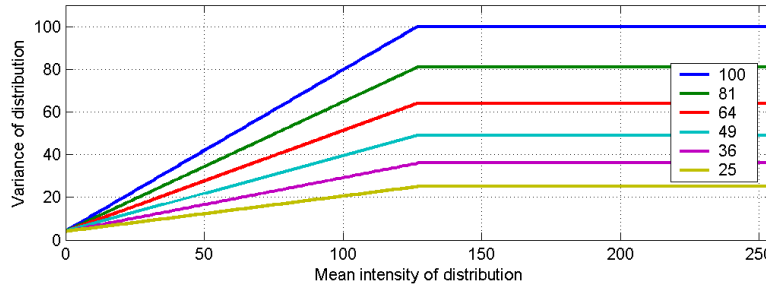


Figure 5.7: Suggested improvement of *K*-means color segmentation. Variance as a function of mean intensity of distribution. 6 piecewise linear functions are tested.

The piecewise linear variance has an offset of 4 for zero mean intensity increasing linearly to a value between 25 and 100 at the mean intensity 128. Above 128 the value remains constant. This is based on the observation that only the darkest areas (distributions with a low mean intensity) should have a small variance. The variance at mean intensity 128 therefore determines the slope of the first part of the function. Other functions than simple piecewise linear functions could also be used, but compared to a fixed value, the piecewise linear function is found to be a relevant choice. The 6 variance functions are denoted by their constant value for high intensities. The performance on the training set, of the 6 functions, are compared to determine the optimum function. The optimization of Javed’s method with improved color segmentation uses the same sets of parameter values as in the optimization of Javed’s original method, cf. table 5.2. The only difference is that the variance is not fixed, but a function of the intensity.

The ROC-curve of the optimization is depicted in figure 5.8. Colors denote values of the correlation threshold which is the most performance sensitive parameter, as in Javed’s original method. The optimal set of parameter values, denoted by an *, are chosen so as to obtain a lower FP at the expense of a lower TP. The values are: *Variance function* = 81, *Merging size threshold* = 10 and *Correlation threshold* = 0.05. Table 5.4 shows the mean values and std. of performance measures. The accuracy is only slightly lower than

	AC	TP	FP	TN	FN
Mean value [%]	75	72	21	79	28
Standard deviation [%]	9	9	16	16	9

Table 5.4: Average training performance of Javed’s method, with improvements in the *K*-means color segmentation, using optimal parameters.

for Javed’s original method. The TP is somewhat lower, but so is FP. Most interesting

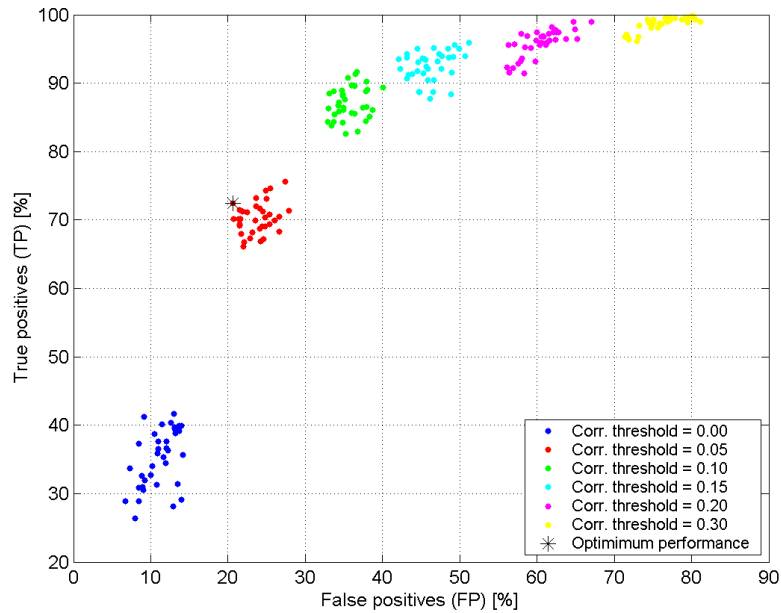


Figure 5.8: ROC-curve for all combinations of parameter values of Javed's method with improved color segmentation from table 5.2. Colors denote correlation threshold values, which is the most sensitive parameter. The performance of the optimal set of parameters is denoted by an *, at the point $(FP, TP) = (21, 72)$.

are the standard deviations, which are lower for all measures, and even halved for the TP and FN. This is an indication of the improved method with the chosen parameters performing more robustly on the training set, than Javed's original method with its chosen parameters. The much smaller variances used in the color segmentation are likely to cause the smaller std.'s in the results. This is due to a larger amount of regions, of smaller size, decreasing the occurrence of large regions consisting of both cast shadow pixels and object pixels.

Figure 5.9 shows that the number of regions indeed are increased, compared to figure 5.5, avoiding the occurrence of large regions consisting of both types of pixels. Due to the amount of regions it can be hard to distinguish all regions in the figure. However, it is stressed that the two regions marked with arrows in figure 5.5 are split into several regions in figure 5.9.

The performance on part of the training set is shown in figure 5.10. The large amount of smaller regions produce a minor decrease in performance for most objects. However, they also produce a major increase in performance in cases that previously had large regions consisting of both types of pixels (cf. Test429 in figure 5.5). This is yet an indication of why the standard deviations of the results are far smaller than for Javed's original method.

This improvement in segmentation is the basis for introducing an extra similarity feature in the classification procedure, as suggested in section 5.6. This feature is derived from an analysis of Finlayson's method for shadow removal using the SVS-204 video

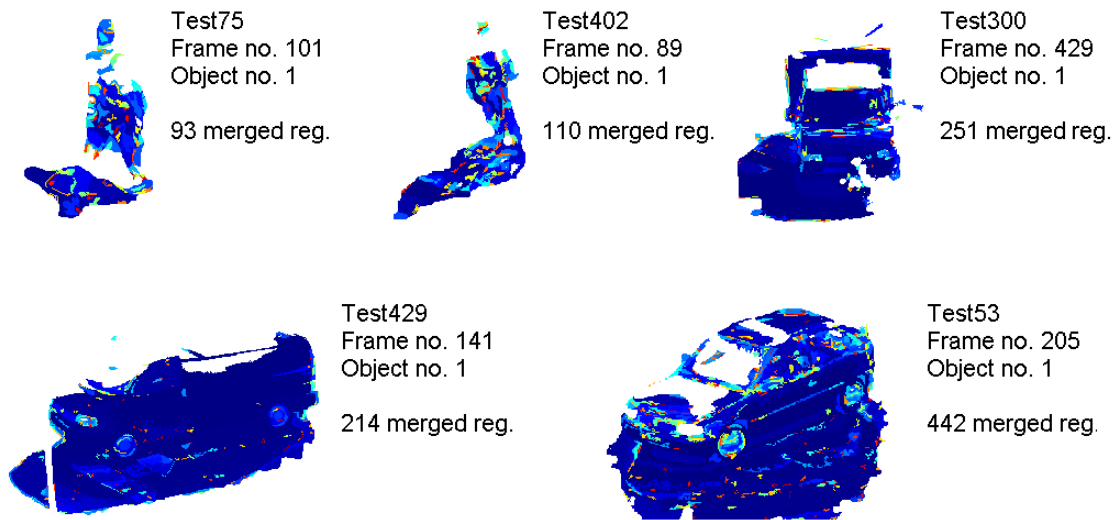


Figure 5.9: Merged regions of part of the training set using Javed's method with improved color segmentation and optimal parameters. Far less large regions consisting of both actual object and actual cast shadow, when compared to figure, 5.5.

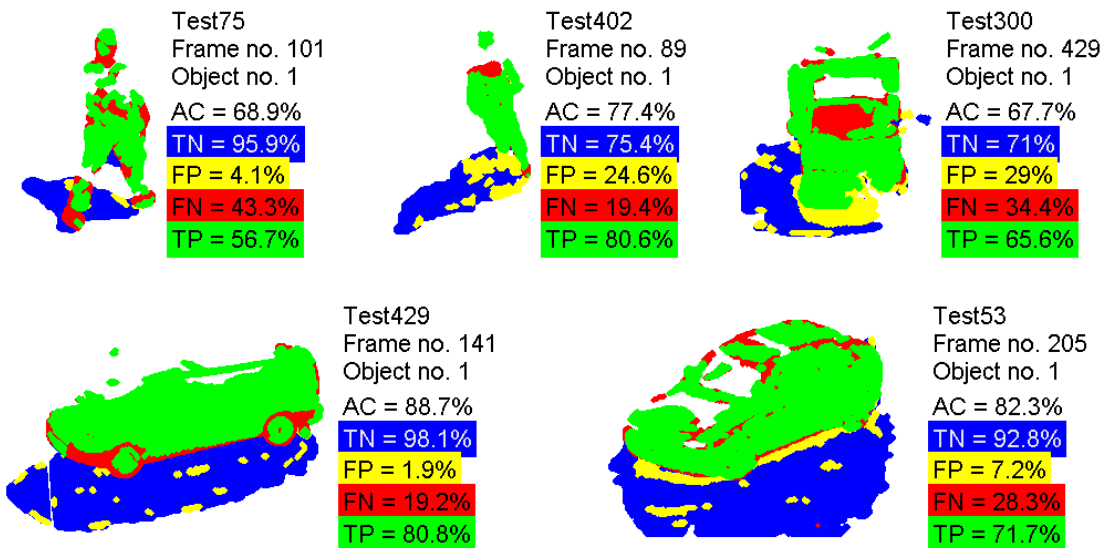


Figure 5.10: Classification of part of the training set using Javed's method with improvements using optimal parameters. [blue, yellow, red, green] denote [TN, FP, FN, TP] respectively.

camera.

5.5 Finlayson's Shadow Removal Applied for Surveillance

This section examines ways of incorporating the elegant physics-based method for shadow removal, suggested by Finlayson (cf. figure 2.4 page 14), in a surveillance application. In the literature the method shows promising results, and therefore it is found relevant to examine the method with the goal of applying it in a surveillance application.

The method makes use of the illumination invariant grayscale image and of the reconstructed "shadow-free" RGB-image. Both are discussed. The objective is to apply parts of Finlayson's ideas to improve the shadow handling of the methods described in previous sections.

5.5.1 Detecting Edges due to Shadows

In order to obtain a "shadow-free" RGB-image, edges due to shadows need to be detected. Finlayson suggests doing this by using the illumination invariant image (cf. section 2.5 and step $3C_F$ in figure 2.4).

Once the illumination invariant direction is obtained from the spectral sensor functions or from a color calibration of the camera (cf. chapter 4), obtaining the illumination invariant grayscale image is simple using equations (2.10),(2.11) and (2.15). Before computing the band-ratio chromaticities, the dynamic range is shifted from $[0, 255]$ to $[1, 256]$ to avoid division by zero.

In chapter 4 it is shown that the illumination invariant image is sensitive to the limited dynamic range of the camera and to the spectral sensor functions of the camera not being delta functions. Because of this, determining edges due to shadows in a robust way becomes very difficult. Finlayson et al. also reports this to be the major drawback of the method [15].

An important observation to make is that a foreground mask is available from the background model in a surveillance application. This can be used to eliminate artifacts from false shadow edges outside the foreground mask, and should be exploited in the detection of shadow edges. Figure 5.11 show the steps applied in the shadow edge detection using Test300 as an example. Figures 5.11(a) and 5.11(b) shows the original image and the corresponding illumination invariant image. The noisy diagonal pattern in dark areas, discussed in section 4.1, should be noted. Figure 5.11(c) is a Canny edge detection of the illumination invariant image, and 5.11(d) is a dilation of these edges. Figure 5.11(e) is the total Canny edge detection of each color band of the original image. The difference between figures 5.11(d) and 5.11(e) should, ideally, reveal edges only due to shadows and not edges due to changes in the surface. That is far from the case due to the limitations of the model and the camera as previously discussed in section 2.5 and chapter 4. Figure 5.11(f) is the difference between figures 5.11(d) and 5.11(e), masked with the foreground mask. It contains a number of false edges and lacks large parts of shadow edges. Canny edge detectors are used because they use two thresholds, detecting both weak and strong edges. Weak edges are then only kept if they are connected to strong edges. The Canny edge detectors used are optimized to a single image but still the result is poor. Appendix C.1 shows the results of using the same parameters on 5 of the training images. These images reveal even worse results.

Figures 5.11(g) and 5.11(h) illustrate how the amount of false shadow edges are reduced by suppressing edges inside the foreground mask. Figure 5.11(g) is a dilated edge of the foreground mask of the object. It is used as a mask on 5.11(f) and results in 5.11(h) after a dilation. The resulting image still has some false shadow edges along the border of the foreground object, and substantial parts of the shadow edges are still not detected.

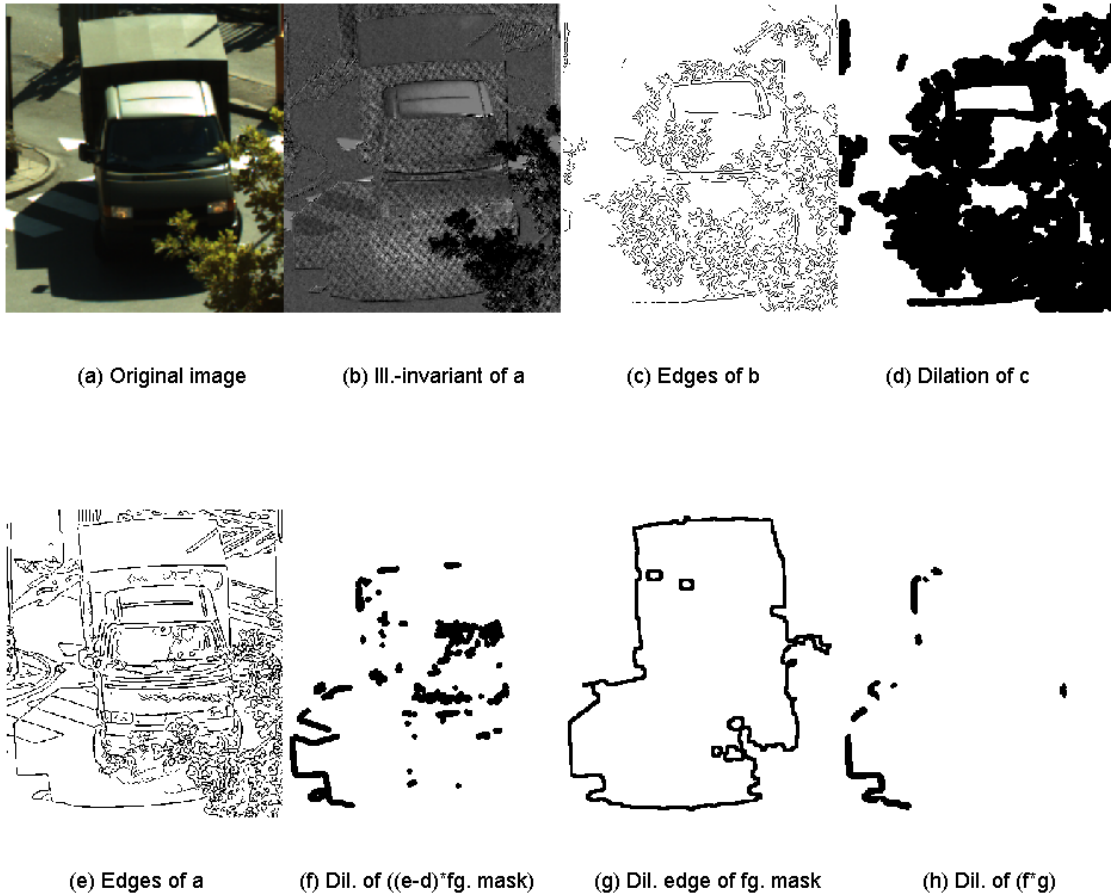


Figure 5.11: *Detection of shadow edges. (a): Original image. (b): Illumination invariant. (c): Edge detection of (b). (d): Dilation of (c). (e): Edges of (a) for all color channels. (f): Dilation of difference between (d) and (e) masked with the foreground mask. (g): Dilation of edge of foreground mask. (h): Dilation of (f) masked with (g).*

However, all the false edges inside the foreground mask in figure 5.11(f) are suppressed. Applying this approach to 5 of the training images (cf. appendix C.1) still produces poor results.

Based on the limitations of the camera described in chapter 4, the assumptions of the model, and the poor results when applying Finlayson's illumination invariant image on the training set, it is assessed to be much too sensitive to be used for shadow detection in the present framework.

Instead another approach is suggested to determine which gradients to suppress before reconstructing the "shadow-free" image. It is not optimal in the sense that it will detect all shadow edges and not detect any false shadow edges. Instead it assumes that shadow edges are most likely to occur along the edge of the foreground pixels, i.e. along the edge of the foreground mask. The idea is to suppress all gradients along the edge of the foreground mask (cf. figure 5.11(g)) and reconstruct a "semi-shadow-free" image, which contains shadows that are removed along the edge. This image contains information that

in section 5.6 is suggested to improve the final segmentation of foreground pixels into shadow regions and object regions. The reason for suppressing gradients all along the edge of the foreground mask (figure 5.11(g)), and not only along parts of the edge as in figure 5.11(h), is that the latter approach does not seem to be the least robust when applied to more than one image. Therefore an approach is taken that is likely to reveal some useful results in the majority of cases due to the fact that cast shadows usually somehow touch the edge of the foreground mask. In these cases the cast shadow will be suppressed revealing a change that can be used in the segmentation.

5.5.2 Reconstructing the RGB-image without Shadows

After determining which edges to suppress in the gradient image, the reconstruction of the "shadow-free" RGB image can be done (cf. step $3D_F$ in figure 2.4 page 14). Finlayson et al. [15] do not mention how they solve the Poisson equation in (2.19), which is repeated here for convenience:

$$\nabla^2 q'(x, y) = \nabla \cdot S(\nabla \rho'(x, y), \nabla g_s(x, y)), \quad (5.6)$$

where ∇^2 is the Laplacian, q' is the logarithm of the image without shadows, and S is the gradient of the logarithm of the original image, with edges suppressed that are due to shadows. In order to solve this equation, Neumann boundary conditions are assumed, i.e. derivatives on boundary normals are assumed to be zero.

Since the image of which the Poisson equation should be solved is rectangular, a Fourier transform method can be applied [31]. More specifically the cosine transform is used since it ensures that the Neumann boundary conditions are valid since the derivative of the cosine is the sine which attains zero value at multiples of π (on the borders of the image). It can be shown [31] that applying the cosine transform to a zero-padded version of $\nabla \cdot S$ for each color band, followed by applying equation (5.7) and the inverse cosine transform, the Poisson equation (5.6) can be solved.

$$\begin{aligned} \widehat{q}'_{mn} &= \frac{\widehat{\nabla \cdot S}_{mn}}{2(\cos \frac{\pi m}{J} + \cos \frac{\pi m}{L} - 2)}, \\ m &= \{0, 1, \dots, J - 1\}, \\ n &= \{0, 1, \dots, L - 1\}, \end{aligned} \quad (5.7)$$

where the *hat* ($\widehat{}$) denotes the discrete cosine transform, and J and L denote the dimensions of the zero-padded image. q' is then exponentiated to reveal q which is uniquely determined except for a multiplicative constant in each color band (cf. section 2.5). This constant is determined by mapping the color bands to the full dynamic range of an 8-bit image, 0 – 255. The mean value of the brightest 5% pixels of each band is mapped to maximum value, 255. Finlayson suggest mapping the mean of the brightest 1% pixels to maximum, but this revealed reconstructed training images where shadow regions were hardly attenuated at all.

Figure 5.12(a) shows an image and a version of it, figure 5.12(b), that is reconstructed without suppressing any gradients. Therefore the two images are similar. Figure 5.12(c)

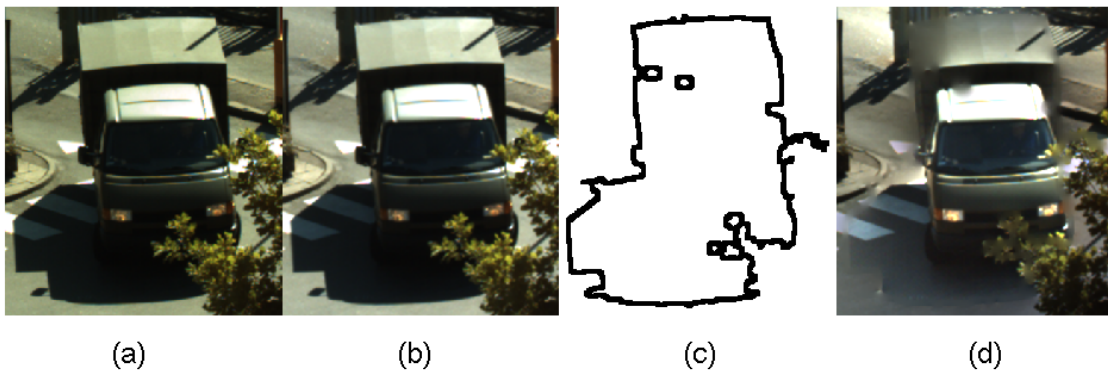


Figure 5.12: *Reconstruction of an image. (a): Original image. (b): Reconstructed image without suppressed gradients. (c): Suggested mask for suppressing gradients. (d): Reconstructed image with suppressed gradients.*

shows the mask suggested in section 5.5.1 for suppressing gradients, and figure 5.12(d) shows the corresponding reconstructed image. Even though both shadow and object gradients are suppressed, figure 5.12(d) clearly contains additional information that could improve the segmentation of cast shadows.

5 of the training images are also reconstructed using the gradient masks of figure 5.11(f) and 5.11(h), to ensure that they do not prove useful as predicted in section 5.5.1. This indeed is the case, supporting the use of the gradient mask of figure 5.12(c) obtained from the total edge of the foreground mask. Section 5.6 suggests how the additional information of figure 5.12(d) can be applied in an enhanced algorithm for segmentation of the cast shadows.

5.6 Enhanced Shadow Removal

In section 5.3 and 5.4 it is shown that the performance of the segmentation algorithms is most sensitive to how the correlation threshold is set. Instead of using only the gradient direction to measure similarity of two regions, an additional similarity feature, (5.8), is suggested based on the reconstruction of an RGB-image with gradients suppressed along the edge of the foreground mask, as described in section 5.5.2. The color segmentation of pixels into regions, and the following region merging is similar to the improved color segmentation described in section 5.4.

Figure 5.13, corresponding to step 3 in figure 2.1 page 7, depicts the total enhanced shadow removal as suggested by the author. Steps with subscript E (red) are the enhanced steps introduced.

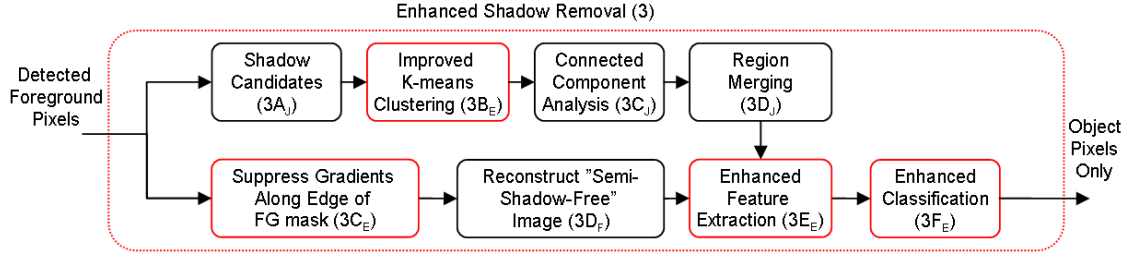


Figure 5.13: Flowchart of enhanced shadow removal as suggested in this thesis. Corresponds to step 3 in figure 2.1. Subscripts denote from where the original idea came: J =Javed, F =Finlayson and E =Enhanced steps suggested by the author (red).

5.6.1 Enhanced Similarity Feature

The new similarity feature compares corresponding pixels of the reconstructed image and the background image, for every color segmented region:

$$CS = \frac{1}{\hat{\sigma}_{R,BG}^2(K-1)} \sum_{i=1}^K (R_i - BG_i)^2, \quad (5.8)$$

where CS is the similarity feature of a region, K is the number of pixels of the region, R and BG are the intensity values of the i 'th pixels in the reconstructed image and the background image, respectively. $\hat{\sigma}_{R,BG}^2$ is a variance normalization factor, which is the estimated variance between all pixels in a background image, BG , and all pixels in a reconstructed image, R , of a new frame *containing no foreground objects*. Estimating this variance normalization factor is illustrated in the upper part of figure 5.14.

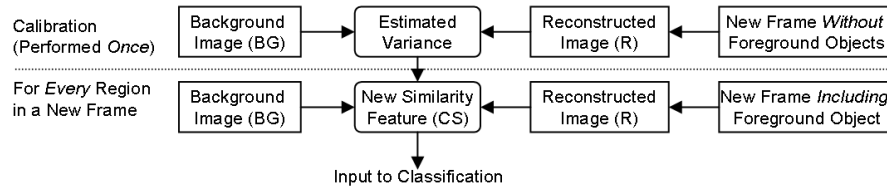


Figure 5.14: Flowchart illustrating the enhanced similarity feature (CS). (Upper): Variance between background image and new frame without any foreground objects is estimated once. (Lower): In a new frame, including detected foreground objects, the enhanced similarity feature (CS) is computed for every region and is a part of step $3E_E$ of figure 5.13.

Performing a variance normalization of CS makes it a relative measure of similarity that, ideally, only contains variation due to the region not being cast shadow, and not contains variation due to the experimental setup and the complex processing of the images. The estimate of the variance is based only on one sequence (Test77), since it was difficult to obtain sequences, without foreground objects, that were static while an entire background model was estimated, and also static for a new frame. The estimate of the variance of 255 is therefore a rough, but useful, estimate. The lower part of figure 5.14 illustrates

how the similarity feature, CS , is obtained for every color segmented region. Figure 5.14 corresponds to step $3E_E$ in figure 5.13.

The CS measures a normalized mean value between regions in the reconstructed foreground image, cf. figure 5.12(d), and corresponding regions in the background image. If the reconstructed image contains shadow regions along the border of the foreground mask, cf. figure 5.12(c), these shadow regions are attenuated in the reconstructed image, making them more similar to the background image. This is the key observation that the enhanced similarity feature, CS , is based on. Therefore a large value of CS corresponds to little similarity, which indicates that the region is part of the object. Small values of CS indicate high similarity, i.e. the region is then part of a cast shadow.

It is emphasized that CS only supplies useful information when the shadow edges are actually part of the edge of the foreground mask. In some cases it will not supply any additional information, e.g. when edges due to objects instead of shadows are suppressed. This will tend to smear neighboring background and object regions, for which reason it is suggested only to apply the CS in cases where the correlation threshold does not produce confident results. This corresponds to introducing a *reject class* [2] for the correlation feature.

Figure 5.15 shows the suggested enhanced classification of color segmented regions, corresponding to step $3F_E$ of figure 5.13. The left part corresponds to the classification

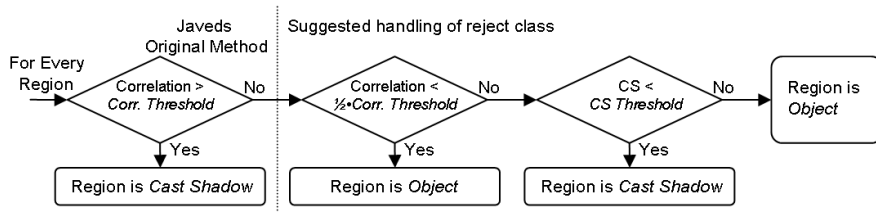


Figure 5.15: Flowchart illustrating the enhanced classification of color regions (step $3F_E$ in figure 5.13). The enhanced similarity feature, (CS), classifies all regions that the correlation feature assign to a reject class ($0.5 \cdot Corr. threshold < Correlation < Corr. threshold \Rightarrow reject class$).

originally suggested by Javed, using a simple correlation threshold. The enhanced classification introduces a reject class if the correlation lies in an interval between 0.5 and 1 times the *Correlation threshold* introduced by Javed [21]. If the regions in the reject class have a CS larger than the CS threshold they are classified as object regions. Otherwise they are classified as cast shadow regions.

This is the total enhanced shadow removal suggested, which is depicted in figure 5.13.

5.6.2 Applying the Enhanced Similarity Feature

Similar to the other methods applied, the enhanced method is also manually optimized, with respect to the training set, for different sets of parameter values. The parameter values are based on initial tests to ensure reasonable performance. Table 5.5 shows the set of parameter values used in the optimization. As previously mentioned the

improved color segmentation of section 5.4 is applied. Values in boldface denote those values chosen as optimal.

Variance function (upper value)	25	49	81	
Merging size threshold [<i>pixels</i>]	10	50	70	100
Correlation interval for reject class	0.05-0.10	0.075 – 0.15	0.10 – 0.20	
CS threshold	3	5	7	

Table 5.5: Parameter values used in optimization of enhanced method for shadow removal. Values in boldface produce optimum average performance on the training examples.

Figure 5.16 shows the ROC-curve, zoomed, for the parameter values, averaged over the training set. Colors denote varying values of the *CS threshold*, and the marker types denote varying intervals of the *correlation reject class*. The latter are denoted by their upper value in the legend. The optimal set of parameters chosen, are denoted by an *, (FP, TP) = (26, 83). Table 5.6 shows the averaged performance measures for the

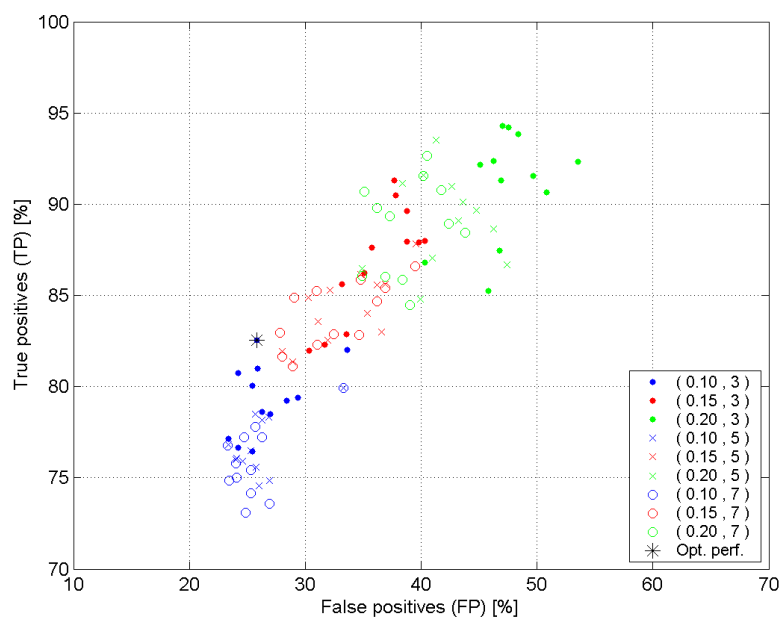


Figure 5.16: ROC-curve for all combinations of parameter values of enhanced method from table 5.2. Colors denote values of the CS threshold, which is the most sensitive parameter. Marker types denote varying intervals of the correlation reject class. The latter are denoted by their upper value in the legend. The performance of the optimal set of parameters is denoted by an *, at the point (FP, TP) = (26, 83).

optimal set of parameters. All of the 5 measures show better performance than did Javed’s method, cf. section 5.3, and this even with a decrease in the standard deviations of the measures. Comparing to Javed’s method with improved color segmentation, section 5.4, the enhanced method is better at detecting object pixels correctly (TP), but worse at detecting shadow pixels correctly (TN). The total accuracy is better for the enhanced

	AC	TP	FP	TN	FN
Mean value [%]	78	83	26	74	17
Standard deviation [%]	12	11	22	22	11

Table 5.6: Average training performance of enhanced method, using optimal parameters.

method, but all of the standard deviations are somewhat larger.

Examining the 5 training examples of figure 5.17, it is evident that the enhanced method improves the classification when comparing to the other methods, figures 5.6 and 5.10. The enhanced method and the method with improved color segmentation both show

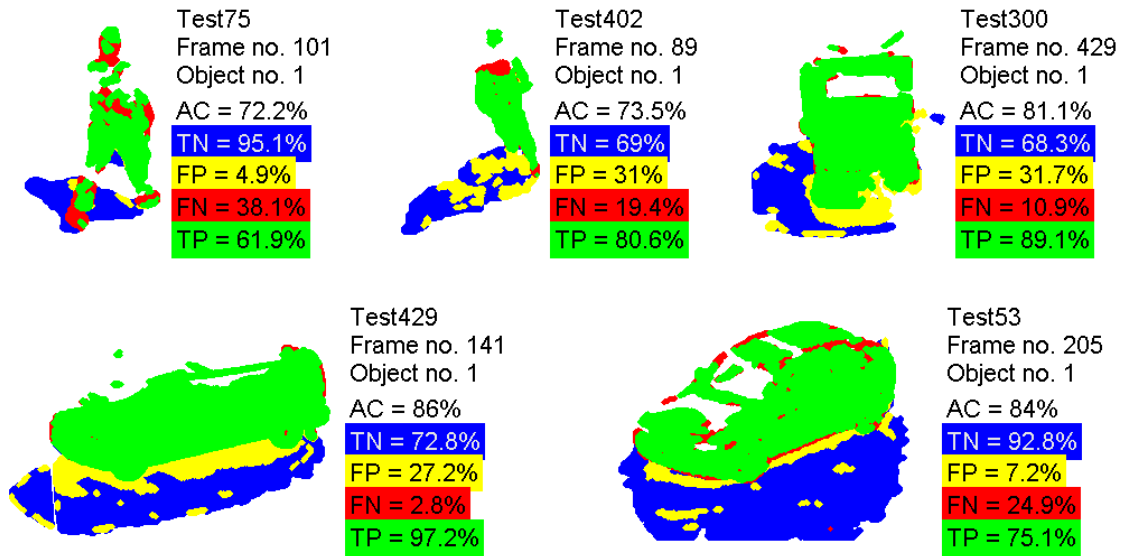


Figure 5.17: Classification of part of the training set using enhanced method and optimal parameters. [blue, yellow, red, green] denote [TN, FP, FN, TP] respectively.

better performance on very dark regions, Test300 and Test429. However, they also show a major decrease in performance on the other 3 objects, although the enhanced method is slightly better than the method with improved color segmentation.

All in all, the enhanced method for shadow removal is found to improve upon the shadow removal suggested by Javed, based on a training set of 18 images, with optimized parameters.

5.7 Summary

Three methods for shadow removal are implemented and optimized manually on a training set consisting of 18 images. Each method classifies foreground pixels, that are darker than their background pixels, as either cast shadow pixels or object pixels (2-class problem). All foreground pixels are manually labelled, to serve as ground truth, when evaluating the classifiers. Several measures of performance are defined, e.g. the overall accuracy (AC), which is the proportion of pixels that are correctly classified.

Javed's original method (*J*) is optimized to the training set and used as a reference. From an analysis of the results, Javed's method is shown to fail in cases where there are large dark regions consisting of both actual cast shadow pixels and actual object pixels. This is due to the limited dynamic range (8 bits) of the video camera, a limitation often encountered in real-world applications. However, an improved color segmentation is suggested (method *I*), resulting in a larger amount of regions of smaller size, that are to be classified. The latter method solves the problems of large dark regions, but is susceptible to smaller regions being misclassified, leading to a decreased accuracy, though it is more robust than Javed's original method.

Finlayson's ideas for shadow removal are then implemented, i.e. the illumination invariant image is derived, from which shadow edges are detected and suppressed, and finally a "shadow-free" image is reconstructed. Due to limitations in deriving the illumination invariant image, described in previous chapters, it was not possible to design a robust shadow edge detection that produced the least promising results on more than one of the images of the training set. Instead it was suggested that the gradients of all pixels along the edge of the foreground mask, obtained from the foreground detection, be suppressed. This is based on the assumption that the cast shadows in the vast majority of cases constitute parts of the edge of the foreground object. Then a "semi-shadow-free" image is obtained. From this image an enhanced similarity feature, *CS*, is derived, which compares regions of the reconstructed "semi-shadow-free" image with the corresponding regions of the background image.

Due to the assumptions when deriving the enhanced similarity feature, it is only applied in cases where the correlation feature suggested by Javed, does not produce a confident classification, i.e. a rejection class is introduced. This is the final enhanced method for shadow removal suggested (method *E*). With optimized parameters it produces slightly better results on the training set, than does Javed's original method.

Chapter 6

Validation and Comparison

Based on the optimized parameters found in chapter 5, the three methods for shadow removal (J, I and E) are validated on the test set, and compared. Validation is the process of determining how the methods generalize to unknown data. It is done by distinguishing between the data used for optimizing model parameters and the data used for determining the performance of the models. How the total data set is split into a training set and a test set is discussed in section 3.2 and appendix B. The test set consists of 72 foreground objects to be classified. On average, the examples of the test set consist of 55% object pixels, with a std. of 17%, and the rest being cast shadow pixels.

Initially, the *absolute* performance of the three methods on the test set is described, followed by the introduction of measures that express *relative* improvements between methods. Then the setup for statistical testing is described, which is applied to determine any significant differences in performance of the three methods.

The methods are compared pair wise using different statistical methods and tests, each contributing to the overall evaluation. A simple binomial variable is used to, pair wise, compare methods, per-example. Furthermore a paired t-test is used to reveal significant differences in both the absolute mean values of the results, and the relative mean values. Only the measures AC, TP and TN are used when testing, since the measures FP and FN are given when the others are known and therefore provide no additional information. A significance level $\alpha = 0.05$ is used through all tests.

6.1 Absolute Performance

Table 6.1 shows the mean and std. of the absolute performance measures, based on the test set, for the three methods. The results of each example can be found in appendix D.

As expected the general performance of all methods on the test set is worse than that based on the training set. This, of course, is due to the fact that the methods have parameters optimized to the training set. However, many of the same trends appear, as appeared on the training set. It is emphasized that only trends can be extracted from the averaged values. Any final conclusions must be based on tests revealing statistical significance. The following trends should be noted:

Method	AC	TP	FP	TN	FN
J - Mean (Std.) [%]	64.9 (17.8)	63.4 (30.0)	35.3 (33.4)	64.7 (33.4)	36.6 (30.0)
I - Mean (Std.) [%]	67.5 (13.8)	63.1 (18.9)	29.3 (23.3)	70.7 (23.3)	36.9 (18.9)
E - Mean (Std.) [%]	69.2 (13.7)	69.7 (18.3)	34.0 (23.9)	66.0 (23.9)	30.3 (18.3)

Table 6.1: Absolute performance of the three methods (J , I and E) based on the test set of 72 examples. Mean values and standard deviations are shown.

- The enhanced method (E) seems to perform better than Javed’s method (J) in all measures as well as in robustness (low std.).
- (E) also performs better than (I) for the measures AC, TP and FN.
- (I) has the same tendency, of a lower std. and a better TN and FP, than (J) does, as the training set also indicated.
- The accuracy of (I) is better, compared to that of (J). This was not the case for the training set.
- However, all the mean values are quite close when taking the std.’s into account.

Figure 6.1 illustrates some of the results from table 6.1. Figure 6.1(a) and 6.1(b) compare the accuracy, for each example in the test, of (I) and (E), to that of (J). If samples lie to the right of the diagonal, the accuracy is better compared to Javed’s method, and vice versa. The mean values are denoted by an \mathbf{x} . There is no general trend for any of the methods to outperform the other, when looking at each example on its own. Still, the average values indicate some trends.

Both plots show that in general, examples that give low AC’s in (J), give better AC’s in both (I) and (E). The opposite is the case for examples that give high AC’s in (J). However, there is a trend that examples with a higher AC in (I) and (E), are improved more than the examples with decreased AC, are decreased. This gives rise to the higher mean values, and supports the conclusions of chapter 5, that fewer examples tend to have much better AC, while more examples tend to have slightly decreased AC.

Figure 6.1(c) shows the mean TP as a function of mean FP for the three methods, based on the training set and the test set respectively. This illustrates the differences in performance collectively. Figure 6.1(d) shows the histograms of the AC of (J) and (E), along side their fitted Gaussians. This is another illustrative way of presenting the tendency of (E) to be higher and more robust.

6.2 Relative Performance

The performance measures introduced in section 5.2 are absolute measures, for each method individually, in the sense that they make no distinction between improving a poor performance, and improving a good performance. However, in the comparison of methods, the following *relative* performance measures are suggested to indicate relative

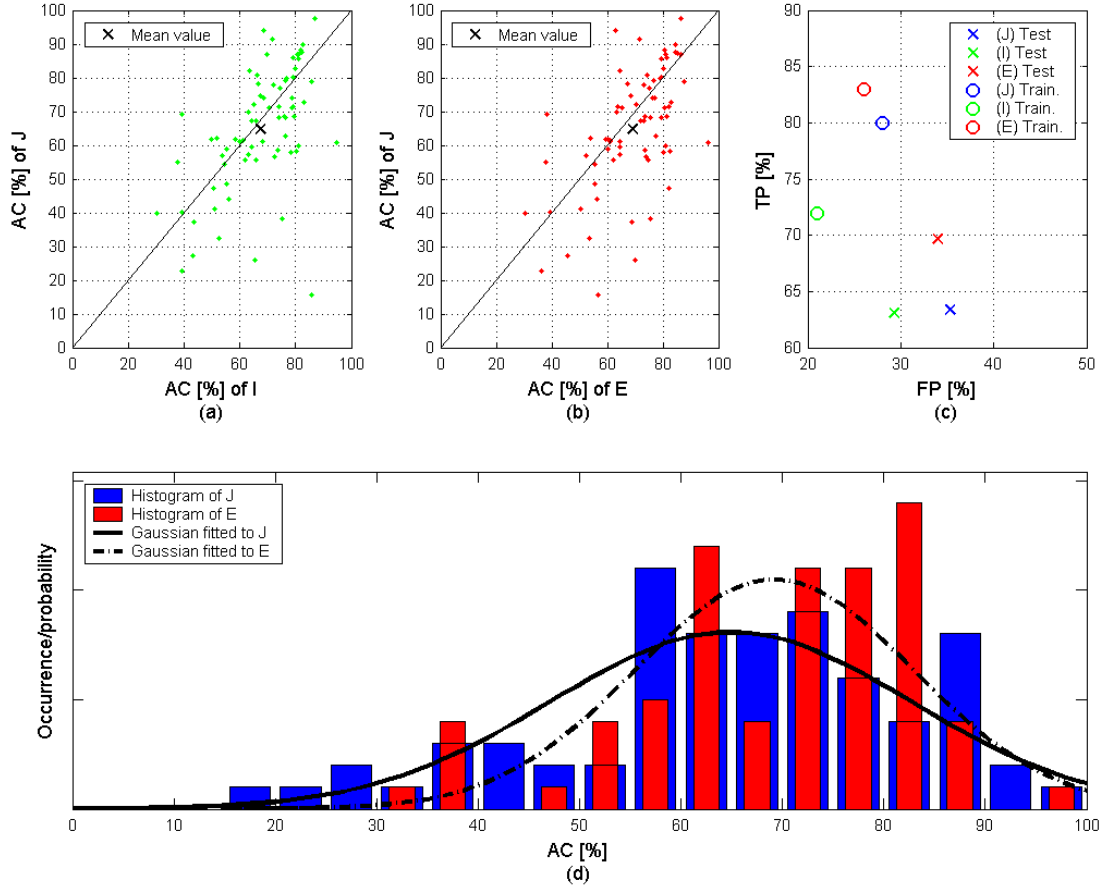


Figure 6.1: Comparison of performance. (a): Accuracy of Javed's method (J), as a function of accuracy of improved method (I), based on the test set. (b): Accuracy of Javed's method (J), as a function of accuracy of enhanced method (E), based on the test set. (c): Mean TP as a function of mean FP for the three methods, based on the training set and the test set. (d): Histograms and fitted Gaussians of J and E, based on the test set.

improvements between two methods:

$$AC_{R,XY} = \frac{AC_Y}{AC_X} - 1, \quad (6.1)$$

$$TP_{R,XY} = \frac{TP_Y}{TP_X} - 1, \quad (6.2)$$

$$TN_{R,XY} = \frac{TN_Y}{TN_X} - 1, \quad (6.3)$$

$$X = \{J, I\}, \quad Y = \{I, E\}, \quad X \neq Y,$$

where R denotes the measure being relative, X denotes the reference method, and Y denotes the method being tested for improvements. Positive values indicate better performance of Y than of X . Only improvements between methods J and I (denoted JI), I and E (denoted IE), and between J and E (denoted JE), are examined explicitly, since it is the intention to determine from which of the new methods (I, J) any improvements

originate. Similar measures $FP_{R,XY}$ and $FN_{R,XY}$ are not suggested, since they are sensitive to near-zero values of the denominator, and they produce redundant information.

Table 6.2 shows the relative improvements in performance based on the test set. E.g. JE show the relative improvements in performance, when using method E instead of method J . Mean values and standard deviations are shown in %. It should be noted that only examples where the denominator of equations (6.1),(6.2) and (6.3) is above zero, are used for computing the relative measures. For the measures $\{AC, TP, TN\}$, $\{72, 72, 66\}$ examples were used respectively. Similar to the absolute measures, only trends can be

Compared methods (XY)	$AC_{R,XY}$	$TP_{R,XY}$	$TN_{R,XY}$
JI - Mean (Std.) [%]	13.5 (60.0)	86.5 (34.7)	31.2 (115)
IE - Mean (Std.) [%]	3.20 (12.0)	13.4 (24.0)	-5.90 (13.7)
JE - Mean (Std.) [%]	14.9 (44.2)	108 (380)	22.1 (113)

Table 6.2: *Relative improvements in performance based on the test set of 72 examples. E.g. JE show the relative improvements in performance, when using method E instead of method J . Mean values and standard deviations are shown in %.*

shown from mean values. Final conclusions must be based on tests producing statistical significant results. The trends of the relative performance measures are similar to those of the absolute measures, with a few exceptions:

- Both methods I and E improves upon method J for all measures.
- Some of the standard deviations are quite large, due to the nature of the relative measure.
- Examples far from the diagonal of figures 6.1a and 6.1b produce relative measures far from zero.
- The mean relative increase in AC, of methods I and E as compared to J , is nearly the same, though E has a smaller std.
- Method E improves upon I in both relative AC and TP, but not in relative TN.

Compared to the absolute measures, the relative measures indicate that the suggested methods, in particular, improve upon Javed’s method, when the latter produces mediocre performance.

6.3 Comparison Per-Example (Binomial)

In this section the methods are compared per-example, to see how often one method outperforms another method.

A random binomial variable p is defined as the proportion of examples, out of the total number of independent examples, that produce e.g. method E superior to method J .

	JI=I>J	IE=E>I	JE=E>J
\hat{p}_{AC}	33/72	50/72	37/72
\hat{p}_{AC} [%]	45.8	69.4	51.4

Table 6.3: Comparison of methods per-example. \hat{p}_{AC} is the estimate of the proportion out of the total test set, that shows higher accuracy for one method compared to another.

Table 6.3 shows the estimate, \hat{p}_{AC} , of the proportion of the examples that show higher accuracy. Fewer examples produce higher AC for I than for J . Only slightly more than half of the examples show higher performance for E than for J . This is also shown in figures 6.1(a) and 6.1(b). The confidence interval (CI) is influenced by the number of examples, as illustrated in figure 6.2. Using the curve of 50 examples produce a 95%

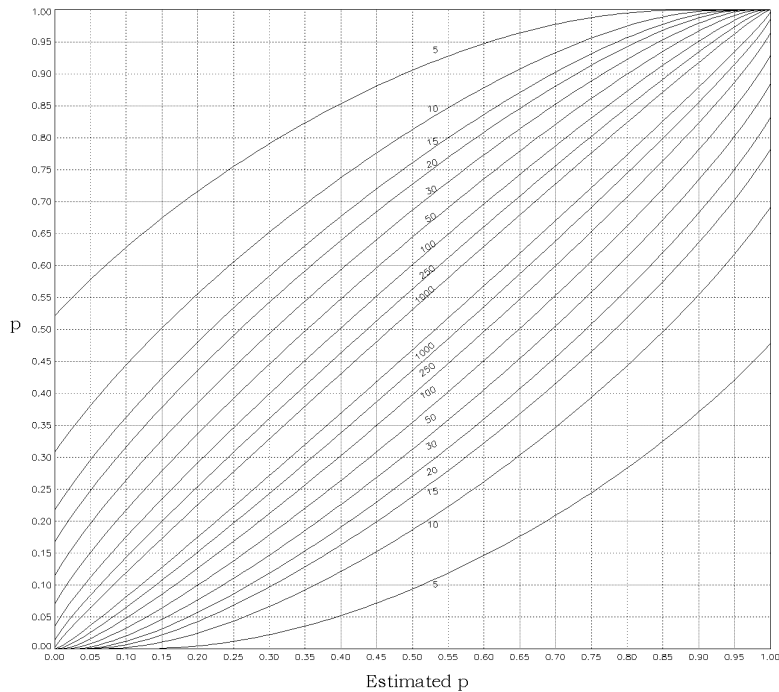


Figure 6.2: 95% confidence interval of the maximum likelihood estimate of a binomial variable, as a function of the number of examples [10].

confidence interval of approximately $[0.36; 0.66]$ for $\hat{p} = 0.51$ [10, 5, 7]. If the CI is computed repeatedly for new observations, 95% of the CI's will contain the true mean value of the variable. Ideally e.g. 250, or more, examples would reveal narrow confidence intervals, but when the estimate of p is so near to 50% the only conclusion to make, is that there is no significant improvement in the proportion of examples with higher accuracy, when using E compared to J . The same conclusion is made when comparing method I to method J . However, when comparing method E to method I , the CI is approximately $[0.54; 0.82]$ for $\hat{p} = 0.69$. Therefore it is likely to obtain better accuracy per-example using method E , compared to using method I .

6.4 Comparison of Means (Paired t-Test)

Instead of comparing the methods per-example, the mean of the performance measures can be compared. This is done by applying a paired t-test, which is described in this section.

The paired t-test is applicable in situations where the same examples of a random sample are treated by two different methods that are to be compared [7]. The paired t-test then tests for any significant differences in the mean values, i.e. is there a significant difference in the way the methods treat the data.

The following example illustrates how a *one-sided* paired t-test is applied for testing if method E produce a significantly higher accuracy, AC , than method J does.

J_{AC} and E_{AC} are random variables of the accuracies produced by applying methods J and E on the detected foreground objects. $\mu_{J_{AC}}$ and $\mu_{E_{AC}}$ are the corresponding expected (mean) values of the accuracies. Then $D_{AC} = E_{AC} - J_{AC}$ is the difference in accuracy of the two methods. The D_{AC} is then assumed to be normally distributed with mean $\mu_{D_{AC}} = \mu_{E_{AC}} - \mu_{J_{AC}}$, and variance $\sigma_{D_{AC}}^2$. The null hypothesis, H_0 , and the alternative hypothesis, H_1 , then are [7]:

$$\begin{aligned} H_0 &: \mu_{D_{AC}} = 0, \\ H_1 &: \mu_{D_{AC}} > 0, \quad (\text{one-sided}), \\ \text{Test statistic value: } t &= \frac{\hat{d}_{AC}}{s_{D_{AC}}/\sqrt{n}}, \\ \text{Rejection region for level } \alpha \text{ test: } t &\geq t_{\alpha, n-1}, \end{aligned} \tag{6.4}$$

where \hat{d}_{AC} and $s_{d_{AC}}$ are the sample mean and std., respectively, computed for the $n = 72$ examples of the test set. $\alpha = 0.05$ is the significance level of the test. The null hypothesis is rejected at a 5% level if the test statistic t lies in the rejection region determined by $t_{\alpha, n-1}$ of the t-distribution [7, 5]. The test is considered significant if the null hypothesis is rejected. If so, method E produces a significantly higher mean accuracy than does method J . In addition to this interpretation of the test, the p -value can be reported. It denotes the highest level of significance, i.e. lowest value, where the test would still reject the null hypothesis. This is useful additional information, but should not be used to set the α -value. Furthermore a lower confidence bound (LCB) for $\mu_{D_{AC}}$ can be estimated, for the one-sided test, using [7]:

$$LCB = \hat{d}_{AC} - t_{\alpha, n-1} \cdot \frac{s_{D_{AC}}}{\sqrt{n}}. \tag{6.5}$$

If the LCB is computed repeatedly for new experiments, with $\alpha = 0.05$, then 95% of the LCB's will be a lower bound of the interval where the true mean value of the variable lies

A one-sided test is used, since it is the intention to examine if methods I and E produce actual improvements for the various measures, not only to test for differences of the measures.

The assumption of D being normally distributed should of course be validated prior to interpreting the results. This is done using the central limit theorem (CLT). The CLT states that for a large number of samples, > 30 [7], the assumption of a normally

distributed variable is valid. When calculating confidence bounds or intervals, more than 40 examples should be used [7]. Therefore the 72 examples of the test set should suffice for the normal assumption to be valid.

6.4.1 Absolute Means

In this section the absolute performance measures (AC, TP and TN) are compared to determine any significant improvements of methods E and I , as compared to J .

Table 6.4 shows results of comparing methods using the paired t-test, described in section 6.4.

Variable (D_{XY})	AC $_{XY}$	TP $_{XY}$	TN $_{XY}$
$JI = I - J$	0 (0.082)	0 (0.530)	1 (0.022)
$IE = E - I$	1 (0.019)	1 (< 0.001)	0 (0.999)
$JE = E - J$	1 (0.009)	1 (0.020)	0 (0.326)

Table 6.4: Statistical comparison of the absolute measures. 0 denote that the mean values cannot be rejected to be equal at a 5% level, and 1 that the difference of the means is significantly positive. p -values are shown in parentheses.

0 denotes that the means cannot be rejected to be equal at a 5% level, and 1 that the difference of the means is significantly positive. The p -values are shown in parentheses. The following conclusions can be made at a significance level of $\alpha = 0.05$:

- AC, TP and FN are not significantly different when applying methods I and J .
- Method I improves the TN and FP compared to both method J and E . The latter can be seen from the p -value, of the TN of IE , being larger than 0.95.
- Method E produces significantly better AC, TP and FN, compared to both I and J .
- Method E does not produce a significantly different TN or FP, when compared to J .

The lower confidence bounds of the difference in mean values, at a 95% confidence level are shown in table 6.5. They should be interpreted as one instance of estimating the lower bound of the interval where the true mean value of the variable lies. If this is done repeatedly, 95% of the LCB's will contain the true value. If above 0, it is very likely that the true mean difference is positive, i.e. that method Y is better than method X . The LCB's show that the difference in true mean values of the absolute AC and TP for method E , are likely to be at least 1.3% above those of method J . Furthermore the difference in mean value of the absolute TN of I is likely to be at least 1.1% above that of J .

Variable (D_{XY})	AC $_{XY}$	TP $_{XY}$	TN $_{XY}$
$JI = I - J[\%]$	-0.47	-5.19	1.13
$IE = E - I[\%]$	0.35	4.69	-6.99
$JE = E - J[\%]$	1.31	1.28	-3.42

Table 6.5: Lower confidence bounds for the differences in mean values for the absolute measures, at a 95% confidence level.

6.4.2 Relative Means

The relative measures of performance are compared to determine any significant improvements.

Table 6.6 shows the results of the comparison using the paired t-test described in section 6.4. The random variable is defined as a difference between logarithms, since it corresponds to the logarithm of the relative measures, and since the logarithmic transform produces the variable D_{XY} as a *sum* of variables, (X and Y), ensuring the assumption of a normal distribution, due to the CLT, to be valid [7]. At a significance level of $\alpha = 0.05$

Variable (D_{XY})	AC $_{R,XY}$	TP $_{R,XY}$	TN $_{R,XY}$
$JI' = \ln(I) - \ln(J)$	1 (0.047)	1 (0.011)	1 (0.012)
$IE' = \ln(E) - \ln(I)$	1 (0.022)	1 (< 0.001)	0 (0.999)
$JE' = \ln(E) - \ln(J)$	1 (0.005)	1 (< 0.001)	0 (0.180)

Table 6.6: Statistical comparison of the relative measures. 0 denote that the mean values cannot be rejected to be equal at a 5% level, and 1 that the difference of the means is significantly positive. *p*-values are shown in parenthesis.

the following conclusions are made:

- Method I improves all relative measures compared to method J .
- E improves AC $_R$ and TP $_R$ compared to both I and J .
- TN $_R$ of E is significantly lower than that of I .
- There is no significant difference between TN $_R$ for J and E .

The 95% LCB's of the differences in mean values of the relative measures are shown in table 6.7. The main result is that it is likely that for future observations, the enhanced

Variable (D_{XY})	AC $_{R,XY}$	TP $_{R,XY}$	TN $_{R,XY}$
JI [%]	0.1	6.5	3.7
IE [%]	0.5	7.8	-10.9
JE [%]	3.1	20.0	-4.1

Table 6.7: Lower confidence bounds for differences in the mean values of the relative measures, at a 95% confidence level.

method, E , will produce an improvement in relative accuracy of at least 3.1%, due to an

improvement in the relative true positive rate of at least 20.0%, as compared to Javed’s method, J . Furthermore it is likely that the improvement in relative accuracy from method I to E is at least 0.5%.

6.5 Summary

In this chapter the performances on the test set of the three methods, (J, I and E), are compared. The test set consists of 72 examples independent of the examples used for optimizing the parameters of the methods. In this way the performance measures are more true estimates of how well the methods generalize. Both absolute and relative performance measures are suggested, that indicate how well the methods perform, individually, and with respect to each other.

A statistical comparison is done in three ways: Comparing methods per-example, comparing the absolute means, and comparing the relative means. The chosen level of significance is $\alpha = 0.05$. The accuracy (AC) is chosen as the most important sole measure, measuring the total detection rate. The comparison of differences in mean values, requires that the differences are normally distributed. This assumption is valid, by applying the central limit theorem, due to the size of the test set.

When comparing the methods per-example, a binomial random variable is defined as the proportion of examples that show better performance of one method as compared to another. Neither method I or E produce a significantly better or worse absolute AC, than does method J , which is used as a reference. This is partly due to the size of the test set, which influences the confidence interval (CI) severely. Method E is likely to produce a better accuracy per-example than method I , with a CI of [0.54, 0.82].

The absolute means of the performance measures are compared using a one-sided, paired, t-test, testing for a significant improvement in difference of the mean values of two methods. The mean values of the absolute AC’s are: $AC_J = 64.9\%$, $AC_I = 67.5\%$ and $AC_E = 69.2\%$. Their corresponding lower confidence bounds (LCB) of the difference in mean values are -0.47% , 0.35% and 1.31% . This means that, at the chosen level of confidence, there is no significant difference between the absolute AC of J and I . However, there is a significant difference between that of E and those of J and I respectively.

For the relative accuracy, the mean values are: $AC_{R,JI} = 13.5\%$, $AC_{R,IE} = 3.2\%$ and $AC_{R,JE} = 14.9\%$, with corresponding LCB’s of the difference in mean values: 0.1% , 0.5% and 3.1% . This shows that both methods I and E produce significantly better relative accuracy than J , and that E produces a significantly better relative accuracy than I does. For convenience, the main results are shown in table 6.8.

When analyzing the other measures of performance, the sources of the general improvements in accuracy becomes evident. Method I improves the absolute and relative detection of cast shadow pixels, TN, as compared to both J and E . However, method E improves the absolute and relative detection of object pixels even more, as compared to both J and I . This, combined with the fact that the examples generally consist of slightly more object pixels than cast shadow pixels (55%), makes method E significantly superior to the other methods.

	[%]
Mean absolute accuracy (AC) of reference method J	64.9
Mean absolute accuracy (AC) of enhanced method E	69.2
95% LCB for the absolute AC of $E - J$	1.3
Mean relative improvement in AC from J to E	14.9
95% LCB for relative AC of $E - J$	3.1
Per-example proportion of AC of E being higher than AC of J	51.4
95% CI for the per-example proportion	[36; 66]

Table 6.8: *Main results of performance. Accuracy is used as the main measure.*

Therefore the final conclusion is that the suggested improvement, I , of Javed’s method, J , is significantly better, as is the suggested enhanced method, E . In spite of the limitations of Finlayson’s suggested method for shadow removal, and of the camera used, the enhanced method for shadow removal, as suggested by the author, have proved superior in a surveillance application, as compared to the other methods implemented.

Chapter 7

Discussion

In this chapter the different parts of the thesis will be discussed in a broader perspective. The results will be compared to the system specifications, and the limitations of the methods will be used as a basis for outlining directions for future work.

7.1 Results

In section 1.3 the system specifications for handling cast shadows were outlined. This was done based on the interests of the DDRE, state-of-the-art within the area, and the requirements of a master thesis. The vast majority of goals, set forth, were achieved:

- A data set of real-world objects was obtained using a digital video camera, available by courtesy of the DDRE.
- The size of the data set ensured statistical significance in the majority of results.
- A state-of-the-art method for shadow removal, Javed’s statistical-based method, was implemented and used as a reference.
- A thorough examination of Finlayson’s physics-based approach for shadow removal was done, leading to the suggestion of an enhanced method, combining elements of Javed’s methods, with new elements.
- All methods were optimized on a training set to produce optimal performance.
- The methods were then validated on a test set to determine their generalization to unknown data.
- The suggested enhanced method for shadow removal showed significant improvements in the detection of object pixels, leading to a significant improvement in both the absolute and the relative overall accuracy. Only when comparing the accuracies per-example, using a binomial measure, was the test set too small to reveal any significant improvements.

Based on the fact that almost all specifications were met, the results are found to be successful. However, it is far from an indication of there being no work left to do.

7.2 Limitations

As emphasized during the thesis, there are several assumptions, of the methods for shadow removal, that limit their use. These limitations are discussed step-by-step in this section.

7.2.1 Data Acquisition

The acquisition of data is the initial problem of automated surveillance applications. Complex outdoor scenes make it very difficult to design algorithms that are robust with respect to illumination, weather types, etc. Therefore it was important for this thesis to obtain and use real-world data from a typical outdoor scene. This of course influenced the results, but gave a much more realistic interpretation of how state-of-the-art methods can be applied in actual applications.

The digital video camera used produced high resolution images at high frame rates. This made real-time considerations meaningless at the present stage, using a standard 3GHz PC. However, real-time implementation is possible if using e.g. dedicated hardware. More important is the limited dynamic range (8-bits) of the camera, which make bright areas saturate, while dark areas show very little response. This affects the methods for shadow removal negatively, since their assumptions are no longer valid in those cases. Furthermore the exposure of the camera was fixed manually for each recorded sequence. This is not possible to do in a real application.

7.2.2 Javed's method

The main drawbacks of Javed's method for shadow removal, J , are the context dependent parameters. The fixed variance used in the color segmentation produces large regions consisting of both actual object and actual cast shadow pixels, when areas of self shadow are adjacent to areas of cast shadow. This would not be the case with images having unlimited dynamic range, though it is a severe limitation to the method when using a standard digital video camera. The methods suggested using improved color segmentation, I and E , handle cases where Javed's method severely fails. The threshold for region merging is also context dependent, but seems to influence the performance less. The correlation threshold is the most context dependent, and therefore performance sensitive, parameter.

Javed's method fails to classify regions correctly, that contain little or no texture, since it uses the gradient directions of regions as a similarity feature.

7.2.3 Finlayson's method

Finlayson suggests deriving an image invariant to the color temperature of the illumination source. The method applies for Planckian light sources, e.g. daylight, if the camera sensor functions are narrowband, and if the surface has diffuse reflection. These assumptions are most often valid, except when the camera sensor functions are only approximately narrowband. Furthermore it is shown that the limited dynamic range of

the camera, severely degrades the quality of the illumination invariant image, making it useless for robust detection of shadow edges, using the present camera.

7.2.4 Enhanced method

The enhanced method for shadow removal, suggested by the author, handles the color segmentation problems and region classification problems of Javed's method, more robustly. However, it still has some limitations. The thresholds of the rejection class of the correlation feature, are context dependent, though the rejection class improves the robustness compared to Javed's classification. Suppressing all gradients along the edge of the foreground mask prior to reconstructing the "semi-shadow-free" image, is far from optimal. If a better shadow edge detection could be achieved, a different classification scheme would most likely improve performance.

7.3 Future Work

The limitations described in the previous sections, are the basis for a number of suggestions concerning future work within the area of shadow removal in automated surveillance applications. These suggestions are intended for the DDRE application. They are however, of interest to anyone working within this area.

In order to fully understand how the SVS-204 camera is affected by the assumptions of Finlayson's illumination invariant image, the following experiments should be performed:

- Exploiting the 10-bits dynamic range of the camera, available according to the manufacturer. At present only 8-bits are available to due limitations in the hardware.
- Investigation of the nature of the noisy diagonal pattern, which is prominent in dark regions of the illumination invariant image, cf. figure 4.5 page 33.
- Investigating the effect of the sensor functions not being ideal delta functions. This could be done in an experimental setup using an artificial Planckian light source, with variable color temperature. Additionally the effect of methods for spectral sharpening could be examined [12].

Furthermore, several parts of the methods for shadow removal should be examined in the future:

- The effect, on the performance, of large parts of an object consisting of self shadow, should be investigated further.
- If better illumination invariant images are produced, the shadow edge detection, as suggested by Finlayson, should be examined further for ways of more robust detection.
- In the classification step other similarity features should be examined for additional information. This could e.g. be a similarity feature comparing the reconstructed image with the foreground image. High similarity would indicate that no shadows

were removed, i.e. indicating that the region is part of the object. However, if the cast shadows are very weak, high similarity could also occur for regions that are part the background.

- Due to the limited amount of time outlined for the thesis, spatial and temporal features have not been examined. These would most likely improve classification. Temporal features exploit information between consecutive frames of the same object. Spatial features should be interpreted as exploiting spatial information at a local region level, i.e. between neighboring regions e.g. similar to the Markov random fields method.
- Feedback could also be introduced to improve segmentation.
- Using active shape models in addition, e.g. the snakes algorithm, could improve the segmentation further.
- Another way of improving performance could be by introducing distance measures. These could, for instance, be obtained using a pulsed laser and gated viewing. This method is also superior to thermal vision, and is able to detect shapes through fog etc. The DDRE has substantial knowledge within the area of gated viewing.
- However, applying more complex methods, introduce more parameters and is computationally expensive. To decide whether or not to seek a further improvement in shadow removal, a foreground region classifier (step 4 in figure 2.1 page 7) should be applied to determine how good a detection rate the shadow removal should produce, for obtaining satisfactory results in the foreground object classification and the later tracking algorithm.
- The fact that the performance on the training set is substantially higher, than the performance on the test set, indicates that a larger training set should be used for optimizing the methods
- If using a larger training set, it would be natural to implement a simple optimization algorithm for optimizing parameters. This would ease the computational costs of the optimization. Since we are dealing with a 2-class classification problem and not a regression problem, minimizing the *cross-entropy* error function would be more appropriate than minimizing the standard *sum-of-squares* error, as described in [2].
- Using a larger test set would decrease confidence intervals and thereby increase significance levels of the statistical tests.

7.4 Perspectives

The need for robust shadow removal in automated surveillance applications is evident. The methods described and implemented in this thesis indicate, that it is possible to design such methods that work fairly well, in an outdoor scenario, on a limited data set.

However, it is important to validate these methods on a larger data set containing other types of foreground objects, obtained in other scenarios. This is the only way to get an indication of how the methods would perform in unrestricted scenarios, on unknown object types.

Today's outdoor surveillance systems are far from perfect in the sense that they are only able to perform simple tasks, due to the high degree complexity in typical outdoor scenarios. However, improving each step in the overall surveillance system is necessary for improving the overall performance. The segmentation steps, in particular, are extremely important to improve, since all the other steps depend upon an accurate segmentation of objects. Therefore this thesis gives a minor, but important, indication of how the shadow handling can be improved, for the specific setup.

Chapter 8

Conclusion

In recent years the Danish Defense Research Establishment (DDRE) has been focusing on the area of automated outdoor video surveillance. The main effort has been on implementing and improving moving object detection and tracking, based on the W^4 -system [19]. One of the major problems in surveillance tasks, are cast shadows. They are detected as part of moving objects, which makes the tracking and classification of objects very difficult. So far the DDRE has implemented an algorithm for detection of foreground objects [18], sensitive to cast shadows. This is the basis for the thesis.

8.1 Implementation of State-of-the-Art Reference Method

A state-of-the-art method (J) for cast shadow detection, suggested by Javed et al. [21], is implemented and applied on a data set obtained with a high resolution digital video camera. The data set consists of 90 different foreground objects detected using a background subtraction algorithm supplied by the DDRE. 18 of the foreground objects constitute a training set used for manually optimizing the performance of the method, with respect to three central parameters. 72 foreground objects constitute the test set, which is used for validation of the method, i.e. to determine the methods ability to generalize to unknown data.

Javed suggests a statistical color segmentation of all pixels in an object, that are darker than the background image. Then a connected component analysis and region merging are applied, and each region is classified as part of a cast shadow, or part of an object. The classification feature applied, is the correlation of the gradient direction of pixels in each region. A simple context dependent threshold is used in the classification.

8.2 Improving Reference Method

Based on the training set, an improvement in the color segmentation of method J is suggested (method I). It improves the segmentation of large dark regions, consisting of both cast shadow pixels and object pixels. Such regions often cause method J to produce a very poor performance. The performance of method I is also manually optimized with respect to its central parameters, using the training set, and validated using the test set.

8.3 Applying Physics-Based Method

A fundamentally different approach for shadow removal is suggested by Finlayson et al. [15]. This method is physics-based, in the sense that it applies laws of physics in the derivation of a grayscale image, invariant to illumination, constrained by a number of assumptions. Ideally, this illumination invariant image does not contain edges due to shadows, only edges due to surface structures. By comparing detected edges of the illumination invariant image, with detected edges of the normal grayscale image, edges due to shadows are found. Then the gradients along the shadow edges are suppressed and a full color "shadow-free" image is reconstructed.

Finlayson's method for shadow removal has not previously been applied in surveillance applications, using a digital video camera. Therefore it is done in this thesis, to determine its applicability in a surveillance application. The conclusion is that the quality of the illumination invariant image is not good enough for a robust detection of shadow edges to be performed. This is due to artifacts introduced by the specific video camera used (SVS-204). Due to a limited dynamic range (8 bits), the areas of shadow, often were so dark that only very little response was detected by the CCD. The use of a relative measure of color in the model, makes the illumination invariant image very sensitive to noise in dark areas, which furthermore makes a robust shadow edge detection very difficult. The limited dynamic range made it difficult to ensure enough response in dark areas, while avoiding saturation of bright areas. Another assumption in the model regards the camera's spectral sensor functions, which should be narrowband. Only further experiments can show how the quality of the illumination invariant image is affected by this assumption.

Even though the elegantly derived illumination invariant image did not apply well using a camera with a dynamic range of 8 bits, the idea of suppressing gradients due to shadow edges, followed by a reconstruction of a "shadow-free" image, still applies. It is used in an enhanced method for shadow removal (method *E*), suggested in this thesis. The assumption made, is that cast shadows are almost always somehow adjacent to the edge of the foreground object detected by the background subtraction algorithm. Suppressing gradients along the border of the foreground mask, prior to the reconstruction, produce a full color "semi-shadow-free" image, where dark shadows often are significantly suppressed. The drawback of this approach is the risk of also suppressing edges due to objects, and thereby degrading the distinction between object and background. Still, in many cases there is additional information in using the "semi-shadow-free" image.

The enhanced method (*E*) suggested, combines the improved color segmentation, with the introduction of a new similarity feature (CS), based on the "semi-shadow-free" image. This new similarity feature is applied in cases where the correlation feature suggested by Javed, assigns a region to a reject class, i.e. when the correlation feature is not a confident measure. The performance of method *E* is manually optimized with respect to the central parameters, using the training set, and validated using the test set.

8.4 Final Results

The performances of the three methods are computed by comparing each foreground object with a manually segmented ground truth. The accuracy (AC) is defined as the percentage of correctly classified pixels in a foreground object, and is used as a primary measure of performance.

The main results are shown in table 8.1. The mean absolute accuracy of method E is 69.2%, 4.3 percentage point higher than that of method J . The 95% lower confidence bound (LCB) of the difference in absolute accuracy, is 1.3 percentage point. It is therefore concluded that method E produces a significantly higher mean accuracy than method J .

	[%]
Mean absolute accuracy (AC) of reference method J	64.9
Mean absolute accuracy (AC) of enhanced method E	69.2
95% LCB for the absolute AC of $E - J$	1.3
Mean relative improvement in AC from J to E	14.9
95% LCB for relative AC of $E - J$	3.1
Per-example proportion of AC of E being higher than AC of J	51.4
95% CI for the per-example proportion	[36; 66]

Table 8.1: Main results of performance (similar to table 6.8).

The mean relative improvement in AC from method J to method E , is 14.9%, with a LCB of 3.1 percentage point. This also shows a significant improvement using method E . The relative measure emphasizes absolute improvements of poor accuracies, showing that method E improves a lot upon method J , in cases where the latter severely fails.

When comparing the accuracies, per-example, using a binomial measure, indicating that the AC of E is the higher, only 51.4% of the examples in the test set show higher AC for E . Due to the number of examples, the 95% confidence interval (CI) is rather wide, and contains 50%. This shows that there is no significant increase in accuracy, per-example, by applying E instead of J .

Neither of the three implemented methods make use of any strict spatial assumptions regarding the composition of the foreground object with cast shadows. However, in cases where method E applies the enhanced classification feature, there is an assumption of the cast shadows to be adjacent to the edge of the foreground mask. This is not at all a strict assumption, since it is only applied when the correlation feature does not produce convincing classification.

8.5 Contributions

The main contributions of this thesis are summarized as follows:

- A state-of-the-art statistical-based method for shadow removal, making no spatial assumptions on the composition of objects, is implemented as reference.

- A data set of 90 relevant foreground objects is obtained, ensuring statistical significant results in two out of three overall performance measures.
- A physics-based method for shadow removal, not previously applied in a video surveillance application, is thoroughly examined, and found not to be directly applicable with the specific camera.
- An enhanced method for shadow removal is suggested, combining an improved color segmentation with the introduction of an enhanced similarity feature.
- The enhanced method for shadow removal significantly improves the mean absolute accuracy (69.2%), and mean relative accuracy (14.9%), at a 5% significance level, compared to the reference method, whose mean absolute accuracy is 64.9%.

All the specifications set forth, were achieved, except for the data set not being large enough to ensure statistical significance at a 5% level, when comparing methods per-example, and that the enhanced similarity feature makes a loose spatial assumption of object composition. However, due to the enhanced feature only being applied when the correlation feature is uncertain, the spatial assumption does not degrade performance, when compared to the reference method.

The final conclusion therefore is, that the suggested enhanced method for shadow removal, on average is better than the state-of-the-art method suggested by Javed. The enhanced method is also more robust than Javed's method, since it tends to improve the accuracy a lot, for examples where J tends to fail completely.

Combining an improved version of Javed's statistical-based method with some of the physics-based ideas of Finlayson, and some new ideas, therefore reveals a better and more robust algorithm for segmentation of cast shadows from moving objects.

The use of the illumination invariant image, suggested by Finlayson, might be able to improve the performance even more, but requires a larger dynamic range than the 8 bits currently available with the present camera provided by the DDRE.

List of Figures

1.1	Types of shadows.	2
2.1	The system architecture of W^4 [19] with additional shadow removal.	7
2.2	Flowchart of shadow removal as suggested by Javed.	10
2.3	Shadow elimination using Javed’s method [21].	11
2.4	Flowchart of shadow removal as suggested by Finlayson.	14
2.5	Finlayson’s approach to shadow removal [15].	16
2.6	Finlayson’s color calibration.	16
3.1	Pixel pattern produced by an optical Bayer filter [32].	23
3.2	Spectral response curves of the SVS-204CFCL digital camera [36].	24
3.3	Using an IR-cut filter.	24
3.4	Foreground objects used for training.	26
4.1	Illumination invariant image using spectral sensor functions.	28
4.2	Color calibration of the camera with varying types of weather.	29
4.3	Color calibration of the camera using direct sunlight only.	31
4.4	Illumination invariant image from calibration.	32
4.5	Illumination invariant sensible to dynamic range	33
5.1	Foreground detection using a kernel-based background model.	36
5.2	Manual labelling of foreground pixels.	37
5.3	ROC-curve for parameters in Javed’s method.	40
5.4	Accuracy of Javed’s method as a function of parameters.	41
5.5	Merged regions of part of the training set using Javed’s method with optimal parameters.	42
5.6	Classification of part of the training set using Javed’s method.	43
5.7	Suggested improvement of K-means color segmentation.	44
5.8	ROC-curve for parameters in Javed’s method with improved color segmentation.	45
5.9	Merged regions of part of the training set using Javed’s method with improvements and optimal parameters.	46
5.10	Classification of part of the training set using Javed’s method with improvements.	46
5.11	Detection of shadow edges.	48

5.12	Reconstruction of an image.	50
5.13	Flowchart of enhanced shadow removal as suggested in this thesis.	51
5.14	Flowchart illustrating the enhanced similarity feature.	51
5.15	Flowchart illustrating the enhanced classification of color regions.	52
5.16	ROC-curve for parameters in enhanced method.	53
5.17	Classification of part of training set using enhanced method.	54
6.1	Comparison of performance.	58
6.2	95% confidence intervals of a binomial experiment.	60
A.1	Macbeth Color Chart.	81
E.1	Flowchart of Matlab routines.	131
F.1	The system architecture of W^4 [19] with additional shadow removal (similar to figure 2.1).	191
F.2	Flowchart of shadow removal as suggested by Javed (similar to figure 2.2).	191
F.3	Flowchart of shadow removal as suggested by Finlayson (similar to figure 2.4).	192
F.4	Flowchart of enhanced shadow removal as suggested in this thesis (similar to figure 5.13).	192
F.5	Flowchart illustrating the enhanced similarity feature (similar to figure 5.14).	192
F.6	Flowchart illustrating the enhanced classification of color regions (similar to figure 5.15).	192

List of Tables

3.1	Obtaining an RGB-image from a Bayer filtered image.	23
3.2	Center frequencies of spectral response curves.	23
3.3	Data set.	25
5.1	Confusion matrix.	37
5.2	Parameter values used in optimization of Javed's method.	39
5.3	Average training performance of Javed's method.	40
5.4	Average training performance of Javed's method with improvements.	44
5.5	Parameter values used in optimization of enhanced method.	53
5.6	Average training performance of enhanced method.	54
6.1	Absolute performance of the three methods (J, I and E) based on the test set.	57
6.2	Relative improvements in performance based on the test set.	59
6.3	Comparison of methods per-example.	60
6.4	Statistical comparison of the absolute measures.	62
6.5	Lower confidence bounds for the differences in mean values of the absolute measures.	63
6.6	Statistical comparison of the relative measures.	63
6.7	Lower confidence bounds for differences in the mean values of the relative measures.	63
6.8	Main results of performance.	65
8.1	Main results of performance (similar to table 6.8).	73
A.1	Color patches of Macbeth Color Chart.	81
D.1	Absolute performance measures for the test set.	108
D.2	Relative performance measures for the test set.	109

Bibliography

- [1] Barrow, HG., Tenenbaum, JM. "Recovering Intrinsic Scene Characteristics From Images". In A.R. Hanson and E.M. Riseman, editors, *Computer Vision Systems*, pp. 3-26. Academic Press, 1978.
- [2] Bishop, CM. "Neural Networks for Pattern Recognition". Oxford University Press, 1995.
- [3] Carstensen, JM. "Image analysis, vision and computer graphics". 2nd ed., *Informat-ics and Mathematical Modelling*, Technical University of Denmark, 2002.
- [4] Chen, CC. "Illumination Invariant Image Enhancement". M.Sc. Thesis, School of Computing Science, Simon Fraser University, Canada, 2002.
- [5] Conradsen, K. "En Introduktion til Statistik - Bind 1A+1B" (in danish). 7th. ed., *Informat-ics and Mathematical Modelling*, Technical University of Denmark, 1999.
- [6] Conradsen, K. "En Introduktion til Statistik - Bind 2" (in danish). 6th. ed., *Infor-matics and Mathematical Modelling*, Technical University of Denmark, 2002.
- [7] Devore, JL. "Probability and Statistics". 5th. ed., Duxbury, Brooks/Cole, 2000.
- [8] Drew, MS., Finlayson, GD. "Spectral Sharpening with Positivity". *Journal of the Optical Society of America A*, Vol.17 no. 8, pp.1361-1370, 2000.
- [9] Drew, MS., Chen, CC., Hordley, ST. Finlayson, GD. "Sensor Transforms for Invari-ant Image Enhancement". *Color Imaging Conference*, pp.325-329, November 2002.
- [10] Duda, RO., Hart, PE. "Pattern Classification and Scene Analysis". John Wiley & Sons, 1973.
- [11] Elgammal, A., Duraiswami, R., Harwood, D., Davis, L. "Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance". *Proceedings of the IEEE*, Vol.90 no.7, pp.1151-1163, 2002.
- [12] Finlayson, GD., Drew, MS., Funt, BV. "Spectral Sharpening: Sensor Transforma-tions for Improved Color Constancy". *Journal of the Optical Society of America A*, Vol.11 no. 5, pp.1553-1563, 1994.
- [13] Finlayson, GD., Hordley, SD. "Color Constancy at a Pixel". *Journal of the Optical Society of America A*, Vol.18 no. 2, pp.253-264, 2001.

- [14] Finlayson, GD., Drew, MS. "4-Sensor Camera Calibration for Image Representation Invariant to Shading, Shadows, Lighting, and Specularities". International Conference on Computer Vision (ICCV), part II, p473-480. IEEE 2001.
- [15] Finlayson, GD., Hordley, SD., Drew, MS. "Removing Shadows from Images". European Conference on Computer Vision (ECCV), part IV, p823-836, 2002.
- [16] Gavrilu, DN., Davis, LS. "3-D model-based tracking of humans in action: a multi-view approach". Computer Vision and Pattern Recognition, Proceedings CVPR, pp. 73-80, 1996.
- [17] Hamilton, HJ. A brief overview of confusion matrices and ROC-curves. University of Regina. Available online. http://www.cs.uregina.ca/~hamilton/courses/831/notes/confusion_matrix/confusion_matrix.html
- [18] Hansen, M. "Automatic Video Surveillance". B.Sc. Thesis, Electrical Engineering (Diplomingeniør eksamensprojekt, Elektro), Ørsted•DTU, Technical University of Denmark, 2004.
- [19] Haritaoglu, I., Harwood, D., Davis, LS. "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.
- [20] Hsieh, JW., Hu, WF., Chang, CJ., Chen, YS. "Shadow elimination for effective moving object detection by Gaussian shadow modeling". Image and Vision Computing 21, pp.505-516, 2003.
- [21] Javed, O., Shah, M. "Tracking And Object Classification For Automated Surveillance". European Conference on Computer Vision (ECCV), part IV, p343-357, 2002.
- [22] Javed, O. E-mail correspondence between Mr. Javed and the author. March 2004.
- [23] Mathworks. "Matlab Release 13.1", ver. 6.5.1, 2003. Including "Image Processing Toolbox" and "Statistics Toolbox".
- [24] McKenna, SJ., Jabri, S., Duric, Z., Wechsler, H. "Tracking Interacting people". Automatic Face and Gesture Recognition, proceedings pp. 348-353, 2000.
- [25] Moeslund, TB., Granum, E. "A Survey of Computer Vision-Based Human Motion Capture". Computer Vision and Image Understanding 81, pp. 231-268, 2001.
- [26] Nadimi, S., Bhanu, B. "Moving Shadow Detection Using a Physics-based Approach". IEEE Proceedings of Pattern Recognition. Vol. 2, pp. 701-704, 2002.
- [27] Park, S., Aggarwal, JK. "Segmentation and Tracking of Interacting Human Body Parts under Occlusion and Shadowing". IEEE Proceedings of the Workshop on Motion and Video Computing. pp. 105-111, 2002.

- [28] Pilemand, T. "Video Surveillance". B.Sc. Thesis, Electrical Engineering (Diplomingeniør eksamensprojekt, Elektro), Ørsted•DTU, Technical University of Denmark, 2003.
- [29] Prati, A., Cucchiara, R., Mikic, I., Trivedi, MM. "Analysis and Detection of shadows in Video Streams: A Comparative Evaluation". IEEE Proceedings of Computer Vision and Pattern Recognition (CVPR). Vol. 2, pp. 571-576, 2001.
- [30] Prati, A., Mikic, I., Trivedi, MM., Cucchiara, R. "Detecting Moving shadows: Algorithms and Evaluation". IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 25, No. 7, pp. 918-923, 2003.
- [31] Press, WH., Teukolsky, SA., Vetterling, WT., Flannery, BP. "Numerical Recipes in C: The Art of Scientific Computing". Cambridge University Press. 2nd ed. 1992.
- [32] Ramanath, R., Snyder, WE., Bilbro, GL., Sander III, WA. "Demosaicking methods for Bayer color arrays". Journal of Electronic Imaging, 11(3), pp.306-315, July 2002.
- [33] Sidenbladh, H. De La Torre, F., Black, MJ. "A Framework for Modeling the Appearance of 3D Articulated Figures". Automatic Face and Gesture Recognition, pp. 368-375, 2000.
- [34] Siebel, NT. "Design and Implementation of People Tracking Algorithms for Visual Surveillance Applications". PhD-Thesis, Computational Vision Group, Department of Computer Science, University of Reading, March 2003.
- [35] Stauffer, C., Grimson, EL. "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.
- [36] SVS-VISTEK GmbH. SVCam SVS-204CFCL (Color Version) Users Manual. www.svs-vistek.com. Ver. 1.6, May 2003.
- [37] Weiss, Y. "Deriving Intrinsic Images from Image Sequences". IEEE Proceedings on the 8th International Conference in Computer Vision, ICCV, Vol.2, pp. 68-75, 2001.
- [38] Worthey, JA., Brill, MB. "Heuristic Analysis of von Kries Color Constancy". Journal of the Optical Society of America A, Vol.3 no. 10, pp.1708-1712, October 1986.
- [39] Wren, CR. Azarbayejani, A., Darrel, T. Pentland, AP. "Pfinder: Real-Time Tracking of the Human Body". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 7, July 1997.

Appendix A

Macbeth Color Chart



Figure A.1: Images of Macbeth Color Chart with different color temperatures of direct sunlight. (Left): Evening sun - yellowish. (Right): Midday sun - greenish.

1-Dark Skin	2-Light Skin	3-Blue Sky	4-Foliage Green	5-Blue Flower	6-Bluish Green
7-Orange	8-Purplish Blue	9-Moderate Red	10-Purple	11-Yellow Green	12-Orange Yellow
13-Blue	14-Green	15-Red	16-Yellow	17-Magenta	18-Cyan
19-White	20-Light Gray	21-Medium Gray 1	22-Medium Gray 2	23-Dark Gray	24-Black

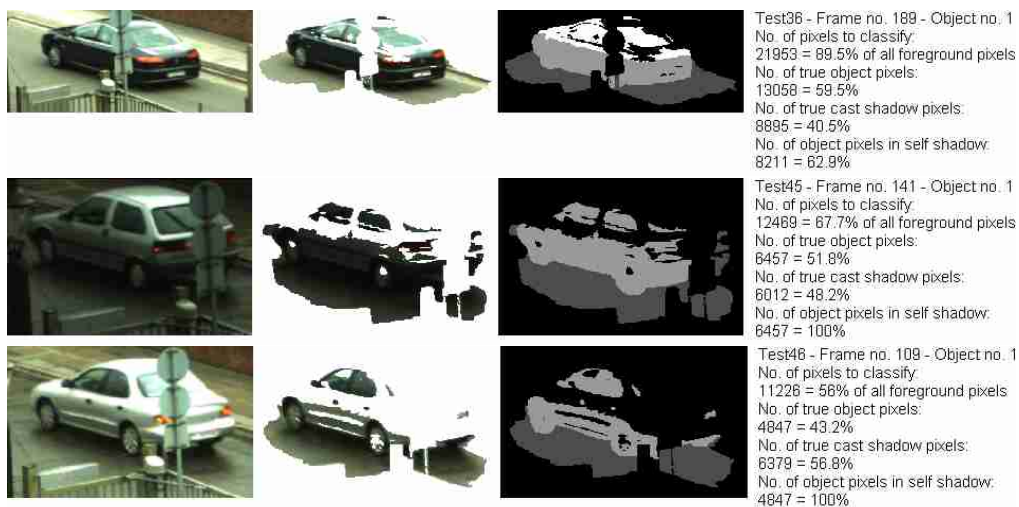
Table A.1: Numbering of color patches of Macbeth Color Chart.






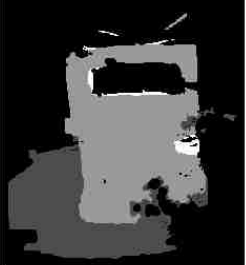















Appendix B

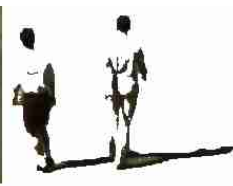
Data Sets - Foreground Objects to Classify

Every foreground object is shown in three zoomed versions: the original image (*left*), the foreground pixels (*middle*), and the manually segmented image (*right*). Foreground objects are classified into three classes: "cast shadow" (dark gray), "self shadow" (light gray), and "object not in shadow" (white). The background is black. The origin of the foreground objects can be traced by the name of the video sequence from which they are obtained (e.g. *Test36*), the frame number, and object number. Furthermore, information of absolute and relative size of classes is given: Absolute number of pixels to be classified and relative to total number of foreground pixels, absolute and relative size of true object and true cast shadow pixels, relative to the number of pixels to be classified. Finally the number of true object pixels in self shadow relative to the total number of true object pixels is given.

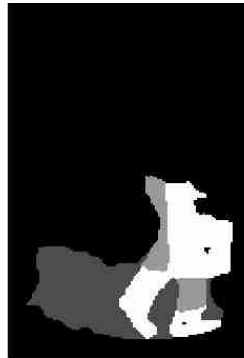
B.1 Training Set



			<p>Test53 - Frame no. 205 - Object no. 1 No. of pixels to classify: 72790 = 89.3% of all foreground pixels No. of true object pixels: 36004 = 49.5% No. of true cast shadow pixels: 36786 = 50.5% No. of object pixels in self shadow: 25786 = 71.6%</p>
			<p>Test300 - Frame no. 429 - Object no. 1 No. of pixels to classify: 49725 = 77.8% of all foreground pixels No. of true object pixels: 30537 = 61.4% No. of true cast shadow pixels: 19188 = 38.6% No. of object pixels in self shadow: 29331 = 96.1%</p>
			<p>Test420 - Frame no. 425 - Object no. 1 No. of pixels to classify: 121436 = 82.3% of all foreground pixels No. of true object pixels: 61148 = 50.4% No. of true cast shadow pixels: 60288 = 49.6% No. of object pixels in self shadow: 61148 = 100%</p>
			<p>Test429 - Frame no. 141 - Object no. 1 No. of pixels to classify: 96516 = 81.6% of all foreground pixels No. of true object pixels: 52389 = 54.3% No. of true cast shadow pixels: 44127 = 45.7% No. of object pixels in self shadow: 52389 = 100%</p>
			<p>Test432 - Frame no. 233 - Object no. 1 No. of pixels to classify: 71200 = 91.2% of all foreground pixels No. of true object pixels: 44204 = 62.1% No. of true cast shadow pixels: 26996 = 37.9% No. of object pixels in self shadow: 38652 = 87.4%</p>
			<p>Test434 - Frame no. 245 - Object no. 1 No. of pixels to classify: 89841 = 87.2% of all foreground pixels No. of true object pixels: 54198 = 60.3% No. of true cast shadow pixels: 35643 = 39.7% No. of object pixels in self shadow: 54198 = 100%</p>
			<p>Test75 - Frame no. 101 - Object no. 1 No. of pixels to classify: 7495 = 84.3% of all foreground pixels No. of true object pixels: 5167 = 68.9% No. of true cast shadow pixels: 2328 = 31.1% No. of object pixels in self shadow: 1105 = 21.4%</p>



Test352 - Frame no. 145 - Object no. 1
 No. of pixels to classify:
 5972 = 61.1% of all foreground pixels
 No. of true object pixels:
 4479 = 75%
 No. of true cast shadow pixels:
 1493 = 25%
 No. of object pixels in self shadow:
 4389 = 98%



Test411 - Frame no. 169 - Object no. 1
 No. of pixels to classify:
 5422 = 67.8% of all foreground pixels
 No. of true object pixels:
 2903 = 53.5%
 No. of true cast shadow pixels:
 2519 = 46.5%
 No. of object pixels in self shadow:
 682 = 23.8%



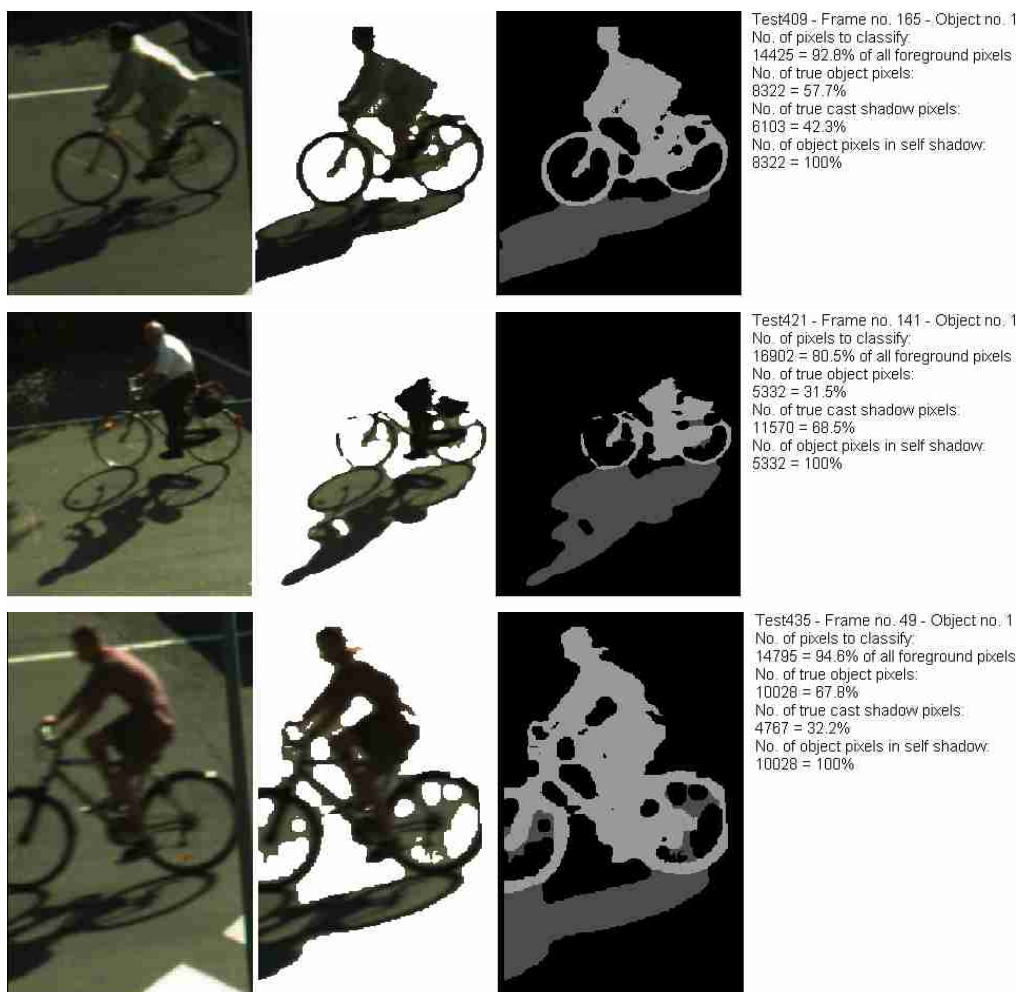
Test419 - Frame no. 297 - Object no. 1
 No. of pixels to classify:
 11631 = 96.3% of all foreground pixels
 No. of true object pixels:
 7114 = 61.2%
 No. of true cast shadow pixels:
 4517 = 38.8%
 No. of object pixels in self shadow:
 7114 = 100%



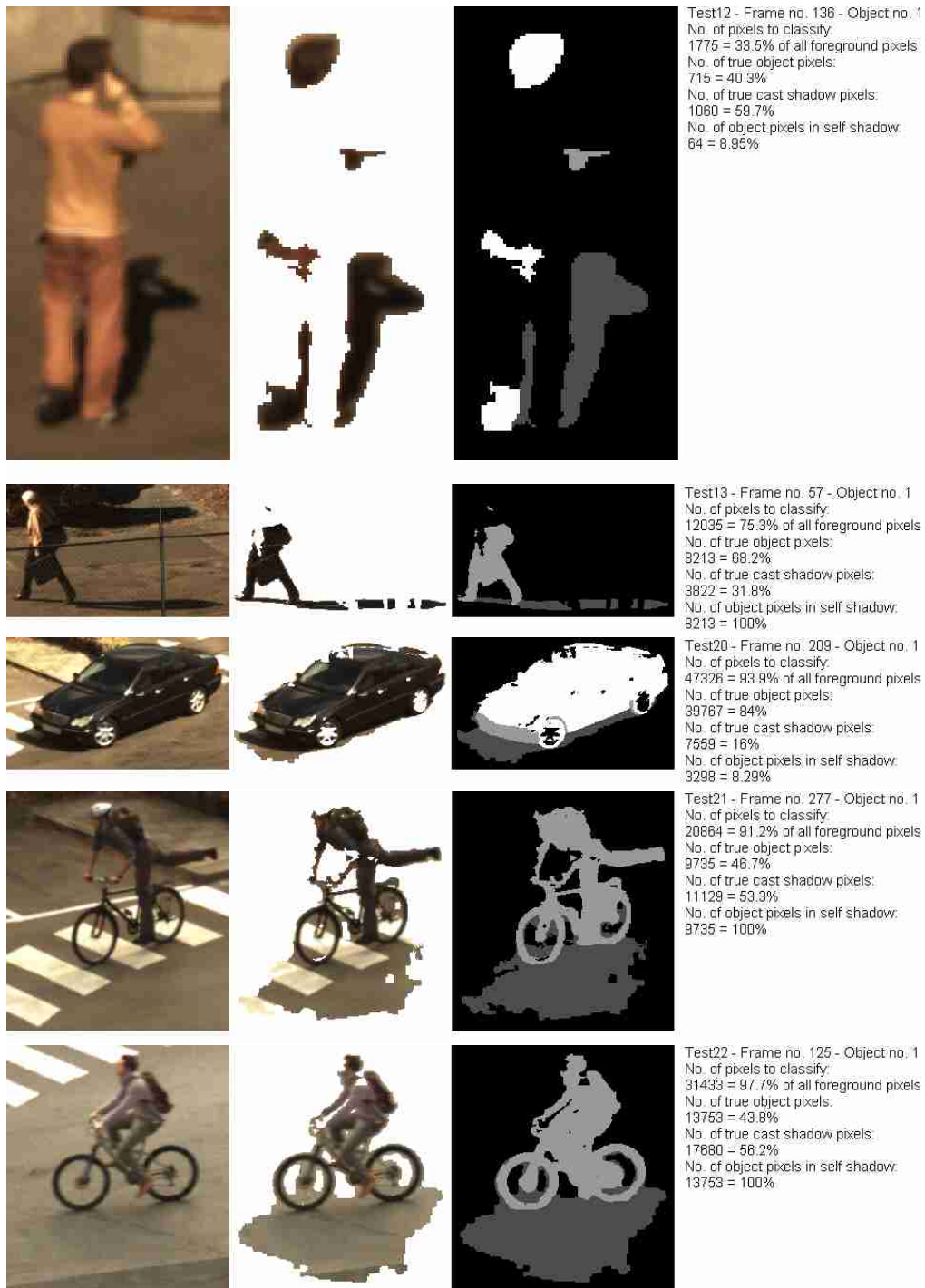
Test400 - Frame no. 101 - Object no. 1
 No. of pixels to classify:
 11418 = 90.8% of all foreground pixels
 No. of true object pixels:
 5246 = 45.9%
 No. of true cast shadow pixels:
 6172 = 54.1%
 No. of object pixels in self shadow:
 5246 = 100%









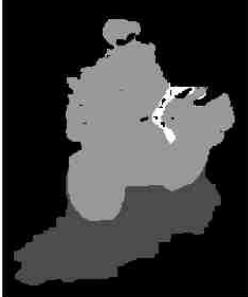











Test402 - Frame no. 89 - Object no. 1
 No. of pixels to classify:
 9074 = 86.6% of all foreground pixels
 No. of true object pixels:
 3532 = 38.9%
 No. of true cast shadow pixels:
 5542 = 61.1%
 No. of object pixels in self shadow:
 3532 = 100%



B.2 Test Set



			<p>Test23 - Frame no. 249 - Object no. 1 No. of pixels to classify: 11987 = 93.6% of all foreground pixels No. of true object pixels: 5895 = 49.2% No. of true cast shadow pixels: 6092 = 50.8% No. of object pixels in self shadow: 5895 = 100%</p>
			<p>Test24 - Frame no. 113 - Object no. 1 No. of pixels to classify: 9207 = 90.9% of all foreground pixels No. of true object pixels: 5703 = 61.9% No. of true cast shadow pixels: 3504 = 38.1% No. of object pixels in self shadow: 5703 = 100%</p>
			<p>Test24 - Frame no. 581 - Object no. 2 No. of pixels to classify: 22989 = 94.9% of all foreground pixels No. of true object pixels: 14471 = 62.9% No. of true cast shadow pixels: 8518 = 37.1% No. of object pixels in self shadow: 14057 = 97.1%</p>
			<p>Test24 - Frame no. 701 - Object no. 3 No. of pixels to classify: 78843 = 83.3% of all foreground pixels No. of true object pixels: 43883 = 57.1% No. of true cast shadow pixels: 32960 = 42.9% No. of object pixels in self shadow: 43883 = 100%</p>
			<p>Test24 - Frame no. 745 - Object no. 4 No. of pixels to classify: 58979 = 85.5% of all foreground pixels No. of true object pixels: 36042 = 61.1% No. of true cast shadow pixels: 22937 = 38.9% No. of object pixels in self shadow: 36042 = 100%</p>
			<p>Test25 - Frame no. 101 - Object no. 1 No. of pixels to classify: 92653 = 91.2% of all foreground pixels No. of true object pixels: 41804 = 45.1% No. of true cast shadow pixels: 50849 = 54.9% No. of object pixels in self shadow: 37364 = 89.4%</p>



Test25 - Frame no. 285 - Object no. 2
 No. of pixels to classify:
 66191 = 72.4% of all foreground pixels
 No. of true object pixels:
 34775 = 52.5%
 No. of true cast shadow pixels:
 31416 = 47.5%
 No. of object pixels in self shadow:
 34775 = 100%



Test27 - Frame no. 737 - Object no. 3
 No. of pixels to classify:
 19254 = 95.6% of all foreground pixels
 No. of true object pixels:
 9660 = 50.2%
 No. of true cast shadow pixels:
 9594 = 49.8%
 No. of object pixels in self shadow:
 9279 = 96.1%



Test29 - Frame no. 105 - Object no. 1
 No. of pixels to classify:
 3829 = 51.7% of all foreground pixels
 No. of true object pixels:
 1324 = 34.6%
 No. of true cast shadow pixels:
 2505 = 65.4%
 No. of object pixels in self shadow:
 1105 = 83.5%



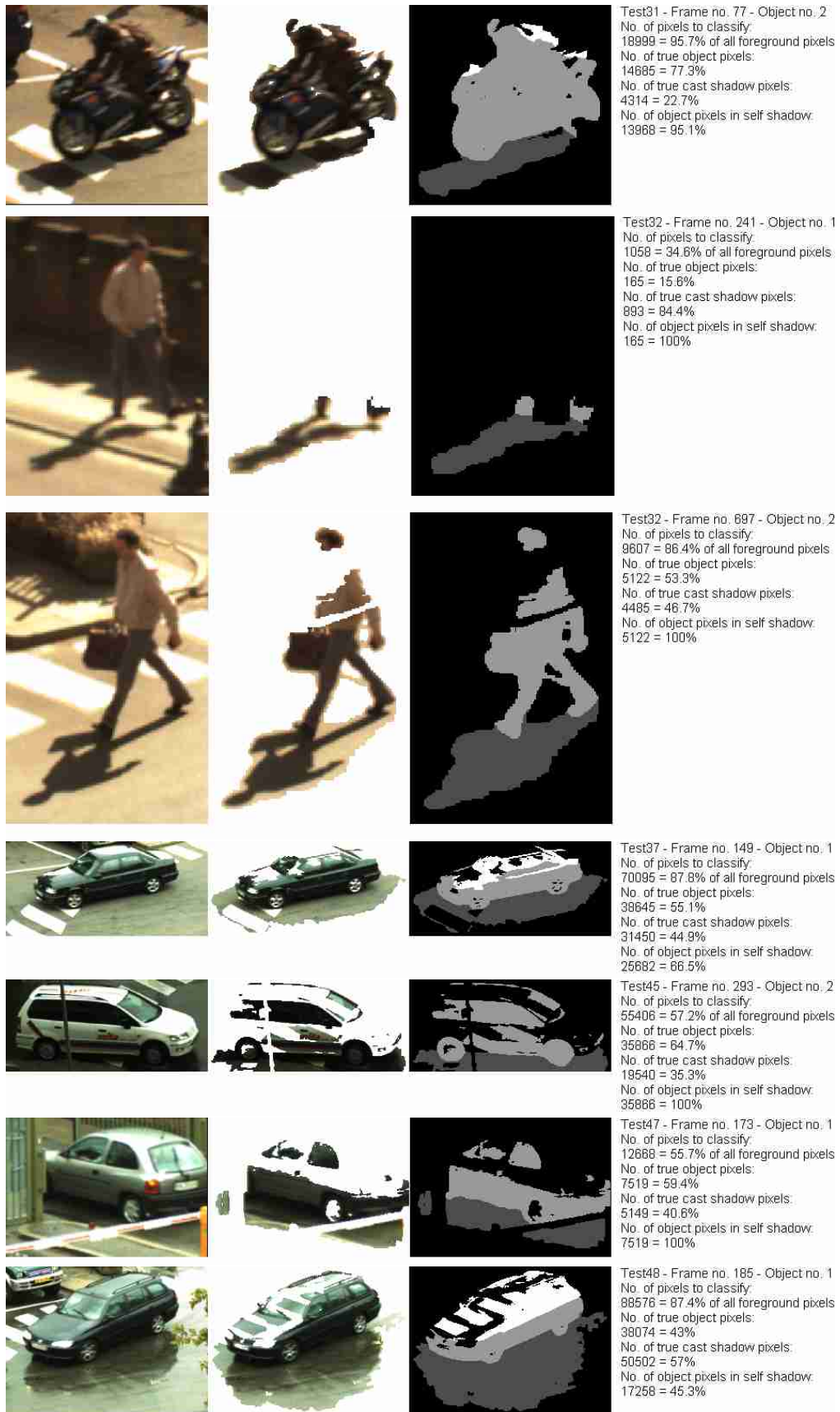
Test29 - Frame no. 141 - Object no. 2
 No. of pixels to classify:
 56391 = 78.7% of all foreground pixels
 No. of true object pixels:
 34655 = 61.5%
 No. of true cast shadow pixels:
 21736 = 38.5%
 No. of object pixels in self shadow:
 34654 = 100%



Test30 - Frame no. 61 - Object no. 1
 No. of pixels to classify:
 27338 = 98.7% of all foreground pixels
 No. of true object pixels:
 14181 = 51.9%
 No. of true cast shadow pixels:
 13157 = 48.1%
 No. of object pixels in self shadow:
 13338 = 94.1%



Test31 - Frame no. 45 - Object no. 1
 No. of pixels to classify:
 13499 = 89.5% of all foreground pixels
 No. of true object pixels:
 9199 = 68.1%
 No. of true cast shadow pixels:
 4300 = 31.8%
 No. of object pixels in self shadow:
 9199 = 100%





Test49 - Frame no. 161 - Object no. 1
 No. of pixels to classify:
 15989 = 71.3% of all foreground pixels
 No. of true object pixels:
 8302 = 51.9%
 No. of true cast shadow pixels:
 7687 = 48.1%
 No. of object pixels in self shadow:
 8302 = 100%



Test52 - Frame no. 81 - Object no. 1
 No. of pixels to classify:
 9791 = 63.4% of all foreground pixels
 No. of true object pixels:
 1135 = 11.6%
 No. of true cast shadow pixels:
 8656 = 88.4%
 No. of object pixels in self shadow:
 1135 = 100%



Test74 - Frame no. 81 - Object no. 1
 No. of pixels to classify:
 11315 = 48.7% of all foreground pixels
 No. of true object pixels:
 8242 = 72.8%
 No. of true cast shadow pixels:
 3073 = 27.2%
 No. of object pixels in self shadow:
 8242 = 100%





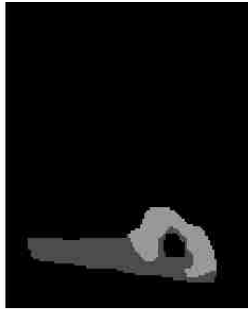













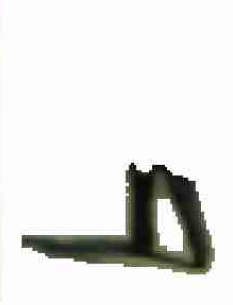
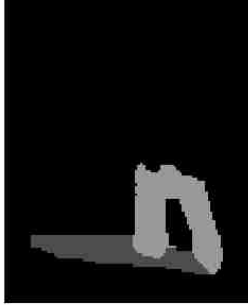



Test75 - Frame no. 181 - Object no. 2
 No. of pixels to classify:
 8991 = 58.8% of all foreground pixels
 No. of true object pixels:
 7414 = 82.5%
 No. of true cast shadow pixels:
 1577 = 17.5%
 No. of object pixels in self shadow:
 7414 = 100%

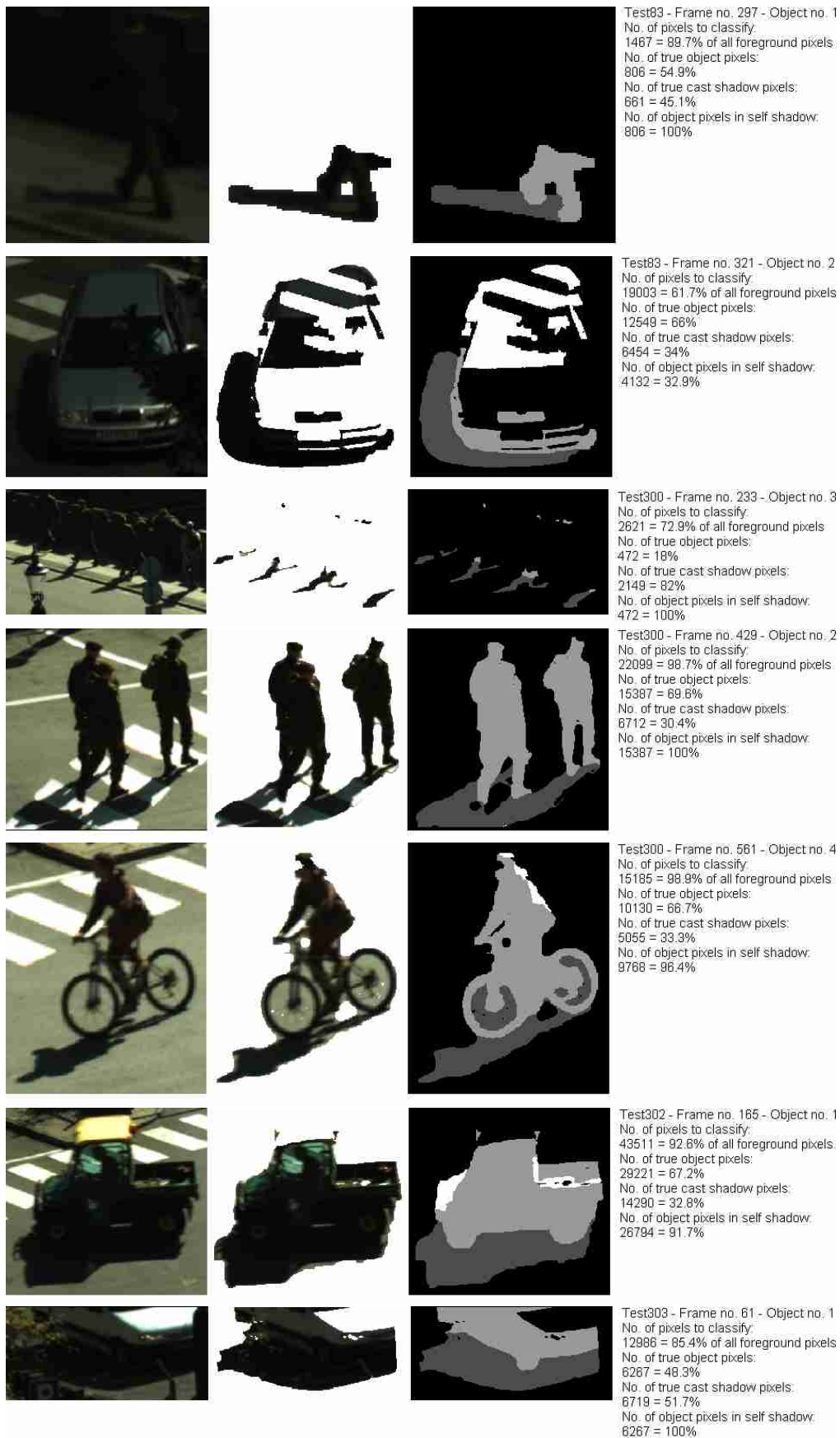


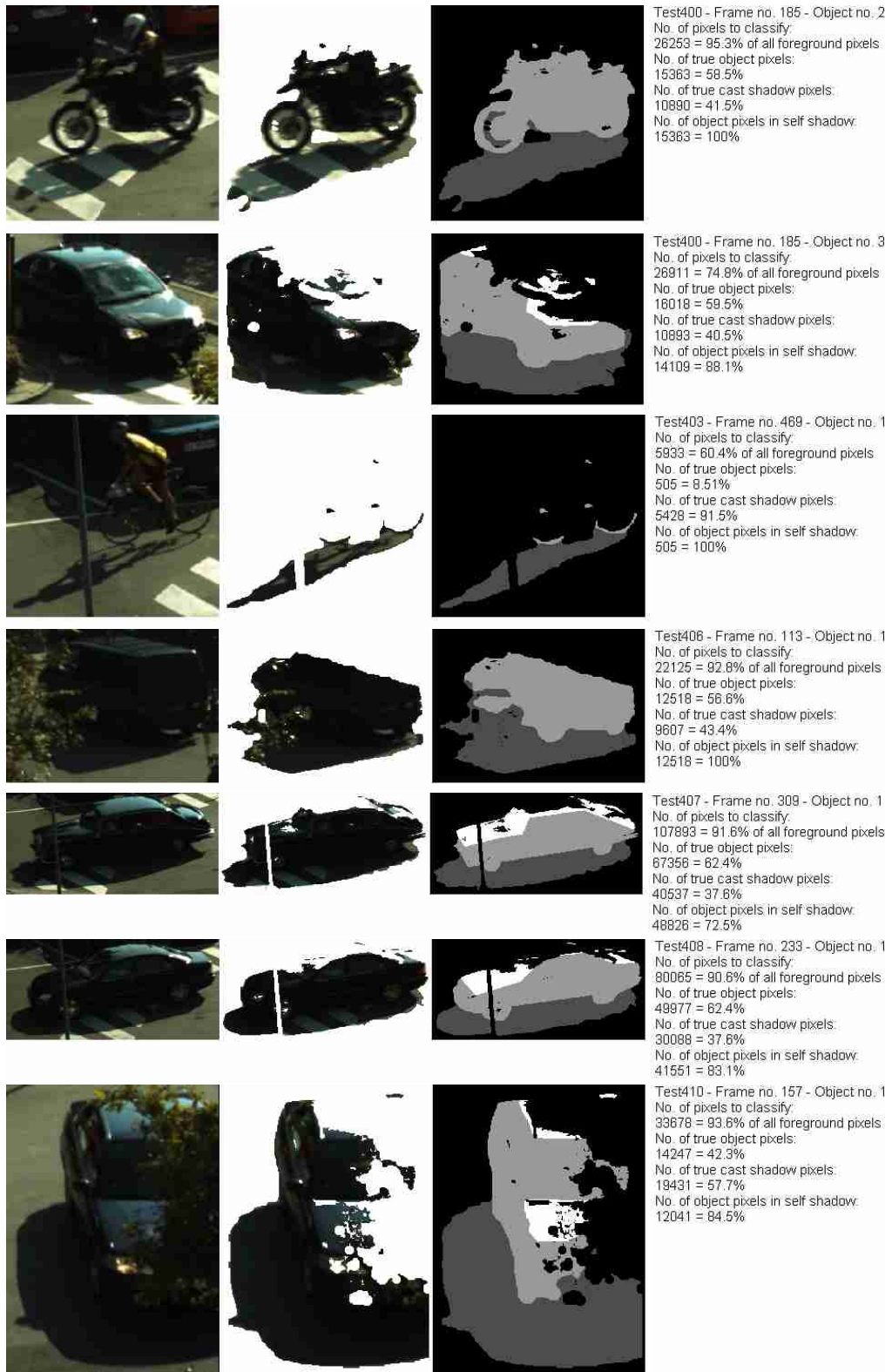
Test76 - Frame no. 93 - Object no. 1
 No. of pixels to classify:
 17366 = 95.9% of all foreground pixels
 No. of true object pixels:
 12688 = 72.9%
 No. of true cast shadow pixels:
 4688 = 27.1%
 No. of object pixels in self shadow:
 9752 = 77%

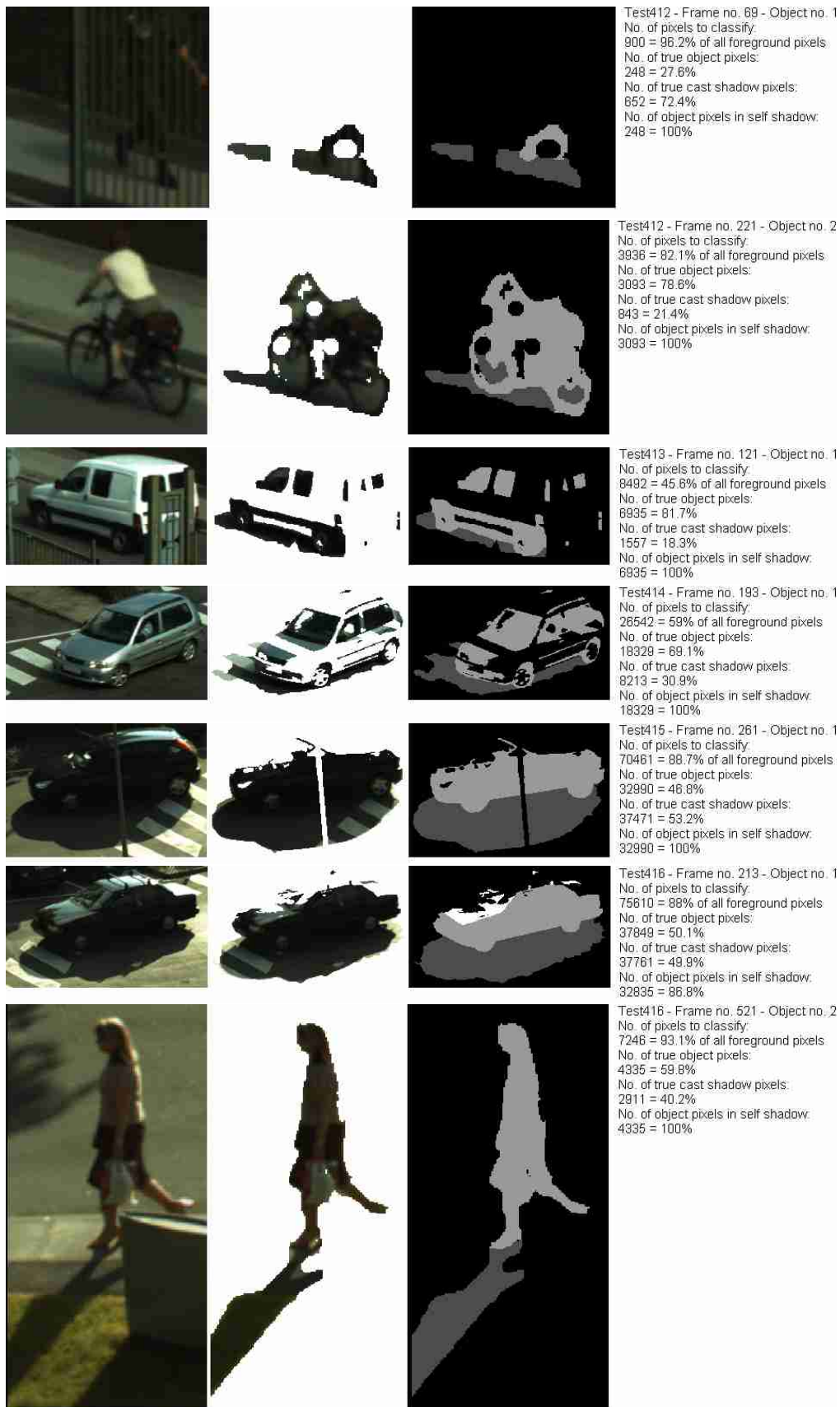







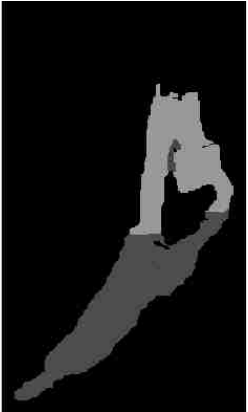















Test77 - Frame no. 105 - Object no. 1
 No. of pixels to classify:
 10189 = 78.8% of all foreground pixels
 No. of true object pixels:
 7949 = 78.2%
 No. of true cast shadow pixels:
 2220 = 21.8%
 No. of object pixels in self shadow:
 7949 = 100%

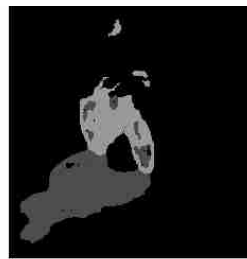
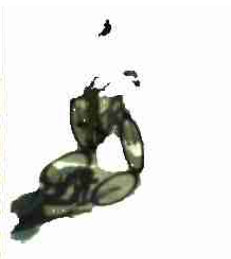
			<p>Test78 - Frame no. 141 - Object no. 1 No. of pixels to classify: 1339 = 54.7% of all foreground pixels No. of true object pixels: 513 = 38.3% No. of true cast shadow pixels: 826 = 61.7% No. of object pixels in self shadow: 513 = 100%</p>
			<p>Test79 - Frame no. 237 - Object no. 1 No. of pixels to classify: 17671 = 58.6% of all foreground pixels No. of true object pixels: 12900 = 73% No. of true cast shadow pixels: 4771 = 27% No. of object pixels in self shadow: 12900 = 100%</p>
			<p>Test80 - Frame no. 89 - Object no. 1 No. of pixels to classify: 19987 = 90.5% of all foreground pixels No. of true object pixels: 16875 = 84.4% No. of true cast shadow pixels: 3112 = 15.6% No. of object pixels in self shadow: 16875 = 100%</p>
			<p>Test81 - Frame no. 277 - Object no. 1 No. of pixels to classify: 31815 = 43.4% of all foreground pixels No. of true object pixels: 20988 = 65.9% No. of true cast shadow pixels: 10847 = 34.1% No. of object pixels in self shadow: 10794 = 51.5%</p>
			<p>Test81 - Frame no. 393 - Object no. 2 No. of pixels to classify: 3915 = 88.1% of all foreground pixels No. of true object pixels: 2568 = 65.6% No. of true cast shadow pixels: 1347 = 34.4% No. of object pixels in self shadow: 2568 = 100%</p>
			<p>Test82 - Frame no. 337 - Object no. 1 No. of pixels to classify: 1368 = 59.3% of all foreground pixels No. of true object pixels: 850 = 62.1% No. of true cast shadow pixels: 518 = 37.9% No. of object pixels in self shadow: 850 = 100%</p>
			<p>Test82 - Frame no. 361 - Object no. 2 No. of pixels to classify: 45926 = 90.7% of all foreground pixels No. of true object pixels: 38594 = 84% No. of true cast shadow pixels: 7332 = 16% No. of object pixels in self shadow: 3919 = 10.2%</p>



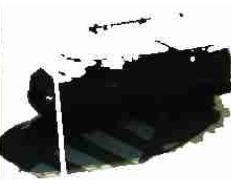




			<p>Test418 - Frame no. 69 - Object no. 1 No. of pixels to classify: 7134 = 97.8% of all foreground pixels No. of true object pixels: 4287 = 60.1% No. of true cast shadow pixels: 2847 = 39.9% No. of object pixels in self shadow: 4287 = 100%</p>
			<p>Test419 - Frame no. 501 - Object no. 2 No. of pixels to classify: 12256 = 84.2% of all foreground pixels No. of true object pixels: 4983 = 40.7% No. of true cast shadow pixels: 7273 = 59.3% No. of object pixels in self shadow: 4983 = 100%</p>
			<p>Test421 - Frame no. 169 - Object no. 2 No. of pixels to classify: 68825 = 84.6% of all foreground pixels No. of true object pixels: 36860 = 53.6% No. of true cast shadow pixels: 31965 = 46.4% No. of object pixels in self shadow: 31875 = 85.9%</p>
			<p>Test422 - Frame no. 249 - Object no. 1 No. of pixels to classify: 17226 = 84.5% of all foreground pixels No. of true object pixels: 6319 = 36.7% No. of true cast shadow pixels: 10907 = 63.3% No. of object pixels in self shadow: 6319 = 100%</p>
			<p>Test423 - Frame no. 219 - Object no. 1 No. of pixels to classify: 68851 = 92.5% of all foreground pixels No. of true object pixels: 34594 = 51.7% No. of true cast shadow pixels: 32257 = 48.3% No. of object pixels in self shadow: 27712 = 80.1%</p>
			<p>Test424 - Frame no. 237 - Object no. 1 No. of pixels to classify: 68133 = 88.7% of all foreground pixels No. of true object pixels: 39778 = 58.4% No. of true cast shadow pixels: 28355 = 41.6% No. of object pixels in self shadow: 34712 = 87.3%</p>
			<p>Test425 - Frame no. 145 - Object no. 1 No. of pixels to classify: 72967 = 80.8% of all foreground pixels No. of true object pixels: 40788 = 55.9% No. of true cast shadow pixels: 32179 = 44.1% No. of object pixels in self shadow: 34467 = 84.5%</p>



Test426 - Frame no. 85 - Object no. 1
 No. of pixels to classify:
 8111 = 79.2% of all foreground pixels
 No. of true object pixels:
 2771 = 34.2%
 No. of true cast shadow pixels:
 5340 = 65.8%
 No. of object pixels in self shadow:
 2771 = 100%



Test428 - Frame no. 193 - Object no. 1
 No. of pixels to classify:
 101184 = 85.7% of all foreground pixels
 No. of true object pixels:
 52558 = 51.9%
 No. of true cast shadow pixels:
 48626 = 48.1%
 No. of object pixels in self shadow:
 52558 = 100%



Test430 - Frame no. 177 - Object no. 1
 No. of pixels to classify:
 20383 = 96% of all foreground pixels
 No. of true object pixels:
 9362 = 45.9%
 No. of true cast shadow pixels:
 11021 = 54.1%
 No. of object pixels in self shadow:
 8828 = 94.3%



Test431 - Frame no. 137 - Object no. 1
 No. of pixels to classify:
 25111 = 97.1% of all foreground pixels
 No. of true object pixels:
 13139 = 52.3%
 No. of true cast shadow pixels:
 11972 = 47.7%
 No. of object pixels in self shadow:
 13139 = 100%



Test433 - Frame no. 109 - Object no. 1
 No. of pixels to classify:
 3785 = 81.9% of all foreground pixels
 No. of true object pixels:
 1047 = 27.7%
 No. of true cast shadow pixels:
 2738 = 72.3%
 No. of object pixels in self shadow:
 1047 = 100%



Test434 - Frame no. 465 - Object no. 2
 No. of pixels to classify:
 13331 = 46.4% of all foreground pixels
 No. of true object pixels:
 0 = 0%
 No. of true cast shadow pixels:
 13331 = 100%
 No. of object pixels in self shadow:
 0 = NaN%

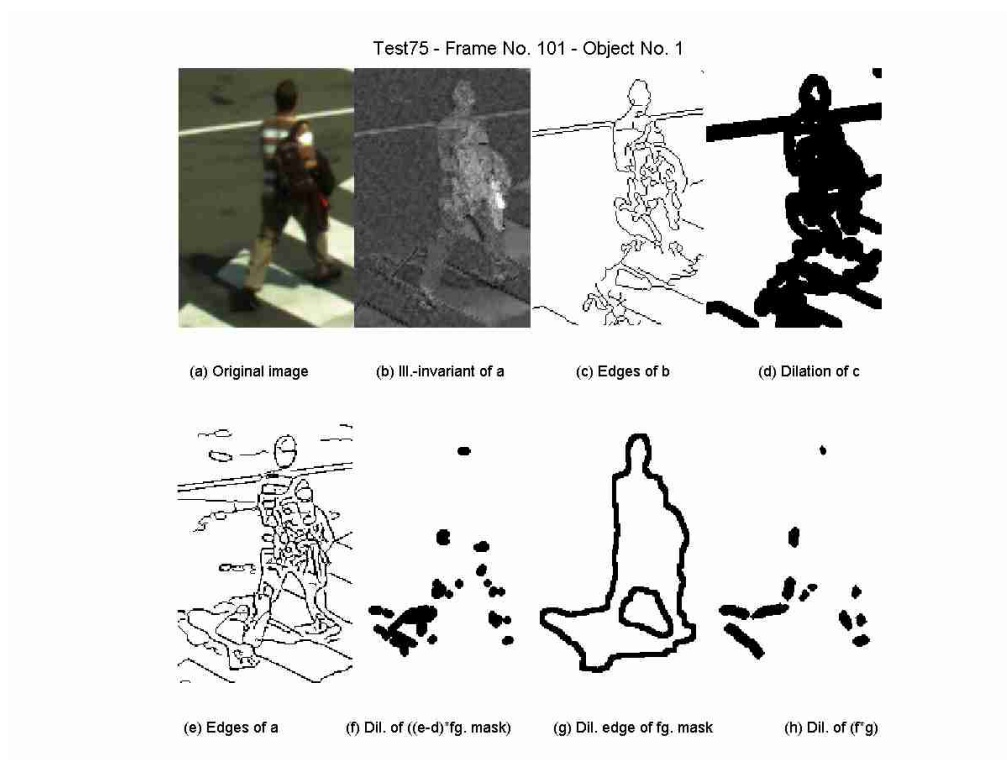


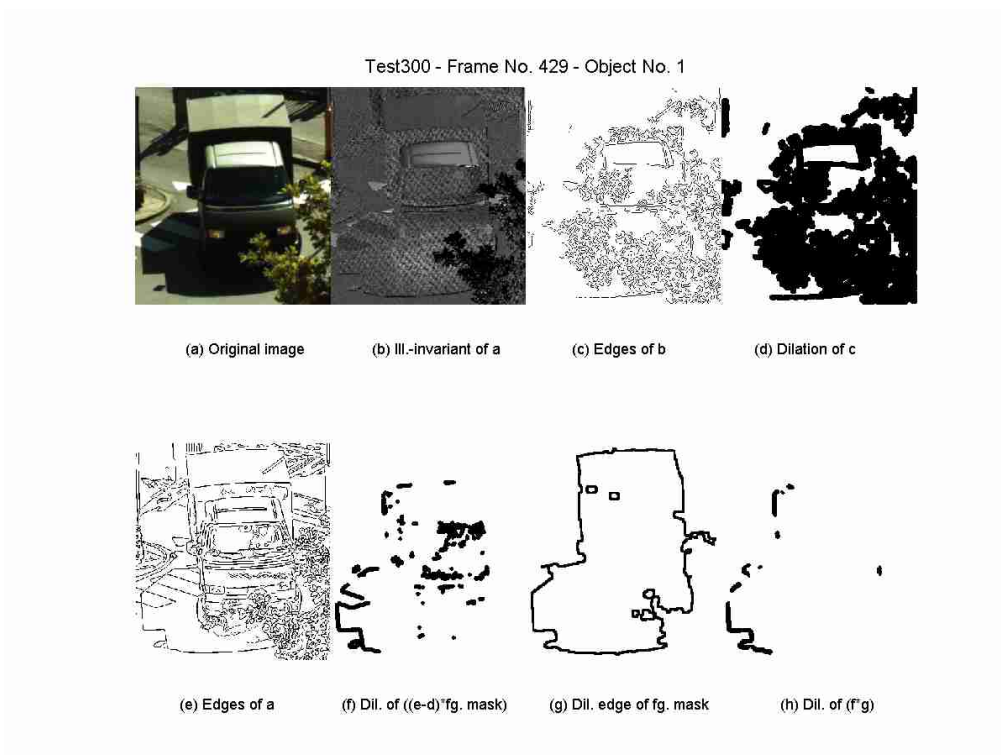
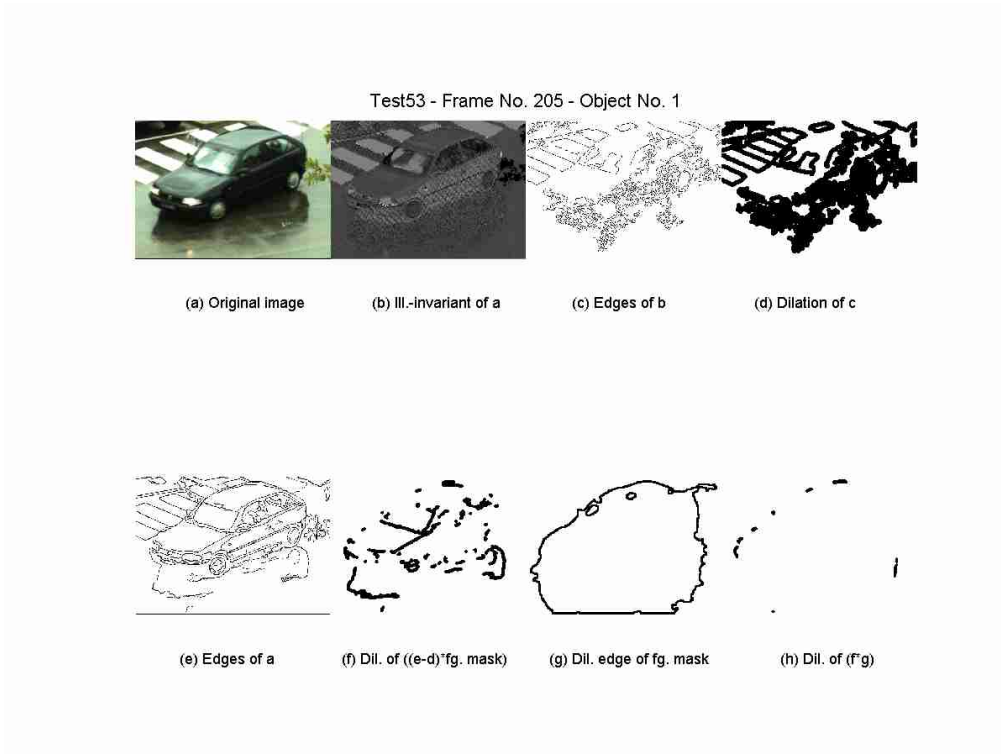
Test435 - Frame no. 297 - Object no. 2
 No. of pixels to classify:
 61966 = 82.8% of all foreground pixels
 No. of true object pixels:
 34205 = 55.2%
 No. of true cast shadow pixels:
 27761 = 44.8%
 No. of object pixels in self shadow:
 34205 = 100%

Appendix C

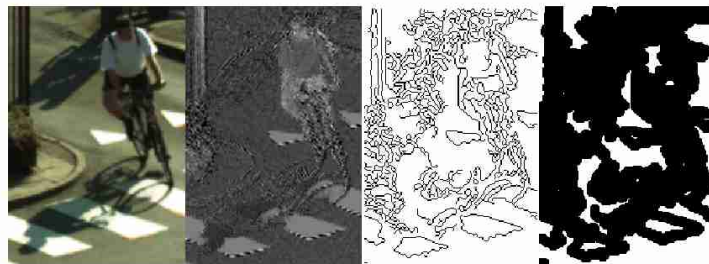
Additional Figures

C.1 Detecting Shadow Edges from Illumination-Invariants





Test402 - Frame No. 89 - Object No. 1

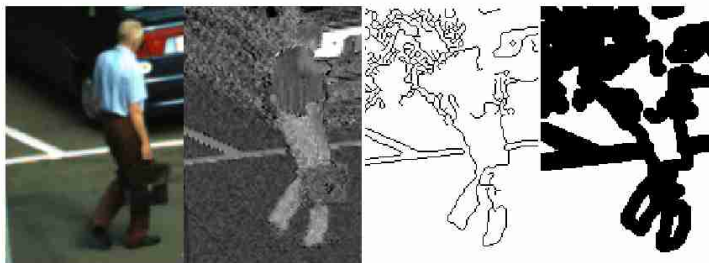


(a) Original image (b) Ill.-invariant of a (c) Edges of b (d) Dilation of c



(e) Edges of a (f) Dil. of ((e-d)*fg. mask) (g) Dil. edge of fg. mask (h) Dil. of (f*g)

Test411 - Frame No. 169 - Object No. 1



(a) Original image (b) Ill.-invariant of a (c) Edges of b (d) Dilation of c



(e) Edges of a (f) Dil. of ((e-d)*fg. mask) (g) Dil. edge of fg. mask (h) Dil. of (f*g)

Appendix D

Additional Results

Several results are shown for every detected foreground object of the training and test sets. 6 subfigures are shown to visualize various steps, and differences in methods, for each example:

Upper left: Merged regions using Javed’s method, J .

Upper middle: Merged regions using improved color segmentation, method I .

Upper right: Reconstructed image, where gradients along the edge of the foreground object are suppressed.

Lower left: Segmentation using Javed’s method, J .

Lower middle: Segmentation using improved color segmentation, method I .

Lower right: Segmentation using enhanced method, E .

Furthermore are shown the number of merged regions using Javed’s and the improved color segmentation, and also the absolute performance measures, in %, for the three methods:

AC: Accuracy - Proportion of correctly classified pixels.

TP: True positive rate - Correctly classified object pixels.

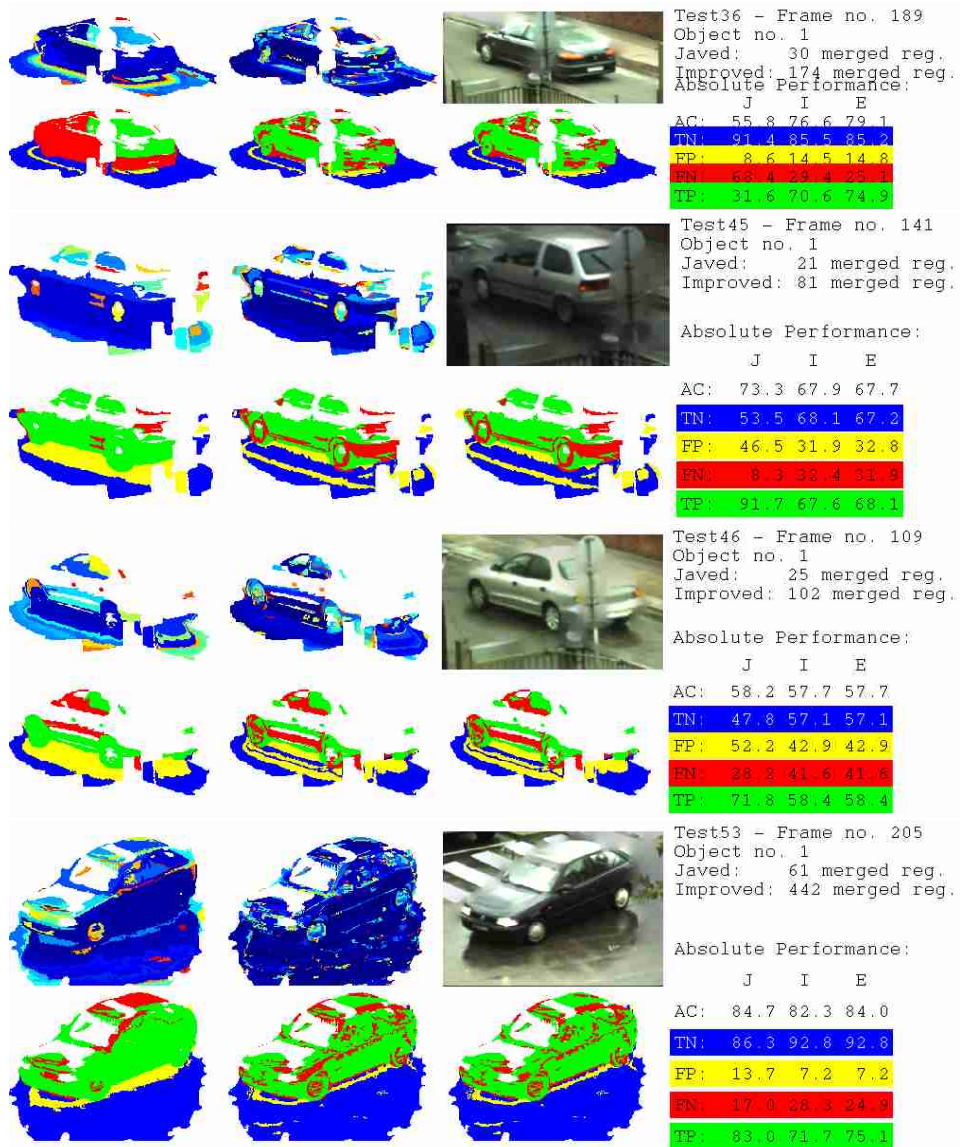
FP: False positive rate - Actual cast shadow pixels classified incorrectly as object pixels.

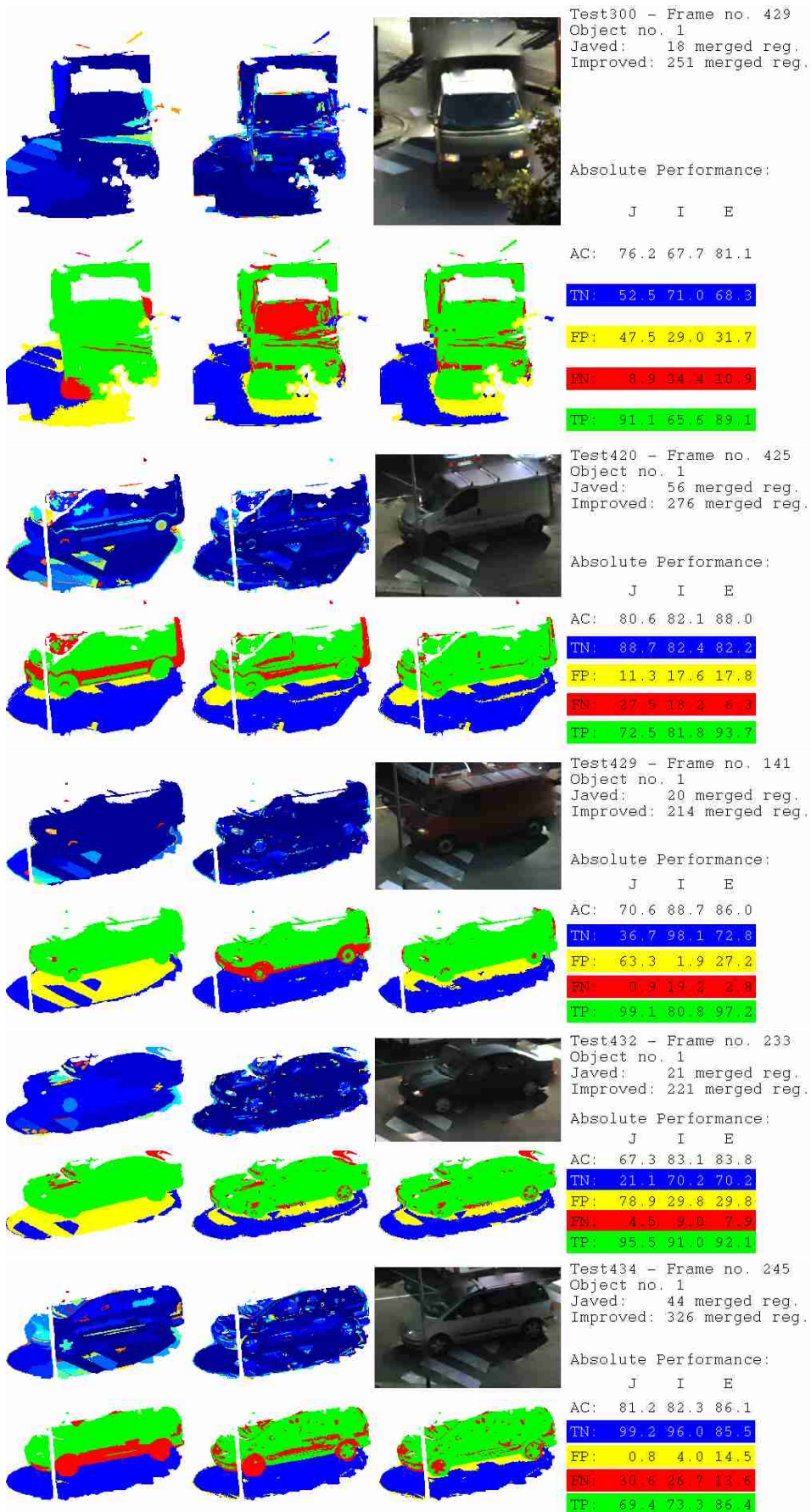
FN: False negative rate - Actual object pixels classified incorrectly as cast shadow pixels.

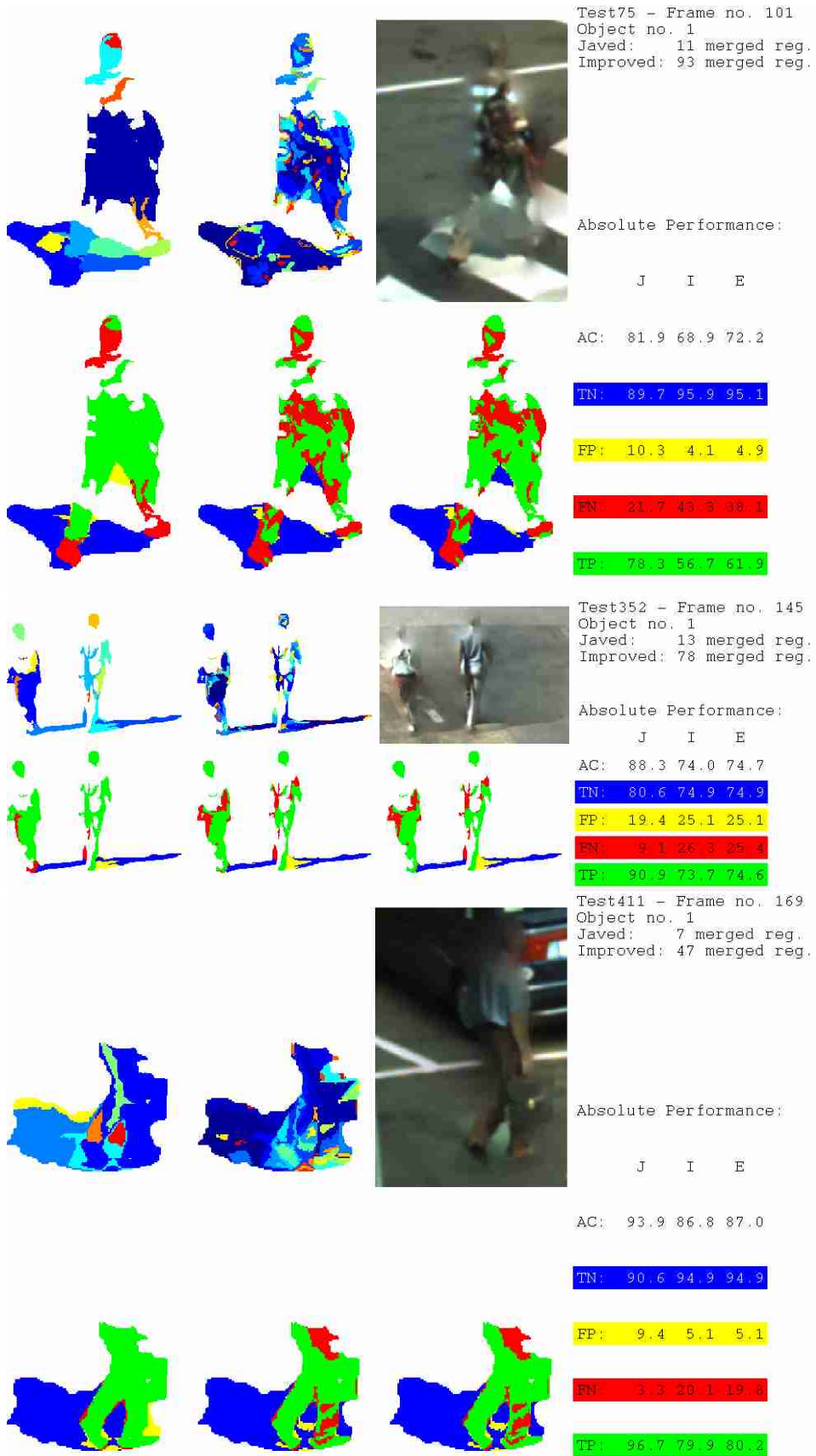
TN: True negative rate - Correctly classified cast shadow pixels.

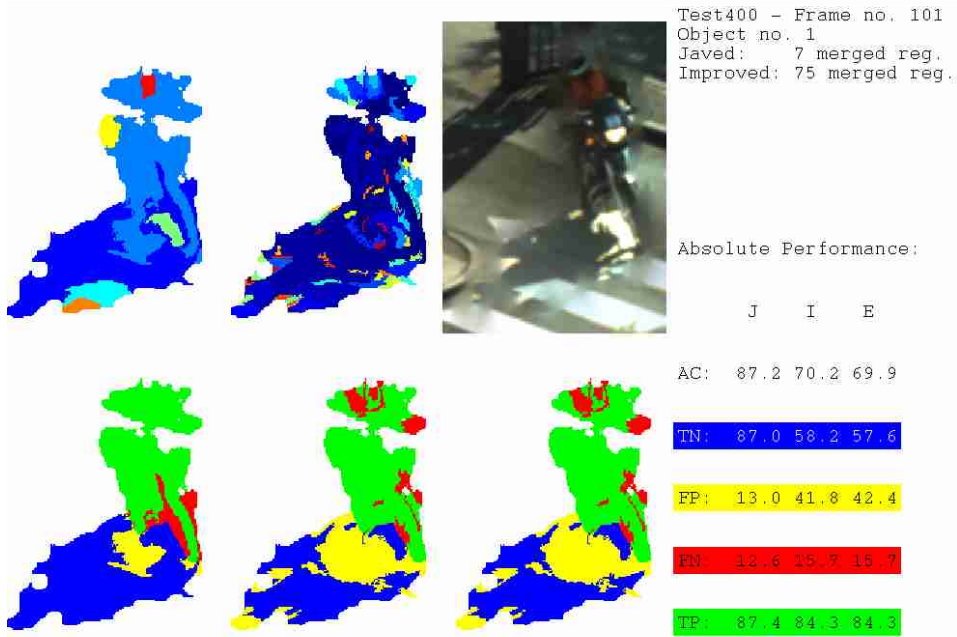
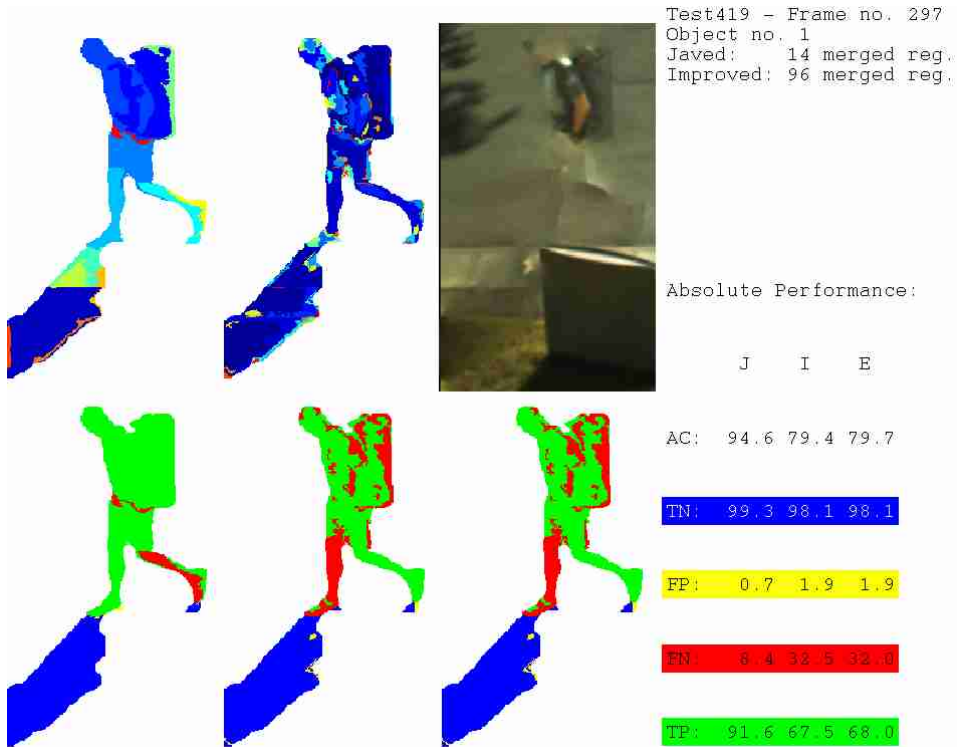
The following colors are used to visualize the performance measures: [*blue, yellow, red, green*] denote [TN, FP, FN, TP] respectively. The validation of methods is based on the test set, for which reason both absolute and relative measures for the test set are shown in tables D.1 and D.2. The relative measures are defined as relative improvements, in %, in performance, when using e.g. method E instead of method J . As an example the relative improvement in accuracy, when using method E instead of method J , is denoted $AC_{R,JE}$.

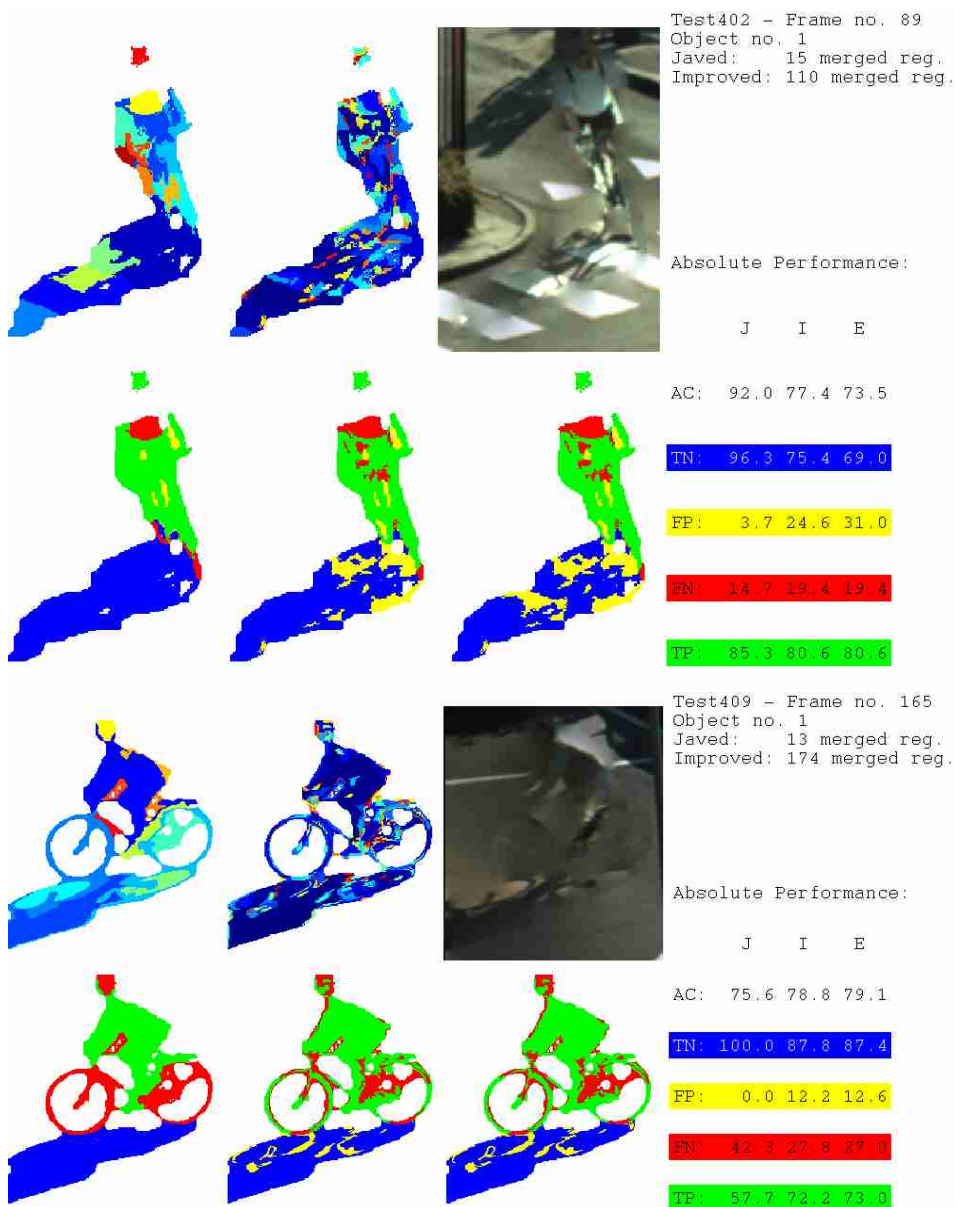
D.1 Performance of Training Set

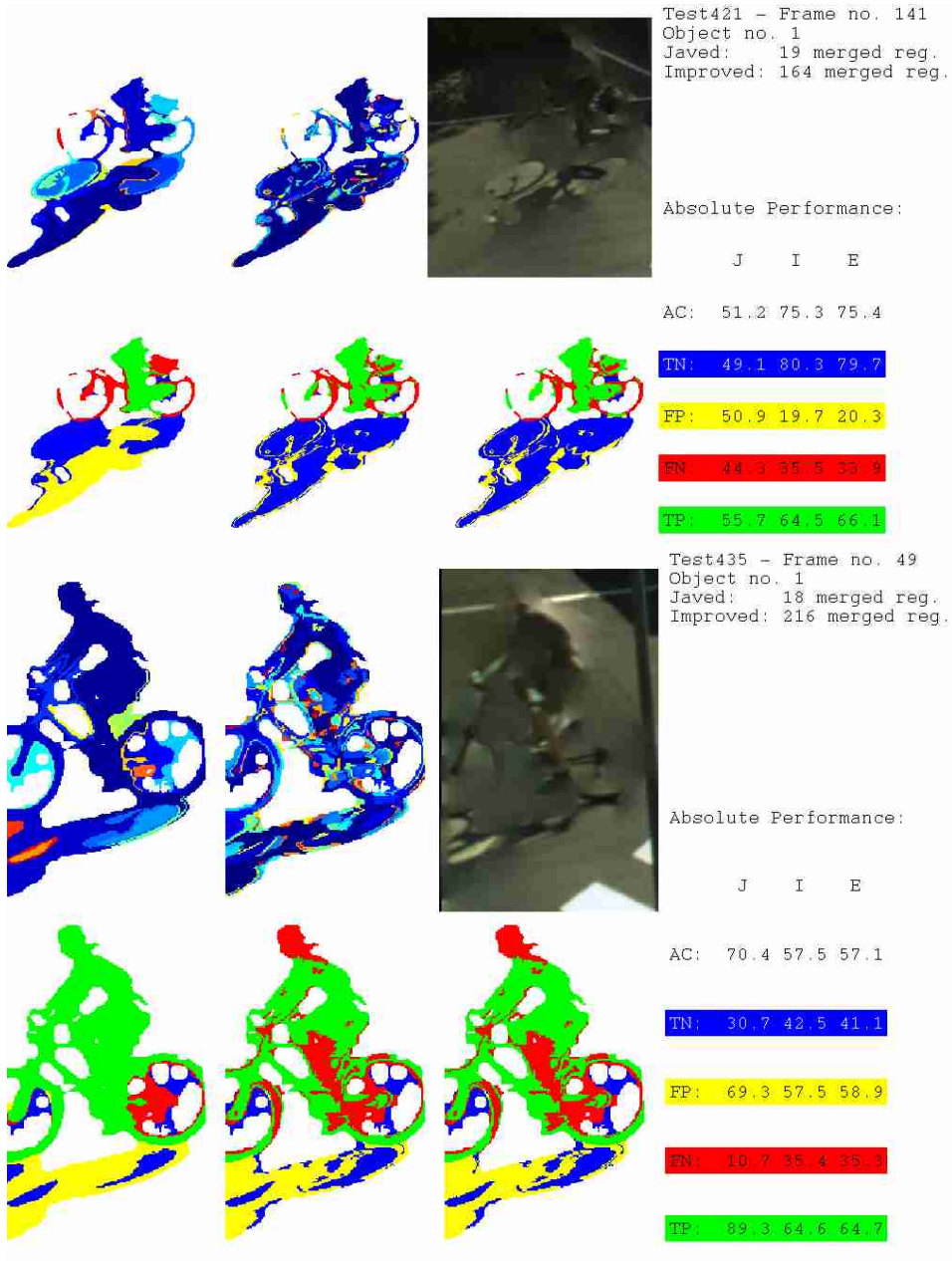












D.2 Performance of Test Set

Abs. performance	AC _{J I E} [%]	TP _{J I E} [%]	FP _{J I E} [%]	FN _{J I E} [%]	TN _{J I E} [%]
Test12-136-1	91.4 71.5 71.5	100.0 75.1 75.1	14.3 30.9 30.9	0.0 24.9 24.9	85.7 69.1 69.1
Test13-57-1	27.4 43.1 45.7	20.4 38.7 42.8	57.6 47.6 48.2	79.6 61.3 57.2	42.4 52.4 51.8
Test20-209-1	80.7 66.4 74.2	90.0 69.4 80.1	68.4 49.0 56.4	10.0 30.6 19.9	31.6 51.0 43.6
Test21-277-1	57.9 78.5 80.2	9.7 59.9 64.4	0.0 5.3 6.0	90.3 40.1 35.6	100.0 94.7 94.0
Test22-125-1	80.3 79.1 79.4	55.1 52.9 53.6	0.0 0.4 0.6	44.9 47.1 46.4	100.0 99.6 99.4
Test23-249-1	82.9 79.9 80.1	67.4 62.9 63.4	2.0 3.6 3.7	32.6 37.1 36.6	98.0 96.4 96.3
Test24-113-1	61.9 49.7 60.3	38.7 29.9 48.8	0.3 18.0 21.1	61.3 70.1 51.2	99.7 82.0 78.9
Test24-581-2	72.2 65.0 70.4	76.2 63.0 81.3	34.7 31.7 48.0	23.8 37.0 18.7	65.3 68.3 52.0
Test24-701-3	61.0 68.2 64.5	84.1 73.8 87.0	69.8 39.3 65.5	15.9 26.2 13.0	30.2 60.7 34.5
Test24-745-4	71.2 70.5 65.1	60.0 58.4 86.0	11.2 10.4 67.9	40.0 41.6 14.0	88.8 89.6 32.1
Test25-101-1	86.9 80.8 86.6	88.0 70.7 90.7	14.1 10.9 16.8	12.0 29.3 9.3	85.9 89.1 83.2
Test25-285-2	87.1 81.5 85.1	89.3 78.4 85.4	15.3 15.2 15.2	10.7 21.6 14.6	84.7 84.8 84.8
Test27-737-3	87.0 69.9 81.0	75.1 43.4 65.5	1.0 3.4 3.4	24.9 56.6 34.5	99.0 96.6 96.6
Test29-105-1	82.0 63.3 64.6	57.2 27.5 32.5	4.9 17.8 18.5	42.8 72.5 67.5	95.1 82.2 81.5
Test29-141-2	68.7 76.0 73.6	89.6 75.8 89.4	69.7 23.6 55.4	10.4 24.2 10.6	30.3 76.4 44.6
Test30-61-1	97.5 86.8 86.5	100.0 81.5 81.5	5.3 7.5 8.2	0.0 18.5 18.5	94.7 92.5 91.8
Test31-45-1	54.3 54.3 55.6	33.0 45.9 51.8	0.0 27.8 36.3	67.0 54.1 48.2	100.0 72.2 63.7
Test31-77-2	74.0 68.4 76.2	81.1 66.1 77.1	50.0 23.6 27.1	18.9 33.9 22.9	50.0 76.4 72.9
Test32-241-1	15.6 85.9 56.7	100.0 49.1 49.7	100.0 7.3 42.0	0.0 50.9 50.3	0.0 92.7 58.0
Test32-697-2	94.1 68.7 63.0	90.8 46.9 80.5	2.1 6.5 57.0	9.2 53.1 19.5	97.9 93.5 43.0
Test37-149-1	89.7 82.5 84.6	88.1 76.3 80.1	8.5 9.9 9.9	11.9 23.7 19.9	91.5 90.1 90.1
Test45-293-2	67.2 64.2 73.1	78.2 67.9 81.7	53.2 42.7 42.7	21.8 32.1 18.3	46.8 57.3 57.3
Test47-173-1	48.6 55.4 55.6	17.7 27.0 27.3	6.3 3.1 3.1	82.3 73.0 72.7	93.7 96.9 96.9
Test48-185-1	87.6 82.1 84.6	86.1 70.2 76.2	11.4 8.9 9.0	13.9 29.8 23.8	88.6 91.1 91.0
Test49-161-1	76.8 74.3 75.2	77.9 72.1 74.3	24.5 23.2 23.8	22.1 27.9 25.7	75.5 76.8 76.2
Test52-61-1	61.6 61.1 61.0	22.8 55.3 55.3	33.3 38.1 38.3	77.2 44.7 44.7	66.7 61.9 61.7
Test74-81-1	61.2 60.0 61.0	54.9 51.8 53.2	22.0 18.1 18.1	45.1 48.2 46.8	78.0 81.9 81.9
Test75-181-2	55.6 61.8 62.2	46.2 67.8 68.2	0.3 66.1 66.1	53.8 32.2 31.8	99.7 33.9 33.9
Test76-93-1	58.8 55.2 60.2	72.6 66.5 73.8	78.5 75.5 76.4	27.4 33.5 26.2	21.5 24.5 23.6
Test77-105-1	37.3 43.5 68.9	22.7 32.4 65.0	10.8 16.8 17.2	77.3 67.6 35.0	89.2 83.2 82.8
Test78-141-1	38.3 75.3 75.3	100.0 53.2 53.2	100.0 11.0 11.0	0.0 46.8 46.8	0.0 89.0 89.0
Test79-237-1	41.1 50.8 50.4	34.7 42.8 42.8	41.4 27.5 29.0	65.3 57.2 57.2	58.6 72.5 71.0
Test80-69-1	25.9 65.4 70.0	12.2 65.0 70.5	0.0 32.3 32.3	87.8 35.0 29.5	100.0 67.7 67.7
Test81-277-1	74.7 67.3 68.4	67.9 57.9 59.7	12.1 14.7 14.7	32.1 42.1 40.3	87.9 85.3 85.3
Test81-393-2	69.9 63.2 63.2	81.9 69.5 69.5	52.9 48.9 48.9	18.1 30.5 30.5	47.1 51.1 51.1
Test82-337-1	62.1 51.8 53.9	100.0 64.9 68.4	100.0 69.9 69.9	0.0 35.1 31.6	0.0 30.1 30.1
Test82-361-2	55.6 65.8 74.6	56.1 68.6 79.1	46.9 48.9 49.0	43.9 31.4 20.9	53.1 51.1 51.0
Test83-297-1	54.9 37.8 37.8	100.0 68.6 68.6	100.0 99.8 99.8	0.0 31.4 31.4	0.0 0.2 0.2
Test83-321-2	39.8 30.4 30.4	60.2 45.8 45.9	100.0 99.6 99.6	39.8 54.2 54.1	0.0 0.4 0.4
Test300-233-3	85.8 81.0 79.5	21.2 80.7 80.7	0.0 18.9 20.8	78.8 19.3 19.3	100.0 81.1 79.2
Test300-429-2	47.4 50.4 82.3	40.2 39.4 85.6	36.2 24.2 25.3	59.8 60.6 14.4	63.8 75.8 74.7
Test300-561-4	71.6 74.2 63.7	90.2 67.6 73.1	65.9 12.5 55.0	9.8 32.4 26.9	34.1 87.5 45.0
Test302-165-1	61.7 58.5 59.3	86.8 82.2 83.6	89.7 90.1 90.4	13.2 17.8 16.4	10.3 9.9 9.6
Test303-61-1	40.3 39.3 39.4	83.4 76.2 76.5	100.0 95.2 95.2	16.6 23.8 23.5	0.0 4.8 4.8
Test400-185-2	79.0 85.7 87.8	65.9 84.5 88.3	2.5 12.6 12.9	34.1 15.5 11.7	97.5 87.4 87.1
Test400-185-3	68.3 74.0 75.3	96.4 87.7 89.8	73.0 46.1 46.1	3.6 12.3 10.2	27.0 53.9 53.9
Test403-469-1	58.2 80.4 82.5	46.7 28.5 54.3	40.8 14.8 14.8	53.3 71.5 45.7	59.2 85.2 85.2
Test406-113-1	58.5 73.7 73.7	100.0 97.6 97.8	95.5 57.5 57.7	0.0 2.4 2.2	4.5 42.5 42.3
Test407-309-1	71.4 79.0 82.0	97.1 85.0 90.0	71.3 31.0 31.2	2.9 15.0 10.0	28.7 69.0 68.8
Test408-233-1	68.1 76.5 79.3	96.7 86.3 90.9	79.3 39.8 39.9	3.3 13.7 9.1	20.7 60.2 60.1
Test410-157-1	59.5 64.3 64.5	5.9 49.0 49.7	1.2 24.6 24.7	94.1 51.0 50.3	98.8 75.4 75.3
Test412-69-1	22.7 39.1 36.3	13.7 52.4 52.4	73.9 66.0 69.8	86.3 47.6 47.6	26.1 34.0 30.2
Test412-221-2	32.4 52.5 53.7	14.0 43.9 45.5	0.0 16.0 16.0	86.0 56.1 54.5	100.0 84.0 84.0
	<i>Continued</i>	<i>on</i>	<i>next</i>	<i>page</i>	

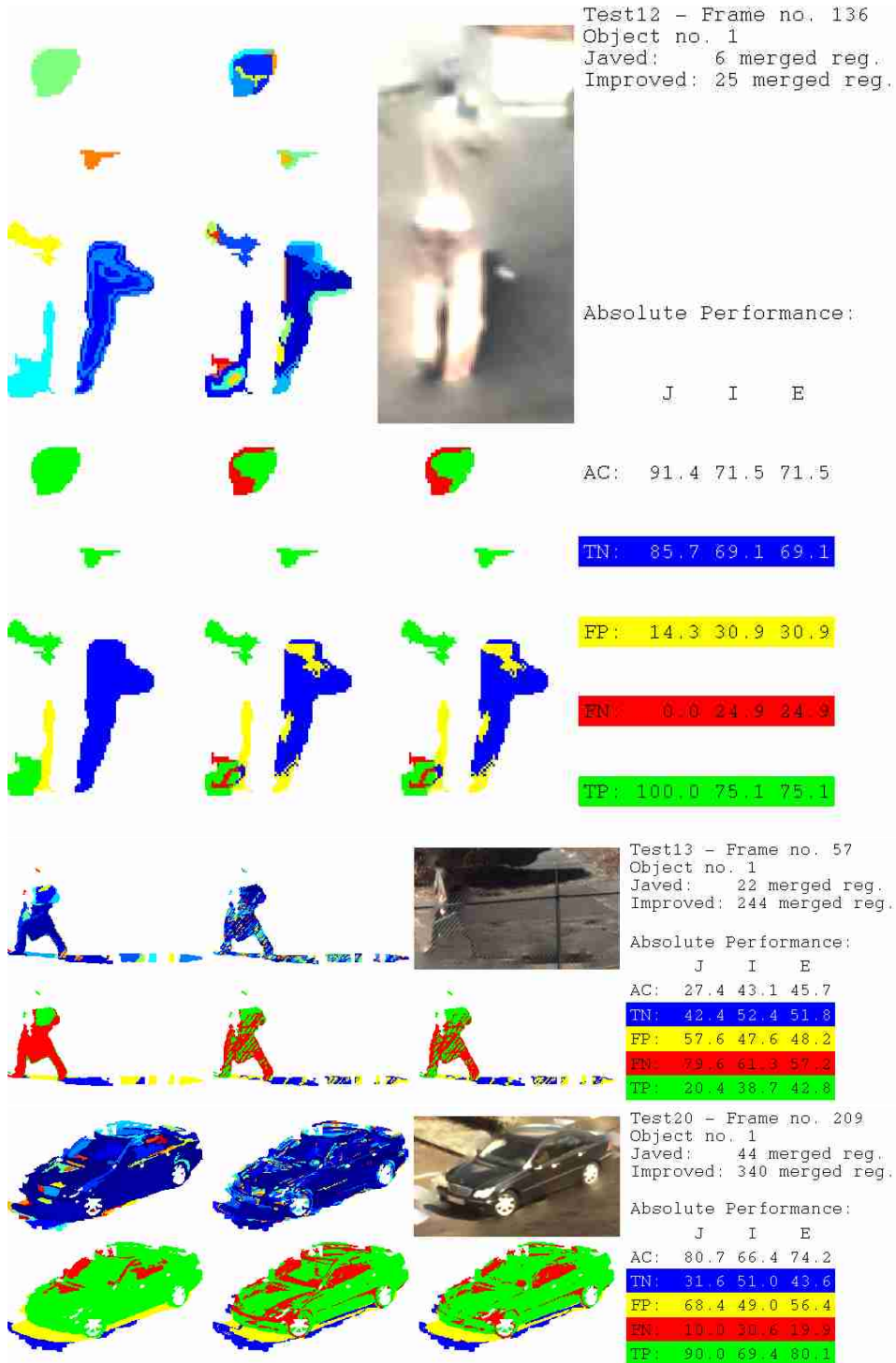
Abs. performance	$AC_{J I E}[\%]$	$TP_{J I E}[\%]$	$FP_{J I E}[\%]$	$FN_{J I E}[\%]$	$TN_{J I E}[\%]$
	<i>Continued</i>	<i>from</i>	<i>previous</i>	<i>page</i>	
Test413-121-1	69.1 39.1 38.2	75.9 33.6 34.1	61.3 36.5 43.7	24.1 66.4 65.9	38.7 63.5 56.3
Test414-193-1	56.5 72.6 73.9	59.2 71.3 73.2	49.5 24.3 24.4	40.8 28.7 26.8	50.5 75.7 75.6
Test415-261-1	68.6 79.6 81.2	63.8 89.9 94.5	27.2 29.5 30.5	36.2 10.1 5.5	72.8 70.5 69.5
Test416-213-1	79.2 76.4 76.8	92.1 92.4 93.3	33.7 39.6 39.7	7.9 7.6 6.7	66.3 60.4 60.3
Test416-521-2	44.1 56.1 56.3	6.6 32.3 32.6	0.0 8.4 8.4	93.4 67.7 67.4	100.0 91.6 91.6
Test418-69-1	57.0 53.8 52.4	28.5 29.1 30.8	0.0 8.9 15.1	71.5 70.9 69.2	100.0 91.1 84.9
Test419-501-2	60.7 94.9 96.1	3.3 89.2 93.1	0.0 1.2 1.9	96.7 10.8 6.9	100.0 98.8 98.1
Test421-169-2	86.0 79.2 81.3	94.0 87.7 91.6	23.1 30.7 30.7	6.0 12.3 8.4	76.9 69.3 69.3
Test422-249-1	57.1 63.1 64.7	23.8 38.2 43.1	23.6 22.4 22.7	76.2 61.8 56.9	76.4 77.6 77.3
Test423-213-1	59.8 80.9 81.0	95.5 90.3 90.7	78.5 29.2 29.3	4.5 9.7 9.3	21.5 70.8 70.7
Test424-237-1	74.0 78.6 76.9	88.3 74.9 81.4	46.0 16.1 29.3	11.7 25.1 18.6	54.0 83.9 70.7
Test425-145-1	62.8 76.5 77.4	63.3 86.9 88.6	37.7 36.7 36.8	36.7 13.1 11.4	62.3 63.3 63.2
Test426-65-1	67.2 65.9 72.1	4.1 42.3 63.8	0.0 21.8 23.6	95.9 57.7 36.2	100.0 78.2 76.4
Test428-193-1	72.8 83.0 82.9	88.0 92.2 92.2	43.6 27.0 27.2	12.0 7.8 7.8	56.4 73.0 72.8
Test430-177-1	79.7 76.8 78.9	65.6 68.7 74.8	8.3 16.3 17.6	34.4 31.3 25.2	91.7 83.7 82.4
Test431-137-1	78.1 66.6 67.1	63.2 51.3 52.8	5.6 16.7 17.1	36.8 48.7 47.2	94.4 83.3 82.9
Test433-109-1	88.2 82.0 80.7	59.1 78.3 78.4	0.6 16.6 18.4	40.9 21.7 21.6	99.4 83.4 81.6
Test434-465-2	78.0 72.1 73.0	64.6 59.5 61.8	11.6 18.1 18.2	35.4 40.5 38.2	88.4 81.9 81.8
Test435-297-2	71.0 78.7 80.5	79.0 83.8 87.3	38.8 27.6 27.8	21.0 16.2 12.7	61.2 72.4 72.2
Mean	64.9 67.5 69.2	63.4 63.1 69.7	35.3 29.3 34.0	64.7 70.7 66.0	36.6 36.9 30.3
Standard deviation	17.8 13.8 13.7	30.0 18.9 18.3	33.4 23.3 23.9	33.4 23.3 23.9	30.0 18.9 18.3

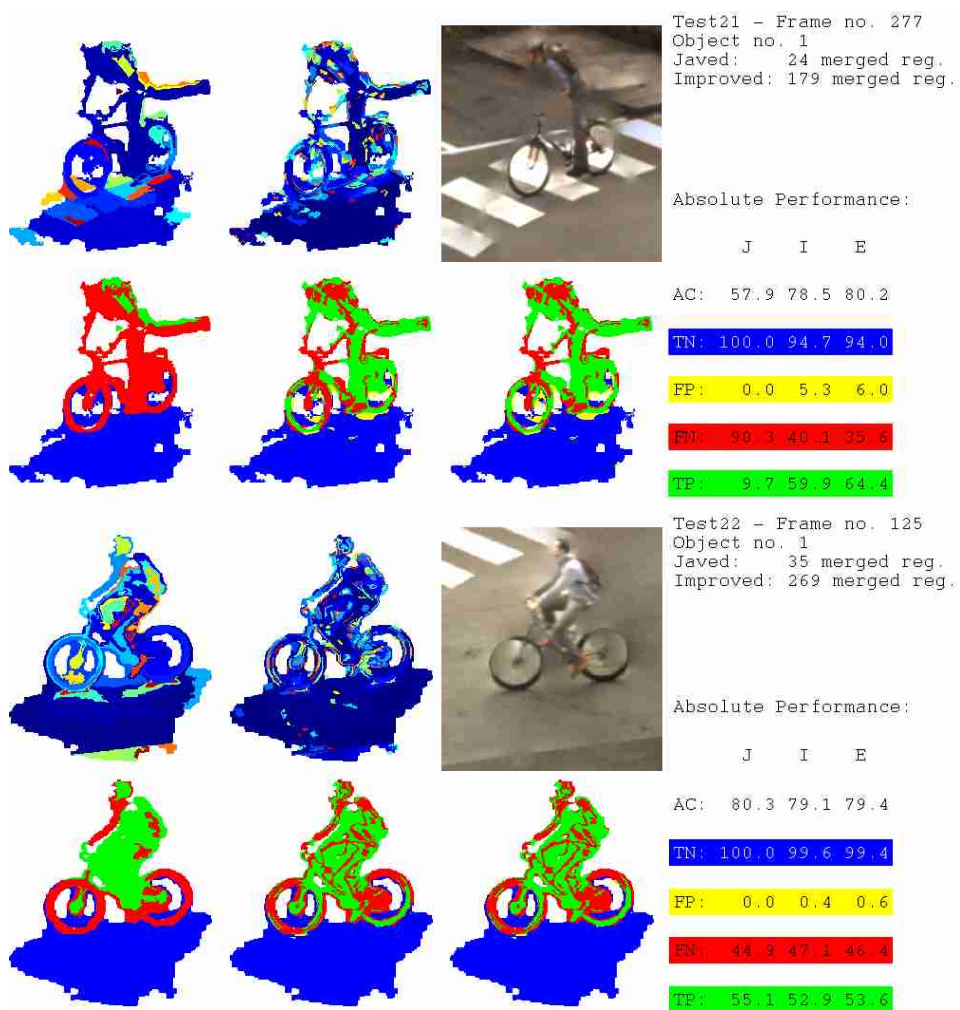
Table D.1: Absolute performance measures for the test set.

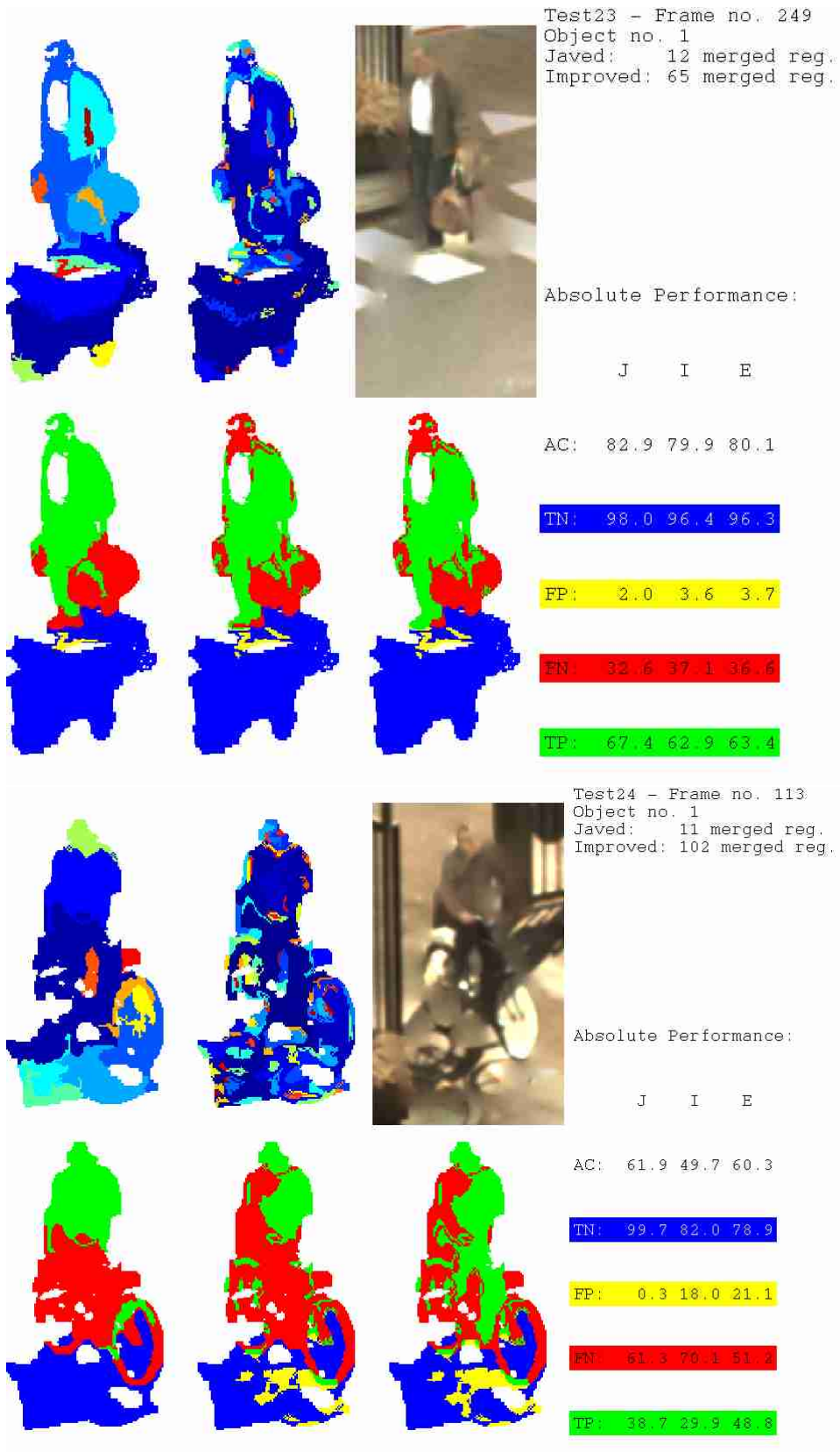
Relative performance	$AC_{R,J I E J E}[\%]$	$TP_{R,J I E J E}[\%]$	$TN_{R,J I E J E}[\%]$
Test12-136-1	-21.8 0.0 -21.8	-24.9 0.0 -24.9	-19.4 0.0 -19.4
Test13-57-1	57.3 6.0 66.8	89.7 10.6 109.8	23.6 -1.1 22.2
Test20-209-1	-17.7 11.7 -8.1	-22.9 15.4 -11.0	61.4 -14.5 38.0
Test21-277-1	35.6 2.2 38.5	517.5 7.5 563.9	-5.3 -0.7 -6.0
Test22-125-1	-1.5 0.4 -1.1	-4.0 1.3 -2.7	-0.4 -0.2 -0.6
Test23-249-1	-3.6 0.3 -3.4	-6.7 0.8 -5.9	-1.6 -0.1 -1.7
Test24-113-1	-19.7 21.3 -2.6	-22.7 63.2 26.1	-17.8 -3.8 -20.9
Test24-581-2	-10.0 8.3 -2.5	-17.3 29.0 6.7	4.6 -23.9 -20.4
Test24-701-3	11.8 -5.4 5.7	-12.2 17.9 3.4	101.0 -43.2 14.2
Test24-745-4	-1.0 -7.7 -8.6	-2.7 47.3 43.3	0.9 -64.2 -63.9
Test25-101-1	-7.0 7.2 -0.3	-19.7 28.3 3.1	3.7 -6.6 -3.1
Test25-285-2	-6.4 4.4 -2.3	-12.2 8.9 -4.4	0.1 0.0 0.1
Test27-737-3	-19.7 15.9 -6.9	-42.2 50.9 -12.8	-2.4 0.0 -2.4
Test29-105-1	-22.8 2.1 -21.2	-51.9 18.2 -43.2	-13.6 -0.9 -14.3
Test29-141-2	10.6 -3.2 7.1	-15.4 17.9 -0.2	152.1 -41.6 47.2
Test30-61-1	-11.0 -0.3 -11.3	-18.5 0.0 -18.5	-2.3 -0.8 -3.1
Test31-45-1	0.0 2.4 2.4	39.1 12.9 57.0	-27.8 -11.8 -36.3
Test31-77-2	-7.6 11.4 3.0	-18.5 16.6 -4.9	52.8 -4.6 45.8
Test32-241-1	450.6 -34.0 263.5	-50.9 1.2 -50.3	Inf -37.4 Inf
Test32-697-2	-27.0 -8.3 -33.0	-48.3 71.6 -11.3	-4.5 -54.0 -56.1
Test37-149-1	-8.0 2.5 -5.7	-13.4 5.0 -9.1	-1.5 0.0 -1.5
Test45-293-2	-4.5 13.9 8.8	-13.2 20.3 4.5	22.4 0.0 22.4
Test47-173-1	14.0 0.4 14.4	52.5 1.1 54.2	3.4 0.0 3.4
Test48-185-1	-6.3 3.0 -3.4	-18.5 8.5 -11.5	2.8 -0.1 2.7
Test49-161-1	-3.3 1.2 -2.1	-7.4 3.1 -4.6	1.7 -0.8 0.9
<i>Continued</i>	<i>on</i>	<i>next</i>	<i>page</i>

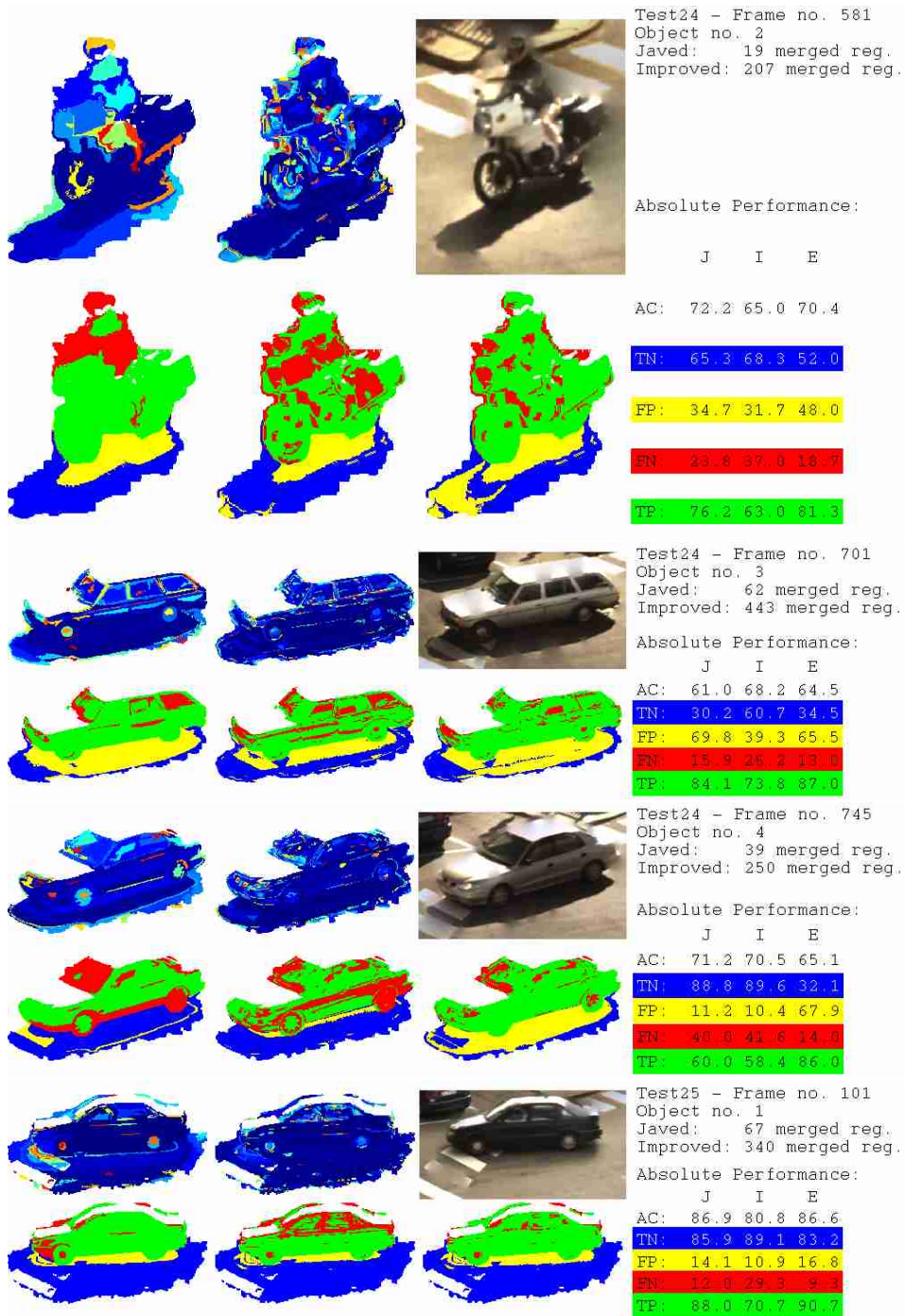
Relative performance	$AC_{R,JI IE JE}[\%]$	$TP_{R,JI IE JE}[\%]$	$TN_{R,JI IE JE}[\%]$
<i>Continued</i>	<i>from</i>	<i>previous</i>	<i>page</i>
Test52-61-1	-0.8 -0.2 -1.0	142.5 0.0 142.5	-7.2 -0.3 -7.5
Test74-81-1	-2.0 1.7 -0.3	-5.6 2.7 -3.1	5.0 0.0 5.0
Test75-181-2	11.2 0.6 11.9	46.8 0.6 47.6	-66.0 0.0 -66.0
Test76-93-1	-6.1 9.1 2.4	-8.4 11.0 1.7	14.0 -3.7 9.8
Test77-105-1	16.6 58.4 84.7	42.7 100.6 186.3	-6.7 -0.5 -7.2
<i>Test78-141-1</i>	96.6 0.0 96.6	-46.8 0.0 -46.8	<i>Inf</i> 0.0 <i>Inf</i>
Test79-237-1	23.6 -0.8 22.6	23.3 0.0 23.3	23.7 -2.1 21.2
Test80-69-1	152.5 7.0 170.3	432.8 8.5 477.9	-32.3 0.0 -32.3
Test81-277-1	-9.9 1.6 -8.4	-14.7 3.1 -12.1	-3.0 0.0 -3.0
Test81-393-2	-9.6 0.0 -9.6	-15.1 0.0 -15.1	8.5 0.0 8.5
<i>Test82-337-1</i>	-16.6 4.1 -13.2	-35.1 5.4 -31.6	<i>Inf</i> 0.0 <i>Inf</i>
Test82-361-2	18.3 13.4 34.2	22.3 15.3 41.0	-3.8 -0.2 -4.0
<i>Test83-297-1</i>	-31.1 0.0 -31.1	-31.4 0.0 -31.4	<i>Inf</i> 0.0 <i>Inf</i>
<i>Test83-321-2</i>	-23.6 0.0 -23.6	-23.9 0.2 -23.8	<i>Inf</i> 0.0 <i>Inf</i>
Test300-233-3	-5.6 -1.9 -7.3	280.7 0.0 280.7	-18.9 -2.3 -20.8
Test300-429-2	6.3 63.3 73.6	-2.0 117.3 112.9	18.8 -1.5 17.1
Test300-561-4	3.6 -14.2 -11.0	-25.1 8.1 -19.0	156.6 -48.6 32.0
Test302-165-1	-5.2 1.4 -3.9	-5.3 1.7 -3.7	-3.9 -3.0 -6.8
<i>Test303-61-1</i>	-2.5 0.3 -2.2	-8.6 0.4 -8.3	<i>Inf</i> 0.0 <i>Inf</i>
Test400-185-2	8.5 2.5 11.1	28.2 4.5 34.0	-10.4 -0.3 -10.7
Test400-185-3	8.3 1.8 10.2	-9.0 2.4 -6.8	99.6 0.0 99.6
Test403-469-1	38.1 2.6 41.8	-39.0 90.5 16.3	43.9 0.0 43.9
Test406-113-1	26.0 0.0 26.0	-2.4 0.2 -2.2	844.4 -0.5 840.0
Test407-309-1	10.6 3.8 14.8	-12.5 5.9 -7.3	140.4 -0.3 139.7
Test408-233-1	12.3 3.7 16.4	-10.8 5.3 -6.0	190.8 -0.2 190.3
Test410-157-1	8.1 0.3 8.4	730.5 1.4 742.4	-23.7 -0.1 -23.8
Test412-69-1	72.2 -7.2 59.9	282.5 0.0 282.5	30.3 -11.2 15.7
Test412-221-2	62.0 2.3 65.7	213.6 3.6 225.0	-16.0 0.0 -16.0
Test413-121-1	-43.4 -2.3 -44.7	-55.7 1.5 -55.1	64.1 -11.3 45.5
Test414-193-1	28.5 1.8 30.8	20.4 2.7 23.6	49.9 -0.1 49.7
Test415-261-1	16.0 2.0 18.4	40.9 5.1 48.1	-3.2 -1.4 -4.5
Test416-213-1	-3.5 0.5 -3.0	0.3 1.0 1.3	-8.9 -0.2 -9.0
Test416-521-2	27.2 0.4 27.7	389.4 0.9 393.9	-8.4 0.0 -8.4
Test418-69-1	-5.6 -2.6 -8.1	2.1 5.8 8.1	-8.9 -6.8 -15.1
Test419-501-2	56.3 1.3 58.3	2603.0 4.4 2721.2	-1.2 -0.7 -1.9
Test421-169-2	-7.9 2.7 -5.5	-6.7 4.4 -2.6	-9.9 0.0 -9.9
Test422-249-1	10.5 2.5 13.3	60.5 12.8 81.1	1.6 -0.4 1.2
Test423-213-1	35.3 0.1 35.5	-5.4 0.4 -5.0	229.3 -0.1 228.8
Test424-237-1	6.2 -2.2 3.9	-15.2 8.7 -7.8	55.4 -15.7 30.9
Test425-145-1	21.8 1.2 23.2	37.3 2.0 40.0	1.6 -0.2 1.4
Test426-65-1	-1.9 9.4 7.3	931.7 50.8 1456.1	-21.8 -2.3 -23.6
Test428-193-1	14.0 -0.1 13.9	4.8 0.0 4.8	29.4 -0.3 29.1
Test430-177-1	-3.6 2.7 -1.0	4.7 8.9 14.0	-8.7 -1.6 -10.1
Test431-137-1	-14.7 0.8 -14.1	-18.8 2.9 -16.5	-11.8 -0.5 -12.2
Test433-109-1	-7.0 -1.6 -8.5	32.5 0.1 32.7	-16.1 -2.2 -17.9
Test434-465-2	-7.6 1.2 -6.4	-7.9 3.9 -4.3	-7.4 -0.1 -7.5
Test435-297-2	10.8 2.3 13.4	6.1 4.2 10.5	18.3 -0.3 18.0
Mean	13.5 3.20 14.9	86.5 13.4 108	31.2 -5.90 22.1
Standard deviation	60.0 12.0 44.2	34.7 24.0 380	115 13.7 113

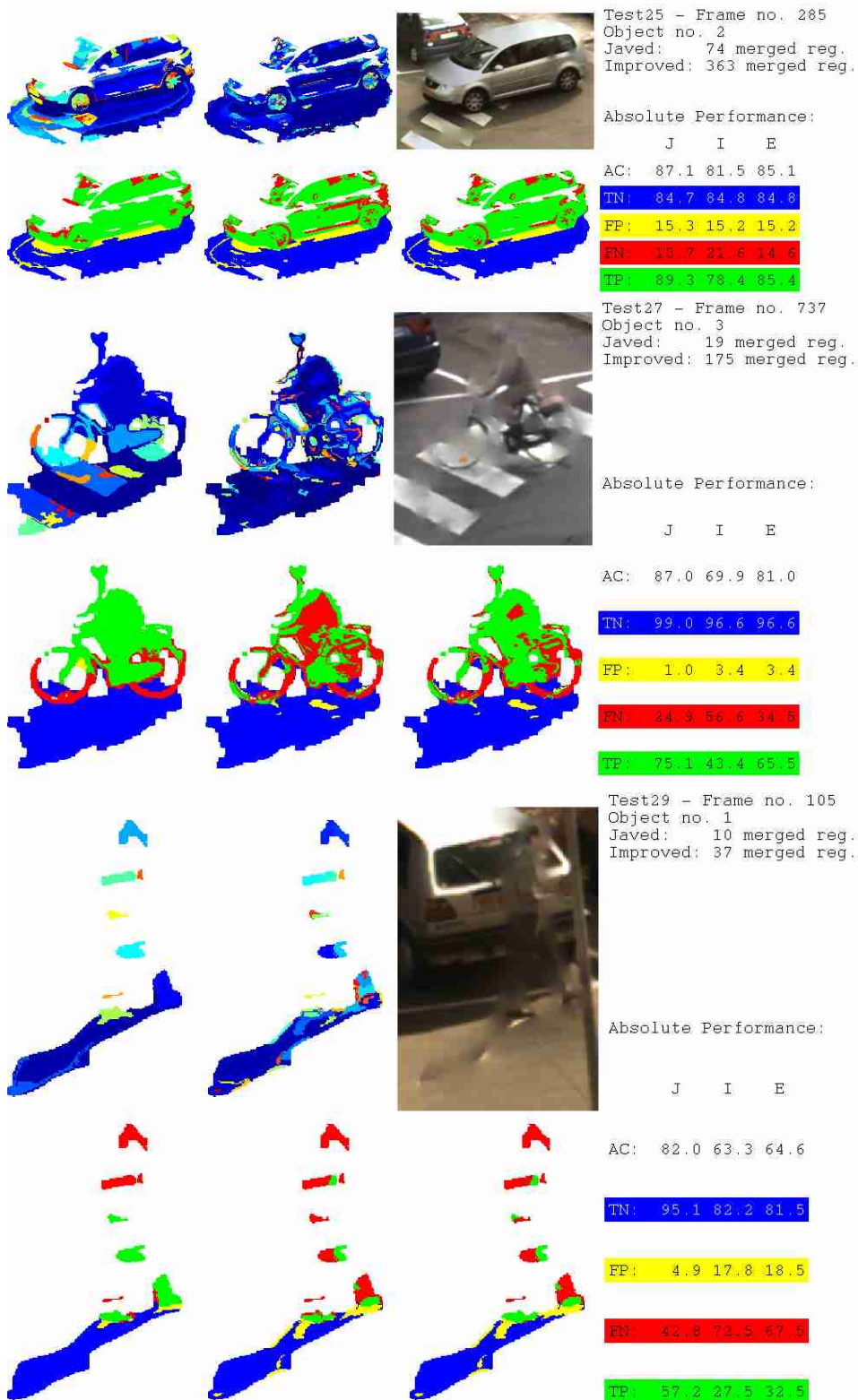
Table D.2: Relative performance measures for the test set. Example: $AC_{R,JE}$ denote the relative improvement, in %, in accuracy, when using method E instead of method J . For the relative TN , only 66 examples are used for computing mean and std. and in the statistical tests. Those names in italic are not used, since they obtain zero value for the absolute TN of method J , and therefore are not defined (*inf*) for the relative measure.

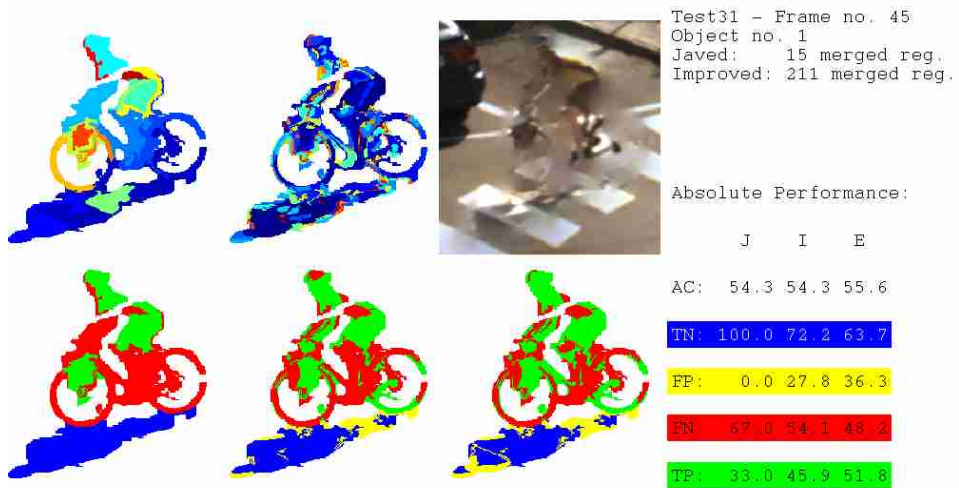
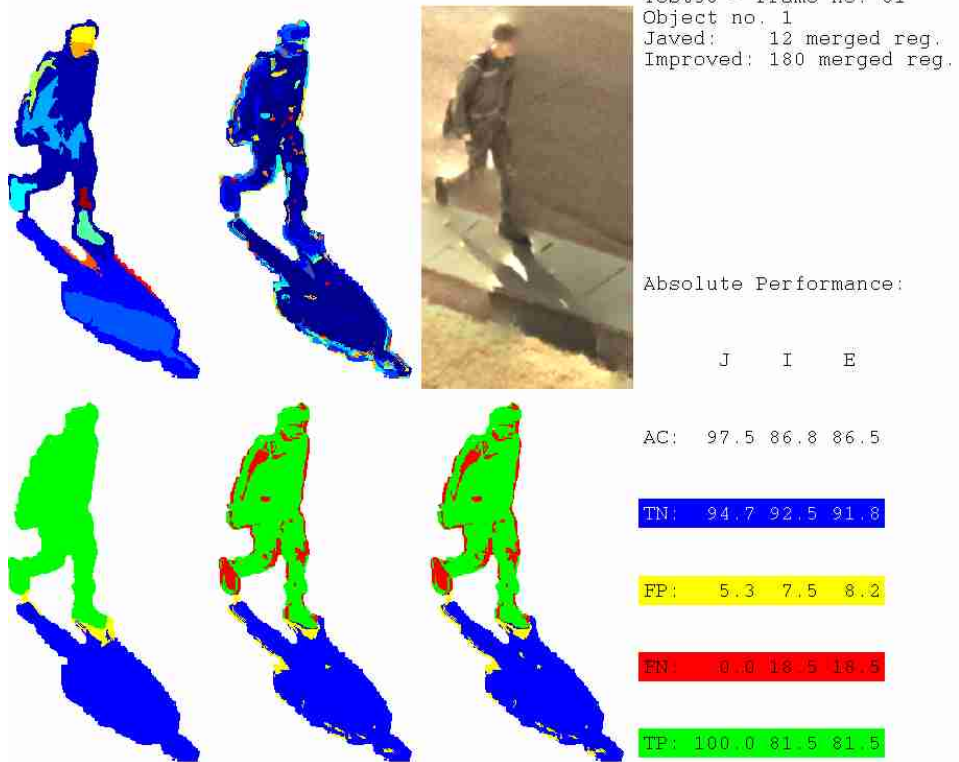
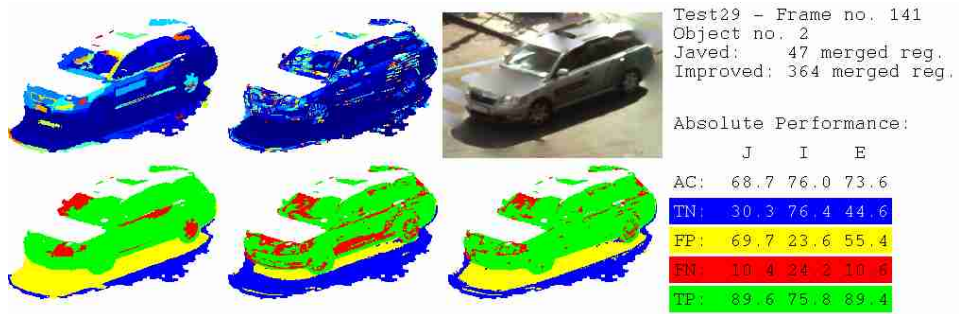


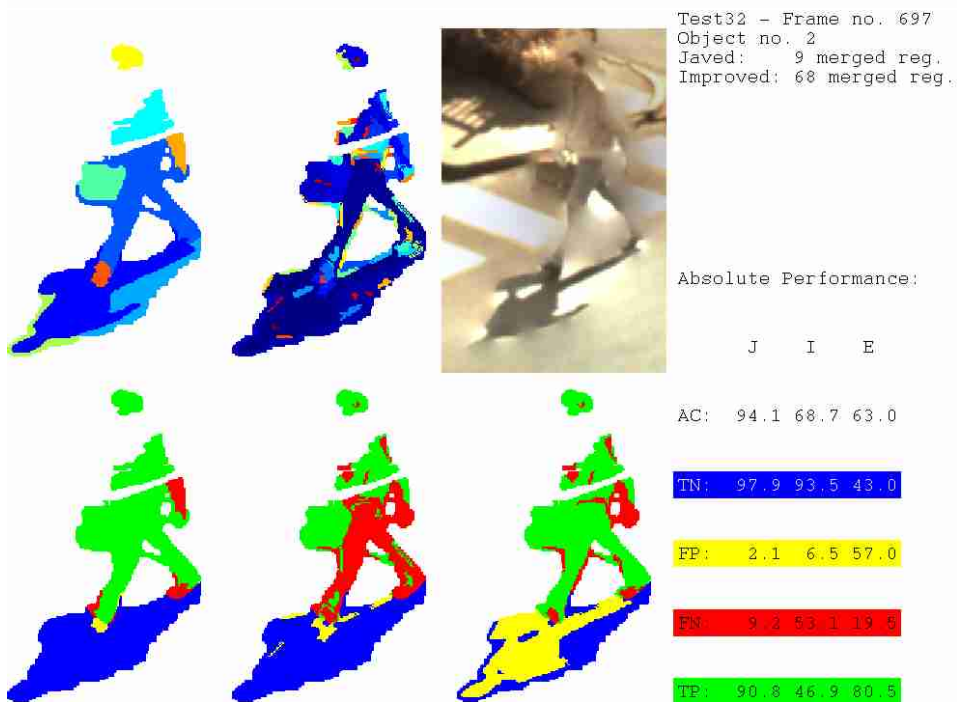
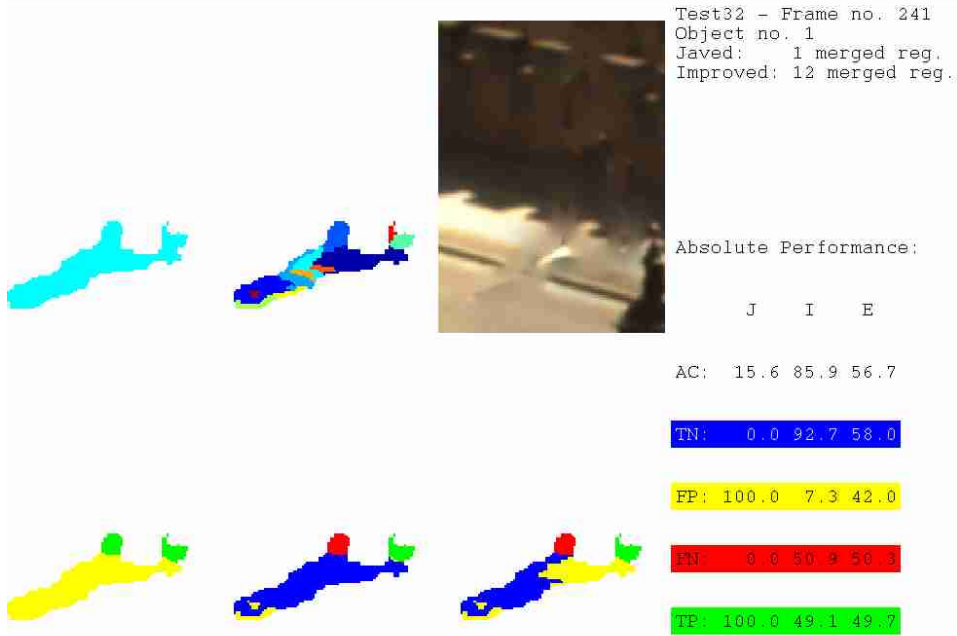
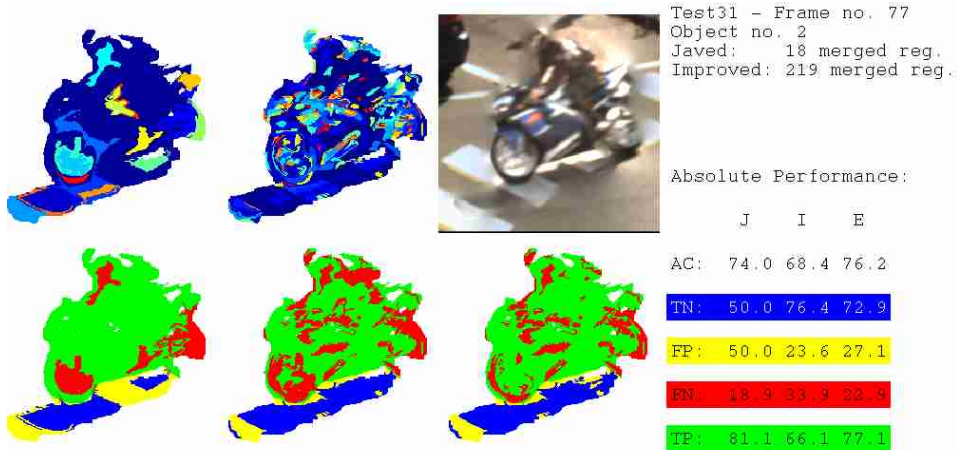


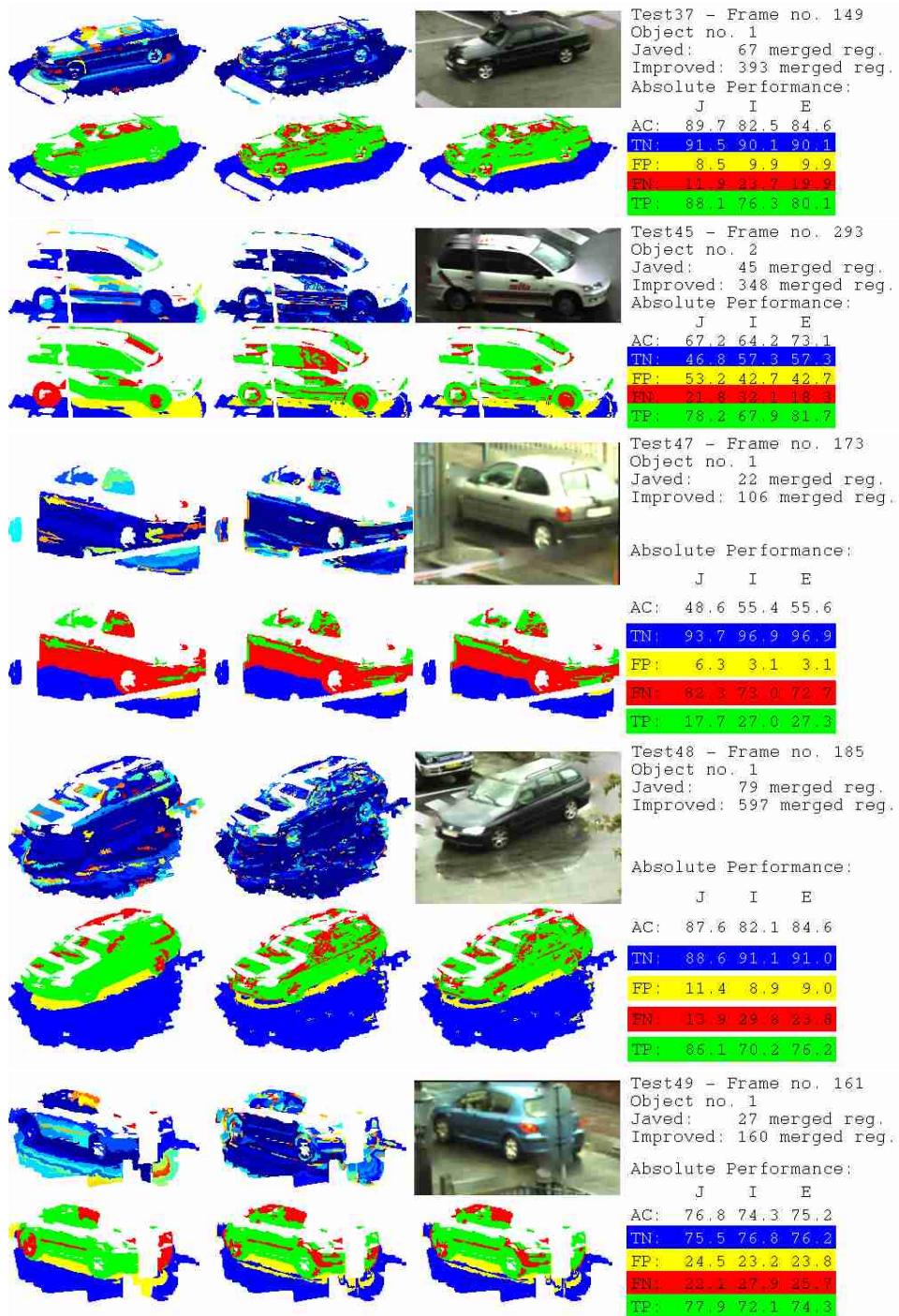














Test52 - Frame no. 61
 Object no. 1
 Javed: 26 merged reg.
 Improved: 98 merged reg.

Absolute Performance:

	J	I	E
--	---	---	---

AC: 61.6 61.1 61.0

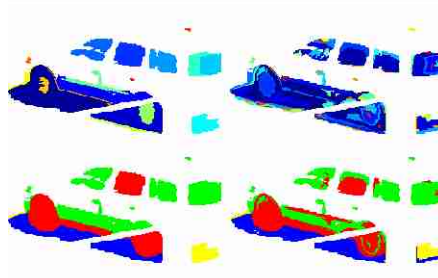

TN: 66.7 61.9 61.7

FP: 33.3 38.1 38.3

FN: 77.2 44.7 44.7

TP: 22.8 55.3 55.3

Test74 - Frame no. 81
 Object no. 1
 Javed: 18 merged reg.
 Improved: 87 merged reg.

Absolute Performance:

	J	I	E
--	---	---	---

AC: 61.2 60.0 61.0


TN: 78.0 81.9 81.9

FP: 22.0 18.1 18.1

FN: 45.1 48.2 46.8

TP: 54.9 51.8 53.2

Test75 - Frame no. 181
 Object no. 2
 Javed: 11 merged reg.
 Improved: 77 merged reg.

Absolute Performance:

	J	I	E
--	---	---	---


AC: 55.6 61.8 62.2

TN: 99.7 33.9 33.9

FP: 0.3 66.1 66.1

FN: 53.8 32.2 31.8


TP: 46.2 67.8 68.2



Test76 - Frame no. 93
 Object no. 1
 Javed: 21 merged reg.
 Improved: 126 merged reg.

Absolute Performance:

	J	I	E
AC:	58.8	55.2	60.2
TN:	21.5	24.5	23.6
FP:	78.5	75.5	76.4
FN:	27.4	33.5	26.2
TP:	72.6	66.5	73.8



Test77 - Frame no. 105
 Object no. 1
 Javed: 15 merged reg.
 Improved: 149 merged reg.

Absolute Performance:

	J	I	E
AC:	37.3	43.5	68.9
TN:	89.2	83.2	82.8
FP:	10.8	16.8	17.2
FN:	77.3	69.6	35.0
TP:	22.7	32.4	65.0



Test78 - Frame no. 141
 Object no. 1
 Javed: 2 merged reg.
 Improved: 18 merged reg.

Absolute Performance:

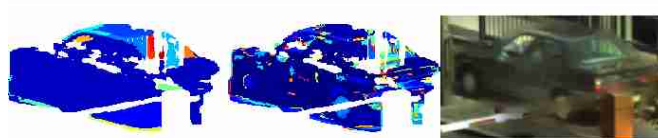
	J	I	E
AC:	38.3	75.3	75.3
TN:	0.0	89.0	89.0
FP:	100.0	11.0	11.0
FN:	0.0	46.9	46.8
TP:	100.0	53.2	53.2



Test79 - Frame no. 237
 Object no. 1
 Javed: 26 merged reg.
 Improved: 146 merged reg.

Absolute Performance:

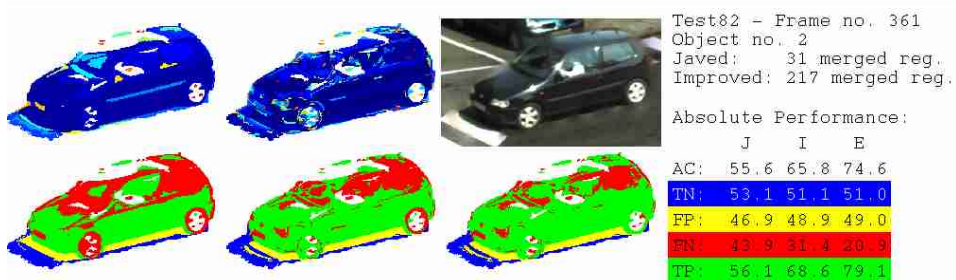
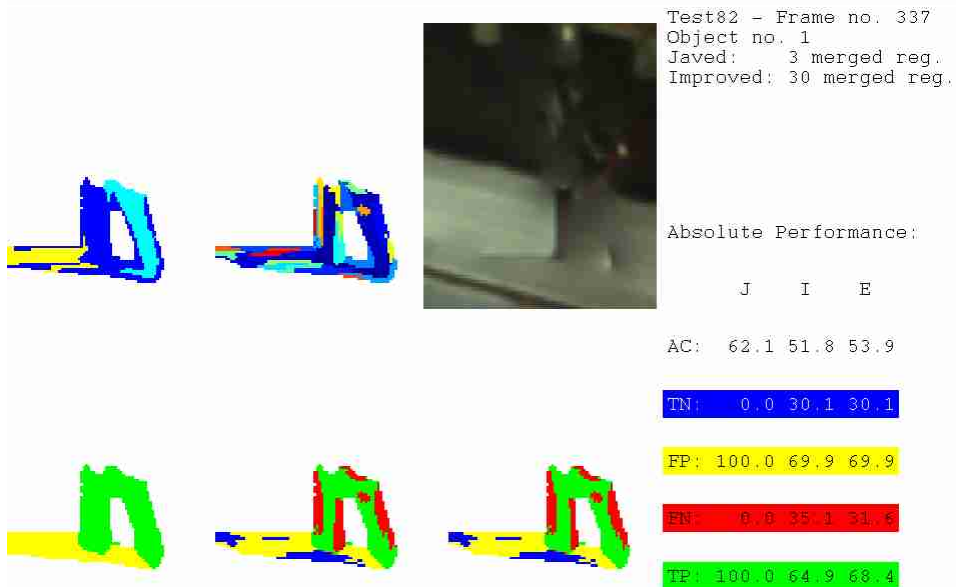
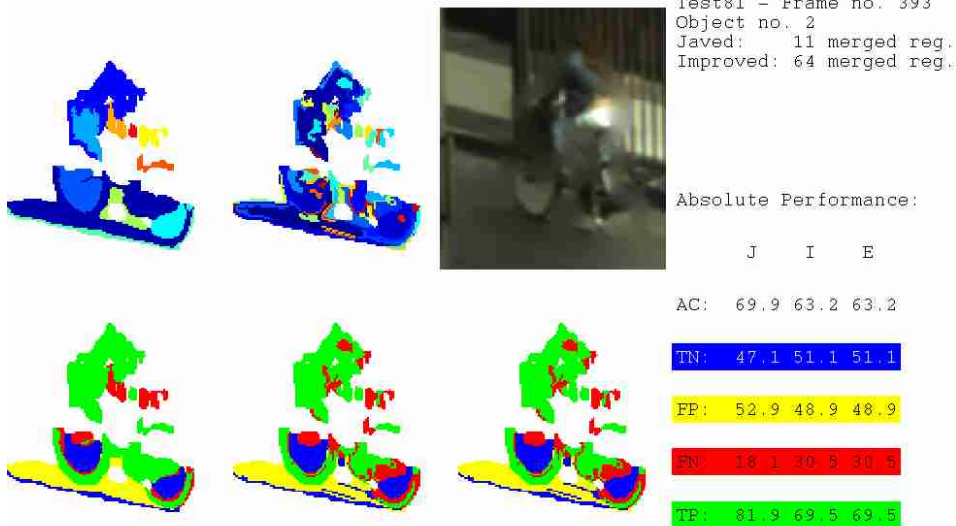
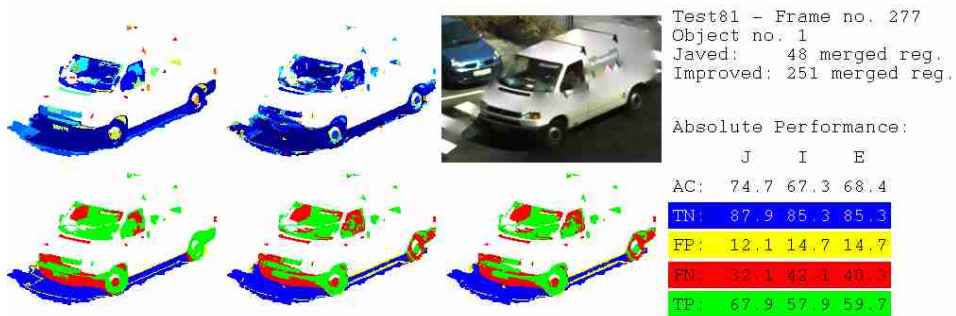
	J	I	E
AC:	41.1	50.8	50.4
TN:	58.6	72.5	71.0
FP:	41.4	27.5	29.0
FN:	65.3	57.2	57.2
TP:	34.7	42.8	42.8

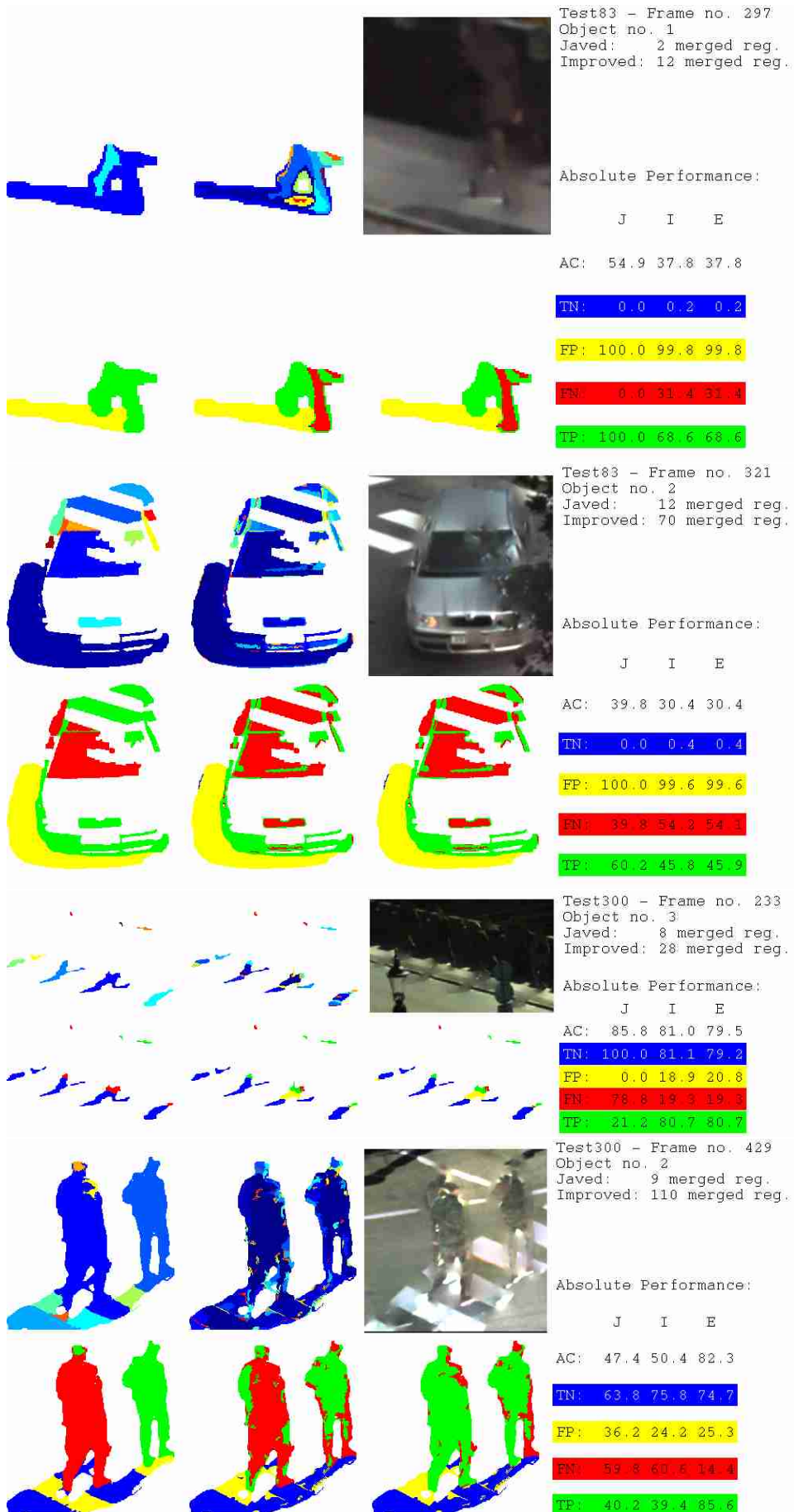


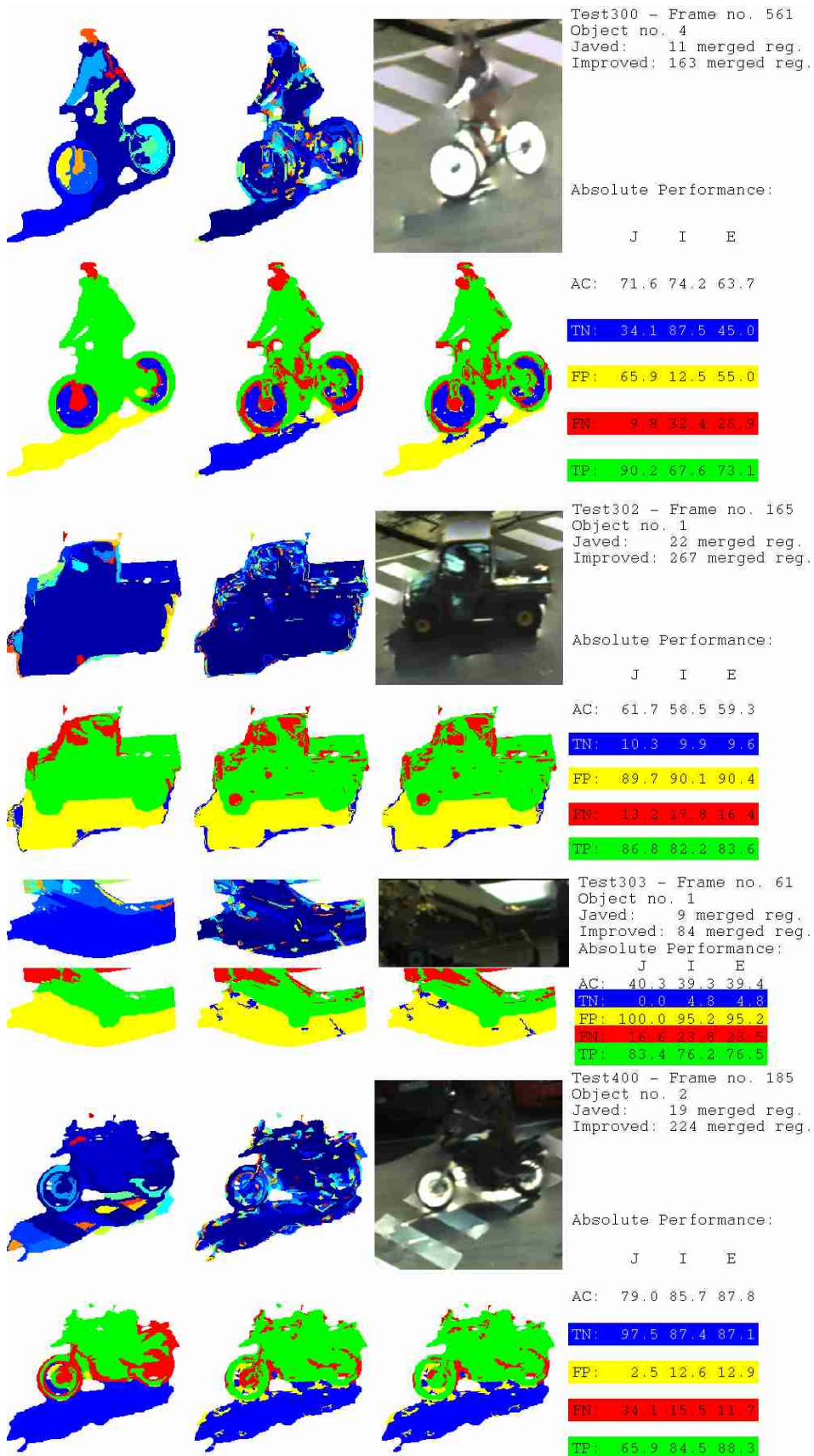
Test80 - Frame no. 69
 Object no. 1
 Javed: 15 merged reg.
 Improved: 156 merged reg.

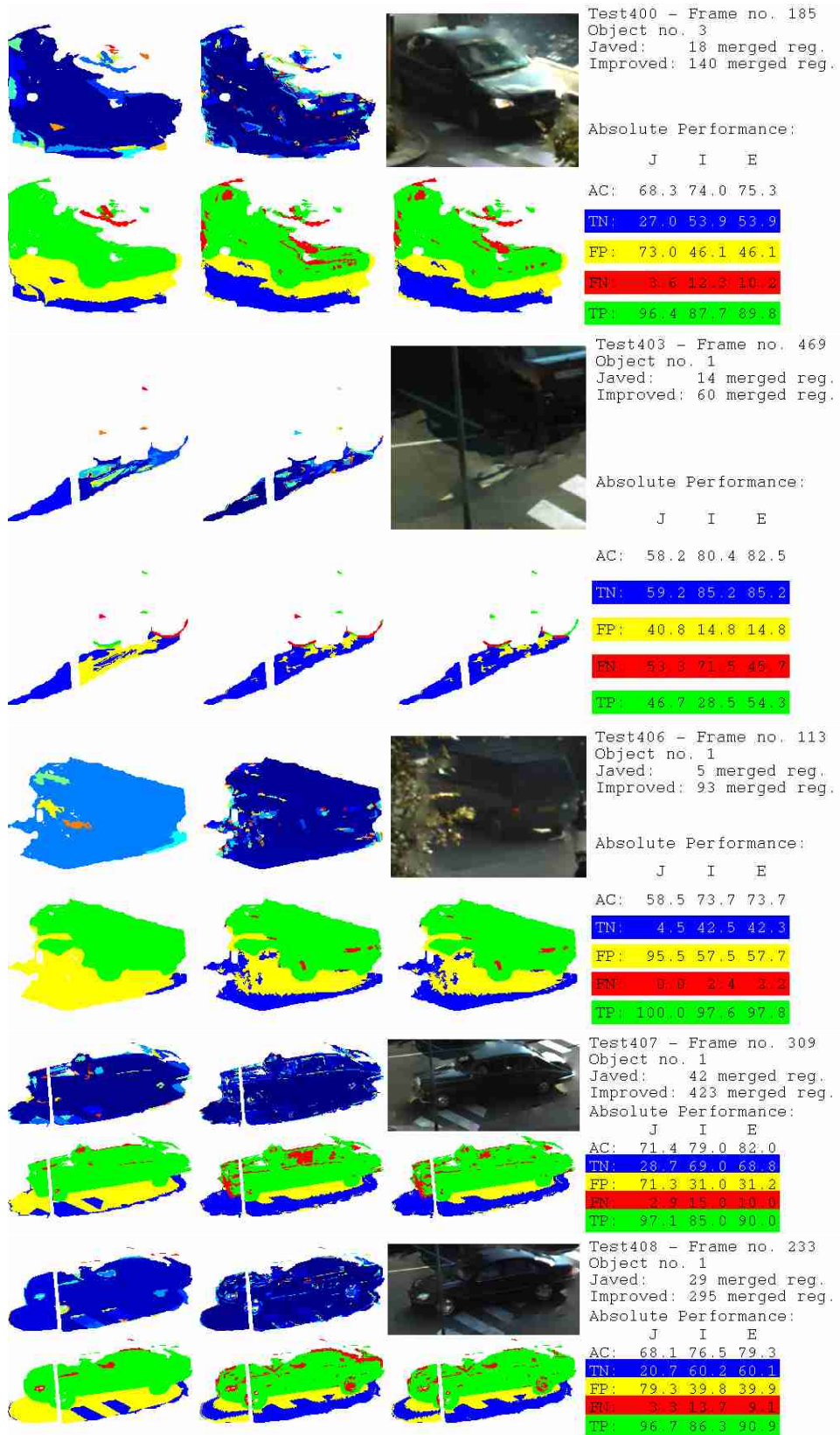
Absolute Performance:

	J	I	E
AC:	25.9	65.4	70.0
TN:	100.0	67.7	67.7
FP:	0.0	32.3	32.3
FN:	87.3	35.0	29.5
TP:	12.2	65.0	70.5







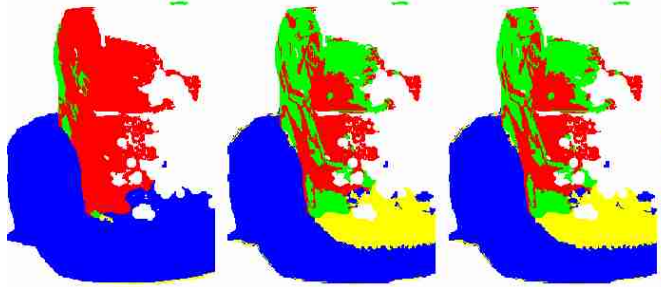





Test410 - Frame no. 157
 Object no. 1
 Javed: 15 merged reg.
 Improved: 181 merged reg.

Absolute Performance:

	J	I	E
--	---	---	---




AC:	59.5	64.3	64.5
TN:	98.8	75.4	75.3
FP:	1.2	24.6	24.7
FN:	94.1	51.0	50.3
TP:	5.9	49.0	49.7



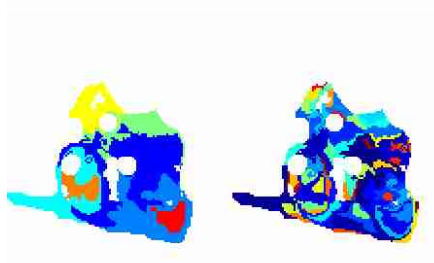
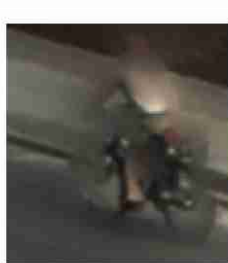

Test412 - Frame no. 69
 Object no. 1
 Javed: 3 merged reg.
 Improved: 20 merged reg.

Absolute Performance:

	J	I	E
--	---	---	---




AC:	22.7	39.1	36.3
TN:	26.1	34.0	30.2
FP:	73.9	66.0	69.8
FN:	86.3	47.8	47.6
TP:	13.7	52.4	52.4

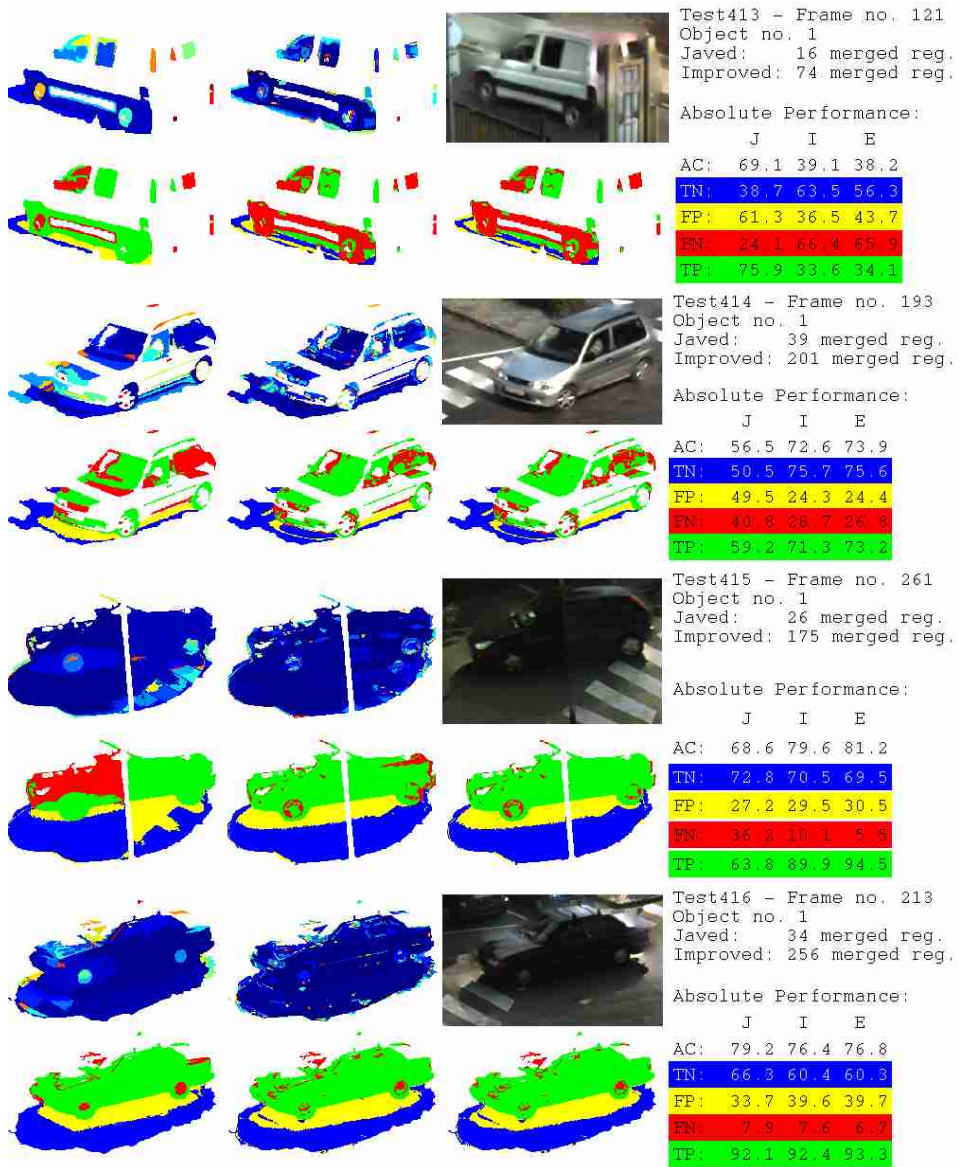
Test412 - Frame no. 221
 Object no. 2
 Javed: 7 merged reg.
 Improved: 60 merged reg.

Absolute Performance:

	J	I	E
--	---	---	---



AC:	32.4	52.5	53.7
TN:	100.0	84.0	84.0
FP:	0.0	16.0	16.0
FN:	86.0	56.1	54.5
TP:	14.0	43.9	45.5

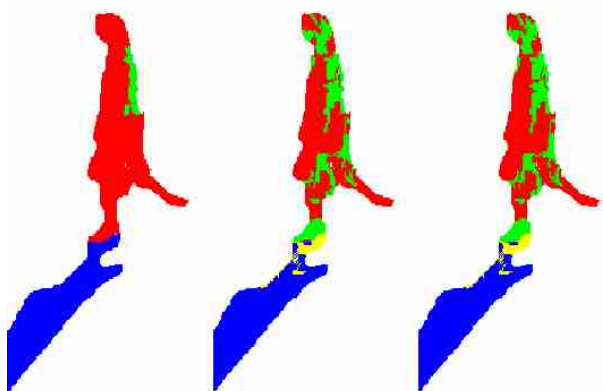




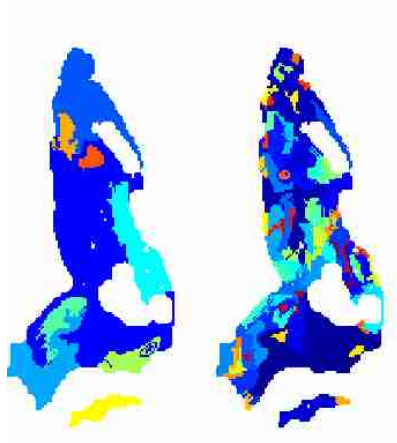


Test416 - Frame no. 521
 Object no. 2
 Javed: 8 merged reg.
 Improved: 78 merged reg.

Absolute Performance:

	J	I	E
--	---	---	---



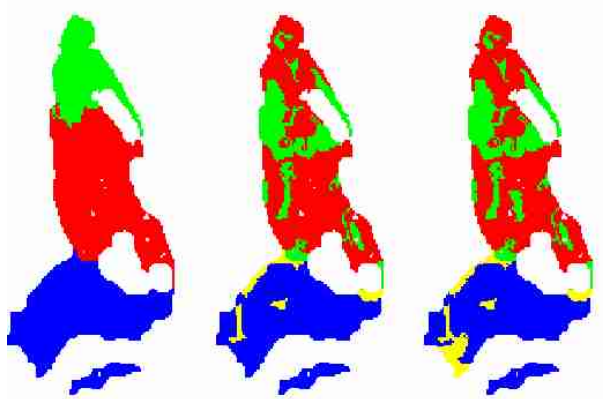
AC:	44.1	56.1	56.3
TN:	100.0	91.6	91.6
FP:	0.0	8.4	8.4
FN:	93.4	67.7	67.4
TP:	6.6	32.3	32.6

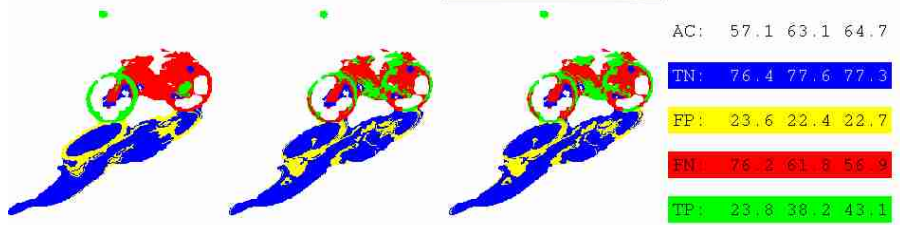
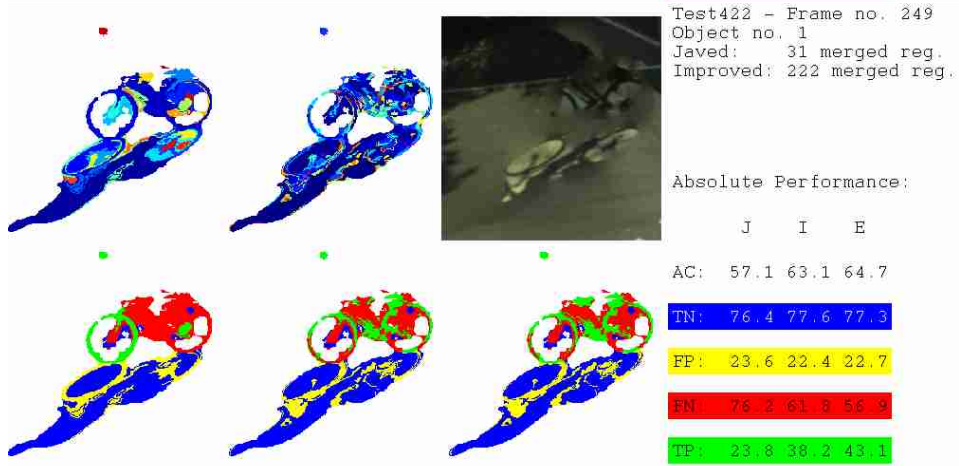
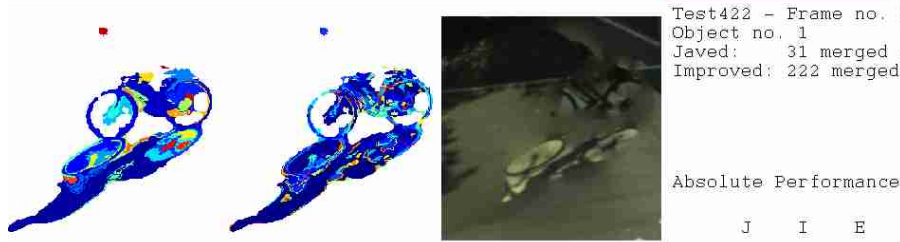
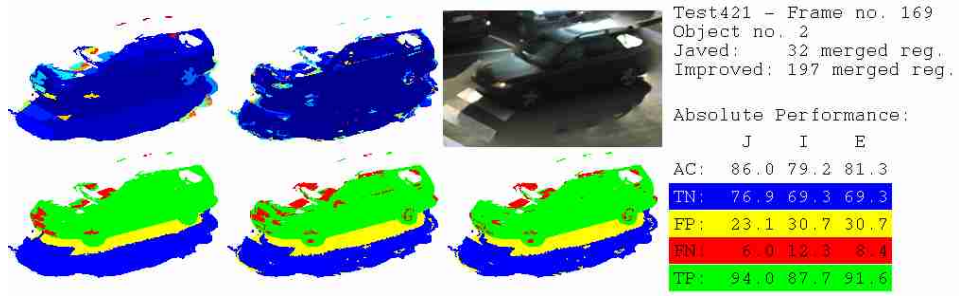
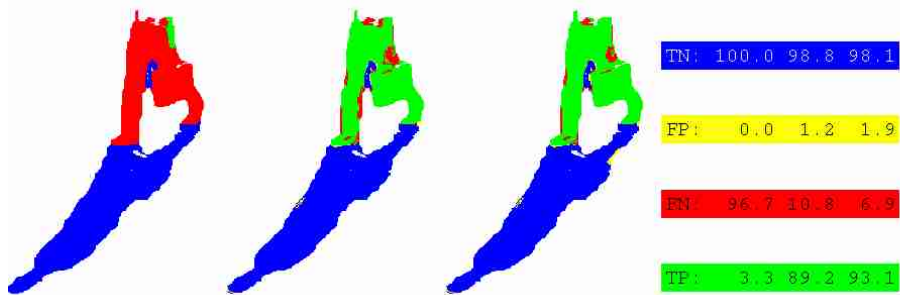
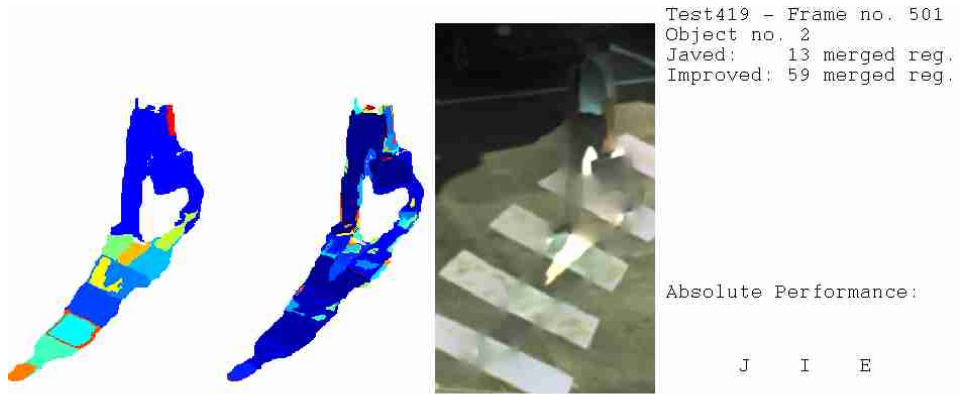
Test418 - Frame no. 69
 Object no. 1
 Javed: 9 merged reg.
 Improved: 81 merged reg.

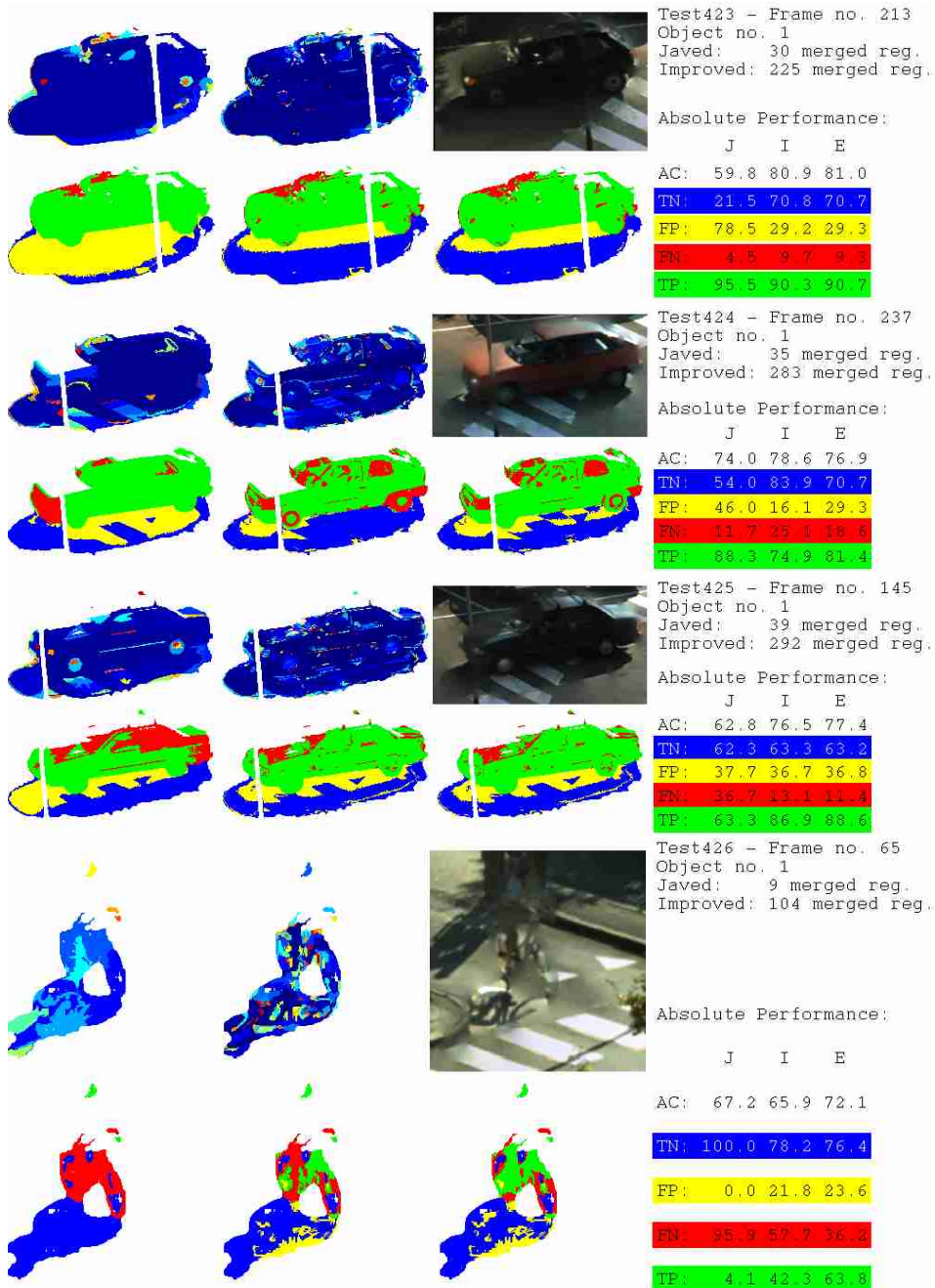
Absolute Performance:

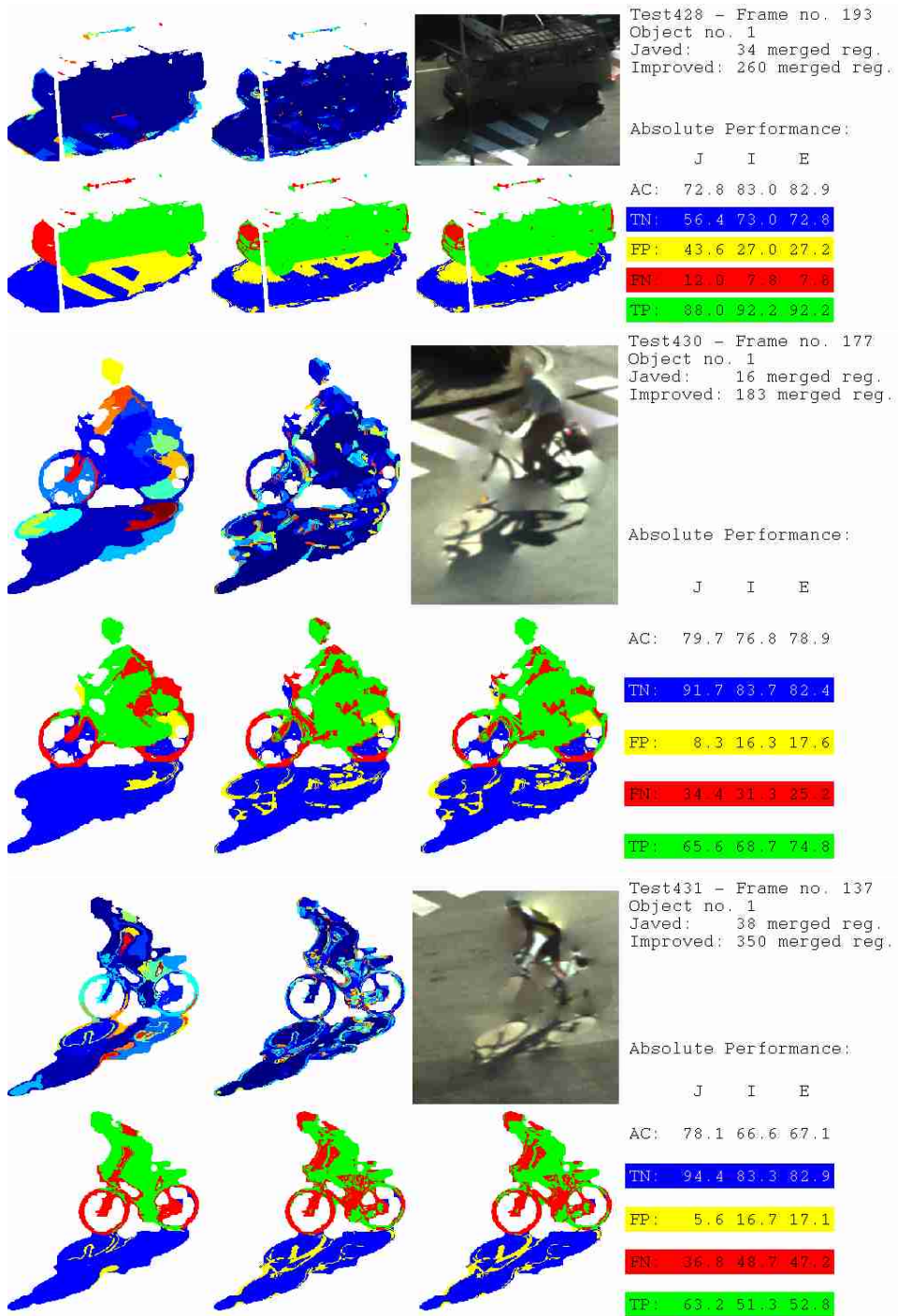
	J	I	E
--	---	---	---

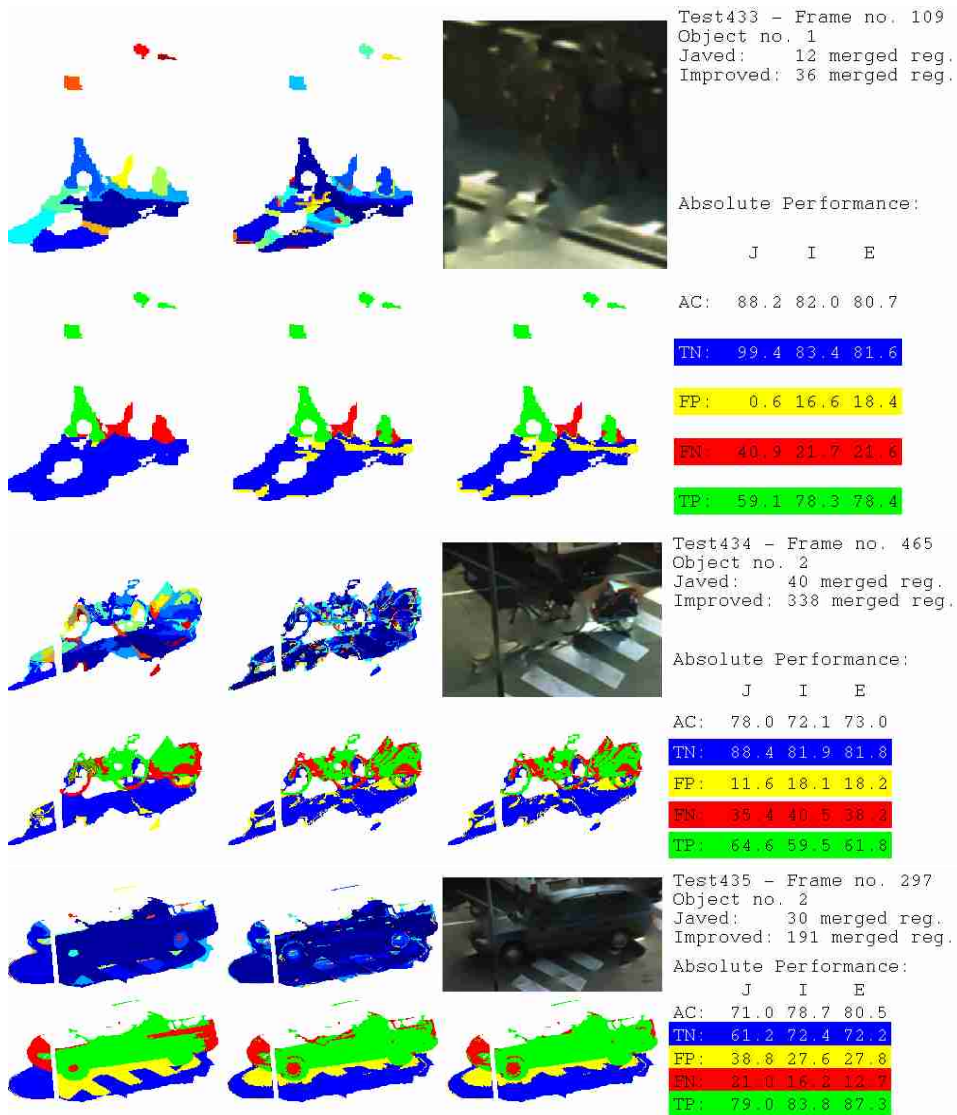


AC:	57.0	53.8	52.4
TN:	100.0	91.1	84.9
FP:	0.0	8.9	15.1
FN:	71.5	76.9	69.2
TP:	28.5	29.1	30.8









Appendix E

Matlab Routines

This appendix contains the Matlab routines used. A description of each file is given in their respective headers. Figure E.1 is a flowchart illustrating how the central Matlab routines interact. Numbers correspond to section titles.

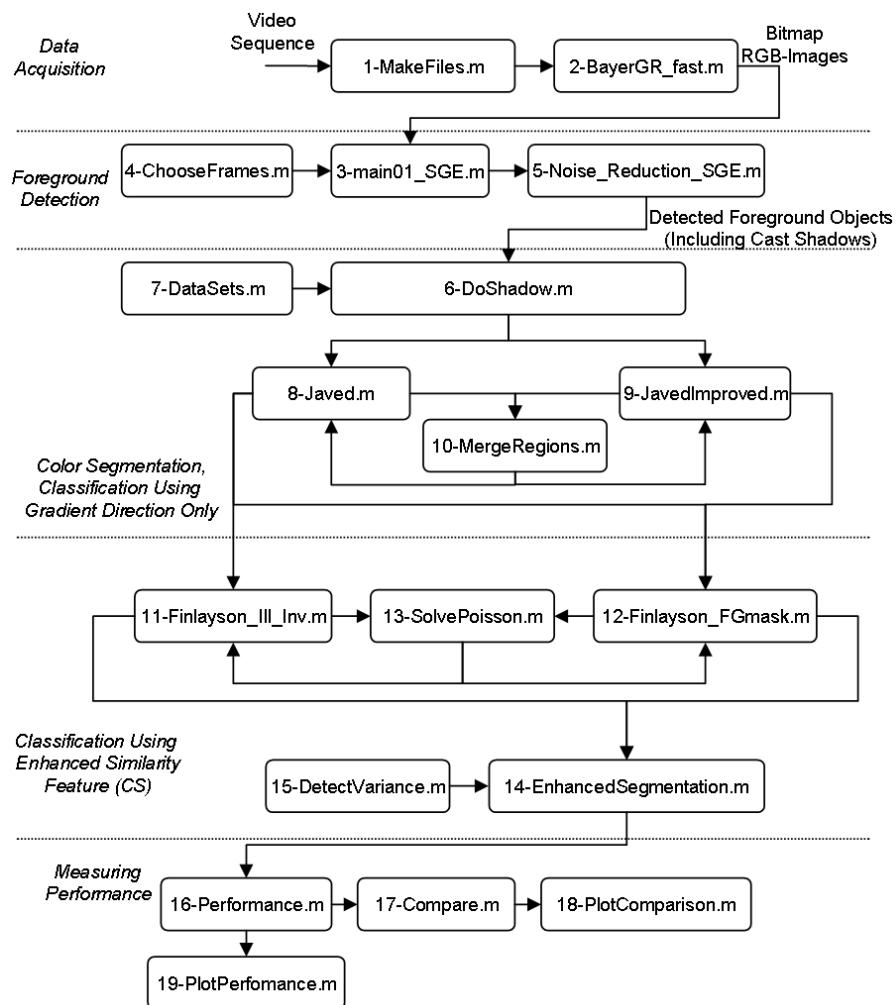


Figure E.1: Flowchart of central Matlab routines.

E.1 MakeFiles.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "MakeFiles.m"
%
% Description: Script that performs reconstruction of full color images
% from Bayer-filtered images on several video sequences at a time.
%
% Input: Video sequences in "avi"-format.
%
% Output: Bitmap Reconstructed RGB-image.
%
% Author: Søren Erbou
% Last Revision: August 6, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;
clear all;
clc;

CalibrationSequence=0;
NO=[424:435];
for j=1:length(NO)
    No=NO(j);
    if CalibrationSequence
        Dir='E:\SGE\Video\CalSeq\';
    else
        Dir='E:\SGE\Video\';
    end
    DestFolder=['Test',num2str(No)];
    if exist([Dir, DestFolder], 'dir')~=7
        OldDir=pwd;
        cd(Dir);
        mkdir(DestFolder);
        cd(OldDir);
    end
    FileInfo=aviinfo([Dir, 'Test', num2str(No), '.avi']);
    StartString=[Dir, 'Test', num2str(No), '_Test', num2str(No), '_'];
    EndString='.bmp';
    Frame=1;
    Frames=1:4:FileInfo.NumFrames;
    for i=1:size(Frames,2)
        TheMovie=aviread([Dir, 'Test', num2str(No), '.avi'], Frames(i));
        Test=TheMovie(1).cdata(:,:,:);
        Test=im2double(Test);
        Test2=BayerGR_fast(Test);
        MiddleString=int2str((i-1)*4+1);
        E=strcat(StartString, MiddleString, EndString);
        imwrite(Test2, E, 'bmp');
        disp(['Test ', num2str(No), ' frame ', num2str(i), '...']);
    end
end
disp('done...');

```

E.2 BayerGR_fast.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "BayerGR_fast.m"
%
% Description: A function that reconstructs a RGB-image from a Bayer-filtered image.
%

```

```

% Input: Bitmap Bayer-filtered image.
%
% Output: Bitmap Reconstructed RGB-image.
%
% Author: Erik Thiesen (ET)
% Last Revision: March 3, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function im=BayerGR_fast(ImBW)

im=zeros([size(ImBW) 3]);
for y=2:2:size(ImBW,1)-1
    for x=2:2:size(ImBW,2)-1
        im(y,x,1)=(ImBW(y-1,x-1)+ImBW(y+1,x-1)+ImBW(y-1,x+1)+ImBW(y+1,x+1))/4;
        im(y,x,2)=(ImBW(y-1,x)+ImBW(y+1,x)+ImBW(y,x-1)+ImBW(y,x+1))/4;
        im(y,x,3)=ImBW(y,x);
    end
end
for y=2:2:size(ImBW,1)-1
    for x=3:2:size(ImBW,2)-1
        im(y,x,1)=(ImBW(y-1,x)+ImBW(y+1,x))/2;
        im(y,x,3)=(ImBW(y,x-1)+ImBW(y,x+1))/2;
        im(y,x,2)=(4*ImBW(y,x)+ImBW(y-1,x-1)+ImBW(y-1,x+1)+ImBW(y+1,x-1)+ImBW(y+1,x+1))/8;
    end
end
for y=3:2:size(ImBW,1)-1
    for x=2:2:size(ImBW,2)-1
        im(y,x,1)=(ImBW(y,x-1)+ImBW(y,x+1))/2;
        im(y,x,2)=(4*ImBW(y,x)+ImBW(y-1,x-1)+ImBW(y-1,x+1)+ImBW(y+1,x-1)+ImBW(y+1,x+1))/8;
        im(y,x,3)=(ImBW(y-1,x)+ImBW(y+1,x))/2;
    end
end
for y=3:2:size(ImBW,1)-1
    for x=3:2:size(ImBW,2)-1
        im(y,x,3)=(ImBW(y-1,x-1)+ImBW(y+1,x-1)+ImBW(y-1,x+1)+ImBW(y+1,x+1))/4;
        im(y,x,2)=(ImBW(y-1,x)+ImBW(y+1,x)+ImBW(y,x-1)+ImBW(y,x+1))/4;
        im(y,x,1)=ImBW(y,x);
    end
end
end

```

E.3 main01_SGE.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "main01_SGE.m"
%
% Description: Script performing Kernel-based background modelling (Elgammal),
% and noise reduction, used for detecting moving foreground objects.
%
% Input: Bitmap images of video sequence to analyze
%
% Output: Bitmap images of detected foreground regions before and after
% noise reduction, and mean image of background model for every analyzed
% frame
%
% Remarks: Input images should be named "Test???.bmp", where ??? is the
% frame number. This is a revised version of the file "main01.m" made by
% Morten Hansen (MH). Remarks in danish are due to MH.
%
% Author: Morten Hansen (MH)
% Revised by: Søren Erbou (SGE)
% Last Revision: September 13, 2004

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc
close all
clear

TESTS=[424:435];
for c=1:length(TESTS)
    % Initialization
    TestNo=TESTS(c);
    TestFolder=['Test',num2str(TestNo)];
    DestFolder=['SGE\Video\'',TestFolder,'_Object'];

    if exist(['E:\'',DestFolder'],'dir')~=7
        Dir=pwd;
        cd('E:\');
        mkdir(DestFolder);
        cd(Dir);
    end
    DestFolder=['E:\'',DestFolder,'\''];
    addpath(['E:\SGE\Video\'',TestFolder]);

    %----Erkl ring af variable-----
    Fit_to_baggrund = 10; %Tid i sek. der skal g  for en forgrundspixel tilskrives baggrunden
    min_areal = 50;      %Minimum areal der skal benyttes i "noise_reduction.m"
    taerskel = 1e-20;    %T rskelv rdi
    N = 10;              %Antallet af frames der skal benyttes i
    %estimattet for pixel intensitets fordelingen

    global FRAME_RATE; %Global variabel
    FRAME_RATE = 20;    %Den frame rate videosekvenserne er optaget med
    global FRAME_JUMP; %Global variabel
    FRAME_JUMP = 4;     %Springet ml frames
    fps = FRAME_RATE/FRAME_JUMP; %Antal frames pr. sekund
    name1 = [TestFolder,'_'];
    name2 = ['.bmp'];
    %Nummeret p  den frame som programmet skal starte ved
    ChooseFrames;
    %----START P  PROGRAM-----
    frame_no = start_frame;
    %----Opstartsfase-----
    %Filnavn p  forgrundsbillede som skal indl ses
    str = num2str(frame_no);
    filename = [name1,str,name2];
    [Y,X,Z] = size(imread(filename));
    %Variabel der holder styr p  hvilken frame der skal overskrives for hver pixel
    var1 = uint8(ones(1,X*Y));

    %Allokerer plads i hukommelsen til N billeder (RGB).
    pic_data = uint8(zeros(N,Y*X,3));
    for n = 1:N;
        pic_in = imread(filename);
        pic_data((n-1)*X*Y*Z+1:n*X*Y*Z) = pic_in(:)';
        frame_no = frame_no + FRAME_JUMP;
        str = num2str(frame_no);
        filename = [name1,str,name2];
    end
    end
    Foreground_history = uint8(zeros(1,Y*X));
    fg_data={};
    fg_ind={};
    fg_count=1;

```



```

%----Efter opstartsfasen-----
foerste_fr = frame_no + FRAME_JUMP;

while (frame_no < stop_frame)
    t=cputime;
    pack;
    Differens = uint8(abs(diff(double(reshape(pic_data,X*Y*Z,N)'),1,1)));
    Medi = uint8(median(double(Differens),1)); %Finder medianen af differensen
    Medi(find(Medi==0))=1;
    MediInv=1./double(Medi);
    KernelConst = (0.68/sqrt(pi))*MediInv;
    ExpConst = -(0.68^2)*MediInv.^2;

    %---Variable tælles op-----
    frame_no = frame_no + FRAME_JUMP;
    %-----

    %----Indlæser nyt billede-----
    str = num2str(frame_no);
    filename = [name1,str,name2];
    pic_temp = imread(filename);
    pic_in = pic_temp(:)';
    %-----

    %----Estimerer intensitets fordelingen for pixels-----
    Pr = zeros(1,X*Y);
    pic_bg_gray = zeros(1,X*Y);
    for i = 1:N,
        Temp = KernelConst.*exp(ExpConst.*(double(pic_in)-...
            double(pic_data((i-1)*X*Y*Z+1:i*X*Y*Z))).^2);
        Pr = Pr + Temp(X*Y+1:2*X*Y).*Temp(2*X*Y+1:end);
    end
    Pr = Pr/N;
    %-----

    %----Evaluering af billede i forhold til tærskelværdi
    Pic_foreground = (Pr < taerskel);
    %-----

    %----Opdatering af baggrundsmodel-----
    %Opdateringen foregår på pixel plan
    %Baggrundspixels anvendes til at opdatere baggrundsmodellen
    IndBG=find(Pic_foreground==0);
    IndBG_rgb=reshape((((1:Z)-1)*ones(1,length(IndBG))*X*Y+...
        repmat(IndBG,Z,1))',1,length(IndBG)*Z);
    pic_data(repmat((double(var1(IndBG))-1).*X*Y*Z,1,3)+IndBG_rgb)=pic_in(IndBG_rgb);

    IndFG=find(Pic_foreground==1); %Forgrundspixels
    IndFG_rgb=uint32(reshape((((1:Z)-1)*ones(1,length(IndFG))*X*Y+...
        repmat(IndFG,Z,1))',1,length(IndFG)*Z));
    %Forgrundspixels historie inkrementeres
    Foreground_history(IndFG) = uint8(double(Foreground_history(IndFG))+1);
    fg_data{fg_count} = pic_in(IndFG_rgb);
    fg_ind{fg_count} = IndFG_rgb;
    %Pixels der har været forgrund længe tilskrives baggrund.
    IndFG2BG=find(Foreground_history>=Fit_to_baggrund*fps);

    if length(IndFG2BG)>0
        IndFG2BG_rgb=reshape((((1:Z)-1)*ones(1,length(IndFG2BG))*X*Y+...
            repmat(IndFG2BG,Z,1))',1,length(IndFG2BG)*Z);

```

```

% Hvis fg-pixels sættes til bg-pixels, skal de sidste
% Fit_to_baggrund*fps antal frames i baggrundsmodellen opdateres til
% forgrundpixelen.
temp_fg_count=fg_count;
for i=0:Fit_to_baggrund*fps-1
    var1_temp=double(var1(IndFG2BG))-i;
    IndTemp=1;
    while length(IndTemp)
        IndTemp=find(var1_temp<1);
        var1_temp(IndTemp)=var1_temp(IndTemp)+N;
    end
    IndTemp2=find(ismember(fg_ind{temp_fg_count},IndFG2BG_rgb)==1);
    pic_data(repmat((var1_temp-1).*X*Y*Z,1,3)+IndFG2BG_rgb) = ...
        fg_data{temp_fg_count}(IndTemp2);

    temp_fg_count=temp_fg_count-1;
    if temp_fg_count<1
        temp_fg_count=Fit_to_baggrund*fps;
    end

end

end
end
Foreground_history([IndFG2BG,IndBG])=uint8(0);
% SGE - Baggrundsgråtonebilleder opdateres
pic_bg_rgb_mean=reshape(uint8(mean(double(reshape(pic_data,Y*X*Z,N)),2)')),Y,X,Z);

%Vælger hvilken baggrundsframe der skal opdateres:
var1([IndBG,IndFG2BG]) = uint8(double(var1([IndBG,IndFG2BG])) + 1);
IndVar1=find(var1>N);
var1(IndVar1)=1;

fg_count=fg_count+1;
if fg_count>Fit_to_baggrund*fps
    fg_count=1;
end

if 1
    %-----
    Pic_foreground = reshape(Pic_foreground,Y,X);
    imwrite(Pic_foreground,[DestFolder,name1,['binært_'],str,'_Noisy',name2]);
    %-----Støjreduktion SGE-----
    bin_pic = Noise_Reduction_SGE(Pic_foreground,min_areal);
    imwrite(bin_pic,[DestFolder,name1,['binært_'],str,'_SGE',name2]);

    %-----
    %-----Shadow removal MH-----
    imwrite(pic_bg_rgb_mean,[DestFolder,name1,'middel_',str,'_bg_rgb_SGE',name2]);

    %-----
    if 0
        %-----Foretager tracking af objekt-----
        track_before_detect(name1,frame_no)
        %-----

        %--Funktion der viser de forskellige stadier i systemet--
        plot_4_windows(name1,frame_no)
        %-----
    end
end
end
%End på while-løkke

```

```

disp([TestFolder,' ', Frame no. ', num2str(frame_no), ' done in ', ...
      num2str(cputime-t,3), ' seconds...']);

clear Medi MediInv KernelConst ExpConst bin_pic Pic_foreground pic_bg_rgb_mean pic_in
clear Differens Temp pic_bg_gray IndBG IndBG_rgb IndFG IndFG_rgb Pr pic_temp
pack;
end
end

disp('Program kørt færdig')

```

E.4 ChooseFrames.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "ChooseFrames.m"
%
% Description: Script that determines the frame numbers of different test
% sequences.
%
% Input: Sequence number and number of images processed so far
%
% Output: Start and Stop frame
%
% Remarks: Used by "main01_SGE.m"
%
% Author: Søren Erbou (SGE)
% Last Revision: June 11, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

switch TestNo
    case 12      % 60,460
        start_frame = 60-(N+1)*FRAME_JUMP;
        stop_frame = 460;
    case 27      % 45,797 ;209-350; Store lysskift 350-700
        start_frame = 209-(N+1)*FRAME_JUMP;
        stop_frame =350;
    otherwise
        start_frame = 45-(N+1)*FRAME_JUMP;
        FileList=dir(['E:\SGE\Video\' ,TestFolder,'*.bmp']);
        FRAMENO=[];
        for q=1:length(FileList)
            UnderscoreInd=find(FileList(q).name=='_');
            FRAMENO(end+1)=str2double(FileList(q).name(UnderscoreInd+1:end-4));
        end
        stop_frame = max(FRAMENO);
end
end

```

E.5 Noise_Reduction_SGE.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Noise_Reduction_SGE.m"
%
% Description: Performs noise reduction on detected foreground objects
%
% Input: Noisy foreground mask
%
% Output: Noisereduced foreground mask
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function pic_out = Noise_Reduction_SGE(pic_in,min_areal_1);

pic_out = zeros(size(pic_in));
% connected component with "8-connected neighbors"
[L,NUM] = bwlabeln(pic_in);
%compute area
S = regionprops(L, 'Area');
% removes small segments
pic_out = ismember(L, find([S.Area] >= min_areal_1));
%remove noise
se = strel('disk',3);
pic_out=imclose(pic_out,se);
pic_out=imopen(pic_out,se);
pic_out=imerode(pic_out,strel('disk',2));

```

E.6 Do_Shadow.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Do_Shadow.m"
%
% Description: Script applying various methods for shadow removal,
% computing performance measures, and comparing methods.
%
% Input: Bitmap images of object to analyze, mask of foreground detected
% object with and without noise, clean version of foreground mask,
% background image, possibly manually labelled image.
%
% Output: Various figures in "png"-format, and images and variables in
% "mat"- or "bmp"-format.
%
% Remarks: Input images should be produced using the "main01_SGE.m" file.
% Before applying "Do_Shadow.m" a clean version of the foreground mask
% should be produced manually, with the extension "Clean_?.bmp" to the
% filename, where ? denotes the object number, (usually only one object per
% sequence). Example: If the noisereduced foreground mask is named
% "Test400_binart_101_SGE.bmp", then the clean version should be named
% "Test400_binart_101_SGE_Clean_1.bmp"
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;
clear all;
clc;

global DestFolder
global TestFolder
global ImNo
global ObjectNo
global Perf;

CHOOSE_DATASET=1; %0=Traing set, 1= Test Set

DoLabelling=1; %0=none, 1=Write image to bmp for manually labelling.
% 1 should be performed prior to applying DoPerformance.
DoShadow=3; %0=none, 1=Javed 2=JavedImproved, 3=Finlayson using FG mask and enhanced segmentation,
%4=Finlayson with foreground mask and enhanced segmentation.
% 1 or 2 should be performed prior to 3 and 4

```

```

UseJavedImproved=1; %0=Use Javeds color segmentation, 1=Use Improved color segmentation
%Determines which color segmentation to use in DoPerformance
DoPerformance=1; %0=none, 1=Javed, 2=Enhanced, 3=figures of objects, 4=figure of training examples
% 3 and 4 produce figures similar to those seen in the report
DoComparison=1; % Collects data from all examples and saves it in a single file for anlysis

Tight=1;
Offset = 0; % How much darker than the background image should shadow candidates be...

%%%% Javed %%%%
% K-Means
VAR=[81]; %[25 36 49 64 81 100];
% Merging
MERGINGSIZE=[10]; %[10 30 50 70 100 150]
% Correlation
CORRTHRES=[0.15]; %[0 0.05 0.1 0.15 0.2 0.3]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Improved %%%%
VAROFFSET=[4];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Enhanced %%%%
RBTHRES=[3]; % [3,5,7]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if CHOOSE_DATASET
    DestFolder=['SGE\Video\FilesTestSet'];
else
    DestFolder=['SGE\Video\FilesTrainingSet'];
end
Drive = 'E:\';

if exist([Drive, DestFolder], 'dir')~=7
    Dir=pwd;
    cd(Drive);
    mkdir([DestFolder]);
    cd(Dir);
end
if exist([Drive, DestFolder, '\Performance'], 'dir')~=7
    Dir=pwd;
    cd(Drive);
    mkdir([DestFolder, '\Performance',]);
    cd(Dir);
end
DestFolder=[Drive, DestFolder, '\'];
addpath(DestFolder);

DATASET=DataSets(CHOOSE_DATASET); % 0=Training Set (18 examples), 1=Test Set (72 examples)

Perf.ac=-1*ones(size(DATASET,1),4);
Perf.tp=-1*ones(size(DATASET,1),4);
Perf.fp=-1*ones(size(DATASET,1),4);
Perf.tn=-1*ones(size(DATASET,1),4);
Perf.fn=-1*ones(size(DATASET,1),4);
Perf.NoOfPixels=-1*ones(size(DATASET,1),4);
Perf.TrueObj=-1*ones(size(DATASET,1),4);
Perf.TrueShadow=-1*ones(size(DATASET,1),4);
Perf.TrueSelfShadow=-1*ones(size(DATASET,1),4);

% Loop over all sets of variables

```

```

for VarCount=1:length(VAR)
    for MerCount=1:length(MERGINGSIZE)
        for CorrCount=1:length(CORRTHRES)
            for VarOffsetCount=1:length(VAROFFSET)
                for RBCount=1:length(RBTHRES)
                    RBThreshold=RBTHRES(RBCount);
                    VarOffset=VAROFFSET(VarOffsetCount);
                    Var=VAR(VarCount);
                    MergingSizeLimit=MERGINGSIZE(MerCount);
                    CorrThreshold=CORRTHRES(CorrCount);
                    if mod(CorrThreshold,0.1)==0
                        Parameters=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit),...
                                    '_Corr',sprintf('%.1f',CorrThreshold),'_'];
                    else
                        Parameters=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit),...
                                    '_Corr',sprintf('%.2f',CorrThreshold)];
                    end
                    ParametersImproved=['_VarOffset',num2str(VarOffset)];
                    ParametersEnhanced=['_RB',sprintf('%.1f',RBThreshold),'_'];
                    count=0;
                    % loop over examples in DATASET
                    for j=[1:size(DATASET,1)]
                        count=count+1;
                        TestNo=DATASET(j,1);
                        ImNo=DATASET(j,2);
                        ObjectNo=DATASET(j,3);
                        TestFolder=['Test',num2str(TestNo)];
                        disp(['Processing ',TestFolder,' - Frame no. ',num2str(ImNo),...
                            ' - Object No. ',num2str(ObjectNo),'...']);

                        addpath(['E:\SGE\Video\'',TestFolder]);
                        addpath(['E:\SGE\Video\'',TestFolder,'_Object']);

                        filename=[TestFolder,'_',num2str(ImNo),'.bmp'];
                        fg_filename=[TestFolder,'_binart_',num2str(ImNo),'_SGE.bmp'];
                        fg_clean_filename=[TestFolder,'_binart_',num2str(ImNo),'_SGE_Clean_',...
                            num2str(ObjectNo),'.bmp'];

                        fg_noisy_filename=[TestFolder,'_binart_',num2str(ImNo),'_Noisy.bmp'];
                        bg_filename=[TestFolder,'_middel_',num2str(ImNo),'_bg_rgb_SGE.bmp'];
                        label_filename=[DestFolder,TestFolder,'_TrueLabel_',num2str(ImNo),'.mat'];

                        pic_in = imread(filename);
                        pic_in_org=pic_in;
                        if (exist(fg_clean_filename)==2)
                            fg_mask_full = double(imread(fg_clean_filename));
                        else
                            fg_mask_full = double(imread(fg_filename));
                        end

                        fg_noisy_mask = double(imread(fg_noisy_filename));
                        bg_in = imread(bg_filename);

                        Stat_All = regionprops(fg_mask_full, 'Area','BoundingBox');
                        Border= 10;

                        if length(Stat_All)
                            [YOrg,XOrg,Z]=size(pic_in_org);
                            Stat_All = regionprops(fg_mask_full, 'Area','BoundingBox');
                            BBoxAll=Stat_All.BoundingBox;
                            BBoxAll=[BBoxAll(1)-Border,BBoxAll(2)-Border,BBoxAll(3)+2*Border,...

```

```

        BBoxAll(4)+2*Border];
if BBoxAll(1)<1
    BBoxAll(3)=BBoxAll(3)-(0.5-BBoxAll(1));
    BBoxAll(1)=0.5;
end
if BBoxAll(1)+BBoxAll(3)>X0rg+1
    BBoxAll(3)=X0rg-BBoxAll(1)-0.5;
end
if BBoxAll(2)<1
    BBoxAll(4)=BBoxAll(4)-(0.5-BBoxAll(2));
    BBoxAll(2)=0.5;
end
if BBoxAll(2)+BBoxAll(4)>Y0rg+1
    BBoxAll(4)=Y0rg-BBoxAll(2)-0.5;
end

if Tight
    pic_in=imcrop(pic_in,BBoxAll);
    fg_mask=imcrop(fg_mask_full,BBoxAll);
    fg_noisy_mask=imcrop(fg_noisy_mask,BBoxAll);
    bg_in=imcrop(bg_in,BBoxAll);
else
    fg_mask=fg_mask_full;
end
[Y,X,Z]=size(pic_in);
fg_mask = fg_mask;
dark_mask = double((pic_in(:,:,1))<(double(bg_in(:,:,1))+Offset)) & ...
    double((pic_in(:,:,2))<(double(bg_in(:,:,2))+Offset)) & ...
    double((pic_in(:,:,3))<(double(bg_in(:,:,3))+Offset)));

dark_mask = dark_mask.*fg_mask;
dark_mask_full=uint8(zeros(size(fg_mask_full)));
dark_mask_full(round(BBoxAll(2):(round(BBoxAll(2))+fix(BBoxAll(4))),...
    round(BBoxAll(1):(round(BBoxAll(1))+fix(BBoxAll(3))))=dark_mask;

[L,NUM] = bwlabeln(dark_mask);
Stat = regionprops(L, 'Area', 'BoundingBox');
dark_mask = ismember(L, find([Stat.Area] >= 20));

fg_only=uint8(zeros(Y,X,Z));
dark_fg_white=uint8(zeros(Y,X,Z));
dark_fg=uint8(zeros(Y,X,Z));
for i=1:Z
    fg_only(:,:,i) = uint8(double(pic_in(:,:,i)).*fg_mask);
    dark_fg(:,:,i) = uint8(double(fg_only(:,:,i)).*dark_mask);
    dark_fg_white(:,:,i)=uint8(double(dark_fg(:,:,i))+...
        (255.*ones(Y,X).*~dark_mask));
end

if DoLabelling==1
    if Tight
        imwrite(dark_fg_white,[DestFolder,TestFolder,'_LabelIm_',...
            num2str(ImNo),'_',num2str(ObjectNo),'_bmp']);
    else
        imwrite(dark_fg_white,[DestFolder,TestFolder,...
            '_LabelImFullSize_',num2str(ImNo),'_',...
            num2str(ObjectNo),'_bmp']);
    end
    F4=figure(4);
    subplot 121
    imshow(pic_in,[0 255]);

```



```

%
% Description: Function returning the final datasets
%
% Input: Variable denoting training set or test set
%
% Output: Vector of appropriate dataset, containing sequence number, image
% number, and object number.
%
% Author: Søren Erbou (SGE)
% Last Revision: August 27, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [DataSet] = DataSets(Set)

TrainingSet=[36,189,1;...   % [TestNo, ImNo, ObjectNo, ...
    45,141,1;...
    46,109,1;...
    53,205,1;...
    300,429,1;...
    420,425,1;...
    429,141,1;...
    432,233,1;...
    434,245,1;...
    75,101,1;...
    352,145,1;...
    411,169,1;...
    419,297,1;...
    400,101,1;...
    402,89,1;...
    409,165,1;...
    421,141,1;...
    435,49,1];

TestSet=[12,136,1;...
    13,57,1;...
    20,209,1;...
    21,277,1;...
    22,125,1;...
    23,249,1;...
    24,113,1;...
    24,581,2;...
    24,701,3;...
    24,745,4;...
    25,101,1;...
    25,285,2;...
    27,737,3;...
    29,105,1;...
    29,141,2;...
    30,61,1;...
    31,45,1;...
    31,77,2;...
    32,241,1;...
    32,697,2;...
    37,149,1;...
    45,293,2;...
    47,173,1;...
    48,185,1;...
    49,161,1;...
    52,61,1;...
    74,81,1;...
    75,181,2;...

```

```

76,93,1;...
77,105,1;...
78,141,1;...
79,237,1;...
80,69,1;...
81,277,1;...
81,393,2;...
82,337,1;...
82,361,2;...
83,297,1;...
83,321,2;...
300,233,3;...
300,429,2;...
300,561,4;...
302,165,1;...
303,61,1;...
400,185,2;...
400,185,3;...
403,469,1;...
406,113,1;...
407,309,1;...
408,233,1;...
410,157,1;...
412,69,1;...
412,221,2;...
413,121,1;...
414,193,1;...
415,261,1;...
416,213,1;...
416,521,2;...
418,69,1;...
419,501,2;...
421,169,2;...
422,249,1;...
423,213,1;...
424,237,1;...
425,145,1;...
426,65,1;...
428,193,1;...
430,177,1;...
431,137,1;...
433,109,1;...
434,465,2;...
435,297,2];

```

```

if Set==0
    DataSet=TrainingSet;
else
    DataSet=TestSet;
end

```

E.8 Javed.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Javed.m"
%
% Description: Function that applies Javed's method for shadow removal.
% K-means, connected component annalysis, region merging and
% classification.
%
% Input: Bitmap images of frame to analyze, mask of foreground detected

```

```

% object without noise, mask of shadow candidates, background image,
% various parameters.
%
% Output: Color segmented image, and correlation vector as "mat"-files
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Javed(pic_fg_rgb,pic_fg_bin_all,pic_bg_rgb,pic_fg_bin_cand,DarknessMargin,...
              MahThres,Var,MergingSizeLimit,CorrThreshold)

global DestFolder
global TestFolder
global ImNo
global ObjectNo

if mod(CorrThreshold,0.1)==0
    Parameters=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit),...
               '_Corr',sprintf('%.1f',CorrThreshold),'_'];
else
    Parameters=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit),...
               '_Corr',sprintf('%.2f',CorrThreshold)];
end
disp([Parameters]);
ind_fg_cand = find(pic_fg_bin_cand==1);
ind_fg_all_not_cand = find(pic_fg_bin_all&~pic_fg_bin_cand);
[Y,X,Z]=size(pic_fg_rgb);

filename_JavedMerged=[DestFolder,TestFolder,'_JavedMerged_',num2str(ImNo),'_',...
                     num2str(ObjectNo),Parameters(1:end-8),'.mat'];

t=cputime;
% Is Javed's color segmentation already done?
if exist(filename_JavedMerged)~=2

    disp('K-Means...');

    %% K-Means
    SigmaInv=(1/Var)*eye(Z);      % Fixed standard deviation on all distributions

    pic=reshape(pic_fg_rgb,Y*X,Z);
    pix_cand_rgb=double(pic(ind_fg_cand,1:Z)');
    no_of_pix=size(pix_cand_rgb,2);
    Mean=pix_cand_rgb(:,1); % Intialised with a distribution centered on the first pixel candidate
    pix_in_dist=1;          % Number of pixels assigned to a specific distribution
    pix_class_raw=1;
    Mah=0;
    for i=2:no_of_pix      % Every pixel is classified
        j=1;
        MatchFound=0;
        while (j<=size(Mean,2))&(~MatchFound) % Tested on every existing distribution
            Centered=pix_cand_rgb(:,i)-Mean(:,j);
            Mah(i)=Centered'*SigmaInv*Centered; % Squared Mahanalobis distance
            if Mah(i)<MahThres^2 % Mahanalobis distance measure used for assignment
                pix_in_dist(j)=pix_in_dist(j)+1;
                Mean(:,j)=Mean(:,j)+1./(pix_in_dist(j)).*(Centered); % Distribution updated
                pix_class_raw(i)=j;
                MatchFound=1;
            else
                if j==size(Mean,2) % New distribution added if no match on existing distributions
                    Mean(:,end+1)=pix_cand_rgb(:,i);
                end
            end
        end
    end
end

```

```

                pix_in_dist(end+1)=1;
                pix_class_raw(i)=j+1;
                MatchFound=1;
            end
            j=j+1;          % If no match in j'th distribution, j is incremented
        end
    end
end

pic_fg_class_raw=uint8(zeros(Y,X));
pic_fg_class_raw(ind_fg_cand)=uint8(pix_class_raw);
pic_fg_class_raw_rgb=label2rgb(pic_fg_class_raw);

%% Connected Components and merging
disp(['... ',num2str(cputime-t,4),' seconds.']);
disp(['Connected Components...']);
t=cputime;
pic_fg_class_conn=zeros(Y,X);
no_of_conn_classes=0;
for j=1:size(Mean,2)
    TempIm=uint8(zeros(Y,X));
    TempIndRow{j}=find(pix_class_raw==j);
    TempIm(ind_fg_cand(TempIndRow{j}))=1;
    [TempImConn,no_Temp_classes]=bwlabel(TempIm,8);
    TempImConn(ind_fg_cand(TempIndRow{j}))=TempImConn(ind_fg_cand(TempIndRow{j}))+...
        no_of_conn_classes;
    pic_fg_class_conn=pic_fg_class_conn+double(TempImConn);
    no_of_conn_classes=no_of_conn_classes+no_Temp_classes;
end
%% Merging
disp(['... ',num2str(cputime-t,4),' seconds.']);
disp(['Merging...']);
t=cputime;

[pic_fg_class_merged,Stats,no_of_merged_classes]=...
    MergeRegions(pic_fg_class_conn,MergingSizeLimit);
pic_fg_class_merged_rgb=label2rgb(pic_fg_class_merged);
else
    disp('Merging exists...');
    load(filename_JavedMerged);
    no_of_merged_classes=max(pic_fg_class_merged(:));
end

%% Gradient
disp(['... ',num2str(cputime-t,4),' seconds.']);
disp(['Classification...']);
t=cputime;

[fx_bg,fy_bg]=gradient(double(pic_bg_rgb));
[fx_fg,fy_fg]=gradient(double(pic_fg_rgb));
theta_bg=reshape(atan2(fy_bg,fx_bg),Y*X,Z);
theta_fg=reshape(atan2(fy_fg,fx_fg),Y*X,Z);
castshadow=[];
object_class=[];
pic_fg_class_final=uint8(zeros(Y,X));

% Classification of regions
for j=1:no_of_merged_classes
    ind_class{j}=find(pic_fg_class_merged==j);
    CorrData=[reshape(theta_fg(ind_class{j},:),length(ind_class{j})*Z,1),...
        reshape(theta_bg(ind_class{j},:),length(ind_class{j})*Z,1)];

```

```

if ~sum(var(CorrData)==0)
    Temp=corrcoef(CorrData);
else
    Temp=[1;0];
end
Corr(j,1)=Temp(2,1);
if Corr(j,1)>CorrThreshold
    castshadow(end+1,1)=j;
    pic_fg_class_final(ind_class{j})=uint8(128);
else
    object_class(end+1,1)=j;
    pic_fg_class_final(ind_class{j})=uint8(255);
end
end

save([DestFolder,TestFolder,'_JavedMerged_',num2str(ImNo),'_',num2str(ObjectNo),...
    Parameters(1:end-8),'.mat'],'pic_fg_class_merged');
save([DestFolder,TestFolder,'_JavedCorr_',num2str(ImNo),'_',num2str(ObjectNo),...
    Parameters(1:end-8),'.mat'],'Corr');
imwrite(pic_fg_class_final,[DestFolder,TestFolder,'_JavedLabel_',num2str(ImNo),...
    '__',num2str(ObjectNo),Parameters,'.bmp']);

pic_fg_class_final(ind_fg_all_not_cand)=uint8(3);
disp(['...',num2str(cputime-t,4),' seconds.']);
pic_fg_rgb_all=uint8([]);
pic_fg_rgb_cand=uint8([]);
pic_out=pic_fg_bin_cand;

```

E.9 JavedImproved.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "JavedImproved.m"
%
% Description: Function that applies the improved color segmentation used
% for shadow removal. K-means, connected component annalysis, region merging and
% classification.
%
% Input: Bitmap images of frame to analyze, mask of foreground detected
% object without noise, mask of shadow candidates, background image,
% various parameters.
%
% Output: Color segmented image, and correlation vector as "mat"-files
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function JavedImproved(pic_fg_rgb,pic_fg_bin_all,pic_bg_rgb,pic_fg_bin_cand,DarknessMargin,...
    MahThres,VarOffset,Var,MergingSizeLimit,CorrThreshold)

global DestFolder
global TestFolder
global ImNo
global ObjectNo

if mod(CorrThreshold,0.1)==0
    Parameters=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit),'_Corr',sprintf('%1f',...
        CorrThreshold),'_'];
else
    Parameters=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit),'_Corr',sprintf('%2f',...
        CorrThreshold)];

```

```

end

ParametersImproved=['_VarOffset',num2str(VarOffset)];
disp([ParametersImproved,Parameters]);

ind_fg_cand = find(pic_fg_bin_cand==1);
ind_fg_all_not_cand = find(pic_fg_bin_all&~pic_fg_bin_cand);
[Y,X,Z]=size(pic_fg_rgb);

% Variance as a function of intensity
VarianceFunction=[VarOffset+[0:127]'.*(Var-VarOffset)/127;Var*ones(128,1)];

filename_JavedMerged=[DestFolder,TestFolder,'_JavedImprovedMerged_',num2str(ImNo),'_',...
    num2str(ObjectNo),ParametersImproved,Parameters(1:end-8),'.mat'];
t=cputime;

% Is the improved color segmentation already done?
if exist(filename_JavedMerged)~=2

    disp('K-Means improved...');

    %% K-Means
    pic=reshape(pic_fg_rgb,Y*X,Z);
    pix_cand_rgb=double(pic(ind_fg_cand,1:Z)');
    no_of_pix=size(pix_cand_rgb,2);
    Mean=pix_cand_rgb(:,1); % Intialised with a distribution centered on the first pixel candidate
    % Variance of distribution a function of illumination
    Variance=VarianceFunction(fix(mean(Mean(:,1)))));
    SigmaInv=(1/Variance)*eye(Z);
    pix_in_dist=1; % Number of pixels assigned to a specific distribution
    pix_class_raw=1;
    Mah=0;
    for i=2:no_of_pix % Every pixel is classified
        j=1;
        MatchFound=0;
        while (j<=size(Mean,2)&(~MatchFound)) % Tested on every existing distribution
            Centered=pix_cand_rgb(:,i)-Mean(:,j);
            SigmaInv=(1/Variance(j))*eye(Z);
            Mah(i)=Centered'*SigmaInv*Centered; % Squared Mahanalobis distance
            if Mah(i)<MahThres^2 % Mahanalobis distance measure used for assignment
                pix_in_dist(j)=pix_in_dist(j)+1;
                Mean(:,j)=Mean(:,j)+1./(pix_in_dist(j)).*(Centered); % Distribution updated
                Variance(j)=VarianceFunction(fix(mean(Mean(:,j)))));
                pix_class_raw(i)=j;
                MatchFound=1;
            else
                if j==size(Mean,2) % New distribution added if no match on existing distributions
                    Mean(:,end+1)=pix_cand_rgb(:,i);
                    Variance(end+1)=VarianceFunction(fix(mean(Mean(:,j)))));
                    pix_in_dist(end+1)=1;
                    pix_class_raw(i)=j+1;
                    MatchFound=1;
                end
                j=j+1; % If no match in j'th distribution, j is incremented
            end
        end
    end
end

pic_fg_class_raw=uint8(zeros(Y,X));
pic_fg_class_raw(ind_fg_cand)=uint8(pix_class_raw);
pic_fg_class_raw_rgb=label2rgb(pic_fg_class_raw);

```

```

%% Connected Components and merging
disp(['... ', num2str(cputime-t,4), ' seconds.']);
disp(['Connected Components...']);
t=cputime;
pic_fg_class_conn=zeros(Y,X);
no_of_conn_classes=0;
for j=1:size(Mean,2)
    TempIm=uint8(zeros(Y,X));
    TempIndRow{j}=find(pix_class_raw==j);
    TempIm(ind_fg_cand(TempIndRow{j}))=1;
    [TempImConn,no_Temp_classes]=bwlabel(TempIm,8);
    TempImConn(ind_fg_cand(TempIndRow{j}))=...
        TempImConn(ind_fg_cand(TempIndRow{j}))+no_of_conn_classes;
    pic_fg_class_conn=pic_fg_class_conn+double(TempImConn);
    no_of_conn_classes=no_of_conn_classes+no_Temp_classes;
end
%% Merging
disp(['... ', num2str(cputime-t,4), ' seconds.']);
disp(['Merging...']);
t=cputime;

[pic_fg_class_merged,Stats,no_of_merged_classes]=...
    MergeRegions(pic_fg_class_conn,MergingSizeLimit);
pic_fg_class_merged_rgb=label2rgb(pic_fg_class_merged);

else
    disp('Merging exists...');
    load(filename_JavedMerged);
    no_of_merged_classes=max(pic_fg_class_merged(:));
end

%% Gradient
disp(['... ', num2str(cputime-t,4), ' seconds.']);
disp(['Classification...']);
t=cputime;

[fx_bg,fy_bg]=gradient(double(pic_bg_rgb));
[fx_fg,fy_fg]=gradient(double(pic_fg_rgb));
theta_bg=reshape(atan2(fy_bg,fx_bg),Y*X,Z);
theta_fg=reshape(atan2(fy_fg,fx_fg),Y*X,Z);
castshadow=[];
object_class=[];
pic_fg_class_final=uint8(zeros(Y,X));
% Classification of regions
for j=1:no_of_merged_classes
    ind_class{j}=find(pic_fg_class_merged==j);
    CorrData=[reshape(theta_fg(ind_class{j},:),length(ind_class{j})*Z,1),...
        reshape(theta_bg(ind_class{j},:),length(ind_class{j})*Z,1)];
    if ~sum(var(CorrData)==0)
        Temp=corrcoef(CorrData);
    else
        Temp=[1;0];
    end
    Corr(j,1)=Temp(2,1);
    if Corr(j,1)>CorrThreshold
        castshadow(end+1,1)=j;
        pic_fg_class_final(ind_class{j})=uint8(128);
    else
        object_class(end+1,1)=j;
        pic_fg_class_final(ind_class{j})=uint8(255);
    end
end

```

```

end
end

save([DestFolder,TestFolder,'_JavedImprovedMerged_',num2str(ImNo),'_',num2str(ObjectNo),...
     ParametersImproved,Parameters(1:end-8),'.mat'],'pic_fg_class_merged');
save([DestFolder,TestFolder,'_JavedImprovedCorr_',num2str(ImNo),'_',num2str(ObjectNo),...
     ParametersImproved,Parameters(1:end-8),'.mat'],'Corr');
imwrite(pic_fg_class_final,[DestFolder,TestFolder,'_JavedImprovedLabel_',num2str(ImNo),...
        '_ ',num2str(ObjectNo),ParametersImproved,Parameters,'.bmp']);
pic_fg_class_final(ind_fg_all_not_cand)=uint8(3);
disp(['...',num2str(cputime-t,4),' seconds.']);

pic_fg_rgb_all=uint8([]);
pic_fg_rgb_cand=uint8([]);
pic_out=pic_fg_bin_cand;

```

E.10 MergeRegions.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "MergeRegions.m"
%
% Description: Function that merges neighboring segments smaller than AreaLimit with
% their largest neighbor.
%
% Input: A labelled image with all classes somehow connected through each other.
%
% Output: The merged image and its statistics.
%
% Author: Søren Erbou (SGE)
% Last Revision: June 12, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ImMerged,stat,NoOfSegments] = MergeRegions(ImLabel,AreaLimit)

% Arrange color segmented regions by size
ImMerged=ImLabel;
LabInd=find(ImMerged>0);
Label=ImMerged(LabInd);
Num=max(ImMerged(:));
Bins=1:Num;
Area=histc(Label,Bins);
[AreaSort,AreaInd]=sort(Area);
AreaSort=AreaSort(end:-1:1);
AreaInd=AreaInd(end:-1:1);
run=1;
SumMerged=0;
ChangeIndex=[];
LargeInd=find(AreaSort>min([1,AreaLimit]));
runlist=AreaInd(LargeInd);

% while there are regions smaller than threshold
while (sum(AreaSort<AreaLimit)>0)&(length(Area)>1)&(run<=length(runlist))

    % Merge smallest region with its largest neighbor,
    if length(Area)==2
        ImMerged=uint8(ImMerged>0);
    else
        ImTemp=uint8(zeros(size(ImMerged)));
        runInd=find(Label==runlist(run));
        ImTemp(LabInd(runInd))=1;
        bwp = bwpack(ImTemp);
    end
end

```



```

bwp_dilated = imdilate(bwp,ones(3,3),'ispacked');
ImTempDil = bwunpack(bwp_dilated, size(ImTemp,1));
ImTempDil = double(ImTempDil>0).*0.5;
ImTemp2=zeros(size(ImMerged));
ImTemp2(LabInd)=Label;
ImTemp2(LabInd(runInd))=0;
ImNeighbor=ImTempDil+double(ImTemp2);
Bins=1:0.5:Num+0.5;
Hist=histc(ImNeighbor(:),Bins);
NeighborInd=double(Hist(2:2:end)>0);
SmallerInd=find(((Area.*NeighborInd)<AreaLimit)&((Area.*NeighborInd)>0)>0);
ChangeList=find(ismember(Label,[SmallerInd]));
Label(ChangeList)=runlist(run);
end

% Update List
SumMerged=SumMerged+length(ChangeList);
if length(ChangeList)>0
    ChangeIndex=[ChangeIndex,runlist(run)];
end

if run==length(runlist)
    if SumMerged>0
        run=0;
        Bins=1:Num;
        runlist=ChangeIndex;
        ChangeIndex=[];
    end
    SumMerged=0;
end
run=run+1;
end

ImMerged(LabInd)=Label;

stat=regionprops(ImMerged,'Area','PixelIdxList');
Area=[stat.Area];
[AreaSort,IndSortArea]=sort(Area);
AreaSort=AreaSort(end:-1:1);
IndSortArea=IndSortArea(end:-1:1);
for j=1:length(Area>0)
    ImMerged(stat(IndSortArea(j)).PixelIdxList)=j;
end
stat=regionprops(ImMerged,'Area','PixelIdxList');

NoOfSegments=length([stat.Area]);

```

E.11 Finlayson_ill_inv.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Finlayson_ill_inv.m"
%
% Description: Function that applies Finlayson's illumination invariant
% image for detection of shadow edges in the gradient image, and use these
% to suppress shadow gradient prior to reconstruction the "shadow-free"
% image.
%
% Input: Bitmap images of frame to analyze, mask of foreground detected
% object without noise, bounding box of object.
%
% Output: Mask of detected shadow gradients and reconstructed image.

```

```

%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Finlayson_Ill_Inv(pic_in_org,BBoxAll,fg_mask)

Pad=4;
ImOrg=pic_in_org;
[Y,X,Z]=size(ImOrg);
Y_Cropped=BBoxAll(4)+1;
X_Cropped=BBoxAll(3)+1;

% Compute illumination invariant
ImGray=rgb2gray(ImOrg);
Im=double(ImOrg)+1;
ImL=log(Im);
ImLpad=zeros(Y+2*Pad,X+2*Pad,Z);
ImLpad(Pad+1:end-Pad,Pad+1:end-Pad,:)=ImL;
[YPad,XPad,Z]=size(ImLpad);

ImLm=reshape([ImLpad(:,:,1)-ImLpad(:,:,2),ImLpad(:,:,3)-ImLpad(:,:,2)],YPad,XPad,2);
gsImFull=reshape(ImLm,YPad*XPad,2)*a;
gsImFull=reshape(gsImFull,YPad,XPad);
gsIm=imcrop(gsImFull(Pad+1:end-Pad,Pad+1:end-Pad),BBoxAll);

% Detect shadow edges

EdgeMask=edge(gsIm,'canny',[0.05 0.3]);
EdgeMaskDil=imdilate(EdgeMask,ones(11));
ImEdgeNorm=zeros(Y_Cropped,X_Cropped);
ST=zeros(Y_Cropped,X_Cropped);
mask_pad=zeros(Y_Cropped,X_Cropped);
ST_All=zeros(Y_Cropped,X_Cropped);
S={};

mask_pad=fg_mask;
ST_mask=imdilate(mask_pad,ones(5))-imerode(mask_pad,ones(9));

% loop over color bands
for n=1:3
    [S{n}(:,:,1),S{n}(:,:,2)]=gradient(ImLpad(:,:,n));

    ImL_Crop=imcrop(ImL,BBoxAll);
    ImEdgeNorm(:,:,n)=edge(ImL_Crop(:,:,n),'canny',[0.01 0.1]);

    ST_All=ST_All|ImEdgeNorm(:,:,n);
end

ImNew=~EdgeMaskDil&ST_All;
ImNew_dark_mask=ImNew&dark_mask;
FinalMask=imdilate(ImNew_dark_mask,strel('disk',4));
FinalMaskEdge=imdilate((FinalMask&ST_mask),strel('disk',2));

ManualMasking=0;
if ManualMasking
    filename_manualmask=[DestFolder,TestFolder,'_ManualMaskReconstruction_',...
        num2str(ImNo),'_',num2str(ObjectNo),'.bmp'];
    imwrite(FinalMaskEdge,filename_manualmask);
    FinalMaskEdge=ST_mask;
end

```

```

FinalMaskFullPad=zeros(YPad,XPad);
FinalMaskFullPad(Pad+fix(BBoxAll(2)):Pad+fix(BBoxAll(2)+BBoxAll(4)),...
    Pad+fix(BBoxAll(1)):Pad+fix(BBoxAll(1)+BBoxAll(3)))=FinalMaskEdge;

% Reconstruct full color "shadow-free" image
for n=1:3
    S{n}=S{n}.*repmat(~FinalMaskFullPad,[1,1,2]);
    Temp=SolvePoisson(S{n});
    Image(:,:,n)=exp(Temp(Pad+1:end-Pad,Pad+1:end-Pad,:));

    Max(n)=max(max(Image(:,:,n)));
    fraction=0.02;
    TopPercentile=[];
    tic
    while (size(TopPercentile,1)/(X*Y))<0.05
        TopPercentile=find(Image(:,:,n)>(1-fraction)*Max(n));
        fraction=fraction+0.005;
    end
    toc
    Temp=reshape(Image(:,:,n),X*Y,1);
    MapImage(n,:)=0,mean(Temp(TopPercentile));
    MapImOrg(n,:)=double([min(min(ImOrg(:,:,n))),max(max(ImOrg(:,:,n)))]);

    Image(:,:,n)=uint8((Image(:,:,n)-MapImage(n,1))/(MapImage(n,2)-MapImage(n,1))*...
        (MapImOrg(n,2)-MapImOrg(n,1))+MapImOrg(n,1));
end

T=uint8(Image);
if ManualMasking
    filename_manualrec=[DestFolder,TestFolder,'_ManualReconstruction_',...
        num2str(ImNo),'_',num2str(ObjectNo),'.bmp'];
    if exist(filename_manualrec)~=2
        imwrite(T,filename_manualrec);
    end
end
T=imcrop(T,BBoxAll);

gsIm=gsIm(Pad+2:end-Pad-1,Pad+2:end-Pad-1);
EdgeMask=EdgeMask(Pad+2:end-Pad-1,Pad+2:end-Pad-1);
ImEdgeNorm=ImEdgeNorm(Pad+2:end-Pad-1,Pad+2:end-Pad-1,:);
STall=STall(Pad+2:end-Pad-1,Pad+2:end-Pad-1)>0;
[Y2,X2]=find(ST==1);

UpperTemp=[uint8(255*(gsIm+0.5)),uint8(255*double(~EdgeMask)),uint8(255*double(~EdgeMaskDil))];
Upper=[pic_in,repmat(UpperTemp,[1,1,3])];
Lower=["~ST_All","FinalMask","~ST_mask","FinalMaskEdge];

F3=figure(3)
subplot 211
imshow(Upper,[0 255])
xlabel([' (a) Original image ',...
    ' (b) Ill.-invariant of a ',...
    ' (c) Edges of b ',...
    ' (d) Dilation of c ', 'FontSize',8]);
title(['TestFolder,' - Frame No. ',num2str(ImNo),' - Object No. ',num2str(ObjectNo)]);

subplot 212
imshow(Lower,[])
xlabel([' (e) Edges of a ',...
    ' (f) Dil. of ((e-d)*fg. mask) ',...

```

```

        (g) Dil. edge of fg. mask      ',...
        (h) Dil. of (f*g)             '], 'FontSize',8);

saveas(F3,[DestFolder,TestFolder,'_FinlaysonEdge_',num2str(ImNo),'_',...
          num2str(ObjectNo),'.png'],'png');
close(F3);

```

E.12 Finlayson_FGmask.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Finlayson_FGmask.m"
%
% Description: Function that use the edges of the foreground mask to suppress
% shadow gradients prior to reconstruction the "semi-shadow-free"
% image.
%
% Input: Bitmap images of frame to analyze, mask of foreground detected
% object without noise, bounding box of object.
%
% Output: Mask of detected shadow gradients and reconstructed image.
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Finlayson_FGmask(pic_in_org,BBoxAll,fg_mask)

disp('Finlayson using foreground mask...')

global DestFolder
global TestFolder
global ImNo
global ObjectNo

filename_edge_rec=[DestFolder,TestFolder,'_EdgeReconstruction_',...
                  num2str(ImNo),'_',num2str(ObjectNo),'.bmp'];
filename_edge_mask=[DestFolder,TestFolder,'_EdgeMask_',num2str(ImNo),...
                   '_ ',num2str(ObjectNo),'.bmp'];

if (exist(filename_edge_rec)~=2)|(exist(filename_edge_mask)~=2)
    Pad=4;
    [Y,X,Z]=size(pic_in_org);
    Y_Cropped=BBoxAll(4)+1;
    X_Cropped=BBoxAll(3)+1;

    ImGray=rgb2gray(pic_in_org);
    Im=double(pic_in_org)+1;
    ImL=log(Im);
    ImLpad=zeros(Y+2*Pad,X+2*Pad,Z);
    ImLpad(Pad+1:end-Pad,Pad+1:end-Pad,:)=ImL;
    [YPad,XPad,Z]=size(ImLpad);

    mask_pad=fg_mask;
    if exist(filename_edge_mask)~=2
        FinalMaskEdge=imdilate(mask_pad,ones(5))-imerode(mask_pad,ones(9));
        imwrite(FinalMaskEdge,filename_edge_mask);
    else
        FinalMaskEdge=(imread(filename_edge_mask))>0;
    end
end
% loop over color bands

```

```

S={};
for n=1:3
    [S{n}(:, :, 1), S{n}(:, :, 2)]=gradient(ImLpad(:, :, n));
end

FinalMaskFullPad=zeros(YPad, XPad);
FinalMaskFullPad(Pad+fix(BBoxAll(2)):Pad+fix(BBoxAll(2)+BBoxAll(4)), ...
    Pad+fix(BBoxAll(1)):Pad+fix(BBoxAll(1)+BBoxAll(3)))=FinalMaskEdge;

% Reconstruct full color "semi-shadow-free" image
for n=1:3
    S{n}=S{n}.*repmat(~FinalMaskFullPad, [1, 1, 2]);
    Temp=SolvePoisson(S{n});
    Image(:, :, n)=exp(Temp(Pad+1:end-Pad, Pad+1:end-Pad, :));

    Max(n)=max(max(Image(:, :, n)));
    fraction=0.02;
    TopPercentile=[];
    while (size(TopPercentile, 1)/(X*Y)<0.05
        TopPercentile=find(Image(:, :, n)>(1-fraction)*Max(n));
        fraction=fraction+0.005;
    end
    Temp=reshape(Image(:, :, n), X*Y, 1);
    MapImage(n, :)= [0, mean(Temp(TopPercentile))];
    MapImOrg(n, :)=double([min(min(pic_in_org(:, :, n))), max(max(pic_in_org(:, :, n)))]);

    Image(:, :, n)=uint8((Image(:, :, n)-MapImage(n, 1))/(MapImage(n, 2)-MapImage(n, 1))*...
        (MapImOrg(n, 2)-MapImOrg(n, 1))+MapImOrg(n, 1));
end
T=uint8(Image);
imwrite(T, filename_edge_rec);
else
    disp('Edge mask and reconstructed image already exist...')
end
disp('Finlayson done...')

```

E.13 SolvePoisson.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "SolvePoisson.m"
%
% Description: Solves Poisson equation with Neumann boundary conditions
% using the discrete cosine transform .
%
% Input: GradImage = Gradient image Fx(:, :, 1), Fy(:, :, 2) of a grayscale image.
% Before taking the gradient, the image should be zeropadded with N>=4
% along the boundaries.
%
% Output: Image = Image retrieved by solving the Poisson equation with
% GradImage=0 on the boundary.
%
% Author: Søren Erbou (SGE)
% Last Revision: April 16, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [Image] = SolvePoisson(GradImage)

% Laplacian
[GradXX, GradXY]=gradient(GradImage(:, :, 1));
[GradYX, GradYY]=gradient(GradImage(:, :, 2));
LapImage=GradXX+GradYY;

```

```
[J,L]=size(LapImage);

% DCT
LapImageF=dct2(LapImage);
mPart= repmat(cos(pi*(0:J-1)/(J)),1,L);
nPart= repmat(cos(pi*(0:L-1)/(L)),J,1);

% Apply factor
Factor=(2*(mPart+nPart-2));
Factor(1)=-1e8;
ImageF=(LapImageF./Factor);

% Inverse DCT
Image=idct2((ImageF));
Image=real(Image);
```

E.14 EnhancedSegmentation.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "EnhancedSegmentation.m"
%
% Description: Function that applies the enhanced segmentation of
% foreground regions.
%
% Input: Bitmap images of frame to analyze, several parameters, files an
% bitmap images of merged regions and correlation vectors
%
% Output: Bitmap images of the classified regions, to be used by "Performance.m"
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function EnhancedSegmentation(pic_in,bg_in,BBoxAll,UseJavedImproved,Parameters,...
                               ParametersImproved,CorrThreshold,RBThreshold,RFThreshold)

global DestFolder
global TestFolder
global ImNo
global ObjectNo

if UseJavedImproved
    filename_JavedMerged=[DestFolder,TestFolder,'_JavedImprovedMerged_',num2str(ImNo),...
                          '_ ',num2str(ObjectNo),ParametersImproved,Parameters(1:end-8),'.mat'];
    filename_JavedCorr=[DestFolder,TestFolder,'_JavedImprovedCorr_',num2str(ImNo),'_',...
                       num2str(ObjectNo),ParametersImproved,Parameters(1:end-8),'.mat'];
else
    filename_JavedMerged=[DestFolder,TestFolder,'_JavedMerged_',num2str(ImNo),'_',...
                          num2str(ObjectNo),Parameters(1:end-8),'.mat'];
    filename_JavedCorr=[DestFolder,TestFolder,'_JavedCorr_',num2str(ImNo),'_',...
                       num2str(ObjectNo),Parameters(1:end-8),'.mat'];
end
filename_edge_rec=[DestFolder,TestFolder,'_EdgeReconstruction_',num2str(ImNo),'_',...
                  num2str(ObjectNo),'.bmp'];
filename_edge_mask=[DestFolder,TestFolder,'_EdgeMask_',num2str(ImNo),'_',num2str(ObjectNo),'.bmp'];

ParametersEnhanced=['_RB',sprintf('% .1f',RBThreshold),'_'];

if (exist(filename_edge_rec)==2)&(exist(filename_edge_mask)==2)
    if exist(filename_JavedMerged)==2
```

```

if exist(filename_JavedCorr)==2
    load(filename_JavedMerged);
    load(filename_JavedCorr);
    pic_rec_org=imread(filename_edge_rec);
    pic_rec=imcrop(pic_rec_org,BBoxAll);
    EdgeMask=double(imread(filename_edge_mask));
    pic_fg_class_merged_masked=pic_fg_class_merged.*~EdgeMask;

    NoOfRegions=max(pic_fg_class_merged(:));
    [Y,X,Z]=size(pic_rec);
    castshadowEnhanced=[];
    object_classEnhanced=[];
    pic_fg_class_Enhanced=uint8(zeros(Y,X));
    VarRecMaskFG=zeros(NoOfRegions,1);
    VarRecMaskBG=zeros(NoOfRegions,1);
    temp=-1*ones(NoOfRegions,1);
    VarCross=255;
    VarFG=12;
    for q=1:NoOfRegions
        ind_class{q}=find(pic_fg_class_merged==q);
        ind_class_no_edge{q}=find(pic_fg_class_merged_masked==q);
        ImTempFG=[];
        ImTempRec=[];
        ImTempBG=[];
        for w=1:Z
            ImTemp=reshape(double(pic_in(:,:,w)),Y*X,1);
            ImTempFG=[ImTempFG;ImTemp(ind_class_no_edge{q})];
            ImTemp=reshape(double(pic_rec(:,:,w)),Y*X,1);
            ImTempRec=[ImTempRec;ImTemp(ind_class_no_edge{q})];
            ImTemp=reshape(double(bg_in(:,:,w)),Y*X,1);
            ImTempBG=[ImTempBG;ImTemp(ind_class_no_edge{q})];
        end
        if length(ind_class_no_edge{q})>10
            VarRecMaskFG(q)=var(ImTempRec-ImTempFG)./VarFG;
            VarRecMaskBG(q)=var(ImTempRec-ImTempBG)./VarCross;
        end

        if (VarRecMaskBG(q)==0)|(Corr(q)>CorrThreshold)|(Corr(q)<(0.5*CorrThreshold))
            if (Corr(q)>0.75*CorrThreshold)
                castshadowEnhanced(end+1,1)=q;
                temp(q)=0;
                pic_fg_class_Enhanced(ind_class{q})=uint8(128);
            else
                object_classEnhanced(end+1,1)=q;
                temp(q)=1;
                pic_fg_class_Enhanced(ind_class{q})=uint8(255);
            end
        else
            if (VarRecMaskBG(q)<RBThreshold)
                castshadowEnhanced(end+1,1)=q;
                temp(q)=-2;
                pic_fg_class_Enhanced(ind_class{q})=uint8(128);
            else
                object_classEnhanced(end+1,1)=q;
                temp(q)=2;
                pic_fg_class_Enhanced(ind_class{q})=uint8(255);
            end
        end
    end
end
if UseJavedImproved
    imwrite(pic_fg_class_Enhanced,[DestFolder,TestFolder,'_EnhancedImprovedLabel_',...

```

```

        num2str(ImNo), '_ ', num2str(ObjectNo), ParametersImproved, Parameters, ...
        ParametersEnhanced, '.bmp'];
    else
        imwrite(pic_fg_class_Enhanced, [DestFolder, TestFolder, '_EnhancedLabel_', ...
            num2str(ImNo), '_ ', num2str(ObjectNo), Parameters, ParametersEnhanced, '.bmp']);
    end

    if UseJavedImproved
        disp('Improved classification reconstructed image and javed improved done...')
    else
        disp('Improved classification reconstructed image done...')
    end
else
    disp({'Improved classification using reconstructed image not done...'; ...
        filename_JavedCorr; '...does not exist'})
end
else
    disp({'Improved classification using reconstructed image not done...'; ...
        filename_JavedMerged; '...does not exist'})
end
else
    disp({'Improved classification using reconstructed image not done...'; ...
        filename_edge_rec, ' and/or ', filename_edge_mask; '...does not exist'})
end
end

```

E.15 DetectVariance.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "DetectVariance.m"
%
% Description: Script that computes the variance normalization factor of
% the enhanced similarity feature, (CS), using sequence Test77.
%
% Input: Images of sequence Test77
%
% Output: Variance
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
close all;
clc;

TestNo=77;
ImNo=49;

TestFolder=['Test', num2str(TestNo)];

addpath(['E:\SGE\Video\' , TestFolder]);
addpath(['E:\SGE\Video\' , TestFolder, '_Object']);
DestFolder=['SGE\Video\TrainingSet'];
Drive = 'E:\';
if exist([Drive, DestFolder], 'dir')~=7
    Dir=pwd;
    cd(Dir);
    mkdir(DestFolder);
    cd(Dir);
end
DestFolder=[Drive, DestFolder, '\'];

```



```

addpath(DestFolder);

filename=[TestFolder,'_',num2str(ImNo),'.bmp'];
bg_filename=[TestFolder,'_middel_',num2str(ImNo),'_bg_rgb_SGE.bmp'];
Back1 = imread(bg_filename);
Back2 = imread(filename);

filename_manualrec_bg=[DestFolder,TestFolder,'_ReconstructionNoMask_bg',num2str(ImNo),'.bmp'];
if exist(filename_manualrec_bg)==2
    Rec1=double(imread(filename_manualrec_bg));
else
    Rec1=Finlay(Back1,'bg',DestFolder,TestFolder,ImNo);
end
filename_manualrec_new=[DestFolder,TestFolder,'_ReconstructionNoMask_new',num2str(ImNo),'.bmp'];
if exist(filename_manualrec_new)==2
    Rec2=double(imread(filename_manualrec_new));
else
    Rec2=Finlay(Back2,'new',DestFolder,TestFolder,ImNo);
end

Dist=5;
Back1=Back1(Dist+1:end-Dist,Dist+1:end-Dist,:);
Back2=Back2(Dist+1:end-Dist,Dist+1:end-Dist,:);
Rec1=Rec1(Dist+1:end-Dist,Dist+1:end-Dist,:);
Rec2=Rec2(Dist+1:end-Dist,Dist+1:end-Dist,:);

[Y,X,Z]=size(Back1);
MeanRecMaskOrg=zeros(X*Y*Z,1);
MeanRecMaskBack=zeros(X*Y*Z,1);
VarRecMaskOrg=zeros(X*Y*Z,1);
VarRecMaskBack=zeros(X*Y*Z,1);

ImTempBack1=[];
ImTempBack2=[];
ImTempRec1=[];
ImTempRec2=[];
for w=1:Z
    ImTemp=reshape(double(Back1(:,:,w)),Y*X,1);
    ImTempBack1=[ImTempBack1;ImTemp];
    ImTemp=reshape(double(Back2(:,:,w)),Y*X,1);
    ImTempBack2=[ImTempBack2;ImTemp];
    ImTemp=reshape(double(Rec1(:,:,w)),Y*X,1);
    ImTempRec1=[ImTempRec1;ImTemp];
    ImTemp=reshape(double(Rec2(:,:,w)),Y*X,1);
    ImTempRec2=[ImTempRec2;ImTemp];
end

Var(1)=sum((ImTempBack1-ImTempRec2).^2)/(X*Y-1);
Var(2)=sum((ImTempBack1-ImTempBack2).^2)/(X*Y-1);
Var(3)=sum((ImTempRec1-ImTempRec2).^2)/(X*Y-1);

function Output=Finlay(pic_in_org,Text,DestFolder,TestFolder,ImNo)

Pad=4;

ImOrg=pic_in_org;
[Y,X,Z]=size(ImOrg);

ImGray=rgb2gray(ImOrg);
Im=double(ImGray)+1;
ImL=log(Im);

```

```

ImLpad=zeros(Y+2*Pad,X+2*Pad,Z);
ImLpad(Pad+1:end-Pad,Pad+1:end-Pad,:)=ImL;
[YPad,XPad,Z]=size(ImLpad);
for n=1:3
    [S{n}(:, :, 1),S{n}(:, :, 2)]=gradient(ImLpad(:, :, n));
    Temp=SolvePoisson(S{n});
    Image(:, :, n)=exp(Temp(Pad+1:end-Pad,Pad+1:end-Pad,:));

    Max(n)=max(max(Image(:, :, n)));
    fraction=0.02;
    TopPercentile=[];
    while (size(TopPercentile,1)/(X*Y))<0.05
        TopPercentile=find(Image(:, :, n)>(1-fraction)*Max(n));
        fraction=fraction+0.005;
    end
    Temp=reshape(Image(:, :, n),X*Y,1);
    MapImage(n,:)= [0,mean(Temp(TopPercentile))];
    MapImOrg(n,:)=double([min(min(ImOrg(:, :, n))),max(max(ImOrg(:, :, n)))]);

    Image(:, :, n)=uint8((Image(:, :, n)-MapImage(n,1))/(MapImage(n,2)-MapImage(n,1))*...
        (MapImOrg(n,2)-MapImOrg(n,1))+MapImOrg(n,1));
end

Output=uint8(Image);

filename_manualrec=[DestFolder,TestFolder,'_ReconstructionNoMask_',Text,num2str(ImNo),'.bmp'];
if exist(filename_manualrec)~=2
    imwrite(Output,filename_manualrec);
end

```

E.16 Performance.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Performance.m"
%
% Description: Script that computes the performance of different methods
% for shadow removal.
%
% Input: Manually labelled images and images classified by the method
% chosen.
%
% Output: "png"-figures of performance
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

filename_label=[DestFolder,TestFolder,'_TrueLabel_',num2str(ImNo),'_',num2str(ObjectNo),'.bmp'];
if UseJavedImproved
    filename_Javed=[DestFolder,TestFolder,'_JavedImprovedLabel_',num2str(ImNo),'_',...
        num2str(ObjectNo),ParametersImproved,Parameters,'.bmp'];
    filename_Enhanced=[DestFolder,TestFolder,'_EnhancedImprovedLabel_',num2str(ImNo),'_',...
        num2str(ObjectNo),ParametersImproved,Parameters,ParametersEnhanced,'.bmp'];
else
    filename_Javed=[DestFolder,TestFolder,'_JavedLabel_',num2str(ImNo),'_',...
        num2str(ObjectNo),Parameters,'.bmp'];
    filename_Enhanced=[DestFolder,TestFolder,'_EnhancedLabel_',num2str(ImNo),'_',...
        num2str(ObjectNo),Parameters,ParametersEnhanced,'.bmp'];
end

```

```

if exist(filename_label)==2
    ImTrue=imread(filename_label);
    [Y,X,Z]=size(ImTrue);
    if Z==3
        ImTrue=dark_mask.*double(rgb2gray(ImTrue));
    else
        ImTrue=dark_mask.*double(ImTrue);
    end
    AllDarkPixelIdxList=find(ImTrue>0);
    Hist=histc(ImTrue(:),[1:max(ImTrue(:))]);
    Vals=find(Hist>0);
    Max=max(ImTrue(:));

    if Vals(1)<150
        TrueShadow=(ImTrue(AllDarkPixelIdxList)==Vals(1));
    else
        TrueShadow=zeros(size(AllDarkPixelIdxList));
    end
    if length(Vals)>1
        if (Vals(2)>150)&(Vals(2)<225)
            TrueSelfShadow=(ImTrue(AllDarkPixelIdxList)==Vals(2));
        else
            TrueSelfShadow=zeros(size(AllDarkPixelIdxList));
        end
    else
        TrueSelfShadow=zeros(size(AllDarkPixelIdxList));
    end
    if length(Vals)>2
        if Vals(3)>225
            TrueDarkObj=(ImTrue(AllDarkPixelIdxList)==Vals(3));
        else
            TrueDarkObj=zeros(size(AllDarkPixelIdxList));
        end
    else
        TrueDarkObj=zeros(size(AllDarkPixelIdxList));
    end

    TrueObj=(TrueSelfShadow|TrueDarkObj);
    ImTrue(AllDarkPixelIdxList(find(TrueShadow)))=0.3;
    ImTrue(AllDarkPixelIdxList(find(TrueSelfShadow)))=0.6;
    ImTrue(AllDarkPixelIdxList(find(TrueDarkObj)))=1;
    imwrite(ImTrue,filename_label);

    NoOfPixels=sum(TrueObj)+sum(TrueShadow);
    SelfShadowSize=sum(TrueSelfShadow);
    disp([' '];' ']);
    disp(['TestFolder,' - Frame ',num2str(ImNo),' - Object ',...
        num2str(ObjectNo),' manually labelled...']);
    disp([num2str(sum(TrueObj)),' = ',num2str(100*sum(TrueObj)/NoOfPixels,3),...
        '% True object pixels +']);
    disp([num2str(sum(TrueShadow)),' = ',num2str(100*sum(TrueShadow)/NoOfPixels,3),...
        '% True cast shadow pixels =']);
    disp([num2str(NoOfPixels),' Pixels classified']);
    disp(' ');
    disp([num2str(SelfShadowSize),' of ',num2str(sum(TrueObj)),' = ',...
        num2str(100*SelfShadowSize/sum(TrueObj),3),'% Self shadow pixels in object']);

    switch DoPerformance
        case 3

```

```

Im=[pic_in,dark_fg_white,255*repmat(ImTrue,[1,1,3]),255*ones(size(pic_in))];
F5=figure(5);
imshow(Im,[0 255]);
Text1={'TestFolder,' - Frame no. ',num2str(ImNo),' - Object no. ',num2str(ObjectNo)};
Text2={'No. of pixels to classify:':[num2str(NoOfPixels),' = ',...
    num2str(100*NoOfPixels/sum(fg_mask(:)),3),'% of all foreground pixels'];...
    ['No. of true object pixels:':[num2str(sum(TrueObj)), ' = ',...
    num2str(100*sum(TrueObj)/NoOfPixels,3),'%'];...
    ['No. of true cast shadow pixels:':[num2str(sum(TrueShadow)),...
    ' = ',num2str(100*sum(TrueShadow)/NoOfPixels,3),'%'];...
    ['No. of object pixels in self shadow:':[num2str(SelfShadowSize),...
    ' = ',num2str(100*SelfShadowSize/sum(TrueObj),3),'%']];

T1=text(3.05*X,0,[Text1;Text2],'VerticalAlignment','top','FontSize',7);
CropPNG(F5,[DestFolder,TestFolder,'_ImageStat_',num2str(ImNo),'_',...
    num2str(ObjectNo),'.png'],'_Cropped');

case 4
Im=[pic_in,dark_fg_white];
if count==1
    F6=figure(6);
else
    figure(F6);
end
if count<4
    subplot(3,3,count);
else
    subplot(3,2,count-1);
end
imshow(Im,[0 255]);
xlabel(['TestFolder,' - Frame ',num2str(ImNo),' - Object ',num2str(ObjectNo)],...
    'VerticalAlignment','bottom','FontSize',8);
if count==5
    saveas(F6,[DestFolder,'TrainingImages.png'],'png');
end

case 1
if exist(filename_Javed)==2
    ImJaved=imread(filename_Javed);
    ImJaved=dark_mask.*double(ImJaved);
    Max=max(ImJaved(:));
    PredObj=(ImJaved(AllDarkPixelIdxList)==Max);
    PredShadow=(ImJaved(AllDarkPixelIdxList)<Max);

    TNJavedPixels=TrueShadow&PredShadow;
    FPJavedPixels=TrueShadow&PredObj;
    FNJavedPixels=TrueObj&PredShadow;
    TPJavedPixels=TrueObj&PredObj;

    ConfMatrixJaved=[sum(TNJavedPixels), sum(FPJavedPixels);...
        sum(FNJavedPixels), sum(TPJavedPixels)];
    % row=true, col=predicted, 1=shadow, 2=object => [a,b;c,d]
    %a/(a+b) = True Negatives
    %b/(a+b) = False Positives
    %c/(c+d) = False Negatives
    %d/(c+d) = True Positives
    %(a+d)/(a+b+c+d) = Accuracy
    TNJaved=round(ConfMatrixJaved(1,1)/sum(ConfMatrixJaved(1,:))*1000)/10;
    FPJaved=round(ConfMatrixJaved(1,2)/sum(ConfMatrixJaved(1,:))*1000)/10;
    FNJaved=round(ConfMatrixJaved(2,1)/sum(ConfMatrixJaved(2,:))*1000)/10;
    TPJaved=round(ConfMatrixJaved(2,2)/sum(ConfMatrixJaved(2,:))*1000)/10;
    ACJaved=round((ConfMatrixJaved(1,1)+ConfMatrixJaved(2,2))/...

```

```

        sum(ConfMatrixJaved(:))*1000)/10;
ConfMatrixJavedPerc=[TNJaved,FPJaved;FNJaved,TPJaved];
if UseJavedImproved
    save([DestFolder,TestFolder,'_JavedImprovedClass_',num2str(ImNo),'_',...
        num2str(ObjectNo),ParametersImproved,Parameters,'.mat'],...
        'TrueObj','TrueShadow','TrueSelfShadow','PredObj','PredShadow',...
        'TNJavedPixels','FNJavedPixels','FPJavedPixels','TPJavedPixels',...
        'ConfMatrixJaved','ConfMatrixJavedPerc','ACJaved');
    disp(' ');
    disp('Javed Improved Performance...');
else
    save([DestFolder,TestFolder,'_JavedClass_',num2str(ImNo),'_',...
        num2str(ObjectNo),Parameters,'.mat'],...
        'TrueObj','TrueShadow','TrueSelfShadow','PredObj','PredShadow',...
        'TNJavedPixels','FNJavedPixels','FPJavedPixels','TPJavedPixels',...
        'ConfMatrixJaved','ConfMatrixJavedPerc','ACJaved');
    disp(' ');
    disp('Javed Performance...');
end

disp('Confusion Matrix:');
disp(ConfMatrixJaved);
disp('Confusion Matrix [%]:');
disp(ConfMatrixJavedPerc);
disp(['Accuracy ( (a+d)/(a+b+c+d) ) = ',num2str(ACJaved,3),' %']);
disp(' ');

ClassPlotInd={};
for j=1:4
    ImTemp=uint8(zeros(Y,X));
    switch j
        case 1
            ImTemp(AllDarkPixelIdxList(find(TNJavedPixels)))=1;
        case 2
            ImTemp(AllDarkPixelIdxList(find(FPJavedPixels)))=1;
        case 3
            ImTemp(AllDarkPixelIdxList(find(FNJavedPixels)))=1;
        case 4
            ImTemp(AllDarkPixelIdxList(find(TPJavedPixels)))=1;
    end
    [ClassPlotInd.R{j},ClassPlotInd.C{j}]=find(ImTemp);
end
F7=figure(7);
set(F7,'name',[TestFolder,' - Frame No. ',num2str(ImNo),' - Object No. ',...
    num2str(ObjectNo)]);
subplot(131)
imshow(pic_in,[0 255]);
title([TestFolder,' - Frame No. ',num2str(ImNo),' - Object No. ',num2str(ObjectNo)]);
subplot(132)
imshow(dark_fg_white,[0 255]);
subplot(133)
imshow(dark_fg_white,[0 255]);
hold on
Colors=[{'b.'},{ 'y.'},{ 'r.'},{ 'g.'},{ 'm.'}];
hClass=-1*ones(4,1);
for j=1:4
    if length(ClassPlotInd.C{j})>0
        hClass(j)=plot(ClassPlotInd.C{j},ClassPlotInd.R{j},Colors{j});
    end
end
end
LegendText=[ 'True Shadow (TN) = ',num2str(TNJaved),'%', 'False Object (FP) = ',...

```

```

        num2str(FPJaved),'%',[ 'False Shadow (FN) = ',num2str(FNJaved),'%'],...
        [ 'True Object (TP) = ',num2str(TPJaved),'%']];
ValidLegends=find(hClass>0);
hL=legend(hClass(ValidLegends),LegendText(ValidLegends));
set(hL,'Position',[0.69 0.03 0.2 0.1]);
hold off
if UseJavedImproved
    title(['Perf. using Javeds Improved method: ',num2str(ACJaved),'% acc.']);
    saveas(F7,[DestFolder,TestFolder,'_JavedImprovedPerformance_',num2str(ImNo),...
        '_ ',num2str(ObjectNo),'.png'],'png');
    saveas(F7,[DestFolder,'Performance\ ',TestFolder,'_JavedImprovedPerformance_',...
        num2str(ImNo),'_',num2str(ObjectNo),ParametersImproved,Parameters,'.png'],'png');
else
    title(['Performance using Javeds method: ',num2str(ACJaved),'% acc.']);
    saveas(F7,[DestFolder,TestFolder,'_JavedPerformance_',num2str(ImNo),'_',...
        num2str(ObjectNo),'.png'],'png');
    saveas(F7,[DestFolder,'Performance\ ',TestFolder,'_JavedPerformance_',...
        num2str(ImNo),'_',num2str(ObjectNo),Parameters,'.png'],'png');
end
end
else
    if UseJavedImproved
        disp(['TestFolder, ' - Frame ',num2str(ImNo),' - Object ',num2str(ObjectNo),...
            ' Javed Improved not done...']);
    else
        disp(['TestFolder, ' - Frame ',num2str(ImNo),' - Object ',num2str(ObjectNo),...
            ' Javed not done...']);
    end
end
end
case 2
if exist(filename_Enhanced)==2
    ImEnhanced=imread(filename_Enhanced);
    ImEnhanced=dark_mask.*double(ImEnhanced);
    Max=max(ImEnhanced(:));
    PredObj=(ImEnhanced(AllDarkPixelIdxList)==Max);
    PredShadow=(ImEnhanced(AllDarkPixelIdxList)<Max);

    TNEnhancedPixels=TrueShadow&PredShadow;
    FPEnhancedPixels=TrueShadow&PredObj;
    FNEnhancedPixels=TrueObj&PredShadow;
    TPEnhancedPixels=TrueObj&PredObj;

    ConfMatrixEnhanced=[sum(TNEnhancedPixels), sum(FPEnhancedPixels);...
        sum(FNEnhancedPixels), sum(TPEnhancedPixels)];
    % row=true, col=predicted, 1=shadow, 2=object => [a,b;c,d]
    %a/(a+b) = True Negatives
    %b/(a+b) = False Positives
    %c/(c+d) = False Negatives
    %d/(c+d) = True Positives
    %(a+d)/(a+b+c+d) = Accuracy
    TNEnhanced=round(ConfMatrixEnhanced(1,1)/sum(ConfMatrixEnhanced(1,:))*1000)/10;
    FPEnhanced=round(ConfMatrixEnhanced(1,2)/sum(ConfMatrixEnhanced(1,:))*1000)/10;
    FNEnhanced=round(ConfMatrixEnhanced(2,1)/sum(ConfMatrixEnhanced(2,:))*1000)/10;
    TPEnhanced=round(ConfMatrixEnhanced(2,2)/sum(ConfMatrixEnhanced(2,:))*1000)/10;
    ACEnhanced=round((ConfMatrixEnhanced(1,1)+ConfMatrixEnhanced(2,2))/...
        sum(ConfMatrixEnhanced(:))*1000)/10;
    ConfMatrixEnhancedPerc=[TNEnhanced,FPEnhanced;FNEnhanced,TPEnhanced];
    if UseJavedImproved
        save([DestFolder,TestFolder,'_EnhancedImprovedClass_',num2str(ImNo),'_',...
            num2str(ObjectNo),ParametersImproved,Parameters,ParametersEnhanced,...
            '.mat'],'TrueObj','TrueShadow','TrueSelfShadow','PredObj','PredShadow',...
            'TNEnhancedPixels','FNEnhancedPixels','FPEnhancedPixels',...

```

```

        'TPEnhancedPixels', 'ConfMatrixEnhanced', 'ConfMatrixEnhancedPerc', ...
        'ACEnhanced');
disp(' ');
disp('Enhanced Improved Performance...');
else
save([DestFolder, TestFolder, '_EnhancedClass_', num2str(ImNo), '_', ...
     num2str(ObjectNo), Parameters, ParametersEnhanced, '.mat'], ...
     'TrueObj', 'TrueShadow', 'TrueSelfShadow', 'PredObj', 'PredShadow', ...
     'TNEnhancedPixels', 'FNEnhancedPixels', 'FPEnhancedPixels', ...
     'TPEnhancedPixels', 'ConfMatrixEnhanced', 'ConfMatrixEnhancedPerc', ...
     'ACEnhanced');
disp(' ');
disp('Enhanced Performance...');
end

disp('Confusion Matrix:');
disp(ConfMatrixEnhanced);
disp('Confusion Matrix [%]:');
disp(ConfMatrixEnhancedPerc);
disp(['Accuracy ( (a+d)/(a+b+c+d) ) = ', num2str(ACEnhanced,3), '%']);
disp(['Precision ( d/(b+d) ) = ', num2str(PEnhanced,3), '%']);
disp(' ');

ClassPlotInd={};
for j=1:4
    ImTemp=uint8(zeros(Y,X));
    switch j
        case 1
            ImTemp(AllDarkPixelIdxList(find(TNEnhancedPixels)))=1;
        case 2
            ImTemp(AllDarkPixelIdxList(find(FPEnhancedPixels)))=1;
        case 3
            ImTemp(AllDarkPixelIdxList(find(FNEnhancedPixels)))=1;
        case 4
            ImTemp(AllDarkPixelIdxList(find(TPEnhancedPixels)))=1;
    end
    [ClassPlotInd.R{j}, ClassPlotInd.C{j}]=find(ImTemp);
end
F8=figure(8);
set(F8, 'name', [TestFolder, ' - Frame No. ', num2str(ImNo), ' - Object No. ', ...
                num2str(ObjectNo)]);
subplot 131
imshow(pic_in, [0 255]);
title([TestFolder, ' - Frame No. ', num2str(ImNo), ' - Object No. ', num2str(ObjectNo)]);
subplot 132
imshow(dark_fg_white, [0 255]);
subplot 133
imshow(dark_fg_white, [0 255]);
hold on
Colors=[{'b.'}, {'y.'}, {'r.'}, {'g.'}, {'m.'}];
hClass=-1*ones(4,1);
for j=1:4
    if length(ClassPlotInd.C{j})>0
        hClass(j)=plot(ClassPlotInd.C{j}, ClassPlotInd.R{j}, Colors{j});
    end
end
LegendText=[['True Shadow (TN) = ', num2str(TNEnhanced), '%'], ...
            ['False Object (FP) = ', num2str(FPEnhanced), '%'], ...
            ['False Shadow (FN) = ', num2str(FNEnhanced), '%'], ...
            ['True Object (TP) = ', num2str(TPEnhanced), '%']];
ValidLegends=find(hClass>0);

```

```

hL=legend(hClass(ValidLegends),LegendText(ValidLegends));
set(hL,'Position',[0.69 0.03 0.2 0.1]);
hold off
if UseJavedImproved
    title(['Perf. - Enhanced Improved method: ',num2str(ACEnhanced),'% acc.']);
    saveas(F8,[DestFolder,TestFolder,'_EnhancedImprovedPerformance_',num2str(ImNo),...
        '_ ',num2str(ObjectNo),'.png'],'png');
    saveas(F8,[DestFolder,'Performance\ ',TestFolder,'_EnhancedImprovedPerformance_',...
        num2str(ImNo),'_',num2str(ObjectNo),ParametersImproved,Parameters,...
        ParametersEnhanced,'.png'],'png');
else
    title(['Perf. - Enhanced. method: ',num2str(ACEnhanced),'% acc.']);
    saveas(F8,[DestFolder,TestFolder,'_EnhancedPerformance_',num2str(ImNo),'_',...
        num2str(ObjectNo),'.png'],'png');
    saveas(F8,[DestFolder,'Performance\ ',TestFolder,'_EnhancedPerformance_',...
        num2str(ImNo),'_',num2str(ObjectNo),Parameters,ParametersEnhanced,...
        '.png'],'png');
end
end
else
    if UseJavedImproved
        disp(['TestFolder,' - Frame ',num2str(ImNo),' - Object ',num2str(ObjectNo),...
            ' Enhanced Improved not done...']);
    else
        disp(['TestFolder,' - Frame ',num2str(ImNo),' - Object ',num2str(ObjectNo),...
            ' Enhanced not done...']);
    end
end
end
end
else
    disp(['TestFolder,' - Frame ',num2str(ImNo),' - Object ',num2str(ObjectNo),' not labelled...']);
end
end

```

E.17 Compare.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Compare.m"
%
% Description: Script that collects performance results from different
% methods in a single file for comparison.
%
% Input: "mat"-files containing results for each method
%
% Output: A single "mat"-file containing all results
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global DestFolder
global TestFolder
global ImNo
global ObjectNo
global Perf;

Var=81;
MergingSizeLimit=[100,10]; %[100,10]
CorrThreshold=[0.05,0.1]; %[0.05,0.1]
VarOffset=4;
RBThreshold=3; %3

Parameters1=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit(1)),'_Corr',...

```



```

%
% Description: Script used for analyzing the results from the test set.
%
% Input: File containg collected results.
%
% Output: Figures showing performance.
%
% Remarks: Applies paired t-tests to show significant differences in mean
% values of performance methods.
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;
clear all;
clc;

DestFolder=['SGE\Video\FilesTestSet'];
Drive = 'E:\';
DestFolder=[Drive, DestFolder, '\'];
load([DestFolder, 'Performance\ComparisonTestSet_RB3.0_.mat']);
DATASET=DataSets(1);

% Mean and Std. of performance measures
Means=round([Mean(Perf.ac,1);Mean(Perf.tp,1);...
            Mean(Perf.fp,1);Mean(Perf.tn,1);Mean(Perf.fn,1)]*10)/10;
Stds=round([std(Perf.ac,1);std(Perf.tp,1);std(Perf.fp,1);...
            std(Perf.tn,1);std(Perf.fn,1)]*10)/10

% Binomial comparison
Comp.ac=[Perf.ac(:,2)>Perf.ac(:,1),Perf.ac(:,4)>Perf.ac(:,2),Perf.ac(:,4)>Perf.ac(:,1)];
Binom(:,1)=sum(Comp.ac)';
Comp.tp=[Perf.tp(:,2)>Perf.tp(:,1),Perf.tp(:,4)>Perf.tp(:,2),Perf.tp(:,4)>Perf.tp(:,1)];
Binom(:,2)=sum(Comp.tp)';
Comp.tn=[Perf.tn(:,2)>Perf.tn(:,1),Perf.tn(:,4)>Perf.tn(:,2),Perf.tn(:,4)>Perf.tn(:,1)];
Binom(:,3)=sum(Comp.tn)';

% Testing for differences in absolute measures
J=[Perf.ac(:, [1]),Perf.tp(:, [1]),Perf.fp(:, [1]),Perf.tn(:, [1]),Perf.fn(:, [1])];
I=[Perf.ac(:, [2]),Perf.tp(:, [2]),Perf.fp(:, [2]),Perf.tn(:, [2]),Perf.fn(:, [2])];
E=[Perf.ac(:, [4]),Perf.tp(:, [4]),Perf.fp(:, [4]),Perf.tn(:, [4]),Perf.fn(:, [4])];

JI=I-J;
IE=E-I;
JE=E-J;

h=[];
p=[];
ci=[];

H=-1*ones(3,3);
P=-1*ones(3,3);
for j=[1,2,4]
    [h(1,j),p(1,j),dum1(1,1:2),stats(1,j)] = ttest(I(:,j),J(:,j),0.05,'right');
    [h(2,j),p(2,j),dum1(2,1:2),stats(2,j)] = ttest(E(:,j),I(:,j),0.05,'right');
    [h(3,j),p(3,j),dum1(3,1:2),stats(3,j)] = ttest(E(:,j),J(:,j),0.05,'right');
    ci(:,j)=dum1(:,1);
    [H(1,j),P(1,j)] = jbtest(JI(:,j),0.05);
    [H(2,j),P(2,j)] = jbtest(IE(:,j),0.05);
    [H(3,j),P(3,j)] = jbtest(JE(:,j),0.05);
end

```

```

end

% Testing for differences in relative measures

NotZeroIndJ{1}=find(J(:,1)>0);
NotZeroIndJ{2}=find(J(:,2)>0);
NotZeroIndJ{3}=find(J(:,3)>0);
NotZeroIndJ{4}=find(J(:,4)>0);
NotZeroIndJ{5}=find(J(:,5)>0);

Jlog=log([Perf.ac(:, [1]),Perf.tp(:, [1]),Perf.fp(:, [1]),Perf.tn(:, [1]),Perf.fn(:, [1])]);
Ilog=log([Perf.ac(:, [2]),Perf.tp(:, [2]),Perf.fp(:, [2]),Perf.tn(:, [2]),Perf.fn(:, [2])]);
Elog=log([Perf.ac(:, [4]),Perf.tp(:, [4]),Perf.fp(:, [4]),Perf.tn(:, [4]),Perf.fn(:, [4])]);

JIlog=Ilog-Jlog;
IElog=Elog-Ilog;
JElog=Elog-Jlog;

hlog=[];
plog=[];
cilog=[];

Hlog=-1*ones(3,3);
Plog=-1*ones(3,3);
for j=[1,2,4]
    [hlog(1,j),plog(1,j),dum1(1,1:2),statslog(1,j)]=...
        ttest(Ilog(NotZeroIndJ{4},j),Jlog(NotZeroIndJ{4},j),0.05,'right');
    [hlog(2,j),plog(2,j),dum1(2,1:2),statslog(2,j)]=...
        ttest(Elog(:,j),Ilog(:,j),0.05,'right');
    [hlog(3,j),plog(3,j),dum1(3,1:2),statslog(3,j)]=...
        ttest(Elog(NotZeroIndJ{4},j),Jlog(NotZeroIndJ{4},j),0.05,'right');
    cilog(:,j)=dum1(:,1);
    [Hlog(1,j),Plog(1,j)] = jbtest(JIlog(NotZeroIndJ{4},j),0.05);
    [Hlog(2,j),Plog(2,j)] = jbtest(IElog(:,j),0.05);
    [Hlog(3,j),Plog(3,j)] = jbtest(JElog(NotZeroIndJ{4},j),0.05);
end

JIrel=I./J;
IERel=E./I;
JERel=E./J;

ObjPart=Perf.TrueObj(:,1)./(Perf.TrueShadow(:,1)+Perf.TrueObj(:,1));

% Plot figures
F3=figure(3);
subplot 231
plot(I(:,1),J(:,1),'g');
hold on
l=line([0 100],[0 100]);
p=plot(Means(2,1),Means(1,1),'kx','linewidth',1,'markersize',6);
set(l,'color','k');
leg(1)=legend(p,'Mean value',2);
axis([0 100 0 100]);
hold off
grid on
xlabel({'AC [%] of I','(a)'},'FontSize',8);
ylabel('AC [%] of J','FontSize',8);

subplot 232
plot(E(:,1),J(:,1),'r');
hold on

```

```

l=line([0 100],[0 100]);
p=plot(Means(4,1),Means(1,1),'kx','linewidth',1,'markersize',6);
set(l,'color','k');
leg(2)=legend(p,'Mean value',2);
axis([0 100 0 100])
hold off
grid on
xlabel({'AC [%] of E','(b)'},'FontSize',8);
ylabel('AC [%] of J','FontSize',8);

subplot 233
ph(1)=plot(Means(1,3),Means(1,2),'bx','linewidth',1,'markersize',6);
hold on
ph(2)=plot(Means(2,3),Means(2,2),'gx','linewidth',1,'markersize',6);
ph(3)=plot(Means(4,3),Means(4,2),'rx','linewidth',1,'markersize',6);
ph(4)=plot(28,80,'bo','linewidth',1,'markersize',6);
ph(5)=plot(21,72,'go','linewidth',1,'markersize',6);
ph(6)=plot(26,83,'ro','linewidth',1,'markersize',6);
axis([20 50 60 90]);
leg(3)=legend(ph,'(J) Test',...
              '(I) Test',...
              '(E) Test',...
              '(J) Train.',...
              '(I) Train.',...
              '(E) Train.',1);

xlabel({'FP [%]','(c)'},'FontSize',8);
ylabel('TP [%]','FontSize',8);
hold off
grid on

Edges=0:5:100;
HistJ=histc(J(:,1),Edges);
HistE=histc(E(:,1),Edges);

resolution=100;
NormalJ(:,1)=[Means(1,1)+4*Stds(1,1)*(-1+1/(2*resolution):2/resolution:1-1/(2*resolution))];
NormalJ(:,2)=[1./(Stds(1,1)*sqrt(2*pi))*exp(-(NormalJ(:,1)-Means(1,1)).^2/(2*Stds(1,1)^2))];
NormalE(:,1)=[Means(4,1)+4*Stds(1,1)*(-1+1/(2*resolution):2/resolution:1-1/(2*resolution))];
NormalE(:,2)=[1./(Stds(4,1)*sqrt(2*pi))*exp(-(NormalE(:,1)-Means(4,1)).^2/(2*Stds(4,1)^2))];

subplot 212
bh(1)=bar(Edges+2.5,HistJ,0.8,'b');
set(gca,'YTickLabel',[]);
hold on
bh(2)=bar(Edges+2.5,HistE,0.5,'r');
bh(3)=plot(NormalJ(:,1),72*5*NormalJ(:,2),'k-', 'linewidth',2);
bh(4)=plot(NormalE(:,1),72*5*NormalE(:,2),'k-', 'linewidth',2);
hold off
axis([0 100 0 1.1*max([HistJ;HistE])]);
xlabel({'AC [%]','(d)'},'FontSize',8)
ylabel('Occurrence/probability','FontSize',8)
leg(4)=legend(bh,'Histogram of J','Histogram of E','Gaussian fitted to J','Gaussian fitted to E',2);

set(leg,'fontsize',6);
Children=get(F3,'children');
set(Children,'FontSize',7);

saveas(F3,[DestFolder,'Performance\CompareTestSet.png'],'png');

```

E.19 PlotPerformance.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "PlotPerformance.m"
%
% Description: Script used for plotting the results from the test set.
%
% Input: Files containg results from color segmentation of classification
% using different methods.
%
% Output: Figure showing performance for each example.
%
% Remarks: Corresponds to the figures appendix D in the report.
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
close all;
clear all;
clc;

CHOOSE_DATASET=1; %0=Traing set, 1= Test Set
if CHOOSE_DATASET
    DestFolder=['SGE\Video\FilesTestSet'];
else
    DestFolder=['SGE\Video\FilesTrainingSet'];
end
Drive = 'E:\';
DestFolder=[Drive, DestFolder, '\'];
addpath(DestFolder);

Var=81;
MergingSizeLimit=[100,10]; %[100,10]
CorrThreshold=[0.05,0.1]; %[0.05,0.1]
VarOffset=4;
RBThreshold=3; %3

Parameters1=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit(1)), '_Corr', ...
    sprintf('%.2f', CorrThreshold(1))];
Parameters2=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit(2)), '_Corr', ...
    sprintf('%.2f', CorrThreshold(1))];
Parameters3=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit(2)), '_Corr', ...
    sprintf('%.1f', CorrThreshold(2)), '_'];
ParametersImproved=['_VarOffset', num2str(VarOffset)];
ParametersEnhanced=['_RB', sprintf('%.1f', RBThreshold), '_'];

DATASET=DataSets(CHOOSE_DATASET); % 0=Training Set (18 examples), 1=Test Set (72 examples)

for j=[1:size(DATASET,1)]
    TestNo=DATASET(j,1);
    ImNo=DATASET(j,2);
    ObjectNo=DATASET(j,3);
    TestFolder=['Test', num2str(TestNo)];
    disp(['Processing ', TestFolder, ' - Frame no. ', num2str(ImNo), ' - Object No. ', ...
        num2str(ObjectNo), '...']);
    addpath(['E:\SGE\Video\', TestFolder, '_Object']);
    addpath(['E:\SGE\Video\Temp']);

    fg_clean_filename=[TestFolder, '_binært_', num2str(ImNo), '_SGE_Clean_', num2str(ObjectNo), '.bmp'];
    fg_mask_full = double(imread(fg_clean_filename));
    Stat_All = regionprops(fg_mask_full, 'Area', 'BoundingBox');
end

```

```

Border= 10;

if length(Stat_All)
    [YOrg,XOrg,Z]=size(fg_mask_full);
    Stat_All = regionprops(fg_mask_full, 'Area','BoundingBox');
    BBoxAll=Stat_All.BoundingBox;
    BBoxAll=[BBoxAll(1)-Border,BBoxAll(2)-Border,BBoxAll(3)+2*Border,BBoxAll(4)+2*Border];
    if BBoxAll(1)<1
        BBoxAll(3)=BBoxAll(3)-(0.5-BBoxAll(1));
        BBoxAll(1)=0.5;
    end
    if BBoxAll(1)+BBoxAll(3)>XOrg+1
        BBoxAll(3)=XOrg-BBoxAll(1)-0.5;
    end
    if BBoxAll(2)<1
        BBoxAll(4)=BBoxAll(4)-(0.5-BBoxAll(2));
        BBoxAll(2)=0.5;
    end
    if BBoxAll(2)+BBoxAll(4)>YOrg+1
        BBoxAll(4)=YOrg-BBoxAll(2)-0.5;
    end
end

filename_Merged{1}=[DestFolder,TestFolder,'_JavedMerged_',num2str(ImNo),'_',...
    num2str(ObjectNo),Parameters1(1:end-8),'.mat'];
filename_Merged{2}=[DestFolder,TestFolder,'_JavedImprovedMerged_',num2str(ImNo),'_',...
    num2str(ObjectNo),ParametersImproved,Parameters2(1:end-8),'.mat'];
filename_edge_rec=[DestFolder,TestFolder,'_EdgeReconstruction_',num2str(ImNo),'_',...
    num2str(ObjectNo),'.bmp'];
file_perf{1}=[DestFolder,TestFolder,'_JavedClass_',num2str(ImNo),'_',num2str(ObjectNo),...
    Parameters1, '.mat'];
file_perf{2}=[DestFolder,TestFolder,'_JavedImprovedClass_',num2str(ImNo),'_',...
    num2str(ObjectNo),ParametersImproved,Parameters2, '.mat'];
file_perf{3}=[DestFolder,TestFolder,'_EnhancedImprovedClass_',num2str(ImNo),'_',...
    num2str(ObjectNo),ParametersImproved,Parameters3,ParametersEnhanced, '.mat'];
filename_label=[DestFolder,TestFolder,'_TrueLabel_',num2str(ImNo),'_',num2str(ObjectNo),'.bmp'];

% load manually labelled image
ImTrue=imread(filename_label);
AllDarkPixelIdxList=find(ImTrue>0);

% load region merged images
for n=1:2
    load(filename_Merged{n});
    NoOfRegions(n)=max(pic_fg_class_merged(:));
    MergedRGB{n}=label2rgb(pic_fg_class_merged);
end

% load reconstructed image
Rec_full=imread(filename_edge_rec);
Rec=imcrop(Rec_full,BBoxAll);
[Y,X,Z]=size(Rec);
ImPerf{1}=uint8(255*ones(Y*X,Z));
ImPerf{2}=uint8(255*ones(Y*X,Z));
ImPerf{3}=uint8(255*ones(Y*X,Z));
Colors=[0,0,255;255,255,0;255,0,0;0,255,0]; % [blue;yellow;red;green]

% load performance file and construct performance figures
for q=1:2

    load(file_perf{q});

```

```

for m=1:4
    ImTemp=uint8(zeros(Y,X));
    switch m
        case 1
            Imperf{q}(AllDarkPixelIdxList(find(TNJavedPixels)),1)=Colors(m,1);
            Imperf{q}(AllDarkPixelIdxList(find(TNJavedPixels)),2)=Colors(m,2);
            Imperf{q}(AllDarkPixelIdxList(find(TNJavedPixels)),3)=Colors(m,3);
        case 2
            Imperf{q}(AllDarkPixelIdxList(find(FPJavedPixels)),1)=Colors(m,1);
            Imperf{q}(AllDarkPixelIdxList(find(FPJavedPixels)),2)=Colors(m,2);
            Imperf{q}(AllDarkPixelIdxList(find(FPJavedPixels)),3)=Colors(m,3);
        case 3
            Imperf{q}(AllDarkPixelIdxList(find(FNJavedPixels)),1)=Colors(m,1);
            Imperf{q}(AllDarkPixelIdxList(find(FNJavedPixels)),2)=Colors(m,2);
            Imperf{q}(AllDarkPixelIdxList(find(FNJavedPixels)),3)=Colors(m,3);
        case 4
            Imperf{q}(AllDarkPixelIdxList(find(TPJavedPixels)),1)=Colors(m,1);
            Imperf{q}(AllDarkPixelIdxList(find(TPJavedPixels)),2)=Colors(m,2);
            Imperf{q}(AllDarkPixelIdxList(find(TPJavedPixels)),3)=Colors(m,3);
    end
end
Imperf{q}=reshape(Imperf{q},Y,X,Z);
Perf{q}=[ACJaved;ConfMatrixJavedPerc(2,2);ConfMatrixJavedPerc(1,2);...
        ConfMatrixJavedPerc(2,1);ConfMatrixJavedPerc(1,1)]; %[AC,TP,FP,FP,FN,TN]
for m=1:5
    PerfText{q}{m}=sprintf('% .1f ',Perf{q}(m));
    if length(PerfText{q}{m})==3
        PerfText{q}{m}=[' ',PerfText{q}{m}];
    else
        if length(PerfText{q}{m})==4
            PerfText{q}{m}=[' ',PerfText{q}{m}];
        end
    end
end
end
end

load(file_perf{3});
for m=1:4
    ImTemp=uint8(zeros(Y,X));
    switch m
        case 1
            Imperf{3}(AllDarkPixelIdxList(find(TNEnhancedPixels)),1)=Colors(m,1);
            Imperf{3}(AllDarkPixelIdxList(find(TNEnhancedPixels)),2)=Colors(m,2);
            Imperf{3}(AllDarkPixelIdxList(find(TNEnhancedPixels)),3)=Colors(m,3);
        case 2
            Imperf{3}(AllDarkPixelIdxList(find(FPEnhancedPixels)),1)=Colors(m,1);
            Imperf{3}(AllDarkPixelIdxList(find(FPEnhancedPixels)),2)=Colors(m,2);
            Imperf{3}(AllDarkPixelIdxList(find(FPEnhancedPixels)),3)=Colors(m,3);
        case 3
            Imperf{3}(AllDarkPixelIdxList(find(FNEnhancedPixels)),1)=Colors(m,1);
            Imperf{3}(AllDarkPixelIdxList(find(FNEnhancedPixels)),2)=Colors(m,2);
            Imperf{3}(AllDarkPixelIdxList(find(FNEnhancedPixels)),3)=Colors(m,3);
        case 4
            Imperf{3}(AllDarkPixelIdxList(find(TPEnhancedPixels)),1)=Colors(m,1);
            Imperf{3}(AllDarkPixelIdxList(find(TPEnhancedPixels)),2)=Colors(m,2);
            Imperf{3}(AllDarkPixelIdxList(find(TPEnhancedPixels)),3)=Colors(m,3);
    end
end
Imperf{3}=reshape(Imperf{3},Y,X,Z);
%[AC,TP,FP,FP,FN,TN]
Perf{3}=[ACEnhanced;ConfMatrixEnhancedPerc(2,2);ConfMatrixEnhancedPerc(1,2);...

```

```

        ConfMatrixEnhancedPerc(2,1);ConfMatrixEnhancedPerc(1,1)];
for m=1:5
    PerfText{3}{m}=sprintf('%.1f',Perf{3}(m));
    if length(PerfText{3}{m})==3
        PerfText{3}{m}=[' ',PerfText{3}{m}];
    else
        if length(PerfText{3}{m})==4
            PerfText{3}{m}=[' ',PerfText{3}{m}];
        end
    end
end

% Final figure of performance, containing the color segmentation using
% methods J and I, the reconstructed "semi-shadow-free" image, and the
% performance results of methods J,I and E.
Image=[MergedRGB{1},MergedRGB{2},Rec,uint8(255*ones(Y,X,Z));ImPerf{1},ImPerf{2},...
    ImPerf{3},uint8(255*ones(Y,X,Z))];
Text1={['TestFolder, ' - Frame no. ',num2str(ImNo)];['Object no. ',num2str(ObjectNo)];...
    ['Javed: ',num2str(NoOfRegions(1)),' merged reg.'];...
    ['Improved: ',num2str(NoOfRegions(2)),' merged reg.'];
Text2={'Absolute Performance:'};
Text3={' ', ' J ', ' I ', ' E '};
TextAC={'AC: ',PerfText{1}{1},PerfText{2}{1},PerfText{3}{1}};
TextTN={'TN: ',PerfText{1}{5},PerfText{2}{5},PerfText{3}{5}};
TextFP={'FP: ',PerfText{1}{3},PerfText{2}{3},PerfText{3}{3}};
TextFN={'FN: ',PerfText{1}{4},PerfText{2}{4},PerfText{3}{4}};
TextTP={'TP: ',PerfText{1}{2},PerfText{2}{2},PerfText{3}{2}};

F40=figure(40);
imshow(Image,[]);
hold on
T1=text(X*3.05,-Y*0.05,Text1,'VerticalAlignment','top','FontName','courier');
T2=text(X*3.05,Y-1.5*Y/5,Text2,'VerticalAlignment','top','FontName','courier');
T3=text(X*3.05,Y-0.5*Y/5,Text3,'VerticalAlignment','top','FontName','courier');
T4=text(X*3.05,Y+0.5*Y/5,TextAC,'VerticalAlignment','top','FontName','courier');
T5=text(X*3.05,Y+1.5*Y/5,TextTN,'VerticalAlignment','top','FontName','courier',...
    'BackgroundColor','b','color',[0.9,0.9,0.9]);
T6=text(X*3.05,Y+2.5*Y/5,TextFP,'VerticalAlignment','top','FontName','courier',...
    'BackgroundColor','y');
T7=text(X*3.05,Y+3.5*Y/5,TextFN,'VerticalAlignment','top','FontName','courier',...
    'BackgroundColor','r');
T8=text(X*3.05,Y+4.5*Y/5,TextTP,'VerticalAlignment','top','FontName','courier',...
    'BackgroundColor','g');
hold off
drawnow;
CropPNG(F40,['DestFolder,TestFolder','_FinalPerformance_',num2str(ImNo),'_',...
    num2str(ObjectNo),'_png'],'_Cropped');
end

```

E.20 Calibration.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "Calibration.m"
%
% Description: Script that performs color calibration of a camera using
% images of the Macbeth color chart.
%
% Input: Images of color chart
%
% Output: Figure of calibration, files of annotation for each image
%

```



```

% Author: Søren Erbou (SGE)
% Last Revision: July 10, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;
clear all;
clc;

MaskingDone=1;      % 0=Define/controlle colorpatch for every image. 1=color patches exist
Set=200;            % sequence no.
Format='bmp';

NoOfColors=24;
Patches=[1:19,20:24]; % Which patches to plot: 1=upper left corner, 6=upper right corner
                    % 18=lower left corner, 24=lower right corner

xlim=[0.5 1024.5];
ylim=[0.5 768.5];
SourceFolder=['E:\SGE\Video\CalSeq\Calibration',num2str(Set),'\'];
Drive = 'E:\';

Files=dir([SourceFolder,'*.',Format]);
MaskInd={};
Mean=zeros(NoOfColors,3,length(Files));
Std=zeros(NoOfColors,3,length(Files));
GetMask=0;
OldMaskExist=0;
for n=1:length(Files)
    filename=Files(n).name;
    ImNo=str2double(filename(find(filename=='t')+1:find(filename=='_')-1));
    pic_in = imread([SourceFolder,filename]);
    [Y,X,Z]=size(pic_in);
    pic_in2=reshape(pic_in,Y*X,Z);

    if exist([SourceFolder,'Test',num2str(ImNo),'_MaskIndex_',num2str(NoOfColors),...
            '_Colors.mat'])==2
        load([SourceFolder,'Test',num2str(ImNo),'_MaskIndex_',num2str(NoOfColors),...
            '_Colors.mat']);
        if ~MaskingDone
            F3=figure(3);
            imshow(pic_in,[])
            hI3=get(F3,'Children');
            set(hI3,'Xlim',xlim,'Ylim',ylim);
            t=title([filename(1:end-4),' - Mask']);
            set(t,'Interpreter','none');
            hold on
            for k=1:NoOfColors
                [IndY,IndX]=ind2sub([Y,X],MaskInd{k});
                plot(IndX,IndY,'b');
            end
            Ans=questdlg('Use this mask?','?');
            if strcmp(Ans,'No');
                GetMask=1;
            else
                GetMask=0;
            end
            close(F3);
        end
        OldMaskExist=1;
    else

```

```

if ~MaskingDone
    if OldMaskExist
        F3=figure(3);
        imshow(pic_in,[])
        hI3=get(F3,'Children');
        set(hI3,'Xlim',xlim,'Ylim',ylim);
        t=title([filename(1:end-4),' - Previous Mask']);
        set(t,'Interpreter','none');
        hold on
        for k=1:NoOfColors
            [IndY,IndX]=ind2sub([Y,X],MaskInd{k});
            plot(IndX,IndY,'b');
        end
        Ans=questdlg('Use previous mask?','?');
        if strcmp(Ans,'No');
            GetMask=1;
        else
            GetMask=0;
        end
        close(F3);
    else
        GetMask=1;
    end
end

end
end

F4=figure('Visible','off');
imshow(pic_in,[]);
hI4=get(F4,'Children');
set(hI4,'Xlim',xlim,'Ylim',ylim);
for k=1:NoOfColors
    if GetMask
        set(F4,'Visible','on')
        MaskInd{k}=find(roipoly);
    end
    Mean(k,1,n)=mean(double(pic_in2(MaskInd{k},1)));
    Mean(k,2,n)=mean(double(pic_in2(MaskInd{k},2)));
    Mean(k,3,n)=mean(double(pic_in2(MaskInd{k},3)));
    Std(k,1,n)=std(double(pic_in2(MaskInd{k},1)));
    Std(k,2,n)=std(double(pic_in2(MaskInd{k},2)));
    Std(k,3,n)=std(double(pic_in2(MaskInd{k},3)));
end
if GetMask
    save([SourceFolder,'Test',num2str(ImNo),'_MaskIndex_',...
        num2str(NoOfColors),'_Colors'],'MaskInd');
    OldMaskExist=1;
end
end

end

MeanRG=reshape(log([(1+Mean(:,1,:))./(1+Mean(:,2,:))]),NoOfColors,length(Files));
MeanBG=reshape(log([(1+Mean(:,3,:))./(1+Mean(:,2,:))]),NoOfColors,length(Files));

Markers6={['b*'],'g*'],'r*'],'y*'],'m*'],'c*'];
Markers24={['b*'],'g*'],'r*'],'y*'],'m*'],'c*'],...
    {'bo'],'go'],'ro'],'yo'],'mo'],'co'],...
    {'bx'],'gx'],'rx'],'yx'],'mx'],'cx'],...
    {'b.'}','g.'}','r.'}','y.'}','m.'}','c.'}];

lambda=[613 540 462]*1e-9; % Center of spectral response[R,G,B]=[613 540 462]*1e-9
q=[0.98 1 0.86]; % Sensitivity

```

```

c1=3.74183e-16;
c2=1.4388e-2;
e=-c2./lambda;
a_m=[1;-(e(1)-e(2))/(e(3)-e(2))];
a=a_m/norm(a_m);
a=[a(2),-a(1)];

p=[];
p1=[];
l=[];
DirChange=[];

LengthFactor=0.5;
Fit=zeros(NoOfColors,2);
MeanColor=[mean(MeanRG,2),mean(MeanBG,2)];
F5=figure;

for j=1:length(Patches)
    p(end+1)=plot(MeanRG(Patches(j),1:end),MeanBG(Patches(j),1:end),Markers24{Patches(j)});
    hold on
    Fit(Patches(j,:),:)=polyfit(MeanRG(Patches(j),:),MeanBG(Patches(j),:),1);
    FitXY(Patches(j,:),:)=Fit(Patches(j),1)/norm([1,Fit(Patches(j),1)]);
    l(end+1)=line([-0.5*LengthFactor*FitXY(Patches(j),1),0.5*LengthFactor*FitXY(Patches(j),1)],...
        [-0.5*LengthFactor*FitXY(Patches(j),2),0.5*LengthFactor*FitXY(Patches(j),2)],...
        'Color',get(p(j),'Color'));
end

if NoOfColors==6
    legend(p,'Cyan','Violet','Magenta','Red','Yellow','Green')
else
    if NoOfColors==24
        Text24={'1-Dark Skin','2-Light Skin','3-Blue sky','4-Foilage Green','5-Blue Flower',...
            '6-Bluish Green','7-Orange','8-Purplish Blue','9-Moderate Red','10-Purple',...
            '11-Yellow Green','12-Orange Yellow','13-Blue','14-Green','15-Red',...
            '16-Yellow','17-Magenta','18-Cyan','19-White','20-Light Gray',...
            '21-Medium Gray 1','22-Medium Gray 2','23-Dark Gray','24-Black'};
        l=legend(p,Text24(Patches));
        set(l,'FontSize',8);
    end
end

ylabel('Ln(B/G)');
xlabel('Ln(R/G)');
AllAngles=180/pi*atan2(FitXY(Patches,2),FitXY(Patches,1));
MeanFitXY=mean(FitXY(Patches,:));
StdAngle=round(std(AllAngles)*10)/10;
MeanAngle=mean(AllAngles);

TrueLine=line([-a(1) a(1)],[-a(2) a(2)],'LineWidth',2,'LineStyle','--');
MeanFith=line([-MeanFitXY(1) MeanFitXY(1)],[-MeanFitXY(2) MeanFitXY(2)],'LineWidth',2,...
    'LineStyle','-','Color','r');
MeanFith=line(MeanFitXY(1)+[-3*MeanFitXY(2) 3*MeanFitXY(2)],...
    MeanFitXY(2)+[3*MeanFitXY(1) -3*MeanFitXY(1)],'LineWidth',2,'LineStyle','-','Color','k');

Angles=round(10*180/pi*[atan2(a(2),a(1)),atan2(MeanFitXY(2),MeanFitXY(1))]/10);
set(gca,'DataAspectRatio',[1 1 1]);
axis([-1.5 2.5 -2 1]);
T1=text(-0.3,0.75,{'Invariant direction derived from:'];[;]...
    ['Spectral sensitivity (blue line) = ',num2str(Angles(1)),' deg.'];...
    ['Calibration of ',num2str(length(Files)),' images (red line) = ',...
    num2str(Angles(2)),' +/- ',num2str(StdAngle),' deg.'], 'FontSize',8);

```

```

title({'Chromaticities of ',num2str(length(Patches)), ' Color Patches from ',...
      num2str(length(Files)), ' Test Images - Set No. ',num2str(Set)];...
      ['Angle from spectral sensors = ',num2str(Angles(1)),...
      ' deg. - Angle from calibration = ',num2str(Angles(2)), ' deg.']);
title('')
saveas(F5,[SourceFolder,'CalibrationSet',num2str(Set),'.png'],'png');

```

E.21 CropPNG.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "CropPNG.m"
%
% Description: Function that crops a "png"-image to its bounding box, to
% avoid thick white borders
%
% Input: Figurehandle, filename and extension of filename to cropped
% figure.
%
% Output: "png"-file of cropped figure.
%
% Author: Søren Erbou (SGE)
% Last Revision: September 14, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

function CropPNG(FigureHandle,Filename,Extension)

saveas(FigureHandle,Filename,'png');
close(FigureHandle);
Im2=imread(Filename);
Im4=uint8(imcomplement(Im2(:,:,1)>254));
Im4Stat=regionprops(Im4,'boundingbox');
Im4Stat.BoundingBox(2)=fix(Im4Stat.BoundingBox(2)*0.98);
Im4Stat.BoundingBox(3)=fix(Im4Stat.BoundingBox(3)*1.01);
Im4Stat.BoundingBox(4)=fix(Im4Stat.BoundingBox(4)*1.04);
Im5=imcrop(Im2,Im4Stat.BoundingBox);
imwrite(Im5,[Filename(1:end-4),Extension,'.png'],'png');

```

E.22 OptimizeJaved.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "OptimizeJaved.m"
%
% Description: Script that collects the performance of the parameter values
% for the training set, for determining optimal performance for Javed's method.
%
% Input: Files with performance results of training set.
%
% Output: Figures showing performance.
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

close all;
clear all;
clc;

addpath(['E:\SGE\Video\FilesTrainingSet']);
DestFolder=['SGE\Video\FilesTrainingSet\Analysis'];

```

```

Drive = 'E:\';
if exist([Drive, DestFolder], 'dir')~=7
    Dir=pwd;
    cd(Dir);
    mkdir(DestFolder);
    cd(Dir);
end
DestFolder=[Drive, DestFolder, '\'];
addpath(DestFolder);

%%%% Javed %%%%
% K-Means
VAR=[25 36 49 64 81 100];
% Merging
MERGINGSIZE=[10 30 50 70 100 150];
% Correlation
CORRTHRES=[0.0 0.05 0.1 0.15 0.2 0.3];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%55

DATASET=DataSets(0) % 0=Training Set (18 examples), 1=Test Set (72 examples)

filename_OptimizeJaved=[DestFolder, 'OptimizeJaved.mat'];
if exist(filename_OptimizeJaved)~=2
    for j=[1:size(DATASET,1)]
        TestNo=DATASET(j,1);
        ImNo=DATASET(j,2);
        ObjectNo=DATASET(j,3);
        TestFolder=['Test', num2str(TestNo)];
        Acc{j, ObjectNo}=zeros(length(VAR), length(MERGINGSIZE), length(CORRTHRES));
        for VarCount=1:length(VAR)
            for MerCount=1:length(MERGINGSIZE)
                for CorrCount=1:length(CORRTHRES)
                    Var=VAR(VarCount);
                    MergingSizeLimit=MERGINGSIZE(MerCount);
                    CorrThreshold=CORRTHRES(CorrCount);
                    if mod(CorrThreshold, 0.1)==0
                        Parameters=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit), ...
                            '_Corr', sprintf('%.1f', CorrThreshold), '_'];
                    else
                        Parameters=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit), ...
                            '_Corr', sprintf('%.2f', CorrThreshold)];
                    end
                    load([TestFolder, '_JavedClass_', num2str(ImNo), '_', num2str(ObjectNo), ...
                        Parameters, '.mat']);
                    Acc{j, ObjectNo}(VarCount, MerCount, CorrCount)=ACJaved;
                    tp{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(2,2);
                    fp{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(1,2);
                    tn{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(1,1);
                    fn{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(2,1);
                    NoOfPixels(j, ObjectNo)=sum(ConfMatrixJaved(:));
                end
            end
        end
        save([filename_OptimizeJaved], 'Acc', 'tp', 'fp', 'tn', 'fn', 'NoOfPixels');
    else
        load([filename_OptimizeJaved]);
    end

for w=1:6

```

```

for q=1:size(DATASET,1)
    ACC{w}(:,:,q)=Acc{q}(:,:,w);
    TP{w}(:,:,q)=tp{q}(:,:,w);
    FP{w}(:,:,q)=fp{q}(:,:,w);
    TN{w}(:,:,q)=tn{q}(:,:,w);
    FN{w}(:,:,q)=fn{q}(:,:,w);
end
Mean(:,:,w)=mean(ACC{w},3);
Std(:,:,w)=std(ACC{w},0,3);
MeanFP(:,:,w)=mean(FP{w},3);
StdFP(:,:,w)=std(FP{w},0,3);
MeanTP(:,:,w)=mean(TP{w},3);
StdTP(:,:,w)=std(TP{w},0,3);
MeanFN(:,:,w)=mean(FN{w},3);
StdFN(:,:,w)=std(FN{w},0,3);
MeanTN(:,:,w)=mean(TN{w},3);
StdTN(:,:,w)=std(TN{w},0,3);
end

[Max,a]=max(Mean(:));
[a,b,c]=ind2sub([length(VAR),length(MERGINGSIZE),length(CORRTHRES)],a);
a=5;
b=5;
c=2;

F30=figure(30)
for w=1:6
    subplot(2,3,w)
    H(w)=mesh(MERGINGSIZE,VAR,Mean(:,:,w));
    axis([10 150 20 100 50 90]);
    ylabel('\sigma^2','FontSize',8);
    xlabel({' Merging size ';'threshold [pixels]'},'FontSize',8);
    zlabel('Accuracy (AC) [%]','FontSize',8);
    title(['Correlation threshold = ',num2str(CORRTHRES(w))],'FontSize',8);
end
set(get(F30,'children'),'FontSize',8);
subplot(2,3,c)
hold on
plot3(MERGINGSIZE(b),VAR(a),Max,'k*','Markersize',8);
hold off
saveas(F30,[DestFolder,'JavedOptimizationAC.png'],'png');

F31=figure(31);
H31(1)=plot(MeanFP(:,b,c),MeanTP(:,b,c),'b-x','linewidth',2,'markersize',8);
hold on
H31(2)=plot(MeanFP(a,:,c),MeanTP(a,:,c),'r-x','linewidth',2,'markersize',8);
H31(3)=plot(reshape([MeanFP(a,b,:)],length(CORRTHRES),1),reshape([MeanTP(a,b,:)],...
    length(CORRTHRES),1),'g-x','linewidth',2,'markersize',8);
axis([20 50 65 95]);
xlabel('False positives (FP) [%]','FontSize',12);
ylabel('True positives (TP) [%]','FontSize',12);
title('ROC-curve','FontSize',12);
grid on
set(get(F31,'children'),'FontSize',12);
hold on
H31(4)=plot(MeanFP(a,b,c),MeanTP(a,b,c),'k*','Markersize',10);
hold off
l=legend(H31,'Fixed variance \sigma^2','Merging threshold','Correlation threshold',...
    'Optimum performance',4);
set(l,'FontSize',10);
saveas(F31,[DestFolder,'JavedOptimizationROC.png'],'png');

```



```

close all;
clear all;
clc;

addpath(['E:\SGE\Video\FilesTrainingSet']);
DestFolder=['SGE\Video\FilesTrainingSet\Analysis'];
Drive = 'E:\';
if exist([Drive, DestFolder], 'dir')~=7
    Dir=pwd;
    cd(Dir);
    mkdir(DestFolder);
    cd(Dir);
end
DestFolder=[Drive, DestFolder, '\'];
addpath(DestFolder);
%%%% Javed %%%%
% K-Means
VAR=[25 36 49 64 81 100];
% Merging
MERGINGSIZE=[10 30 50 70 100 150];
% Correlation
CORRTHRES=[0.0 0.05 0.1 0.15 0.2 0.3];
%%%%%%%%%%%%%%55
VAROFFSET=[4];

DATASET=DataSets(0); % 0=Training Set (18 examples), 1=Test Set (72 examples)
filename_OptimizeJavedImproved=[DestFolder, 'OptimizeJavedImproved.mat'];
if exist(filename_OptimizeJavedImproved)~=2
    for j=[1:size(DATASET,1)]
        TestNo=DATASET(j,1);
        ImNo=DATASET(j,2);
        ObjectNo=DATASET(j,3);
        TestFolder=['Test', num2str(TestNo)];
        Acc{j, ObjectNo}=zeros(length(VAR), length(MERGINGSIZE), length(CORRTHRES));
        for VarCount=1:length(VAR)
            for MerCount=1:length(MERGINGSIZE)
                for CorrCount=1:length(CORRTHRES)
                    for VarOffsetCount=1:length(VAROFFSET)
                        VarOffset=VAROFFSET(VarOffsetCount);
                        Var=VAR(VarCount);
                        MergingSizeLimit=MERGINGSIZE(MerCount);
                        CorrThreshold=CORRTHRES(CorrCount);
                        if mod(CorrThreshold, 0.1)==0
                            Parameters=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit), ...
                                '_Corr', sprintf('%0.1f', CorrThreshold), '_'];
                        else
                            Parameters=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit), ...
                                '_Corr', sprintf('%0.2f', CorrThreshold)];
                        end
                        ParametersImproved=['_VarOffset', num2str(VarOffset)];

                        load([TestFolder, '_JavedImprovedClass_', num2str(ImNo), '_', ...
                            num2str(ObjectNo), ParametersImproved, Parameters, '.mat']);
                        Acc{j, ObjectNo}(VarCount, MerCount, CorrCount)=ACJaved;
                        tp{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(2, 2);
                        fp{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(1, 2);
                        tn{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(1, 1);
                        fn{j, ObjectNo}(VarCount, MerCount, CorrCount)=ConfMatrixJavedPerc(2, 1);
                        NoOfPixels(j, ObjectNo)=sum(ConfMatrixJaved(:));
                    end
                end
            end
        end
    end
end

```



```

                end
            end
        end
        save([filename_OptimizeJavedImproved],'Acc','tp','fp','tn','fn','NoOfPixels');
    else
        load([filename_OptimizeJavedImproved]);
    end

    for w=1:6
        for q=1:size(DATASET,1)
            ACC{w}(:,:,q)=Acc{q}(:,:,w);
            TP{w}(:,:,q)=tp{q}(:,:,w);
            FP{w}(:,:,q)=fp{q}(:,:,w);
            TN{w}(:,:,q)=tn{q}(:,:,w);
            FN{w}(:,:,q)=fn{q}(:,:,w);
        end
        Mean(:,:,w)=mean(ACC{w},3);
        Std(:,:,w)=std(ACC{w},0,3);
        MeanFP(:,:,w)=mean(FP{w},3);
        StdFP(:,:,w)=std(FP{w},0,3);
        MeanTP(:,:,w)=mean(TP{w},3);
        StdTP(:,:,w)=std(TP{w},0,3);
        MeanFN(:,:,w)=mean(FN{w},3);
        StdFN(:,:,w)=std(FN{w},0,3);
        MeanTN(:,:,w)=mean(TN{w},3);
        StdTN(:,:,w)=std(TN{w},0,3);
    end

    [Max,a]=max(Mean(:));
    [a,b,c]=ind2sub([length(VAR),length(MERGINGSIZE),length(CORRTHRES)],a);
    a=5;
    b=1;
    c=2;
    F30=figure(30)
    for w=1:6
        subplot(2,3,w)
        H(w)=mesh(MERGINGSIZE,VAR,Mean(:,:,w));
        axis([10 150 20 100 50 95]);
        ylabel('\sigma^2','FontSize',8);
        xlabel({' Merging size ', 'threshold [pixels]'},'FontSize',8);
        zlabel('Accuracy (AC) [%]','FontSize',8);
        title(['Correlation threshold = ',num2str(CORRTHRES(w))],'FontSize',8);
    end
    set(get(F30,'children'),'FontSize',8);
    subplot(2,3,c)
    hold on
    plot3(MERGINGSIZE(b),VAR(a),Max+1,'k*','Markersize',8);
    hold off
    saveas(F30,[DestFolder,'JavedImprovedOptimizationAC.png'],'png');

    F31=figure(31);
    H31(1)=plot(MeanFP(:,b,c),MeanTP(:,b,c),'b-x','linewidth',2,'markersize',8);
    hold on
    H31(2)=plot(MeanFP(a,:,c),MeanTP(a,:,c),'r-x','linewidth',2,'markersize',8);
    H31(3)=plot(reshape([MeanFP(a,b,:)],length(CORRTHRES),1),reshape([MeanTP(a,b,:)],...
        length(CORRTHRES),1),'g-x','linewidth',2,'markersize',8);
    xlabel('False positives (FP) [%]','FontSize',12);
    ylabel('True positives (TP) [%]','FontSize',12);
    title('ROC-curve','FontSize',12);
    grid on

```

```

set(get(F31,'children'),'FontSize',12);
hold on
H31(4)=plot(MeanFP(a,b,c),MeanTP(a,b,c),'k*','Markersize',10);
hold off
l=legend(H31,'Fixed variance \sigma^2','Merging threshold','Correlation threshold',...
         'Optimum performance',4);
set(l,'FontSize',10);
saveas(F31,[DestFolder,'JavedImprovedOptimizationROC.png'],'png');

F32=figure(32)
for w=1:6
    subplot(2,3,w)
    H(w)=mesh(MERGINGSIZE,VAR,MeanFP(:,:,w));
    axis([10 150 20 100 10 90]);
    ylabel('\sigma^2','FontSize',8);
    xlabel({' Merging size ';'threshold [pixels]'},'FontSize',8);
    zlabel('False objects (FP) [%]','FontSize',8);
    title(['Correlation threshold = ',num2str(CORRTHRES(w))],'FontSize',8);
end
set(get(F32,'children'),'FontSize',8);
subplot(2,3,c)
hold on
plot3(MERGINGSIZE(b),VAR(a),MeanFP(a,b,c)+1,'k*','Markersize',8);
hold off
saveas(F32,[DestFolder,'JavedImprovedOptimizationFP.png'],'png');

F33=figure(33);
Markers={'b.','r.','g.','c.','m.','y.'};
for w=1:6
    H33(w)=plot(reshape(MeanFP(:,:,w),36,1),reshape(MeanTP(:,:,w),36,1),...
               Markers{w},'markersize',12);
    hold on
end
xlabel('False positives (FP) [%]','FontSize',12);
ylabel('True positives (TP) [%]','FontSize',12);
title('ROC-curve','FontSize',12);
grid on
set(get(F33,'children'),'FontSize',12);
hold on
H33(7)=plot(MeanFP(a,b,c),MeanTP(a,b,c),'k*','Markersize',10);
hold off
l=legend(H33,['Corr. threshold = ',sprintf('.2f',CORRTHRES(1))],...
         ['Corr. threshold = ',sprintf('.2f',CORRTHRES(2))],...
         ['Corr. threshold = ',sprintf('.2f',CORRTHRES(3))],...
         ['Corr. threshold = ',sprintf('.2f',CORRTHRES(4))],...
         ['Corr. threshold = ',sprintf('.2f',CORRTHRES(5))],...
         ['Corr. threshold = ',sprintf('.2f',CORRTHRES(6))],...
         'Optimum performance',4);
set(l,'FontSize',10);
saveas(F33,[DestFolder,'JavedImprovedOptimizationROCA11.png'],'png');
round([Mean(a,b,c),Std(a,b,c);MeanTP(a,b,c),StdTP(a,b,c);MeanFP(a,b,c),...
      StdFP(a,b,c);MeanTN(a,b,c),StdTN(a,b,c);MeanFN(a,b,c),StdFN(a,b,c)])

```

E.24 OptimizeEnhanced.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Filename: "OptimizeEnhanced.m"
%
% Description: Script that collects the performance of the parameter values
% for the training set, for determining optimal performance for the enhanced

```

```

% method using improved color segmentation.
%
% Input: Files with performance results of training set.
%
% Output: Figures showing performance.
%
% Author: Søren Erbou (SGE)
% Last Revision: September 13, 2004
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;
clear all;
clc;

addpath(['E:\SGE\Video\FilesTrainingSet']);
DestFolder=['SGE\Video\FilesTrainingSet\Analysis'];
Drive = 'E:\';
if exist([Drive, DestFolder], 'dir')~=7
    Dir=pwd;
    cd(Dir);
    mkdir(DestFolder);
    cd(Dir);
end
DestFolder=[Drive, DestFolder, '\'];
addpath(DestFolder);
%%%% Javed %%%%
% K-Means
VAR=[25, 49, 81];
% Merging
MERGINGSIZE=[10, 50, 70, 100];
% Correlation
CORRTHRES=[0.1, 0.15, 0.2];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%55
VAROFFSET=[4];
%%%% Enhanced %%%%
RBTHRES=[3,5,7];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

DATASET=DataSets(0); % 0=Training Set (18 examples), 1=Test Set (72 examples)
filename_OptimizeEnhanced=[DestFolder, 'OptimizeEnhanced.mat'];
if exist(filename_OptimizeEnhanced)~=2
    for j=[1:size(DATASET,1)]
        TestNo=DATASET(j,1);
        ImNo=DATASET(j,2);
        ObjectNo=DATASET(j,3);
        TestFolder=['Test', num2str(TestNo)];
        Acc{j, ObjectNo}=zeros(length(VAR), length(MERGINGSIZE), length(CORRTHRES), length(RBTHRES));
        for VarCount=1:length(VAR)
            for MerCount=1:length(MERGINGSIZE)
                for CorrCount=1:length(CORRTHRES)
                    for VarOffsetCount=1:length(VAROFFSET)
                        for RBCount=1:length(RBTHRES)
                            RBThreshold=RBTHRES(RBCount);
                            VarOffset=VAROFFSET(VarOffsetCount);
                            Var=VAR(VarCount);
                            MergingSizeLimit=MERGINGSIZE(MerCount);
                            CorrThreshold=CORRTHRES(CorrCount);
                            if mod(CorrThreshold, 0.1)==0
                                Parameters=['_Var', num2str(Var), '_Mer', num2str(MergingSizeLimit), ...
                                    '_Corr', sprintf('%0.1f', CorrThreshold), '_'];
                            end
                        end
                    end
                end
            end
        end
    end
end

```

```

        Parameters=['_Var',num2str(Var),'_Mer',num2str(MergingSizeLimit),...
            '_Corr',sprintf('%.2f',CorrThreshold)];
    end
    ParametersImproved=['_VarOffset',num2str(VarOffset)];
    ParametersEnhanced=['_RB',sprintf('%.1f',RBThreshold),'_'];

    load([TestFolder,'_EnhancedImprovedClass_',num2str(ImNo),'_',...
        num2str(ObjectNo),ParametersImproved,Parameters,...
        ParametersEnhanced,'.mat']);
    switch RBThreshold
        case 3
            Acc3{j,ObjectNo}(VarCount,MerCount,CorrCount)=ACEnhanced;
            tp3{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(2,2);
            fp3{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(1,2);
            tn3{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(1,1);
            fn3{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(2,1);
        case 5
            Acc5{j,ObjectNo}(VarCount,MerCount,CorrCount)=ACEnhanced;
            tp5{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(2,2);
            fp5{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(1,2);
            tn5{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(1,1);
            fn5{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(2,1);
        case 7
            Acc7{j,ObjectNo}(VarCount,MerCount,CorrCount)=ACEnhanced;
            tp7{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(2,2);
            fp7{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(1,2);
            tn7{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(1,1);
            fn7{j,ObjectNo}(VarCount,MerCount,CorrCount)=...
                ConfMatrixEnhancedPerc(2,1);
    end
end
end
end
end
    NoOfPixels(j,ObjectNo)=sum(ConfMatrixEnhanced(:));
end
save([filename_OptimizeEnhanced],'Acc3','tp3','fp3','tn3','fn3','Acc5','tp5','fp5','tn5',...
    'fn5','Acc7','tp7','fp7','tn7','fn7','NoOfPixels');
else
    load([filename_OptimizeEnhanced]);
end

for w=1:size(CORRTHRES,2)
    for q=1:size(DATASET,1)
        ACC3{w}(:,:,q)=Acc3{q}(:,:,w);
        TP3{w}(:,:,q)=tp3{q}(:,:,w);
        FP3{w}(:,:,q)=fp3{q}(:,:,w);
        TN3{w}(:,:,q)=tn3{q}(:,:,w);
        FN3{w}(:,:,q)=fn3{q}(:,:,w);
    end
end

```

```

    ACC5{w}(:, :, q)=acc5{q}(:, :, w);
    TP5{w}(:, :, q)=tp5{q}(:, :, w);
    FP5{w}(:, :, q)=fp5{q}(:, :, w);
    TN5{w}(:, :, q)=tn5{q}(:, :, w);
    FN5{w}(:, :, q)=fn5{q}(:, :, w);
    ACC7{w}(:, :, q)=acc7{q}(:, :, w);
    TP7{w}(:, :, q)=tp7{q}(:, :, w);
    FP7{w}(:, :, q)=fp7{q}(:, :, w);
    TN7{w}(:, :, q)=tn7{q}(:, :, w);
    FN7{w}(:, :, q)=fn7{q}(:, :, w);
end
Mean3(:, :, w)=mean(ACC3{w}, 3);
Std3(:, :, w)=std(ACC3{w}, 0, 3);
MeanFP3(:, :, w)=mean(FP3{w}, 3);
StdFP3(:, :, w)=std(FP3{w}, 0, 3);
MeanTP3(:, :, w)=mean(TP3{w}, 3);
StdTP3(:, :, w)=std(TP3{w}, 0, 3);
MeanFN3(:, :, w)=mean(FN3{w}, 3);
StdFN3(:, :, w)=std(FN3{w}, 0, 3);
MeanTN3(:, :, w)=mean(TN3{w}, 3);
StdTN3(:, :, w)=std(TN3{w}, 0, 3);
Mean5(:, :, w)=mean(ACC5{w}, 3);
Std5(:, :, w)=std(ACC5{w}, 0, 3);
MeanFP5(:, :, w)=mean(FP5{w}, 3);
StdFP5(:, :, w)=std(FP5{w}, 0, 3);
MeanTP5(:, :, w)=mean(TP5{w}, 3);
StdTP5(:, :, w)=std(TP5{w}, 0, 3);
MeanFN5(:, :, w)=mean(FN5{w}, 3);
StdFN5(:, :, w)=std(FN5{w}, 0, 3);
MeanTN5(:, :, w)=mean(TN5{w}, 3);
StdTN5(:, :, w)=std(TN5{w}, 0, 3);
Mean7(:, :, w)=mean(ACC7{w}, 3);
Std7(:, :, w)=std(ACC7{w}, 0, 3);
MeanFP7(:, :, w)=mean(FP7{w}, 3);
StdFP7(:, :, w)=std(FP7{w}, 0, 3);
MeanTP7(:, :, w)=mean(TP7{w}, 3);
StdTP7(:, :, w)=std(TP7{w}, 0, 3);
MeanFN7(:, :, w)=mean(FN7{w}, 3);
StdFN7(:, :, w)=std(FN7{w}, 0, 3);
MeanTN7(:, :, w)=mean(TN7{w}, 3);
StdTN7(:, :, w)=std(TN7{w}, 0, 3);
end

[Max, a]=max(Mean(:));
[a, b, c]=ind2sub( [length(VAR), length(MERGINGSIZE), length(CORRTHRES)], a);
a=5;
b=1;
c=2;
F30=figure(30)
for w=1:6
    subplot(2,3,w)
    H(w)=mesh(MERGINGSIZE, VAR, Mean(:, :, w));
    axis([10 150 20 100 50 95]);
    ylabel('\sigma^2', 'FontSize', 8);
    xlabel(' Merging size ', 'threshold [pixels]');
    zlabel('Accuracy (AC) [%]', 'FontSize', 8);
    title(['Correlation threshold = ', num2str(CORRTHRES(w))], 'FontSize', 8);
end
set(get(F30, 'children'), 'FontSize', 8);
subplot(2,3,c)
hold on

```

```

plot3(MERGINGSIZE(b),VAR(a),Max+1,'k*','Markersize',8);
hold off
saveas(F30,[DestFolder,'EnhancedOptimizationAC.png'],'png');

F31=figure(31);
H31(1)=plot(MeanFP(:,b,c),MeanTP(:,b,c),'b-x','linewidth',2,'markersize',8);
hold on
H31(2)=plot(MeanFP(a,:,c),MeanTP(a,:,c),'r-x','linewidth',2,'markersize',8);
H31(3)=plot(reshape([MeanFP(a,b,:)],length(CORRTHRES),1),reshape([MeanTP(a,b,:)],...
    length(CORRTHRES),1),'g-x','linewidth',2,'markersize',8);
xlabel('False positives (FP) [%]','FontSize',12);
ylabel('True positives (TP) [%]','FontSize',12);
title('ROC-curve','FontSize',12);
grid on
set(get(F31,'children'),'FontSize',12);
hold on
H31(4)=plot(MeanFP(a,b,c),MeanTP(a,b,c),'k*','Markersize',10);
hold off
l=legend(H31,'Fixed variance \sigma^2','Merging threshold','Correlation threshold',...
    'Optimum performance',4);
set(l,'FontSize',10);
saveas(F31,[DestFolder,'EnhancedOptimizationROC.png'],'png');

F32=figure(32)
for w=1:6
    subplot(2,3,w)
    H(w)=mesh(MERGINGSIZE,VAR,MeanFP(:,:,w));
    axis([10 150 20 100 10 90]);
    ylabel('\sigma^2','FontSize',8);
    xlabel('Merging size ','threshold [pixels]','FontSize',8);
    ylabel('False objects (FP) [%]','FontSize',8);
    title(['Correlation threshold = ',num2str(CORRTHRES(w))],'FontSize',8);
end
set(get(F32,'children'),'FontSize',8);
subplot(2,3,c)
hold on
plot3(MERGINGSIZE(b),VAR(a),MeanFP(a,b,c)+1,'k*','Markersize',8);
hold off
saveas(F32,[DestFolder,'EnhancedOptimizationFP.png'],'png');
a=[1,3];
b=[2,2];
c=[1,2];
F33=figure(33);
Markers1={'b.','r.','g.'};
Markers2={'bx','rx','gx'};
Markers3={'bo','ro','go'};
H33=zeros(size(CORRTHRES,2),3);
for w=1:size(CORRTHRES,2)
    H33(w,1)=plot(reshape(MeanFP3(:,:,w),12,1),reshape(MeanTP3(:,:,w),12,1),...
        Markers1{w},'markersize',12);
    hold on
    H33(w,2)=plot(reshape(MeanFP5(:,:,w),12,1),reshape(MeanTP5(:,:,w),12,1),...
        Markers2{w},'markersize',6);
    H33(w,3)=plot(reshape(MeanFP7(:,:,w),12,1),reshape(MeanTP7(:,:,w),12,1),...
        Markers3{w},'markersize',6);
end
H33=H33(:);
xlabel('False positives (FP) [%]','FontSize',12);
ylabel('True positives (TP) [%]','FontSize',12);
grid on
set(get(F33,'children'),'FontSize',12);

```

```

hold on
H33(end+1)=plot(MeanFP3(a(1),b(1),c(1)),MeanTP3(a(1),b(1),c(1)), 'k*', 'MarkerSize', 10);
hold off
l=legend(H33(:), ['( ', sprintf('.2f', CORRTHRES(1)), ' ', ' ', sprintf('%1.f', RBTHRES(1)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(2)), ' ', ' ', sprintf('%1.f', RBTHRES(1)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(3)), ' ', ' ', sprintf('%1.f', RBTHRES(1)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(1)), ' ', ' ', sprintf('%1.f', RBTHRES(2)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(2)), ' ', ' ', sprintf('%1.f', RBTHRES(2)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(3)), ' ', ' ', sprintf('%1.f', RBTHRES(2)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(1)), ' ', ' ', sprintf('%1.f', RBTHRES(3)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(2)), ' ', ' ', sprintf('%1.f', RBTHRES(3)), ' )'], ...
    ['( ', sprintf('.2f', CORRTHRES(3)), ' ', ' ', sprintf('%1.f', RBTHRES(3)), ' )'], ...
    'Opt. perf.', 4);
set(l, 'FontSize', 10);
saveas(F33, [DestFolder, 'EnhancedOptimizationROCA11.png'], 'png');
round([Mean3(a(1),b(1),c(1)), Std3(a(1),b(1),c(1)); MeanTP3(a(1),b(1),c(1)), StdTP3(a(1),b(1), ...
    c(1)); MeanFP3(a(1),b(1),c(1)), StdFP3(a(1),b(1),c(1)); MeanTN3(a(1),b(1),c(1)), ...
    StdTN3(a(1),b(1),c(1)); MeanFN3(a(1),b(1),c(1)), StdFN3(a(1),b(1),c(1))])
round([Mean7(a(2),b(2),c(2)), Std7(a(2),b(2),c(2)); MeanTP7(a(2),b(2),c(2)), ...
    StdTP7(a(2),b(2),c(2)); MeanFP7(a(2),b(2),c(2)), StdFP7(a(2),b(2),c(2)); MeanTN7(a(1), ...
    b(1),c(1)), StdTN7(a(1),b(1),c(1)); MeanFN7(a(1),b(1),c(1)), StdFN7(a(1),b(1),c(1))])

```


Appendix F

Flowcharts

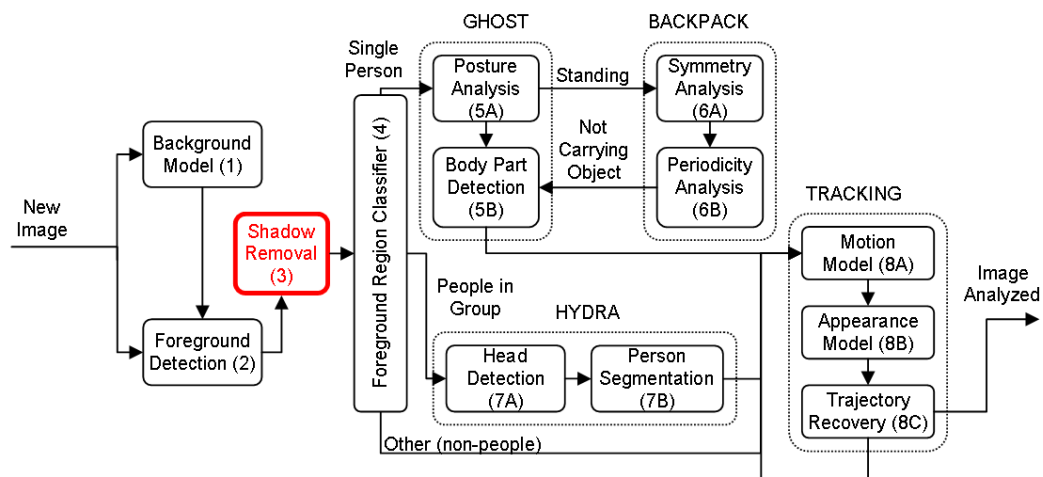


Figure F.1: The system architecture of W^4 [19] with additional shadow removal. Step 3 (red) indicates when the shadow removal should be performed. Similar to figure 2.1.

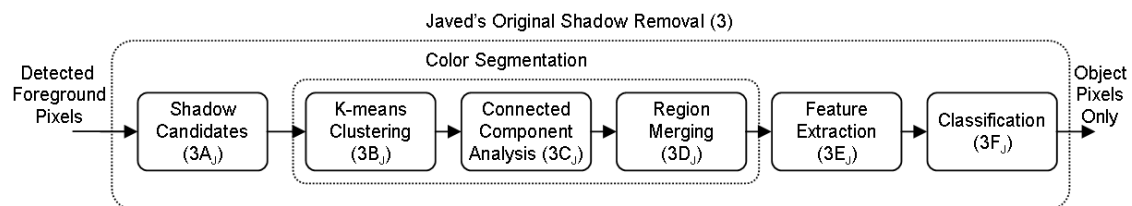


Figure F.2: Flowchart of shadow removal as suggested by Javed. Corresponds to step 3 in figure F.1. Similar to figure 2.2.

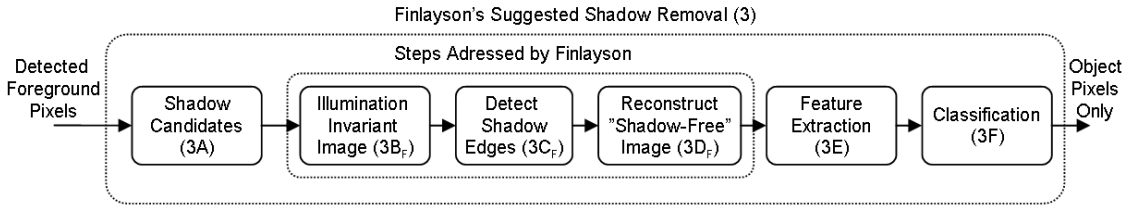


Figure F.3: Flowchart of shadow removal as suggested by Finlayson. Corresponds to step 3 in figure F.1. Similar to figure 2.4.

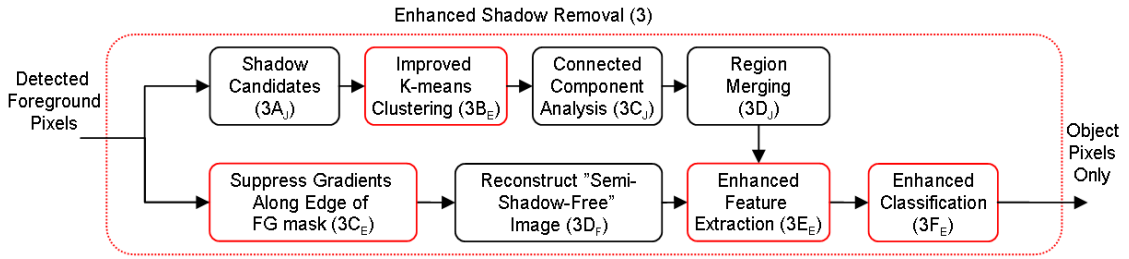


Figure F.4: Flowchart of enhanced shadow removal as suggested in this thesis. Corresponds to step 3 in figure F.1. Subscripts denote from where the original idea came: J=Javed, F=Finlayson and E=Enhanced steps suggested by the author (red). Similar to figure 5.13

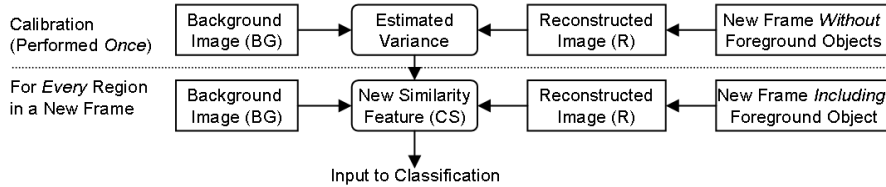


Figure F.5: Flowchart illustrating the enhanced similarity feature (CS). (Upper): Variance between background image and new frame without any foreground objects is estimated once. (Lower): In a new frame, including detected foreground objects, the enhanced similarity feature (CS) is computed for every region and is a part of step 3E_E of figure F.4. Similar to figure 5.14.

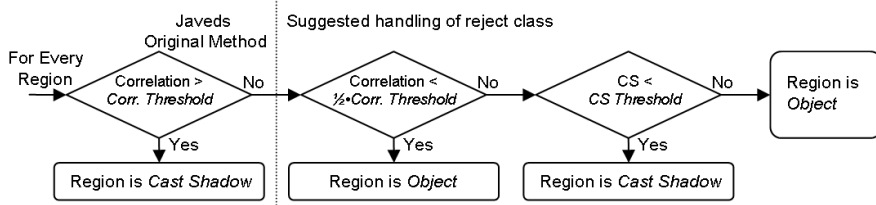


Figure F.6: Flowchart illustrating the enhanced classification of color regions (step 3F_E in figure F.4). The enhanced similarity feature, (CS), classifies all regions that the correlation feature assign to a reject class ($0.5 \cdot \text{Corr. threshold} < \text{Correlation} < \text{Corr. threshold} \Rightarrow \text{reject class}$). Similar to figure 5.15.