Road Scene Content Analysis for Driver Assistance and Autonomous Driving

A dissertation presented to

the faculty of

the Russ College of Engineering and Technology of Ohio University

In partial fulfillment

of the requirements for the degree

Doctor of Philosophy

Melih Altun

May 2015

©2015 Melih Altun. All Rights Reserved.

This dissertation titled

Road Scene Content Analysis for Driver Assistance and Autonomous Driving

by

MELIH ALTUN

has been approved for

the School of Electrical Engineering and Computer Science

and the Russ College of Engineering and Technology by

Mehmet Celenk

Professor of Electrical Engineering and Computer Science

Dennis Irwin

Dean, Russ College of Engineering and Technology

ABSTRACT

ALTUN, MELIH, Ph.D., May 2015, Electrical Engineering and Computer Science

Road Scene Content Analysis for Driver Assistance and Autonomous Driving

Director of Dissertation: Mehmet Celenk

This research aims to develop a vision based driver assistance system that achieves scene awareness using video frames obtained from a dashboard camera. A saliency image map is formed with features pertinent to the driving scene. This saliency map, based on contour and motion sensitive human visual perception, is devised by extracting spatial, spectral and temporal information from the input frames and applying data fusion. Fusion output contains high level descriptors for segment boundaries and non-stationary objects. Following the segmentation and foreground object detection stage, an adaptive Bayesian learning framework classifies road surface regions and the detected foreground objects are tracked via Kalman filtering. In turn, this oversees potential collisions with the tracked objects. Furthermore, the vehicle path is used in conjunction with the extracted road information to detect deviations from the road surfaces. The system forms an augmented reality output in which video frames are context enhanced with the object tracking and road surface information.

The proposed scene driven vision system improves the driver's situational awareness by enabling adaptive road surface classification, object tracking and collision estimation. As experimental results demonstrate, context aware low level to high level information fusion based on human vision model produces superior segmentation, tracking and classification results that lead to high level abstraction of driving scene.

ACKNOWLEDGEMENTS

This dissertation would not have been possible without the constant support and guidance of my advisor Dr. Mehmet Celenk. The knowledge, wisdom and motivation he granted me have been the biggest driving forces of this research. For that, I am deeply grateful. His mentorship has been a *sine qua non* for my doctoral study and I am honored to have him as my mentor.

I also would like to express my gratitude for my committee members: Dr. David Chelberg, Dr. Jeffrey Dill, Dr. Jundong Liu, Dr. Hans Kruse, Dr. Gursel Suer and Dr. Xiaoping Annie Shen. Their teaching, comments and contributions have been essential in shaping this research.

My thanks also go to other Ohio University professors who have been my instructors during my doctoral study and the administration of School of EECS. I feel privileged to be a part of Ohio University family.

Last but not least, I would like to thank my parents Ferda Altun and Ali Ihsan Altun who have never stopped supporting me unconditionally for my entire life and my sister Aysu Altun who has been the greatest source of joy in my life since the day she was born.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

1. INTRODUCTION

The first chapter presents the motivations, objectives and the significance of this dissertation research.

1.1 Making Intelligent Vehicles Scene Aware

Most people who grew up with the science fiction culture of 80's expected futuristic concepts such as self driving cars to become reality sometime in early 2000's. Although not seeing these cars around after three decades is a bit disappointing, intelligent vehicle research exhibits potential towards this goal. Over the years cars have become safer, faster, more fuel efficient, and somewhat intelligent with the new driver assistance systems. There is even some encouraging progress on autonomous vehicles. Yet these autonomous vehicle prototypes are far from being commercially available. In order to produce these intelligent vehicles that can drive themselves in virtually every situation, they need to be able to perceive their environment well, adapt to changing conditions and find general solutions to problems. This is how human intelligence behaves and in order for the machines to be good at human tasks, artificial intelligence (AI) must simulate this natural intelligence. Developing new methods to achieve this task is one of the important motivations behind this research. Of course, an intelligent vehicle that can easily pass the Turing test [1] is well beyond the scope of this research. Yet, Computer Vision (CV) and learning methods developed in this research can become the visual system, or in other words, the eyes of an intelligent vehicle.

The purpose of this research is to develop a scene aware driver assistance system by using CV methods with inputs obtained from a dashboard camera only and no other

remote sensing devices. The works in this field focus on specific problems such as road line and lane detection [2], vehicle detection and tracking [3], etc. So far not much work has been done on generating a fully structured description of the driving scene. This is mainly due to the difficulty of mimicking human perception with CV algorithms, which brings us back to developing better AI approaches issue.

The reason why AI methods for image understanding have not achieved significant developments over the past few decades is not the lack of proper methods. For almost any problem multiple methods have been developed, each with varying performances under different conditions. Take image segmentation for example. There are methods based on clustering [4] [5], texture [6], edges [7], etc. For video segmentation there are also methods that involve motion detection [8]. Given the right conditions, any one of these methods can outperform others. However, these conditions cannot be generalized. Therefore it is impossible to pick a prevalent solution. Yet, when there are multiple solutions to a problem AI researchers usually go with one of the possible methods and claim their method can beat other methods. Human intelligence works with a quite contrary manner though. Human mind considers all possible solutions to a problem and based on the conflicts and agreements of those solutions, it forms an answer. To see our environment and to identify objects, we do not pick a single source of information from color, edge, texture and motion. We form our perception by using all available information as a whole. This approach used by natural intelligence can be useful for AI methods as well.

In image understanding problems color, texture edges, etc. form the low-level features. Between the low level image features and high degree abstraction that describes

the scene contents there is usually a semantic gap [9]. Individual low level features are not sufficient to close this semantic gap. The focus of this research is to combine the results of multiple image processing methods in a methodical manner that will yield useful, high-level list of image contents. Simply put, the CV methods developed in this research focus on producing a general solution by combining solutions of multiple methods. It is the hope of the researchers that this work will form a substantial step towards identification of high-level semantic objects in a scene.

1.2 Driver Assistance Systems and Intelligent Vehicles

As of year 2014 there are many car models with driver assistance systems such as parking assistance, lane departure warnings, emergency brake assistance, adaptive cruise control, etc. [10]. Furthermore, autonomous car prototypes are being developed by Google [11] and many major automobile manufacturers. These vehicles contain several different sensors such as global positioning system (GPS), Radio detection and ranging (Radar), Light detection and ranging (LIDAR), etc. Figure 1.1 displays a drawing of Google car with the sensors on board. Moreover, these autonomous cars are being designed in such a way that they will be able to communicate with other autonomous cars in their vicinity and form sensor networks [12].

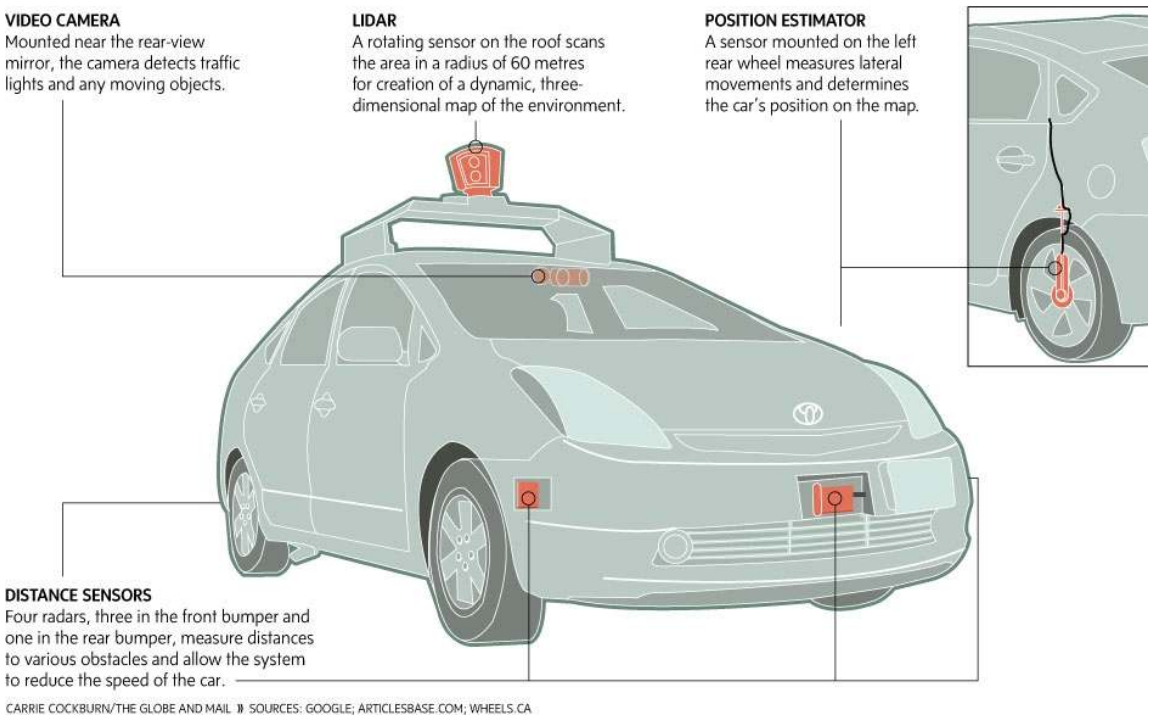

Figure 1.1: Google Car with multiple sensors [13]

Humans on the other hand, use their eyes to perceive their environment and to assess the driving scenery. They do not have lasers. They do not emit or receive radio waves and ultrasonic waves. They do not have an internal GPS. Additionally, they certainly do not have a telepathic link to nearby drivers. Yet, they can drive the cars by using visual information only. Therefore, the question naturally arises; Is it possible to come up with a system that sees its environment with a single dashboard camera instead of using all those sensors?

If a system connected to a camera can mimic a driver's vision, it should be sufficient for intelligent vehicles and driver assistance systems. This research aims to develop a system that depends only on visual data flow and no other sensory information such as Radar or LIDAR. The system will be able to identify road surface and

background objects as well as the foreground objects. The segmentation of these objects is enabled by combining low level spatial, spectral and temporal data obtained from camera video stream into higher level descriptors. Segmented regions are identified by classifying them into learned categories. The system detects and tracks the foreground objects just like a human driver following the motion and position of other nearby cars. Learning and classification are performed by Bayesian inference methods and tracking is performed by Kalman filtering. The combined output of segmentation, learning and tracking will form a driving scene description which will model the environment in a fully structured way. Fig.1.2 demonstrates a simplified representation of the proposed system. A block diagram of the proposed system can be seen in Fig. 1.3. Context enhanced output image can be shown on a dashboard display, or on a head up display (HUD). These technologies can be seen in Fig. 1.4.

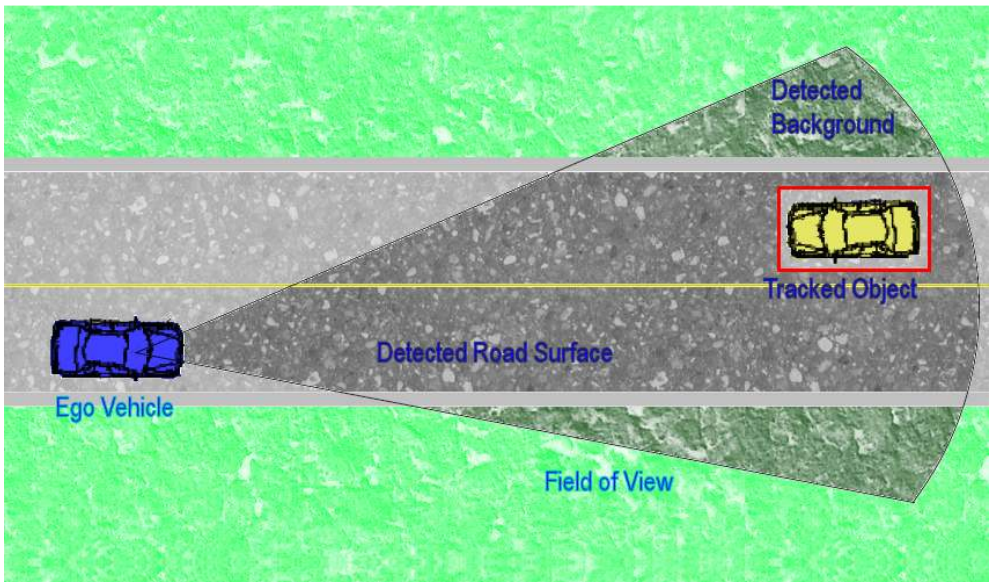

Figure 1.2: Simplified system representation

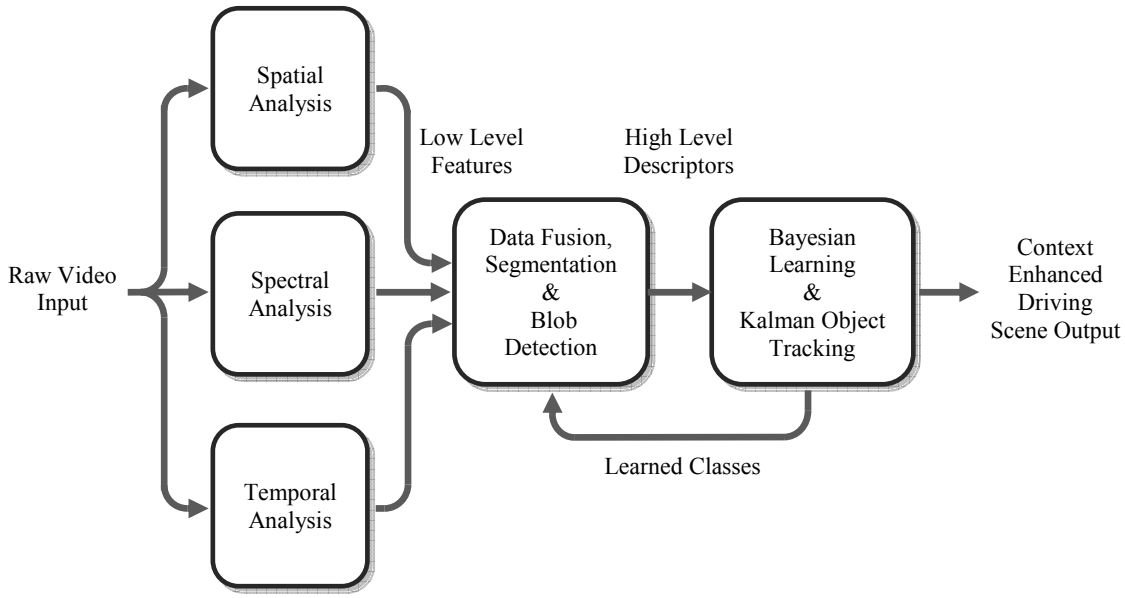

Figure 1.3: Overall diagram of the proposed system

1.3 Potential Benefits of Learning Road Patterns and Surrounding Environment with a Computer Vision application

The major contribution of this research is achieving scene segmentation. The task of combining spatial, spectral and temporal information is accomplished through information fusion. In order to bridge the "semantic gap" mentioned earlier, low level features are combined into high level descriptors. These descriptors are then used for segmentation and learning. This process produces very promising results where the semantic gap is closed – something that has not been achievable by use of low level features only. Combining multiple methods to obtain a better solution approach produces better features for segmentation, object tracking and region learning, resulting in better scene understanding.

Another contribution of this dissertation is the removal of the necessity for non visual sensory devices (such as LIDAR) on advanced driver assistance and automated driving systems. The proposed system only consists of a visual camera and a device with enough computational power to carry on computer vision operations. Utilization of such systems will simplify the implementation of intelligent cars and driver assistance systems. In addition, making driver assistance systems more affordable will enable more consumers to obtain vehicles with those assistance systems and potentially reduce the number of accidents.

Figure 1.4: Dashboard (left) and head up displays (right) [14]

2. BACKGROUND

This chapter provides brief descriptions of data fusion, computer vision and machine learning methods used in this research. Also a review of current research in driving assistance systems and intelligent vehicles is presented.

2.1 Data Fusion: Better Features to Learn

Machine learning (ML), is one of the most substantial branches of Artificial Intelligence (AI). It contains many algorithms for classification problems, including neural networks (NN), support vector machine (SVM) and expectance maximization (EM). ML also includes non parametric methods such as k-nearest neighbor (k-NN). Each one of these methods finds extensive use in practice. However, each has mostly been applied to the areas where it is most effective; so far there has been no method that works well in all applications. With so many widely used methods already in existence, a promising approach is to focus on the feature set, or, in plain words, what needs to be learned. Until now, the features used for learning and classification have been low level. Such features leave gaps in the *semantics* of classification; the higher conceptually positioned an object is, the harder it is to classify it with low level features. If we consider the image segmentation and understanding problems, as mentioned in the previous section, low level features are not capable of closing the "semantic gap". Consider the example of vehicle vision: at the lower end we have the raw video stream from a traveling car. Spectral, spatial and temporal information can be obtained from the video stream with common methods. However, the information is limited and inadequate for obtaining a meaningful scene description such as identification of the road, other cars and

background. In order to reach the expected high-level semantic description, the gap must be closed by information that contains high level concepts formed by combining low level information. They must belong to a conceptual set that make sense to humans. As an anecdote, our brains work in similar fashion. They find all potentially helpful features and combine them into a meaningful set, which then is used to identify actual objects.

In computer science terms, incorporating data from multiple sources into a single output is called information fusion or data fusion [15] [16]. The purpose of data fusion is to generate information that surpasses the individual sources used for fusion. When applied to images (i.e., image fusion) these methods find applications in military target detection and tracking [17], surveillance [18] [19], remote sensing [20], medical imaging [21], etc. Fusion needs a basis, which is a set of shared attributes used as a guide to merge the information content into one meaningful presentation. Among other methods [15], fusion can be based on entropy, fuzzy logic, wavelets and principal component analysis results. Image fusion involves images obtained from two or more different sources. For example: cameras operating on different spectral bands (i.e. visual camera, infrared camera, ultraviolet camera, etc.). Fig. 2.1 demonstrates how fusing images in multiple spectral bands can be useful in low light conditions.

In this study fusion operation is carried out with images obtained from a single camera. Each video frame from the camera goes through three different analysis procedures. These procedures are spatial, spectral and temporal analysis. These analysis procedures produce various low level image features such as color clustering, local variance, texture energy and optical flow. In order to obtain an image with higher information content, these low level features are combined in such a way that the images

with higher information contents obtain higher weights. The level of information that a low level image contains is measured by its entropy. Therefore, the fusion operation proposed in this research is entropy based adaptive fusion. The output of this fusion operation is a saliency image that shows the important parts of the input image. These are the parts that a human observer will pay attention to. This saliency image will be discussed in more detail in the next section. The saliency image enables the detection and tracking of objects in the scene. Saliency image and the detected objects form the higher level descriptors that are required to close the semantic gap. Figure 2.2a depicts this semantic gap and Fig. 2.2b shows how this gap is closed with higher level descriptors.

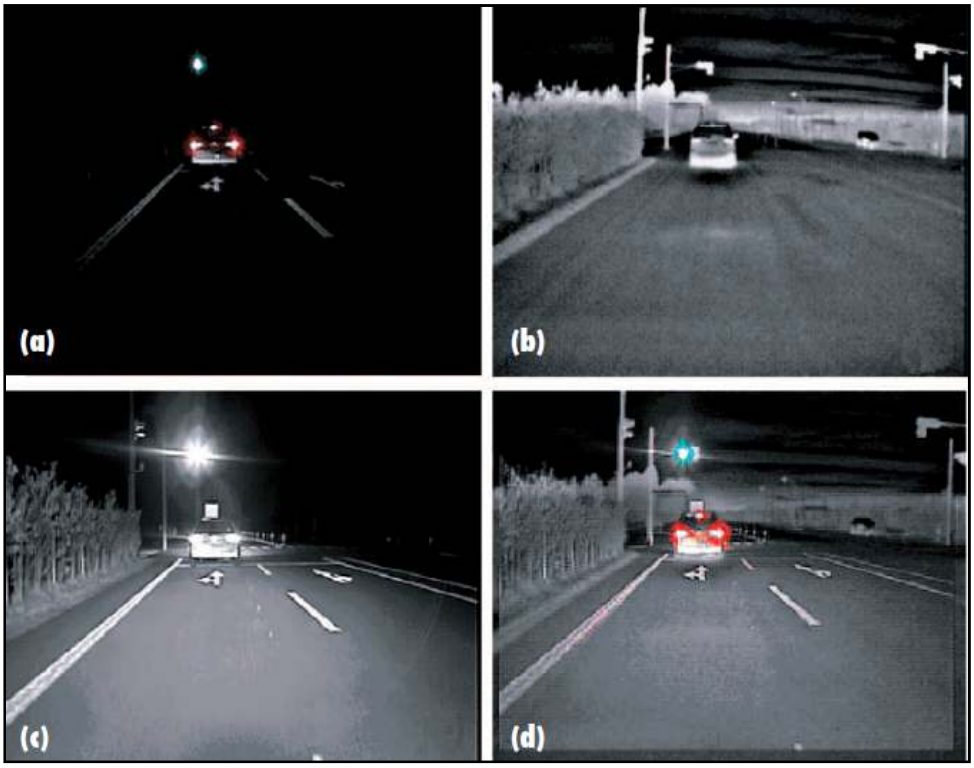

Figure 2.1: Visual (a), long wave IR (b), near IR (c) night time driving images and wavelet based fusion (d) [10].

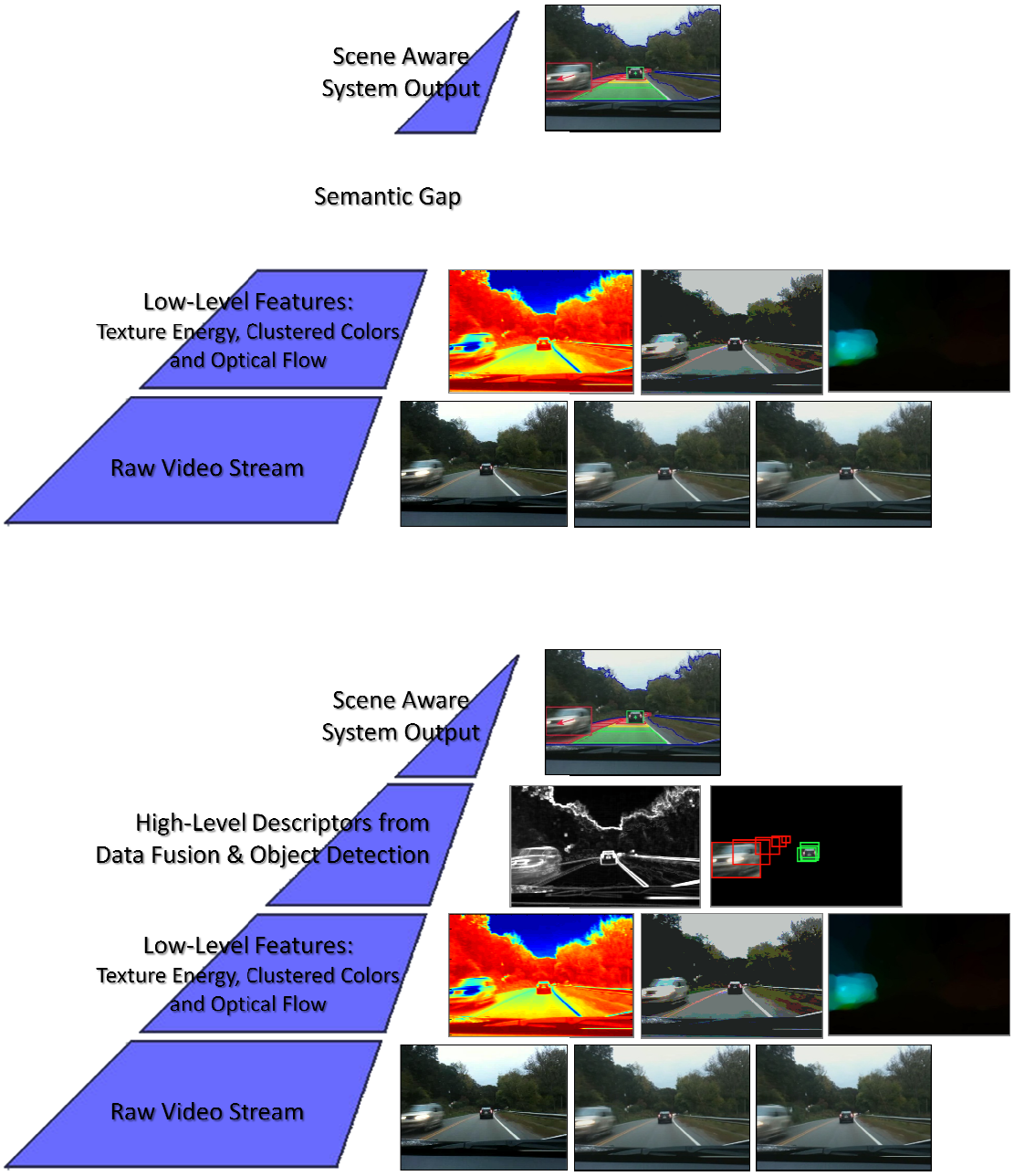

Figure 2.2: Bridging the semantic gap with data fusion and object detection

Now let's investigate how this high level information containing saliency image can be formed from low level features. From a theoretical point of view, ideal information fusion methods find and utilize more informative sources. If an image or, part of an image is more informative than the others, it will be given more weight compared to the other sources while forming the fused output. Hence, the challenge in information fusion is to measure the information content of a source. A good starting point is to use Shannon's entropy. Just like the entropy in thermodynamics, entropy used in information theory is a measure of chaos and randomness. Shannon's entropy is the most common entropy measure in information theory. It is given by:

$$H = \sum_i p_i \log_2 \frac{1}{p_i} \tag{2.1}$$

where p_i is the probability of selected data level.

High entropy means high randomness. The higher the randomness of a data stream, the less likely it is to have useful information. This is a very simple approach which, as shown later, does not always hold. However following this approach may lead to better ones.

The fused image must have more contribution from those features to maximize its information content. The fusion operation must look like:

$$I_f = \sum_i f(H_i)\, I_i \tag{2.2}$$

Here I_f is the fused image, I_i is an image representing one of the feature maps (spatial, spectral and temporal) and $f(H_i)$ forms a coefficient as a function of feature map entropy.

A more generalized version of eqn. (2.2) would be

$$I_f = \sum_i \alpha_i I_i \tag{2.3}$$

where $f(H_i)$ is replaced by the coefficient α. Probability distribution of fused image is given by its normalized histogram

$$p(I_f) = p(I_f(x,y)) = Hist(I_f(x,y)) \tag{2.4}$$

The entropy of this distribution is given by Shannon's formula:

$$H(I_f) = \sum p(I_f)\, log_2 \frac{1}{p(I_f)} \tag{2.5}$$

$$H(I_f) = \sum p(\sum_i \alpha_i I_i)\, log_2 [p(\sum_i \alpha_i I_i)]^{-1} \tag{2.6}$$

We want to find the optimum α vector that maximizes $H(I_f)$. We expect optimum α to give us the I_f that contains the most information. Therefore, we need to find the partial derivative of $H(I_f)$ w.r.t α and set it to zero.

$$\frac{\partial}{\partial \alpha} H(I_f) = 0 \tag{2.7}$$

This will give us the critical points. Using second derivatives we can find the minimums among these critical points and pick the global minimum.

$$\frac{\partial}{\partial \alpha} H(I_f) = \frac{\partial}{\partial \alpha} \sum p(I_f)\, log_2 \frac{1}{p(I_f)} = \sum \frac{\partial}{\partial \alpha} p(I_f)\, log_2 \frac{1}{p(I_f)} = 0 \tag{2.8}$$

$$\Longrightarrow \frac{1}{ln2} \sum p'(I_f) \left[ln \left(p(I_f) \right) + 1 \right] = 0 \tag{2.9}$$

where $p'(I_f)$ is the partial derivative of $p(I_f)$ w.r.t. α and *ln* is the natural logarithm.

Surely we need to add constraints to this optimization problem and solve the problem subject to these constraints.

$$\frac{\partial}{\partial\alpha}H(I_f) + \lambda_1\frac{\partial}{\partial\alpha}c_1(\alpha) + \lambda_2\frac{\partial}{\partial\alpha}c_2(\alpha) + \cdots = 0 \tag{2.10}$$

where c_i is a constraint and λ_i is a Lagrange multiplier.

The constraints need to be defined. We know sum of normalized histogram of the fused image must equal one:

$$\sum_k p(I_f) = 1 \tag{2.11}$$

Here *k* is the range of intensity values. Same goes for the input features:

$$\sum_k p(I_i) = 1 \tag{2.12}$$

There are corner cases such as a blank input image. If there is an input with zero entropy (i.e. $(I_X) = 0$) its contribution will dominate:

$$\alpha_X = 1 \tag{2.13}$$

And

$$\alpha_i = 0 \quad \forall i \neq X \tag{2.14}$$

That means we have to add the constraint:

$$H(I_i) > 0 \; \forall i \tag{2.15}$$

What about non-zero inputs with very little information in them, such as images with only a few non-zero pixels or periodic images? These will generate low entropy values and a similar condition may arise. We need to change the previous constraint to:

$$H(I_i) > \varepsilon \; \forall i \tag{2.16}$$

How can we define a lower limit to entropy? What is the entropy value boundary between meaningful information and lack of information? Picking an arbitrary limit will surely lead to controversial results. Clearly, a better approach is needed. Entropy of a single source is not sufficient to measure information content relative to other sources.

In order for a probabilistic data to be informative, we need circumstances (i.e., joint or conditional probabilities with other variables) that set the background. A thought experiment can be done to follow this idea. Assume we are in January and we consider the chance of snowfall.

$$P(snow\ in\ January) = \; x \tag{2.17}$$

The entropy of a single event is given by:

$$H(x) = \; log_2 \frac{1}{P(snow\ in\ January)} \tag{2.18}$$

Any number we use to replace x will not represent much information by itself because we don't know anything about location or any other relevant information. Now let's say we are in Athens, Ohio:

$$P(snow\ in\ January \mid Athens, OH\ ) = \; p1 \tag{2.19}$$

We know some winter days in Athens are snowy. So given that condition, $p1$ will have some information value. Now consider another example:

$$P(snow\ in\ January \mid Anchorage, AK) = p2 \quad (2.20)$$

This time we know the probability will be very close to one and it will contain little or no information:

$$H = log_2 1/1 = 0 \quad (2.21)$$

In fact, snowfall in Alaska during winter is hardly a surprise for anybody. Now consider this example:

$$P(snow\ in\ January \mid Los\ Angeles, CA) = p3 \quad (2.22)$$

p3 will be very close to zero given the climate of Southern California. If meteorological findings indicate a real chance of snow, the amount of information is enormous. This leads to the theory that, a good information measure should use background knowledge and frequency of events together.

As mentioned before, simple entropy is not sufficient to quantify information content. However, with the new found hindsight, conditional entropy or mutual information can provide a better information measure. A similar concept was studied in [22]. The entropy based information fusion approach in [22] starts with the entropy of sources rather than the entropy of the fused output.

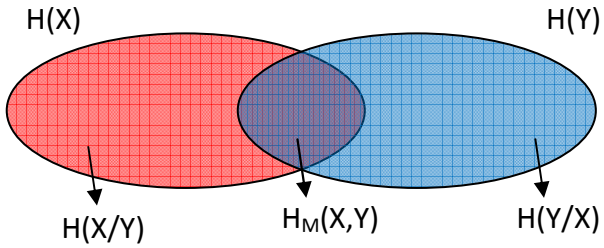

Figure 2.3: Illustration of conditional entropy and mutual information

The inputs X and Y can have common information and information exclusive to one of them (i.e., mutual information and conditional entropy). Starting with two sources if we want to emphasize the redundant information in our fused output we must maximize mutual information: $H_M(X,Y)$. If we want to emphasize information specific to individual sources we need to maximize conditional entropies: $H(X/Y)$ U $H(Y/X)$ where U is the union operator and / is the set complement operator.

This suggests that mutual information between images that represent different features must be maximized to find the best combination of those images.

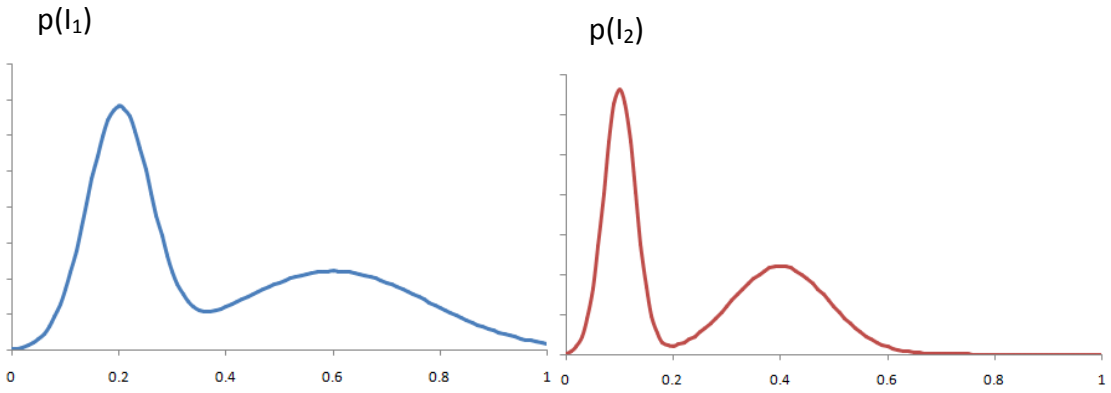

Figure 2.4: Two sample histograms.

Assume we start with two inputs with the histograms shown in Fig. 2.4. The fused image will be combination of the two:

$$I_f = \alpha I_1 + \beta I_2 \tag{2.23}$$

Finding optimal I_f boils down to finding the ratio of α and β coefficients for I_1 and I_2. Since the output is going to be normalized this can be further reduced to:

$$I_f = \alpha I_1 + I_2 \tag{2.24}$$

Next we need to select which of the two inputs is going to be scaled. Mutual information is going to be maximum when the two images are most similar, which also means when their histograms are most similar.

I_1 has more variance compared to I_2. Therefore, we can pick an $\alpha<1$ value and use it to scale I_1 and squeeze its histogram or we can pick $\alpha>1$ and use it to stretch the histogram of I_2. If we use $\alpha>1$ then some of the pixels in I_2 will be scaled beyond the maximum brightness level allowed. We are going to either lose those pixels or limit them to max value (1 or 255, etc). The histogram modifier functions will look like one of the cases below.

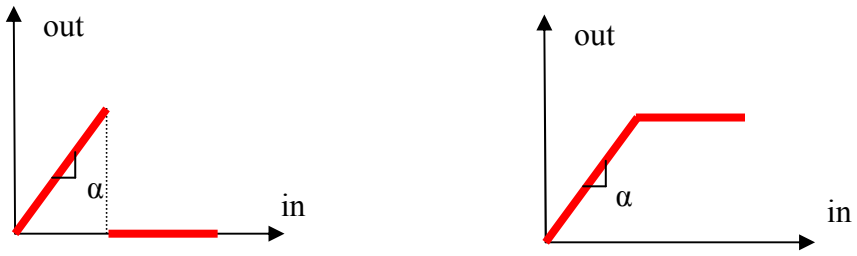

Figure 2.5: Histogram modifier functions with $\alpha>1$

Either case will introduce non-linearity and result in information loss. Thus, we should select $\alpha<1$ and scale I_1 (the one with the larger variance) to squeeze its histogram and make it look like I_2.

Our aim is to maximize mutual information $H_M(\alpha I_1, I_2)$ with respect to α.

$$\frac{\partial}{\partial \alpha} H_M(\alpha I_1, I_2) = 0 \tag{2.25}$$

$$\frac{\partial}{\partial \alpha}\sum_{i_1}\sum_{i_2} p(\alpha I_1, I_2) log_2 \frac{p(\alpha I_1, I_2)}{p(\alpha I_1)p(I_2)} = 0 \quad (2.26)$$

where $p(\alpha\ I_1,\ I_2\ )$ is the joint histogram of two images. If we follow the math from here we will see that finding a 2D distribution as a function of I_1 and I_2 without loss of generality is quite difficult. Moreover, as eqn. (2.28) shows multivariate mutual information calculation is complicated.

$$H_M(I_1, I_2, \dots, I_N) = H_M(I_1, I_2, \dots, I_{N-1}) - H_M(I_1, I_2, \dots, I_{N-1} |\ I_N) \quad (2.27)$$

To eliminate such complications mean square error (MSE) between feature maps can be utilized. [23] states that there is a differential relation between Mutual information and minimum MSE. [23] investigates input and output of transmission channels with Gaussian noise. Although it investigates a different topic, there are quite a lot of similarities and they basically claim Mutual information and MSE are very relevant. They derive the equations below which define the relation between MSE and Mutual Information.

$$\frac{d}{d\ snr} H_M(X, Y) = \frac{1}{2} MSE(X|Y) \quad (2.28)$$

$$H_M(X, Y) = \frac{1}{2}\int_0^{snr} MSE(\gamma) d\gamma \quad (2.29)$$

What is claimed in this research is, for a specific scaling factor, the difference between two images to be fused will be minimum. Hence, the redundancy between images will be maximized.

$$I_2 = \alpha^* I_1 + Error \quad (2.30)$$

The approach in [23] is, given an input X a communication channel and an output Y, output will be some noise plus X scaled by signal to noise ratio (snr).

$$Y = \sqrt{snr}\, X + \mathcal{N} \tag{2.31}$$

The two approaches are parallel in this aspect. Therefore minimizing MSE between feature map histograms can generate similar results to mutual information approach.

$$MSE = \frac{1}{T}\int_{k=0}^{1}[p_{\alpha I1}(k) - p_{I2}(k)]^2 dk \tag{2.32}$$

We can model probability distributions as Gaussian mixtures and T=1 since histograms are normalized:

$$= \int_{k=0}^{1}\left[\sum_i c_i\, \mathcal{N}(k|\alpha\mu_i, \alpha\sigma_i) - \sum_j d_j\, \mathcal{N}\left(k\middle|\mu_j, \sigma_j\right)\right]^2 dk \tag{2.33}$$

Now minimize MSE w.r.t α, subject to $\sum_i c_i = 1$ and $\sum_j d_j = 1$

$$\frac{\partial}{\partial\alpha}\int_{k=0}^{1}\left[\sum_i c_i\, \mathcal{N}(k|\alpha\mu_i, \alpha\sigma_i) - \sum_j d_j\, \mathcal{N}\left(k\middle|\mu_j, \sigma_j\right)\right]^2 dk - \lambda_1\frac{\partial}{\partial\alpha}\left(\sum_i c_i - 1\right) - \lambda_2\frac{\partial}{\partial\alpha}\left(\sum_j d_j - 1\right) = 0 \tag{2.34}$$

Constraints are not dependent on alpha so they drop out. Note also that the second Gaussian mixture is not dependent on alpha as well.

$$= \int_{k=0}^{1}\frac{\partial}{\partial\alpha}\left[\sum_i c_i\, \mathcal{N}(k|\alpha\mu_i, \alpha\sigma_i) - \sum_j d_j\, \mathcal{N}\left(k\middle|\mu_j, \sigma_j\right)\right]^2 dk = 0 \tag{2.35}$$

$$\int_{k=0}^{1}\frac{\partial}{\partial\alpha}\left[G_1(k,\alpha) - G_2(k)\right]^2 dk = \int_{k=0}^{1} 2\left[G_1(k,\alpha) - G_2(k)\right]\frac{\partial}{\partial\alpha}G_1(k,\alpha)\, dk = 0 \tag{2.36}$$

This suggests the trivial solution:

$$[G_1(k,\alpha) - G_2(k)] = 0 \tag{2.37}$$

Alternatively, considering G_1 is a normalized probability distribution, hence, $G_1(k) \geq 0$ for $k \in [0,1]$ and as α increases, stretching histogram values will monotonically decrease, generalized mean value theorem for integration can be applied [24].

$$2\,[G_1(c,\alpha) - G_2(c)]\int_{k=0}^{1}\frac{\partial}{\partial\alpha}G_1(k,\alpha)\, dk = 0 \tag{2.38}$$

where c is a constant within the interval [0,1]. The term outside the integration operation is a constant and can be omitted:

$$\int_{k=0}^{1} \frac{\partial}{\partial \alpha} G_1(k,\alpha)\, dk = 0 \tag{2.39}$$

Eqn. (2.37) suggests G_1 and G_2 are identical, which is highly unlikely. Solving eqn. (2.39) is the more likely to produce results. Eqn. (2.39) is sufficient for Matlab evaluation. However, this can be further evaluated:

$$\int_{k=0}^{1} \frac{\partial}{\partial \alpha} G_1(k,\alpha)\, dk = \int_{k=0}^{1} \frac{\partial}{\partial \alpha} \sum_i c_i \frac{1}{\alpha\, \sigma_i \sqrt{2\pi}} \exp\left[-0.5 \left(\frac{k-\alpha\, \mu_i}{i}\right)^2 \right] dk = 0 \tag{2.40}$$

$$\int_{k=0}^{1} \sum_i c_i \frac{-1}{\alpha^4 {\sigma_i}^3 \sqrt{2\pi}} \exp\left[-0.5 \left(\frac{k-\alpha\, \mu_i}{i}\right)^2 \right] (\alpha^2 {\sigma_i}^2 + \alpha\, k\, \mu_i - k^2) dk = 0 \tag{2.41}$$

The first part is a scaled Gaussian and due to the negative sign it is negative, i.e.,

$$\frac{-c_i}{\alpha^4 {\sigma_i}^3 \sqrt{2\pi}} \exp\left[-0.5 \left(\frac{k-\alpha\, \mu_i}{i}\right)^2 \right] < 0 \tag{2.42}$$

which leads to:

$$\int_{k=0}^{1} \sum_i (\alpha^2 {\sigma_i}^2 + \alpha\, k\, \mu_i - k^2)\, dk = 0 \tag{2.43}$$

This suggests, alternative to Eqn. (2.38). Gaussians can be fitted to histogram data and α can be directly calculated. For a single Gaussian α evaluates to:

$$\alpha = \frac{\mu\, k \pm \sqrt{\mu^2 + 4\sigma^2}}{2\, \sigma} \tag{2.44}$$

The histograms of the features we extracted are indeed uni-modal almost all the time. Hence, by looking at the mean and variance, α can be computed.

Setting partial derivative w.r.t α to zero will give us the critical points. We need to select which of these critical points are minimizers of the MSE. The definition of a local minimum is given as:

$$f(\alpha^*) \leq f(\alpha),\ \forall \alpha\, \epsilon\, (\alpha^* - \varepsilon, \alpha^* + \varepsilon) \tag{2.45}$$

Given the interval $(\alpha^* - \varepsilon, \alpha^* + \varepsilon)$ is continuously differentiable up to second order,

$$f_\alpha(\alpha^*) = 0 \tag{2.46}$$

indicates α^* is a critical point, and

$$f_{\alpha\alpha}(\alpha^*) > 0 \tag{2.47}$$

indicates α^* is a local minimum. But, given these conditions we do not need to take the second derivative. As Fig. 2.6 demonstrates if the first derivative is continuous, the second condition tells us a local minimum will be a negative to positive zero crossing at α^* location.

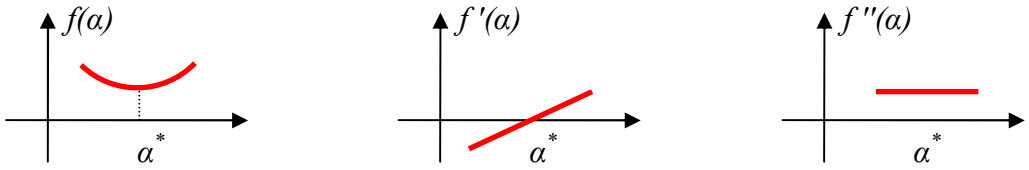

Figure 2.6: Finding local minimums with derivatives

With this MSE based fusion method, relative information content of two sources can be found. By finding the coefficient that minimizes the error between two feature sets using Eqn. (2.39) and finding a negative to positive zero crossing, information common to both sources can be emphasized. The following chapters will demonstrate how this can be extended to multiple sources of information.

2.2 Mimicking Human Vision with Computer Vision

To come up with a computer vision method that mimics human vision we first have to have an understanding of how human vision works. By finding out important

steps in our perception, we can find reasonable features that will be essential in defining scene contents and closing the semantic gap mentioned earlier.

The light receptors in the human eye are located at the retina, which forms the inner surface of the eye. The receptor cells around fovea, where lens focuses light rays, are mostly cone cells that are sensitive to color. Rod cells are in mainly concentrated around the periphery of fovea. These receptor cells are more sensitive to light intensity. The image that forms on the fovea is the center of our gaze. This is where most of the visual information comes from. The emphasis on this region can be understood by examining the connections of receptor cells and neurons. In this central region most receptor neuron connections are one to one where as in the peripheral vision regions many receptors are connected to one neuron [25]. It is not hard to guess that our attention goes to where we directly look.

Our eyes automatically cluster similar colors and intensities. Neurons connected to receptors that receive similar hue and intensity levels fire at the same rate. Therefore, perception begins at the receptor layer by separation of these spatial regions. But what captures our attention most are the transitions in color and intensity. These regions with high contrast form the region boundaries, or contours of an object. The importance of contours in human perception can be seen easily. When people are asked to draw an object, whether these people are toddlers or professional painters, they start by drawing the contours of an object. The phenomenon was first observed by Ernst Mach. According to his theory the perceived image is the actual image minus the Laplacian of the image multiplied with a coefficient, as shown in Eqn 2.48. This has a sharpening effect on the

perceived image [26]. In fact, finding zero crossings of an image Laplacian is an effective edge detection method in image processing.

$$I_{perceived}(x,y) = I(x,y) - c\nabla^2 I(x,y) \tag{2.48}$$

On the retina layer the effect of this is, neurons near the positive side of a transition get exhibited and fire more intensely, whereas the neurons on the negative side of the transition get inhibited. This phenomenon is now known as Mach band and it forms the basis of contour vision Fig. 2.7 demonstrates this effect in 1D.

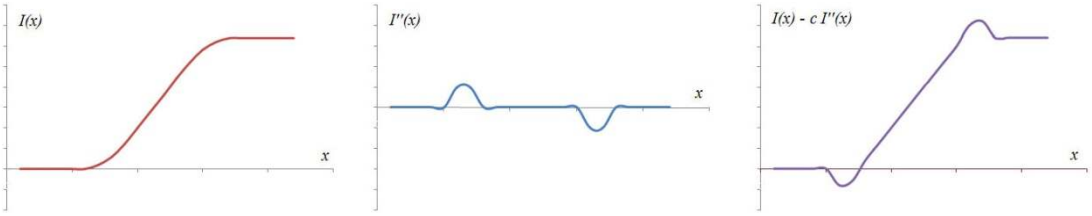

Figure 2.7: Illustration of sharpened transitions due to Mach effect

Humans also detect textures and texture changes easily. One of the most significant works on texture segmentation [6] lists texture components as periodicity, directionality and randomness. When our eyes detect small changes in contrast in periodic, directional or random manner, we recognize that as a texture. Over a large region these contrast changes add up and become easy to recognize. Although texture properties are spectral rather than spatial our perception clusters textures same way it clusters color and intensity, and finds boundaries between different textures. Boundaries of textures are added to our contour vision.

Another observational variable that is emphasized in human vision is motion. There are tracking neurons in visual system that fire upon detection of motion. The faster the observed object goes the more exhibited these neurons become [26]. This explains why moving objects immediately catch our attention.

Detected contours from colors, intensity and texture, plus the motion get synthesized and become our visual stimulus. The combined output of these visual features forms a map showing where most of our attention goes in a scene. Continuous long curves in this map depict the boundaries of regions, whereas blobs with high concentration of contours or motion define foreground objects. Therefore, at the retinal receptor layer, not only segmentation is done but also object-ground distinction is made. However, recognition of detected objects does not occur until observed image is transmitted to the visual cortex. The brain completes the last step of perception by carrying out recognition of known objects and classification of known backgrounds [25]. When this mental construction step is completed visual perception turns into cognition and we understand what we see.

With computer vision methods, similar to human eye, spatial (color, intensity) and spectral (texture) information can be extracted. Moreover, motion detection algorithms such as optical flow can find temporal changes just like the tracker neurons in our eyes. When these data are combined using the image fusion methods mentioned in previous section, the result will be a saliency map showing object and region contours as well as objects in motion. This saliency map models the visual stimulus after synthesis of spatial, spectral and temporal information. From this saliency map, objects of interest can be extracted by blob detection algorithms such as Difference of Gaussians (DoG) or

Laplacian of Gaussians (LoG). Also, by using the contours in the saliency map, regions in an image can be segmented. Detected blobs and segments form the higher order features that can be used by learning and classification algorithms.

Just like the optic nerves transferring clustered regions and object ground information to the visual cortex, the resulting segments are sent to classification algorithms which are trained to find certain regions. Detected objects can be processed by object recognition and tracking algorithms to have a better understanding of the scene. These last steps simulate the mental construction in the visual cortex and complete the scene modeling. Fig. 2.8 compares visual perception model described here with the proposed system.

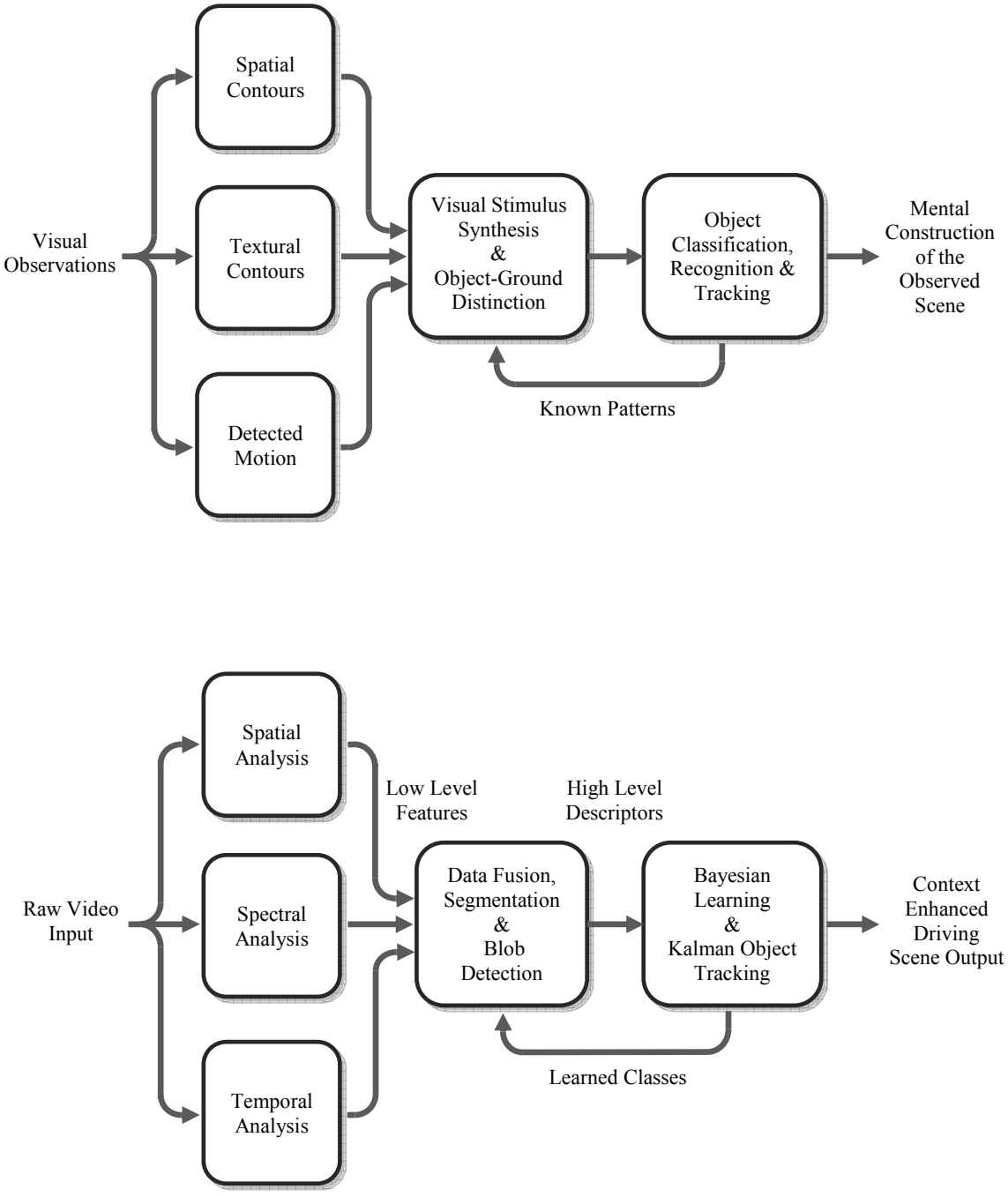

Figure 2.8: Revisit of overall system diagram with comparison to visual model

2.3 Learning from Observations

Once high level features are obtained through information fusion, these features can be learned to categorize objects. Many machine learning methods have been developed for various classification problems. Among these, Bayesian-based inference algorithms have become mainstream in ML research due to their practical applicability. Besides they are the more compatible human learning than any other learning [27]. Conditioning is an important part of human learning. Our beliefs are based on available information and they can change in the presence of new information. Strong evidence against our thoughts can easily change our way of thinking. Bayes' decision theory [28] formalizes this with conditional probabilities.

Bayesian learning is good model for general human learning and human visual system is no exception for that. Studies indicate the way the visual perception works can be modeled by Bayesian inference [29] [30]. Bayesian learning provides a good rationalization for mental construction in visual cortex.

Another important notion presented in [31] is Bayesian learning methods can work without the presence of negative examples. This is not the case for most other ML methods such as neural networks, support vector machines (SVM), nearest neighbor algorithms etc. Consider the two feature, two class sample dataset in Fig. 2.9. A dataset like this can be learned by SVMs and perceptrons. However, as can be inferred from Fig 2.10a, an SVM method needs to choose support vectors from both classes to find the maximum margin separation. A perceptron needs inputs from instances of both classes in order to update its decision boundary.

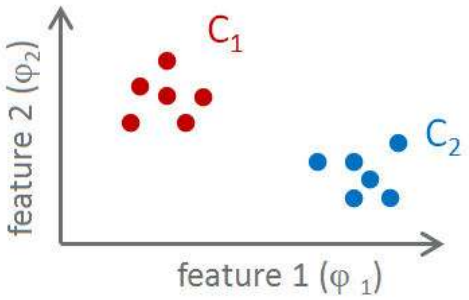

Figure 2.9: A sample two class data set in a 2D feature space

When a Bayesian method such as Maximum Likelihood is utilized learning can be achieved by finding models that fit classes. These models can be based on statistical properties such as mean and variance. Fig 2.11 depicts maximum likelihood learning with class probability distributions formed by training data. Contrary to other learning methods Bayesian learning can be applied to a single class (i.e. no negative training instances) as shown in Fig. 2.12.

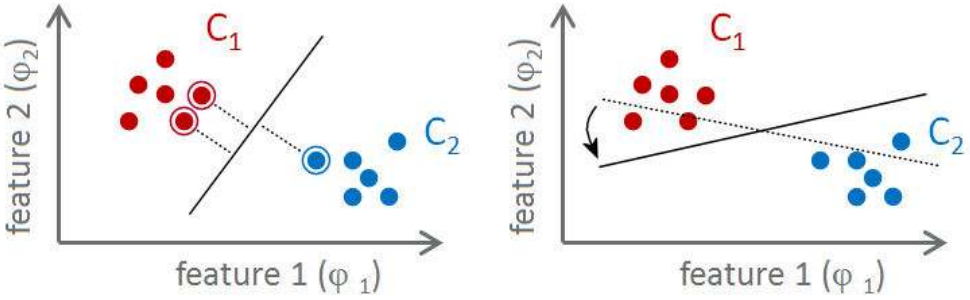

Figure 2.10: SVM learning (left) and perceptron learning (right)

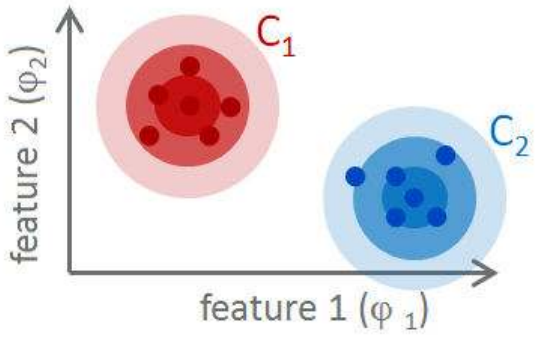

Figure 2.11: Illustration of maximum likelihood learning

In the multiclass case, when class models are formed, a test instance can be classified using the conditional probabilities. The decision based on conditional probabilities can be formulized as the eqns. (2.49a) and (2.49b). Fig. 2.13 illustrates this classification step.

$$P(C_1|x) > P(C_2|x),\ x \epsilon C_1 \tag{2.49a}$$

$$P(C_1|x) < P(C_2|x),\ x \epsilon C_2 \tag{2.49b}$$

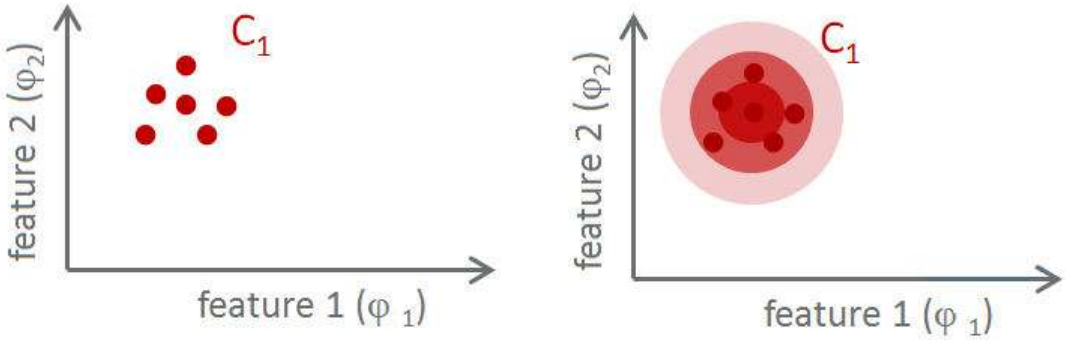

Figure 2.12: Maximum likelihood learning with a single class

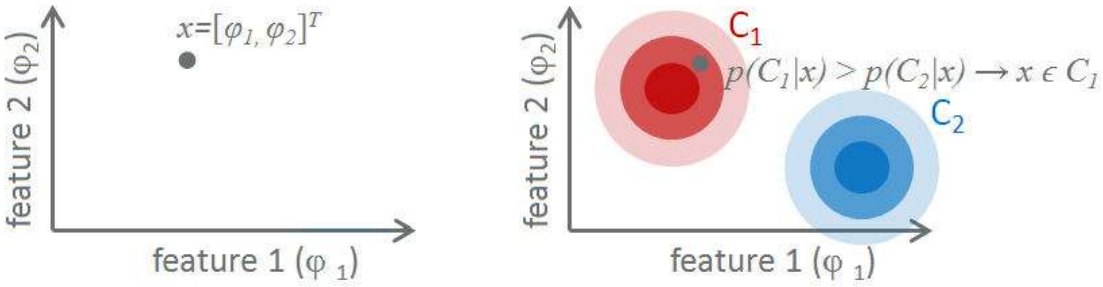

Figure 2.13: Maximum Likelihood classification with conditional probabilities

As discussed in the previous section, lower level image features can be utilized to segment the image and extract foreground objects. From the segmented image, important regions such as road surface can be learned and identified by using Bayesian learning methods. With proper features a maximum likelihood method should be able to find regions that are important for the driver. If there are multiple segments with common properties, these regions can be grown by merging them after being classified under the same class. Image region recognition and classification is a subject of object ontology [32] [33]. Ontological properties of segmented regions such as shape, position, color, size, etc. can be used as features for learning and classification.

Another important aspect of this research is the detection and tracking of foreground objects. Based on the past observations, a predicted path for these objects can be found and possible collisions can be detected. The obtained positions of these foreground objects in each video frame can be considered as observations. The actual positions and motions of these objects are their states. As the time progresses, these states change. Predicting the correct state of an object based on past observations can be achieved by learning.

When there are multiple states, defining state transitions by conditional probabilities and selecting a possible next state based on those probabilities is a direct application of Bayes decision theory. Systems such as the one shown in Fig. 2.14 are called Bayesian networks [27].

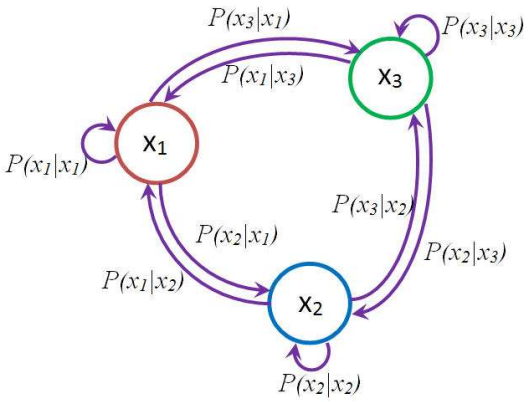

Figure 2.14: A sample Bayesian network depicting states and transition probabilities as a directed graph

Markov Chains and their variation Hidden Markov Models (HMM) are directly derived from Bayesian networks [28]. When we have a sequence of states their state transitions can be stated in terms of their probabilities. Consider the Bayesian system in Fig 2.14. The probability of observing a state sequence $\{x_1, x_3, x_3, x_2\}$ is given by the joint probability:

$$P(\{x_1, x_3, x_3, x_2\}) = P(x_1)P(x_3|x_1)P(x_3|x_3)P(x_2|x_3) \tag{2.50}$$

In fact state sequence of a Markov Chain can be formulized by:

$$P(X = \{x(t_1), \dots, x(t_N)\}) = P\big(x(t_1)\big) \prod_{i=1}^{N-1} P\big(x(t_{i+1})|x(t_i)\big) \tag{2.51}$$

In the case of HMM, states are not directly visible but there are possible observations associated with each state. Fig. 2.15 illustrates a Hidden Markov process where there are hidden states and observations.

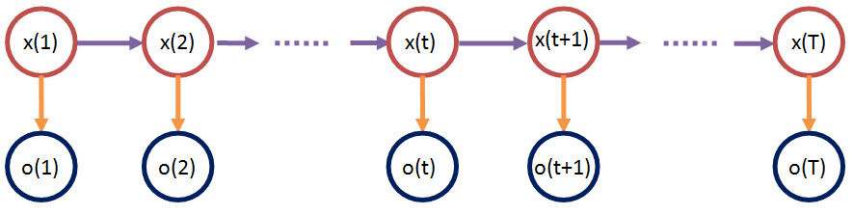

Figure 2.15: Hidden Markov Model as sequence of observations and hidden states.

HMMs are widely used for temporal pattern recognition. An HMM sequence can be formulated as:

$$P(O) = \sum_X P(O|X)P(X) \tag{2.52}$$

where the probability of getting a specific sequence of observations is defined by conditional probability of getting the specific observations given a specific state sequence, summed over all possible state sequences. Although the formulation is simple, calculations grow exponentially as the number of possible states and observations increase. Finding the most probable sequence of states is an even harder problem.

$$\hat{X} = argmax_X P(X|O, \varphi) \tag{2.53}$$

Here, X is the sequence of states, O is the sequence of observations and φ is the model that includes states transition probabilities and the probability of observations from

specific states. Viterbi algorithm [28] can find that optimal solution without too much computational complexity. However, for driving scenarios where the number of possible states and observations are practically infinite, these methods are not applicable. To find a Bayesian model that that fits our needs we need to explore the Markov process models.

Consider Fig 2.16. When a hidden variable is selected the next possible state is given by conditional variables. Possible observations are also given by conditional probabilities.

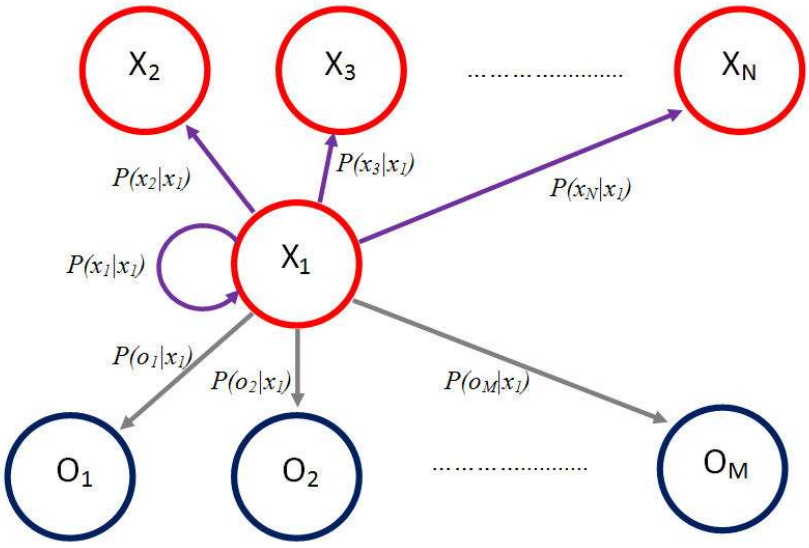

Figure 2.16: Possible observations and state transitions for a given state

By obtaining the same set of probabilities from each hidden state, a state transition and a measurement matrix can be formed as in Eqns. (2.54) and (2.55). Using these matrices, the observations and predicted state transitions can be found by a linear set of operations.

$$F = \begin{bmatrix} P(x_1|x_1) & .. & P(x_1|x_N) \\ P(x_2|x_1) & & \\ \vdots & & \vdots \\ P(x_N|x_1) & .. & P(x_N|x_N) \end{bmatrix}_{N\times N} \tag{2.54}$$

$$H = \begin{bmatrix} P(o_1|x_1) & .. & P(o_1|x_N) \\ P(o_2|x_1) & & \\ \vdots & & \vdots \\ P(o_M|x_1) & .. & P(o_M|x_N) \end{bmatrix}_{M\times N} \tag{2.55}$$

A state vector would look like:

$$X = \begin{bmatrix} P(x_1) \\ P(x_{12}) \\ \vdots \\ P(x_N) \end{bmatrix}_{N\times 1} \tag{2.56}$$

Observations and state predictions will be given by:

$$\hat{X}_{t+1} = F \cdot X_t \tag{2.57}$$

$$Z_t = H \cdot X_t \tag{2.58}$$

where $\hat{X}_{t+1}$ is the predicted next state and Z is the observation matrix.

If this system was deterministic instead of probabilistic, we would see whole numbers, rules and formulas in these matrices instead of probabilities. It is in fact possible to start with a deterministic system model and add uncertainty parameters that would diffuse the state transition and observation probabilities within themselves. Consider adding a process uncertainty parameter w to Eqn (2.57).

$$\hat{X}_{t+1} = F \cdot X_t + w \tag{2.59}$$

As opposed to prediction equation in (2.57), which returns a vector of probabilities for each state, deterministic model in (2.59) will return a single expected state. The extra uncertainty parameter, however, will lead to a covariance matrix *P* which estimates the accuracy of the prediction.

$$\hat{P}_{t+1} = F \cdot P_t \cdot F^T + Q \tag{2.60}$$

Here *Q* is the process noise covariance obtained from *w*.

Similar to prediction equation, observation in (2.48) will receive a measurement uncertainty parameter *v*.

$$Z_t = H \cdot X_t + v \tag{2.61}$$

The prediction error can be calculated using this observation:

$$Y_t = Z_t - H \cdot \hat{P}_{t+1} \tag{2.62}$$

An estimate of the accuracy of the observation can be calculated by finding the covariance matrix of the prediction error.

$$S_t = H \cdot \hat{P}_{t+1} \cdot H^T + R \tag{2.63}$$

Here R is the measurement uncertainty covariance obtained from *v*.

By now it should be obvious that this modified Markov model has become the prediction and error measurement steps of a Kalman filter. As stated in [34] Kalman filters are in fact related to HMMs and many other mixture models. Even though Kalman filters use a set o linear equations instead of a probabilistic state space, the underlying principles are the same and they are based on Bayesian decision theory.

Kalman filter provides an excellent model that has temporal tracking and estimation capabilities. Just like HMMs it is based on the Bayesian inference principals, which is compatible with the human cognition. Also, contrary to HMMs, it uses a set of linear equations to get the predicted output, instead of a searching a vast state space for an optimal solution.

2.4 Related Intelligent Vehicle Research

Intelligent vehicles research has started more than two decades ago. Although the first examples did not go beyond primitive methods for segmentation and lane detection [35] it has quickly grown in the last fifteen years. Although most of the research focuses on automobiles, [36] indicates some research activities also go around heavy trucks, busses and other forms of public transportation and also military vehicles. Majority of intelligent vehicle research in the literature is in line with the current autonomous vehicle design trends, which involve multiple sensors and sensor networks. Studies such as [37] [38] provide frameworks for combining a group of sensors such as LIDAR, radar, GPS, etc. Detection rate can benefit from using a variety of sensor but costs will go up as the number of built in sensors increases. These sensors usually work independent of each other and they may generate warnings and notifications at different times. [39] argues

some of these notifications may not carry much significance for the driver and proposes a sensor data fusion method that generates more informative warnings and notifications. Works such as [40] [41] focus on sensor networks and multi vehicle cooperative driving. Such approaches enable trajectory planning to avoid collisions. However, [10] argues that, with vehicles and sensors being produced by various different companies sensor interoperability will be a challenge for sensor networks.

On the other hand, vehicle detection methods that use vision only systems have also been presented in the past with varying performances. The system in [42] is a vehicle detection method depends on the presence of vehicle shadows and the symmetrical appearance of the vehicles. Symmetrical appearance can be a strong feature most of the time. However, this feature fails when vehicles are viewed from sideways and diagonal angles. Similarly the method proposed in [43] cannot detect objects with reflective surfaces. The methods in [44] [45] can detect and track objects, estimate collisions and give warnings. Both methods perform well in low speed urban environments but they cannot adapt to highway speeds. Lane detection and tracking systems have also [2] [46] been widely studied. However they are prone to failures when lane markers are occluded by other vehicles and they are useless on roads with no lines such as dirt and gravel roads.

Some vision based methods use edge features to detect vehicles. The proposed method in [47] uses edge based constraint filters to find cars. A recently published vehicle detection and collision warning system [3] uses vertical as well as horizontal edges and shadows under the vehicles as features. The method achieves an impressive detection ratio of 97%.

The proposed method in this dissertation also uses edges and car shadows as vehicle descriptor features. Unlike the other methods, the edges are obtained from combined contours from spatial and spectral features instead of simply obtaining the edge map of the grayscale input. Moreover, relative speed, which is a temporal feature, is also added to increase detection. Vehicles travelling at much lower velocities and vehicles going in the opposite direction will have significant relative speed and they are much more likely to be detected when this relative speed is utilized.

Our research aims to accomplish more than just detecting vehicles. We also identify the road surface in this research. A key ingredient is proper scene segmentation, which is achieved through feature fusion. Substantial research studies have been published on segmentation and scene understanding. Few of them focus on feature fusion and incorporation of temporal elements. For instance, [48] uses fusion of directional spatial features such as Eigen vectors, gradients and temporal saliency for vehicle detection from air. In [49] optical flow and a pixel based similarity metric are utilized to segment road scenes. [50] also uses optical flow for traffic analysis using a fixed camera directed at an intersection. Color clustering and local binary patterns (LBP) as a texture descriptor are employed for road scene segmentation in [51]. The method presented in [52] generates large feature vectors using motion boundary histograms (MBH) to classify road scenes. Some of these methods such as the latter case turn out to be inefficient for video processing due to the size of their feature space.

As mentioned in the previous sections, the method developed in this research utilizes spatial spectral and temporal information fusion for segmentation and object detection. Previous works explored combining pairs of these three information sources

such as spatial and spectral [51], spatial and temporal [48] or spectral and temporal [49]. There are also methods that combine texture and motion with color information which is a spatial feature [53] [54]. These methods use structural tensors to extract texture and motion information and attempt to segment the image based on feature vectors obtained for all pixels. These are parametric models and they require apriori information such as number of segments and class means. These requirements cause unsupervised clustering to be very difficult. In most cases they can only segment the image into two regions. Furthermore, none of the previously applied fusion methods are based on a human perception model. The fusion result of our work simulates the visual stimulus formed by the retinal receptor layer in order to enable better segmentation and scene understanding based on Bayesian inference.

The final output of this work is a context enhanced scene that displays the tracked objects, road surface, and the background by highlighting road and foreground objects. The output can be considered as an augmented reality since it contains information that was not readily available in the original video frames. There are some works focused on creating meaningful augmented reality scenes for driver assistance. [55] relies on classical lane detection techniques such as line direction and vanishing point detection to find the road surface. [56] uses a commercially available range finding device to indicate the direction of nearby obstacles with arrows. In this project, road surfaces are identified by Bayesian learning and classification. The foreground objects are not just found but also tracked to estimate their trajectory and give warnings in case of a possible collision risk.

The main reasons that set this research apart from previously published work are: using a combination of multiple low level features to obtain better scene descriptors and aiming complete scene understanding instead of focusing on a specific task such as vehicle detection and tracking.

3. EXTRACTION OF LOW LEVEL FEATURES

This chapter explains how spatial, spectral and temporal features are obtained from the video stream input. This step is in essence similar to the responses of nerves connected to the cone and rod receptors in the eye.

3.1 Extraction of Spatial Information

Spatial features are obtained by analyzing the video frames based on color and intensity.

3.1.1 Color Clustering and Finding Cluster Contours

The first step in obtaining spatial information is finding pixels with similar colors and clustering them. In order to group pixels of similar color into clusters a clustering algorithm needs to be applied. Although k-Means Clustering [27] is a commonly used method, selection of optimal number of clusters (k) requires multiple calls to this method eventually increasing computational workload. Instead a fast converging histogram based thresholding method is preferred in this work. The selected clustering method is a variation of work presented in [57]. With this method Otsu threshold is applied recursively. Given an image histogram Otsu's method [58] finds an optimal threshold and divides the image into a dark and a light class. The optimal threshold is selected in such a way that it minimizes the between class scatter value

$$S = \sum_{i=1}^{2} P_i(\mu_0 - \mu_i)^2 \tag{3.1}$$

where P_i is the class probability μ_0 is the global mean and μ_i is the class mean. Each time a histogram is separated into a lower and a higher histogram Otsu algorithm is called

again on these lower and higher histograms and the segment is divided into further smaller segments until one of the stop conditions occur. These conditions are:

1. Class probability becomes too small:

$$P_i < \theta_1 \tag{3.2}$$

2. Class means get too close:

$$|\mu_i - \mu_j| < \theta_2 \tag{3.3}$$

3. A class mean gets close to the class threshold

$$\mu_i - l_{i-1} < \theta_3 \quad or \quad l_{i+1} - \mu_i < \theta_3 \tag{3.4}$$

where θ_1,θ_2, θ_3 are thresholds, l_{i-1} is the intensity level that separates class C_i from C_{i-1} and l_{i+1} is the intensity level that separates C_i from C_{i+1}.

An important point here is the selection of color space where clustering will be applied. RGB color space is good and efficient for displaying colors on screens. However, works such as [4] has shown that RGB is not very linear in terms of representing distances between colors. Clustering with color spaces that use polar coordinates performs much better due to more accurate color coordinate representations. In this work HSV color space is chosen for color clustering. Recursive Otsu segmentation is applied to each channel of the HSV image. A sample frame clustered by this method can be seen below in Fig 3.1.

Figure 3.1: Original Frame and recursive Otsu color clustering result

As mentioned in the previous chapter what is important for human perception and also for the proposed visions system here is the contours of this clustered image. To find those contours an edge detection process is applied to the clustered image. The edge detection method in this context is the Sobel edge detector [25]. Sobel edge detector is the discrete approximation of magnitude of the image gradient given as:

$$|\nabla I(x.y)| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2} \tag{3.5}$$

The discrete approximations to partial derivatives are given by:

$$\frac{\partial I(x,y)}{\partial x} \cong I(x,y) * H_x(x,y) \tag{3.6}$$

$$\frac{\partial I(x,y)}{\partial y} \cong I(x,y) * H_y(x,y) \tag{3.7}$$

where * is the convolution operator and H_x and H_y are horizontal and vertical Sobel operators:

$$H_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad H_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{3.8}$$

When edge detection is applied to the color clustered image, contours of the clusters will be obtained. The resulting contour image of the color clustered sample frame is depicted in Fig 3.2.

Figure 3.2: Contours of the color clustered image

3.1.2 Finding Local Variance

Local variance is another important spatial feature. It is calculated as:

$$I_{local\_var}(x, y) = var(I_s(x, y)) \tag{3.9}$$

where I_s is a N by N subsample of *I(x,y)* around a selected pixel. Local variance is applied to grayscale images in order to find the contours of different intensities.

Local variance is a strong contour descriptor. In regions where image intensity chances abruptly, local variance will be much higher compared to uniform regions. Although local variance is listed as a spatial feature, it also contains some texture

information. In uniform image regions local variance will be close to zero. In regions with texture local variance will be higher due to the variations in pixel intensities. In that sense it can be considered as a hybrid feature but the contributions from edges are much higher than the contributions from texture.

Fig. 3.3 displays the input frame reduced to grayscale and its local variance. Since local variance itself is a contour descriptor, no other edge detection step is necessary.

Figure 3.3: Grayscale input frame and its local variance map

3.2 Extraction of Spectral Information

There are several methods that can be used to find different textures in an image. Some of these methods are: maximum probability, image moments, contrast, homogeneity, and entropy measures [59]. All of these methods use co-occurrence matrices for their calculations. Gray level co-occurrence matrices are statistical measures about repeating nature of pixel levels having similar values [28]. Gray level co-occurrence matrices have to be calculated for four directions (0°, 45°, 90°, 135°) and as the number of gray levels increase, co-occurrence matrices become more complex.

Texture detection methods based on gray level co-occurrence matrices are memory intensive and slow.

Another possible method to locate different textures is finding different energy levels in local regions. There are many methods to compute image energies. Transform energies are preferred in this research due to their easy and fast calculation. Parseval's theorem and energy relationship of cosine transform indicate signal energy is preserved after discrete Fourier transform (DFT) and discrete cosine transform (DCT) [25]. Moreover, by separating the DC component, energy due to gray level and energy due to transitions can be found.

3.2.1 Discrete Fourier Transform Energy

The DFT of a small *N* by *N* image sub-block can be calculated as:

$$I(k_1, k_2) = \sum_{n1=0}^{N-1} \sum_{n2=0}^{N-1} i(n_1, n_2) e^{-j\left(\frac{2\pi}{N}\right)k_1 n_1} e^{-j\left(\frac{2\pi}{N}\right)k_2 n_2} \qquad (3.10)$$

for $0 \leq k_1 \leq N-1$ and $0 \leq k_2 \leq N-1$.

The total energy spectral density of the sub image is given by:

$$E_i = \sum_{k1=0}^{N-1} \sum_{k2=0}^{N-1} \left[I(k_1, k_2) I^*(k_1, k_2) \right] \qquad (3.11)$$

where I^* is the complex conjugate of I. Here the power of the DC component $I^2(0,0)$ represents energy due to mean intensity level in that region. If DC component is removed we will find the energy due to gray level changes. This corresponds to textural energy where there are no significant edges in intensity.

$$E_i' = \sum_{k1=0}^{N-1} \sum_{k2=0}^{N-1} [I(k_1, k_2) I^*(k_1, k_2)] - [I(0,0)]^2 \quad (3.12)$$

When these sub image blocks are repeated for the entire image with intervals of N, the combined DFT gives the periodogram of the image. Periodogram is basically 2D extension of spectrogram in 1D signals. The energy of this periodogram gives us the spectral energy content of local regions in the image.

Fig 3.4 shows the DFT energy for the sample video frame calculated with the method above. The resulting image shows similar spectral energy levels on road surface and the sky portion of the image. Spectral energy is significantly higher in the region with trees.

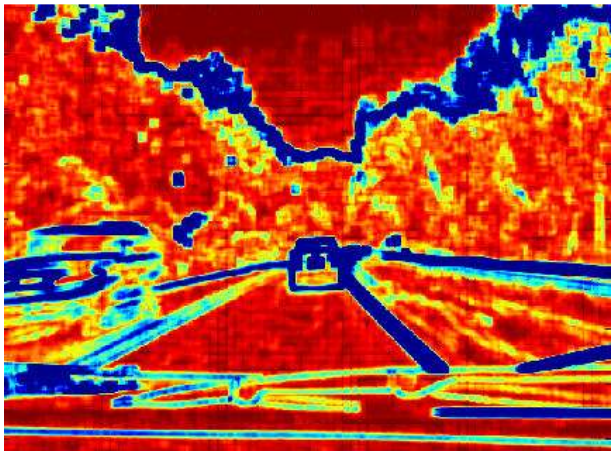

Figure 3.4: Discrete Fourier Transform energy for the sample frame

3.2.2 Discrete Cosine Transform Energy

Discrete cosine transform of an N by N sub image is given as:

$$C(k_1, k_2) = \sum_{n1=0}^{N-1} \sum_{n2=0}^{N-1} 4i(n_1, n_2) cos\left(\frac{\pi}{2N}\right) k_1(2n_1 + 1)\, cos\left(\frac{\pi}{2N}\right) k_2(2n_2 + 1) \quad (3.13)$$

for $0 \leq k_1 \leq N-1$ and $0 \leq k_2 \leq N-1$.

The energy of the transformed block is given by:

$$E_c = \frac{1}{4\,N_1 N_2} \sum_{k1=0}^{N-1} \sum_{k2=0}^{N-1} w_1(k_1) w_2(k_2) [C(k_1, k_2)]^2 \qquad (3.14)$$

where $w_i(k_i) = \begin{cases} 0.5\,, k_i = 0 \\ 1\,, 1 \leq k_i < N \end{cases}$

Again if we remove the DC component, we will find the textural energy due to gray level changes:

$$E_c{}' = \frac{1}{4\,N_1 N_2} \left(\sum_{k1=0}^{N-1} \sum_{k2=0}^{N-1} w_1(k_1) w_2(k_2) [C(k_1, k_2)]^2 - 0.25\,[C(0,0)]^2 \right) \qquad (3.15)$$

Fig 3.5 displays the DCT spectral energy content for the sample video frame. The result is more detailed than the DFT spectral energy image. Different energy levels of the sky, forest and road portions can clearly be seen.

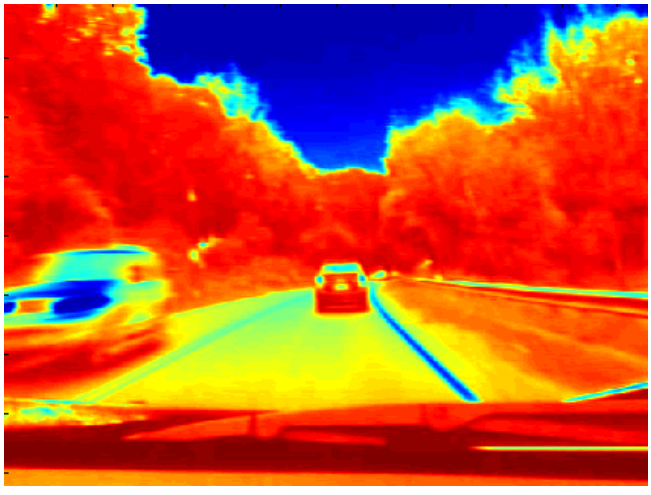

Figure 3.5: Energy map obtained from discrete cosine transform for the sample frame

To obtain a segmentation based on textural energy, cosine transform energy map is clustered using recursive Otsu method, which is also used in color clustering. Following this step contours are extracted from the clustered image. The results of clustering and contour extraction for the sample frame are shown in Fig 3.6.

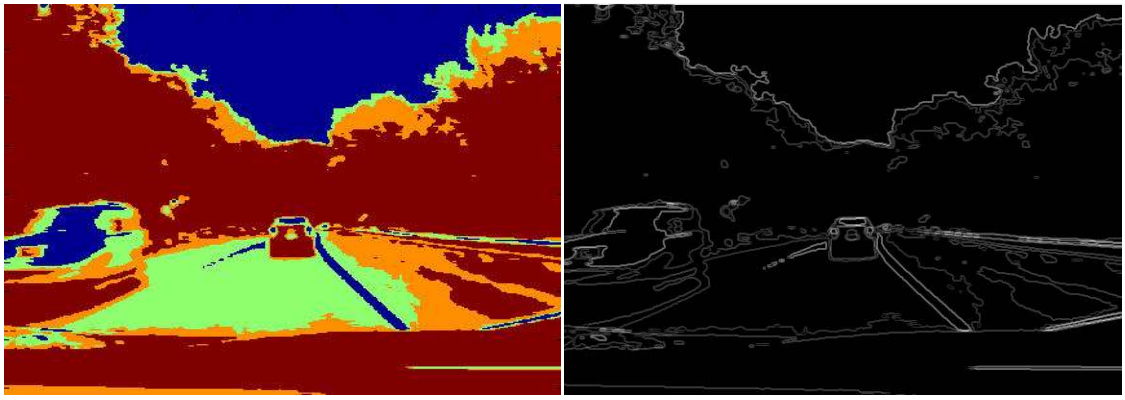

Figure 3.6: Texture segmentation with DCT energy and respective contours

3.3 Extraction of Temporal Information with Optical Flow

Temporal analysis is performed by calculating the optical flow of video sequence. Optical flow is a powerful tool for detecting motion in a video. Regions containing moving objects in a video can be identified with optical flow. In addition to motion detection, the direction and magnitude of this motion can be calculated with optical flow. Many methods have been proposed for computation of optical flow [60] [61] [62]. For real time performance requirements a method with low computational cost is selected. The approach for optical flow is mostly parallel to the method proposed in [63]. It begins by calculating the spatiotemporal gradient of the video sequence $I(x,y,t)$. Then the

gradient is smoothed by a 3D Gaussian filter using 3D linear convolution as in equations (4) and (5).

$$\nabla I(x,y,t) = \left[\frac{\partial I(x,y,t)}{\partial x}\frac{\partial I(x,y,t)}{\partial y}\frac{\partial I(x,y,t)}{\partial t}\right]' \tag{3.16}$$

$$\overline{\nabla I}(x,y,t) = \nabla I(x,y,t) * H(x,y,t) \tag{3.17}$$

Here, *H* is the 3D Gaussian function, * is the convolution operation and the symbol ' represents transposition operation. After this step the tensor matrix is obtained using the smoothed gradient vector.

$$T = \overline{\nabla I} \cdot \overline{\nabla I}' = \begin{bmatrix} t_1 & t_4 & t_5 \\ t_4 & t_2 & t_6 \\ t_5 & t_6 & t_3 \end{bmatrix} \tag{3.18}$$

Following the calculation of tensors, the velocity vectors are calculated using parameters t_1 through t_6, which are the elements of the symmetric tensor matrix.

$$v_x = \frac{t_6 t_4 - t_5 t_2}{t_1 t_2 - {t_4}^2} \text{ and } v_y = \frac{t_5 t_4 - t_6 t_1}{t_1 t_2 - {t_4}^2} \tag{3.19}$$

Finally the velocity vectors for each pixel coordinate are smoothed by a 2D median filter [25] to obtain the final x and y component values of optical flow vector field. In Fig 3.7 two consecutive frames are displayed. Fig 3.8 represents the optical flow

calculated by the input frames. Notice the hue indicates the direction as shown by the small circle on the lower right corner and brightness indicates magnitude of the motion.

Figure 3.7: Consecutive video frames as input to optical flow

Figure 3.8: Optical flow calculated from frames in Fig. 3.7. Notice the difference between the flow of the background and the car travelling in opposite direction.

This concludes the feature extraction part. The next chapter will explain how extracted spatial, spectral and temporal information are combined into a saliency image with image data fusion.

4. DATA FUSION AND IMAGE SEGMENTATION

The previous chapter focused on extracting different types of information from input video frames. In this chapter the focus will be on putting these information back together in such a way that the amount of information is maximized. The final goal of this research is to construct a context enhanced output that will increase the situational awareness of the driver. The first step in this process is obtaining enhanced features to use in segmentation and object detection. These enhanced features must be more informative than the lower level features, otherwise they will not be effective in closing the semantic gap mentioned in previous chapters.

4.1 Data Fusion

The process of combining multiple sources of data into a more informative form is called data fusion. When these data sources are images, the process is called image data fusion or image fusion in short. There are multiple approaches for fusion [15]. Some fusion methods rely on fixed set of rules or fuzzy logic. This approach is called decision level fusion. Others such as principle component analysis (PCA) fusion extract some values (major eigenvalues for example) and use them as coefficients to combine pixels or sub blocks with certain ratios. This approach is called pixel or data level fusion. Another possible fusion method is feature level fusion, which examines the information content of features and combines them accordingly.

Since we are mainly dealing with image features such as spatial, spectral and temporal features, the selected approach in this research is feature level fusion. The next step after selecting fusion level is the selection of fusion method. Ideally the features with

higher information must have more contribution to the fused output compared to other features with less information. In chapter 2 we started with an entropy based approach to measure information content and derived a fusion model that uses mean square error (MSE). Starting with two images I_1 and I_2 we found a method to combine their information by scaling one of them in such a way that emphasizes their common information.

$$I_f = \alpha I_1 + I_2 \tag{4.1}$$

The parameter α is calculated by minimizing the MSE between two image histograms:

$$MSE = \frac{1}{T}\int_{k=0}^{1}[p_{\alpha I1}(k) - p_{I2}(k)]^2 dk \tag{4.2}$$

The local minimums are given by finding the negative to positive zero crossings of the equation:

$$\int_0^1 \frac{\partial}{\partial \alpha} P_{I1}(k, \alpha)\, dk = 0 \tag{4.3}$$

As stated earlier there are multiple image features coming from spatial, spectral and temporal elements of the video sequence. The spatial, spectral and temporal feature fusion model is based on human vision. Humans look for contrasts in motion, color, intensity and texture and combine them to identify the contours of objects. Therefore, the proposed scene segmentation worked by combining those features in a way that emphasized the contours common to all or most of them. Maximizing the mutual information between image features or minimizing their differences (MSE) should provide such results. In fact, [23] claims that mutual information and minimum MSE are connected with a differential equation:

$$H_M(X, Y) = \frac{1}{2}\int_0^{snr} MSE(\gamma) d\gamma \tag{4.4}$$

These findings suggest if we find the MSE of two image histograms versus the scalar α and integrate, we should have a scaled version of mutual information. Matlab results confirm this relation.

To demonstrate consider I_1 and I_2, edge and color features extracted from the sample frame respectively.

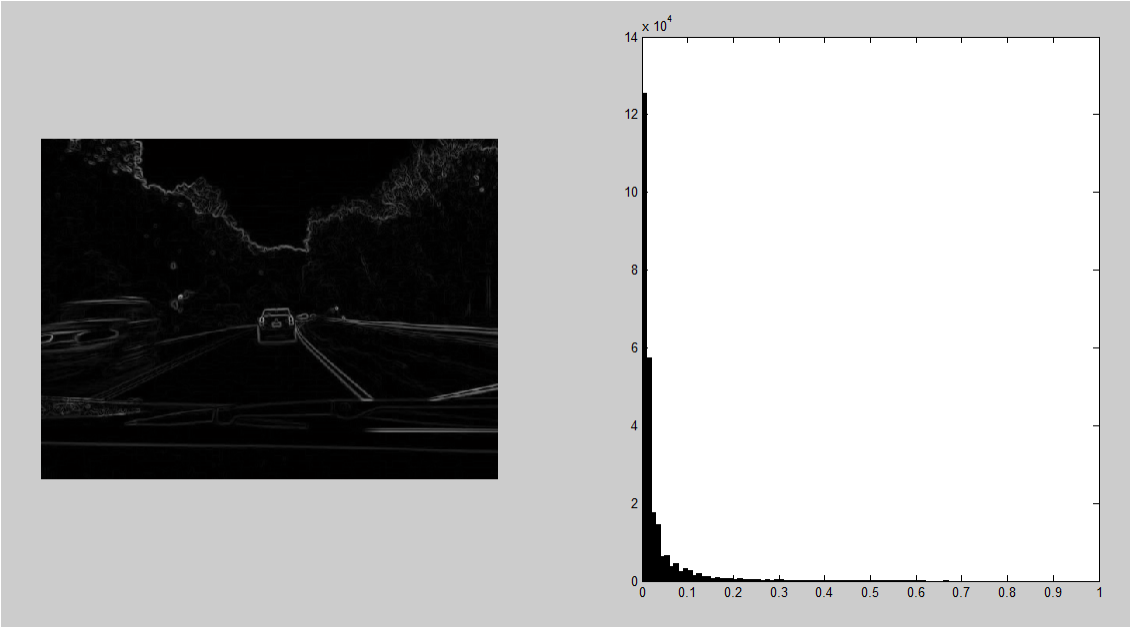

Figure 4.1: I_1, edge features of the sample frame and its histogram

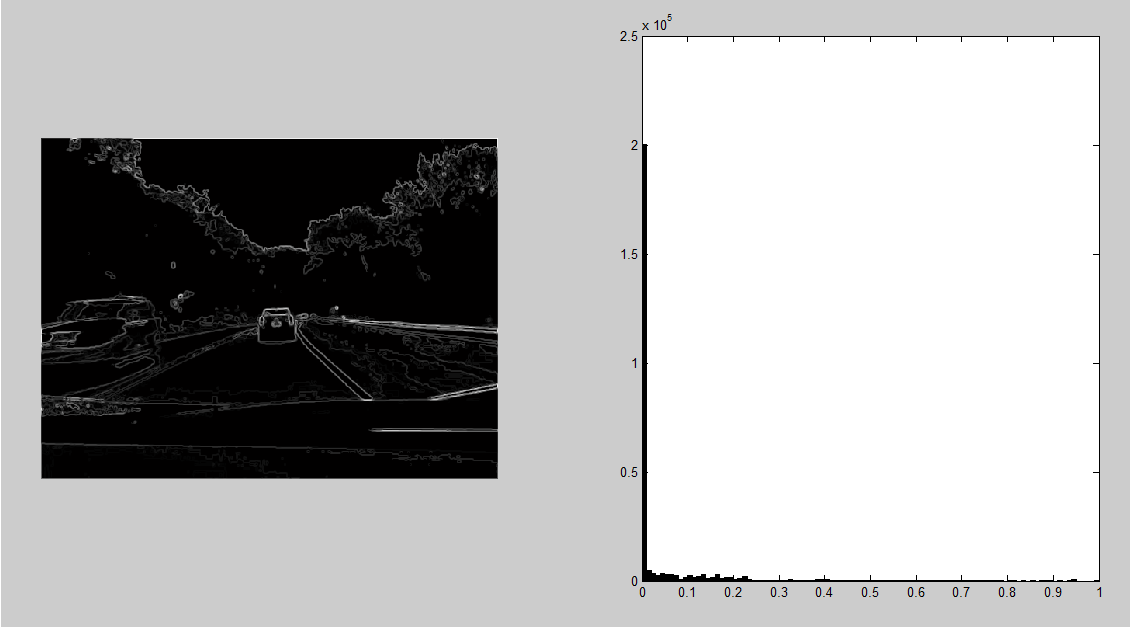

Figure 4.2: I_2, color features of the sample frame and its histogram

Since I_1 has greater variance it will be scaled with alpha coefficient to match I_2 histogram. Fig. 4.3 shows MSE of their histograms versus alpha:

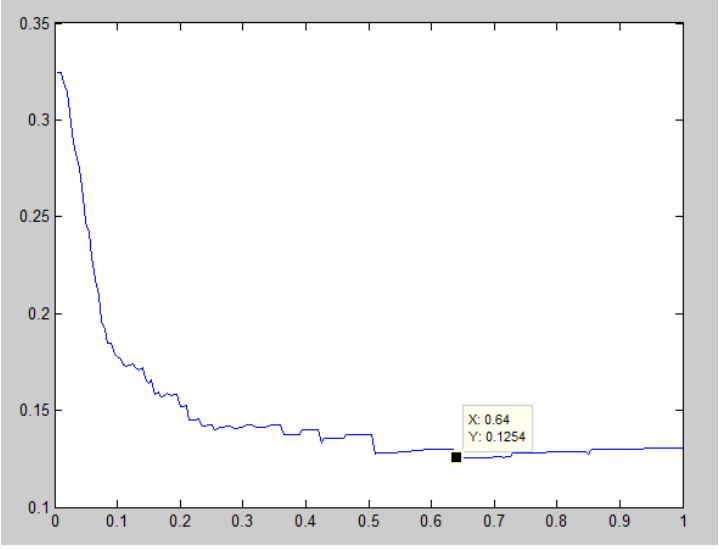

Figure 4.3: MSE{p(α I_1) | p(I_2) } vs. α

The next plot overlays $\int_{k=0}^{1} \frac{\partial}{\partial \alpha} p(\alpha\, I_1(k))\, dk$ on MSE plot. According to our optimization when this statement equals zero, it is a critical point for MSE.

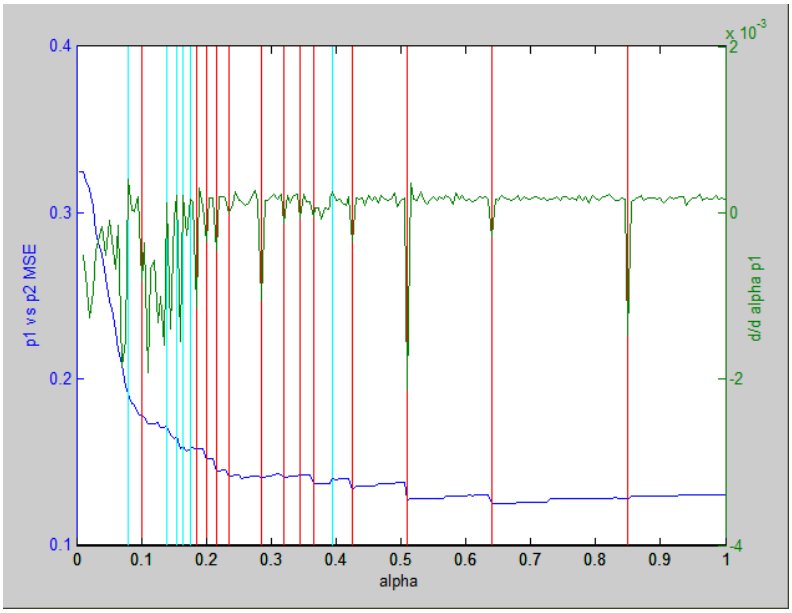

Figure 4.4: $\int_{k=0}^{1} \frac{\partial}{\partial \alpha} p(\alpha\, I_1(k))\, dk$ and critical points. Positive to negative zero crossings (shown with red lines) are local minimums and negative to positive zero crossings (cyan lines) are local maximums.

This shows critical points of MSE can be found accurately and we do not have to search the entire $\alpha \in [0,1]$ interval but rather find the critical points to find min MSE.

When the result is compared with the Mutual Information versus alpha plot we find that one of the local maxes is at α=0.64, which is found to be min MSE generating value as shown in Fig 4.5.

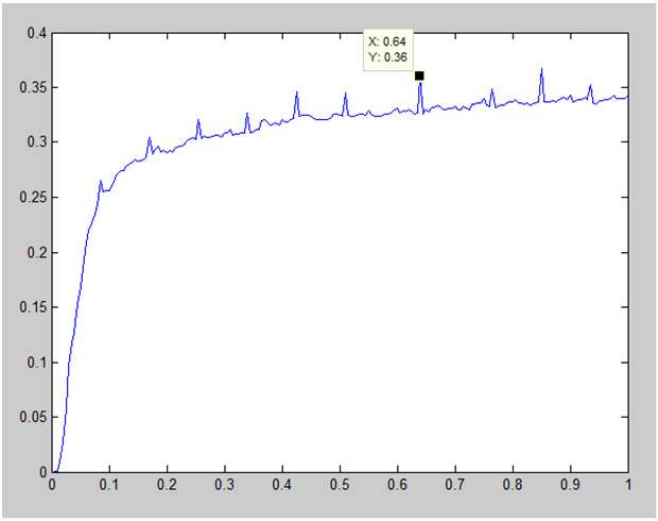

Figure 4.5: $H_M(\alpha\, I_1, I_2)$ vs. α

When the sum of the MSE is plotted versus alpha as in Fig. 4.6, we obtain results that look similar to Mutual Information plot in Fig 4.5. This demonstrates the differential relation between Mutual Information and MSE.

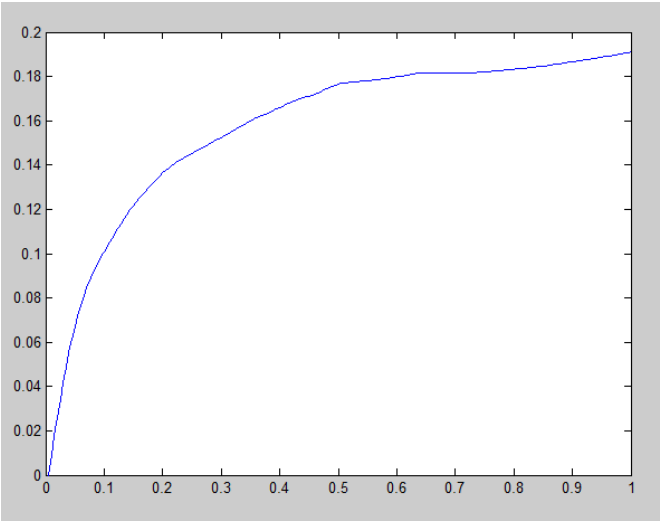

Figure 4.6: $\sum_{\alpha} MSE(\alpha)$ vs α

The example demonstrates how the two images are combined by finding an optimal scaling coefficient. We combine more than two images however. As mentioned earlier this would complicate things if mutual information was used for fusion. With the minimum MSE approach we keep the image with smaller variance constant and scale the intensity of the other image. By extending this approach we can select the image with the minimum variance (among edges, variance, colors, optical flow, spectral energy, etc) and find a scaling factor for the others.

This gives a scaling rate (α^*) for each feature map and the scaling rate for the one with the minimum variance will be 1. The fused image will be given by:

$$I_f = I_{\min\_var} + \sum_i \alpha_i{}^* I_i \qquad \text{where } I_i \neq I_{\min\_var} \tag{4.5}$$

Resulting fused image is given in Fig. 4.7.

Figure 4.7: Fused saliency image from low level features

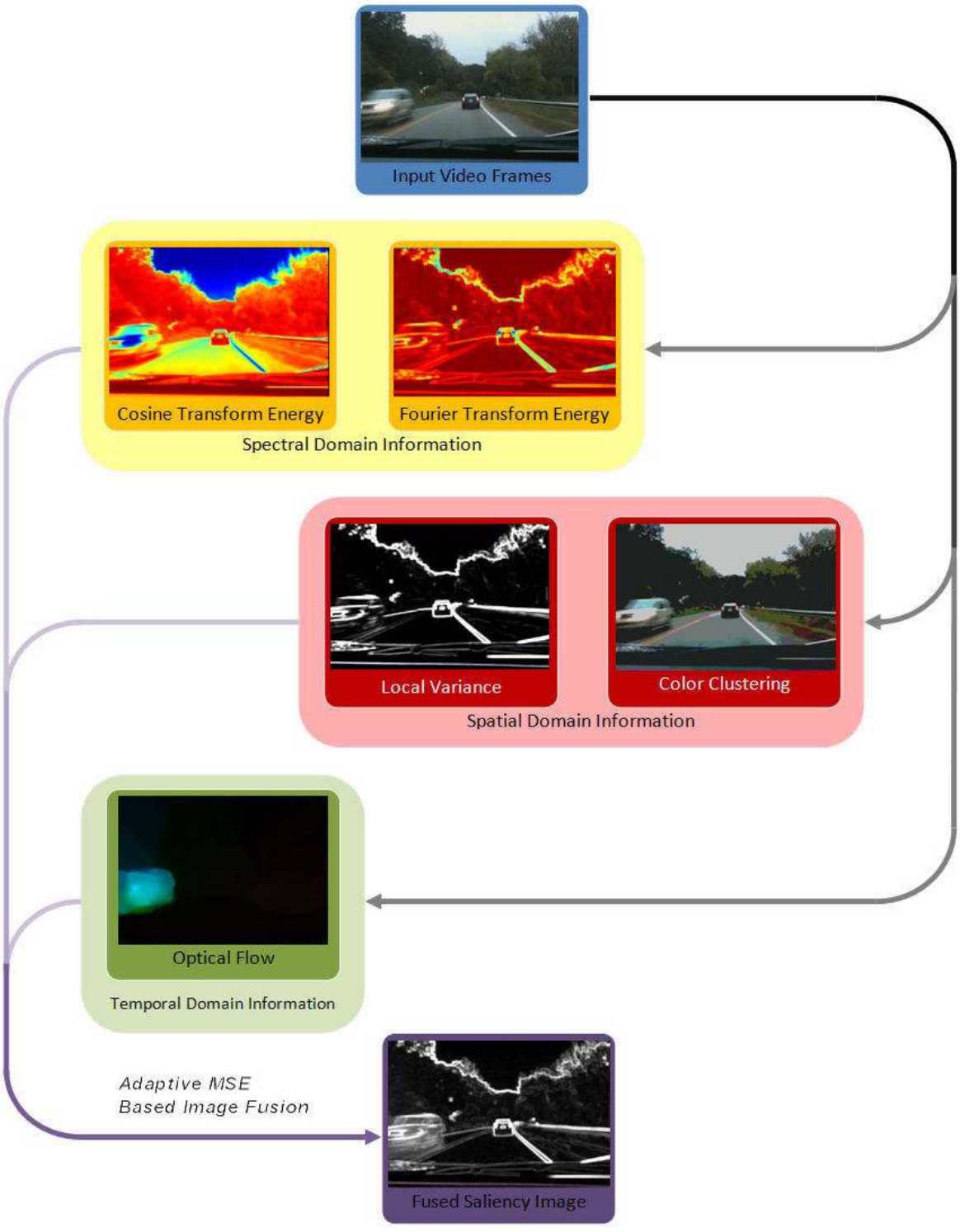

Figure 4.8: Overall feature extraction and fusion process

Fused image in Fig. 4.7 emphasizes all the important regions in the video frame. The road boundaries, the tree line, the road barrier on the right and the two cars are clearly visible. Fused feature images can be treated as saliency images. It is a combination of all features that stand out to human eye. Fig. 4.8 summarizes feature extraction and fusion process.

4.2 Image Segmentation

Segmentation based solely on color or texture clustering usually produces less than perfect results as seen in Figs. 3.1 and 3.6. Although they contain considerable amount of information, they are not sufficient for a good segmentation result. The fused saliency image in Fig. 4.7, on the other hand, clearly shows the boundaries for significant regions in the image. This can be exploited for better image segmentation. To achieve that, the saliency image is first thresholded to extract region boundaries. The resulting binary image shows boundaries for regions such as sky, trees and the road. The boundaries at this stage are very wide and coarse. However, they are sufficient to obtain the major regions in the image. Regions separated by boundaries are labeled as seen in Fig. 4.9. The major regions seen in the image are the sky, the trees, the road, the region between road shoulder and the barrier, and also the dashboard of the car. The gray portions are marked as unknown. These unknown labeled regions need to be assigned to a neighboring segment or to a new segment. For example, gray pixels on the horizon either belong to the segment with sky or the segment with the trees. The large gray blob on the left, which is mostly the passing car, should have its own segment.

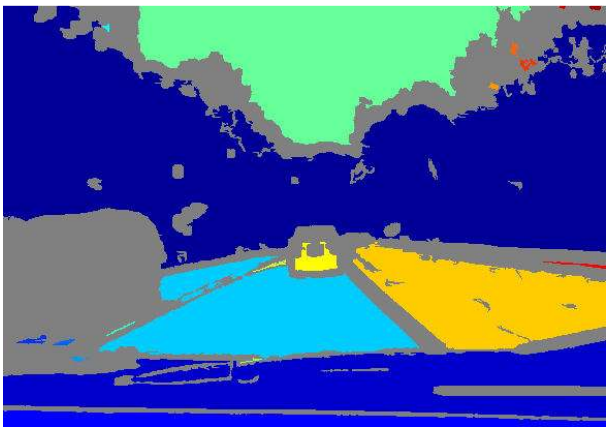

Figure 4.9: First stage of image segmentation with the major regions

In the next stage a combination of contours from texture and color clustering is used to obtain another segmentation map, which can be seen in Fig 4.10. This second segmented image has much more regions but it does not have coarse boundaries and unassigned regions. Next it is compared with the segmented image with all the major regions. In the second segmented image if two selected pixel coordinates from different segments belong to the same segment in the first segmented image, then the two segments are merged.

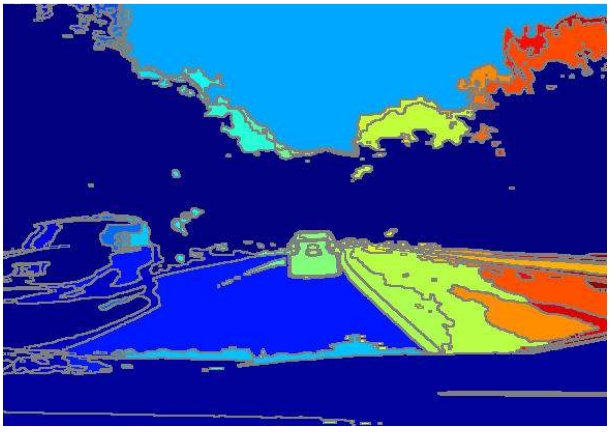

Figure 4.10: Finer segmentation before region growing step

Not all segments in the second segmented image will be matched with a segment from the first image. These are the segments that are fully under unknown portions in the coarse segmented image. The mask for these segments is displayed in Fig 4.11a. The nearby segments in Fig4.6a are morphologically merged and Fig 4.11b is obtained. New segments are assigned to these regions. Fig 4.12 shows the segments after all merging and new segment assignments.

Figure 4.11: Unknown regions in the intermediate segmentation step

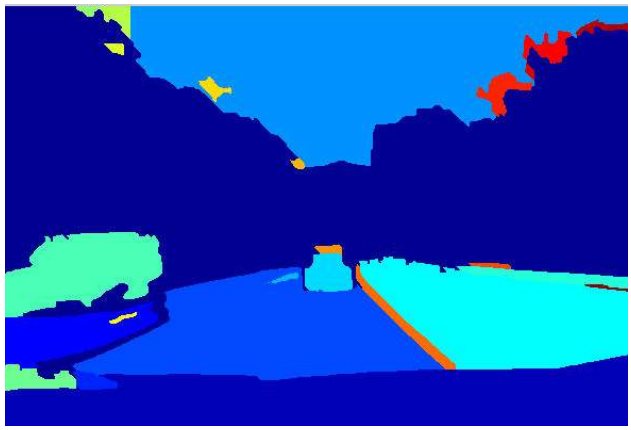

Figure 4.12: Segmentation after unknown regions are assigned

A final region growing step is applied to segments with small sizes. They are assigned to their nearest neighbor in terms of color and texture, unless they are on a high saliency region. The resulting overall segmentation for the sample frame is illustrated in Fig 4.13. Fig 4.14 shows the segment boundaries overlaid with the video frame. Figure 4.15 summarizes the segmentation process.

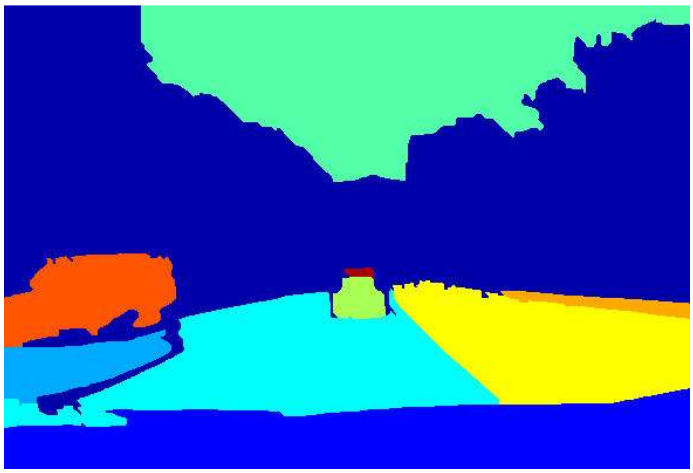

Figure 4.13: Segmented driving scene after final region growing

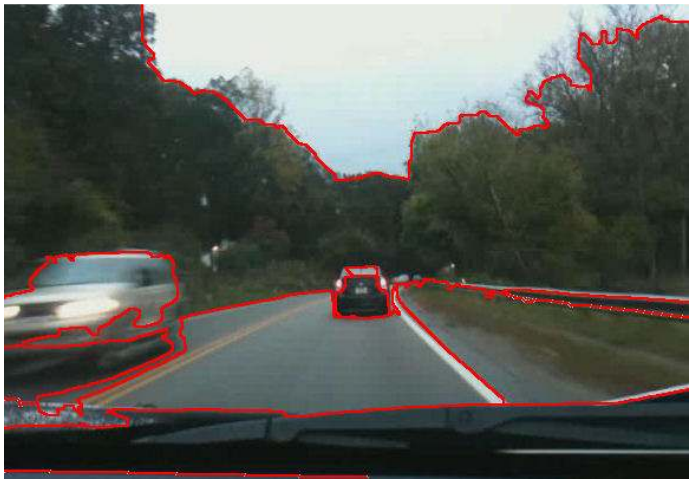

Figure 4.14: Input video frame with segment boundaries

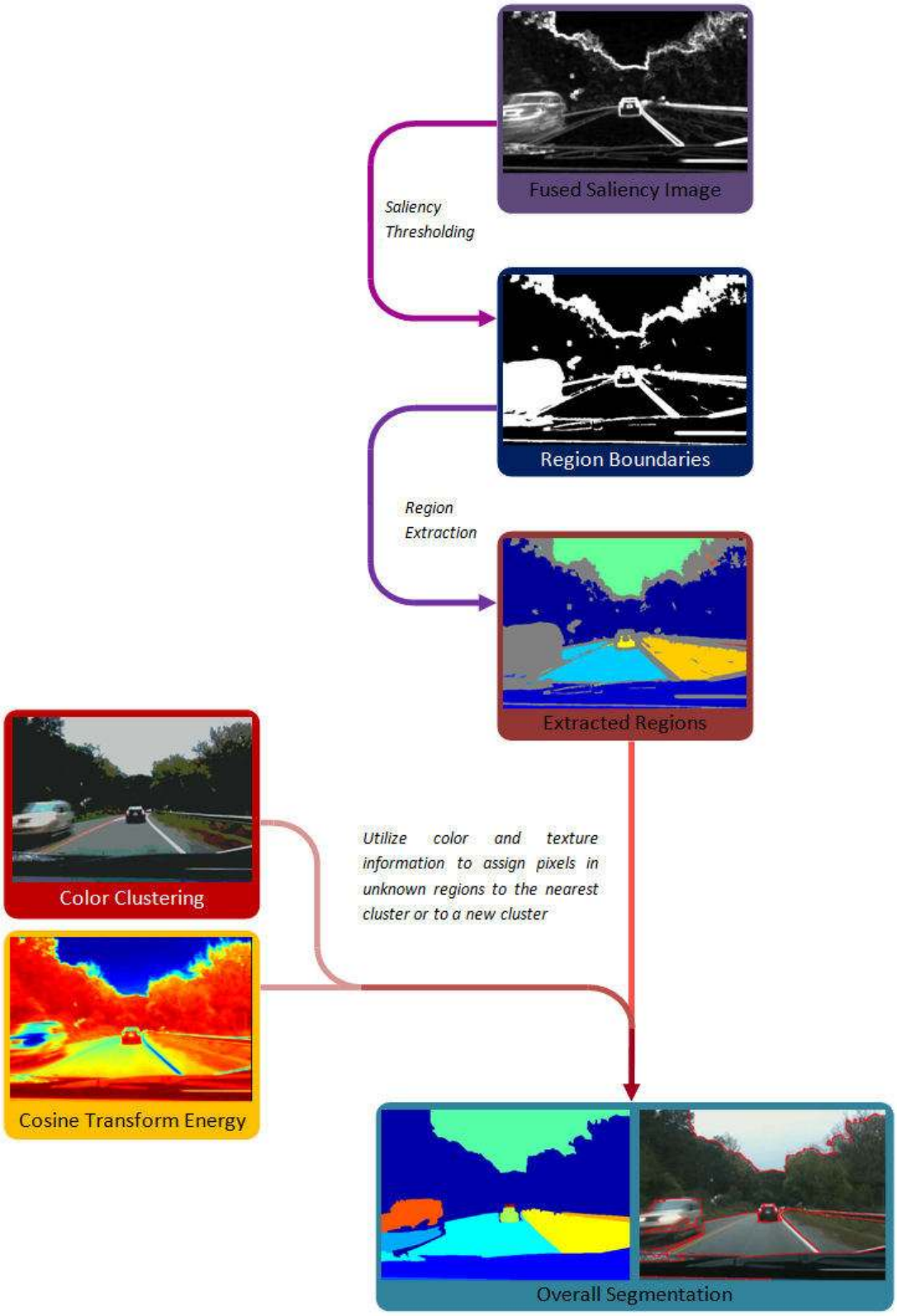

Figure 4.15: Video frame segmentation steps

4.3 Blob and Vehicle Detection

4.3.1 Blob Detection with Gaussian Scale Space

In the previous section fused saliency image helped us find the significant region boundaries in the image and enabled segmentation. Fused saliency image holds another key information for this research. The objects on the road are clearly indicated in the saliency image with their dense edges and motion. When compared to their low intensity neighborhood, these regions stand out. Humans can easily see these regions, but for a computer vision algorithm to detect these, a blob detection algorithm is necessary.

In this research Gaussian scale space (GSS) approach is chosen for blob detection. GSS is commonly applied to images for scale invariant image matching and detection of blobs in selected scales (or sizes) [64]. GSS is a family of images obtained by progressively blurring an input image with Gaussian filters. Consider the convolution operation with a Gaussian 2D symmetrical Gaussian below:

$$\bar{I}(x, y, \sigma_j) = I(x, y) * G(x, y, \sigma_j) \tag{4.6}$$

$$\bar{I}(x, y, \sigma_j) = I(x, y) * \frac{1}{2\,\pi\sigma_j} e^{-\left(\frac{x^2+y^2}{2\,{\sigma_j}^2}\right)} \tag{4.7}$$

Here the scale of the filtered image $\bar{I}$ is given by σ_j, which is the variance of the Gaussian in both dimensions. As scale parameter σ_j gets progressively larger, smoother images are obtained. GSS is constructed by applying a Gaussian smoothing filter up to a certain scale and recording the set of smoothed images. The result is a 3D image, the three dimensions being *x, y*, and scale.

From this point two approaches can be used to find blobs in an image, difference of Gaussians (DoG) and Laplacian of Gaussians (LoG). The DoG is obtained as:

$$\frac{\partial}{\partial\sigma}G(x,y,\sigma) \approx \frac{\sigma}{\Delta\sigma}(G(x,y,\sigma+\Delta\sigma) - G(x,y,\sigma-\Delta\sigma) \tag{4.8}$$

When DoG filter is applied to an image:

$$I(x,y) * \frac{\partial}{\partial\sigma}G(x,y,\sigma) = \frac{\partial}{\partial\sigma}[(I(x,y) * G(x,y,\sigma)] = \frac{\partial}{\partial\sigma}\bar{I}(x,y,\sigma) \tag{4.9}$$

$$\frac{\partial}{\partial\sigma}\bar{I}(x,y,\sigma) \cong \frac{\sigma_j}{\Delta\sigma}\left[\bar{I}\left(x,y,\sigma_j+\Delta\sigma\right) - \bar{I}\left(x,y,\sigma_j-\Delta\sigma\right)\right] \tag{4.10}$$

By taking $\Delta\sigma = (\sigma_j - \sigma_{j-1}) / 2$, Eqn. 4.10 can be reduced to:

$$\frac{\partial}{\partial\sigma}\bar{I}(x,y,\sigma) \cong \frac{1}{2}\left[\bar{I}\left(x,y,\sigma_j\right) - \bar{I}\left(x,y,\sigma_{j-1}\right)\right] \tag{4.11}$$

Therefore, DoG of a GSS image can be found by going through the entire scale space (σ_{min} to σ_{max}) and taking the differences of consecutive blurred images. A local minimum is observed on DoG when a blob radius roughly matches $\sqrt{2\,\sigma_j}$. By finding these local minimums, blobs with different sizes contained in an image can be found.

Another method for finding blobs with GSS is using LoG. LoG is obtained by:

$$\nabla^2 G(x,y,\sigma) = \frac{\partial^2}{\partial x^2}G(x,y,\sigma) + \frac{\partial^2}{\partial y^2}G(x,y,\sigma) \tag{4.12}$$

Applying LoG filter to an image yields:

$$I(x,y) * \nabla^2 G(x,y,\sigma) = \nabla^2[(I(x,y) * G(x,y,\sigma)] = \nabla^2\bar{I}(x,y,\sigma) \tag{4.13}$$

$$\nabla^2 \bar{I}(x,y,\sigma) = \frac{\partial^2}{\partial x^2}\bar{I}(x,y,\sigma) + \frac{\partial^2}{\partial y^2}\bar{I}(x,y,\sigma) \tag{4.14}$$

Much like the Sobel operators for directional derivatives there are finite impulse response (FIR) filters that provide good discrete approximations to Laplacian [25].

$$H_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad H_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad H_3 = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix} \tag{4.15}$$

Any of these filters can be used to obtain the LoG of a scale space image.

$$\nabla^2 \bar{I}(x,y,\sigma) \cong H * \bar{I}(x,y,\sigma) \tag{4.16}$$

Similar to DoG a local minimum on LoG indicates a blob with a size around $\sqrt{2\,\sigma_j}$. After applying both methods to scale space of fused saliency image, LoG is observed to produce slightly better results. An important parameter while finding the blobs in saliency image is the range of scales. This range must be carefully selected. Since saliency image mainly consists of relatively thin contours rather than large blobs, detection of these contours must be avoided. Fig. 4.16 illustrates what happens when σ_{min} is too small and contours get detected.

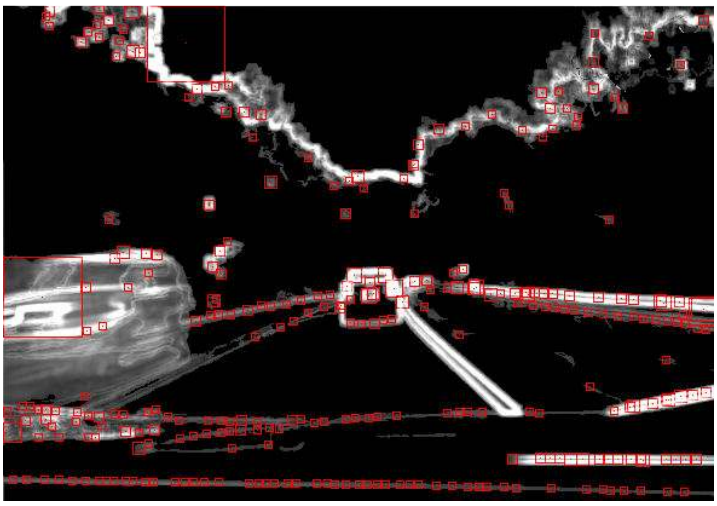

Figure 4.16: Thin edges get picked up by the blob detector when $\sigma_{min} < 10$

Fig. 4.17a shows blob detection results when σ_{min} issue is fixed and 4.17b displays blobs after being thresholded at the selected scale. Note the red dots at the center of each frame marking the blobs represent the centroids of the blobs.

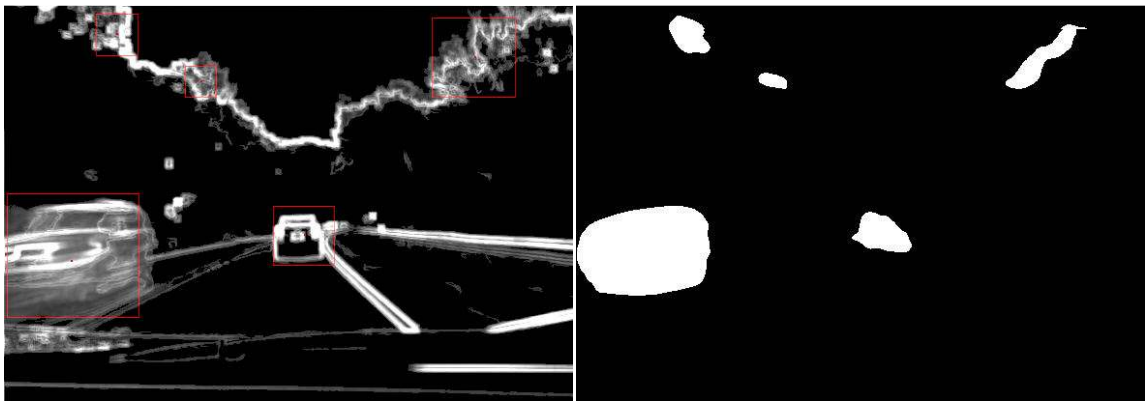

Figure 4.17: Detected blobs in the saliency image and their rough shapes

4.3.2 Finding Vehicles Among Detected Blobs

Blob detection with LoG finds the regions of interest in saliency image. However, some of these blobs might belong to the background and not carry relevant information. As stated in section 2.4 Related Intelligent Vehicle Research, vehicle features such as horizontal edges and shadows have been the most useful features for identifying vehicles. In addition to these features we have utilized relative motion obtained from optical flow, as a third useful feature.

To find the vehicles, first the edge map of the region around a detected blob is obtained. In order to find horizontal edges, directional filters such as Gabor filters [28] [65] can be utilized. Another possible method is taking the Hough transform of the edge map [59]. Hough transform can identify lines in an image by their angles and length. Each line produces an intersection of curves. These points where Hough curves intersect can be found by searching local maximums. Hough transform coordinates of a local maximum indicates the angle of the corresponding line and its distance to the origin in image coordinates. Horizontal edges are strong indicators of vehicular features. Therefore, if the Hough transform of the edge map produces more local maximums near the angles 0° and 360° that means most of the lines are horizontal, or near horizontal. Fig 4.18 displays the edge map of a blob containing a car and its Hough transform. This is the car on the left side of the sample frame.

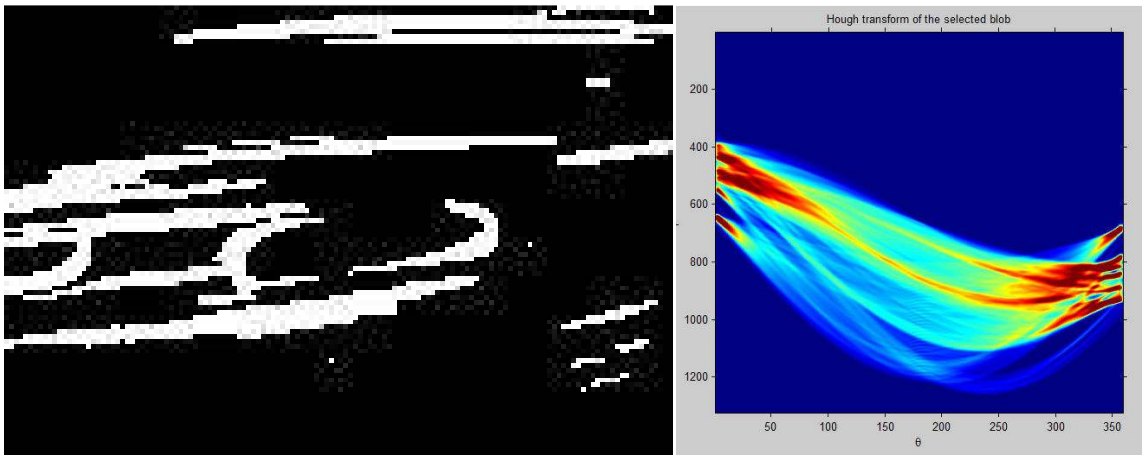

Figure 4.18: Edges of an image region containing a car and its Hough transform

Another important feature for finding vehicles in images and videos is the shadow. In daylight conditions all vehicles form a shadow over the road surface. This shadow can be detected by averaging the grayscale blob image in x direction. The result is a projection of the image on y direction. For blobs containing a vehicle, this projection will show a local minimum near its end. This is where the intensity levels drop as the shadow of the car becomes apparent and rise again afterwards. For the car example given in Fig 4.19a, average gray levels clearly indicate a shadow as seen in Fig 4.19b.

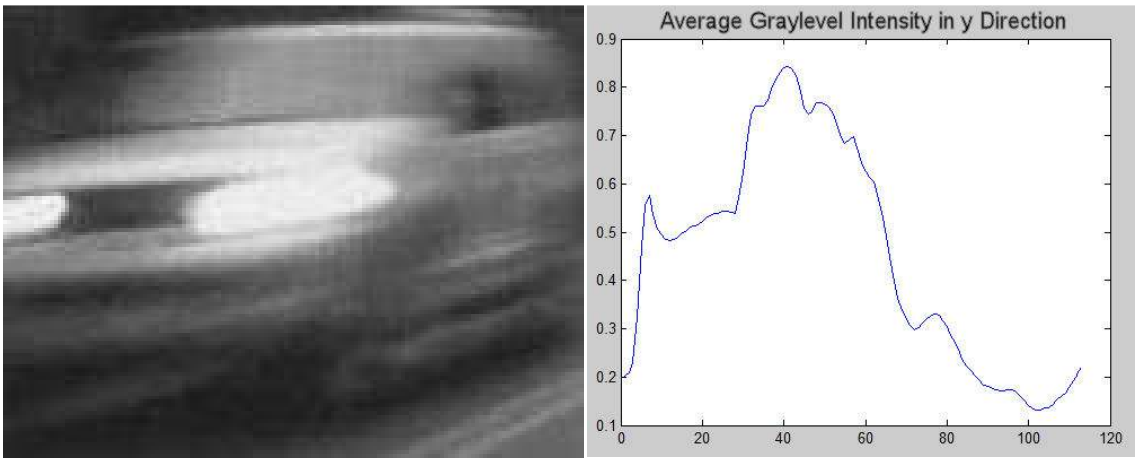

Figure 4.19: Selected region and its average intensity in y direction.

A third feature used for vehicle identification is the average optical flow of the selected region compared to the average flow of a larger region around it. The more relative motion the blob has compared to its surroundings, the more likely it is to be a vehicle. Therefore, the difference in average flow magnitude is taken as a vehicle identification feature. The feature is calculated as:

$$Avg.FlowDifference = \frac{1}{A_1}\sum_{R_1}|\vec{v}(x,y)| - \frac{1}{A_2}\sum_{R_2}|\vec{v}(x,y)| \qquad (4.17)$$

where $\vec{v}$ is the optical flow vector field, R_1 is the selected blob region, A_1 is the area of this region, R_2 is the region around R_1 and A_2 is the area of R_2. Fig 4.20 illustrates this feature on the optical flow frame obtained from the sample input.

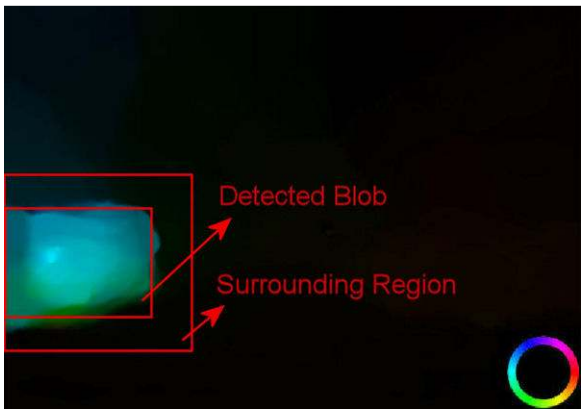

Figure 4.20: Optical flow of a blob and the region around it

By combining these three features a confidence level is calculated. If this confidence level is above a certain threshold, region containing the selected blob is marked as a vehicle. Fig. 4.21 shows the vehicle identification results for the sample frame. Of the five blobs detected in previous step, two of them are correctly identified as cars. Fig. 4.22 summarizes blob detection using Gaussian scale space and vehicle detection processes.

Figure 4.21: Results of vehicle detection on sample frame

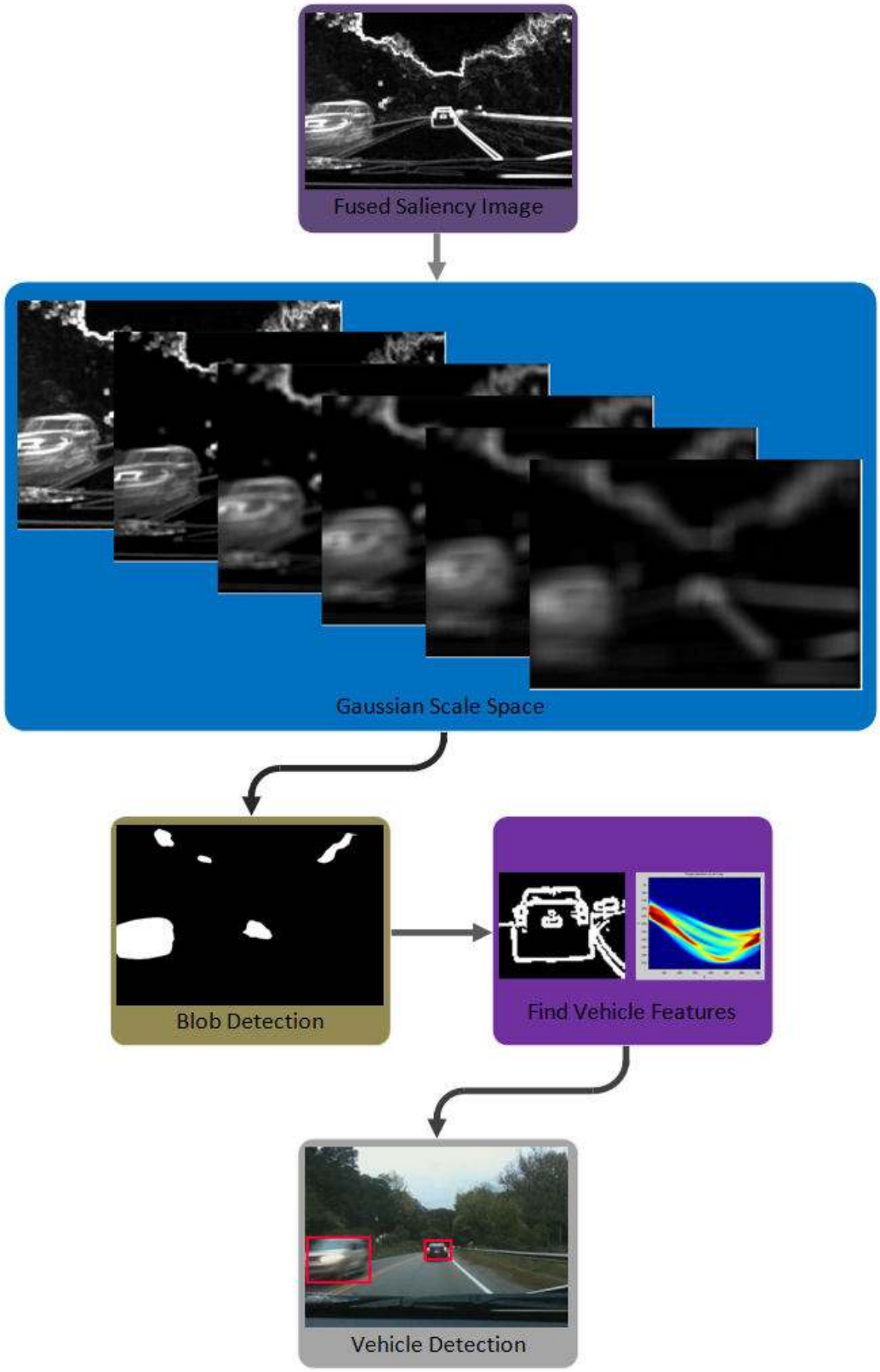

Figure 4.22: Blob and vehicle detection steps

5. BAYESIAN INFERENCE BASED LEARNING AND OBJECT TRACKING

In Chapter two we defined computer vision and learning approaches based on human visual perception and cognition. As mentioned before the construction of visual stimuli, clustering similar regions and object ground distinctions occur in the retinal layer. These processes correspond to low level feature extraction and generation of fused saliency image, segmentation and blob detection in our method which were explained in previous chapters. The recognition of known regions, objects and tracking of these objects however, occur in the visual cortex section of the brain [26]. It is certain that for our method to succeed we need learning methods that are compatible with the ones that humans use. In this chapter, the application of learning and estimation methods based on Bayesian decision theory to the results of segmentation and object detection methods will be covered. The learning and classification mechanisms will ultimately help us identify regions in the scene and track objects.

5.1 Road Surface Identification with Progressive Maximum Likelihood

So far we have segmented the driving scene into various background regions and foreground objects. The background regions include the road, the sky and possibly the various elements of the scenery such as buildings in a city, fields, forest, desert, mountains, etc. Among all these regions the one that matters most to the driver is the road. Therefore, segments that belong to the road surface must be identified first. The rest of the background regions do not carry as much importance and they can be grouped together. Merging non-road objects into a single background region actually simplifies the learning task greatly. Instead of learning all the possible background types, the

method only has to learn how to classify roads surfaces. When road surfaces are learned, the classification step will categorize all segmented regions into road and non-road classes. The result will be a final region growing steps that merges all road segments into a single region and merges all other background segments into another region.

In Chapter 2 under Learning from Observations section, we discussed that a suitable Bayesian learning method for segment identification could be Maximum Likelihood algorithm. As mentioned in the same section, region identification in images is a subject of object ontology. Therefore, ontological features are needed to achieve learning and classification. These ontological features can be position, size (area), color, and shape of the region as given in [32]. The features selected for maximum likelihood learning in this research are position, area, color, and shape of the regions as seen in Fig. 5.1.

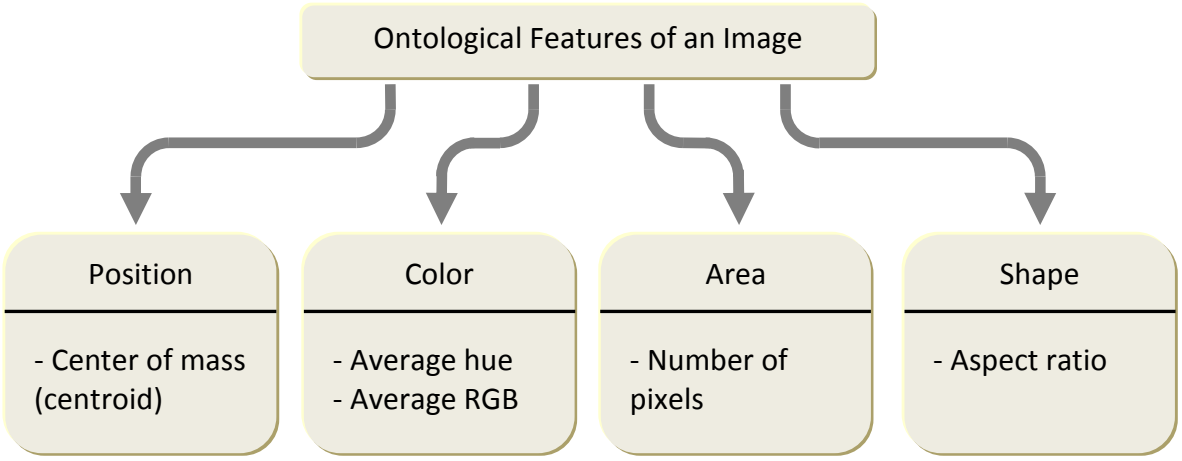

Figure 5.1: Ontological features used for Maximum Likelihood classification

These features can be obtained by examining the values and distributions of pixels in each segment. Area and position of regions can be extracted directly from raw image moments of a segment [28]. These moments are obtained by:

$$m_{pq} = \sum_x \sum_y I(x,y) x^p y^q \tag{5.1}$$

From 5.1, if we obtain a region mask

$$I(x,y) = \begin{cases} 1, & \{x,y\} \in R \\ 0, & otherwise \end{cases} \tag{5.2}$$

where R represents selected region, area of a region is simply m_{00} which is simply the number of pixels in R. The area centroid will be given by:

$$\{\bar{x}, \bar{y}\} = \left\{\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right\} \tag{5.3}$$

Color features can be calculated by taking the average of pixel RGB and Hue values in the selected region. Although taking hue value might seem redundant, the fact that hue is independent of brightness makes it a strong feature in cases where illumination can affect the outcome of the classification. For example a region of road covered by a shadow of a tree or illuminated with the headlights of an approaching car will have different colors than the rest of the road, yet their average hue should not differ much. Also, the relation between RGB and Hue is non-linear. Therefore hue needs to be a separate feature if it is going to be utilized.

The last ontological feature used for classification is the shape of the region. There are possible methods to quantify the shape of a region such as Fourier descriptors,

invariant features, etc. However, for simplicity only the aspect ratio of the region is utilized in this work. Aspect ratio is obtained by dividing maximum extend of the region in x direction by its maximum extend in y direction.

These ontological features are high level image features towards scene understanding. These features could not have been obtained if lower level features had not been utilized for creating higher level descriptors such as segmented images or extracted foreground objects.

If features for similar regions are assumed to have Gaussian distributions, classes can be modeled by their means and covariance [28]. Training stage of maximum likelihood algorithm is as simple as calculating these statistical values. In order to classify road regions an initial training set is required. To provide generality the training set is formed by taking several feature vectors from multiple videos with different road characteristics. These roads with different characteristics include, separated highway, county roads, busy town streets, etc. as well as similar roads under different weather and illumination conditions. With this initial training set classification of road and non-road regions can begin as soon as the first video frame.

For classification Bayesian decision theory is used as formulized by the Eqn (5.4). At the classification stage conditional class probabilities are compared. If a test instance is more likely to belong to one class then the other, the instance is assigned to the more likely class. The likelihood of x belonging to a class C_i can be found by the discriminant function $y_i(x)$. The discriminant function, given in eqn. (5.5) is a similarity measure between test instance x and the class C_i.

$$P(C_1|x_0) \lesseqgtr P(C_2|x_0) \tag{5.4}$$

$$y_i(\vec{x}) = -\frac{1}{2}(\vec{x} - \vec{\mu}_i)^T \Sigma_i^{-1}(\vec{x} - \vec{\mu}_i) - \frac{1}{2}\ln(|\Sigma_i|) + \ln(P(C_i)) \tag{5.5}$$

Here Σ_i represents the covariance of class C_i and μ_i represents the mean of C_i. (5.5) is a variation of Mahalonobis distance between test instance $\vec{x}$ and the class mean $\vec{\mu}_i$.The derivation of this function from a priori class probabilities is given in Appendix A.

As the ego vehicle travels, the driving scenery is expected to change. In order for the system to be adaptable to these changes, test instances that are close to the road class mean (i.e. high similarity measure) are added to the training set. If all new additional training samples are held in the training set, the learning model will converge to a very general model with a full memory of different road structures. Although this might reduce the number of misclassified road segments, it might also increase the number of false positives by making the model more inclusive than it actually needs to be. A solution to this issue is found by introducing an age parameter to the training instances. In fact many works that use Gaussian or mixture models for segmentation and region extraction, such as [66] [67], have similar learning rate parameters. These learning rate parameters give more emphasis to recent training samples to ensure temporal adaptability.

With this progressive learning method, while the video frames progress and new training instances are added to the road model, features that are older than a certain limit are removed from the model. In other words old road patterns are forgotten. This ensures that the model converges to the recent road conditions and adapts to the changes. The

diagram in Fig. 5.2 depicts this progressive maximum likelihood method. The result of applying region identification method to the sample frame can be seen in Figs. 5.3 and 5.4.

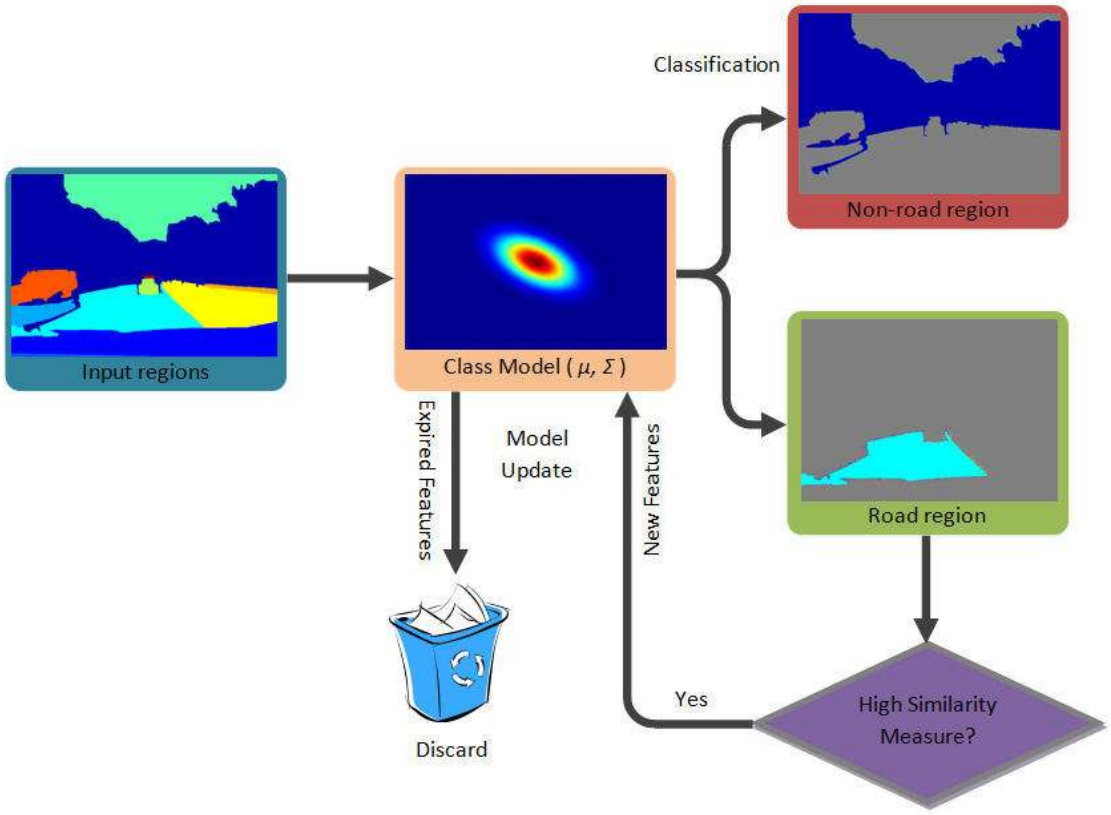

Figure 5.2: Progressive maximum likelihood learning and classification

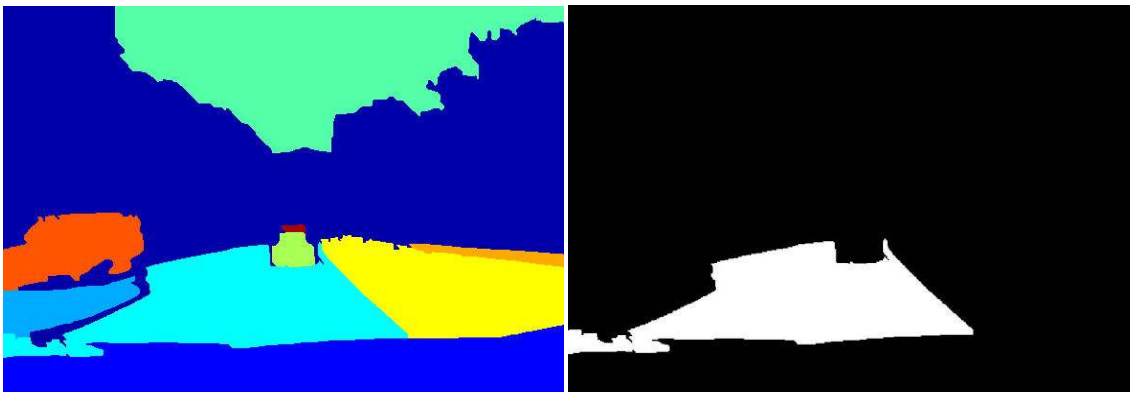

Figure 5.3: Segmentation of the scene and the classification result for the road model

Figure 5.4: Overlay of classification result on the sample frame

5.2 Foreground Object Tracking with Kalman Filtering

In Chapter 2 we made the connection between Kalman filters and Bayesian inference methods via Hidden Markov Models. Kalman filters are indeed state estimation methods based on past observations. Foreground regions extracted by blob detection methods may not be true representations of the actual positions of objects due to shadows, illumination, motion blur and distortions due to camera sensors. Kalman filters can not only provide a better estimate of the positions of these objects, due to the

temporal nature of the data, they can also find the velocity of the objects and track their positions.

Kalman tracking begins with the prediction stage. The predicted state vector is obtained by multiplying state transition matrix with the current state and adding noise parameter. The predicted uncertainty matrix is obtained by combining current uncertainty matrix with the state transition matrix and the process noise covariance as given below.

$$\hat{X}_{t+1} = F \cdot X_t + w \tag{5.6}$$

$$\hat{P}_{t+1} = F \cdot P_t \cdot F^T + Q \tag{5.7}$$

The state vector has six parameters: x, y positions, size of the blob, change in x, y per frame, which gives the velocities v_x v_y, and the change in blob size.

$$X_t = \left[x, y, A, v_x, v_y, A'\right]^T \tag{5.8}$$

The contents of the other Kalman vectors and matrices are presented in Appendix B.

The next step in Kalman filtering is the calculation of error and Kalman Gain. The observation vector is formed by the measurement matrix, a measurement vector from blob detection and measurement noise.

$$Z_t = H \cdot X_{BD} + v \tag{5.9}$$

Prediction error is calculated by subtracting measurement matrix times the prediction from the result of (5.9).

$$Y_t = Z_t - H \cdot \hat{X}_{t+1} \tag{5.10}$$

$$S_t = H \cdot \hat{P}_{t+1} \cdot H^T + R \tag{5.11}$$

The covariance of this prediction error is given by adding measurement noise covariance to the combination of uncertainty and measurement matrices (5.11). The prediction error covariance is utilized in calculation of the Kalman gain.

$$K_t = (\hat{P}_{t+1} \cdot H^T) \cdot {S_t}^{-1} \tag{5.12}$$

Kalman gain adjusts the ratio of contributions from predictions and measurements. High Kalman gain values imply more confidence on the model and the predicted state. With low Kalman gain values, the updated state tends more towards the measurement. The update equations for the state vector and the uncertainty matrix are given below.

$$X_{t+1} = \hat{X}_{t+1} + K_t \cdot Y_t \tag{5.13}$$

$$P_{t+1} = (I - K_t \cdot H) \cdot \hat{P}_{t+1} \tag{5.14}$$

As stated in (5.8) Kalman model predicts positions and velocities as well as the size of the blobs. Position information provides a more accurate representation of the tracked object location. Velocity can be utilized to find the future path of the object and detect possible collisions. With the given position and velocity information the current

trajectory of object can be calculated. If the calculated trajectory points towards the ego vehicle that indicates the object carries a collision risk.

Another important feature of using Kalman filter for object tracking is its ability to deal with missing information. If a tracked object does not get detected by the detection methods due to noise, occlusions or imperfections in the detection approach, Kalman filter can continue tracking the object with predictions only. In this case the filter will only use the model to predict the position and velocity of the object and it will not use its measurement and update parts. When the tracked object gets picked up by the detection algorithms again, the filter will continue to compare its predictions with the measurements and update its state estimate.

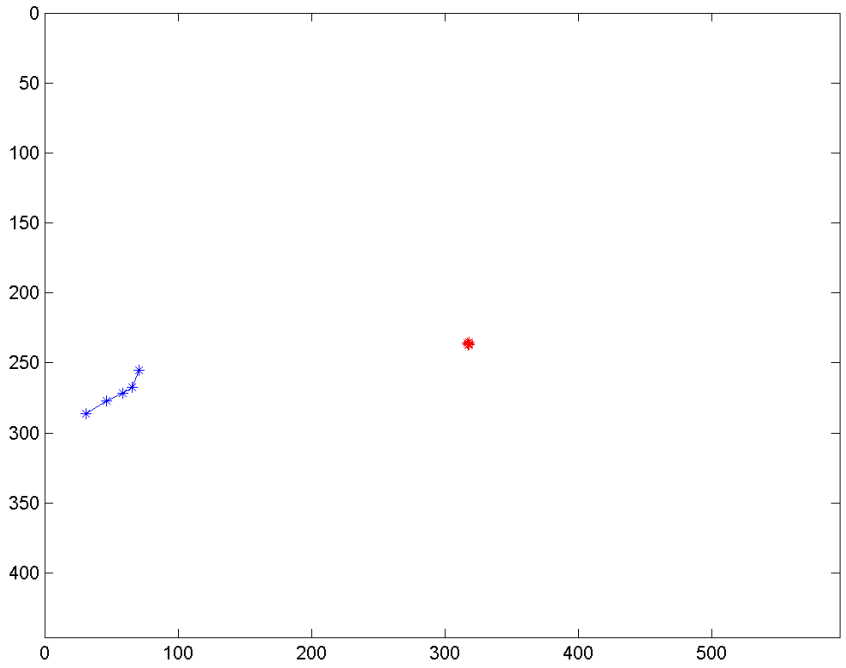

Figure 5.5: Kalman tracking results for the two vehicles in the sample frame

Fig. 5.5 displays the trajectories of the two vehicles on the sample frame, which are obtained by tracking both objects over five frames. The trajectories indicate the car in front is travelling forward with a roughly equal speed and not changing its position relative to the ego vehicle. The car on the left is travelling in the opposite direction and it will go out of the camera field of view from the left side. Fig. 5.6 depicts an overview of Kalman filter method used in this research.

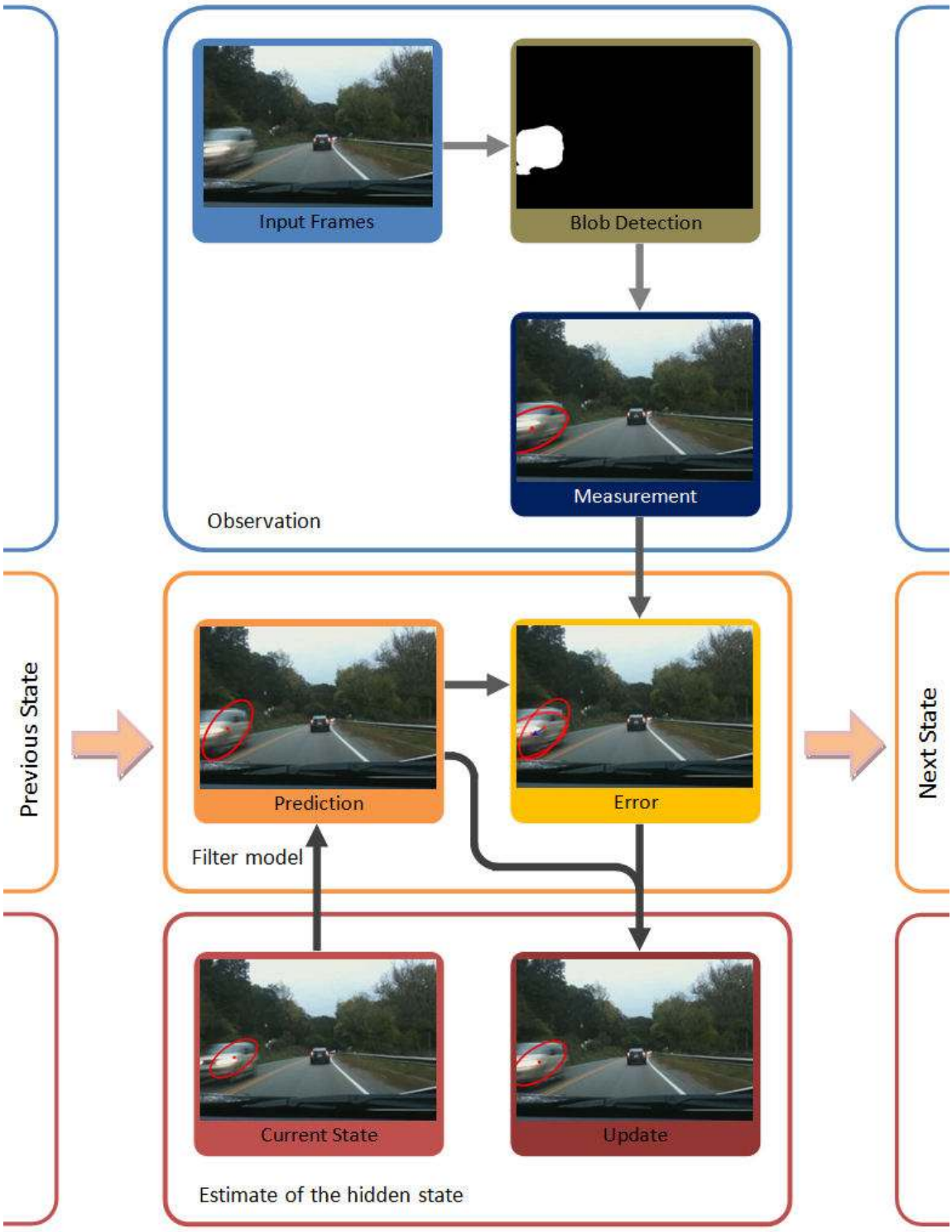

Figure 5.6: Block diagram of the Kalman Filter implementation

6. CONTEXT ENHANCED SCENE GENERATION

In the previous chapters we described computer vision and pattern recognition methods for video analysis. These methods started with extraction of low level information such as color, spectral energy and motion. By combining these information into more advanced descriptors and applying learning methods we achieved high level information such as road surface extraction and tracking of foreground objects. The last step in obtaining a high level abstraction of the video frames that will lead to scene understanding is the combination of these high level information.

The main objective of driving is to keep the vehicle on the road while keeping a safe distance to nearby objects until we reach our destination. A driver assistance system must be able to provide that information. This information can be in context augmented video frames that combine the region identification and object tracking data. Also, based on the tracking data, a system must estimate the trajectory of objects and be aware of any collision risks. This chapter will cover a context enhanced scene generation method based on the object detection and tracking data as well as the road surface identification data.

6.1 Overlaying Road Surface, Vehicle Detection and Tracking Data

So far we have already located vehicles in the scene, tracked their motion and identified the road surface. In order to build a context enhanced frame we need all these information. Also, since we have an estimate of the vehicle motions, we can use that information to mark regions that are near or on the path of the other vehicles. These marked regions will represent the high collision risk areas that the driver needs to avoid. To obtain these regions we use the velocity vector estimates from Kalman output. If the

detected object has a positive v_y velocity component, hence moving towards the bottom of the frame, a gradient is formed starting with the lower boundary of the region with detected region and extending to a length proportional to the magnitude of the velocity estimate. Orientation of this gradient region is parallel to the velocity vector estimate. Fig. 6.1 illustrates this high collision risk regions for the cars in the sample video frame along with the boundaries of the regions containing vehicles and their estimated motion.

If v_y is not positive, which means object is not closing in on the ego vehicle, a small region behind the object is still marked as a high risk area in order to remind the driver to leave a safe distance and not follow too close.

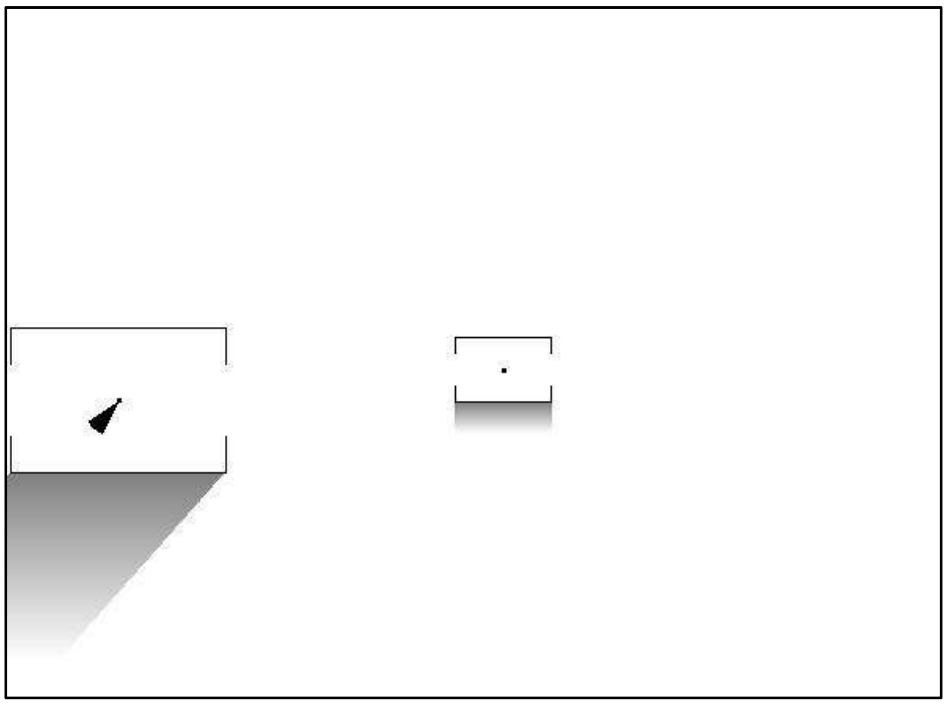

Figure 6.1: Vehicle locations and high collision risk regions

When the object locations and high risk regions are combined with the road surface information, a region mapping as shown below in Fig. 6.2 is obtained.

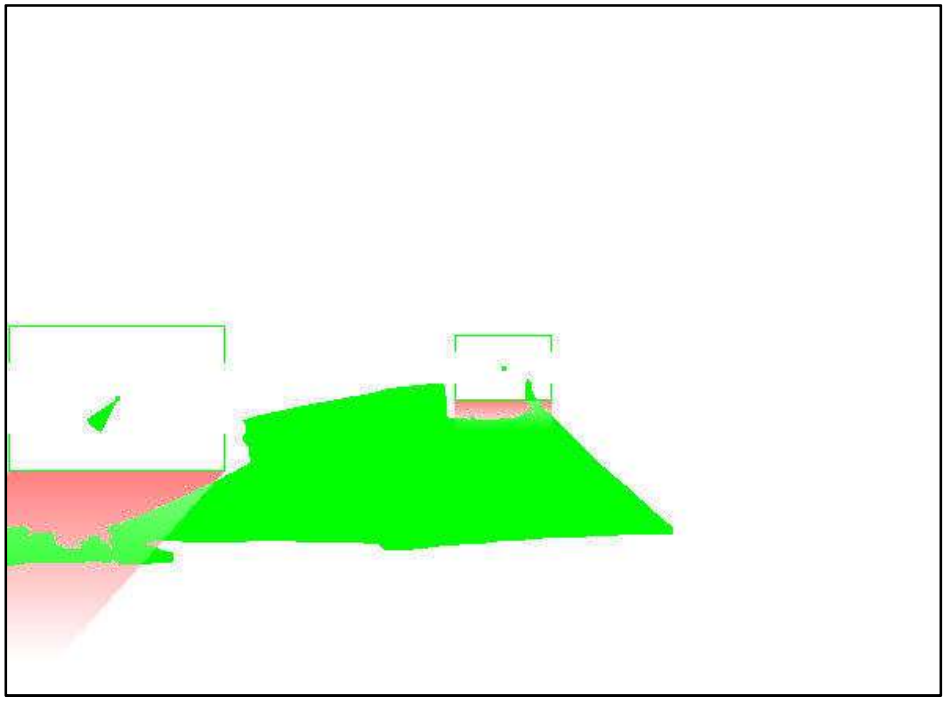

Figure 6.2: Region mapping that displays road surface, object locations, their predicted realtive motions and high collision risk areas

Perhaps the best way to display this information is to overlay it on the visual frame and enhance its contextual information. Fig. 6.3 displays the context enhanced video frame. The context enhanced frame clearly shows the safe regions on the road and informs the driver not to go into regions marked with red where the ego vehicle can end up in the path of another vehicle. The driving scene supplemented by the contextual vehicle and road information can be considered as an example of augmented reality.

Figure 6.3: Context enhanced video frame

6.2 Detection of Possible Collisions and Road Departure

With the predicted relative velocities of objects in the driving scene and their locations, it is possible to find whether or not they are in a collision course with the ego vehicle. However, this is not straightforward since these velocities and locations are in projected camera coordinates instead of real world coordinates. Therefore, we need to find the extent of the ego vehicle and the road in front of the vehicle in camera coordinates. The perspective transformation can be achieved by using the pin hole camera model [64]. Consider the example in Fig. 6.4. Point P in the scene will be projected to the point p on the focal plane of the camera. The x coordinate of p , l_x' , can be calculated by using focal distance f, and x, z coordinates of P; l_x and l_z. From similar triangles (6.1) gives the value of l_x' in terms of f, l_x and l_z. Similarly (6.2) can be derived for calculating the projection in y, z coordinates.

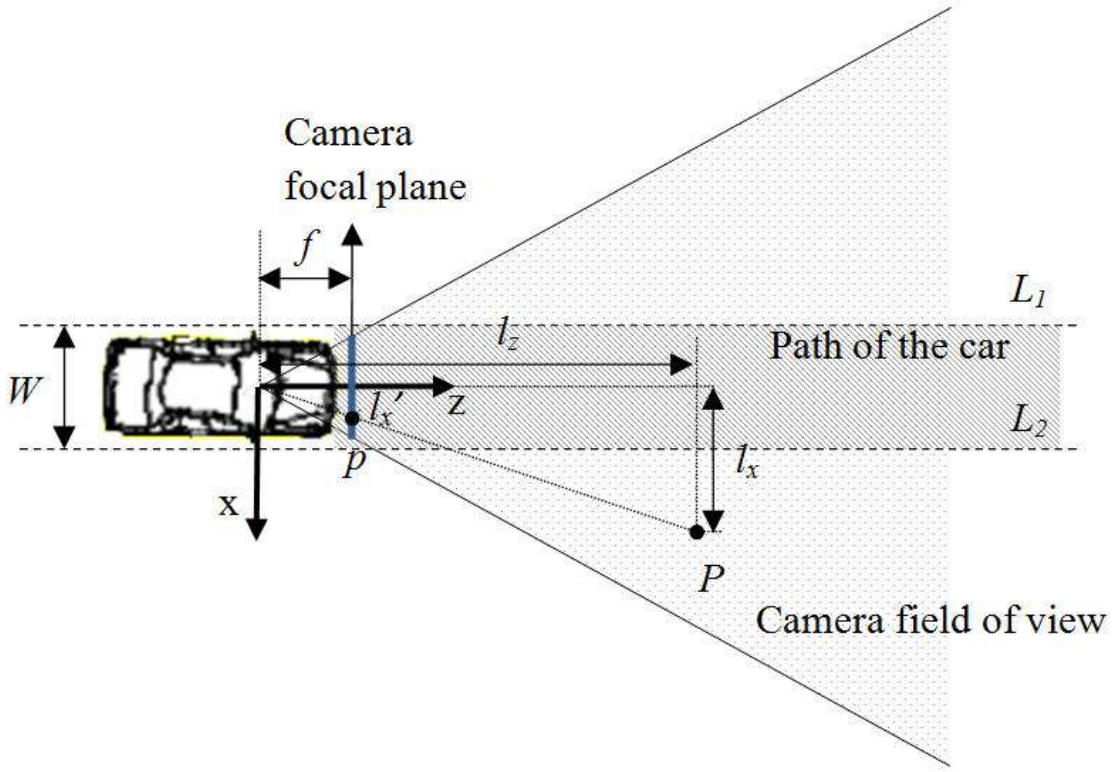

Figure 6.4: Transforming a scene into perspective projection

$$\frac{l_x'}{f} = \frac{l_x}{l_z} \tag{6.1}$$

$$\frac{l_y'}{f} = \frac{l_y}{l_z} \tag{6.2}$$

Overall coordinate transformation is given by:

$$(x, y, z) \Rightarrow \left(\frac{x \cdot f}{z}, \frac{y \cdot f}{z}, f\right) \tag{6.3}$$

With, perspective transformation it is possible to find the equations for the two dashed lines, L_1 and L_2, in Fig. 6.4, which form the boundaries for the road right in front of the car. Anything within these lines is in the path of the vehicle.

Let us assume the camera is 1.5m above the road surface and located halfway between both sides of the vehicle. Assume the center of the camera sensor is the origin and focal length is 0.1m. If we also assume the car width W is 2 meters, the parametric line equations for L_1 and L_2in 3D world coordinates will be:

$$L_1\colon\ x = -1 \quad y = -1.5 \quad z = t \tag{6.4}$$

$$L_2\colon\ x = 1 \quad y = -1.5 \quad z = t \tag{6.5}$$

Applying (6.3) to L_1 and L_2 with $f = 0.1$ gives:

$$L_1\colon\ x = -\frac{1}{10t}y = -\frac{1.5}{10t}z = f \tag{6.4}$$

$$L_2\colon\ x = \frac{1}{10t}y = -\frac{1.5}{10t}z = f \tag{6.5}$$

Changing parametric equations into Cartesian givesL_1 and L_2 in perspective projection plane:

$$L_1\colon\ y = \frac{3}{2}x \tag{6.6}$$

$$L_2\colon\ y = -\frac{3}{2}x \tag{6.7}$$

Even though values from an actual implementation might produce different equations, the assumed values for camera height, focal length and vehicle width are reasonably average values. Therefore, using (6.6) and (6.7) for collision and road

departure warnings should perform well. Fig 6.5 illustrates the L_1 and L_2 in camera perspective coordinates. The triangle formed by L_1, L_2 and the lower boundary of the camera image corresponds to the path of the vehicle (assuming the vehicle is going straight), which is represented by the shaded region in Fig 6.4.

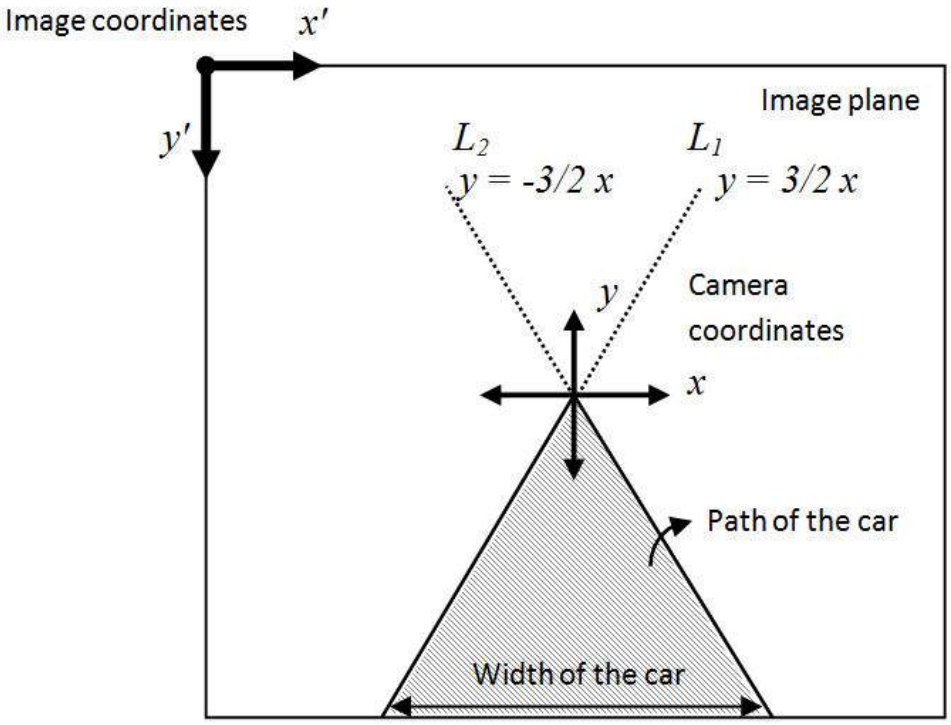

Figure 6.5: Width and path of the vehicle after perspective transformation

Fig. 6.5 shows that the width of the vehicle is 2/3 of the total image width. Since we already have predicted motion vectors and location of objects we can suggest that any object that moves towards the center two thirds of the lower image boundary is a collision risk. Therefore, with this method collision detection is as simple as solving the intersection of two lines.

For the road surface departure, the area selected as the path of the car is compared with the road surface recognition results. Ideally road surface coverage over the vehicle path region should be 100%. If this ratio drops below a certain percentage it is a strong indication that the vehicle is going, or about to go off road. Since the road ahead of the vehicle might be occluded by another vehicle, regions occupied by vehicles are also counted as road surface in this calculation. Figs. 6.6 and 6.7 illustrate collision detection and road surface departure detection methods.

Figure 6.6: Detection of possible collisions based on estimated object motions

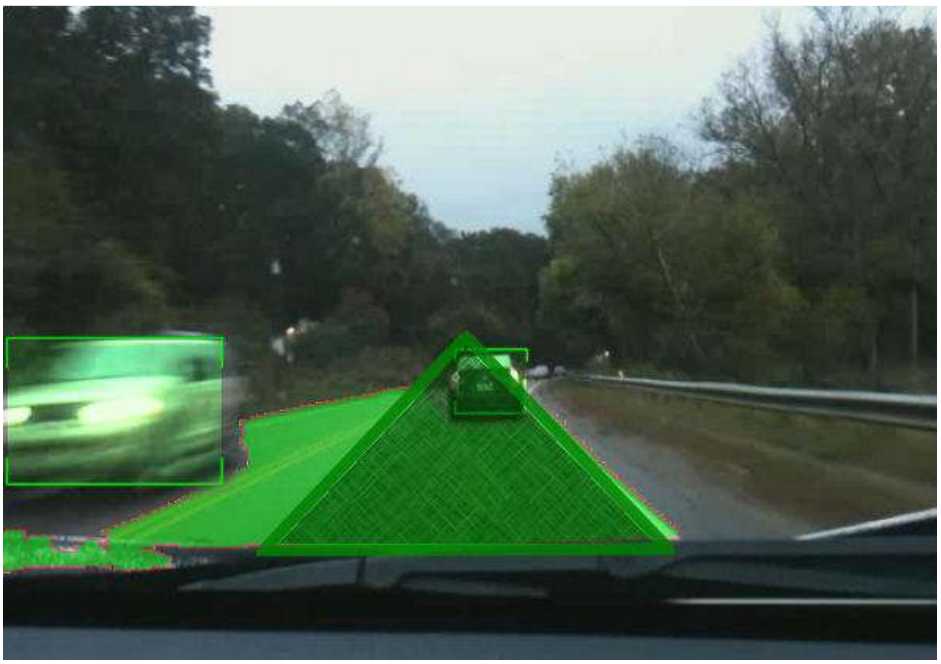

Figure 6.7: Illustration of road departure detection by comparison of detected road surface and vehicle path. Dark green triangle represents the path of the vehicle. It must overlap with the road or other vehicles; otherwise outcome is resolved as road departure.

7. RESULTS

This chapter presents the results obtained by the computer vision and machine learning methods discussed in this dissertation. The methods described herein have been implemented with MATLAB and its image processing toolbox on a personal computer. The methods are tested on several daytime driving videos taken on various types of roads, and on different weather conditions. The roads include city streets highways and county roads. The weather conditions include sunny, overcast, rainy and snowy weather. The video lengths differ from under one minute to several minutes. Tested video resolutions are 640x480 and 1280x720. Another dataset from [68] is also used to obtain accuracy results using ground truth frames. The results show consistency as long as the camera is focused on the road ahead, there are no obstructions in front of the camera and no significant camera shake occurs. Sample frames obtained from these videos can be seen in Fig 7.1.

Figure 7.1: Sample Frames from the videos used in the research

Overall the performance of the fusion, segmentation, vehicle detection, road surface classification and object tracking results are measured. The results are promising

for almost all videos examined. The only video where segmentation and object detection methods produce poor results is taken under rainy conditions, where the camera is unable to focus on the road due to raindrops on the windshield and camera view is frequently blocked by the windshield wipers. Even though the input frames obtained from that video are deemed unreliable, they demonstrated that unimpaired sight is vital for vision systems as it is for human drivers.

7.1 Data Fusion and Image Segmentation and Blob Detection Results

In Chapter 3 the methods for extracting low level image features were presented and in Chapter 4 these low level features were fused using entropy. The purpose of image data fusion is to come up with a fused dataset that is more informative than the components used for fusion. Minimizing mean square error (MSE) between features to be combined maximizes information common to all sources (i.e. redundant information). This improves segmentation and road detection results. As shown later in road detection results, high detection accuracy is achieved.

Segmentation results exceed many known color, texture and intensity based segmentation methods. As the saliency image carries more information than the low level features, segmentation and region growing based on fused saliency generates far better results than the methods that rely on a single spatial or spectral feature. Moreover, blob detection from fused saliency images generates blobs from the most prominent regions of the frames. These prominent regions often correspond to parts of the frame with the highest motion and textural complexity. These regions, in a sense, capture the attention of the vision system just as they would capture the attention of a person. These in fact

confirm our initial claim that, utilizing information fusion to combine low level features into high level descriptors lead to better results. Figs. 7.3 to 7.6 demonstrate these saliency, segmentation and blob detection results.

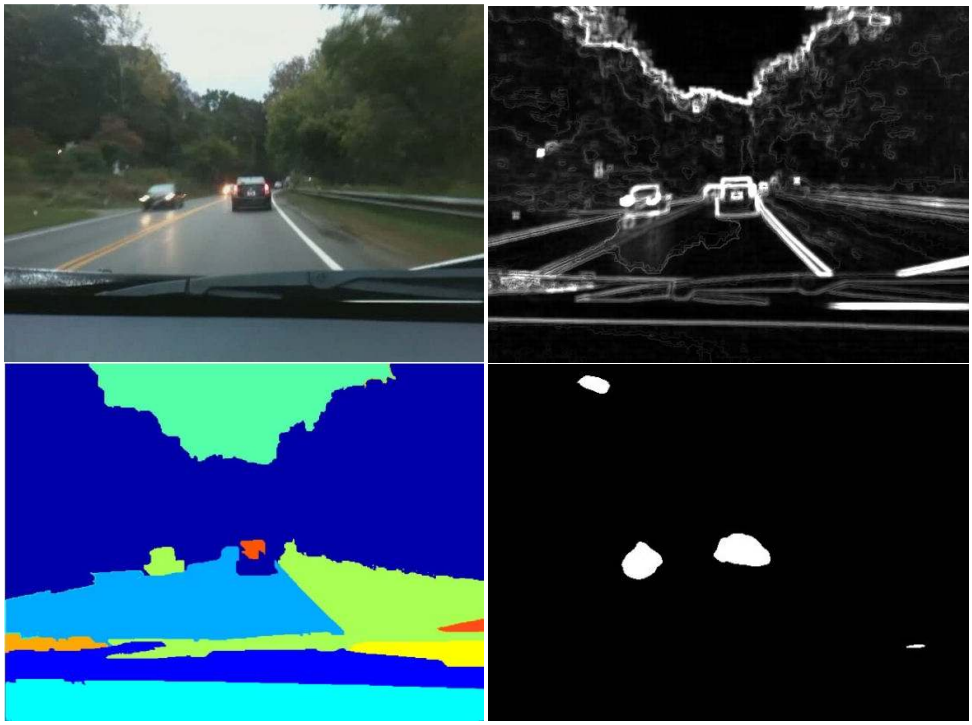

Figure 7.2: Fused saliency, segmentation and blob detection for the given video frame

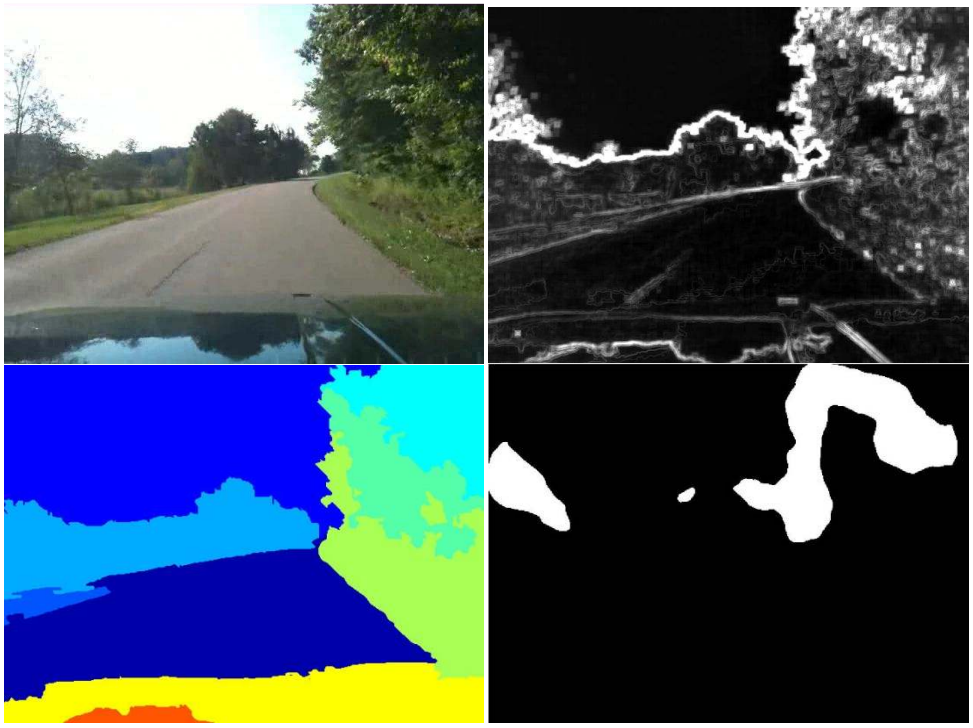

Figure 7.3: Fused saliency, segmentation and blob detection for the given video frame

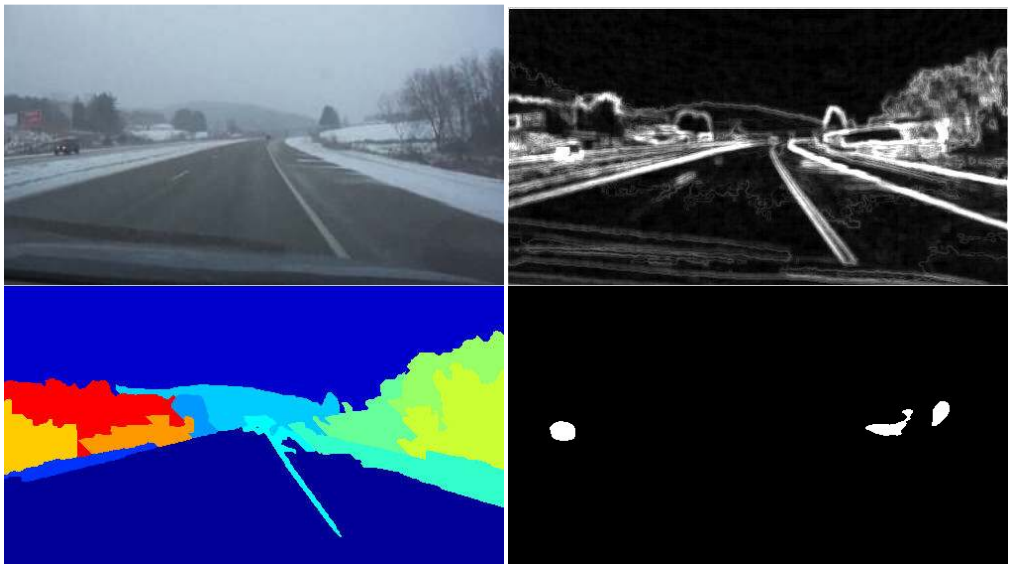

Figure 7.4: Fused saliency, segmentation and blob detection for the given video frame

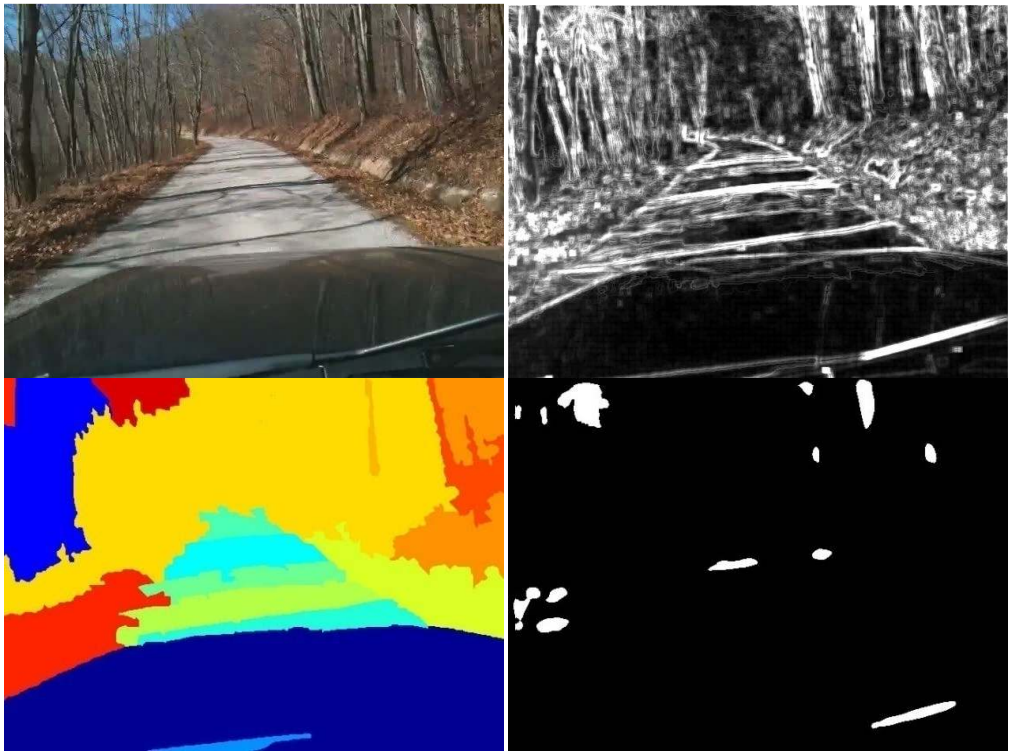

Figure 7.5: Fused saliency segmentation and blob detection for the given video frame

Notice the complicated background in Fig. 7.6. Due to the dark shadows of the trees, the road is segmented into multiple regions instead of one. Segmentation is inadequate in such cases. Therefore learning and recognition algorithms are required for proper road surface detection. Figure 7.7 illustrates the failed segmentation case mentioned earlier. In this case the image being out of focus was already causing problems. When the wiper swipes the windscreen, its motion dominates the saliency map. Following segmentation and blob detection operations are not very accurate after that.

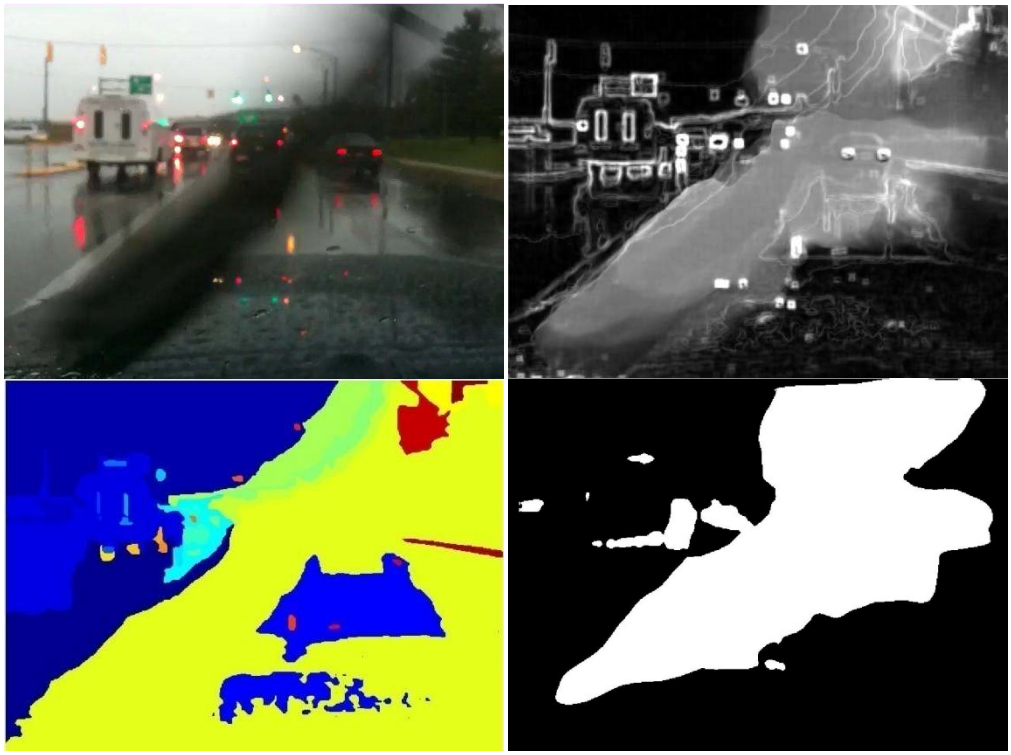

Figure 7.6: Failed segmentation and blob detection due to windscreen wiper

Apart from Fig 7.7, we see segmentations and blob detections display high accuracy. Segmentation results are compatible with the original images and blob detection highlights regions involving vehicles or background regions with dense texture.

7.2 Vehicle Detection Results

The vehicle detection methods applied in this research are mostly parallel to the ones in recent vehicle detection research. As stated in Chapter 2, detecting parallel edges and shadows under the vehicles are common approaches for most published work in this field. However, optical flow is added as an additional feature and search is limited to high saliency blobs in this research. Table 7.1 summarizes the number of vehicle detections for the sample videos used in this research.

Table 7.1: Vehicle detection ratios for the sample video sequences

Video Name	Detected # of Vehicles	Actual # of Vehicles
IMG_0608_001.avi	1	1
IMG_0770.avi	12	12
M4H00005_2.avi	3	3
M4H00005_7.avi	15	16
IMG_0990.avi	3	4
IMG_0768.avi	11	11
IMG_0769.avi	15	16
IMG_0931.avi	0	0
IMG_0610.avi	5	5
Total	65	68

The detection ratios given in Table 7.1 suggest an overall detection accuracy of 95.6% which is in par with the state of the art vehicle detection methods [3] [69] [70]. The missed vehicles in all three cases are far away from the ego vehicle and they are travelling in the same direction. The ego vehicle never gets close enough to capture enough features to detect them. All vehicles travelling in opposite direction and parked on both sides of the roads are detected with the proposed method. There are a few cases of false detections. These can potentially be avoided with the addition of some new features. One of these false detections and a few of other detection results are shown in Fig. 7.8.

Figure 7.7: Sample vehicle detection results. Notice the upper left image has a fence by the road highlighted as a vehicle along with two cars in the image.

7.3 Road Surface Classification and Road Departure Detection

In this research a dynamic Bayesian inference method is developed for learning and classification of road surfaces. As explained in detail in Chapter 5, the method uses maximum likelihood to train road models and test the segments in order to classify them as road surface or background.

The results of the classification can be seen in Figure 7.9. Even though classification accuracy heavily depends on the outcome of segmentation, a well trained model can compensate for minor flaws. For example in Fig. 7.6 the road is segmented into multiple regions due to shadows from nearby trees. Since these regions have similar ontological properties, which generate high similarity measure when compared with the road class model, they are all classified as roads. Fig. 7.9 depicts road classification results from multiple video frames.

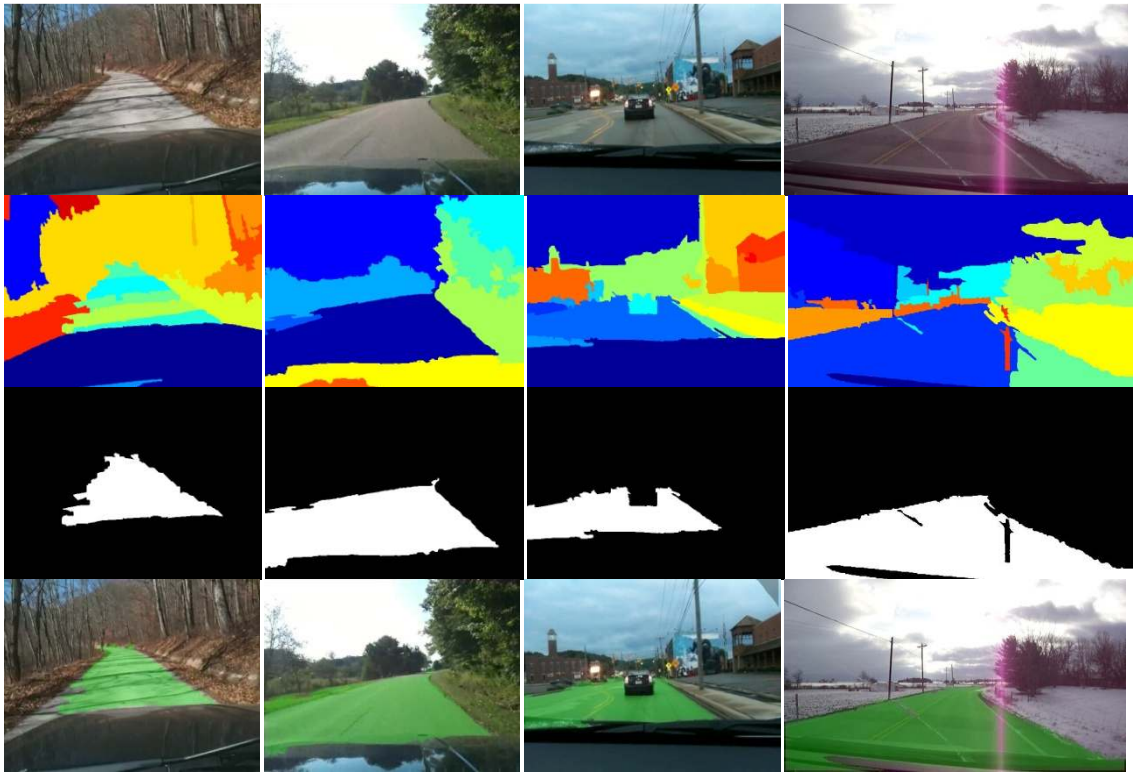

Figure 7.8: Road surface classification results. The first row displays frames from sample videos. Second row is the segmentation results and the third row displays road surface classification results. The last row overlays the detected road surfaces on visual frames.

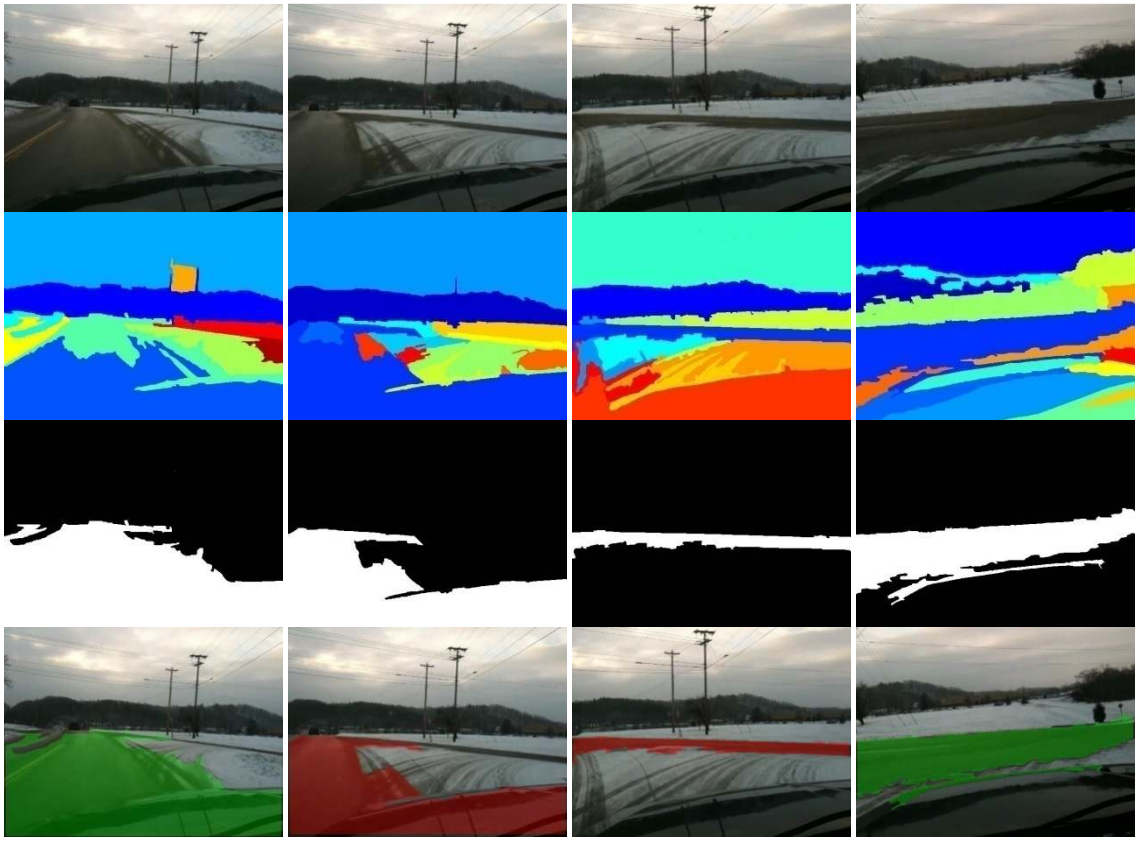

Figure 7.9: Road surface departure detection results. The first row displays frames from a sample video. Second row is the segmentation results and the third row displays road surface classification results. The last row shows road surface departure results, where red color indicates road departure.

In Fig. 7.10 road surface departure detection results, based on method defined in Chapter 6, are shown. In this figure sample frames from a video portion, where the ego vehicle goes over a snow covered road portion to join a side road, are shown. Snow covered road section is not recognized as road and as the road coverage over vehicle path reduces, the method detects that as a road departure. Later, as the vehicle joins the side road, departure warning ceases.

Another set of experiments were done on a data set from [68]. Some video frames on this dataset are manually segmented to form ground truth. Although this sort of ground truth formation can be considered subjective, as long as people who do the segmentation abide certain set of rules based on common sense, results will be consistent with small variations within statistical error margins.

The ground truth images are color coded. Purple tones indicate road and lane markings. Red regions are buildings, yellow is vegetation and gray is the sky. By searching the purple colors on the ground truth images, road surfaces can be obtained.

The segmented images, shown previously, do not have such color coding. Colors only indicate label numbers which are arbitrary. The road surfaces are found by classification using ontological features as explained in chapter 5.

Figures 7.11, and 7.12 demonstrate the results obtained from aforementioned dataset. Dataset contains four different video sequences. Two sample frames from each are shown here. The first row is the input scene. The second row is the saliency frame extracted with the new information fusion method. 3rd row is the segmentation result. 4th row contains the extracted road surface from segmented image. The next row shows the ground truth for the input frame. It is followed by the road surface extracted from the ground truth. The last row shows the augmented scene output with segmentation results.

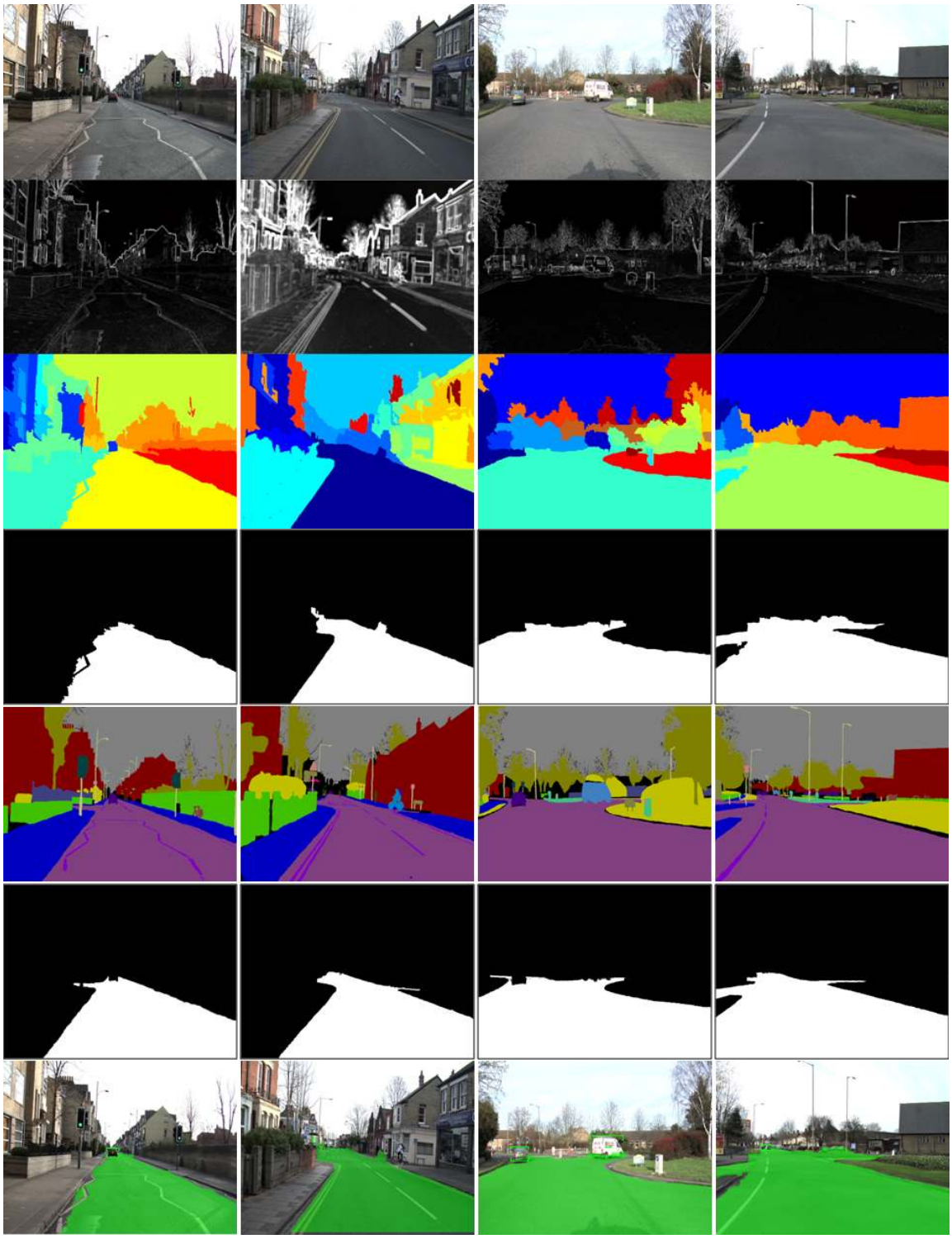

Figure 7.10 Sample frames (row 1), their saliency (row 2), segmentation results (row 3), road detection results (row 4), ground truth (row 5), ground truth of road surface (row 6) and augmented scene results (row 7).

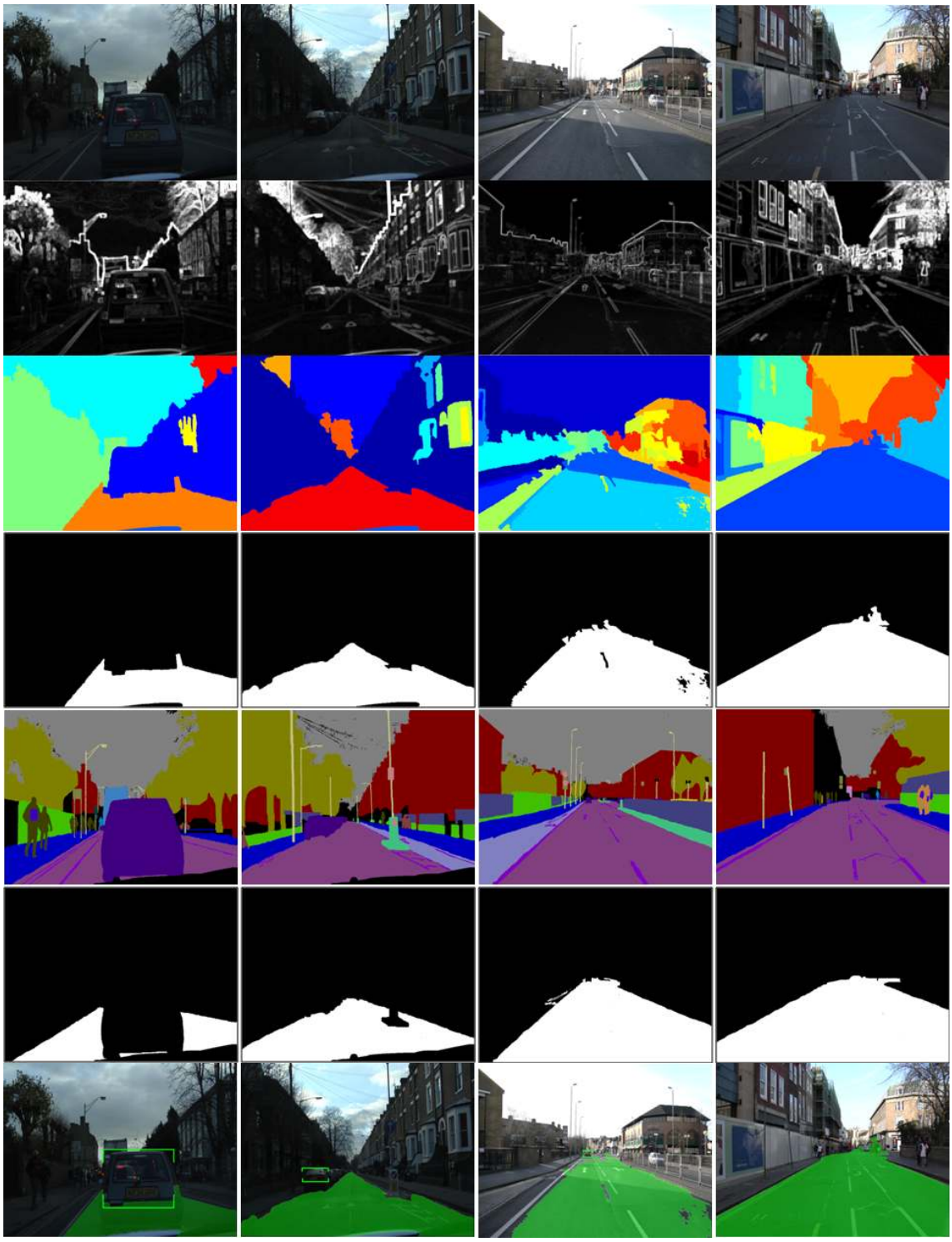

Figure 7.11: Sample frames (row 1), their saliency (row 2), segmentation results (row 3), road detection results (row 4), ground truth (row 5), ground truth of road surface (row 6) and augmented scene results (row 7).

The accuracy of road detection is obtained by comparing the road elements in ground truth image (road, lane markers) to the extracted road surface from segmentation image. Measured accuracy is given by:

$$Road\ Detection\ Accuracy = \frac{1}{M\,N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} f(i,j) \tag{7.1}$$

where M and N are image dimension and f is given by:

$$f(i,j) = 1 \;\; if \; I_R(i,j) = I_G(i,j), \quad f(i,j) = 0 \; otherwise \tag{7.2}$$

Here I_R is the segmented road binary image and I_G is the ground truth for road surface.

Accuracy measurements are taken for a series of frames in each video and averaged to give the overall accuracy.

Table 7.2 below summarizes the road surface extraction accuracy on four videos. Figures 7.12 and 7.13 compare road detection results and ground truth of road surfaces.

Table 7.2: Road surface detection accuracy results

Video Sequence	Road Detection Accuracy	Standard Deviation
005VD	0.9728	0.0194
0006R0	0.9769	0.0208
0016E5	0.9773	0.0264
0001TP	0.9074	0.0762

Figure 7.12 Sample frames (row 1), road detection results (row 2), ground truth of road surface (row 3) and comparison of detection results with ground truth (row 4). Gray regions show mismatches.

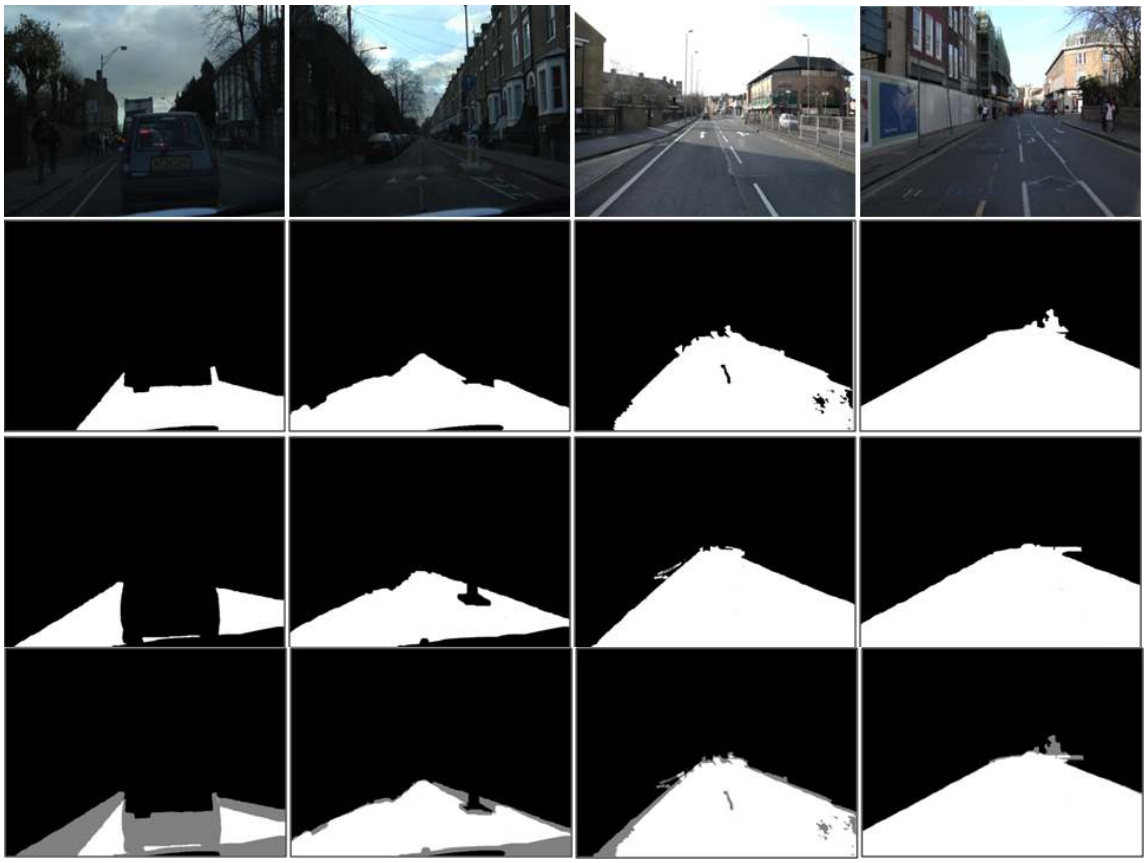

Figure 7.13 Sample frames (row 1), road detection results (row 2), ground truth of road surface (row 3) and comparison of detection results with ground truth (row 4). Gray regions show mismatches.

As seen below detected road surfaces (2nd row) and ground truth (3rd row) show great resemblance. The last row shows the segmentation results and ground truth overlaid. White regions are the parts marked as road on both segmented image and ground truth. Black regions are the non-road segments in both images. The gray parts show mismatches. The gray regions constitute a very small part of the entire overlaid image. As indicated in Table 7.2 overall accuracy is above 90% for all four videos. Three of them are actually around 97% accuracy. The last video is taken on overcast weather with not much light to increase contrast. This results in lower detection accuracy.

7.4 Kalman Tracking &Collision Detection Results

Kalman filter is implemented to track and estimate the motions of detected objects. With blob and vehicle detection methods, which were made possible by high level descriptors obtained from information fusion, we can obtain parameters for Kalman filter. The filter used in this project is a second order filter. Therefore, it estimates the velocities of objects as well as their position. This helps us predict the direction they are headed as well as a more accurate location than the centroids obtained from the blob detection.

Figure 7.14: Sample video sequence for Kalman tracking

Fig 7.15 illustrates Kalman tracking for two objects in the scene given by the frames in Fig. 7.14. Of the two objects, one is travelling in the same direction as the ego vehicle at about the same speed. Hence, its relative position does not change much. The other object is travelling in the opposite direction and its relative speed is gradually increasing due to perspective. This can easily be visualized with Kalman tracking as shown in the Kalman estimated position plot overlaid with the video frame.

Figure 7.15: Kalman tracking of two vehicles

As discussed in Chapter 6, the information obtained from Fig. 7.12 helps us predict the trajectory of tracked vehicles. The objects headed for the ego vehicle are immediately marked as a collision risk. Unfortunately (for the researcher, fortunately for the drivers) there were not very many possible collision events in the sample video dataset. Fig. 7.17 shows the event with the highest collision potential, where the driver of the car in front brakes hard and rapidly comes to a stop. The ego vehicle quickly closes in before coming to a stop as well. During that interval, the car in front seems to be going towards the ego vehicle due their relativistic motion.

Figure 7.16: Frame sequence with a high collision risk event

The implemented method detects the estimated velocity vector of the tracked object points towards the ego vehicle. To display the detection, the frame marking the detected vehicle turns from green to red as shown in Fig. 7.17. In a real implementation there may be audio or tactile warnings as well for the driver.

Figure 7.17: Object is detected as a collision risk

7.5 Context Enhanced Scene Outputs

Context enhanced frames contain the highest level information formed in this research. Since context enhanced frames embody real world information plus additional information obtained by the system they can be considered as augmented reality video frames. Identified road surfaces, tracked objects and their estimated trajectories are displayed in these augmented reality outputs along with the image components from the original frame. A few of these context enhanced scene results are displayed in Figs. 7.18-7.21. The results exhibit fairly good depictions of the driving scenes.

Figure 7.18: Context enhanced scene for the collision risk instance in Fig. 7.17

Figure 7.19: Context enhanced scene in winter driving conditions

Figure 7.20: Context enhanced scene in a residential area intersection

Figure 7.21: Context enhanced scene with road surface departure detection

7.6 Noise Immunity

To measure how well the method does under noisy conditions we introduced artificial Gaussian noise into a sample frame and measured segmentation accuracy with varying degrees of noise. 11 copies of the original frame are made with progressively worsening noise. There is approx. 5 dB SNR drop on each example. Figures 7.22 to 7.24 show those frames and segmentation results along with the comparison to ground truth. The first image is the original frame, followed by noisy frames. The second row is the segmentation and the 3rd row is the road surface detection result. In this example, the same white Gaussian noise with zero-mean and different variances are introduced to all color channels indiscriminately. In the event of different noise sources contaminating color channels differently, the presented experimentation needs to be carried out on the vector-valued color video function as being aimed to pursue in our future research work.

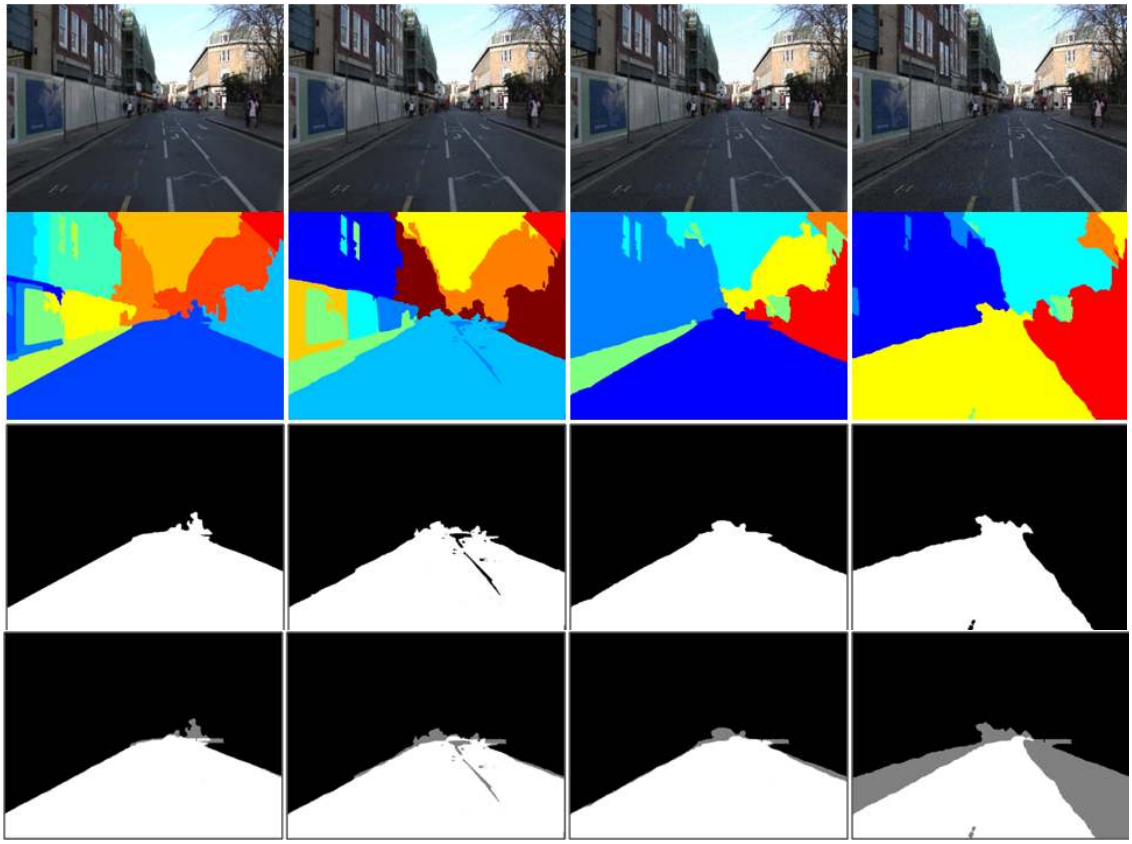

Figure 7.22: Segmentation and road detection results of original image (1st column – image 0) samples degraded with Gaussian noise (images 1, 2 and 3).

Each test image except for the first one is degraded by a Gaussian white noise of varying strength. The signal to noise ratio (SNR) is calculated by:

$$SNR = 10\ log_{10}\left(\frac{P_S}{P_N}\right) \tag{7.3}$$

Where P_S and P_N represent signal and noise power respectively.

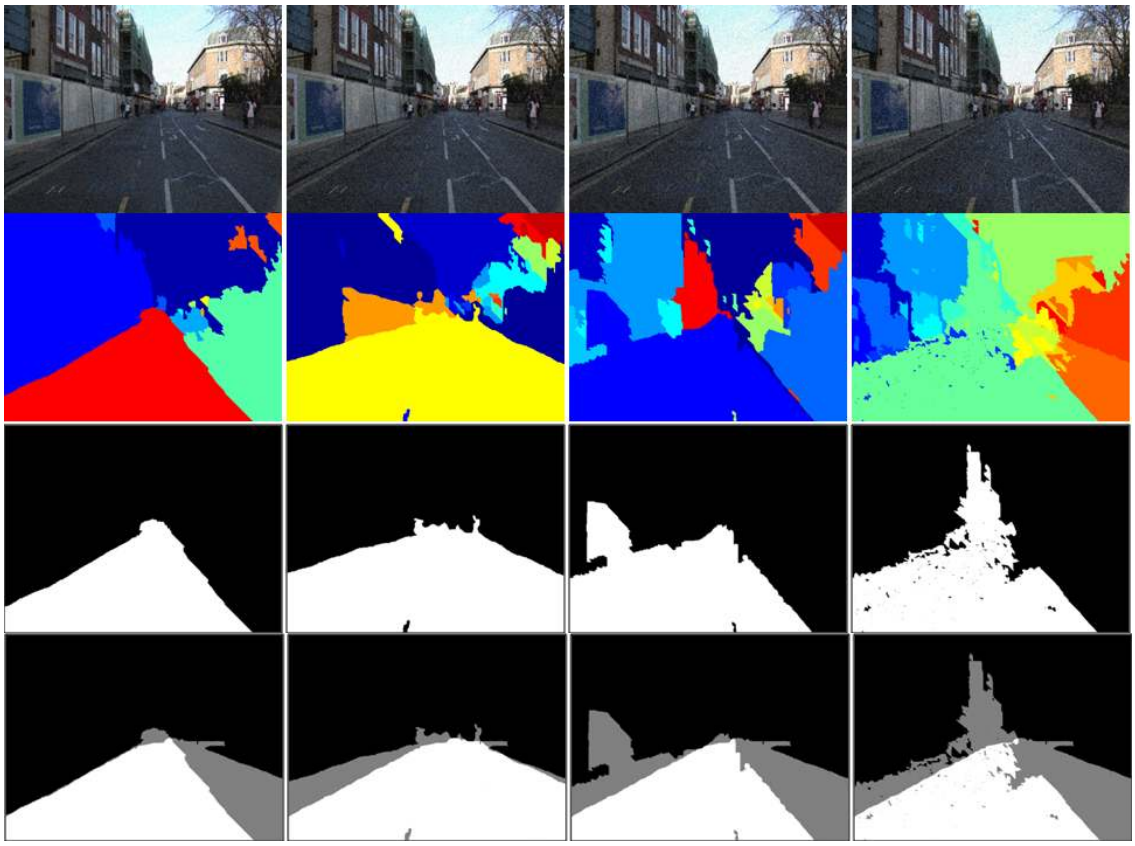

Figure 7.23: Segmentation and road detection results of samples degraded with Gaussian noise (images 4, 5, 6 and 7).

Signal power is given by the mean square of the image intensity.

$$P_S = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j}^{N-1} I^2(i,j) \tag{7.4}$$

Since Gaussian noise is a zero mean random process, its power is given by its variance:

$$P_N = var(N) = {\sigma_N}^2 \tag{7.5}$$

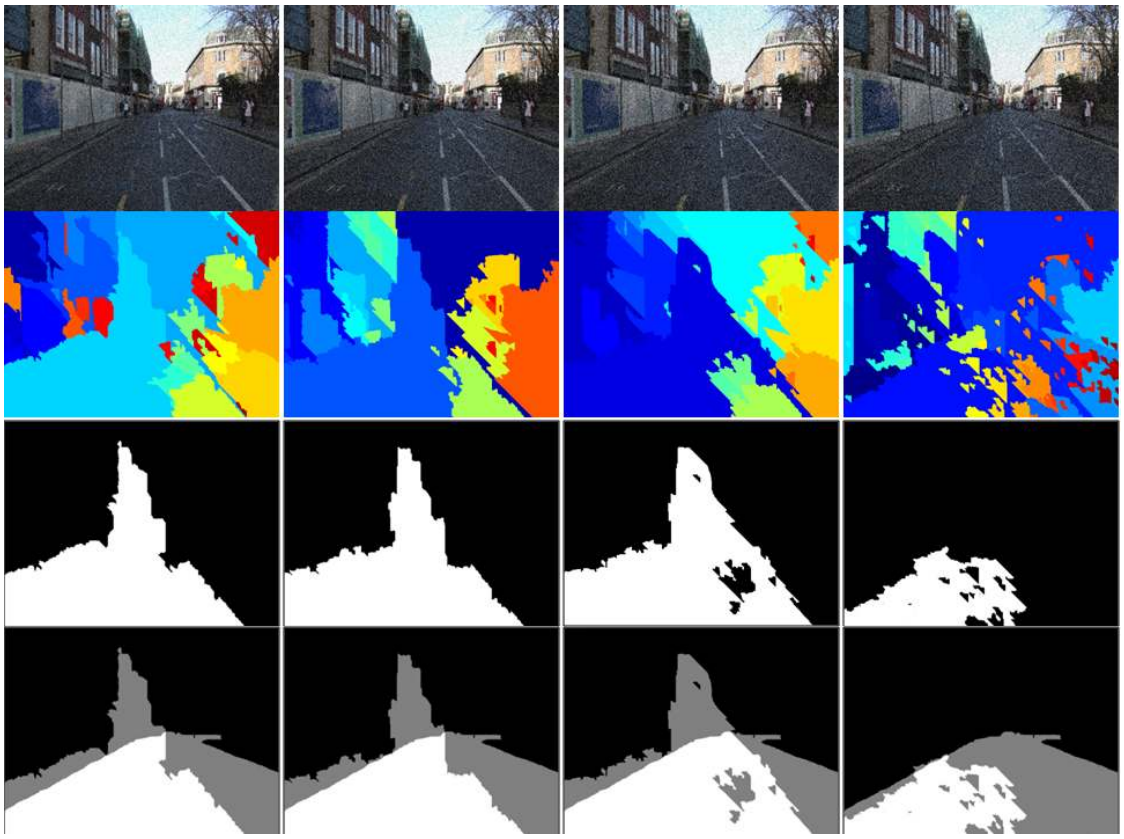

Figure 7.24: Segmentation and road detection results of samples degraded with Gaussian noise (images 8, 9, 10 and 11)

The following plots and table show the SNR of each noisy frame and accuracy of road detection on these frames. As the results indicate even with a 50dB overall drop in SNR, accuracy decreases only by about 20%. Even when the segmentation suffers from extreme noise injected into the frames, road surface extraction is still generating plausible results.

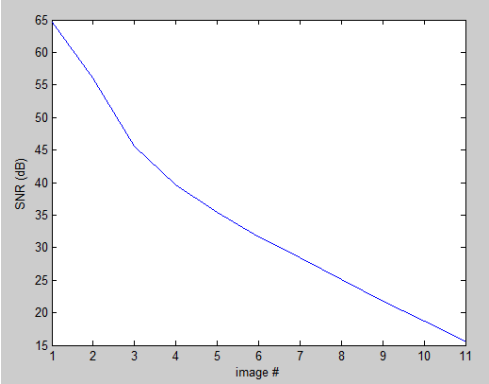

Figure 7.25: SNR of 11 noisy sample frames

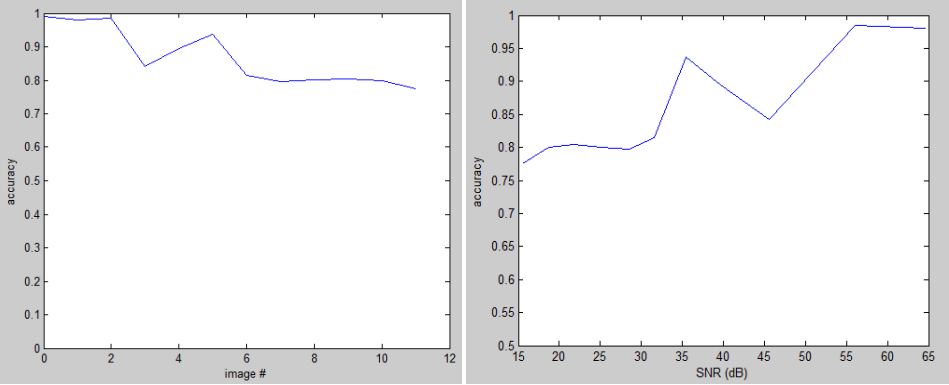

Figure 7.26: Accuracy of road detection versus image number and image SNR

Table 7.3: Noise immunity results

image	SNR	accuracy
0	original	0.9901
1	64.60	0.981
2	56.08	0.9846
3	45.57	0.8423
4	39.64	0.895
5	35.42	0.9374
6	31.58	0.8141
7	28.48	0.7969
8	25.08	0.8002
9	21.73	0.805
10	18.74	0.7998
11	15.55	0.7758

8. CONCLUSIONS AND FUTURE RESEARCH

The motivation behind this dissertation is combining multiple approaches for an image understanding problem, in order to solve it much more effectively. As stated earlier, human vision combines spatial, spectral and temporal data for visual perception. Instead of using one of these sources, combining all of the available information yields better results for human perception and cognition. Similar approach can be beneficial for computer vision as well. Forming an approach to a problem based on the conflicts and agreements of different methods can generate solutions that are much more general than solutions obtained with a single method.

The main contribution of this research is the formation of a computer vision approach that produces higher level image descriptors by fusing low level image features together. These high level descriptors can close the semantic gap in image understanding problems. Utilization of high level features lead to better segmentation, better tracking and eventually better scene understanding. Experimental results of this work indicate mimicking human vision to achieve a computer vision based driver assistance system produces highly robust results. This confirms our initial premise that AI can benefit from simulating natural intelligence.

The objective of this research is the formation of a computer vision based driver assistance system, which can be utilized in autonomous vehicles as well. Apart from the current driver assistance design trends, this method only uses a visual camera instead of various range finding sensors. Such an implementation will reduce the complexity of intelligent vehicle designs as well as their costs. There are existing vision only driver assistance systems. But these are aimed at specific tasks such as lane detection systems.

Development of a more comprehensive visual system will be a big step towards scene awareness.

The method described in this dissertation begins by obtaining low level spatial, spectral and temporal features from video frames. These features are fused into a saliency map that highlights object contours, heavy textures and motion. Information fusion based on optimizing minimum square error between image features maximizes the redundant information contained in the saliency map. Based on this saliency map images are segmented into regions. Moreover, blobs containing high information content are found. Detected blobs that show vehicular features are marked as vehicles and tracked by a Kalman filter. Segmented regions on the other hand, are compared with a learned road model. This road model dynamically updates itself as new roads appear with the changing scenery. Tracked objects and segments identified as road surfaces are utilized in generation of context enhanced video frames.

Context enhanced scene outputs, which are the end results of this research, forms a high degree abstraction of the scene, where the things that matter the most to drivers are detected and highlighted. These include road surface, vehicle positions and trajectories, detection of potentially hazardous situations such as collision risk and road surface departure. All these information are combined with the actual scene forming an augmented reality output. The results are highly informative for a driver as well as a potential vision only intelligent driving system. Presented results demonstrate the effectiveness of combining multiple visual features together in order to acquire more informative features. As the ground truth comparisons indicate road surface detection accuracy is over 90%. Moreover, the proposed method performs well under noise, not

losing much accuracy even with SNR drops of 50dB. Combining features from an image sequence to analyze a driving scene shows a comprehensive computer vision system can be employed in realization of multipurpose driver assistance systems and autonomous vehicles, without any additional sensors.

All experimentation on this research is carried out with daytime driving videos. In the future these methods can be improved to work with nighttime driving scenarios as well. Integration of infrared cameras can further improve the nighttime performance of the system. Another possible development is the use of stereo cameras to obtain depth perception. Stereo vision can significantly increase the information content of the video frames. Objects can be detected and tracked better when their distances to the ego vehicle are known. A rough estimate of object depths can be obtained with mono vision (single camera) systems as well. However, this requires inverse perspective transformation. Including inverse perspective transformation to this method can be another future research direction.

Along with night time and stereo vision, our method is open to many other possible developments. With the rapid advances in intelligent vehicle research and development, more vehicles will be equipped with driver assistance systems and these systems will eventually be replaced with fully autonomous vehicles. With this trend, the need for scene aware vision systems will be more and more essential. We hope that using information fusion in order to simulate human perception and cognition will be a substantial foundation for future intelligent vehicle research.

BIBLIOGRAPHY

[1] P. Norvig, S.J. Russell, *Artificial Intelligence: A Modern Approach*.: Pearson, 2003.

[2] M.Bimpas, G.Thomaidis, M.Tsogas, M.Netto, S.Mammar, A.Beutner, N.Möhler, T.Wirthgen, S.Zipser, A.Etemad, M.Da Lio, R.Cicilloni, A.Amditis, "A Situation-Adaptive Lane-Keeping Support System: Overview of the SAFELANE Approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 617-629, 2010.

[3] Y.Chen, C.Kao, Y.Li, C.Chen, B.Wu, "Detection, A Vision-Based Collision Warning System by Surrounding Vehicles," *KSII Transactions on Internet & Information Systems*, vol. 6, no. 4, pp. 1203-1222, 2012.

[4] M.Celenk, "A color clustering technique for image segmentation," *Computer Vision, Graphics, and Image Processing*, vol. 52, no. 2, pp. 145-170, 1990.

[5] G.B.Coleman, H.C.Andrews, "Image segmentation by clustering," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 773-785, 1979.

[6] R.W.Picard, F.Liu, "Periodicity, directionality, and randomness: Wold features for image modeling and retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 7, pp. 722-733, 1996.

[7] D.K.Yau, A.K.Elmagarmid, W.G.Aref, F.Jianping, "Automatic image segmentation by integrating color-edge extraction and seeded region growing," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1454-1466, 2001.

[8] M.Celenk, Q.Fu, "Video Object Segmentation Using Spatio-Temporal Information and Marked-Watershed Operation," *International Conference on Image Processing, Computer Vision & Pattern Recognition, IPCV*, pp. 593-598, 2013.

[9] P.H. Lewis, P.G. Enser, C.J. Sandom, J.S. Hare, "Mind the Gap: Another look at the problem of the semantic gap in image retrieval," *Proc. SPIE, Multimedia Content Analysis, Management, and Retrieval*, vol. 6073, pp. 607309-607309, 2006.

[10] J.Song, F.Y.Wang, W.Niehsen, N.Zheng, L.Li, "New developments and research trends for intelligent vehicles," *IEEE Intelligent Systems*, vol. 20, no. 4, pp. 10- 14, 2005.

[11] J.Markoff, "Google cars drive themselves, in traffic," *The New York Times 10*, vol. A1, 2010.

[12] M. Fakher el Deen, S. El-Kader, M.M. Nabeel, "Intelligent Vehicle Recognition based on Wireless Sensor Network," *International Journal of Computer Science Issues*, vol. 10, no. 4, pp. 164-174, 2013.

[13] M.Bolduc, "Tech Blog: Mass produced driverless cars in 10 years?," *[Online] Available: http://www.tradingfloor.com/posts/tech-blog-mass-produced-driverless-cars-in-10-years-19721628 [Accessed 14-Jan-2014]*, 2012.

[14] S.Y.Cheng, M.M.Trivedi, A.Doshi, "A novel active heads-up display for driver assistance," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 91, no. 1, pp. 85-93, 2009.

[15] J.R.Raol, *Multi-Sensor Data Fusion with MATLAB*.: CRC Press, 2009.

[16] H.B.Mitchell, *Multi-Sensor Data Fusion*.: Springer-Verlag, 2007.

[17] S.Narayanan, A.C.Muller, "Cognitively-engineered multisensor image fusion for military applications," *Information Fusion*, vol. 10, no. 2, pp. 137-149, 2009.

[18] M.Celenk, M.Altun, "Image Fusion Based Video Content Enhancement for Surveillance and Tracking," *Journal of Graphics, Vision and Image Processing, GVIP*, vol. 12, no. 2, pp. 9-14, 2012.

[19] R.Laganière, Z.Liu, "Context enhancement through infrared vision: a modified fusion scheme," *Signal, Image and Video Processing*, vol. 1, no. 4, pp. 293-301, 2007.

[20] D.A.Fay, W.D.Ross, A.M.Waxman, D.B.Ireland, J.P.Racamato, M.Aguilar, "Real-time Fusion of Low-Light CCD and Uncooled IR Imagery for Color Night Vision," *Proc. SPIE*, vol. 3364, no. 124, pp. 124-135, 1998.

[21] K.Doi, H.MacMahon, D.Hassell, M.Giger, A.Kano, "Digital Image Subtraction of Temporally Sequential Chest," *Medical Physics*, vol. 21, no. 3, pp. 453-461, 1994.

[22] J.B.Choquel, B.Fassinut-Mombot, "A new probabilistic and entropy fusion approach for management of information sources," *Information Fusion*, vol. 5, no. 1, pp. 35-47, 2004.

[23] S.Shamai, S.Verdú, D.Guo, "Mutual information and minimum mean-square error in Gaussian channels," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1261-1282, 2005.

[24] M.D.Weir, G.B.Thomas, *Thomas' Calculus 11th Edition*.: Addison Wesley, 2005.

[25] J.S.Lim, *Two-Dimensional Signal and Image Processing*.: Prentice Hall, 1990.

[26] S.Watanabe, *Pattern Recognition: Human and Mechanical*.: Wiley, 1985.

[27] C.M.Bishop, *Pattern Recognition and Machine Learning*.: Springer, 2006.

[28] K.Koutroumbas, S.Theodoridis, *Pattern Recognition. 4th ed.*: Academic Press, 2009.

[29] J.V.Stone, "Footprints Sticking Out of the Sand (Part II): Children's Bayesian Priors For Shape and Lighting Direction," *Perception*, vol. 40, no. 2, pp. 175-190, 2011.

[30] D.Purves, "A Primer on Probabilistic Approaches to Visual Perception," *[Online] http://www.purveslab.net/research/primer.html [Accessed:3-Oct-2013]*, 2005.

[31] J.B.Tenenbaum, "Bayesian modelling of human concept learning," *Advances in Neural Information Processing Systems*, vol. 11, 1999.

[32] I.Kompatsiaris, M.G.Strintzis, V.Mezaris, "Region-based image retrieval using an object ontology and relevance feedback," *EURASIP Journal on Applied Signal Processing*, pp. 886-901, 2004.

[33] I.Kompatsiaris, M.G.Strintzis, V.Mezaris, "An ontology approach to object-based image retrieval," *Image Processing, ICIP 2003*, pp. II-511, Proceedings of International Conference on Image Processing, ICIP 2003. 2003.

[34] S.Roweis, "A Unifying Review of Linear Gaussian Models," *Neural computation*, vol. 11, no. 2, pp. 305-345, 1999.

[35] P.Martinet, J.Gallice, F.Jurie, "A global road scene analysis system for autonomous vehicles," *IFAC Conference on Intelligent Autonomous Vehicle*, pp. 19-24, 1995.

[36] R.Bishop, "A survey of intelligent vehicle applications worldwide," *Proceedings of the IEEE Intelligent Vehicles Symposium*, vol. 4, pp. 25-30, 2000.

[37] L.Bombini, A.Broggi, P.Zani, P.Cerri, P.Grisleri, P.Medici, M.Bertozzi, "GOLD: A framework for developing intelligent-vehicle vision applications," *IEEE Intelligent Systems*, vol. 23, no. 1, pp. 69-71, 2008.

[38] M.Pasquier, T.X.P.Diem, "From operational to tactical driving: A hybrid learning approach for autonomous vehicles," *IEEE 10th International Conference on Control, Automation, Robotics and Vision, ICARCV08*, pp. 285-290, 2008.

[39] X.Li, J.Wan, C.Qiao, C.Wu, A.Wagh, "Human centric data fusion in vehicular cyber-physical systems," *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 684-689, 2011.

[40] E.Egea-Lopez, J.B.Tomas-Gabarron, J.Garcia-Haro, Z.J.Haas, C.Garcia-Costa, "A stochastic model for chain collisions of vehicles equipped with vehicular communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 503-518, 2012.

[41] J.Huang, H.Tan, "DGPS-based vehicle-to-vehicle cooperative collision warning: Engineering feasibility viewpoints," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 415-428, 2006.

[42] F.C.A.Groen, M.B.VanLeeuwen, "Vehicle Detection with a Mobile Camera: Spotting Midrange, Distant, and Passing Cars," *IEEE Robotics & Automation Magazine*, vol. 12, no. 1, pp. 37-45, 2005.

[43] F.Collange, F.Jurie, P.Martinet, X.Clady, "Object Tracking with a Pan-Tilt-Zoom Camera: Application to Car Driving Assistance," *International Conference on Robotics and Automation*, vol. 2, pp. 1653-1658, 2001.

[44] R.Danescu, T.Marita, F.Oniga, C.Pocol, S.Sobol, C.Tomiuc, C.Vancea, M. M.Meinecke, T.Graf, T.B.To, M.A.Obojski, S.Nedevschi, "A Sensor for Urban Driving Assistance Systems Based on Dense Stereovision," *IEEE Intelligent Vehicles Symposium*, pp. 276-283, 2007.

[45] T.Marchand, P.Bouthemy, G.Lefaix, "Motion-Based Obstacle Detection and Tracking for CarDriving Assistance," *Proc. 16th Int. Conf. Pattern Recognition*, vol. 4, pp. 74-77, 2002.

[46] C.Koutsimanis, M.Tsogas, A.Amditis, A.Polychronopoulos, "Prediction of unintentional lane departure using evidence theory," *IEEE 8th International Conference on Information Fusion*, vol. 2, pp. 1396-1403, 2005.

[47] N.Srinivasa, "Vision-based vehicle detection and tracking method for forward collision warning in automobiles," *IEEE Intelligent Vehicle Symposium*, pp. 626-631, 2002.

[48] F.Bunyak, P.Kumar, I.Ersoy, S.Jaeger, K.Ganguli, A.Haridas, J.Fraser, R.M.Rao, G.Seetharaman, K.Palaniappan, "Efficient feature extraction and likelihood fusion for vehicle tracking in low frame rate airborne video," *13th Conf. on Information fusion (FUSION)*, pp. 1-8, 2010.

[49] D.Munoz, J.A.Bagnell, M.Hebert, O.Miksik, "Efficient temporal consistency for streaming video scene analysis," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 133-139, 2013.

[50] J.Liu, M.Shah, Y.Yang, "Video scene understanding using multi-scale analysis," in *IEEE 12th International Conference on Computer Vision*, 2009, pp. 1669-1676.

[51] T.Gevers, Y.LeCun, A.M.Lopez, J.M.Alvarez, "Road scene segmentation from a single image," *Computer Vision–ECCV*, pp. 376-389, 2012.

[52] C.Kotsiourou, A.Amditis, A.Bolovinou, "Dynamic road scene classification: Combining motion with a visual vocabulary model," *16th International Conference on information Fusion (FUSION)*, pp. 1151-1158, 2013.

[53] M.Rousson, R.Deriche, J.Weickert, T.Brox, "Unsupervised segmentation incorporating colour, texture, and motion," *Computer analysis of images and patterns*, pp. 353-360, 2003.

[54] M.Rousson, R.Deriche, J.Weickert, T.Brox, "Colour, texture, and motion in level set based segmentation and tracking," *Image and Vision Computing*, vol. 28, no. 3, pp. 376-390, 2010.

[55] T.P.Breckon, I.Katramados, A.Kheyrollahi, L.Bordes, "Adaptive object placement for augmented reality use in driver assistance systems," *[Online] Available: http://breckon.eu/toby/publications/papers/bordes11ar.pdf [Accessed 23-Feb-2014]*, 2011.

[56] I.Thouvenin, V.Frémont, V.Cherfaoui, P.George, "DAARIA: Driver assistance by augmented reality for intelligent automobile," *IEEE Intelligent Vehicles Symposium (IV)*, pp. 1043-1048, 2012.

[57] J.N.Said, C.Y.Suen, M.Cheriet, "A recursive thresholding technique for image segmentation," *IEEE Transactions on Image Processing*, vol. 7, no. 6, pp. 918-921, 1998.

[58] N.Otsu, "A threshold selection method from gray level histograms," *IEEE Trans. Syst. Man Cybernetics*, vol. 9, pp. 62-66, 1979.

[59] J.R.Parker, *Algorithms for image processing and computer vision*.: John Wiley & Sons, 2010.

[60] T.Kanade, B.Lucas, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proc. Seventh International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

[61] B.Schunck, B.Horn, "Determining Optical Flow," *Artificial Intelligence*, vol. 17, pp. 185-203, 1981.

[62] S.Roth, J.Lewis, M.J.Black, D.Sun, "Learning Optical Flow," *Proc. European Conf. Computer Vision*, vol. 3, pp. 83-97, 2008.

[63] D.J.Lee, B.E.Nelson, J.K.Archibald, B.B.Edwards, Z.Wei, "FPGA-Based Embedded Motion Estimation Sensor," *Int. Journal of Reconfigerable Computing*, vol. 2008, no. 636145, p. 8, 2008.

[64] J.P.Siebert, B.Cyganek, *An introduction to 3D computer vision techniques and algorithms*.: John Wiley & Sons, 2011.

[65] F.Farrokhnia, A.K.Jain, "Unsupervised texture segmentation using Gabor filters," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 14-19, 1990.

[66] C.Stauffer, W.E.L.Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," *Proc. Conf. Comp. Vision and Pattern Rec*, vol. 2, pp. 246-252, 1999.

[67] D.S.Lee, "Effective Gaussian Mixture Learning for Video Background Subtraction," *IEEE Trans. Pattern Analysis & Machine Intelligence*, vol. 27, no. 5, pp. 827-832, 2005.

[68] J.Fauqueur, R.Cipolla, G.J.Brostow, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88-97, 2009.

[69] G.Bebis, R.Miller, Z.Sun, "Monocular precrash vehicle detection: features and classifiers," *IEEE Trans. image processing*, vol. 15, no. 7, pp. 2019-2034, 2006.

[70] S.Y.Oh, J.K.Kang, Y.W.Ryu, S.Y.Kim, "Front and rear vehicle detection and tracking in the day and night times using vision and sonar sensor fusion," *Proc. IEEE Intelligent Robots and Systems Conference*, pp. 2173-2178, 2005.

APPENDIX A: DERIVATION OF MAXIMUM LIKELIHOOD FROM BAYES RULE

Let us start with the assumption that we have two classes with a priori probabilities $P(C_1)$ and $P(C_2)$. Class probability distribution functions (pdf) are then given by: $p(x|C_1)$and $p(x|C_2)$. For a given x to find the probabilities $P(C_1|x)$ and $P(C_2|x)$and find out which one is more likely we need Bayes rule. According to Bayes rule if individual probabilities are known, a conditional probability can be derived from the other:

$$P(C_i|x) = \frac{p(x|C_i)\,P(C_i)}{p(x)} \tag{A.1}$$

Bayesian decisions are made with inequalities. For a given event x_0 if probability of C_1 is higher than C_2, x_0 belongs to class 1.

$$P(C_1|x_0) \lesseqgtr P(C_2|x_0) \tag{A.2}$$

Starting with the inequality in (A.2)and applying Bayes Rule gives:

$$P(C_1|x) \lesseqgtr P(C_2|x) = \frac{p(x|C_1)\,P(C_1)}{p(x)} \lesseqgtr \frac{p(x|C_2)\,P(C_2)}{p(x)} \tag{A.3}$$

$$p(x|C_1)\,P(C_1) \lesseqgtr p(x|C_2)\,P(C_2) \tag{A.4}$$

Since the features are assumed to have normal distributions and there are multiple features, the classes are modeled by multidimensional Gaussian probability distributions as in

$$P(x|C_i) = \frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma_i|}}\, e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_i)^T \Sigma^{-1} (\vec{x}-\vec{\mu}_i)} \tag{A.5}$$

where x is the feature vector, n is the number of dimensions,Σ_i is the class covariance matrix and μ_i is the class mean. Then (A.4) can be rewritten as:

$$\frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma_1|}}\, e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_1)^T{\Sigma_1}^{-1}(\vec{x}-\vec{\mu}_1)}P(C_1) \lessgtr \frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma_2|}}\, e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_2)^T{\Sigma_2}^{-1}(\vec{x}-\vec{\mu}_2)}P(C_2)$$

(A.6)

To simplify, natural logarithms of both sides are taken:

$$-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln(|\Sigma_1|) - \frac{1}{2}(\vec{x}-\vec{\mu}_1)^T{\Sigma_1}^{-1}(\vec{x}-\vec{\mu}_1) + \ln\left(P(C_1)\right) \lessgtr -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln(|\Sigma_2|) - \frac{1}{2}(\vec{x}-\vec{\mu}_2)^T{\Sigma_2}^{-1}(\vec{x}-\vec{\mu}_2) + \ln\left(P(C_2)\right)$$

(A.7)

The first term $-\frac{n}{2}\ln(2\pi)$ is a constant and it will be the same for all classes. It can be ignored. $-\frac{1}{2}\ln(|\Sigma_i|)$ and $\ln\left(P(C_i)\right)$ are class dependent constants

Bayesian discriminant function (i.e., similarity measure) is then given by:

$$y_i(\vec{x}) = -\frac{1}{2}(\vec{x}-\vec{\mu}_i)^T{\Sigma_i}^{-1}(\vec{x}-\vec{\mu}_i) - \frac{1}{2}\ln(|\Sigma_i|) + \ln\left(P(C_i)\right) \qquad \text{(A.8)}$$

$$y_1(\vec{x}) > y_2(\vec{x}) \rightarrow \vec{x} \in C_1 \qquad \text{(A.9)}$$

If classes are assumed to be equiprobable the last term: $\ln\left(P(C_i)\right)$ can be eliminated as well. Decision boundaries between classes are found by finding curves where discriminant functions are equal to each other.

$$y_i(\vec{x}) - y_j(\vec{x}) = 0 \Longrightarrow y_{ij}(\vec{x}) \qquad \text{(A.10)}$$

APPENDIX B: KALMAN MATRICES

Kalman state vector:

$$X_t = [x, y, A, v_x, v_x, A']^T \tag{B.1}$$

Where *x*, *y* are coordinates of the blob centroid, *A* is the scale of the blob, v_x, v_y are changes in *x*, *y* per frame i.e. velocities, and *A'* is the change in blob size

Kalman forcing matrix (i.e. State transition matrix):

$$F = \begin{bmatrix} 1 & 0 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 & dt & 0 \\ 0 & 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{B.2}$$

where *dt* is the frame difference. *dt* = 1 if consecutive frames are used.

Process uncertainty vector (process noise):

$$w = \begin{bmatrix} \frac{dt^2}{2} & \frac{dt^2}{2} & \frac{dt^2}{2} & 0 & 0 & 0 \end{bmatrix}^T \tag{B.3}$$

Process uncertainty covariance:

$$Q = \begin{bmatrix} dt^4/4 & 0 & 0 & dt^3/2 & 0 & 0 \\ 0 & dt^4/4 & 0 & 0 & dt^3/2 & 0 \\ 0 & 0 & dt^4/4 & 0 & 0 & dt^3/2 \\ dt^3/2 & 0 & 0 & dt^2 & 0 & 0 \\ 0 & dt^3/2 & 0 & 0 & dt^2 & 0 \\ 0 & 0 & dt^3/2 & 0 & 0 & dt^2 \end{bmatrix} \tag{B.4}$$

Initial prediction uncertainty:

$$P_{init} = \begin{bmatrix} 0.4 & 0 & 0 & 0.4 & 0 & 0 \\ 0 & 0.4 & 0 & 0 & 0.4 & 0 \\ 0 & 0 & 3 & 0 & 0 & 2 \\ 0.4 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0.4 & 0 & 0 & 0.7 & 0 \\ 0 & 0 & 1.4 & 0 & 0 & 1.6 \end{bmatrix} \quad \text{(B.5)}$$

Measurement matrix:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{(B.6)}$$

Measurement uncertainty vector:

$$v = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T \quad \text{(B.7)}$$

Measurement uncertainty covariance:

$$R = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{bmatrix} \quad \text{(B.8)}$$

Thesis and Dissertation Services