# Intelligent Headlight Control Using Camera Sensors

Ying Li and Sharath Pankanti

IBM T.J. Watson Research Center, NY
{yingli, sharat}@us.ibm.com

## ABSTRACT

This paper describes our recent work on intelligently controlling a vehicle's headlights using a forward-facing camera sensor. Specifically, we aim to automatically control its beam state (high beam or low beam) during a night-time drive based on the detection of oncoming/overtaking/leading traffics as well as urban areas from the videos captured by the camera. A three-level decision framework is proposed which includes various types of image and video content analysis, an SVM-based learning mechanism and a frame-level decision making mechanism. Both video and context information have been exploited to accomplish the task. Online test drives as well as offline evaluations on tens of videos have validated the robustness and effectiveness of the proposed system.

**Categories and Subject Descriptors:** I.5.2 [Pattern Recognition]: Pattern Analysis

**General Terms:** algorithm, design, experimentation

**Keywords:** Intelligent headlight control, camera sensor, SVM, machine learning, image/video content analysis, driving context

## 1. INTRODUCTION

Driving at night is usually more dangerous than driving during the day. Pedestrians and cyclists on the roads are especially at high risk due to the limited visibility of motorists at night. In fact, a recent study by University of Michigan Transportation Research Institute (UMTRI) found that pedestrians are about four to six times more vulnerable at night than during the day [1]. This raises the importance of maximizing a driver's forward vision for night-time driving safety purpose. One way to achieve this is to improving the utilization of the vehicle's high-beam headlight so that the drivers can look far ahead for traffic signs, road geometry, other vehicles, pedestrians and potential hazards. Nevertheless, a recent study by U.S. Department of Transportation shows that, on average, drivers use their high beams less than 25% of the time during which conditions justified their use [2].

This motivates the automobile industry to look into various intelligent headlight control (IHC) systems, to aim for automatically and optimally controlling the headlight of an automobile during a night-time drive. Consequently, drivers no longer need manually and repeatedly switch the beams and thus concentrate more on actual "driving".

### 1.1 Related Work

To our best knowledge, only a few research efforts have been published in this area, yet there are several such systems being prototyped or deployed in the current market. For instance, the $SmartBeam^{TM}$, developed by Gentex [3], uses a customized and forward-facing CMOS image sensor to acquire images in front of the vehicle, which are then processed to detect the existence of headlamps of oncoming traffic or taillamps of preceding vehicles. Appropriate headlamp switching is then performed based on such detection. With similar ideas, Mobileye [4] developed an adaptive headlight control system which also considers the scenario of lit/urban areas, while $SmartBeam^{TM}$ does not.

Recently, Mercedes-Benz announced that it is going to integrate an Intelligent Light System and Adaptive Highbeam Assist system into its future models [5]. Specifically, such system can adjust the range of the headlamps automatically based on the distance to oncoming vehicles or moving vehicles in front with their lights on. Similar research efforts have been proposed before. For instance, [6] employs a fully variable light distribution to realize a dazzle-free high beam by using arrays of LEDs. Specifically, once another vehicle is detected, only the LED that generates the spot covering the detected vehicle is turned off, while all other regions are still being illuminated by other LEDs. Likewise, Konning *et al.* argue that a more advanced IHC system should be able to dynamically adapt the position of the illumination cut-off just below the next vehicle in front of the ego-vehicle [7].

Generally speaking, a common approach to intelligent headlight control is to detect potential light objects using some image processing algorithms, then apply certain heuristic rules to decide if high beam should be used or not. While such solution is relatively easier and quicker to develop, it usually suffers from the drawbacks such as difficult deployment to different geographical regions, lack of robustness to the change of weather and road conditions, as well as expensive system fine-tuning. Consequently, machine-learning based approaches become more preferable. One such effort is reported in [8], where a Real-AdaBoost learning machine is applied to train 4 classifiers for small/non-small headlight, and small/non-small taillight, using various appearance-based features. Nevertheless, other important features such as motion are neglected which could have helped boost the learning performance. Moreover, street lights, as another important light object, are not considered.

### 1.2 Proposed Approach

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
*UCVP'09,* November 5, 2009, Boston, MA.
Copyright 2009 ACM 978-1-60558-692-2-1/09/11 ...$10.00.

In this work, we aim to develop an intelligent headlight control system using a forward-facing camera sensor that works at real time. Our target is to detect oncoming and leading traffic as far as 1000 meters and 400 meters away, respectively, on straight and flat roads under dry weather condition. In addition, when an overtaking vehicle or an urban area is detected, the system should also switch to low beam as soon as possible. Note that here we define an urban area as well-lit streets with lights.

To achieve such challenging target, we propose a three-level decision framework as shown in Figure 1. Specifically, the first level contains a blob detection module which, given an image captured from a night-time drive, finds bright spots that stand out from the dark background. The image shown in the figure contains an oncoming vehicle with two headlights (and their road reflections) on a country road.

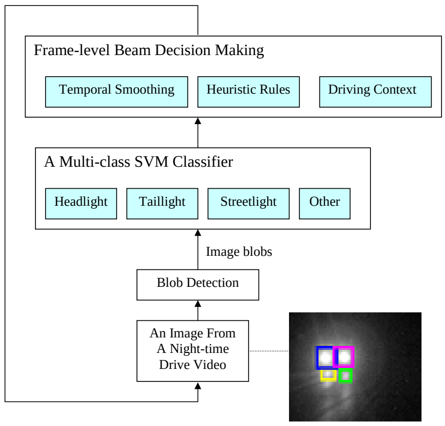



**Figure 1: The proposed three-level decision framework for intelligent headlight control (IHC).**

In the second level, SVM (Support Vector Machine), selected among many existing machine learning approaches due to its good performance in a variety of tasks, is applied to recognize different image blobs which include candidates of headlight, taillight, streetlight, and other object. Here the other objects mainly includes infrastructure elements such as traffic signs and road reflectors that are reflected to bright by our own emitted light, as well as objects that light up by themselves. A multi-class SVM classifier is trained for this purpose.

Finally, in the third level, a frame-level decision making process is carried out to determine the actual beam state (*i.e.*, high beam or low beam) of the vehicle for the current frame based on SVM classification result, a set of heuristic rules and the driving context derived from the vehicle's CAN bus (Controller-area network). A temporal smoothing process is further employed in this case to ensure an un-abrupt beam change.

The rest of this paper is organized as follows. Section 2 introduces the system setup in a real car. Section 3 then briefly describes the blob detection module which forms the building block of the entire IHC system. Section 4 presents the SVM-based learning and classification scheme where blob-level feature extraction is detailed. In Section 5, we illustrate the final frame-level beam decision making process. Extensive experiments and detailed performance evaluation are reported in Section 6. Finally, we conclude the paper in Section 7 with future plans.

## 2. SYSTEM SETUP

We have worked with a major Tier 1 automotive supplier (referred to as Company M) to set up a demo car so as to capture the videos and test out the proposed system at realtime. This company has many years of experience in the area of IHC, and has provided us many valuable suggestions, as well as set requirements on the design, installation and performance evaluation of the system.

The camera that we use for this work has a high dynamic range. It captures colored video at $640 \times 480$ resolution, $42^0 \times 31.5^0$ field of view, with 35 frames per second. It works with 12 bits per pixel using a logarithmic curve, which is essential to avoid fully saturated light spots. The camera is mounted on the windshield right behind the rear mirror and faces forward. Its captured video is processed by a PC installed on the car.

Figure 2 shows the detailed system setup framework. Specifically, PC 1 is solely used to capture and store the video from the camera, while PC 2 is the main workhorse which hosts the entire IHC system. Both computers interact with the CAN bus at realtime. Finally, the IHC decision is passed on to the CAN bus controller which puts it into real action.

This system setup also allows us to use multiple cameras when needed, although some other issues such as camera synchronization and decision fusion need to be considered in this case.

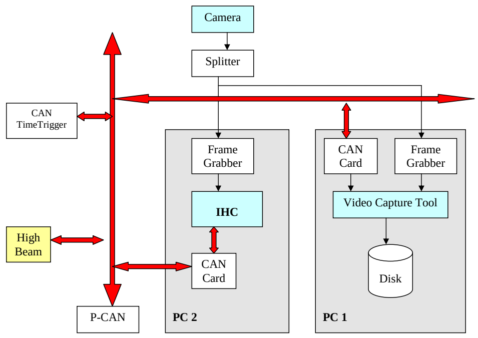



**Figure 2: The system setup framework.**

## 3. BLOB DETECTION

This module applies a standard connected component analysis to identify bright spots in an image which is taken during nighttime. These blobs are then passed onto the SVM classifier for type recognition. Note that by applying such spatial attention function at the first level of the system, a substantial computational effort can be saved from running the classification engine at all possible positions, scales or orientations within the image.

Two thresholds are used in such detection process: $T_0$, which thresholds all pixels that form the blob, and $T_1$, which thresholds the "core" pixels within the blob. As the core part is generally brighter than its surrounding part (which we call *halo*) within a blob, we set $T_1$ to be slightly larger than $T_0$. The image in Figure 1 shows an example of detected blobs (indicated by four bounding boxes) containing the two headlights and their road reflections.

The last step of this module is to filter out blob whose area is either too large or too small to be considered as potential headlight, taillight or streetlight.

## 4. SVM-BASED BLOB LEARNING AND CLASSIFICATION

The two major phases involved in this module are SVM training and SVM classification. Specifically, the training phase produces a multi-class SVM model which learns the following four different types of blobs: *headlight* (HL), *taillight* (TL), *streetlight* (SL), and *other*. The classification phase then uses such model to recognize the type for an unknown blob. Note that we have also performed cross validation to select optimal model parameters and to avoid model over-fitting.

The detailed flowchart for such two processes is shown in Figure 3. Specifically, during the training phase, we first identify a set of videos that contain representative data in terms of objects of interest. Qualified blobs are then detected within each frame, and their types are manually annotated. Next, we extract a list of features from each blob, and such feature vectors (plus the proper class labels) are finally used as training samples for SVM learning. The LibSVM tool [9], which has been reported with ease of use and good performance, is chosen for this work.

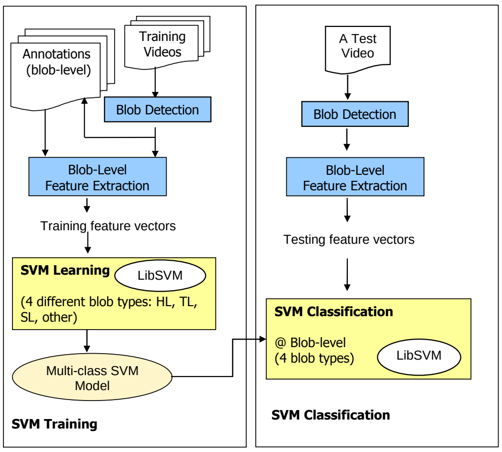



**Figure 3: The flowchart of SVM training and classification.**

Now, during the classification phase, given a test video, we first repeat the same process of blob detection and feature extraction to form testing feature vectors. Then the LibSVM is used to recognize each blob type using the pre-trained SVM model.

Below, we detail on the list of features extracted from each blob.

## 4.1 SVM Feature Extraction

In total, 26 features are extracted from each image blob, which can be grouped into 6 categories as follows.

**1. Position features**, which include the x and y coordinates of the gravitational centroid of the blob. This is motivated by the observation that lights from different traffic (oncoming, leading, crossing) and from street lamps are generally located at different spatial locations within the image. The challenge is, when headlights, taillights and street lights are far away, they all appear to be close to the vanishing point.

**2. Brightness features**, which include the average and variance of intensity of all pixels within the blob. The intuition is that, close-by light objects are generally brighter than far away objects or reflected objects. Also, headlights are generally brighter than taillights. The challenge is, sometimes, reflected objects could be very bright which causes potential false positives.

**3. Shape features**, which describe a blob's shape in eight different aspects. Specifically, they include: the area of the blob (in unit of pixels); the ratio of the blob area over the size of its bounding box; the aspect ratio and angle of the blob, which are calculated from the moments; the average and variance of the blob's radius; size of the halo, and ratio of the halo over the blob's area.

The intuition for selecting these shape features is that, blobs corresponding to the headlight, taillight and streetlight usually appear to be round in the image, although their actual shapes could be slightly different. Nevertheless, blobs for other objects are likely to be of irregular shapes. The challenge here is that, the reflected spots on the other objects tend to be of round shape as well.

**4. Spatial relation features**, which describes the spatial relationship among image blobs. This is motivated by the observation that in most cases, headlights or taillights are in pairs, while street lights are generally lined up along roadside in multitude. Consequently, we propose to extract some features to capture such spatial relationship among blobs based on both pair analysis and line (or slightly curved line) analysis. For instance, the number of buddies for a blob within detected group; the mathematical representation of detected line, and the horizontal/vertical distances between the two ends of the line or pair.

Some of these spatial relation features have turned out to be very effective, as verified by the feature saliency analysis; nevertheless, challenges still exist. For instance, some urban areas have very sparsely distributed street lights, which ends up with only one or two street lights in each frame.

**5. Color features**, which describe the dominant hue (H), average saturation (S) and average value (V) of each blob. Since different types of lights generally present different colors, for instance, taillight is reddish, headlight is usually bright white, while street lights could be yellow, color features can be a big help for SVM learning.

**6. Motion features**, which describe the trajectory of each blob from frame to frame. This is motivated by the observation that headlight, taillight and streetlight generally have different moving directions. For instance, headlights are moving towards us, taillights are moving away, while street lights are normally moving outwards. In total, five motion features are extracted which include the motion displacement and moving speed in both x and y directions, plus the moving direction of the blob. Theoretically, such motion features are expected to be very useful, nevertheless, a big challenge here is that when objects are far away, the extracted motion information tends to be less accurate.

# 5. FRAME-LEVEL BEAM DECISION MAKING

After recognizing all blobs within each frame using the pre-trained SVM model, our last step is to make the beam decision. Below, we elaborate on the proposed decision making mechanism, which is built upon a temporal smoothing process, some heuristic rules and the driving context.

For the ease of illustration, we assume that each frame $f$ has two beam states, the *hidden state* (HS) and the actual *display state* (DS). The difference between these two states is that, HS is determined based on what is happening in the current frame, while DS is determined based on not only the current moment, but also the past. In another word, it takes history into account.

We first describe how the HS of frame $f$ is determined. Assuming that a high beam is the beam state by default, we apply the following three rules to set $HS_f$ to low beam.

**Rule 1:** If there is at least 1 headlight or 1 taillight, or 3 streetlight blobs detected in frame $f$, then $HS_f$ is low beam. Note that the 3-streetlight heuristic comes from some informal definition of an urban area. In reality, strictly following such urban rule results in decision errors, which will be detailed later.

**Rule 2:** If frame $f$ contains more blobs than a pre-defined threshold (which is set to 15 based on experiments), we set $HS_f$ to low beam. This comes from the observation that in urban areas, there are usually many city lights which results in a lot more blobs than in videos containing only dark country roads.

**Rule 3:** If the weighted product of the ambient brightness and the average edge energy in frame $f$ is greater than a pre-defined threshold, we set $HS_f$ to low beam. This is based on the observation that a well-illuminated urban area usually demonstrates a higher amount of ambient light than other situations. Moreover, due to the existence of buildings in urban areas, we normally observe more edges in such videos.

Now, to determine frame $f$'s display state (DS), we take the following reality factors as well as heuristics into consideration.

**1.** The beam change should not be too frequent, *i.e.,* there should be an acceptable temporal period between two consecutive beam changes, otherwise, it becomes annoying. Currently, we set such temporal period to be equal with the video's frame rate ($fps$). During implementation, we use variable $gapCount$ to track the number of frames between two beam switches.

**2.** As blob recognition errors are unavoidable, we have proposed the following two ways to address this issue: 1) we shall not make the final beam decision solely based on the current frame, instead, we apply a temporal smoothing process to take the *beam history* into account. A sliding window of size $fps$ is used for this purpose; and 2) once we enter the low-beam state due to the detection of headlight, taillight or streetlight, we start tracking such light objects from frame to frame, so as to correct occasional missed detections. Consequently, as long as the objects are consistently tracked, a consistent beam decision will be maintained.

Keeping the above considerations in mind, we now detail the four steps that determine frame $f$'s DS state.

**Step 1.** If frame $f$'s HS is the same as its preceding frame's DS, then we directly display its HS without any change. This is the perfect scenario.

**Step 2.** If we just have a beam change within a second, *i.e.* $gapCount < fps$, then no matter what is $f$'s HS, we shall maintain the existing beam state. No change is allowed in this case.

**Step 3.** If the existing beam state is low beam, which has been on for a sufficiently long period, and if the current frame's HS votes for high beam, then we shall track the detected light objects from the previous frame. If the tracking is successful, we stay with the low beam.

**Step 4.** If a beam change is still likely needed, we shall confirm such decision by polling the history. Specifically, we check the HS state of all frames in the sliding window that covers the last one second, and see if a certain percentage of them have the same HS as such $HS_f$. If yes, a change decision will be confirmed, and we set $DS_f$ to $HS_f$; Otherwise, we maintain the existing beam state that is on display.

A threshold $T_c$ is applied in this process, which is assigned to different values depending on different situations. Specifically, if we tend to change the beam from low to high, then $T_c$ is set to 0.8, meaning that 80% of preceding frames in the window have to support such $HS_f$ in order to make the change happen. On the contrary, if we will potentially change the beam from high to low, then $T_c$ is set to 0.3, meaning that we only need fewer preceding frames to support such change. The theory behind such varying preference is that, we want a quicker response to switch from high beam to low beam, so as to avoid dazzling other drivers. On the hand other, when it comes to switch from low beam to high beam, we prefer a more assured decision by examining a few more frames.

## 5.1 Exploiting Driving Context

Driving context, which could be obtained from the vehicle's CAN bus, provides another type of important information that should be considered to further improve the system performance. Consequently, two additional rules, which exploit two types of such context, namely, the *steering angle* and *speed*, have been applied.

Specifically, the first rule is applied to the scenario where the road becomes curved or when there is a roundabout. In such situation, we tend to lose track of the streetlight or taillight, which have been visible when the road was straight, due to the "forward-facingness" of the camera. To address this issue, we constantly monitor the road condition based on the steering angle value. Once such value is above a certain threshold, we declare that a curved road is identified. Consequently, we maintain the detection of street lights and taillights if they are previously recognized. We continue with such "assumed" recognition until the road goes back to straight. After that, we resume the normal beam decision process as described earlier.

In contrast, the second rule is applied to improve the robustness of urban area detection. As pointed out earlier, due to un-robust streetlight detection and poorly illuminated urban areas with very few street lights, strictly following the "3-streetlight" urban rule could result in missed detection. On the other hand, urban areas all have strict speed limits to be observed by drivers. Based on this, we propose that once we enter a confirmed urban area, a low beam should be continuously maintained as long as there is at least one streetlight detected, and the driving speed is below a certain threshold.

# 6. PERFORMANCE EVALUATION

The proposed system has been evaluated in both online and offline modes[1]. Specifically, the online mode refers to observing the system performance at real time when taking the host car for a drive. In this case, the performance is evaluated more qualitatively and subjectively. For the offline mode, we test the system using tens of night-drive videos recorded under dry weather, various road conditions and in changing urban and country-side scenarios. Three different metrics are then defined to measure the performance.

Due to the limit of space, we will not report the online test result here, which is mainly conducted by our partnering company M.

## 6.1 Performance Metrics

The offline system performance is measured in terms of false negative value (FNV), false positive value (FPV) and overall errors. Specifically, the false negative value (FNV) is calculated as:

$$FNV = \sum_{k=1}^{MD} \eta |t_k^{off} - t_k^{on}| + \sum_{i=1}^{LD} \delta_i \eta |t_i - t_0| + \sum_{j=1}^{EA} \delta_j \eta |t_j - t_0| + \sum_{k=1}^{FA} \eta |t_k^{on} - t_k^{off}|, \quad (1)$$

where $MD$, $LD$, $EA$ and $FA$ stand for *missing deactivation*, *late deactivation*, *early activation* and *false activation*, respectively. These are four typical types of false negatives that leaves high beam on by mistake. Moreover, $t_0$ indicates the ideal switching moment given by the ground truth, $t_i$ and $t_j$ are the switching moments indicated by the system. For a missing deactivation, $t_k^{off}$ is the time when the high beam should have been deactivated, and $t_k^{on}$ is the time when the high beam should have been re-activated.

[1]Note: The performance of the system presented here does not reflect the performance of existing commercial offerings of Company M

They are similarly defined for the case of false activation as well. The two $\delta_i$ and $\delta_j$ are used to accommodate for tolerable variance ($\Delta$) between a system decision and a human decision. Specifically, if $\Delta$ is greater than 0.25 second, they are set to 1, otherwise, 0. Finally, $\eta$ is a weighting factor, which specifies how strongly each error case should be punished. For now, it is equally set to be $1/\Delta$.

Similarly, the false positive value (FPV) is defined as follows.

$$FPV = \sum_{k=1}^{FD} \eta |t_k^{off} - t_k^{on}| + \sum_{i=1}^{LA} \delta_i \eta |t_i - t_0| + \sum_{j=1}^{ED} \delta_j \eta |t_j - t_0| + \sum_{k=1}^{MA} \eta |t_k^{on} - t_k^{off}|, \tag{2}$$

where $FD$, $LA$, $ED$ and $MA$ stand for *false deactivation*, *late activation*, *early deactivation* and *missing activation*, respectively. These are four typical types of false positives that leaves low beam on by mistake.

Finally, the overall error function is defined as

$$Error = 10 \times FNV + FPV, \tag{3}$$

which clearly states that false negatives are more un-tolerable than false positives. This also explains why we assign different values to the threshold $T_c$ during the decision making process. Note that the factor of "10" in this equation is not randomly decided by ourselves, but rather, it is derived based upon some automotive manufacturer's requirement.

## 6.2 Performance Report

The SVM model used for the performance evaluation is trained with 29 videos, which amounts to 30 minutes in total. The number of headlight (HL), taillight (TL), and streetlight (SL) objects annotated for this training set are tabulated in Table 1. Moreover, we have also reported the distinct or unique number of HL, TL and SL, which gives readers a rough idea of how many oncoming/leading vehicles as well as unique street lights are contained in this set.

**Table 1: Statistics of the training set.**

| | |
|---|---|
| Number of HL objects | 9773 |
| Number of TL objects | 12192 |
| Number of SL objects | 8462 |
| Number of unique HL objects | 93 |
| Number of unique TL objects | 15 |
| Number of unique SL objects | 46 |

One interesting observation from table 1 is that, while we have a fairly large number of taillight objects, the unique number of taillights is much smaller than others. This is due to the fact that some videos contain the same leading vehicle throughout the entire trip, thus producing as few as 2 distinct taillights. Consequently, when we choose the training videos, we have strived to obtain as many unique light objects as possible, so that the SVM model can learn diverse types of headlight, taillight and streetlight.

Below, we report the system performance in the following two dimensions: 1) the blob classification performance; and 2) the IHC system performance, as we consider them both important measurements. Moreover, all test videos are from a different set than the training videos.

For the former evaluation, we randomly selected 12 pre-annotated videos, and measured their blob-level SVM classification results against the ground truth in terms of *false rejection* (FR), *false acceptance* (FA), and classification accuracy. Specifically, we calculated the FR and FA for each object class from the confusion matrix constructed for each test video. As a result, we achieved around 90% average blob classification accuracy, and the average FR and FA are both 0.1, which is quite acceptable. Moreover, please note that the mis-classification between HL and TL does not really affect the IHC system performance, since the same beam decision will be made as long as either of them is detected.

For the latter evaluation, the best way is to compare our proposed system with the state-of-art IHC systems on the market. Nevertheless, as it is practically impossible to obtain the software of these proprietary systems, we are not able to conduct such fair comparison during offline test. However, a general evaluation of our system against the Gentex [3] was indeed conducted by our partnering company M during the online test, and the overall observation is that our system generally performs worse than Gentex.

The overall impression of our proposed IHC system is: 1) for most of time, it quickly switches to low beam for oncoming traffic; 2) it performs generally well in well-illuminated urban areas; 3) it fails to recognize leading traffic in a distance (yet still within the targeted 400 meter range); and 4) on curved road, it tends to generate false positives due to road reflectors and traffic signs.

Table 2 presents the offline performance measurement for 13 test videos, which as will be discussed later, shows us how the system performance varies with different driving situations. Note that all performance data has been normalized by the video duration. For these 13 videos, the first six are randomly selected from our test set, while the rest are carefully chosen due to the diverse and challenging driving situations contained within them. All used ground truth in this case, were collected during the video capturing process from both the driver and the passenger in the front seat through some special input device. While both of them have attempted to be "well-behaved" drivers who diligently use high-beams in justified conditions, we have observed quite some variations between their behaviors.

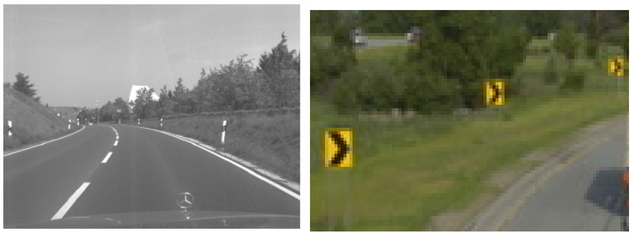
**Table 2: IHC performance evaluation on 13 videos.**

| Video | Length (min) | FNV | FPV | Error |
|---|---|---|---|---|
| Video 1 | 4.2 | 14.5 | 8.1 | 153.3 |
| Video 2 | 3.3 | 2.7 | 24 | 50.5 |
| Video 3 | 3.27 | 13.1 | 1.0 | 132.1 |
| Video 4 | 1.5 | 0 | 0 | 0 |
| Video 5 | 1.2 | 0 | 0 | 0 |
| Video 6 | 1.9 | 0 | 0 | 0 |
| Video 7 | 4.6 | 40.3 | 5.2 | 406.2 |
| Video 8 | 5.3 | 66.5 | 24.6 | 686.53 |
| Video 9 | 4.6 | 57.6 | 7.7 | 583.6 |
| Video 10 | 8.4 | 31.4 | 10.0 | 326.5 |
| Video 11 | 3.8 | 31.1 | 35.2 | 346.06 |
| Video 12 | 10.6 | 11.8 | 10.8 | 129.1 |
| Video 13 | 8.3 | 12.5 | 16.2 | 140.58 |

Below, we give a detailed analysis of the scenarios that have caused false positives and false negatives as observed from the Table 2. Specifically, the two major situations that lead to false positives are:

**1. Road reflectors on curved road.** In our test videos, there are reflective markers installed along the roadside for the safety and information purpose. Specifically, every 50 meters on the road, there would be two markers symmetrically lined up on each side. One such example is shown in Figure 4 (a). In this case, when the road becomes curvy right in front of the controlled vehicle, those reflective markers would appear like headlights of oncoming cars, thus triggering a low beam decision.

**2. Traffic signs on curved road.** One typical scenario that almost always causes false positive is when the controlled car directly faces a curved road that has a set of curve signs on the shoulder. One example of such sign is shown in Figure 4 (b). These reflective signs contain directional arrows, and once they are directly shined upon, they tend to be recognized as headlights.



(a) (b)

**Figure 4: (a) Reflective markers at every 50 meters along the road, and (b) curve traffic sign on curved road.**

In contrast, false negatives tend to occur in the following scenarios.

**1. Leading traffic with distant taillights.** This is the number one reason that results in undesired high beams. Specifically, while the leading traffic is far away, yet still within reasonable "viewable" distance, our system fails to detect its taillights, and switches on high beam. Referring back to Table 2, videos 7-11 all contain heavy driving on a country road following some leading traffic, and due to the failure of recognizing distant taillights, we have thus observed fairly large number of false negatives. Examining further deeper into the system behavior, we discovered that most of these distant taillights were missed by the blob detector in the first place, due to their low intensities. This will be improved in our next version of blob detector.

**2. Dim urban areas.** Sometimes, an urban area could be poorly illuminated. For instance, when the vehicle drives at the edge of a village where there are buildings on one side, yet fields on the other side, or when it just enters a village. In such cases, not only will no or fewer street lights be detected, but also not much edge or ambient light are present. Consequently, the system tends to turn on high beam. While it looks natural for the system to perform this way, it is however, required to use low beam as the vehicle is within an urban area.

Referring back to videos 12 and 13 in Table 2, both of them contain heavy driving in urban areas, and most of their false negatives occur during such poorly illuminated situations.

Finally, there are three other issues that potentially affect the system performance as well as its evaluation.

**1. Defects in the object annotation.** Various types of defects in the annotation data such as duplicate object ID for different objects and un-annotated blobs that correspond to valid objects, would unavoidably affect the trained model and consequently the classification result.

**2. The unbalanced nature of the training data.** The current SVM model is trained with very unbalanced data among the four object classes. Specifically, the training data of taillight is much less than that of headlight and streetlight, due to the difficulty of collecting it. On the other hand, the "other" object class has significantly too much data. Such unbalanced nature of the training data has unavoidably affected the performance of the trained model.

**3. The un-justified parameters ($\Delta$ and $\eta$) in the performance metrics.** Currently, the $\Delta$, which captures the variance between different human drivers, is set to 0.25 second. However, our study on driver variance shows that it could go up to 2 seconds. Moreover, an even bigger variance between drivers from different countries has been observed. On the other hand, since some scenario like missing deactivation is more severe than others like late deactivation, we may consider using different weighting factors ($\eta$) accordingly.

# 7. CONCLUSION AND FUTURE WORK

A three-level decision framework is proposed in this paper which aims to automatically control the switch between high beam and low beam of an automobile during a night drive using a forward-facing camera sensor. Specifically, it consists of an image blob detector at the lowest level, an SVM-based blob learning and recognition module in the middle, and a frame-level decision making at the highest level. Various types of information including image, video and driving context, has been exploited to achieve the task.

Extensive offline evaluation as well as some online evaluation have proved the effectiveness of the proposed framework. On the other hand, the current performance is still lagging behind the leading IHC product on the market, and we are planning to improve the system performance in the following ways: 1) enhance the current blob detector so as to detect both bright (HL- and SL-like) and dim (TL-like) blobs; and 2) train the SVM model with weighting factors so as to accommodate the un-balanced training data, or to try multiple binary classifiers instead of a single multi-class classifier.

# 8. REFERENCES


[1] J. Sullivan and M. Flannagan, "Assessing the potential benefit of adaptive headlighting using crash databases," *Report No. UMTRI-99-21, The University of Michigan Transportation Research Institute*, 1999.

[2] U.S. National Highway Traffic Safety Administration, "Traffic safety facts," 2000.

[3] C. Schadel and D. Falb, "SmartBeam–A high-beam assist," *Proc. of the 7th International Symposium on Automotive Lighting*, Darmstadt, Germany, 2007.

[4] Mobileye, "Adaptive headlight control," *http://www.mobileye.com/default.asp?PageID=222*.

[5] Mercedes-Benz, "Mercedes-Benz TecDay special feature: New headlamp and night-view systems," *http://www.emercedesbenz.com/Nov08/12_001504_Mercedes_Benz_TecDay_Special_Feature_New_Headlamp_And_Night_View_Systems.html*, November 2008.

[6] B. Hummel, "The future of the LED headlamp (in German)," *Haus der Technik - LED in der Lichttechnik*, Essen, Germany, 2008.

[7] T. Konning, C. Amsel, and I. Hoffmann, "Light has to go where it is needed: Future light based driver assistance systems," *Proc. of the 7th International Symposium on Automotive Lighting*, Darmstadt, Germany, 2007.

[8] A. Lopez and et al., "Nighttime vehicle detection for intelligent headlight control," *The 10th International Conference on Advanced Concepts for Intelligent Vision Systems*, 2008.

[9] C. C. Chang and C. J. Lin, "LIBSVM–A library for support vector machines," *http://www.csie.ntu.edu.tw/ cjlin/libsvm*.