Research Article

Evaluation of Image Enhancement Techniques for Vision-Based Navigation under Low Illumination

Mohamed Aladem (✉), Stanley Baek (✉), and Samir A. Rawashdeh (✉)

College of Engineering and Computer Science, University of Michigan-Dearborn, Michigan 48128, USA

Correspondence should be addressed to Samir A. Rawashdeh; srawa@umich.edu

Received 5 September 2018; Revised 29 December 2018; Accepted 29 January 2019; Published 20 March 2019

Academic Editor: Tarek M. Sobh

Copyright © 2019 Mohamed Aladem et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Cameras are valuable sensors for robotics perception tasks. Among these perception tasks are motion estimation, localization, and object detection. Cameras are attractive sensors because they are passive and relatively cheap and can provide rich information. However, being passive sensors, they rely on external illumination from the environment which means that their performance degrades in low-light conditions. In this paper, we present and investigate four methods to enhance images under challenging night conditions. The findings are relevant to a wide range of feature-based vision systems, such as tracking for augmented reality, image registration, localization, and mapping, as well as deep learning-based object detectors. As autonomous mobile robots are expected to operate under low-illumination conditions at night, evaluation is based on state-of-the-art systems for motion estimation, localization, and object detection.

1. Introduction

Many computer vision algorithms, in particular structure-from-motion problems, rely on detection and tracking of feature points. For example, image registration problems where aerial images are automatically aligned and stitched together require anchor points to correctly align images [1, 2]. In the domain of augmented reality, feature points in the environment are used to track the movement of a target visual marker and correctly render virtual object over reality [3]. In robotics, cameras and the tracking of key points in view are commonly used for localization and mapping [4, 5]. Performing these tasks in low-light conditions (e.g., at night) is problematic because distinct objects in the scene are fewer, and the ability to correctly associate them over multiple images becomes unreliable. Moreover, object detection, recognition, and image classification tasks have been recently tackled with great success using deep learning approaches [6, 7]. In this paper, we focus on preprocessing approaches that can be used to enhance image quality, with emphasis on mobile robot navigation.

Robot localization and motion estimation are fundamental tasks for modern autonomous robotics applications. Performing said tasks accurately and in real-time is essential for robotic perception and control. Simultaneous localization and mapping (SLAM) and visual odometry (VO) are the powerful tools that provide such robot localization and motion estimation. SLAM is where the agent (e.g., a mobile robot or an autonomous vehicle) is building a map of the environment while simultaneously localizing itself within this map. VO is where the agent is estimating its position and orientation in previously unknown environment, online and using the input of a camera attached to it. Excellent introduction and survey of visual SLAM are by Younes et al. [8], and those for visual odometry are by Scaramuzza and Fraundorfer [9].

From these surveys, we have observed that visual SLAM and visual odometry systems are evaluated at daytime under typical lighting conditions. However, autonomous mobile systems such as self-driving cars should be expected to work at night and under low-light conditions as well. Utilizing active sensors and other methods (such as LIDARs) is necessary to provide functionality under these conditions. However, expanding the operation range of passive cameras through image enhancement approaches is compelling to maintain sensor redundancy under these conditions, which
is a factor to functional safety. Prior work focusing on achieving operation of feature-based vision algorithms at night is limited, which motivated this work.

LIDARs are active sensors and can be used to perform the localization tasks at night, as in the work by Dong and Barfoot [10]. Maintaining a suite of sensors consisting of heterogeneous sensing modes is advantageous, as it protects against failure modes of a particular sensor. For example, being active sensors, LIDARs are expected to interfere with each other when they are in proximity, while passive vision systems do not have this problem. Moreover, LIDARs are more expensive than cameras. Brunner et al. [11] combined visual and infrared imaging in a visual SLAM system to solve the localization problem under day and night conditions. Nelson et al. [12] have presented a solution for localization at night by localizing with respect to artificial light sources that are expected to be present at night in urban environments. MacTavish et al. [13] have developed a visual odometry system and relied on mounted headlights on the mobile robot platform as the main illumination source.

Cameras are attractive sensors as they are relatively cheap, passive, and are capable of providing rich information about the environment. Moreover, this information can be used for tasks other than localization, such as scene segmentation or object recognition. In this paper, we investigate potential image preprocessing techniques to enhance night images for vision-based robotic perception. We introduce four different image preprocessing techniques and study their performance on three systems:

(i) ORB-SLAM2 [14], which is a state-of-the-art visual SLAM system.

(ii) LIBVISO2 [15], which is a well-known visual odometry system.

(iii) LVT [4], our previously developed visual odometry system, which we briefly describe in Section 2.3.

Although our primary focus is on feature-based visual odometry and visual SLAM systems, we additionally study the performance of presented preprocessing techniques on an object detector based on a deep convolutional neural network. Luckily, our findings are applicable to a wide range of computer vision tasks, such as image stitching, registration and recognition. Low-light image enhancement affects feature-based algorithms at two main steps. First is feature detection. Feature or interest point detection depends on contrast/gradient in the image, and a better lit image provides more distinctive features. Second is matching detected interest points between images. Matching interest points can be performed by comparing the pixel intensities of the image patch around each interest point directly using some similarity measure. A descriptor which acts as a signature for that image patch can also be computed and different descriptors can be compared. In either case, image enhancement in this case can provide more unique descriptors and result in more reliable matching. Such algorithms can be operating normally with no preprocessing under typical lighting conditions. Once the mean scene luminance drops below a certain threshold, our preprocessing is activated to enhance the incoming low-light images.

2. Visual Odometry and Visual SLAM

In the following subsections, we will briefly introduce the three visual odometry or visual SLAM systems used in our experiments.

2.1. ORB-SLAM2. ORB-SLAM2 is a state-of-the-art visual SLAM system. It is a full visual SLAM system meaning it is continuously building a map of the environment while simultaneously localizing itself in that map. ORB-SLAM2 supports map reuse, loop closing, and relocalization capabilities. Their system uses ORB features [17] for the three major tasks of tracking, mapping, and place recognition which is used for loop closing and relocalization [14].

2.2. LIBVISO2. LIBVISO2 is a well-known visual odometry system. It computes motion estimates between consecutive frames. The system uses custom filters for detecting sparse features in images. It employs what they call circular matching when finding matches between consecutive frames. That is, if a feature in the current left image is matched with one in the previous left image, which in turn is matched to a feature in the previous right image, then to the current right image, and finally if that matches the starting feature in the current left image then it is declared a circular match and passes for egomotion estimation. Camera motion is then computed by minimizing the reprojection errors, then a Kalman filter is used to refine the obtained velocity estimates [15].

2.3. LVT. LVT, or lightweight visual tracking, is our previously developed visual odometry system [4]. The system requires a stereo camera; thus the main problem of scale ambiguity found in monocular vision is avoided. The visual odometry algorithm’s main steps are summarized in Figure 1. Camera’s pose is the complete six degrees of freedom transformation (translation + rotation). The world reference coordinate frame is set at the pose of the first frame of the sequence.

The algorithm is feature-based, relying on extracting point or corner-like features from images. We have adopted the adaptive and generic accelerated segment test (AGAST) [18] corner detector. No scale pyramid of any image is built, and the corners are extracted from the full-size images. For each detected feature, a feature descriptor is computed. A feature descriptor acts like a signature for that feature. Matching between features can then be performed by comparing their descriptors. In our algorithm, binary robust independent elementary features (BRIEF) [19] descriptors are used. BRIEF is a binary descriptor where the descriptor vector is in the form of a binary string. This binary string is computed by selecting a set of location pairs in the image patch in a unique way. The paper describes five different sampling strategies. Then for each selected location pair, compare the pixel intensity at the first point with that of the second one. If the intensity is larger, then append 1 to the descriptor string; otherwise append 0. To ensure good
feature distribution across the image, we perform adaptive nonmaximal suppression as described by Brown et al. [20].

Our system starts by matching features between the left and right images of the first stereo frame. Those matches are used to triangulate 3D points which are added to the local map. This local map is transient and consists of a sparse set of 3D points. It is internal to the system and used solely for the purpose of pose estimation; it is not an attempt to build a global consistent map of the environment.

For each subsequent frame, the frame's pose is first predicted using a simple motion model. Then the 3D points in the local map are projected onto the left image. Now, for each projected 3D point, we search its neighborhood for the best matching image feature that was detected in the feature detection step. The 3D-2D matches found from this local map tracking step are then used in the subsequent pose estimation operation. Finding the camera pose is formalized as an optimization problem to find the optimal orientation $R^*$ and position $t^*$ that minimizes the reprojection error between the matched 3D points and the image 2D features:

$$\{ R^*, t^* \} = \arg\min_{R \in SO(3), t \in \mathbb{R}^3} \sum_{i \in S} \rho \left( \left\| x^i - \pi (RX^i + t) \right\|^2 \right)$$

where $x^i \in \mathbb{R}^2$ are image features, $X^i \in \mathbb{R}^3$ are world 3D points, $S$ is the set of all matches, $\rho$ is the Cauchy cost function, $\pi$ is the projection function, $R \in SO(3)$ is the orientation, and $t \in \mathbb{R}^3$ is the position. This minimization problem is solved iteratively using the Levenberg–Marquardt algorithm. Furthermore, outliers are detected and excluded, and the optimization is run for a second time with the inlier set. After that, local map maintenance is performed which encompasses triangulating new 3D points from untracked image features if the number of tracked points drops and removing map points that were not tracked for a preset number of frames.

3. Object Detection

Detection of dynamic or potentially dynamic objects, for example, other traffic participants, is another important perception task. Recently, methods using deep learning [21] approaches have prevailed. Attention to deep learning methods sparked when AlexNet [7] won the ImageNet classification challenge [22]. In this paper, we will evaluate a detector based on convolutional neural network for detecting cars at night and with our image preprocessing techniques. This study is useful as such preprocessing is not learned and can potentially enhance the generalization of a pretrained object detector.

The detector that we will use is the Single Shot MultiBox Detector (SSD) [6]. VGG16 [23] is used as a base network for SSD. We have obtained a pretrained network provided by the authors of SSD which is available online [24]. This network was pretrained on Microsoft COCO [25] then fine-tuned on Pascal Visual Object Classes [26]. Input images to the network have the size of 300×300 pixels.

4. Image Preprocessing Techniques

In this section, we describe the four image preprocessing techniques, namely, Gamma Correction, L*a*b* + CLAHE, RG + CLAHE, and Bioinspired Retina Model. CLAHE stands for contrast limited adaptive histogram equalization and will be explained in Section 4.2.

4.1. Gamma Correction. Gamma correction refers to the process of adjusting the luminance of an input image using a nonlinear mapping of the following form:

$$image_{output} = image_{input}^{\gamma}$$

where $\gamma$ is the gamma value, and the input image pixel intensities must be in the range $[0,1]$. In our evaluation, a gamma value of 1.5 has been used.

Here we apply the same mapping to each pixel. Research has been done to develop more advanced adaptive enhancement techniques. For example, Huang et al. [27] have demonstrated an automatic transformation technique through gamma correction and probability distribution of luminance values. Rahman et al. [28] demonstrate a technique where they classify the image content first, then apply adaptive gamma correction based on this image classification information. As we can see, there are many proposed methods, each with its own characteristics. All those methods will add to the complexity of the image enhancement step. Hence, we...
decided to use the base gamma correction which is fast and easy to implement.

4.2. $L^*a^*b^*$ + CLAHE. In this technique, we will convert the image from the traditional RGB color space into the $L^*a^*b^*$ one. This color space consists of the following three components: $L^*$ represents lightness or intensity; $a^*$ and $b^*$ represent the color dimensions. We will extract the lightness channel and then distribute its intensity values as uniformly as possible. This distribution is performed using contrast limited adaptive histogram equalization (CLAHE) technique. In adaptive histogram equalization, multiple histograms, each corresponding to a different section of the image, are computed and then used to redistribute the intensity values across the image. However, this overamplifies the noise in relatively homogeneous regions of the image. CLAHE circumvents this problem by limiting the amplifications. After the CLAHE is applied on the lightness channel, the $L^*a^*b^*$ image is converted back into the RGB color space then grayscale.

4.3. RG + CLAHE. In this technique, we will remove the blue channel from the RGB image and convert it to grayscale by averaging the remaining red and green channels. Blue light is not strong in artificial lights used in vehicle headlamps and streetlights, unlike during daytime under sunlight. We observed the blue channel to be very noisy, as have others in prior work [29, 30]. Excluding the blue channel results in a less noisy image. After that, the same limited adaptive histogram equalization (CLAHE) technique described in the $L^*a^*b^*$ color space technique section is applied to the resulting grayscale image.

4.4. Bioinspired Retina Model. Building models to simulate parts of the human visual system offers a valuable and attractive solution to several computer vision tasks. In this paper, we will use the retina model by Benoit et al. [16] which is available in the open-source OpenCV library. The high-level retina model is shown in Figure 2. This retina model separates spatial and temporal information into two output channels: the parvocellular pathway (parvo), which is related to details extraction; and the magnocellular pathway (magnno), which is related to detecting motion and events.

The interactions between cells in the Outer Plexiform Layer (OPL) are modeled with a nonseparable spatio-temporal filter. The transfer function for this filter for a 1D signal is defined in Equation (3) where $f_s$ and $f_t$ are the spatial and temporal frequencies, respectively.

$$F_{OPL}(f_s, f_t) = F_{ph}(f_s, f_t) \cdot \left[1 - F_h(f_s, f_t)\right]$$

where $F_{ph}(f_s, f_t) = \frac{1}{1 + \beta_{ph} + 2\alpha_{ph} \cdot \left(1 - \cos(2\pi f_s) + j2\pi \tau_{ph} f_t\right)}$ and

$$F_h(f_s, f_t) = \frac{1}{1 + \beta_h + 2\alpha_h \cdot (1 - \cos(2\pi f_s) + j2\pi \tau_{ph} f_t)}$$

(3)

In Equation (3), $\beta$ is the filter gain, $\tau$ and $\alpha$ are the temporal and spatial filtering constants, respectively. The effect of OPL filter is removing spatio-temporal noise and enhancing contours. The photoreceptors cells found in the Outer Plexiform Layer (OPL) and the ganglion cells of the Parvo channel in the Inner Plexiform Layer (IPL) are modeled by Michaelis–Menten law [31]. The Michaelis–Menten relation which is normalized for luminance range of $[0, V_{\text{max}}]$ is

$$C(p) = \frac{R(p)}{R(p) + R_0(p)} \cdot V_{\text{max}} + R_0(p)$$

(4)

$$R_0(p) = V_0 \cdot L(p) + V_{\text{max}} \cdot (1 - V_0)$$

(5)

where, $C(p), R(p)$ are the adjusted and the current luminance of the photoreceptor $p$, $R_0(p)$ is its compression parameter, and $L(p)$ is the local luminance of its neighborhood. $V_0$ is a static compression parameter, and $V_{\text{max}}$ is the maximum allowed pixel value in the image. It is impossible to cover the full details of the retina model in this paper; more information can be found in [16].

This retina model is capable of luminance and detail enhancement. It performs local logarithmic luminance compression which allows both very bright and very dark areas to be visible. It also performs spectral whitening which attenuates the mean luminance energy and enhances mid-frequencies, which correspond to details. We will be using the output of the parvo channel. The parameter values used as input to the retinal model implemented in OpenCV are listed in Table 1. Those values were set heuristically to achieve a smooth image.

The result of applying the aforementioned preprocessing techniques on a sample test night image is shown in Figure 3.

![Figure 2: High-level retina model (adapted from Benoit et al. [16]).](Image)
5. Evaluation and Discussion

For evaluation, we have collected a dataset by driving around the University of Michigan-Dearborn campus at night. The sequence is sequence 28 of our own previously collected adverse weather dataset (http://sar-lab.net/adverse-weather-dataset/). The traveled loop is shown in Figure 4. This dataset was collected on November 17th, 2017, and has an approximate length of 1.49 miles. The dataset was collected by a stereo camera at a resolution of 1280x720 and at a rate of 20Hz.

Table 1: Parameter values used for the parvo channel in the retina model.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Photoreceptors Local Adaptation Sensitivity</td>
<td>0.29</td>
</tr>
<tr>
<td>Photoreceptors Temporal Constant</td>
<td>0.89</td>
</tr>
<tr>
<td>Photoreceptors Spatial Constant</td>
<td>0.53</td>
</tr>
<tr>
<td>Horizontal Cells Gain</td>
<td>0.3</td>
</tr>
<tr>
<td>Hcells Temporal Constant</td>
<td>0.5</td>
</tr>
<tr>
<td>Hcells Spatial Constant</td>
<td>4.0</td>
</tr>
<tr>
<td>Ganglion Cells Sensitivity</td>
<td>0.29</td>
</tr>
</tbody>
</table>


In this evaluation, we will use sequence 3 of our previously mentioned dataset beside sequence 28. Sequence 3 was collected by traveling the same loop as sequence 28 but in daytime. This daytime dataset should be useful for comparison. The result estimated paths from the three evaluated systems, on the daytime, raw night, and preprocessed night
sequence with the described techniques, are plotted on maps and shown in Table 2.

From Table 2, several observations can be made. First, all three systems succeed in completing the morning dataset. In the raw night dataset, only LIBVISO2 was able to keep track to the end. Gamma correction helped ORB-SLAM2 to complete the dataset and give good estimate of the traveled path. The L*a*b* + CLAHE technique failed to help any of the systems in processing the dataset or improving the results. The RG + CLAHE technique helped ORB-SLAM2 to complete the dataset to the end without losing tracking but did not assist in improving the quality of trajectory estimation. Finally, the bioinspired retina method was the only one to help all the three systems to complete the dataset to the end without losing tracking, and it resulted in the best trajectory estimate for all three systems.

More insights can be obtained by extracting some statistics from our visual odometry system (LVT) running on all the preprocessed night datasets. We will extract three statistics:

1. Map size: This represents the number of 3D points that have been triangulated and currently reside in the local map.
2. 2D-3D Associations: This is the number of 3D map points that were successfully associated with 2D image features in the local map tracking step (see Figure 1) and thus was used for pose estimation.
3. Inliers Count: This is the number of found inliers after outlier rejection which is performed as part of pose estimation.

A plot of these statistics for a segment of 200 frames is shown in Figure 5. A quantitative presentation of these values is reported in Table 3 where the mean and standard deviation values are computed over the number of successfully tracked frames. Table 3 also reports the number of tracked frames which shows how long the visual odometry was able to keep track before failing, mostly likely due to tracking too few points in the scene. From Figure 5 and Table 3, we note that daytime results are the best as expected. All preprocessing methods provide substantial improvement over the raw unprocessed night sequence. Bioinspired technique has the best values generally. It is also interesting to observe that L*a*b* + CLAHE technique gives high values for map size and data associations but low inlier count, which means that most of the associations are bad. Gamma correction method appears to give the second-best results after the bioinspired.

5.2. Object Detection Evaluation. In this section, we evaluate the effect of the presented image preprocessing techniques on the convolutional neural network-based object detector mentioned previously in Section 3. We have identified three subsequences of the same night sequence 28, in which there is a nearby car in the field of view. The three identified subsequences of sequence 28 are the ones that start and end at frames: (#710 - #730), (#2400 - #2415), and (#5050 - #5080). The first frame of each of these three subsequences is shown in Table 4 along with the four presented preprocessing techniques applied.

The goal of the evaluation then becomes to determine how many frames this target car was successfully detected with each preprocessing technique applied as well as in the raw frame without any preprocessing. The confidence of the detector in each detection is also reported for more quantitative evaluation. The results are reported in Table 5, and histograms of detector confidence in car detection in all subsequences frames are shown in Figure 6. We can see that L*a*b* + CLAHE technique helped the detector to succeed for the greatest number of frames and with high detection confidences. RG + CLAHE and Gamma Correction techniques closely trail it in terms of number of successful detections and detection confidences. Surprisingly, the Bioinspired Retina Model resulted in the least improvement. This is potentially related to the details enhancement performed by the retina and that may conflict with the internal representation of the objects in the deep learning model initially trained on nondetail enhanced images.

5.3. Runtime Performance. For runtime performance evaluation, we have measured the time to perform the preprocessing computation on each frame. The evaluation was performed on a laptop computer running the Ubuntu 16.04 operating system with an Intel i7-7700HQ CPU. It should be mentioned that the bioinspired model that is available in the OpenCV library which we used has GPU acceleration support; however, we opted to use the CPU implementation instead so that all methods are evaluated on the CPU. The results are reported in Table 6. Gamma correction technique is the fastest because the mapping performed by Equation (2) described before can be precomputed in a lookup table to be used at runtime.

6. Conclusions

In this paper, we have presented four different techniques with the goal of enhancing night and very low-light images. Such enhancement expands the operation range of passive cameras and is compelling in order to maintain sensor
Table 2: Result of the three systems on sequence 3 (Daytime - unprocessed) and the night sequence 28 raw unprocessed and processed with presented techniques. Estimated paths are in blue while driven paths are in red (Attribution: Google Maps).

<table>
<thead>
<tr>
<th>Technique</th>
<th>ORB-SLAM2</th>
<th>LIBVISO2</th>
<th>LVT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daytime</td>
<td><img src="image1" alt="Daytime Result" /></td>
<td><img src="image2" alt="Daytime Result" /></td>
<td><img src="image3" alt="Daytime Result" /></td>
</tr>
<tr>
<td>Night Raw (unprocessed)</td>
<td><img src="image4" alt="Night Raw Result" /></td>
<td><img src="image5" alt="Night Raw Result" /></td>
<td><img src="image6" alt="Night Raw Result" /></td>
</tr>
<tr>
<td>Gamma Correction</td>
<td><img src="image7" alt="Gamma Correction Result" /></td>
<td><img src="image8" alt="Gamma Correction Result" /></td>
<td><img src="image9" alt="Gamma Correction Result" /></td>
</tr>
<tr>
<td>L<em>a</em>b* + CLAHE</td>
<td><img src="image10" alt="Lab* + CLAHE Result" /></td>
<td><img src="image11" alt="Lab* + CLAHE Result" /></td>
<td><img src="image12" alt="Lab* + CLAHE Result" /></td>
</tr>
<tr>
<td>RG + CLAHE</td>
<td><img src="image13" alt="RG + CLAHE Result" /></td>
<td><img src="image14" alt="RG + CLAHE Result" /></td>
<td><img src="image15" alt="RG + CLAHE Result" /></td>
</tr>
<tr>
<td></td>
<td>ORB-SLAM2</td>
<td>LIBVISO2</td>
<td>LVT</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Bioinspired</td>
<td><img src="image1.png" alt="Map" /></td>
<td><img src="image2.png" alt="Map" /></td>
<td><img src="image3.png" alt="Map" /></td>
</tr>
</tbody>
</table>
Table 3: Mean ± standard deviation values of the extracted metric values from our visual odometry system for the different datasets, along with the number of successfully tracked frames.

<table>
<thead>
<tr>
<th></th>
<th>Map Size</th>
<th>2D-3D Associations</th>
<th>Inliers Count</th>
<th>Tracked Frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daytime</td>
<td>249.3 ± 103.3</td>
<td>649.8 ± 159.8</td>
<td>244.0 ± 100.0</td>
<td>8750/8750</td>
</tr>
<tr>
<td>Night raw</td>
<td>159.8 ± 249.3</td>
<td>83.6 ± 50.5</td>
<td>39.5 ± 33.1</td>
<td>4042/8300</td>
</tr>
<tr>
<td>Gamma correction</td>
<td>100.0 ± 249.3</td>
<td>124.0 ± 52.6</td>
<td>52.5 ± 33.5</td>
<td>4404/8300</td>
</tr>
<tr>
<td>L<em>a</em>b+ + CLAHE</td>
<td>100.0 ± 249.3</td>
<td>123.6 ± 52.6</td>
<td>52.5 ± 33.5</td>
<td>4404/8300</td>
</tr>
<tr>
<td>RG + CLAHE</td>
<td>100.0 ± 249.3</td>
<td>123.6 ± 52.6</td>
<td>52.5 ± 33.5</td>
<td>4404/8300</td>
</tr>
<tr>
<td>Bioinspired</td>
<td>100.0 ± 249.3</td>
<td>123.6 ± 52.6</td>
<td>52.5 ± 33.5</td>
<td>4404/8300</td>
</tr>
</tbody>
</table>
Table 4: The first frame of each of the subsequence used in object detection evaluation. Red box is drawn around the target car in case it was detected.

<table>
<thead>
<tr>
<th></th>
<th>Frame #710</th>
<th>Frame #2400</th>
<th>Frame #5050</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw (unprocessed)</td>
<td><img src="image1" alt="Frame #710" /></td>
<td><img src="image2" alt="Frame #2400" /></td>
<td><img src="image3" alt="Frame #5050" /></td>
</tr>
<tr>
<td>Gamma correction</td>
<td><img src="image4" alt="Frame #710" /></td>
<td><img src="image5" alt="Frame #2400" /></td>
<td><img src="image6" alt="Frame #5050" /></td>
</tr>
<tr>
<td>L<em>a</em>b*+CLAHE</td>
<td><img src="image7" alt="Frame #710" /></td>
<td><img src="image8" alt="Frame #2400" /></td>
<td><img src="image9" alt="Frame #5050" /></td>
</tr>
<tr>
<td>RG+CLAHE</td>
<td><img src="image10" alt="Frame #710" /></td>
<td><img src="image11" alt="Frame #2400" /></td>
<td><img src="image12" alt="Frame #5050" /></td>
</tr>
</tbody>
</table>
Table 4: Continued.

<table>
<thead>
<tr>
<th>Bioinspired</th>
<th>Frame #710</th>
<th>Frame #3400</th>
<th>Frame #5050</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Figure 5: Status information from our visual odometry system on different datasets. (a) Number of points in the local map. (b) Number of matched 3D map points with 2D image features to be used in pose estimation. (c) Number of found inliers after pose estimation.

Table 5: Object detector results for ‘Car’ class in the three subsequences with and without preprocessing. Reported results are the mean confidence in the detections, and the detection ratio represents the number of frames in which the detection was successful to the total number of frames in the subsequence.

<table>
<thead>
<tr>
<th>Subsequence</th>
<th>Statistic</th>
<th>Technique</th>
<th>Raw (unprocessed)</th>
<th>Gamma correction</th>
<th>L<em>a</em>b* + CLAHE</th>
<th>RG + CLAHE</th>
<th>Bioinspired</th>
</tr>
</thead>
<tbody>
<tr>
<td>710-730</td>
<td>Mean conf. [%]</td>
<td>73.7</td>
<td>82.9</td>
<td>77.2</td>
<td>85.8</td>
<td>74.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Detection rate</td>
<td>13/21</td>
<td>20/21</td>
<td>20/21</td>
<td>21/21</td>
<td>17/21</td>
<td></td>
</tr>
<tr>
<td>2400-2415</td>
<td>Mean conf. [%]</td>
<td>71.3</td>
<td>69.4</td>
<td>75.1</td>
<td>71.0</td>
<td>68.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Detection rate</td>
<td>12/16</td>
<td>11/16</td>
<td>12/16</td>
<td>15/16</td>
<td>14/16</td>
<td></td>
</tr>
<tr>
<td>5050-5080</td>
<td>Mean conf. [%]</td>
<td>67.3</td>
<td>67.4</td>
<td>67.2</td>
<td>63.5</td>
<td>61.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Detection rate</td>
<td>18/31</td>
<td>21/31</td>
<td>24/31</td>
<td>18/31</td>
<td>19/31</td>
<td></td>
</tr>
<tr>
<td>Combined</td>
<td>Mean conf. [%]</td>
<td>70.3</td>
<td>73.8</td>
<td>72.5</td>
<td>74.3</td>
<td>68.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Detection rate</td>
<td>43/68</td>
<td>52/68</td>
<td>56/68</td>
<td>54/68</td>
<td>50/68</td>
<td></td>
</tr>
</tbody>
</table>

redundancy under these challenging low-light conditions. The presented image enhancements techniques are applicable to a wide variety of computer vision tasks such as image stitching, registration, and recognition. However, in this paper we focused on assisting feature-based visual odometry and visual SLAM systems in operating in such challenging night conditions. Moreover, we studied the performance of presented preprocessing techniques on enhancing the performance of deep learning-based object detector. This is necessary as such vision-based systems are important for autonomous mobile robots which are expected to operate under all illumination conditions.
We have observed that using a retina model inspired from human visual system resulted in best enhancement for visual odometry and visual SLAM systems. However, this comes at the cost of higher computational requirements. Depending on the application, this might be a bottleneck. That said, the authors of the Retina model provide a GPU implementation of their model with the OpenCV library, but we have used the CPU for fair comparison with the other techniques. Gamma correction results came in second after the bioinspired method, but it beats it in terms of computational performance significantly. In fact, gamma correction is the fastest method as it can be reduced to a simple table lookup operation at runtime.

In the object detector evaluation, the retina model did not perform as well as expected. The black-box nature of deep neural network makes it challenging to reason about the cause. Finally, the studied techniques in this paper are all based on conventional image processing methods. An interesting idea to peruse in future work is to learn the preprocessing stage. That is, apply deep learning techniques to train a neural network that would perform this image enhancement for robotic perception. Our focus in this paper was studying potential image preprocessing techniques to enhance the performance of vision-based perception algorithms at night and low-light conditions. The activation of these methods and studying the boundary effects, that is, when transition happens between no image preprocessing...
and when preprocessing is activated due to scene illumination conditions, is left for future work.

**Data Availability**

The night images dataset used in the evaluation performed in this study is sequence 28, and the daytime is sequence 3 of the adverse weather dataset available at http://sar-lab.net/adverse-weather-dataset/.

**Conflicts of Interest**

The authors declare no conflicts of interest.

**Acknowledgments**

The authors acknowledge the undergraduate research assistants Aaron Cofield and Hisham Alawneh for their contributions in data collection. Data collection for this research was supported by a Research Initiation & Development Grant from the University of Michigan–Dearborn Office of Research and Sponsored Programs.

**References**


