**IEEE P802.15**

**Wireless Personal Area Networks**

|  |  |  |
| --- | --- | --- |
| Project | IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) | |
| Title | **Draft D0 Performance Enhancement of Vehicular Communication using Neural Network** | |
| Date Submitted | [September 2021] | |
| Source | Md Faisal Ahmed and Yeong Min Jang | Voice: [ ] Fax: [ ] E-mail: [yjang@kookmin.ac.kr] |
| Re: |  | |
| Abstract | This document discusses about the BER reduction technique using neural network in mobile condition. | |
| Purpose | In order to reduce the BER during the moving scenarios of vehicle, we have proposed a neural network based feature extraction technique for the reformation of the stripe pattern. | |
| Notice | This document has been prepared to assist the IEEE P802.15. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein. | |
| Release | The contributor acknowledges and accepts that this contribution becomes the property of IEEE and may be made publicly available by P802.15. | |

1. System Overview:

We have designed the OCC system using a small LED and an LED array as transmitter and a low-frame rate camera (30 fps) as receiver. We have developed a neural network (NN) to effectively detect the LED(s) in static and mobile scenarios. We have simulated the mobile environment by moving the camera manually. In the existing literature, the BER is high while considering mobility in OCC. We have applied a technique, called Feature Matching, to reduce the BER. The NN employment for the detection of LED(s) and performance enhancement in mobile scenarios are explained in here. Figure 1-1 shows the overall structure of the implemented OCC system, where the input data collected from the sensor and output data retrieved from the receiver are depicted. Moreover, the data pattern of transmitted signals is also demonstrated in binary form. The string and character data produce 1 byte of the binary, which is decoded in the receiver and converted using ASCII code for each output data. Figure 1-1 also shows the dark and bright stripes generated in the image sensor due to the rolling-shutter effect.

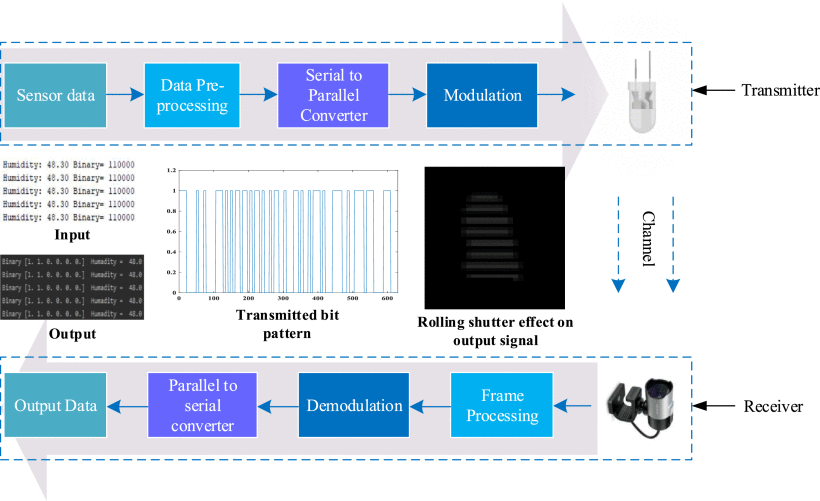


Figure 1.1: The overall OCC system architecture.

1. Training and testing architecture:

An NN model is designed to detect the LED and to support mobility during the detection process. When a frame is processed, the LED position in the image frame is compared with the position measured in the previous frames. If the position is changed, then the scenario is defined as mobile. For the LED detection in this scenario, we have trained 70% of total acquired images using darknet\_no\_gpu.exe and OpenCV in Python. In the training process, we used an LED type called coco\_obj.name where yolov3.weights was used as the weight configuration. Another configuration, called yolov3-tiny\_obj.cfg is used for labeling of the detected LED image to find its exact position. In Figure 2-1 the training architecture is shown along with the stripe pattern reformation. After detection in the moving scenario, the LED image can experience two issues,

The image can be deformed by inclining the LED at any angle and

Some vectors of the LED image can be displaced.



Figure 2-1: Training architecture of neural network based stripe pattern reformation.

As a solution, we have used a technique, referred to as feature matching. In this technique, first, each 5×5 kernel matrix in the image patches are checked using NN regression and compared with reference stripes. The angle yielded due to the deformation is updated using the original inverse deformation, and the displaced vector field is partially reconstructed by filtering. Afterward, every point of the stripes is resampled using spatial transformation to produce a warped image. Using backpropagation connected with NN regression, the steps are repeated in a loop until the original stripes are completely reconstructed. The whole testing procedure of feature matching is depicted in Figure 2-2.



Figure 2-2: Testing architecture of stripe pattern reconstruction.

Due to the movement, the stripes can overlap with each other, consequently increasing the BER. To remediate the issue, we have trained 60% of the total transmitted bits. The trained sets of bits are combinations of character, string, integer, and symbol. If any stripe of the testing image experiences an overlap, it is resolved using the trained dataset. For example, if the NN encounters a sequence with an altered bit, it uses probability to predict the sequence is how much similar to the pre-trained sequence. If the probability exceeds a pre-defined threshold, the sequence is recognized as identical to the pre-trained sequence and the altered bit is replaced accordingly. Besides, in terms of decoding numerical data specifically, the symbol defined before the data is utilized to predict the probability. It is worth noting that the higher the probability (defined in percentage), the faster the recovery from the error.

1. Flow diagram of proposed neural network scheme

The neural network mainly operated based on the feature extraction technique of the stripe pattern. However, the given flow diagram shows the training and testing procedure and checking the similarity of each iteration. In every iteration, the relative position of the stripe is change by checking the similarity with the reference pattern. Figure 3-1 illustrates the entire flow diagram of the proposed neural network.



Figure 3-1: Flow diagram of proposed neural network technique.

1. Interference Subtraction and results

Background lights and neighboring LEDs can create noise and interference in the image sensor, respectively. We have developed a function in Python to find the areas in which the source LED and the interfering light sources appear. The decoding process becomes challenging if the intensity of the background lights is higher or equal to that of the LED. Otherwise, in reasonable conditions, the reconstruction of the signal is possible using a proper combination of a filter and equalizer. A testbed platform is illustrated in Figure 4-1 (a). As shown in Figure 4-1 (b), the detection process is performed by selecting a key point of the light source and subsequently, the width and height of the LED are included constructing a boundary region. As shown in Figure 4-1 (c), the noise generated by the reflection of the surrounding light source is illustrated in the binary image. The LED stripe pattern, after filtering out the noise, is shown in Figure 4-1 (d).

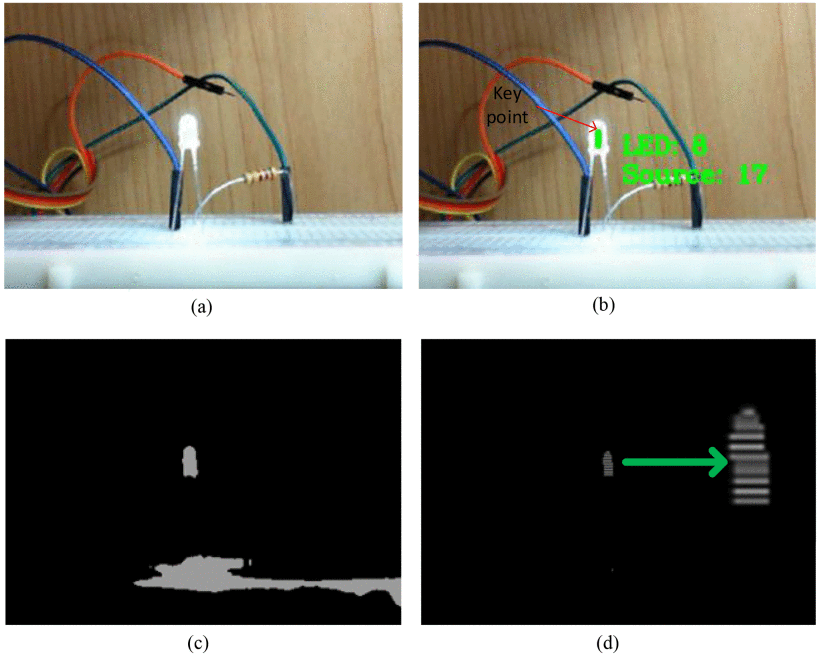


Figure 4-1: (a) Exact LED image during data transmission. (b) detection and classification of LED using NN. (c) LED image with noise. (d) stripe pattern after filtering out the noise.

On the other hand, interference occurs if a LED light source unexpectedly appears inside the image sensor FOV. In particular, after detection, several frames are captured and analyzed. If the stripe pattern of a specific area are changed in the subsequent frames, it is recognized as the source LED. Otherwise, it is categorized as the interfering LED and is filtered out subsequently from the image frame.