Real-time Raindrop Detection System for Autonomous Driving Applications

Abstract –

Context: The presence of raindrop induced image distortion has a significant negative impact on the performance of a wide range of all-weather visual sensing applications especially in the context of vehicle autonomy. Raindrops on the windshield can confuse the visual sensing application of an autonomous vehicle severely. A key solution of this problem is a robust raindrop detection system such that the potential for performance degradation in effected image regions can be identified.

Aims: The aim for this project is to investigate if a solution using Deep Learning approach can solve the problem of raindrop classification and develop a raindrop classification system accordingly. Subsequently, develop a raindrop detection system based on a region proposal algorithm and the raindrop classification system in order to detect raindrops on a windshield. It also aims to evaluate and optimise the performance of the raindrop detection system by using different region proposal algorithms. Furthermore, this project aims to discuss the potential combination of the raindrop detection system with Stereo Vision.

Method: A raindrop classification system based on Convolution Neural Network is built as the fundamental prerequisite of the raindrop detection system (RDS). The RDS is built by concatenating region proposal algorithm with the raindrop classification system. Three region proposal algorithms are implemented in order to optimise the performance of the RDS. A dataset with in-frame raindrop annotation is constructed and categorized into different test sets in order to evaluate the performance of the RDS under different background conditions. A raindrop removal and left/right frame screening method is designed and implemented in order to discuss the potential combination of the RDS with Stereo Vision.

Results: The maximum test accuracy gained for the raindrop classification system is 94.7%. For the RDS, the maximum recall gained is 0.91, however the respective precision is 0.55, a negative correlation between the recall and the precision is observed. Performance optimisation is achieved by using different region proposal algorithms. The potential combination of the RDS and Stereo Vision is demonstrated with the raindrop removal and frame screening method.

Conclusions: Deep Learning approach is very appropriate for raindrop classification problem. The RDS based on this raindrop classification system performed very well in detecting raindrops on a windshield, however the detection precision is affected by other objects within complicated backgrounds. This performance of the RDS proves the feasibility and the potential of Deep Learning approach in raindrop detection problem.

Keywords – Raindrop Detection, Raindrop Classification, Deep Learning, Convolutional Neural Network, All-weather Computer Vision.

I. INTRODUCTION

Autonomous vehicles can detect surroundings by using various techniques such as GPS navigation, radar technology, and Computer Vision. In the context of Computer Vision, the camera is the essential hardware within the system. It captures the environment around the

vehicle and allows the Computer Vision system to analyse and gain high-level understanding from the input video, such as obstacle detection, lane detection, and traffic sign classification.

However, innovation comes with risks. Autonomous vehicles occasionally struggle to "see" in rainy weather as the raindrops tend to confuse the visual sensing applications. For example, an adherent raindrop on the windshield may be classified as a rock and causes emergency brake of the autonomous vehicle. A key solution of this problem is to develop a raindrop detection system such that the raindrops can be detected and the potential of performance degradation in effected image regions can be identified.

Prior work in the field has looked to address the raindrop detection problem by using a combination of saliency map and Support Vector Machine (SVM) classifier method (Wu *et al*., 2012) and achieved a recall (detection rate) of 0.76. Notably, the recent work by (Webster and Breckon, 2015) extended (Wu *et al*., 2012) and proposed an enhanced saliency map combined with SVM/Random Forest classifier method, and increased the recall to 0.86. The major aim of this project is to develop a raindrop detection system (RDS) with a different approach, Deep Learning Convolution Neural Network (CNN), and evaluate to what extent different region proposal algorithms combined with the CNN can be adapted to the raindrop detection problem and evaluate the performance.

A. Nature and Challenges of Raindrop Detection Problem

The raindrop detection problem is essentially a pattern classification problem resolved with the aid of region proposal algorithms. A general workflow of RDS is shown in Figure 1. In Figure 1(a), an image (one video frame captured by a forward facing digital camera within a vehicle) is passed into the RDS. Figure 1(b) shows a demonstration of the regions proposed (in green) within an image. In this step, different region proposal algorithms generate different regions in the aspect of size, location and quantity. Figure 1(c) is the pre-trained raindrop classification system based on CNN. The raindrop classification system takes the proposed regions as inputs and outputs the classification of the region to be either raindrop or non-raindrop. Finally, in Figure 1(d), the RDS labels the region which is classified as a raindrop as shown in the top right hand corner.

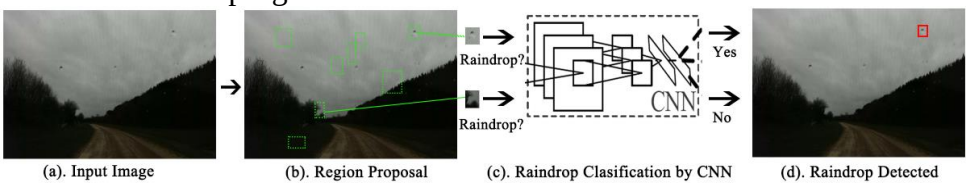

Figure 1. Raindrop detection system workflow.

One of the challenges of the RDS appears in the step of region proposal. It is explicit that the proposed regions within the image have a significant effect on the performance of the detection system. If the proposed regions ingeniously avoid all raindrops within the image, the recall of the RDS would always be 0 no matter how good the raindrop classification system is. However, if the RDS proposed every possible location within the image, the recall could increase to a peak value (the precision will reduce), however this is computationally expensive and infeasible to apply to real world applications in the future.

Another challenge of the RDS appears in the raindrop classification system. The raindrop classification system is the 'brain' of the RDS and the classification accuracy is one of the deterministic factors of the performance of RDS. With a high classification accuracy, the classification system based on CNN should eliminate all proposed regions without raindrops while keeping the raindrop regions. However, as discussed in Section III, this is particular

hard for raindrop classification within a complicated windshield background context. As shown in Figure 2, the shape, colour and texture of raindrops varies significantly in different context. For example, raindrop Figure 2(f) and (g) have a completely different colour. In Figure 2(a), the raindrop has very little contrast difference between the raindrop and the sky. Furthermore, there are numerous objects within a context that are similar to a raindrop in both shape and colour which can affect the decision of the classification system.

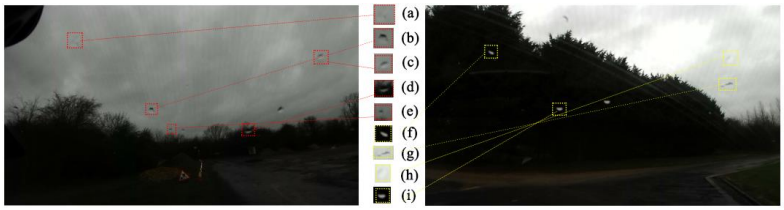

Figure 2. Shape and colour of raindrops vary in different context.

B. Project Objectives and Achievement

For this project, there are two major objectives, the first objective is to develop the raindrop classification system by using CNN approach. The purpose of this objective is to construct the fundamental prerequisite of the RDS as explained earlier in the workflow of RDS. This objective was achieved by training one of the most popular CNN architecture AlexNet (Krizhevsky, 2012) with 8000 training data and numerous parameter configuration experiments. As a contrasting CNN architecture, GoogLeNet (Szegedy *et al.*, 2015) was also implemented in order to compare the performance between two different architectures.

The second objective is to utilise the raindrop classification system and concatenate with a region proposal algorithm to construct the RDS. For this project, three region proposal algorithms were selected from three different image processing categories, the primitive Sliding Window algorithm from image localisation, Super Pixel algorithm from image segmentation, and Selective Search from a combination of image localisation and segmentation. Investigation of the performance change can be gained by adapting these three region proposal algorithms, and the most appropriate algorithm can be found. As a part of performance evaluation of RDS, a dataset with in-frame raindrop annotation was constructed and divided into several different test sets to assess the robustness of the RDS under different conditions. An extended objective of this project is to discuss the application of the RDS. A raindrop removal and frame screening method was implemented based on the RDS in order to demonstrate the combination of the RDS with Stereo Vision.

By successfully fulfilling the above objectives, a maximum test accuracy of 94.7% was gained for the raindrop classification system. For the RDS, the maximum recall gained was 0.91 which is above the state of the art and the three region proposal algorithms were evaluated in three aspects: the recall, precision and computation time. Raindrop removal and frame screening methods were implemented and demonstrated.

II. RELATED WORK

Although the presence of raindrop induces image distortion has significant negative impact on the performance of the visual sensing applications of autonomous vehicles, however, the research that has been conducted on how to detect raindrops is rather sparse. In the early researches, raindrop detection was performed by looking at the movement of raindrops in consecutive frames of a video (Garg and Nayar, 2004). However, these movement detection

methods require the background to remain static and it is designed for CCTVs. Researches in the field of raindrop detection for the real-time is a relatively novel field. In this section, overview of different techniques in the field of raindrop detection such as consecutive frames analysis and single frame real-time detection are given.

A. How Does A Camera See Raindrops?

To understand how a system detects raindrops, it is important to understand when a camera sees a raindrop. In (Garg and Nayar, 2005), the author explained that the raindrop produced high frequency spatio-temporal intensity fluctuations which further cause increases in pixel intensity in a specific region and this is how a camera 'sees' a raindrop. The research also indicated that the visibility of the raindrop is dependent on various factors such as the camera settings. To alter the camera parameters such as exposure time and the depth of field can form a naive raindrop removal effect.

However, changes in camera parameters, such as an increase of exposure time may result in motion blur from any movement in the scene. Therefore, this early raindrop removal method is not suitable for the visual sensing system of an autonomous vehicle as the captured scene contains real-time movements.

B. Raindrop Detection in Consecutive Frames

Another raindrop detection approach is to derive the intensity of a given frame by using a Kalman Filter (Park and Lee, 2008). The estimated intensity is calculated for the next frame, raindrop regions are therefore identified via the comparison of the actually intensity and the estimated intensity. However, the assumption of this algorithm is that the camera and the scene are always fixed. The algorithm assumes that the background does not differ in a number of frames, otherwise the estimated intensity of the frame would always be wrong and the algorithm will fail. In this case, this algorithm has limitations on automotive use.

To estimate the motion trajectories of an object in successive frames, (Shen and Xue, 2009) proposed an algorithm for raindrop detection problem. The author made use of the motion data to differentiate raindrops from other moving objects. As the movement of a non-raindrop object is predicted for the next frame, subtracting the next frame from the current frame would generate the regions where raindrops are not present and reversely, the regions that contains the raindrop movement can be discovered. This algorithm assumes the raindrop movement is much faster than other objects in the scene. This is true for raindrops within the air, however, as the raindrop tend to adhere to the windshield of the vehicle with little movement in most of the situations, this algorithm is not suitable for autonomous vehicle.

C. Raindrop Detection in Single Frame

A novel approach that detects raindrops on a windshield using only one single image from an in-vehicle camera was proposed by (Halimeh and Roser, 2009). The research focused on the geometric-photometric properties of a raindrop and establishes a relationship between the raindrop and the environment. The proposed algorithm describes the refractive properties of a raindrop. It traces the rays going through the raindrop from the environment in to the camera and determines the part of the scene refracted by the raindrop in order to detect the raindrops within an image. A fundamental assumption for this research is that all raindrops follow one geometric photometric property. However, as shown previously in Figure 2, the raindrops differ in many aspects especially in shape, size and colour.

Observation shows that raindrop are high contrast, round in shape and texture-less regions compared to regions contain other objects in the image. Therefore, by analysing the colour, texture and shape characteristics of raindrops in the image and produce a global saliency map of the raindrops within the image, work by (Wu *et al.*, 2012) identifies possible raindrop candidates, which are small locally salient droplets in a global saliency map as shown in Figure 3(B).

Figure 3. Proposed raindrop candidates (Wu *et al.*, 2012).
(A) Original image with region of interest (yellow rectangle). (B) Raindrop candidates.

The global saliency map is formed by combining the colour texture and shape map as shown in Figure 4. As illustrated in Figure 4, the global salient map was formed by a weighted combination of three other saliency maps, the colour, texture and shape (blue rectangles on the left) saliency map with Gaussian Pyramid and Hough Transform algorithms respectively. After the global saliency map is generated, the candidate regions are then studied by a Support Vector Machine to classify if the potential candidate is a raindrop. The workflow is very similar to this project, but this project adapted different region proposal and raindrop classification methods. The result for (Wu *et al.*, 2012) are promising, a recall of 0.76 and precision of 0.79 was gained.

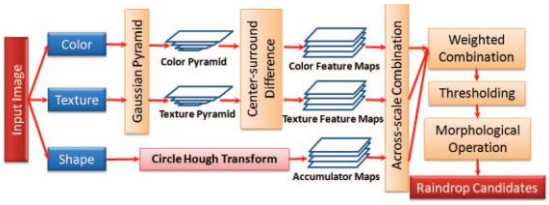

Figure 4. Proposed raindrop saliency map generation workflow (Wu *et al.*, 2012).

The work of (Webster and Breckon, 2015) evaluated the saliency method for raindrop detection proposed by (Wu *et al.*, 2012) with the focus on improving both recall and precision, (Webster and Breckon, 2015) extended (Wu *et al.*, 2012) by adapting an extended feature descriptor comprising localized shape descriptor by using (Hu, 1962) and the saliency and texture information are isolated from the overall scene context. Furthermore, the research of (Webster and Breckon, 2015) also adapted a new raindrop classification method Random Forest classification to compare with the original Support Vector Machine. The result achieved by (Webster and Breckon, 2015) was recall of 0.86 and precision of 0.92, which is the state of the art result for the real-time raindrop detection problem.

D. Summary

Although many researches have been conducted into raindrop detection problem, most of these researches were designed for stationary cameras such as CCTV rather than moving autonomous vehicles. Two researches explicitly aimed to work within the context of the autonomous vehicle real-time processing are (Wu *et al.*, 2012) and (Webster and Breckon, 2014). This project looks to extend the (Webster and Breckon, 2015), therefore both results produced by (Wu *et al.*, 2012) and (Webster and Breckon, 2015) are important performance indicators and contrasts of this project.

III. SOLUTION

The solution designed to build the raindrop classification system and the raindrop detection system (RDS) is presented in this section. The testing, verification and performance optimisation process for both systems are also outlined. To start, an overview of the general architecture of the solution is given, subsequently, the detailed solution and the underlying algorithms are described in the sub-sections.

A. Solution Architecture Overview

The overview of the solution architecture is illustrated in Figure 5. As explained earlier in Section I, this project can be divided into two major parts, the raindrop classification system and the RDS. Consequently, two timelines are shown in Figure 5, the one on the top is the major procedures for the construction and optimisation of the raindrop classification system, and the one at the bottom is for the RDS respectively. As the RDS is based on the raindrop classification system, therefore interactions between the two systems are frequent.

The raindrop classification system starts with the preparation of the training and testing dataset (Webster-Dataset) as shown in Figure 5 (a). The detail of the dataset and dataset expansion steps will be given in the sub-section. Subsequently, both CNN architectures AlexNet and GoogLeNet are implemented with the open source Deep Learning library TensorFlow and its high level API TFLearn (Martin *et al.*, 2016). Training and testing carried out a number of times in order to optimise the parameter configurations for both CNN architectures as shown in Figure 5(c). Finally for the raindrop classification system, after the performance analysis and comparison for both CNN architectures, a well-trained CNN model for raindrop classification can be obtained.

The RDS begins with the construction of the windshield dataset with in-frame raindrop annotations (Windshield-Dataset) as shown in Figure 5(e). This dataset is used for performance evaluation purpose. Subsequently, by assembling the well-trained CNN model for raindrop classification and three region proposal algorithms (Sliding Window, Selective Search and Super Pixel) one at a time, the RDS is formed as shown in Figure 5(g). Various experiments were conducted to test the performance and to optimise the parameters configuration for each region proposal algorithm. The well-trained CNN model is re-trained repeatedly with new training data extracted from the Windshield-Dataset in order to improve the robustness of the RDS in diverse background conditions.

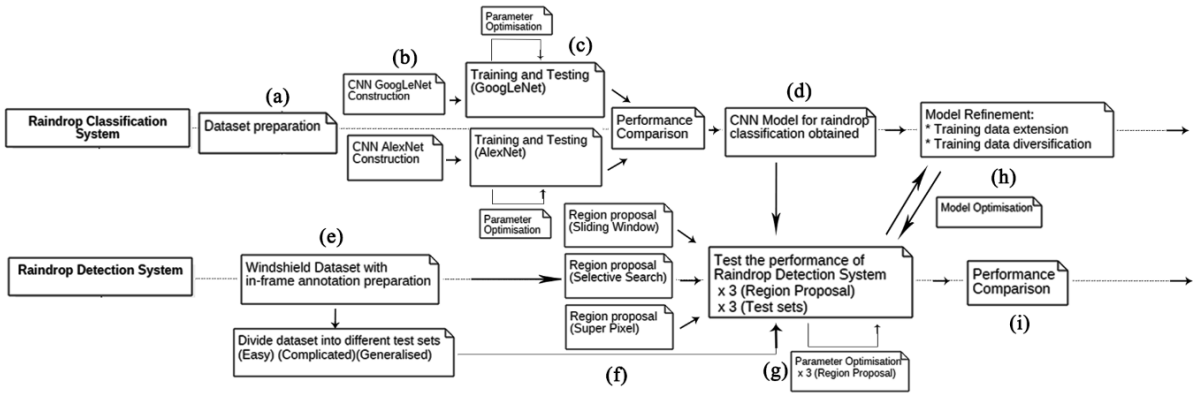

Figure 5. Overview of the solution design. (a) - (i) are the main components of the solution.

B. Raindrop Classification System

i. Dataset and CNN

The prerequisite step before constructing the raindrop classification system is to prepare the dataset. The original dataset (Webster-Dataset) for this project was inherited from (Webster and Breckon, 2015), the images in the dataset were extracted from a source video gathered using a forward facing digital camera mounted behind a car windshield under a variety of road environments. There are 16,000 images in the dataset and approximately 8000 contains raindrop and 8000 without raindrop. The dataset was divided into approximately 8000 training data (4000 raindrop/40000 non-raindrop) and 8000 testing data (4000 raindrop/4000 non-raindrop).

There are various techniques to classify the objects such as Support Vector Machine adapted by (Webster and Breckon, 2015). However, in recent years, as a category under Deep Learning, the Convolutional Neural Network (CNN) has been proved very effective in image recognition and classification. The Large Scale Visual Recognition Challenge (ILSVRC) evaluates algorithms and CNN architecture for image classification at large scale every year and becomes the most important competition in this field. For this project, the ILSVRC winner CNN architectures AlexNet (Krizhevsky, 2012) and GoogLeNet (Szegedy *et al.*, 2015) were adapted. The reason to use AlexNet is that this architecture is one of the most popular CNN architecture in the field of image recognition and has been cited more than 6,000 times in image recognition researches, it was the winner of ILSVRC 2012 with a top 5 error rate (given an image, the model cannot output the correct classification with its top 5 predictions) of 15.4% which is very powerful. The reason to adapt GoogLeNet is because the novel concept Inception Module it introduced that makes the GoogLeNet have 22 layers whereas AlexNet only has 13, but GoogLeNet has 12 times fewer parameters than AlexNet. The GoogLeNet was the winner of ILSVRC 2014 with a top 5 error rate 6.7%.

While the detailed explanation of the underlying theory of Deep Learning and CNN is beyond the scope of this paper, however, before the introduction of AlexNet and GoogLeNet, some basic concepts about CNN should be outlined. The major difference between CNN and Neural Network (NN) is the convolutional layer in CNN. For image classification problem, the scalability is always a problem for NN. A 200 × 200 image will produce 40,000 input units and 2 billion parameters in a fully connected layer of NN. However, the convolution operation provided by the convolutional layer in CNN extracts features by using a set of filters (e.g. Edge detection, Gaussian blur, etc.) from the input image and preserves the spatial relationship between pixels by learning image features using small squares of input data. After the convolution operation is done, a set of feature maps of the original image will be generated. Additionally, another important layer in CNN is the Spatial Pooling Layer. This layer reduces the dimensionality of each feature map but retains the most important information. The widely used Pooling Layer in nowadays is the max and mean Pooling Layer. To introduce non-linearity in the CNN, each convolutional layer also adapted an activation function exactly like the hidden layers in NN, the widely used activation functions for CNN are ‘ReLU’, ‘Softmax’ and ‘Tanh’ (Hahnloser *et al.*, 2000).

ii. AlexNet

The architecture of AlexNet is shown in Figure 6. The 5 cuboids in the middle of the architecture are the convolutional layers and there are 3 max pooling layers implemented after the first, second and the last convolutional layer. The 3 rectangles at the tail of the

architecture is the fully-connected layer. ReLu non-linearity is applied to the output of every convolutional and fully-connected layer. The equation of ReLU is shown in Eq. (1). The benefit of using ReLU rather than Tanh is the sparsity of connections as the *max* function in Eq. (1) assures that learning will only happen in the neuron with a positive input. Therefore, the sparsity arises when $a \leq 0$. The more such units exist in a layer, the sparser the resulting representation. Consequently, the training speed by adapting ReLU is much faster.

$$f(x) = \max(0, a) \tag{1}$$

An additional local response normalization is applied after applying the ReLU non-linearity in the first and second convolutional layers. The additional local response normalization implements a form of lateral inhibition, which is the capacity of an excited neuron to subdue its neighbours. The benefit of adding this extra normalization is when this normalization is applied to the local neighbourhood of an excited neuron, this neuron will become even more sensitive compared to its neighbours and dampen the responses that are uniformly large in any given local neighbourhood. Subsequently, this feature will boost the neuron with relatively larger activations and create a clearer contrast in that area.

To prevent overfitting, (Krizhevsky, 2012) adapted two different methods in AlexNet. The first method is Dropout, which consist of setting to zero of the output of each hidden neuron with probability 0.5. The neurons which eliminated from this step do not contribute to the forward pass or back-propagation. A benefit of this feature is the AlexNet samples a different architecture for every input. The second method is the Data Augmentation, which is widely used in many Machine Learning researches. The author used image translation or horizontal reflection methods to create artificial image data to enlarge the dataset.

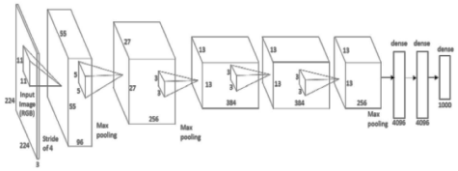

Figure 6. Illustration of the architecture of AlexNet (Krizhevsky, 2012)

iii. GoogLeNet

The complete architecture of GoogLeNet is shown in Figure 7(a). At first glance, the GoogLeNet is notably larger than AlexNet. However, different combinations and quantity of convolutional layers and pooling layers are not novel, the significant difference between GoogLeNet and AlexNet is that GoogLeNet runs in parallel rather than sequentially like AlexNet. As shown in Figure 7(a), GoogLeNet is mainly formed by 9 identical blocks, which is called the 'Inception Module'. A closer look at the Inception Module (90 degree rotation) is shown in Figure 7(b). In AlexNet and many other traditional CNNs, for each layer, a choice of operation has to be made between the convolution and pooling. For the convolution operation, a choice of filter size also has to be made. As described in the beginning of this section, all different types of operations in CNN is particular useful. Therefore, the idea of the Inception Module is to perform convolution operation with filter size 1×1, 3×3, 5×5 and pooling operations at the same time in parallel. However, the problem of this idea is the numerous parameters used within each layer of such a parallel operation. However, if a low dimension 1×1convolution operation is performed before the 3×3 and 5×5 convolution operation as shown in Figure 7(b), the yellow blocks, the dimension of the input can be reduced sharply and therefore the size of the parameters can be reduced accordingly. As shown in Figure 7(b), there is a single 1×1 convolution layer on the left of the Inception

Module which guarantees the minimum information loss by this dimension reduction method. The green blocks are a concatenate layer, which assembles the output of each operation and passes on-to the next Inception Module.

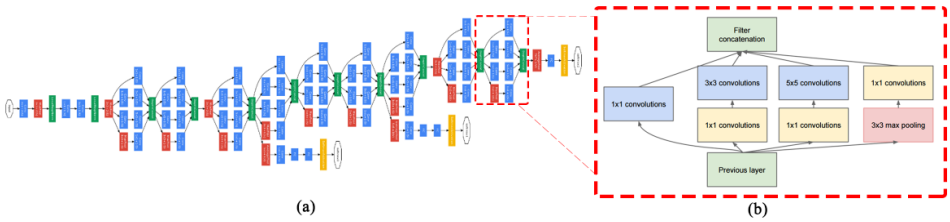

Figure 7. Illustration of the architecture of GoogLeNet. (a) Complete architecture of GoogLeNet. (b) Inception Module. (Szegedy and Liu, 2015)

iv. Implementation, Testing and Improvement

Both AlexNet and GoogLeNet were implemented using Python with its open source Deep Learning library TensorFlow and high-level API TFLearn. The built-in functions provided by TFLearn are particularly convenient and intuitive in the layer construction. For example, to build a convolutional layer, only the number of neurons, filter size, stride size and activation function have to be declared. The standard image data format in TensorFlow is named 'tensor', which is a similar numerical representation of image like other image processing libraries. The 'image to tensor' transformation function is provided by TFLearn.

The performance of the raindrop classification system is heavily affected by the training settings. Similarly to most of machine learning techniques, the most important parameters for CNN training is the number of epoch, batch size, validation set ratio and learning rate. The epoch and batch size is the determinate factor of the number of iterations, which is the number of passes of a batch size of training data. The more iterations during the training, the higher training accuracy gained, however this also leads to overfitting and reduce testing accuracy. The learning rate is specified by the author of AlexNet in order to achieve the best performance of the architecture.

After the training of the CNN, the model of this CNN containing the weights for each layer is saved. This saved model is used for future trainings on top of this trained CNN with new training data, and consequently, the performance optimisation is achieved. This model will be embedded as a brain into the RDS as shown previously in the solution architecture overview in Figure 5(d). This model is also used to test the performance with the test data (8000 raindrop and non-raindrop images) prepared earlier. For each image, the model will generate the classification (a number between 0 and 1, the number larger than 0.5 indicates raindrop). The parameter configurations were tested through multiple experiments according to the test accuracy.

C. Raindrop Detection System

i. Dataset with In-frame Raindrop Annotation

As previously shown in Figure 5(e), the construction of the RDS starts with the preparation of dataset with in-frame raindrop annotations (Windshield-Dataset). The importance in having this dataset is that during the performance evaluation stage of the RDS, the raindrop annotation which indicates the exact location of raindrops will become the ground truth value for that image, and the detection accuracy can be assessed consequently. To build this dataset,

similar to the Webster-Dataset, four clips of videos were taken by using a forward facing camera mounted behind a car windshield under rainy weather as the 'raw material', subsequently, the videos were decomposed into approximately 30,000 frames (1280×720 image resolution). Different from the Webster-Dataset, this dataset is mainly used for testing purpose, in this case, only the frames containing raindrops are retained, therefore thousands of consecutive frames were eliminated as it was not raining at that period. To increase the diversity of the dataset, frames with raindrops in various background conditions are collected. As shown in Figure 8, (a) is the most common background conditions, whereas (b) and (c) can be used to test the RDS in backlighting conditions, (d) and (e) both have distractions such as tree branches and windshield wipers to test the performance of the RDS under interference. As a result of carefully screening, approximately 400 images were collected as the candidate images for the dataset.

Figure 8. Images with various background conditions.

To add annotation of the raindrops within these images, the most popular image labelling tool, LabelMe (Russell *et al.*, 2008) was used. LabelMe is a web-based image annotation platform with succinct functions. By uploading the images to the platform, it allows user to label objects and name the objects. After labelling, user can download the images back with an extra annotation folder containing the label information for each image respectively. The label information is stored in a XML file with the coordinate information for each annotation, as it is in XML format, this coordinate information is convenient to extract and consequently, the raindrop location on an image is known during the testing stage.

The Windshield-Dataset is further divided into 4 sub test sets. One generalised test set with 208 images containing all different kind of background conditions, this test set is the main test set for the RDS performance evaluation. Two small test sets in which one contains only the images with simple background conditions (simple set) such as Figure 8(a), one contains images with only complicated background conditions (complicated set) such as Figure 8 (d) and (e). The other set is prepared as material to extract more raindrop and non-raindrop images in order to expand the training set of the raindrop classification system. This will be explained in detail later in the performance optimisation section. After the dataset is ready, the next step is to implement the region proposal algorithms.

ii. Region Proposal Algorithms

(1) **Sliding Window**

For any object detection system, if the object classification system is considered as the brain, the region proposal algorithm can be seen as the eyes. The detection system needs to develop a way to 'look at' the image in order to detect an object. In this case, the most intuitive method is to examine every location within an image as to not miss any potential object location. This type of method is called exhaustive search which is a very primitive image localisation method and the most popular algorithm within this category is Sliding Window. In image processing, the name Sliding Window is very self-explanatory. The basic idea is to move a window along the image, where certain image processing is performed on the underlying sub-image.

In this project, to construct the RDS, by using Sliding Window algorithm to slide a box around an image and crop the sub-region inside the box, a Region of Interest (ROI) is obtained. Subsequently, using the raindrop classification system to classify this ROI, retains the location information of the ROI as if it has been classified as a raindrop. When every location within the image has been examined, the retained ROI locations are therefore the detected raindrops within this image and raindrop detection is complete. However, the Sliding Window algorithm itself has several drawbacks. Searching every possible location is computationally infeasible, therefore the size of the sliding box and the stride of sliding has to be fixed. Even in this case, the number of locations to visit still remains considerably huge and the computation time is long. As this is a drawback that has been proven in the RDS performance evaluation, other types of region proposal methods are considered in order to optimise the performance.

(2) **Super Pixel**

The Sliding Window algorithm uses the pixel-grid as underlying representation, the sliding of fixed-size box is not a natural representation of visual scenes and it is an artefact of a digital imaging process. A more natural and most importantly, more efficient method in object detection problem is image segmentation. In this project, the second region proposal algorithm adapted is the Super Pixel algorithm (Ren and Malik, 2003). The idea of Super Pixel algorithm is to partition an image into hundreds of segments in which (Ren and Malik, 2003) named 'superpixels' and therefore reduces the complexity of images from hundreds of thousands of pixels to only a few hundreds of superpixels. Superpixels are perceptually meaningful as each superpixel is a perceptually consistent unit of colour, texture and shape and therefore the pixels belonging to each superpixel is the same set of pixels for the same object in an image.

There are several different approaches realised the Super Pixel concept, the one adapted in this project is (Van den Bergh and Boix, 2012) and the authors named this approach as SEEDS. SEEDS starts from a complete superpixel partitioning, and the algorithm iteratively refines the partitioning. The refinement is done by moving the boundaries of the superpixels and the aim is to coincide the boundaries of partitioning to object boundaries within the image. To determine the appropriate movement of boundaries, a hill-climbing algorithm is used. Based on the hill-climbing algorithm, a proposed movement for refinement of the superpixels is accepted if the result of the objective function increases.

The objective function is shown in Eq. (2), the aim of boundary movement is to maximise the $E(s)$ where s is the partitioning (superpixels) of the image. The $H(s)$ and $G(s)$ evaluates the colour distribution and the shape of the partitioning respectively. The equation of $H(s)$ and $G(s)$ is shown in Eq. (3) and Eq. (4). In Eq. (3), A_k is the set of pixels in one superpixel, and cA_k is the colour histogram of this set of pixels. $\Psi(cA_k)$ is the function that enforces the colour histogram is concentrated in as few colours as possible. The $\Psi(cA_k)$ reaches its maximum when the histogram is concentrated in one bin and reaches its minimum in case that all colour bins take the same value, which means the objective function $E(s)$ reaches a maximum value when each pixel within a superpixel has the same colour. Applied to the raindrop detection problem, this would partition the raindrop within the image as a superpixel and propose this superpixel as a ROI to the raindrop classification system.

$$E(s) = H(s) + \gamma G(s) \tag{2}$$

$$H(s) = \sum\nolimits_k \Psi(cA_k) \tag{3}$$

$$G(s) = \sum_i \sum_k (bN_i(k))^2 \quad (4)$$

In Eq. (4), N_i is a square patch of size $N \times N$ around a pixel i, the idea is that a superpixel has a better shape when most of the patches contain pixels from one unique superpixel. Therefore, the *G(s)* uses the same measure of quality as *H(s).* If the patch N_i contains pixels set k within a unique superpixel, *G(s)* is at its maximum. The nature of the *G(s)* is to penalise patches containing several superpixel labeling and reduce the amount of pixels closed to a boundary, and thus enforces regular shapes. This *G(s)* can ensure the rounded shape of a raindrop which will not be cut into irregular shapes and reduce the classification accuracy. A demonstration of SEEDS is shown in Figure 9, (a) is the initialized partition, (b) and (c) is the iterations of exchanges pixels on the boundaries between neighbouring superpixels. After the iterations of boundaries movement is over, the superpixels is determined, in this project, the final superpixels are the ROIs within each image and is passed to the raindrop classification system same as the Sliding Window algorithm.

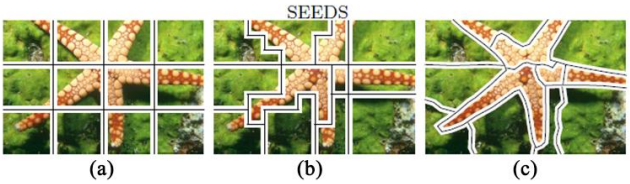

Figure 9. SEEDS demonstration (Van den Bergh and Boix, 2012)

(3) **Selective Search**

The work of (Girshick and Donahue, 2014) proposed a region proposal algorithm that consist both concepts of image localisation and segmentation. The Selective Search algorithm is intuitive, at the outset stage, the algorithm randomly propose a large number of regions within the image, this step is very similar to exhaustive search, subsequently, the algorithm groups similar regions into one region by considering the colour, shape and location of the regions and selects final candidates regions from the grouped regions, this step is utilising the concept of image segmentation.

iii. Implementation, Testing and Improvements

As explained above, the RDS is a combination of the raindrop classification system and region proposal algorithm, as the raindrop classification CNN model is gained from previous step, therefore, the construction of the RDS to this stage is left with the implement of region proposal algorithms. In this project, the image processing open source library, OpenCV (Bradski, 2000) was used to implement the three region proposal algorithms. Similar to the raindrop classification system, the parameter configuration for three region proposal algorithms was obtained from numerous experiments. The Windshield-Dataset prepared was used to test the RDS with different region proposal algorithms against different background conditions as explained previous. The best results for each algorithm was collected and compared with each other in order to evaluate the most appropriate algorithm for raindrop detection problem.

For the performance evaluation of the RDS, the recall and the precision is the most important performance index. As shown in Figure 10(a), in the initial tests, although the RDS managed to detect all 3 raindrops within the image, it also generated many error detections. The tree branches and the road are major distraction of RDS. The reason is because under certain conditions, raindrops and non-raindrop objects appear very similar in terms of colour, texture and shape, especially around the tree branches. As shown in Figure 11, (a) is a set of

extracted regions of raindrop and (b) is set of extracted regions of non-raindrop objects and they look very similar to each other. In these circumstances, an effective way to reduce the distraction effect is to expand the training data in the Webster-Dataset of the raindrop classification system with these distraction objects, as a result, the classification system would have chance to learn about these objects. As mentioned above, apart from 3 different test sets in the Windshield-Dataset, an additional set of images was separated from the 400 candidate images. In this case, thousands of tree branches, road, traffic signs and other objects were extracted from the material images in this set, and separated into training data and testing data. To prevent the loss of generalisation, approximately 30% of raindrop and non-raindrop images from the original Webster-Dataset were also randomly chosen and mixed with these distraction objects in order to form an enhanced Webster-Dataset. As explained earlier, TensorFlow provides functions to continue training a CNN with a saved model with its trained weights for each layer and, by utilising this function, several new trainings were conducted on the saved model with expanded training and testing data in order to further refine the model. As shown in Figure 10(b) and (c), this model refinement method optimised the performance of the RDS significantly.

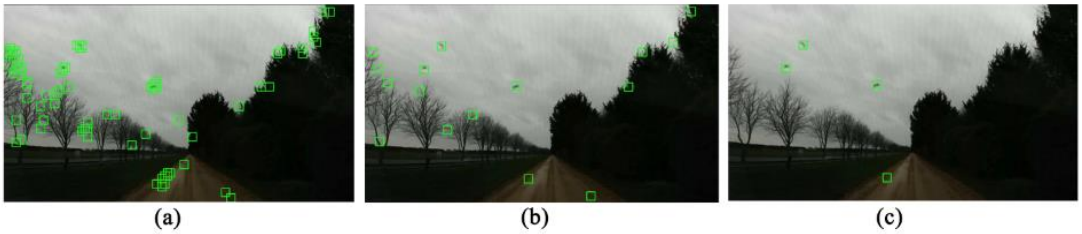

Figure 10. Result of model refinement. (a) The raindrop classification system detected all 3 raindrops in the image but interfered by distractions. (b) Train the model with new training data. (c) Result after several round of model refinement.

Figure 11. Raindrops and non-raindrops are similar in colour, texture and shape under certain conditions. (a) Raindrops. (b) Non-raindrops.

D. Application

According to the basic concept of Stereo Vision, the disparity is the measurement of feature displacement between the left and right images. This means that if the system extract features for the same object from both images, then the depth can be recovered from the matched features. In this case, the raindrops adhere to the windshield in front of the two cameras becomes a distraction of the Stereo Vision system. As the raindrops in front of each camera is randomly located, if the system tries to match these raindrops, the feature matching will fail. In order to prevent this, two solutions were proposed and implemented as a demonstration of the combination of the RDS with Stereo Vision. The first method is to manually remove the raindrops within the left and right frames and send the 'clean' frames to the Stereo Vision application. An Inpainting algorithm (Telea, 2004) was adapted in this project in order to remove the detection raindrops by the RDS. Another method is to prevent the frames with huge number of raindrops to be processed by the Stereo Vision application, a frame screening method was implemented in order to discard the frames if they contain too many raindrops, and a threshold is set in order to adjust the level of screening. The underlying idea of this method is to discard the frames that have a high possibility to cause matching failures as they

contain too many distracting objects. As this is a discussion of the future application of the RDS, therefore only a demonstration program is built for the purpose of discussion and consequently no performance evaluation is done for the demo program.

IV. RESULTS

In this section, the results of the raindrop classification system and the RDS are presented. Parameter configuration, model optimisation, experiment settings and performance analysis are shown. For both raindrop classification and the RDS, every experiment was done more than two times with the exactly same parameters or conditions in order to validate the experiment's results and also to eliminate arbitrary error results, the mean of the results was calculated as the final result and listed in the tables. During all parameter configuration experiments, when one parameter was altered, the others were frozen to keep a baseline in order to validate the correctness of the parameter configuration experiment. The experiment environment system specification is outlined in Table 1.

TABLE 1. EXPERIMENT ENVIRONMENT SYSTEM SPECIFICATION

Component	Title of Component
Processor	Intel Core i5-4200M 2.50 GHz
GPU	NVIDIA GeForce 840M
Ram	8GB RDIMM
Operating System	Linux Ubuntu 16.04

A. Raindrop Classification System

The first set of experiments was done for the raindrop classification system by using three CNN architectures AlexNet, AlexNet-Shrink and GoogLeNet. The important parameters for CNN training are the number of epoch and the learning rate. As for both AlexNet and GoogLeNet architectures, the learning rate was suggested by the author in order to achieve the best capability and to avoid overshooting, therefore, the suggested value 0.001 was chosen and fixed. Figure 12 and 13 shows the training (dot) and testing (cross) accuracy against epoch for AlexNet and GoogLeNet. As shown in Figure 12, the AlexNet reaches its peak training accuracy 99.81% at epoch around 150, the testing accuracy also reaches its peak value 94.69% at the same time. After this, the training accuracy remains at this level, however, if the training keeps running, severe overfitting occurs immediately and the test accuracy will reduce to less than 90%. An additional CNN architecture 'AlexNet-Shrink' was constructed by reducing half of the neurons in each layer of AlexNet and observe the performance change, as shown in Table 2, the computation time reduced by approximately 6%, however the testing accuracy also degraded by 1%. As shown in Table 2, although both peak training and testing accuracy for GoogLeNet are less than AlexNet due to the dimension reduction concept of Inception Module, however the parallel operation provided by the Inception Module reduced the training time significantly. From Figure 13, the overfitting occurs after achieving the peak testing accuracy of 93.63% at epoch 200 and reduces to 88% at epoch 250. The true positive rate (TPR) and true negative rate (TNR) for all three architectures are also presented in Table 2. As shown in Table 2, the AlexNet architecture has a higher TNR, which indicates that this architecture is slightly more robust at classifying non-raindrop objects than raindrops. The peak training (99.81%) and testing accuracy (94.69%) is gained from the AlexNet architecture. For object classification problem, the testing accuracy is the only performance

indicator, in this case, the model used in RDS is trained from the AlexNet.

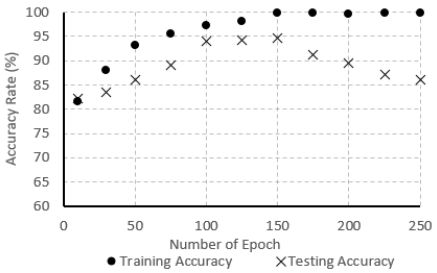

Figure 12. Number of epoch vs. accuracy rate for AlexNet

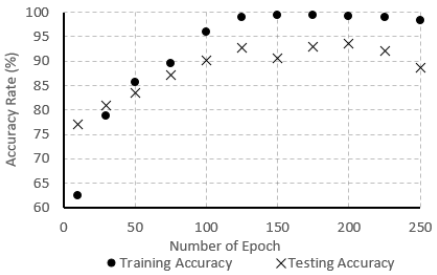

Figure 13. Number of epoch vs. accuracy rate for GoogLeNet

TABLE 2. BEST TESTING ACCURACY ACHIEVED FOR ALEXNET AND GOOGLENET

CNN architecture	Training accuracy (%)	Testing accuracy (%)	TPR (%)	TNR (%)	Training time (Hour)
AlexNet, Epoch = 150	99.81	94.69	92.69	96.96	7.6
AlexNet-Shrink, Epoch = 150	98.70	93.77	91.41	96.13	7.1
GoogLeNet, Epoch = 200	99.14	93.63	96.27	90.63	2.4

B. Raindrop Detection System

All parameter configuration experiments of the three region proposal algorithms were performed by using the generalized test set with 208 images containing all kinds of background conditions in the Windshield-Dataset. After the parameters are settled, the 'simple' and 'complicated' test sets are used to evaluate the robustness of the RDS. The final performance evaluation of the RDS is tested against the generalized test set. The complete parameter configuration experiment data is too huge to be included into this paper, therefore only the most representative parameter configurations are listed. For raindrop detection problem, there are two important performance indicators, the first one is the Recall, also known as the true positive rate (TPR), which is the fraction of relevant instances that are retrieved, in this project, it reflects how many raindrops have been detected from the ground truth raindrops. The second indicator is the Precision, known as positive predictive value (PPV), the fraction of retrieved instances that are relevant, in this project, it reflect the detection accuracy, in other words, the lower the precision, the higher the error detections generated. The work by (Webster, and Breckon, 2015) also included the Accuracy of the detection, however, as different algorithms were adapted in this project, for example the Sliding Window algorithm proposes 8000 regions within one image, as most of the non-raindrop regions are eliminated by the RDS, the true negative (TN) of the RDS for one image would be approximately 7990 regions whereas the true positive (TP), false positive (FP) and false negative (FN) is only a number below 10, in this case, the only decisive variable for calculating the Accuracy will only be the extremely high TN and leads to an extremely high Accuracy. As a result, the Accuracy for the RDS in this project is not a meaningful statistical measurement.

The first set of experiments was done on the Sliding Window algorithm because the Sliding Window is the most primitive algorithm used in object detection problem. As explained earlier, the size and the stride size of the sliding box is fixed, as a result of this, there is no valuable parameter configuration results for Sliding Window. However, in this project, the Sliding Window algorithm is used to evaluate the model refinement. As shown in

Table 3, by using the model trained by the original Webster-Dataset, the precision of 0.25 is much lower than the state of the art. An illustration was shown in Figure 10(a) in the previous section that the RDS generated numerous error detections. However, by adding more distraction images such as the sky, road, tree branches and wipers into the Webster-Dataset gradually and re-trains the model, as shown in Table 3, the precision increased from 0.25 to 0.55 and the recall also increased from 0.83 to 0.91.

TABLE 3. MODEL REFINEMENT WITH SLIDING WINDOW

Sliding Window	Recall	Precision
Original	0.83	0.25
+ sky, road	0.87	0.39
+ tree branches	0.89	0.53
+ windshield wiper	**0.91**	**0.55**
Simple set	0.94	0.91
Complicated set	0.85	0.36

The parameter configuration experiment results of Super Pixel and Selective Search is shown in Table 4 and Table 5. For Super Pixel, the performance determinate parameter is the number of superpixels (*num_sp*) and the number of iterations (*i*). For Selective Search, the '*scale'* indicates to the preferred size of the proposed regions and '*max'* is the maximum size of the region, if 'max' is reached, the grouping of regions stops.

The reason to list these results is because some of the parameter configuration either generates the peak recall or precision value of the algorithm, or generates a good balance between the recall and the precision. The grey highlighted rows in the tables are the representative results selected for this particular algorithm as the results shows a good balance between the recall and the precision. As shown in Table 4, the Super Pixel algorithm reached a peak recall at 0.95 and a peak Precision at 0.71, however the recall and precision balance is not ideal for these parameter configurations. In Table 5, the performance result of Selective Search algorithm is listed, however, the Super Pixel and Sliding Window algorithm outperformed the Selective Search algorithm in both recall and precision. The best recall achieved by Selective Search is only 0.71, although the precision appears to be better than the representative result of the other two algorithms, however, this is a result of the negative correlation between the recall and precision which will be discussed in the next section.

TABLE 4. PARAMETER CONFIGURATION EXPERIMENT RESULTS FOR SUPER PIXEL

Super Pixel	Recall	Precision
num_sp =700, i = 5	0.60	**0.71**
num_sp = 1300, i = 5	**0.95**	0.39
num_sp = 1500, i = 5	0.86	0.27
num_sp = 1000, i = 12	0.89	0.57
num_sp = 1000, i = 20	0.83	0.69
Simple set	0.94	0.90
Complicated set	0.84	0.34

TABLE 5. PARAMETER CONFIGURATION EXPERIMENT RESULTS FOR SELECTIVE SEARCH

Selective Search	Recall	Precision
Scale = 500, max = 50	0.64	**0.84**
Scale = 250, max = 50	0.67	0.58
Scale = 50, max = 50	0.74	0.54
Scale = 10, max = 100	**0.71**	0.61
Scale = 10, max = 200	0.48	**0.84**
Simple set	0.79	0.87
Complicated set	0.64	0.41

Figure 14 is a visual comparison of the performance of three algorithms. The y-axis is a scale without specific unit as the bars have different units. The grey bar on the right indicates the computation time for an algorithm to process one image. The computation time to process one image for Sliding Window, Super Pixel and Selective Search are 82, 13, 27 seconds respectively. This means that, although the Sliding Window algorithm outperformed the

Super Pixel in terms of recall slightly, in terms of computational efficiency, the Super Pixel algorithm is 6 times faster than the Sliding Window algorithm.

In order to evaluate the robustness of the RDS, the 'simple testing set' and the 'complicated testing set' is used and the results are shown in the last two rows in Table 3, 4 and 5. The results gained by Sliding Window and Super Pixel algorithm by using the 'simple' set are promising, the recall is better than the state of art. For 'complicated' set, the recall is still at the same level of the state of art, however, the precision drops rapidly.

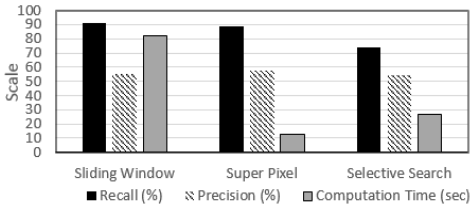

Figure 14. The best performance generated by three region proposal algorithms

The result of the RDS is illustrated in Figure 15. In Figure 15 row (a), the green areas are the proposed regions by three algorithms from left to right, and in row (b), the green boxes are the detection results generated by the RDS by using that particular region proposal algorithm and the red boxes are the ground truth raindrops. In Figure 15 (b) Sliding Window result, all 4 raindrops have been detected, however 2 error detections can be observed at the bottom left of the image. The raindrop removal and disparity map generated by Stereo Vision application after the raindrops are removed is shown in Figure 16, as shown in (b), all 3 raindrops have been removed from the original image (a). The frame screening method is implemented as described in the Solution section, a pair of frames can be automatically discarded according to pre-set criteria. These results proves the ability of the RDS to assist Stereo Vision applications in rainy weathers by removing raindrops or discard frames covered by raindrops before being processed by Stereo Vision application, consequently, the matching failure would be decreased.

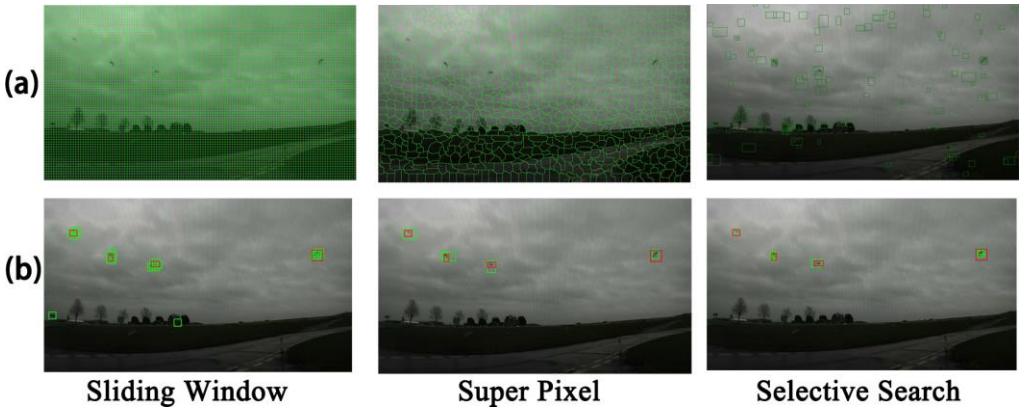

Figure 15. Demonstration of three region proposal algorithm and the respective detection results.

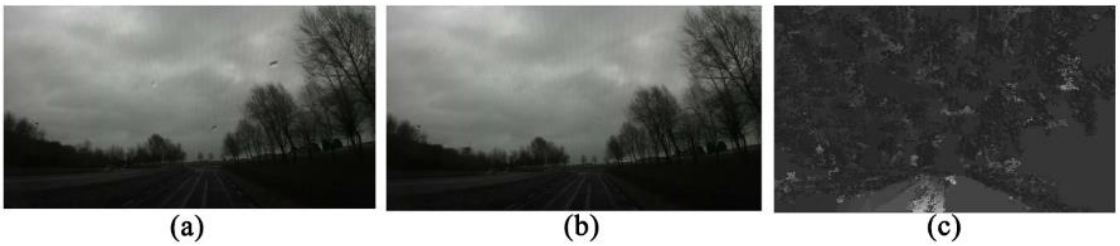

Figure 16. Demonstration of raindrop removal and disparity map. (a) Original image. (b) Image after raindrop removal. (c)Disparity map generated by Stereo Vision application.

V. EVALUATION

In this section, an evaluation of the solution is given by using the results presented in the previous section. The strength and weakness of the solution is discussed and, the performance of the RDS is compared with the state of the art.

A. Strength and Limitation

The result shown in Table 2 proves that CNN is very appropriate for raindrop classification problem. There is only 1.06% difference between the testing accuracy of AlexNet and GoogLeNet which further proves the generalisation and indicates that both CNN architectures are practical in this field. The training time shows that if the computational efficiency is sensitive, the GoogLeNet can be a better option. Overfitting is a major problem in many Deep Learning approaches and also in this project, however the turning point of overfitting was revealed by numerous experiments as shown in Figure 12 and 13 for both architectures.

The parameter configuration experiment results shown in Table 4 and 5 exhibit the ability of the RDS to be easily optimised by altering just a few parameters of the region proposal algorithm. This is a great advantage compared to some other algorithms, in which the optimisation can only be gained through re-designing. The peak recall and precision result gained through the experiments is valuable, the peak recall of 0.95 gained by Super Pixel and the peak precision of 0.84 gained by the Selective Search shows the extreme capability of each algorithm, and indicates the relationship between certain parameters and the performance. Additionally, these data can be used as a reference in further research.

As shown in Table 3, the RDS was very responsive to the model refinement, the expanded training data generalised the original Webster-Dataset, and increased the recall and precision gradually. This result testified to an efficient way to optimise the performance of the RDS.

In the robustness evaluation, the results shown in the last two rows of Table 3, 4 and 5 indicates that all three algorithms are robust in good background conditions, the RDS were able to detect most raindrops within an image. However, in complicated background conditions, the RDS shows a degradation in the performance, especially in precision. The RDS generates number of error detections due to distractions, such as tree branches. Figure 17 further demonstrates the relationship between the recall and the precision by calculating the mean of the results generated during the parameter configuration experiments for all three algorithms. The negative correlation between the precision and the recall becomes a significant limitation during the performance optimisation of the RDS. High recall or precision can be generated by sacrificing against each other, however this is not practical in real-world applications.

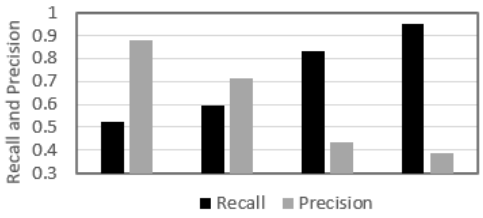

Figure 17. An illustration of the negative correlation between the recall and the precision

B. A Comparison with the State Of The Art

The performance results of the RDS by using three different region proposal algorithms are listed in the first three rows in Table 6 and the performance results of (Webster and Breckon,

2015) and (Wu *et al*., 2012) are listed in the last three rows in Table 6. By comparing the results generated by three region proposal algorithms in this project, the performance of the Selective Search falls under the average performance. The Sliding Window and Super Pixel algorithm achieved very similar recall and precision result, however, the Super Pixel algorithm wins in terms of efficiency with a huge pace. As the RDS is designed for real-time use, therefore, the Super Pixel generates the best performance within the three region proposal algorithms comprehensively.

As shown in Table 6, from previous work, the highest recall generated was 0.87 from (Webster and Breckon, 2015), in this case, the recall gained by using the Sliding Window (0.91) or Super Pixel (0.89) has surpassed the state of the art in raindrop detection problem by 4% in terms of recall. However, as mentioned earlier, the balance between the precision and the recall is a main limitation of the RDS, both (Webster and Breckon, 2015) and (Wu *et al*., 2012) had generated a relatively high precision and achieved a good balance. The highest precision gained by the RDS system is only 0.69 which is under the state of the art.

TABLE 6. PERFORMANCE COMPARISON WITH (WEBSTER AND BRECKON, 2015) AND (WU *ET AL*., 2012)

Different Approaches	Recall	Precision
Sliding Window + CNN	**0.91**	0.55
Super Pixel + CNN (1)	0.89	0.57
Super Pixel + CNN (2)	0.83	0.69
Selective Search + CNN	0.71	0.61
Webster: Enhanced saliency map + Random Forest	0.87	0.75
Webster: Enhanced saliency map + SVM	0.86	**0.92**
Wu: Saliency map + SVM	0.76	0.79

C. Improvement

If given a chance to repeat the project, the extension objective of this project, the combination of the RDS with Stereo Vision part would be carefully revised. This is because although both of the combination plans, the raindrop removal method and the frame screening method were successfully implemented and the result was demonstrated in the previous section. However the performance of this combination is only proved visually and it is hard to evaluate the performance through experiments scientifically at this stage within the scope of this project. It may have been better to spend more time in further optimising the recall and the precision of the RDS.

VI. CONCLUSION

In terms of fulfilling the aims, the project was an overall success. The raindrop classification system by using CNN approach was implemented and verified with two different CNN architectures, the AlexNet and GoogLeNet. The RDS based on CNN approach combined with three region proposal algorithms, Sliding Window, Super Pixel and Selective Search algorithms were also implemented and the performance was evaluated. The robustness of the RDS was evaluated in various background conditions, the strength in the recall and the limitations of the negative correlation between recall and precision of the RDS was discussed. The raindrop removal and frame screening method were implemented and demonstrated, the possibility of the combination of RDS with Stereo Vision was discussed.

The main contributions of this project are as follows: by achieving a testing accuracy of

94.69%, the capability of the Deep Learning CNN approach is verified and performed very well in the field of raindrop classification problem. The peak recall of 0.91 proved that the region proposal algorithms combined with CNN approach is very capable in the field of raindrop detection problem. Three region proposal algorithms were tested and the recall, precision and computation time was evaluated and compared.

A suitable direction for future work would be to investigate more methods in order to minimise the negative correlation between the recall and precision, and subsequently, increase the robustness of the RDS under complicated background conditions. A combination of image pre-processing prior to raindrop detection could be a direction. Another path of investigation would be the optimisation of the computation time to process each image, as the RDS is designed for real-time processing, therefore the computation time will become an important requirement in the future.

REFERENCES

Bradski G., (2000). "The OpenCV Library", in *Dr Dobbs Journal of Software Tools,* pp. 120-125.

Garg K., Nayar S., (2004). "Detection and removal of rain from videos", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 528-535.

Garg K., Nayar S., (2005). "When does a camera see rain?", in *Proceedings of the IEEE International Conference on Computer Vision,* pp. 1067-1074.

Girshick R., Donahue J., Darrell T., and Malik J. (2014). "Rich feature hierarchies for accurate object detection and semantic segmentation", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 580-587.

Hahnloser R, Sarpeshkar R, Mahowald M., (2000). "Raindrop detection on car windshields using geometric-photometric environment construction and intensity-based correlation", in *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 610-615.

Halimeh J., Roser M., (2009). "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit.", in *Nature.* pp. 405.

Hu M., (1962). "Visual pattern recognition by moment invariants", *in Information Theory, IEEE Transactions on*, pp. 179-187.

Krizhevsky A., Sutskever I., Hinton G., (2012). "ImageNet Classification with Deep Convolutional Neural Networks", in *Advances In Neural Information Processing Systems*, pp. 1-9.

Martin A., et al., (2016). "TensorFlow: a system for large-scale machine learning", in *OSDI'16 Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation,* pp. 265-283.

Park W., Lee K., (2008). "Rain Removal Using Kalman Filter in Video", in *2008 International Conference on Smart Manufacturing Application*, pp. 494-497.

Ren X., Malik J., (2003). "Learning a classification model for segmentation", in *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 10-17 vol.1.

Russell B. C., Torralba A., Murphy K.P., Freeman W. T., (2008). "LabelMe: a database and web-based tool for image annotation", in *International Journal of Computer Vision,* pp. 157-173.

Shen M., Xue P., (2011). "A fast algorithm for rain detection and removal from videos", in *Proceedings - IEEE International Conference on Multimedia and Expo*, pp. 1-6.

Szegedy C., Liu W., Jia Y., (2015). "Going deeper with convolutions", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 941-944.

Van den Bergn M., Boix X., Roig G. Van G., (2015). "SEEDS: Superpixels Extracted Via Energy-Driven Sampling", in *International Journal of Computer Vision*, pp. 298-314.

Webster D. D., Breckon, P. T., (2015). "Improved Raindrop Detection Using Combined Shape and Saliency Descriptors with Scene Context isolation", in *Image Processing (ICIP), 2015 IEEE International Conference, pp*. 4376 - 4380.

Wu Q., Zhang W., and Kumar V.B., (2012). "Raindrop detection and removal using salient visual features", in *Proceedings – International Conference on Image Processing*, pp. 941-944.

You S., Tan R., Kawakami R., and Ikeuchi K., (2013). "Adherent raindrop detection and removal in video", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1035-1042.