UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

# CS/IT Honours

# Final Paper 2019

## Title: TREE SEGMENTATION BY COMBINING FCNS WITH ENGINEERED FEATURES

Author: Charl Ritter
Project Abbreviation: TREESEG
Supervisor(s): Assoc. Prof. Patrick Marais and Assoc. Prof. Deshen Moodley

| **Category** | *Min* | *Max* | Chosen |
|---|---|---|---|
| Requirement Analysis and Design | *0* | *20* | |
| Theoretical Analysis | *0* | *25* | |
| Experiment Design and Execution | *0* | *20* | |
| System Development and Implementation | *0* | *15* | |
| Results, Findings and Conclusion | *10* | *20* | |
| Aim Formulation and Background Work | *10* | *15* | |
| Quality of Paper Writing and Presentation | *10* | | 10 |
| Quality of Deliverables | *10* | | 10 |
| Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*) | *0* | *10* | |
| **Total marks** | | **80** | |

# TREE SEGMENTATION BY COMBINING FCNS WITH ENGINEERED FEATURES

Charl Ritter
Department of Computer Science
University of Cape Town
RTTCHA002@myuct.ac.za

## ABSTRACT

With the technological advances in the agricultural sector and drone technology, as well as a surge in the popularity of Neural Networks and Convolutional Neural Networks there exists an interest to investigate whether the image segmentation performance of Convolutional Neural Networks can be improved for tree data. This interest is explored further in this paper, by posing the question; whether using engineered features as inputs to neural networks can improve its segmentation performance for tree data. To test this, the following research questions were asked. The first question being whether the use of engineered features has a positive influence on the accuracy of a neural network when segmenting tree data. The second question being whether the segmentation accuracy of certain engineered features is influenced by the number of layers added to it.

To answer these questions a feature extraction layer, that pre-processes all the images and extracts the various features, was developed. A presentation layer was developed for testing purposes and an FCN was implemented. Eight different feature extractors transform methods were tested. These transforms fell into either the per-pixel transform or small-width feature extractor categories. To evaluate the results of the testing, the mean IOU and average precision metrics of the various transforms were compared.

After analysing the data from the tests, the following became apparent. Some of the feature extractors do positively influence the accuracy of the FCN. Adding multiple layers to the feature extractors was also found to improve the accuracy of the FCNs predictions. The per-pixel transforms were most competent at improving the segmentation performance of the FCN. Additionally, it was found that mask generation needs to be rigorous, otherwise, the use of feature extractors will not aid the FCN any more than normal RGB images will.

## CCS CONCEPTS

• Machine Learning • Image Processing • Computer Vision

## KEYWORDS

Image segmentation, Convolutional Neural Networks, Fully Convolutional Network, Per-pixel Transforms, Small-Width Feature Extractor, Large-Width Feature Extractor.

## 1 Introduction

The agricultural sector plays a big role in the way humans currently live their lives. The sector has been subject to a lot of advancements in the past 60 years. Advances like modern irrigation, GMOs, pesticides and fertilisers allowed farmers to increase the output of their farms while decreasing the labour needed to maintain the farm. In recent years with the new advances in drone and Artificial Neural Networks (ANNs) technology, farmers can now monitor the health of their farm by having it surveyed by drones. These drones use two cameras. The first is a high-resolution camera, which captures RGB (Red, Green, Blue) and Digital Elevation Model (DEM) images. The second camera captures the Normalized Difference Vegetation Index (NDVI) and Near Infrared (NIR) images. These images can then be analysed by a trained ANN. The trained ANN will segment the crops from any background and using the different layers of the image, it will determine whether the crops are healthy and alert the farmers of any abnormalities.

Although ANNs achieve state of the art segmentation accuracy for many applications, ANNs still produce segmented objects that have inaccurate boundaries or have low-quality resolutions. Furthermore, these architectures are data-intensive, requiring huge training samples to learn basic image features [7, 11].

The research conducted in this paper arose from the issues described above. This paper is a partial continuation of previous research conducted by the University of Cape Town Computer Science Honours students, to determine whether Convolutional Neural Networks (CNNs) or Random Forest Models (RFMs) are more accurate at segmenting tree data [2, 8]. In these respective papers, it was found that CNNs outperformed the RFMs. It was also identified by Aerobotics, a company that offers several

drone-based services for farmland monitoring and management, that using engineered features may be able to improve the segmentation of ANNs. The question posed in this paper is, whether using engineered features as inputs to CNNs can improve its segmentation performance for tree data. The types of CNNs were limited to U-Nets, Fully Convolutional Networks (FCNs) and Atrous Convolutional Networks.

Presented in this paper is the research, results and subsequent conclusions drawn from exploring the question posed above, by experimenting on an FCN. An FCN is a neural network that uses a fully connected layer to improve segmentation [7]. In total eight different types of feature extractors were implemented. Five of the feature extractors were Per-pixel transforms and the rest were Small-width feature extractors.

The project team hypothesized that the use of engineered features as inputs for the various CNNs would improve the CNN's segmentation performance on tree data. It was, however, observed that segmentation performances between different unsupervised architectures had an almost negligible difference in the projects preceding this one [2, 8]. Similarly, many research projects dedicate a huge amount of resources into developing new CNNs or engineered feature models, only to experience a very small increase in either efficiency, speed or both. Thus, although it was hypothesized that there would be an improvement in the segmentation performance for tree data of CNNs, whether the improvement will be significant is uncertain. The outcome of this project allows any group wishing to conduct research in this field to decide whether they wish to; adopt this technology into their projects, invest more resources into further research on certain aspects of the project or to discontinue research in this field in favour of another. It also displays the effectiveness or ineffectiveness of feature extractors to improve image segmentation of CNNs, allowing other researchers to build on this premise and develop better feature extractor methods or combinations thereof.

The project team devised the following research questions for this project:

1. Do pre-processed engineered features used as inputs to neural networks positively influence the accuracy of the neural network when segmenting tree crops, over the base neural network band?
2. Is the segmentation accuracy of certain engineered features positively influenced by the number of layers added to it?

# 2 Background

## 2.1 Artificial Neural Networks

ANNs are architectures which use various connected layers. The various layers consist of input and output layers, as well as different hidden layers. The results of each layer are passed to the next layer and the operation weights are adjusted by the error produced in the classification. ANNs are excellent tools for finding patterns.

A high-level example of how an ANN works would be, if the network is intended to recognize an object in an image, the first layer may analyse something like the brightness of its pixels. The next layer could then identify any edges in the image. After this, another layer may recognize textures and shapes, and so on. By the time the nth layer is reached, the ANN will have created complex feature detectors. It can then determine certain image elements which are commonly found together. After labelling of the outputs and by using backpropagation, the network can perform classification tasks on its own.

### *2.1.1 Convolutional Neural Networks*

There is a wide variety of ANNs that perform image segmentation. The highest performing of these are CNNs. CNNs are machine learning models that use layers of connected neurons to perform an extremely wide variety of tasks, especially image classification [7]. This is done by assigning importance to different objects in an image and the ability to differentiate one object from another. CNNs make strong and mostly correct assumptions about the nature of images, which has been proven by Krizhevsky et al. Their results showed that a large, deep convolutional neural network can achieve record-breaking results on highly challenging datasets, using purely supervised learning [6].

### *2.1.2 Fully Convolutional Networks*

An FCN is defined by Guo et al., as an extension of the CNN model, where the basic idea is to make the CNN take an input of arbitrary-sized images [3]. The main characteristic that differentiates an FCN from other CNNs, is that the last fully connected layer is substituted by another convolution layer with a large receptive field. One of the drawbacks of FCNs is that, because of the fixed nature of the size of the receptive field, if an object is substantially larger or smaller than the size of the receptive field, it could be mislabelled or fragmented [9]. Another drawback is that the resolution of the feature map is downsampled due to all the convolutions and pooling layers that the data goes through. This leads to low-resolution predictions, which in turn leads to fuzzy object edges [3].

## 2.2 Image Features

### *2.2.1 Per-Pixel Transforms*

These are modifications dedicated to transforming single pixels. The primary focus of manipulating the pixel values of images is changing the format of the colours they to represent, as well as isolating the components those formats consist of. Some examples of this type of transform, are; LAB, HSI, HSL, ICA, and PCA colour space transform [20].

#### 2.2.2 *Small-Width Feature Extractors*

These are modifications dedicated to transforming a small neighbourhood of pixels. The focus of this transformation type being, modifications looking specifically at smaller objects in an image or at parts of an image. Some examples of this type of transform, are; Edge-detector, Mean-shift and Histogram Equalisation transforms [1, 4, 10, 13, 17, 23].

#### 2.2.3 *Large-Width Feature Extractors*

These are modifications dedicated to transforming a large neighbourhood of pixels [22]. The process of segmenting a neighbourhood of pixels typically follows one or more pre-processing steps to isolate key information. However, sometimes these strategies can be applied to the source image depending on the application. Some examples of this type of transform, are; Hough and Templating Matching transforms.

## 3 Related Work

There seemed to be no other papers directly dealing with the questions asked in this paper, however, there were papers related to feature extraction of vegetation imagery that guided this project.

In terms of the feature extraction of vegetation imagery. Wang et al., focused primarily on the LAB colour space method, which is a small feature extractor. They used this method to identify trees in images by looking for colours matching the green spectrum. They found that the method was very fast but had some shortcomings when the shades of green of different plants that were next to each other were too similar. This resulted in them being grouped [18]. In a follow-up paper, Wang et al. separated individual trees from the surrounding landscape in images. For images with simple backgrounds, they proposed using Edge Detectors, however for images with more complex backgrounds they devised their own method, based on Edge-detectors coupled with colour space methods, of which they recommended the LAB colour space method and mathematical morphology based on their results [19].

Zheng et al., firstly applied two per-pixel feature extractors, namely HSI colour spaces and RGB colour spaces, to the images. After this, they applied a small feature extractor, namely Mean-shift, to the images. This separated the images into two parts, green and non-green vegetation. They tested this algorithm on 100 images of different plant types, with different illuminations and soil types and achieved a median of mis-segmentation of 4.2% between green and non-green vegetation. They did note however that although this method produced an improved result, it suffered from a long runtime [23]. In the paper by Tao et al., the Mean-shift feature extractor was also used. They, however, applied the feature extractor first to the images and then they applied the Normalized-cuts method. By applying the Mean-shift method first, it allows the Normalized-cuts method to be applied to the resulting segments of the Mean-shift, rather than directly to the image pixels. They found this combination requires significantly lower computational complexity and therefore, is feasible for real-time image processing while yielding superior image segmentation performance [13]. Finally, Tellaeche et al., used a per pixel feature extractors and large feature extractors. They first apply an RGB colour space method to the images, which results in a binary image. This image shows crops as white and the soil, stones and residual as black. After the binary image is obtained, they apply a Hough transform to it, to detect the furrows of the crops in the image. Tellaeche et al., did not comment on the performance of this combination on its own, but that when used with Support Vector Machines, they found that it performed well, although results did vary in overexposed images [16].

## 4 Framework Design and Development

The project framework was split into three distinct stages, namely the pre-processing stage for the creation of engineered features, the testing stage for testing the impact on accuracy that these features have on the respective CNN models and thirdly, the implementation of a presentation layer to aid in the collection of statistical outputs. It was hypothesized that the use of the pre-processed image features would lead to an increase in the accuracy of the various CNN models tested.

### 4.1 Design

The overall structure of the software that needed to test the hypotheses was split into the following layers; a feature extraction layer, a CNN layer and a Tensorflow presentation layer. The overall structure of the implementation can be seen in Figure 1. The raw image data were fed into the feature extraction layer, where the various transforms were applied to the image data and saved. The image masks and plain RGB images were also saved in this layer to use in validation and obtaining the ground truth. All the processed images were then fed into the CNN layer. This layer was broken into the three different types of CNNs, as it was tested independently by the members of the project team. The model was trained and evaluated on the image data in various tests. Finally, the results from the testing were sent to the presentation layer, where it was stored and presented.

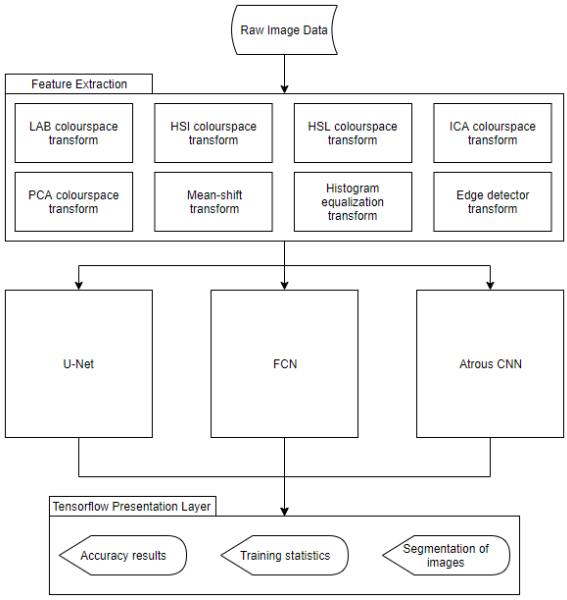


**Figure 1: Structure of the planned implementation**

## 4.2 Development Tools

This project was developed in Python, as Python has many frameworks and libraries useful for the project. The libraries that were used for this project were; OpenCV, ImageMagick, Matplotlib, NumPy, SciKit-Learn, SciKit-Image, pymeanshift, KittiSeg and TensorFlow. NumPy was used to create numpy arrays out of the H5 files containing the image data. OpenCV, ImageMagick, Matplotlib, SciKit-Learn, SciKit-Image and pymeanshift were used to perform the transformations on the dataset. KittiSeg was used as the FCN implementation. TensorFlow was used to monitor the training and evaluation of models.

## 4.3 Implementation

### *4.3.1 Driver Class and Feature Extractors*

The first stage required the creation of a driver class for the image feature extraction. This was done by the combined efforts of the group. Implementing each of the actual transformations within this driver was done individually, resulting in the driver class having eight different feature extractors.

For the Per-pixel transforms, the project team implemented LAB, HIS, HSL, ICA and PCA colour space transforms. For the actual implementations of these five, the LAB and HSI colour space transforms used the ImageMagick library, the HSL colour space transform used the matplotlib library, the ICA colour space transform used the OpenCV library and the PCA colour space transform used a method from a GitHub repository.

For the small-width feature extractors, the project team implemented Edge-detector, Mean-shift and Histogram Equalisation transforms. For the actual implementations of these three, the Edge-detector and Histogram Equalisation transforms was implemented by using the OpenCV library, and for the Mean-shift transform a GitHub repository implementation was used fjean's pymeanshift [5]. This GitHub repository was decided on, as OpenCV had no Mean-Shift method. The repository allowed for the easy tweaking of parameters and upon visual inspection produced the most competent Mean-shifts from the different repository's that were explored.

Finally, for the large-width feature extractors, the project team originally planned to implement Hough transforms and Template matching, however, after analysis of the datasets, the large-width feature extractors were abandoned, as it would not be helpful for the type of data provided.

### *4.3.2 Convolutional Neural Networks Implementation*

As input for the next stage, the feature extractors were applied to raw images. These images could then be fed into the various CNN models. To establish a base-accuracy to compare to, the CNN models were also trained on raw image data. The second stage revolved around testing these transformations with three different CNN models, which was performed individually. Each team member explored one of the three CNN models.

For the implementation of the FCN and its training and evaluation environment, a GitHub repository implementation was used MarvinTeichmann's KittiSeg [14, 15]. Contained in this repository, is MarvinTeichmann's implementation of an FCN, that was based off the paper by Long, et al [7], and the KittiSeg pipeline, which made the training and evaluation of models much easier. This repository was chosen as;

- the FCN was based off a well-cited paper.
- the pipeline made it easy to tweak the FCN parameters, as well as train and evaluate using the Aerobotics datasets.
- the FCN made use of a VGG16 numpy file that contained pre-trained weights, which reduced the training time needed to produce a competent model.
- the Kittiseg project was created originally to segment images of roads into two classes, road and not road. This project deals with segmenting tree data into two classes, tree and not tree. Thus, minimal tweaking was needed for the FCN.

There were minimal changes made to the code to make it usable for this project. A config file was written in order to use the Aerobotics dataset and to tweak the parameters of the FCN, a method was removed that would check for and then download the original CityScape dataset, a dataset of road images that the KittiSeg project used, and the calculation of the mean Intersection Over Union (IOU) metric, as well as the appending of said metric to a log file and tensorboard, was added.

*4.3.3 Presentation Layer*

The third stage was to implement a presentation layer to collate and clearly show various statistical outputs measured from training. The TensorFlow library was used to achieve this. It was used as it is a leading framework in the machine learning industry and has vast support and documentation which allowed the easy implementation of the planned design.

# 5 Experiment Design and Execution

## 5.1 Hardware

All the tests were run on the same machine. This machine had an Intel Core i7 3770 CPU, 16GB DDR3 RAM and an NVIDIA GeForce GTX 1060 6GB GPU, which was running Ubuntu 18.04.

## 5.2 Training Data

The training data used consisted of datasets provided by Aerobotics. These files were in the H5 file format, which consists of multidimensional arrays of data. These files were created from the multispectral images taken from drones owned by Aerobotics.

These files contained different layers. As mentioned before, each image contained an RGB layer, a DEM layer, which is a 3D representation of a terrain's surface created from a terrain's elevation data [21], an NDVI layer, which measures the difference between the light that is a reflected and absorbed by vegetation [12], an NIR layer, which picks up all the infrared light in the range of 700nm-1400nm on the electromagnetic spectrum and a mask layer, which is the ideal a layer used in training to compare.

Two data sets of differing sizes were considered to perform tests on. A small dataset, which consisted of 636 images and a large dataset which consisted of 1942 images. As time was limited and a large dataset takes more time to train with, it was necessary to establish whether a large dataset would improve the segmentation performance of the FCN versus an FCN trained using a small dataset. Eight tests were run, four on each dataset, to determine which performed best. The results from these tests showed that the FCN trained on the small dataset performed slightly better than the FCN trained on the large dataset and thus the small dataset was used in all subsequent tests.

The images in the small dataset were split 90/10, where 90% was used to train the model and 10% was used to evaluate the model. To make up the 10% of the dataset used for validation, the first 10% of every 100 images were extracted and added to the validation dataset, to ensure the validation dataset represents the whole dataset and not just a niche section of it. Among all the tests conducted, the training and evaluation images were kept the same and in the same order to ensure consistency. Some examples of the test data are provided in Figure 2.

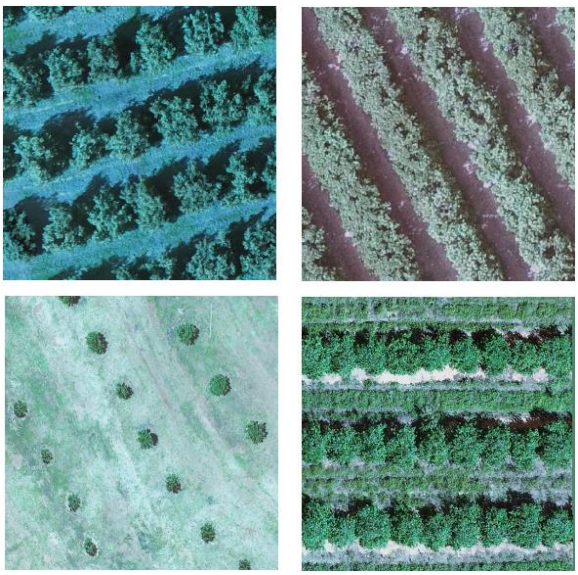

**Figure 2: Examples of the test data**

## 5.3 Overview of Experiment

To ensure the experiment was rigorous, the reproducibility of the experiment was of utmost importance. The first step of the experiment was to train and evaluate the FCN model on unprocessed RGB images to observe the base accuracy of the model. After this, the FCN was trained and evaluated on the pre-processed engineered features to validate the improvement in segmentation accuracy. Great care was taken to ensure the raw and transformed images were in the same order and came from the same base image. Once this was completed, the FCN model was trained and evaluated again on the ground truth RGB images and on all the feature extractors, with the only difference being that the images also had the DEM layer added. This test would then be able to determine whether adding layers, specifically the DEM layer, would improve the segmentation accuracy of the feature extractors. For this experiment, all tests ran for exactly 5000 epochs.

To decide how many epochs each test should run for, the FCN was trained on the RGB single-layer images while the rate of increase in the mean IOU was determined every 2500 epochs. For the first 2500 epochs, the mean IOU increased by 6.18%. For the second 2500 epochs, the mean IOU increased by only 0.09%. After making this observation, it was concluded that the rate of increase in the mean IOU, using the FCN trained on the RGB single-layer images, did not materially increase after 5000 epochs. It was therefore decided to end all tests after 5000 epochs.

Validation of the prediction against the original image was performed every 100 epochs. To make visual comparison possible, the prediction was overlaid over the original image. In the prediction, the colour orange denotes trees while the colour blue denotes everything that is not trees. A series of predictions is shown in Figures 3, 5 and 6.

During training the TensorFlow framework provided real-time metrics, allowing the evaluation of the segmentation accuracy. Segmentation accuracy validation is represented as a percentage change measuring the predicted segmentation map to the correct one.

### 5.4 Evaluation

The base segmentation performance of the FCN model was determined by training and evaluating it on the raw RGB images and comparing this segmentation performance to the segmentation performance of the models trained and evaluated on feature extractors. This experiment was further split into two parts. Firstly, the segmentation performance of the FCN when given only transforms made of single layers were tested. Secondly, the segmentation performance of the FCN when given transforms made of multiple layers was tested. To answer the other research questions, the segmentation performance on a single layer was compared to the segmentation performance on multiple layers.

The two metrics used to asses segmentation performance was mean IOU and the average precision of the models. IOU looks at the ground-truth bounding boxes and the predicted bounding boxes from the model. It then takes the area of overlap between the two boxes, over the area of the union of the boxes. This then gives a metric between zero and one hundred on how closely these two boxes match, zero being not at all and one hundred being perfect. Thus, the IOU shows how closely the model predicted the bounding box of the object versus the actual bounding box of the object, for a prediction. Mean IOU is then just the average IOU taken over the course of the model's evaluation and training. Precision, on the other hand, expresses the percentage of the relevant predictions. This metric is also evaluated between zero and one hundred, zero meaning that the model made many false-positive readings, while one hundred means that there were no false positives. Thus, average precision is just the precision averaged over the course of the model's evaluation and training. The accuracy was also computed, although as this project deals with an imbalanced classification problem, it is not a good measure for assessing the model's performance. It was therefore not considered in the evaluation of the FCN. Mean IOU and average precision are defined by the following formulas. Where given;

Precision = P, True Positives = TP,

False Positive = FP, False Negative = FN,

Total Precision = TOTP, Total IOU = TIOU,

n = Current Epoch,

One can define;

$$P(n) = \frac{\mathrm{TP(n)}}{\mathrm{TP(n) + FP(n)}}$$

$$IOU(n) = \frac{\mathrm{TP(n)}}{\mathrm{TP(n) + FP(n) + FN(n)}}$$

$$TOTP(n) = P(n) + TOTP(n-1)$$

$$TIOU(n) = IOU(n) + TIOU(n-1)$$

And thus;

$$Average\ Precision(n) = \frac{TOTP(\mathrm{n})}{\mathrm{n}}$$

$$Mean\ IOU(n) = \frac{\mathrm{TIOU(n)}}{\mathrm{n}}$$

## 6 Results and Discussion

### 6.1 Single Layer Transforms

The research related to the first research question is discussed both in this section and the following section. The first question was whether the use of pre-processed engineered features as inputs to neural networks has a positive influence on the accuracy of the said neural network, when segmenting tree crops, over the base (raw image band) neural network band.

As per the experimental design, the base truth was first established by training and evaluating the FCN on the single-layer RGB images. After training it for 5000 epochs, it yielded an average precision of 92.11% and a mean IOU of 78.32%. As can be seen in Figure 3, the FCN made an inaccurate first prediction, but after 5000 epochs the prediction was quite accurate.

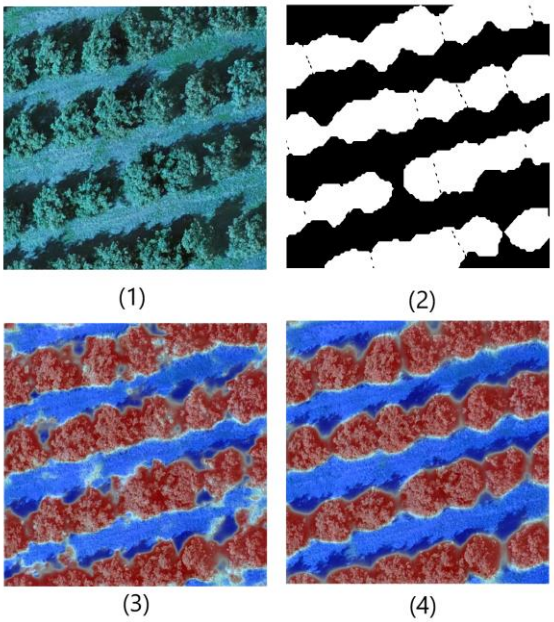


**Figure 3: Image depicting the RGB input image (1), the mask of the image (2), the first prediction made (3) and the last prediction made (4)**

After the base truth was established, the feature extracted inputs were trained and evaluated on the FCN. All the outputs from these tests can be seen in Table 1. The mean IOUs from these tests were mostly less than the ground truth's, hovering around the mid 77%s and low 78%s. On average the mean IOU of the eight transforms was 75.54%, which is 2.78% less than the ground truth. Two notable outliers were the Edge Detector and HSI colour space transforms. These two performed exceptionally poor, with the Edge Detector having a mean IOU of 68.52% and HSI a mean IOU of 67.6%. Only one transform performed better than the ground truth test, which was the PCA colour space transform. It had a mean IOU of 79.16%, which is an increase of 0.84%.

The lack of improvement shown by the FCN when trained on the feature extracted inputs could be due to various factors. The first factor could be that on this level of detail, the use of the normal RGB images as inputs teaches the FCN enough tree elements to be able to competently segment tree from non-tree. Another factor could be that none of the feature extractors used was optimised for extracting features from tree data. A final factor to consider was that the masks of the images provided were not as accurate as one would desire it to be. This resulted in the FCN teaching itself tree elements that produced predictions that were more competent than the mask. On comparison of the prediction with the mask, the prediction would be assigned an incorrect low IOU and precision score. This hampers the learning process of the FCN as it discourages the FCN from learning elements that would produce predictions that are more competent than the mask.

Although the first two causes may be true, the tests performed in this experiment does not prove either. This was not, however, the case for the final factor. It was observed that the average precisions of all transforms were much higher than the mean IOUs. As can be seen in the formulas of the two metrics, the only difference between these two metrics is that the IOU also adds the false negatives (FNs) produced by the FCN to the denominator. With the mean IOU being consistently lower than the average precision, it implies that a consistently high FN count is generated. With the high occurrence of FNs, it can be deduced that the masks are not sufficiently precise, meaning it includes a large area of non-tree inside of the area it claims to be tree. This conclusion was supported visually by inspecting the input image versus the mask in Figure 3. The separation of some trees was only denoted by a dotted line and there were instances where clear spaces can be seen between trees on the RGB image, but the mask shows these trees as being connected. This can be seen in Figure 3 in the second row from the top between the first two trees from left in images (1) and (2).

After inspection of all the single-layer mean IOU TensorFlow charts, it was seen that although the mean IOU measured in most of the transform's plateaued after 5000 epochs, the LAB and PCA colour space transforms had not plateaued yet. These two transforms showed increases of 1.81% and 1.63%, respectively in the second 2500 epochs. It was therefore decided to run the two transforms again for 7500 epochs to see whether the rate of change in the mean IOU stabilised between 5000 and 7500 epochs.

The mean IOU for the test ran for 7500 epochs using the PCA transform barely increased from the one that ran for 5000 epochs. The 7500 epochs test only yielded a 0.08% increase in mean IOU, reaching 79.24%. The LAB colour space transforms mean IOU, however, showed a larger increase, ending 1.05% higher after 7500 epochs, reaching 79.94%. By extrapolating the rate of increase in mean IOU for the RGB test measured over the last 2500 epochs it follows that the FCN trained on RGB images could increase at most by a further 0.09% if ran for a further 2500 epochs, up to 7500 epochs, reaching 78.26%. Thus, the LAB and PCA colour space transforms would, at minimum, outperform the base RGB test by 1.68% and 0.98% respectively. These increases were noteworthy because these were the only transforms that did not plateau after 5000 epochs and that improved on the RGB's performance in the single-layer category. This indicates that PCA and LAB transforms are more appropriate for tree data. It would have been preferable to retest all the transforms to ensure that the accuracy and IOU have in fact stabilised, but due to limited time, only the transforms that showed the showed the highest rates of increase were retested. The charts of the two transforms that ran for 7500 epochs can be seen in Figure 4.

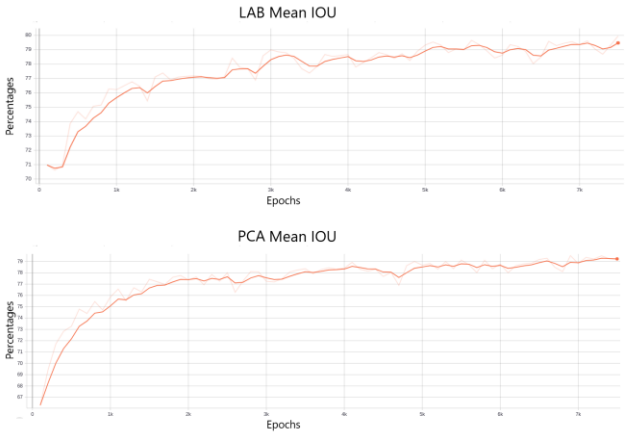


**Figure 4: The Mean IOU charts produced by TensorFlow after 7500 epochs for the single-layer LAB and PCA colour space transforms**

Another observation made, was that barring the two transforms that performed exceptionally poor and the Histogram Equalisation, all transforms slightly outperformed the ground truth's average precision. The best again being the PCA colour space transform, outperforming the ground truth's average precision by 0.8%. It was also seen that when disregarding the two outliers, the mean average precision of all the transforms was 0.13% higher

than the ground truth's average precision. This means that on average when an FCN is given a single layer transform as input, the model makes fewer FP predictions when using feature extracted inputs, compared to when using only RGB images.

| Transforms | Mean IOU | Accuracy | Average Precision |
|---|---|---|---|
| RGB (Ground truth) | 78.32% | 92.00% | 92.11% |
| Mean-Shift | 78.06% | 92.00% | 92.22% |
| Edge Detector | 68.52% | 88.00% | 87.66% |
| Histogram Equalisation | 77.97% | 93.00% | 92.06% |
| LAB | 77.69% | 92.00% | 92.37% |
| HSI | 67.60% | 85.00% | 86.61% |
| HSL | 77.58% | 91.00% | 92.27% |
| PCA | 79.16% | 91.00% | 92.91% |
| ICA | 77.47% | 92.00% | 92.23% |

**Table 1: Metric outputs from tests conducted using single-layer transforms**

Figure 5 depicts the single-layer image that was evaluated on, as well as the model's first and last predictions, for both the worst-performing transform and the best performing transform. For the HSI transform, the FCN produced a poor first prediction, while the PCA transform, produced a decent prediction already.

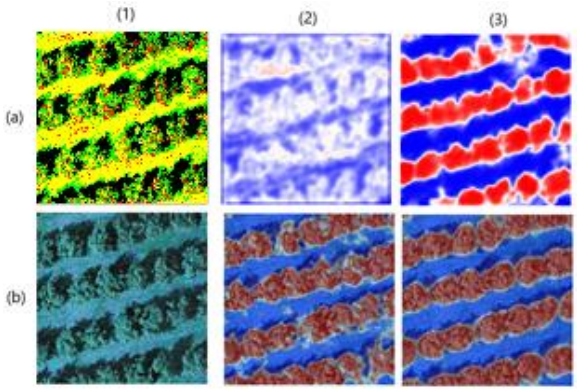


**Figure 5: Image depicting the HSI's input image (a1), first prediction made by the FCN (a2) and last prediction made by the FCN (a3), and the PCA's input image (b1), first prediction made by the FCN (b2) and last prediction made by the FCN (b3)**

## 6.2 Multiple Layer Transforms

For these tests, the same process was followed as was the case for the single-layer transforms. The multiple-layer transforms yielded a ground truth of an average precision of 92.94% and a mean IOU of 79.7%. As can be seen in Figure 6, with the added DEM layer, the FCN made an overly tight first prediction, however, its last prediction was very close to the provided mask.

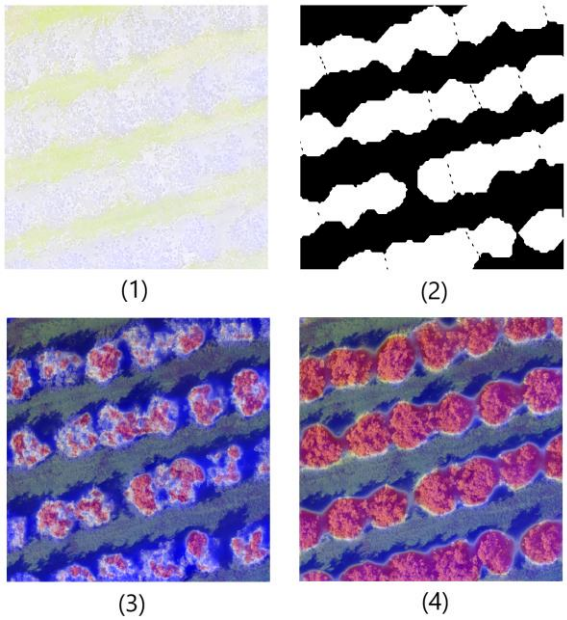


**Figure 6: Image depicting the multiple layer RGB input image (1), the mask of the image (2), the first prediction made (3) and the last prediction (4)**

For these tests, all outputs can be seen in Table 2. The mean IOUs again did not fluctuate much from the ground truth. On average the mean IOU of the eight transforms was 78.32%, which is 1.38% less than the ground truth. The worst performing transforms in this round of testing was the HSI and HSL transforms. When disregarding these two outliers, the average mean IOU of the six transforms was only 0.04% less than the ground truth. There were three transforms that performed better than the ground truth and these were the Mean-Shift, Histogram Equalisation and ICA transforms. The highest of the three, ICA, had a mean IOU that was 1.75% higher than that of the ground truth. Although the transforms in this section performed much better than the transforms in the previous section. Generally, there was still a lack of improvement over the ground truth. The reasons for this lack of improvement could be due to the same factors as was discussed in the single-layer transforms section.

After inspection of all the multiple-layer mean IOU TensorFlow charts, it was seen that four of the mean IOU values had not plateaued after 5000 epochs. Of these four, only the two that showed the highest rates of increase during the last 2500 epochs were retested, also up to 7500 epochs. The two retested inputs were RGB, which had a 2.47% increase in mean IOU during the second 2500 epochs, and PCA, which had a 2.17% increase in mean IOU during the second 2500 epochs.

The PCA's mean IOU at 7500 epochs did not increase much from that measured in the previous test that ran only up to 5000 epochs. It only increased by 0.27%, reaching 79.62%. In the case of the RGB test, the mean IOU also did not increase much, ending 0.19% higher at 7500 epochs, reaching 79.89%. The charts of the two transforms that ran

for 7500 epochs are shown in Figure 7. From this, it was deduced that extending the tests to 7500 epochs does not make a significant difference in either mean IOU or average accuracy for multiple-layer transforms.

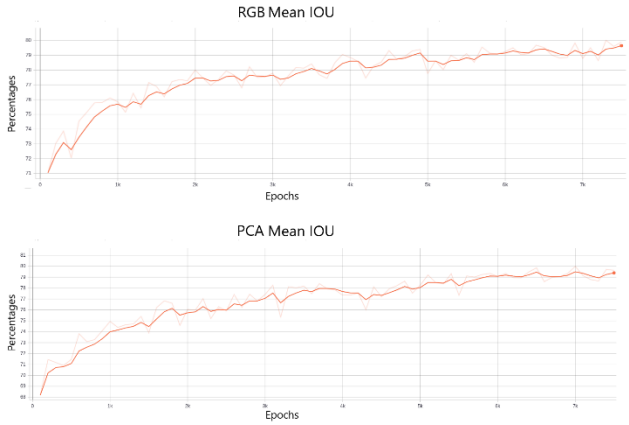


**Figure 7: The Mean IOU charts produced by TensorFlow after 7500 epochs for the multiple-layer RGB images and PCA colour space transform**

For average precision, the two worst tests in this metric were still HSI and HSL, and three transforms outperformed the ground truth, which was Mean-Shift, Histogram Equalisation and ICA. When disregarding the two outliers again, the mean average precision of the transforms was 0.17% less than the ground truth's average precision. This leads to the deduction, that on average when an FCN is given a multiple-layer transform as input, the model makes slightly more false-positive predictions when using feature extracted inputs.

| Transforms | Mean IOU | Accuracy | Average Precision |
|---|---|---|---|
| RGB (Ground truth) | 79.70% | 92.00% | 92.94% |
| Mean-Shift | 80.30% | 92.00% | 93.06% |
| Edge Detector | 79.16% | 93.00% | 92.45% |
| Histogram Equalisation | 79.87% | 91.00% | 92.84% |
| LAB | 77.69% | 92.00% | 92.37% |
| HSI | 74.60% | 91.00% | 91.11% |
| HSL | 74.15% | 91.00% | 91.06% |
| PCA | 79.35% | 92.00% | 92.59% |
| ICA | 81.45% | 93.00% | 93.29% |

**Table 2: Metric outputs from tests conducted using multiple-layer transforms**

## 6.3 Single – vs Multiple Layer

The research related to the second research question is discussed in this section. The question being is whether the segmentation accuracy of certain engineered features was positively influenced by the number of layers added to it.

When comparing the results from Table 1 and 2, the single-layer transforms had an average mean IOU of 75.51%, while the multiple-layer transforms had an average mean IOU of 78.32%, which is an improvement of 2.81%. The average precision of the multiple-layer inputs also outperformed the single-layer input by 1.31%.

The difference can also be seen when comparing Figure 3 and Figure 6. The initial prediction in Figure 3 is quite loose and over predicts the trees, while the initial prediction in Figure 6 is overly tight and underpredicted the trees. The final prediction of both are very competent, however, when examined a bit closer, Figure 6's final prediction picked up some of the gaps between the trees, while Figure 3's final prediction was not as well adapted to picking up these gaps.

Thus, it can be deduced that when adding additional layers, the new tree elements the FCN learns from the new layer is more valuable than what it learns from the feature extracted inputs alone, as the ground truth and transforms performed very similarly.

# 7 Conclusions

In conclusion, the use of certain engineered features did increase the accuracy of the FCN when segmenting tree crops, over the base (raw image band) neural network band, although very slightly. Generally, the impacts were not always positive, however, of the per-pixels transforms the LAB, PCA and ICA colour space transforms performed the best on the data and of the small-width feature extractors it was the Mean Shift and Histogram Equalization transforms that performed the best on the data.

The high average precision and low mean IOU pointed to the fact that the predictions produced a high number of false-negative predictions. This was caused by the image masks that were not a good enough representation of the actual trees. The relatively poor quality of the masks was a detriment to the project, as the FCN cannot be expected to perform well when the masks training it are not accurate representations of the real-life data. It will be beneficial to ensure that a rigorous mask extractor is created.

From the tests performed with more epochs, it could be seen that the FCNs trained on the single-layer images did not all plateau after the 5000 epochs. This leads to the conclusion that for single-layer inputs, the FCN needs to be trained longer on feature extractors and that the feature extractors will over time overtake the RGB's segmentation performance. For multiple-layer inputs, only insignificant improvements were observed after 5000 epochs.

When comparing the single and multiple layer inputs with each other, it was observed that using inputs that utilise multiple layers does improve the segmentation performance of the FCN, over just using single layer transforms. Thus, it can be concluded that the segmentation accuracy of certain

engineered features is indeed influenced by the number of layers added to it.

## 8 Limitations

There were limitations experienced throughout the course of the project. The first and biggest limitation is the time allocated to the project. This project was allocated a six-week time span, which greatly limited the scope of experimentation and the ability to solve other issues faced throughout the course of the project. This, regrettably, lead to the team having to decrease the scope of this project, as well as the amount of testing conducted.

The next limitation that was faced was the implementation of the Edge Detector. The OpenCV and two different GitHub repositories were considered, but all three implementations noticed too much detail, causing the image to be a mess of edges (trees, leaves and ground). It seemed as if these implementations were not appropriately blurring the image. All three's parameters were tweaked to try and improve the blurring, to no avail. In the end, no more time could be spent attempting to fix the Edge Detector. The project team, therefore, settled on the OpenCV implementation of the Edge Detector, because it could differentiate between tree and ground despite it still being a mess of edges, leading to the mess of edges to be contained within the area that was tree data.

The last limitation experienced was that the FCN was built to read images in the JPG or PNG formats. The team's initial plan was to read in numpy arrays, to be able to pack together as many layers and combinations as the project team desired. However, in this FCN's case, it was not a possibility. An attempt was made to change the way that images were read, to allow the FCN to rather use numpy arrays. Initially, it seemed to be an easy task, as the image files were transformed into numpy arrays. It was however found that that the code was hardcoded to work specifically with numpy arrays created from a four-channel image. If the numpy array does not have that exact number of channels, the FCN would crash when reading the data. The project team believed this issue could be solved, however, the project team did not have the time to analyse and rewrite the part of the KittiSeg project code that read and prepared the input files, as well as overlaid the predictions onto the input image.

## 9 Ethical, Professional and Legal Issues

There were no ethical issues experienced during this project as the project team did not; do research on humans or animals, use personal data, or generally interact with people outside of the project team. There also was no dispute over the ownership of the intellectual property, as the engineered features were not novel ones provided by Aerobotics and the intellectual property of the platform the project team designed to test the hypothesis on belong to the team and the supervisors.

Software licensed under GPL-3.0 was extended (MarvinTeichmann' KittiSeg [14] & fjean's pymeanshift [5]). This means the sections handling mean-shift transformations, as well as the FCN implementation were released under the same licence.

## 10 Future Work

As stated earlier, there were a few limitations to this project that could be solved in future projects. With these limitations solved, future projects could explore using more feature extractors. Future projects could also go into deeper detail on how much the different types of layers improve the segmentation performance of the FCN by their addition or subtraction.

Another next step for this project is to experiment with combinations of feature extractors and how these combinations impact the segmentation of tree data by FCNs. Thereby testing to see if transform combinations might be able to improve segmentation performance and possibly finding a gold standard of transforms that can be applied to images containing tree data, that greatly improve the segmentation performance of CNNs.

## ACKNOWLEDGEMENTS

I would like to thank my project partners, Michael Scott and Fergus Strangways-Dixon, for their support and diligence throughout the course of this project, as well as Michael Malahe, from Aerobotics, who provided us with the datasets needed to train and evaluate the CNN models on. I would also like to thank Associate Professor Patrick Marais and Associate Professor Deshen Moodley for their advice and guidance throughout the course of this project.

## REFERENCES

[1] Abdou, I. and Pratt, W., 1979. Quantitative design and evaluation of enhancement/thresholding edge detectors. Proceedings of the IEEE, 67(5), pp.753-763. DOI:https://ieeexplore-ieee-org.ezproxy.uct.ac.za/stamp/stamp.jsp?tp=&arnumber=1455594

[2] Finnis, J. 2018. Random Forest Classification of Tree Crops on Farming Land., pp.1-13. DOI:http://projects.cs.uct.ac.za/honsproj/cgibin/view/2018/finnis_motsumi.zip/deliverables/Final%20Paper%2 0(Final).pdf

[3] Guo, Y., Liu, Y., Georgiou, T. and Lew, M. 2017. A review of semantic segmentation using deep neural networks. International Journal of Multimedia Information Retrieval, 7(2), pp.87-93. DOI:https://link.springer.com/content/pdf/10.1007%2Fs13735- 017-0141-z.pdf

[4] Heath, M., Sarkar, S., Sanocki, T. and Bowyer, K., 1998. Comparison of Edge Detectors. Computer Vision and Image Understanding, 69(1), pp.38-54. DOI:https://www.sciencedirect.com/science/article/pii/S1077314297905877

[5] Jean, F., 2019. fjean/pymeanshift. GitHub. DOI:https://github.com/fjean/pymeanshift

[6] Krizhevsky, A., Sutskever, I. and Hinton, G., 2017. ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), pp.84 -90. DOI:https://papers.nips.cc/paper/4824 -imagenet -classification - with -deep -convolutional -neural -networks.pdf

[7] Long, J., Shelhammer, E. and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. The IEEE Conference on

Computer Vision and Pattern Recognition (CPVR) (2015), 3431-3440. DOI:https://doi.org/10.1109/cvpr.2015.7298965

[8] Motsumi, N. 2016.: III -CNN: Image -to -Image Inception CNN for Pixel -Wise Segmentation to extract tree and tree boundaries. , pp.1 - 12. DOI:http://projects.cs.uct.ac.za/honsproj/cgi - bin/view/2018/finnis_motsumi.zip/deliverables/mstnto015Final.pdf

[9] Noh, H., Hong, S. and Han, B. 2015. Learning Deconvolution Network for Semantic Segmentation. 2015 IEEE International Conference on Computer Vision (ICCV)(2015). DOI:http://dx.doi.org/10.1109/iccv.2015.178

[10] Paris, S. and Durand, F., 2007. A Topological Approach to Hierarchical Segmentation using Mean Shift. 2007 IEEE Conference on Computer Vision and Pattern Recognition(2007). DOI:http://dx.doi.org/10.1109/cvpr.2007.383228

[11] Ronneberger, O., Fischer, P. and Brox, T. 2015. U-Net Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015 9351 (2015), 234–241. DOI:http://dx.doi.org/10.1007/978-3-319-24574- 4_28

[12] Ryan, S., Cross, P., Winnie, J., Hay, C., Bowers, J. and Getz, W., 2012. The utility of normalized difference vegetation index for predicting African buffalo forage quality. The Journal of Wildlife Management, 76(7), pp.1499-1508. DOI: https://www-jstor-org.ezproxy.uct.ac.za/stable/pdf/23251450.pdf?refreqid=excelsior%3Afab80063884b7e94b98b276bbbf1bb4d

[13] Tao, W., Jin, H. and Zhang, Y. 2007. Color Image Segmentation Based on Mean Shift and Normalized Cuts. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 37(5), pp.1382-1389. DOI:https://ieeexploreieeeorg.ezproxy.uct.ac.za/stamp/stamp.jsp?tp=&arnumber=4305291

[14] Teichmann, M., 2019. MarvinTeichmann/KittiSeg. GitHub. DOI:https://github.com/MarvinTeichmann/KittiSeg

[15] Teichmann, M., Weber, M., Zöllner, M., Cipolla, R. and Urtasun, R. 2018. MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving. IEEE Intelligent Vehicles Symposium (IV), 2018, pp. 1013-1020. DOI:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8500504&isnumber=8500355

[16] Tellaeche, A., Pajares, G., Burgos-Artizzu, X. and Ribeiro, A. 2011. A computer vision approach for weeds identification through Support Vector Machines. Applied Soft Computing 11, 1 (2011), 908–915. DOI: http://dx.doi.org/10.1016/j.asoc.2010.01.011

[17] Wang, J., Thiesson, B., Xu, Y. and Cohen, M., 2004. Image and Video Segmentation by Anisotropic Kernel Mean Shift. Lecture Notes in Computer Science Computer Vision- ECCV 2004(2004), 238–249. DOI:http://dx.doi.org/10.1007/978-3-540-24671-8_19

[18] Wang, X., Huang, X. and Fu, H. 2010. A Colour Texture Segmentation Method to Extract Tree Image in Complex Scene. 2010 International Conference on Machine Vision and Human machine Interface (2010). DOI:http://dx.doi.org/10.1109/mvhi.2010.138

[19] Wang, X., Huang, X. and Fu, H. 2009. The Study of Colour Tree Image Segmentation. 2009 Second International Workshop on Computer Science and Engineering (2009), 303 307. DOI:http://dx.doi.org/10.1109/wcse.2009.818

[20] Xiaosong Wang, Xinyuan Huang, and Hui Fu. 2010. A Colour-Texture Segmentation Method to Extract Tree Image in Complex Scene. 2010 International Conference on Machine Vision and Human-machine Interface (2010). DOI:http://dx.doi.org/10.1109/mvhi.2010.138

[21] Yaman, A., Yilmaz, H. and Bilgilioglu, S., 2019. Comparison of Point Accuracies on Digital Elevation Model Obtained from Digital Air Photographs with Different Specifications. International Journal of Environment and Geoinformatics, 6(1), pp.131-134. DOI:https://dergipark.org.tr/download/article-file/695121

[22] Zhang, Y. 2006. An Overview of Image and Video Segmentation in the Last 40 Years. Advances in Image and Video Segmentation (2006), 1–16. DOI:http://dx.doi.org/10.4018/978-1-59140-753-9.ch001

[23] Zheng, L., Zhang, J. and Wang, Q. 2009. Mean-shift-based color segmentation of images containing green vegetation. Computers and Electronics in Agriculture, 65(1), pp.93-98. DOI:https://www.sciencedirect.com/science/article/pii/S01681699080182 8