Computers in Industry 98 (2018) 183–191

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

Real-time object detection in agricultural/remote environments using the multiple-expert colour feature extreme learning machine (MEC-ELM)

Edmund J. Sadgrove*, Greg Falzon, David Miron, David W. Lamb

Precision Agriculture Research Group, University of New England, Armidale, NSW 2351, Australia

ARTICLE INFO

Article history:
Received 31 August 2017
Received in revised form 6 March 2018
Accepted 15 March 2018
Available online 22 March 2018

Keywords:
Extreme learning machine
Object detection
Machine vision
Unmanned aerial vehicle
Agriculture
Robotics

ABSTRACT

It is necessary for autonomous robotics in agriculture to provide real time feedback, but due to a diverse array of objects and lack of landscape uniformity this objective is inherently complex. The current study presents two implementations of the multiple-expert colour feature extreme learning machine (MEC-ELM). The MEC-ELM is a cascading algorithm that has been implemented along side a summed area table (SAT) for fast feature extraction and object classification, for a fully functioning object detection algorithm. The MEC-ELM is an implementation of the colour feature extreme learning machine (CF-ELM), which is an extreme learning machine (ELM) with a partially connected hidden layer; taking three colour bands as inputs. The colour implementation used with the SAT enable the MEC-ELM to find and classify objects quickly, with 84% precision and 91% recall in weed detection in the Y'UV colour space and in 0.5 s per frame. The colour implementation is however limited to low resolution images and for this reason a colour level co-occurrence matrix (CLCM) variant of the MEC-ELM is proposed. This variant uses the SAT to produce a CLCM and texture analyses, with texture values processed as an input to the MEC-ELM. This enabled the MEC-ELM to achieve 78–85% precision and 81–93% recall in cattle, weed and quad bike detection and in times between 1 and 2 s per frame. Both implementations were benchmarked on a standard i7 mobile processor. Thus the results presented in this paper demonstrated that the MEC-ELM with SAT grid and CLCM makes an ideal candidate for fast object detection in complex and/or agricultural landscapes.

1. Introduction

Agriculture systems require autonomous robotics for weed spraying, livestock detection and vehicle safety. The ability to detect and process objects quickly is a desire of many of these systems. Agricultural scenarios can be exceedingly complex as compared to other industrial based robotics systems. Object detection algorithms in agriculture may compete with any number of structurally similar or diverse objects. This makes for a complex environment with the potential for many false positives and false negatives. A potential solution to this problem is to adopt a cascading or multiple expert approach. These types of solutions have varying levels of success in both flora and fauna detection and there are numerous implementations, ranging from substratal to more complex. These approaches include simple colour and texture detection, as well as broad template matching and have been implemented for weed, horse and wildlife detection [1–3]. The more complex solutions use a wide range of techniques, which can include large data structures such as deep or exemplar neural networks and varying levels of texture and shape based analyses. This includes for the ripeness of bananas and for other generic object detectors [4–7].

The approaches discussed often rely on grey-scale images and in many cases, the detection of prominent features or stand out colour attributes. Processing speed is the key advantage in this case. Deep learning architectures are much slower and more complex and for this reason are often avoided in place of real time solutions. Notably, with a high-end GPU it is possible to process a large number of frames per second [8], but in remote mobile computing GPUs might not be attainable. A disadvantage of many approaches is the reliance on prominent key characteristics in the classification process. This often leads to poor overall classification accuracy, particularly in more complex scenarios, e.g., in weed detection there may not be any prominent features; in the case of cattle detection there may be variations in colour between the

* Corresponding author.

https://doi.org/10.1016/j.compind.2018.03.014
0166-3615/

same breed and in other cases an object's features may appear different at different angles or rotations.

The goal of this research is then to explore methods that can be used to deliver both fast and accurate feature extraction and object classification. For this, an implementation of the Multiple-expert colour feature extreme learning machine (MEC-ELM) [9] is proposed. The MEC-ELM is a cascading implementation of the Colour Feature Extreme Learn Machine (CF-ELM) [10], which is itself an implementation of the Extreme Learning Machine (ELM) [11] for colour object detection. The ELM in part due to its efficient implementation and fast processing speeds has demonstrated suitability for computer vision based problems in the agricultural scenarios. Including for soybean classifications [12], unmanned aerial vision for palm tree detection [13] and has been benchmarked using notable feature extraction techniques [7]. The MEC-ELM can be used as both a feature extraction and classification technique. This can be achieved by adopting the summed area table (SAT) [14] (or integral image) and thereby reducing a landscape image (or video frame) to a grid of coloured blocks. The purpose of this is to provide a generic approach to HAAR features [15] and hence take advantage of the SATs fast, multi-scale feature extraction architecture. Fast processing may require low resolution image data and this can result in a potential loss of pixel based information. To meet this challenge, the output of the SAT grid is used to generate three colour level co-occurrence matrices (CLCM) [16], one for each colour band (red, green and blue). The outcome will be a texture based analysis, with the values provided as input of each CF-ELM. The two implementations of the MEC-ELM were implemented and processing speeds and overall accuracy compared. The algorithm is designed with the objective of fast object detection in complex and unpredictable terrain, making it an ideal candidate for use in the agriculture industry. The objective in this case is to implement the MEC-ELM as a weed detector for spraying, unobtrusive cattle tracking and as a vehicle interaction and avoidance tool. This paper will benchmark the MEC-ELM with pre-recorded video data, for eventual use in a remote laptop interface for interaction with an unmanned aerial vehicle (UAV or drone) and stationary surveillance devices.

2. Theory/calculation

2.1. Multiple-expert colour extreme learning machine (MEC-ELM)

The ELM is a single layer, feed forward neural network that is known for its fast and analytical training phase. In this phase the output of the neural networks hidden layer is stored in a matrix designated **H** the output weights are then determined analytically. This can be expressed [17,18]:

$$\mathbf{H}(\mathbf{W}_1\cdots,\mathbf{W}_{\tilde{N}},b_1\cdots,b_{\tilde{N}},\mathbf{x}_1\cdots,\mathbf{x}_N) = \begin{bmatrix} g(\mathbf{W}_1\cdot\mathbf{x}_1+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{x}_1+b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{W}_1\cdot\mathbf{x}_N+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{x}_N+b_{\tilde{N}}) \end{bmatrix} N\cdot\tilde{N} \tag{1}$$

where **H** is the hidden layer output matrix, N is the number of samples used in the training phase and $\tilde{N}$ is the number of neurons in the hidden layer. In the activation function $g()$ [19], **W** is the input weight, **x** is the input sample pixels and b is the bias.

The Colour Feature Extreme Learning Machine (CF-ELM) is similar in architecture to the ELM, comprising of a single layer, feed forward, neural network with a partially connected hidden layer and a fully connected output layer. The hidden layer is divided into 3 sections and this gives the CF-ELM the ability to be used with different colour models, including red, green, blue (RGB), luminance, chrominance red, chrominance blue (Y'UV) and hue, saturation, value (HSV) [20], Y'UV (also known as YCrCb) is defined by the international telecommunications union as ITU-R B.601 [32]. Equivalent to the standard ELM it uses randomly assigned weights in the hidden layer and by using the pseudo inverse it can analytically determine the output weights from a fully connected output layer. By dividing the hidden layer into 3 sections, each colour attribute can be processed in a separate section of the hidden layer and it is for this reason that the number of neurons in the hidden layer must be a multiple of 3. By storing the output of these 3 sections into 3 sections of the **H** matrix it allows the matrix to be used to determined the output weights in the same way as the standard ELM. For Y'UV the CF-ELM hidden layer can be expressed [10].

$$\mathbf{H}(\mathbf{W}_1\cdots,\mathbf{W}_{\tilde{N}},b_1\cdots,b_{\tilde{N}},\mathbf{Y'}_1\cdots,\mathbf{Y'}_N,\mathbf{U}_1\cdots,\mathbf{U}_N,\mathbf{V}_1\cdots,\mathbf{V}_N) = \begin{bmatrix} g(\mathbf{W}_1\cdot\mathbf{Y'}_1+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{Y'}_1+b_{\tilde{n}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{W}_1\cdot\mathbf{Y'}_N+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{Y'}_N+b_{\tilde{n}}) \\ g(\mathbf{W}_1\cdot\mathbf{U}_1+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{U}_1+b_{\tilde{n}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{W}_1\cdot\mathbf{U}_N+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{U}_N+b_{\tilde{n}}) \\ g(\mathbf{W}_1\cdot\mathbf{V}_1+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{V}_1+b_{\tilde{n}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{W}_1\cdot\mathbf{V}_N+b_1) & \cdots & g(\mathbf{W}_{\tilde{N}}\cdot\mathbf{V}_N+b_{\tilde{n}}) \end{bmatrix} 3N\cdot\tilde{N} \tag{2}$$

where **Y'**, **U** and **V** are equal to the individual colour pixel matrices for each image and are stored in the **H** matrix at the output of the hidden layer. The hidden layer process is repeated in the output layer, with the output β becoming the input multiplier for the output weights β. The output **T** of the CF-ELM is then the result of $\beta\cdot\mathbf{H}$.

$$\mathbf{T} = \beta\cdot\mathbf{H} \tag{3}$$

Here β can be expressed:

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix} \tilde{N}\cdot m \tag{4}$$

where m is the number of neurons in the output layer, which is equivalent to the number of outputs of the ANN. The matrix of target outputs **T** can be expressed as:

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix} N\cdot m \tag{5}$$

where for each $\mathbf{T}_N$ the value is stored based on the input training sample and its desired output. This leaves β as the one unknown, by making β the subject we get:

$$\beta = \mathbf{H}^{-1}\cdot\mathbf{T} \tag{6}$$

where $\mathbf{H}^{-1}$ is the Moore-Penrose pseudo inverse of matrix **H**. The output values of this process are then stored in β and used as the weights in the output layer removing the need for a long gradient descent based training process.

The MEC-ELM is then a set of CF-ELMs, where each CF-ELM can be trained on a different set of sample images, different colour system or image analysis techniques. The MEC-ELM becomes a global consensus of all CF-ELMs or experts, the goal is then to find individual CF-ELMS of high classification accuracy but with varying consensus [21]. The method used in this paper was inspired by the Examplar SVM [22] and the Ensemble ELM (EN-ELM) [23], where training samples are divided among different instances of CF-ELMs. The training phase of the MEC-ELM is depicted in Fig. 1,

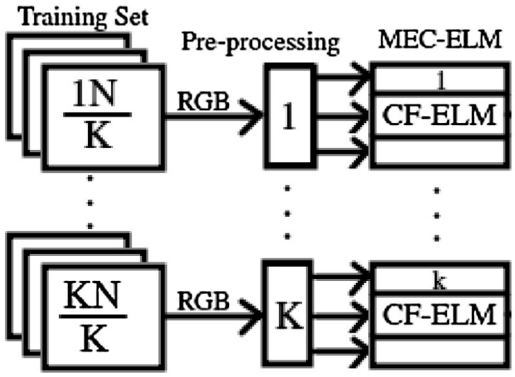

Fig. 1. The MEC-ELM with each CF-ELM trained on a different portion of the training set.

where K is the number of CF-ELMs, each instance is trained on one kth of the training set and $\frac{1N}{K}$ to $\frac{KN}{K}$ (or N) denotes the final image for each instance. A predetermined number of values is sent to the input of each CF-ELM, where RGB values are first processed, this will be discussed further in the methodology in Section 3.1.

2.2. Summed area table (SAT)

The SAT, which is also known as the integral image (II), was made popular by Viola and Jones object detection framework [24]. The SAT allows fast object detection by converting an image into a summed representation of all pixel values and from these values a HAAR based feature set can be used for broad template matching. The HAAR based feature set can be quite large and the AdaBoost algorithm [25] is often used to find features most commonly associated with a set of training images. The HAAR features are rectangular based features with alternating dark and light areas designed to match the dark and light areas (or light intensity) associated with a target object. These blocks match the block based summation values found in the SAT, which means that these dark and light areas can be matched to areas within the image very quickly. Only four sets of coordinates are required to calculate the light intensity of one sub block. To create the SAT all pixel values in the table are stored as summations of the pixel value at the current location and the preceding summation values. In this fashion, the SAT can be generated from just one pass over the image. This can be expressed [14]:

$$II(i,j) = (i,j) + (i - \Delta i, j) + (i, j - \Delta j) - (i - \Delta i, j - \Delta j) \tag{7}$$

where $II(i, j)$ is a single coordinate of dimensions i and j of the SAT and the decrements (Δi, Δj) refer to the preceding pixel summation value; here Δi, $\Delta j = 1$. The sum light intensity value of any rectangular area can then be calculated using the four coordinates of each corner of the rectangle. This can be expressed:

$$S(i, j, i - w, j - h) = (i,j) + (i - w, j - h) - (i - w, j) - (i, j - h) \tag{8}$$

where S refers to the sum of the rectangle's pixel values, with bottom right coordinates i and j and dimensions w (width) and h (height). The resulting magnitudes are then normalised by divide the magnitude by the number of pixels within the rectangle. This will produce an average light intensity value between 0 and 255. In this paper red, green and blue SATs were created instead of one SAT for grey-scale (or light intensity). This allowed the CF-ELM to work with each individual colour band as required.

2.3. Colour-level co-occurrence matrix (CLCM)

The CLCM [16] (or grey level GLCM) is an image analysis tool based on light intensity and looks at the relationship between pixels and their nearest neighbour [26]. The CLCM has 3 or more possible variants, including first order, second order and third order. Each order describes a tabulation of different combinations of pixels. The first order is essentially a one dimensional histogram of pixel intensities, it does not look at the relationship between neighbouring pixels and can be used to determine statistics such as mean and variance [27]. Second order CLCMs look at the relationship between a pixel and one of its neighbours, creating a two dimensional array (or matrix) of combination occurrences. The second order can be used for texture analyses and will be utilised in this paper. Third or higher order CLCMs look at 2 or more of a pixel's nearest neighbours. These implementations are computationally more expensive and more difficult to interpret, for this reason they were not used in this research. The initialisation of a second order co-occurrence matrix can be expressed [28]:

$$P(i,j) = \sum_{x=1}^{n}\sum_{y=1}^{n} \begin{cases} 1, & \text{if } I(x,y) = i \text{ and } I(x + \Delta x, y + \Delta y) = j \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where $P(i, j)$ represents total events and is an individual element in the CLCM, n by n is the dimensions of a rectangular image, x and y are coordinates within an image and Δx and Δy are the distance from a pixel to its neighbour in the right direction (these same combinations are checked in the left direction as well), in this case Δx, $\Delta y = 1$. For this research the intensity levels were reduced from 256 to 16 during initialisation of the CLCM. This was done to reduce the occurrence of nil values. After initialisation values are normalised by dividing all values in the CLCM by all possible combinations in the CLCM. Total potential combinations can be calculated from known dimensions $2 \times n \times n$, where 2 represents the left and right directions. Texture analyses values can then be calculated from the values in the CLCM matrix. Five statistics were chosen; these values were selected as they were the least computationally intensive and provided good results in preliminary testing. These included energy, entropy, contrast, homogeneity and CLCM mean, which can be expressed [28,26]:

$$\text{Energy} = \sum_{i,j=0}^{D-1} P(i,j)^2, \tag{10}$$

$$\text{Entropy} = \sum_{i,j=0}^{D-1} -\ln(P(i,j)) \times P(i,j), \tag{11}$$

$$\text{Contrast} = \sum_{i,j=0}^{D-1} P(i,j) \times (i-j)^2, \tag{12}$$

$$\text{Homogeneity} = \sum_{i,j=0}^{D-1} \frac{P(i,j)}{1 + (i-j)^2}, \tag{13}$$

$$\text{CLCM Mean} = \sum_{i,j=0}^{D-1} i \times P(i,j) \tag{14}$$

where D is the number of values within the CLCM. In this research three CLCMs were created to match the three colour band requirements of the CF-ELM. These included a red level, green level and blue level co-occurrence matrix. To avoid confusion these will be referred to as a colour-level co-occurrence matrix (CLCM), but it is noteworthy that a grey-level (GLCM) is more common in literature.

3. Methodology

All instances of the MEC-ELM were programmed and tested using the *C* programming language, *C* was chosen due to portability and faster processing speeds. It is also possible to use variants of C in many embedded devices [29]. All benchmarking was conducted using a laptop computer with a Linux based system, 16 gigabytes of ram, a solid state drive and a 4th generation *i*7 mobile processor. All time benchmarking was conducted using the *clock_g ettime* function with the *CLOCK_M ONOTONIC* option from the *time. h* library. All images were stored as *JPEG* and decompressed using the *jpegIO. h* library (also known as *libjepg*), before pre-processing, images were stored in memory as RGB values between 0 and 255. *JPEG* was chosen for storage and file transfer reasons, it is also an output available in a number of remote camera interfaces. Images were stored using 4:4:4 sub-sampling which is perceptually lossless and were saved using the default setting 4:2:0 (lossy) in *libjpeg* for user feedback/external processing (smaller file). In agriculture this means that the lesser quality images are better prepared for the potential wireless transfer constraints of remote environments.

3.1. The algorithm

The algorithm is a multiple stage process based primarily on Sadgrove et al. [9,10] for the initialisation of the CF-ELM for remote computer based classification. These stages included an (i) initialisation stage: where the neural network is initialised into memory with initial random weights, (ii) weight biasing stage: where the weights were biased to the training data and is based on the CIW-ELM [30], (iii) training stage: where the output weights were analytically determined with assistance from the C lapack library [31], (iv) a tuning stage: where optimal threshold values were found based on a tuning set of 100 images, this stage was also used to assist in an off line training process, where optimal individual weight sets for each CF-ELM were saved in text based data files so that they can be imported instead of retraining, (V) testing stage: where the individual CF-ELMS were tested on the image frames. The weight biasing, training, tuning and testing stages differed for each of the two implementations and from the standard CF-ELM algorithm. These differences will be discussed. Of the two implementations of the MEC-ELM tested in this paper the primary difference is that one utilises the CLCM and the other does not. This means that the CLCM MEC-ELM has CF-ELMs with just 5 inputs to match the 5 texture values. The Y'UV MEC-ELM has inputs matching the number of grid blocks from the integral image. In both cases the grid blocks will be referred to as the SAT grid. In this implementation the RGB image is converted to Y'UV before being processed as a sat grid.

- *Frame Extraction*: prior to testing, each video described in Section 3.2 was extracted into individual frames. The frames were extracted from *MPEG* video format to *JPEG* using the *FFmpeg* package available to bash in Linux. The frames were extracted at two frame per second to match the expected processing time of the algorithm, although this did not take into account the CLCM and tracking algorithms. The video used for testing in the case of the cattle aerial footage was only 18 s long, for this reason six frames per second were extracted instead. To elaborate, the extraction of two frames per second was deemed adequate for object detection in remote environments, but the emphasis in this paper was on evaluating the effectiveness of the algorithm and for this reason more frames were required.
- *The SAT Grid*: during the training, testing and tuning processes, all images were first converted to a summed area table. The SAT was then used to extract an average colour grid of the image or target area. There was no limit on the number of coloured blocks within the grid, but in this research between 25 and 100 blocks were used and this depended on how successful classification was at each block number. As processing speed was a priority, lower block numbers were preferred. For the Y'UV variant of the MEC-ELM these values were sent directly to each CF-ELM for processing. This is on display in Fig. 2, where 3SAT is the three individual summed area tables for the colour bands Y', U and V. There is also an example of an image converted to a SAT grid with 196 blocks in Fig. 3.
- *Scaling*: Each frame was read as a selection of rectangles. The rectangle size was produced by dividing the width and height of the image frame by a number determined to be effective in pretesting. At each iteration the search rectangle was moved through the image frame (typically by 10–20 pixels at a time). To search with a larger or smaller rectangle the number for dividing was increased or decreased and this caused a scale change.
- *The CLCM*: The block RGB values extracted from the SAT were processed into a CLCM for use with the CLCM variant of the MEC-ELM. During this process the red, green and blue level co-occurrence matrices were produced and from each an energy, entropy, contrast, homogeneity and CLCM mean value were produced. The five values were then sent to the five inputs of each CF-ELM for processing. A diagram is on display in Fig. 4. Note the extra stage 3CLCM, where R, G and B SATs are used to make 3 CLCMs for texture values energy, entropy, contrast, homogeneity and CLCM mean.

Fig. 3. Left image is a SAT grid of the image on the right with 196 blocks.

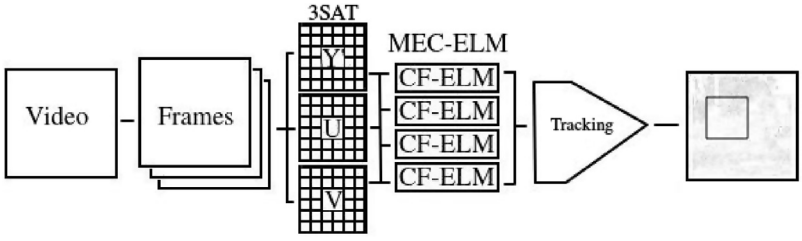

Fig. 2. Diagram of the standard Y'UV variant of the MEC-ELM.

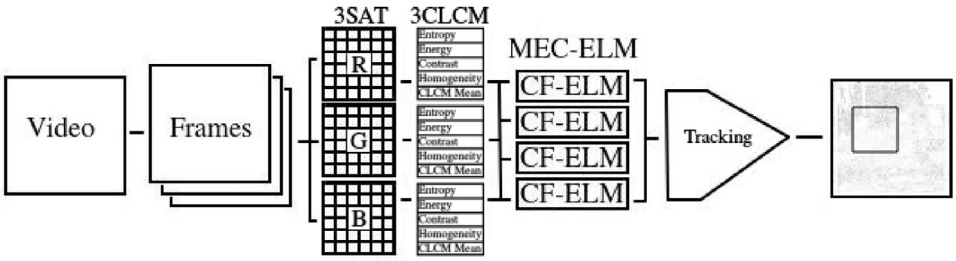

Fig. 4. Diagram of the CLCM variant of the MEC-ELM.

- *Object Detection*: to improve the precision and recall rates of the algorithm, areas of interest were saved every time a tested rectangle within the image frame produced a positive classification. If an area produced multiple classifications at multiple scales and in close proximity, a square was drawn in the area based on saved coordinates. Each classification was stored with an (x, y) coordinate number for the top right hand corner and the bottom left hand corner of the rectangle. A classification was considered in the same area if coordinates for a tested rectangle were within a percentile area of previously saved coordinates. The amount of classifications that triggered a square to be drawn differed for each test set. Typically if 1–10 detections of the target object were found in an area, this was a good indication that something was there. Each search area was within 10–100% radius of the first classified rectangle.
- *Tracking Assist*: to assist tracking objects across multiple frames, a single neuron extreme learning machine (SNELM) was utilised and the output of the SAT was used as it's input. The SNELM was initialised based on just one image and the RGB values were reduced to a single 16 bit colour value [33]. Preliminary testing with grey-level values did not improve results. The training image was be arbitrarily selected from the tuning set. After an object is detected the base SNELM is updated to reflect the detected object. In the proceeding frame the SNELM is then primed to detect the same or similar object. As training is based on a single image, the pseudo inverse is not required, this can be expressed:

$$\beta = \frac{\mathbf{T}}{(65536 \times \mathbf{R} + 256 \times \mathbf{G} + \mathbf{B}) \times C} \tag{15}$$

where R, G and B are the arrays of colour values from the grid extracted from the SAT and C is the number of grid blocks. β is one dimensional and serves as an input weights container, with length equal to C and $\mathbf{T}$ contains only one value, as only one image is used in training. This value is a target and can be set to 1. Processing an object can then be expressed:

$$\text{Output} = \sum_{i=0}^{C-1} \beta(i) \times (65536 \times \mathbf{R} + 256 \times \mathbf{G} + \mathbf{B}) \tag{16}$$

the output is then checked against a predefined threshold (typically between 0.1 and 1).

3.2. Datasets

Four datasets were used in training, tuning and testing. The number of images in each dataset is listed in Table 1. The tuning set contained 50 images of the object from the training set and 50 images of random surrounding terrain from the test frames. All images used in testing and tuning were 100 by 100 pixels in resolution. This resolution was chosen based on a trade off between the higher precision of high resolution images and an improvement in processing time. High resolution images were not considered important due to the SAT grid producing a low resolution version of each image. The resolution of the video frames is based on the resolution of the original pre-recorded video. The four datasets were chosen based on a potential use in the agriculture industry, a particular emphasis was placed on unmanned aerial vehicle (UAV or drone) technology. The datasets included, a weed detection scenario involving *Cirsium vulgare* (or Bull Thistle), two cattle detection scenarios and an all terrain vehicle (ATV or quad bike) scenario. The Bull Thistle scenario was chosen, as thistle competes with other more valuable pasture, which is a potential problem for livestock. Detection of the thistle allows for both weed surveillance and targeted spraying, saving money, pasture and the environment. The two cattle detection scenarios involved cattle recorded by a river bed with a stationary surveillance camera and cattle recorded by a drone hovering over green pasture. The cattle detection scenario displays the algorithm's potential as a cattle tracking and counting algorithm. The quad bike scenario involved a rider driving through a rural country side. Vehicle detection can be used in both accident detection and prevention. The technical aspects of each dataset are listed:

Table 1
Number of images in each dataset, with resolution of video frames.

Dataset	Training	Tuning	Testing	Resolution
Bull Thistle	688	100	120	2000, 1500
Cattle Drone	1331	100	110	1920, 1080
Cattle Stationary	500	100	120	640, 480
ATV	608	100	144	1280, 720

- *Bull Thistle*: this dataset was sourced in two different ways. The training set was photographed using a 10 mega-pixel Fujifilm hand held camera on default settings and at a fixed distance of 2 m to simulate aerial surveillance. They were captured in *JPEG* format at a resolution of 3648 by 2736 before cropping to 100 by 100. The video used to produce the testing frames were captured by a DJI Phantom quadcopter in MP4 for 60 s at 4069 by 2178 pixels, at a distance of 10 m and on default settings. Both were captured from pasture fields on the University of New England SMART Farm (Long 151° 35 min 40 s E, Latitude 30° 26 min 09 s S), in sunny conditions between midday and late afternoon.
- *Cattle Drone*: This data was sourced from van Gemert et al [34] allowing comparison of results of the MEC-ELM and the Exemplar Support Vector Machines (Exemplar-SVM) presented. The video footage was recorded using an AscTec Pelican quadcopter, with a mounted GoPro 3: Black Edition action camera, in overcast conditions and at varying distances (around 30–50 m). The video was recorded in 1920 by 1080 pixels at 60 frames per second. The video selected for testing was 18 s long. The training data was made up of cropped and resized images from the remaining videos in the dataset. To extend the training dataset, images were flipped vertically using a bash command.

- *Cattle Stationary*: the dataset is a collection of surveillance videos taken from a nearby farm using a Scoutguard SG860C camera at 640 by 480 pixels, at 16 frames per second for 60 s and AVI format (before conversion to MP4). The videos included Poll Hereford cattle surrounding a river bed and other areas for grazing. The training set was made up of images cropped from some of these videos. A video not used to generate the training images was used as the test video. Each video was captured at different times of day and in different weather conditions. The distance to the cow depended on how close they got to the camera.
- *ATV*: The video of the quad bike rider was captured using a DJI Phantom 3 Adv in areas of Lough Coolin and Mount Gable in Ireland in overcast conditions and at varying distances (around 20–50 m). Precise specifications of the video were not available. The video was downloaded from Youtube for fair use in MP4 format and at a resolution of 1280×720 pixels. The video was submitted to Youtube by user and channel name "JGK Drone" [35]. As only one video was available the training and test set were made up of different sections of the 3 min 37 s video, exactly 144 frames or 1 min and 12 s of video were used for testing sequences and the rest was used as the training set. Areas that were overly dark or shadowy were avoided for reasons that will be explored in the discussion, although these were still used in the training dataset. Frames used in testing were cropped to surround the quad bike. Images were flipped vertically using bash to increase the number of training images. Some training images overlapped slightly with the test set.

4. Results

The results from each dataset are split with the Y'UV input results in Table 2 and the CLCM input results in Table 3. The tables include the name of each dataset, the number of blocks used in the SAT grid, the average accuracy of individual CF-ELMs achieved in the tuning phase, the average time taken in testing per frame, the precision and recall rates for all frames. Accuracy, precision and recall can be expressed:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{17}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{18}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{19}$$

where TP is true positive, TN is true negative, FP is false positive and FN false negatives.

The results in Tables 2 and 3 indicate better overall performance in the cattle and ATV datasets for the MEC-ELM with CLCM texture inputs. The MEC-ELM with Y'UV inputs performed best on the thistle dataset with a precision rate of 98% and recall at 84%. The Y'UV variant produced much faster processing times, with all datasets processing around half a second to a second per frame. The CLCM variant took between 1 and 2 s longer due to the time needed to process the CLCM and get the texture values. The Y'UV variant did not perform very well in the cattle datasets. In both cases it required more grid blocks and recorded many false positives. This included the reflection of the cattle in the water in the case of the stationary camera and a large number of false positives in the tree line and on the helipad in the drone dataset, this dataset was also the slowest of the four because of the multiple scales required to get true positives. The CLCM variant recorded a number of false positives on the helipad as well, particularly in the last ten frames. Discarding the last ten frames puts the precision at 84%. Sample output frames from each of the 4 datasets are in Fig. 5, where B (cattle stationary) and D (bull thistle) are from the Y'UV variant and A (quad bike) and C (cattle drone) are from the CLCM variant. Note the detection of the cow reflection in B. This problem did not occur in the CLCM variant.

In Fig. 6 there is a receiver operator characteristic (ROC) curve that is depicting the average classification rates for different sized SAT grids, with area under the curve (AUC). There were four different sized SAT grids tested and these results were conducted using the tuning images, with 50 images of the target object and 50 images of surrounding landscape. For this test the thistle dataset with the Y'UV variant was chosen. This was chosen as a demonstration dataset, as the difference between true positive and false positive rates appeared most obvious. The test was conducted once with a SAT grid of sizes 9, 25, 100 and 2500. The four MEC-ELMS were trained and test on the tuning images and the average TP and FP rates from the four were saved. The average rates were then used to make the ROC curve. As can be seen the larger the SAT grid the better the results, with the best results coming from the SAT grid with 2500 blocks. Notably the results are not as good as the tuning accuracies found in the result Tables 2 and 4. The testing in this case was based on an on-line training method were all weights were randomised and tested immediately after. The MEC-ELMs used to get the precision and recall rates were trained using an off-line method, where the CF-ELMS were initialised a number of times and the weights for the CF-ELM producing the best results was saved in a text based file [9]. This meant the results in Fig. 6 were therefore based on the average of randomly trained classifiers, rather than the best of 100 classifiers. In further testing processing with 9 blocks took an average of 0.35 s per frame, 25 blocks took 0.44 s per frame, 100 blocks took 1.1 s per frame and 2500 blocks took around 67 s per from.

5. Discussion

Multiple expert or cascading approaches to object detection are often limited to rudimentary approaches and this is mostly due to time constraints, particularly in the pursuit of real time results. This paper has presented a multiple-expert colour feature extreme learning (MEC-ELM) for real time feature extraction and object classification. The algorithm allowed real time results by adopting a summed area table and hence, converted the images into a low resolution and multiple scale environment. This allowed the MEC-ELM to process video frames and classify objects in less than a second per frame for Y'UV inputs and 1–2 s in the case of the CLCM implementation on a standard mobile CPU. The

Table 2
Testing results for each dataset using Y'UV inputs based on all frames of test videos, with accuracy detected during the tuning stage.

Y'UV Dataset	SAT grid blocks	Accuracy in tuning	Time/frame	Precision	Recall
Bull Thistle	25	97%	0.44(s)	98%	84%
Cattle Drone	100	81%	1.19(s)	23%	53%
Cattle Stationary	100	89.25%	0.53(s)	84%	78%
ATV	25	93%	0.49(s)	91%	78%

Table 3
Testing results for each dataset using the CLCM texture values based on all frames of the test videos, with accuracy detected during the tuning stage.

CLCM	Dataset	SAT grid blocks	Accuracy in tuning	Time/frame	Precision	Recall
	Bull Thistle	25	91%	1.2(s)	84%	81%
	Cattle Drone	25	93.5%	1.9(s)	78%	93%
	Cattle Stationary	25	94%	1.3(s)	85%	84%
	ATV	25	90%	2.3(s)	84%	91%

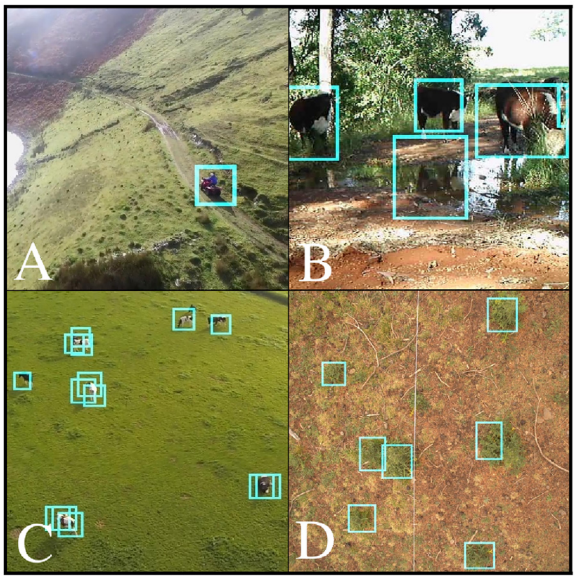

Fig. 5. Sample result from detection in each dataset.

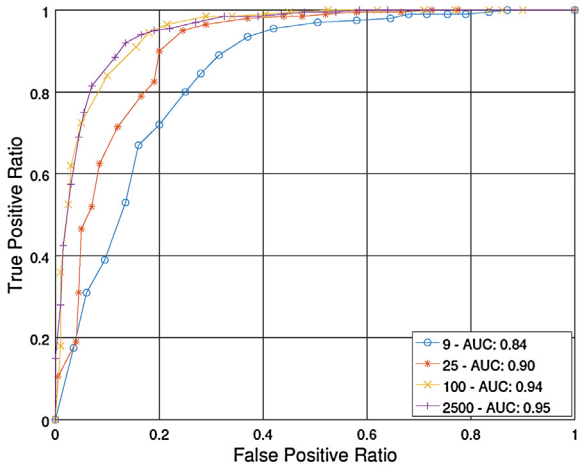

Fig. 6. ROC curve measuring the average classification rates for different sized SAT grids.

multiple scale environment gave the MEC-ELM the ability to process at multiple scales per second, with 2–4 different scales used in each of the datasets. This means that objects can be detected up close, far away or in smaller forms (a calf or thistle in rosette). Another advantage of using the SAT grid is the ability to produce low resolution images. Although this would be a problem for classifiers that require high resolution images, a cascading approach such as the MEC-ELM can improve accuracy by forming a consensus among experts. The SAT grid is essentially a *naive* scale [36] and at 25 blocks per image, the resolution would be considered quite low. There are many advantages to this, by reducing sections of the image into blocks there is the possibility to reduce noise in the image and by reducing the size of each frame, the video feed could be processed at a much lower resolution. This would decrease the processing times by a considerable amount. This would also have an impact on hardware restrictions, that is, processing smaller images would lessen the memory requirements for on board computing and allow faster transfer speeds while lessoning storage issues.

Low resolution images have some drawbacks, loss of pixel information for example can be a serious problem. Efforts should be made to preserve and/or retrieve some of this lost data and this is reason the CLCM variant of the MEC-ELM was proposed. The CLCM was slower than the Y'UV variant, but managed to alleviate some of the problems found in the cattle datasets. The Y'UV variant, although very good at detecting objects where colour is of some importance (the red quad bike and green thistle), struggled detecting multi-coloured objects, such as black and white cows and delivered a lot of false positives. The CLCM removed many of the false positives and was able to detect multiple colour objects with little difficulty; 1–2 s at multiple scales. This result was an improvement when compared to similar solutions in literature. The "Verschoor Aerial Cow Dataset" was previously tested at 32–144 s per frame and at around 66% precision and 87% recall [34]. The CLCM variant, although it was only tested on one video from the dataset, produced 78% precision and 94% recall in 1.9 s. These times make the MEC-ELM comparable to the fast processing Yolo object detection framework, which has processing times around 6–12 s per frame [37] on a standard CPU. It is however difficult to compare at this stage, as Yolo is commonly benchmarked on a high end GPU and the above citation was programmed in *Python* and not *C*. A comparison between *C* and *Python* is available in Fourment and Gillings [38]. This paper has limited itself to a mobile CPU for the purpose of field based analysis. It is conceivable however that GPUs will become more commonplace in the agricultural environment. Conveniently the SAT Grid is quite adaptable to different resource constraints and as the availability of hardware resources increases, the number of blocks used could also increase. This will increase the accuracy of the algorithm as the constraints are lessened. At lower resolutions it will still be advantages to a GPU, allowing processing of much larger areas while keeping processing times down.

Notably, the results in the dataset were not fully balanced, the differences between precision and recall in some of the datasets for example could have been much closer. This could be achieved with more precise tuning or a well established tracking algorithm [39,40]. The tracking algorithm developed was minimal to save processing time, future research could explore more sophisticated tracking algorithms [41]. Another issue that could be resolved is the problem of shadows or overcast. The quad bike video for example went through a lot of transitional lighting and the classification would suffer during these phases. This can depend on the training set of course, but the incorporation of illumination invariant or illumination robust images would likely further improve the performance of the algorithm.

The results demonstrate the ability of the MEC-ELM as a low resolution real-time object detection algorithm for drone and

surveillance video capture in the agriculture industry. Choosing between the best algorithm depends on the dataset, with the Y'UV performing better in weeds and the CLCM performing better in cattle detection. While they both performed well in ATV detection. The choice of datasets and capture methods give the algorithms a good indication of how they will perform in a live scenario. The algorithms were benchmarked on a laptop, while real-time wireless transfer has already been establish [42]. Calibrating the algorithm for live use should now be a process of choosing the right frame rate and compression methods. JPEG images were chosen for this reason, as it produces smaller image sizes for image transfer. Global position systems (GPS) can be incorporated into drone and surveillance equipment to give location feedback.

The usefulness of the algorithms can be exemplified in each of the chosen datasets. In weed detection, the algorithm could be used to locate infestations quickly. A farm hand or ground based robot could then be used to deliver chemical to affected areas. Delivering chemicals to precise areas rather than entire areas would save chemical and reduce cost. The cattle detection scenario could be used to track and count cattle in an unobtrusive way. ATV detection could be used for vehicle safety monitoring, locating missing or injured persons and in collision avoidance systems. In each case the algorithm could be used in a stand alone device such as a drone, unmanned ground vehicle (UGV) or stationary surveillance camera.

This paper has demonstrated through quantifiable properties the MEC-ELM's potential as a real time object detection algorithm for on board and/or remote computer based technology. The datasets used placed particular emphasis on drones and displayed the algorithm's ability to perform with aerial video data. This is particularly important given the convenience of drones, with their ability to fly long distances over difficult terrain and report back results in real time [42]. This point is particularly important in agriculture, but the MEC-ELM was designed to be a generic approach to object detection in complex environments and for this reason could prove useful in many different scenarios, particularly where fast processing is required.

6. Conclusion

Two implementations of the multiple expert colour feature extreme learning have been benchmarked as real time feature extraction and object classification algorithms. This included a low resolution Y'UV colour implementation and a CLCM texture based implementation. The Y'UV implementation produced superior results in the datasets where colours were uniform and a defining characteristic, including weed detection with a precision of 98% and a recall of 84%, while producing lower accuracy in multiple coloured datasets, such as cattle detection. The CLCM variant was able to produce consistent results through all the datasets and higher results overall in three, including quad bike detection with 84% precision and 91% recall, cattle detection from a stationary camera at 85% precision and 84% recall and cattle detection from a drone, at 78% precision and 93% recall. The Y'UV variant was able to process video frames in half a second for three of the datasets and just over 1 s for cattle detection from a drone. The CLCM processing times were between 1 and 2 s. These processing times indicate the algorithms ability to process the images within a time frame suitable for agricultural robotics applications. This is particularly notable given its ability as a low resolution classifier. Future research will involve testing the algorithm on an embedded device attached to a drone or land vehicle and testing the algorithm in different lighting conditions using different illumination equalisation techniques. There is also potential for testing the MEC-ELM on a GPU for a comparison to similar real-time object detection algorithms.

Acknowledgements

Dr Paul Meek, NSW Department of Primary Industries for the provision of cattle surveillance data. Animal Ethics UNE AEC12-042, collected under scientific licence Sci Lic SL 100634.

Mr Andrew Rieker of V-TOL Aerospace PTY Limited, for providing footage of weeds using their quadcopter.

Mr Paul Arnott, UNE Kirby Smart Farm, for access to paddocks for photography of weeds.

Mr E. Sadgrove is supported by an Australian Postgraduate Award.

References

[1] T. Berge, S. Goldberg, K. Kaspersen, J. Netland, Towards machine vision based site-specific weed management in cereals, Comput. Electron. Agric. 81 (2012) 79–86.
[2] M.S. Uddin, A.Y. Akhi, Horse detection using Haar like features, Int. J. Comput. Theory Eng. 8 (5) (2016) 415–418.
[3] J. Wawerla, S. Marshall, G. Mori, K. Rothley, P. Sabzmeydani, Bearcam: automated wildlife monitoring at the arctic circle, Mach. Vis. Appl. 20 (2009) 303–317, doi:http://dx.doi.org/10.1007/s00138-008-0128-0.
[4] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015) 779–788.
[5] T. Malisiewicz, A. Gupta, A.A. Efros, Ensemble of Exemplar-SVMs for object detection and beyond, International Conference on Computer Vision (ICCV), IEEE, 2011, pp. 89–96.
[6] P.M. Pandiyan, C. Hema, P. Krishnan, S. Radzi, Colour recognition algorithm using a neural network model in determining the ripeness of a banana, International Conference on Man-Machine Systems (ICoMMS), Batu Ferringhi, Malaysia, 2009 pp. 2B71–2B–74.
[7] J. Xu, H. Zhou, G.-B. Huang, Extreme learning machine based fast object recognition, 15th International Conference on Information Fusion (FUSION), IEEE, Singapore, 2012, pp. 1490–1496.
[8] X. Chen, H. Mulam, An implementation of faster RCNN with study for region sampling, Tech. rep., Cornell University, 2017.
[9] E.J. Sadgrove, G. Falzon, D. Miron, D. Lamb, Fast object detection in pastoral landscapes using a multiple expert colour feature extreme learning machine, The International Tri-Conference for Precision Agriculture (PA17), 1st Asian-Australia Conference on precision Pastures and Livestock Farming (2017).
[10] E.J. Sadgrove, G. Falzon, D. Miron, D. Lamb, Fast object detection in pastoral landscapes using a colour feature extreme learning machine, Comput. Electron. Agric. 139 (2017) 204–212.
[11] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (2006) 489–501.
[12] R. Moreno, F. Corona, A. Lendasse, M. Graña, L.S. Galvão, Extreme learning machines for soybean classification in remote sensing hyperspectral images, Neurocomputing 128 (2014) 207–216, doi:http://dx.doi.org/10.1016/j.neucom.2013.03.057. http://www.sciencedirect.com/science/article/pii/S0925231213010102.
[13] S. Malek, Y. Bazi, N. Alajlan, Efficient framework for palm tree detection in UAV images, IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens. 7 (12) (2014) 4692–4703.
[14] G. Facciolo, N. Limare, E. Meinhardt-Llopis, Integral images for block matching, Image Process. On Line 4 (2014) 344–369.
[15] A. Mohamed, A. Issam, B. Mohamed, B. Abdellatif, Real-time detection of vehicles using the Haar-like features and artificial neuron networks, Procedia Comput. Sci. 73 (2015) 24–31.
[16] M. Benco, R. Hudec, S. Matuska, M. Zachariasova, One-dimensional color-level co-occurrence matrices, Elecktro, IEEE, 2012, doi:http://dx.doi.org/10.1109/ELEKTRO.2012.6225600.
[17] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: a new learning scheme of feedforward neural networks, Proceedings. 2004 IEEE International Joint Conference on Neural Networks, vol. 2 (2004) 985–990.
[18] Ö.F. Ertuğrul, Y. Kaya, A detailed analysis on extreme learning machine and novel approaches based on ELM, (2015) . http://www.openscienceonline.com/journal/ajcse.
[19] G.S.S. da Gomes, T.B. Ludermir, L.M.M.R. Lima, Comparison of new activation functions in neural network for forecasting financial time series, Neural Comput. Appl. 20 (3) (2011) 417–439.
[20] M. Loesdau, S. Chabrier, A. Gabillon, Hue and saturation in the RGB colour space, Conference: 6th International Conference, Vol. 8509 of Lecture Notes in Computer Science, Cherbourg, France, 2014, pp. 203–212.
[21] Studio Encoding Parameters of Digital Television for Standard 4:3 and Widescreen 16:9 Aspect Ratios: Recommendation ITU-R BT.601-6: (Question ITU-R 1/6), International Telecommunication Union, 2007. URL https://books.google.com.au/books?id=NpAwPwAACAAJ.
[22] S. Bashbaghi, E. Granger, R. Sabourin, G.-A. Bilodeau, Ensembles of exemplar-svms for video face recognition from a single sample per person, 12th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (2015) 1–6.

[23] N. Liu, H. Wang, Ensemble based extreme learning machine, IEEE Signal Process. Lett. 17 (8) (2010) 754–757.
[24] P. Viola, M. Jones, Robust real-time object detection, Int. J. Comput. Vis. 57 (2) (2004) 137–154.
[25] R. Wang, Adaboost for feature selection, classification and its relation with SVM, a review, Phys. Procedia 25 (2012) 800–807.
[26] A.H. Bishak, Z. Ghandriz, T. Taheri, Face recognition using a co-occurrence matrix of local average binary pattern (CMLABP), Cyber J. Multidiscip. J. Sci. Technol. J. Sel. Areas Telecommun. (JSAT) (2012) 15–19.
[27] G.N. Srinivasan, G. Shoba, Statistical texture analysis, World Academy of Science, Engineering and Technology, vol. 36 (2008) 1264–1269.
[28] A. Eleyan, H. Demirel, Co-occurrence matrix and its statistical features as a new approach for face recognition, Turk. J. Elect. Eng. Comput. Sci. 19 (1) (2011) 97–107.
[29] J. Leskela, J. Nikula, M. Salmela, OpenCL embedded profile prototype in mobile device, Signal Processing Systems, IEEE, 2009, doi:http://dx.doi.org/10.1109/SIPS.2009.5336267.
[30] J. Tapson, P. de Chazal, A. van Schaik, Explicit computation of input weights in extreme learning machines, International Conference on Extreme Learning Machines, Vol. 1 of Algorithms and Theories, Springer, Switzerland, 2015, pp. 41–49.
[31] Netlib.org, The LAPACKE C interface to LAPACK, 2013, November. http://www.netlib.org/lapack/lapacke.html.
[32] Studio encoding parameters of digital television for standard 4:3 and widescreen 16:9 aspect ratios, Tech. rep., International Telecommunications Union, Electronic Publication, Geneva, 2015.
[33] S. Hartig, Basic image analysis and manipulation in imagej Chapter 14, (2013) Unit14.15.
[34] J.C. van Gemert, C.R. Verschoor, P. Mettes, K. Epema, L.P. Koh, S. Wich, Nature conservation drones for automatic localization and counting of animals, European Conference on Computer Vision Workshop, ECCV Workshop on Computer Vision in Vehicle Technology (2014) 255–270.
[35] JGK Drone, Quad biking drone footage, 2016, November. https://www.youtube.com/watch?v=x5l-y7bWD9E.
[36] D.E. Culler, J.P. Singh, A. Gupta, Workload-driven evaluation, Parallel Computer Architecture: A Hardware/software Approach, 1st ed., (1998) , pp. 266.
[37] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, (2016, December) .
[38] M. Fourment, M. Gillings, A comparison of common programming languages used in bioinformatics, J. BMC Bioinf. 9 (2008) 82.
[39] D. Ghosh, N. Kaabouch, A survey on image mosaicing techniques, J. Vis. Commun. Image Represent. 34 (2016) 1–11.
[40] B.D. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision (IJCAI), Proceedings of the 7th International Joint Conference on Artificial Intelligence, vol. 81 (1981).
[41] Y. Wu, J. Lim, M.-H. Yang, Object tracking benchmark, IEEE Trans. Pattern Anal. Mach. Intell. 37 (9) (2015) 1834–1848, doi:http://dx.doi.org/10.1109/TPAMI.2014.2388226.
[42] G. Zhou, C. Li, P. Cheng, Unmanned aerial vehicle (UAV) real-time video registration for forest fire monitoring, Vol. 3 of Conference: Geoscience and Remote Sensing Symposium, IEEE International, Seoul, South Korea, 2005.