Image edge detection using SVM regression model for UAV autonomous navigation

Gracieth C. Batista[1], Osamu Saotome[1], Elcio H. Shiguemori[2], Wanessa da Silva[3], Haroldo F. de Campos Velho[3]

[1]*Dept. of Electronic Devices and Systems, Technological Institute of Aeronautics (ITA)*
Praça Marechal Eduardo Gomes, 50 – Vila das Acacias, São José dos Campos, 12228-900, SP,Brazil
gracieth@ita.br, osaotome@gmail.com
[2]*Institute for Advanced Studies (IEAv), Dept. of Aerospace Science and Technology (DCTA)*
Av. Brigadeiro Faria Lima, 1941 – Parque Martin Cerere, São José dos Campos, 12227-000, SP, Brazil
elciohs@gmail.com
[3]*National Institute for Space Research (INPE)*
Av. dos Astronautas, 1758, São José dos Campos, 12227-010, SP, Brazil
w.nessa.w@gmail.com, haroldo.camposvelho@inpe.br

Abstract. Currently, unmanned aerial vehicle (UAV) has been used for many applications. Few applications include agriculture, engineering, monitoring, rescue, entertainment, and others. One research topic involving UAV is related to autonomous navigation. A standard procedure for this process to UAV is to combine inertial sensor information with the Global Navigation Satellite System (GNSS) signal. However, some factors can interfere with the GNSS signal associated with natural phenomena or malicious attacks (jamming or spoofing). One alternative to overcome using the GNSS signal is to apply an image processing approach based on matching UAV images and georeferenced images. There is a great computational effort on this approach for computing the image edge extraction. A Support Vector Machine (SVM) regression model, also known as Support Vector Regression - SVR, is employed for edge detection to reduce computational load and processing time. Our proposal consists of three general steps; first: pre-processing of the images, where frames of 3x3 pixels were obtained for characterizing edge or non-edge patterns; second, SVR models were trained, where the predictors were normalized; and finally, an i.i.d. (independently and identically distributed) test set was used to predict SVR respective responses. The better-performing model was acquired using the Gaussian kernel function compared to two other kernel functions (linear and polynomial). Its generalization error is that the out-of-sample mean-squared error (MSE) was 18 times less than the Linear kernel MSE error. The success rate was 99.98% of accuracy.

Keywords: UAV - unmanned aerial vehicle, Image edge detection, SVM regression model.

1 Introduction

Unmanned aerial vehicle (UAV) is a technology widely used nowadays for many applications: agriculture, rescue services, surveillance, environmental monitoring, just citing a few of them. Our paper is focused on UAV autonomous navigation. For adddressing this feature, one strategy is to combine signals from an inertial navigation system (INS) onboard and data from a global navigation satellite system (GNSS). Natural phenomema, such as ionospheric cintilation, or electronic attacks (jamming or spoofing) can cause fails or interference with the GNSS signal. Image processing can be an option to be an alternative of use of GNSS signal.

Conte and Doherty [1], Braga and co-authors [2] can be mentioned by employing image processing for the UAV positioning without to use a GNSS signal. Another approach using image template matching was presented by Torres et al. [3]. Indeed, computer vision schemes can be used as a viable alternative to the GNSS.

The methodology adopted here follows the proposal of Silva et al. [4]. The formulation is to extract edges from the two images: one is catched from the drone camare, and another one is a georeferenced satellite image. After that, a correlation of these two segmented images is computed, where the maximum correlation point indicates the UAV position – Figure 1 displays the steps for UAV positioning using image processing. Several pre-processing

procedures are performed, including to converte both images to gray scale, adjust the UAV with rotation and space scale to the satellite image, possible low-pass filter, and image binarization process (black and white image). One key issue and the most expensive computational effort in the image processing is addressed by edge extraction.

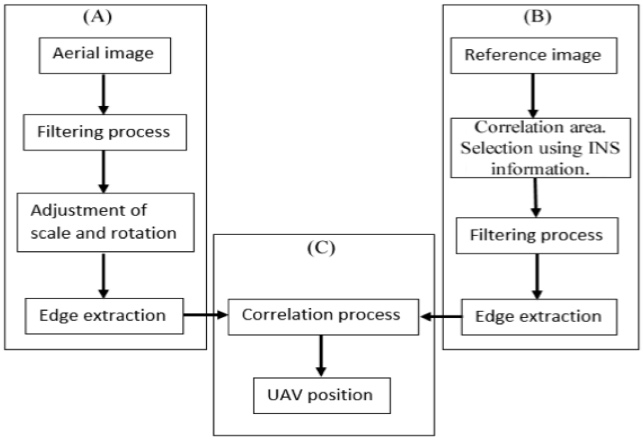

Figure 1. Methodology for UAV positioning by image processing.

This paper deals on the investigation of the support vector machine (SVM) for edge extraction. The SVM approach will be also evaluated in comparison with another machine learning method for edge extraction: the artificial neural network. The SVM is configured for function estimation, typified as support vector regression (SVR). The multilayer perceptron (MLP) is the neural network (SNN) employed, and the back-propagation algorithm is applied during the learning phase – the scheme to identify the best weights for fully connected the artificial neurons.

The paper proceeds as follows. Section 2 describes some concepts with mathematical modeling of the applied machine learning methods and the details about their applications in this proposal. Section 3 shows the results obtained for both methodologies; and finally, Section 4 stresses the conclusions and ideas for future work.

2 Machine learning approaches

Machine learning is a class of algorithms where the estimation operator is partially configured by using data associated to a specific application. Basically, they are data driven computational schemes. The process can be understood as a part of artificial intelligence. Such procedures allow to design a computer model for making predictions or decisions under situation without of the training dataset. Only supervised algorithms are employed as technique for image edge extrators.

2.1 Artificial neural networks: multilayer perceptron (MLP)

The MLP is a supervised feedforward neural network [5]. The MLP-NN has an input layer (receiving data for processing), one or more hidden layers (where artificial neurons are the processing units fully connected), and one output layer – this last layer can or cannot be composed of computational neurons. Considering the square difference between the reference values and the MLP-NN output denoted by $e(k)$, the *back-propagation algorithm* (BPA) is the scheme for learning process using $e(k)$. The connection weights can be iteratively adjusted by the BPA as following:

$$\Delta w(k+1) = \alpha \Delta w(k) + \eta \delta(k) y(k) \tag{1}$$

where $w(k+1) = w(k) + \Delta w(k+1)$, being w the weight matrix and k the number of iteration (*epoch*), α is the parameter called *momentum*, η is the *learning rate* value, $y(k)$ is the output computed by the MLP-NN, and $\delta(k) \equiv \partial\{e(k)\}/\partial w$ is the gradient of the square difference between the reference and the NN output. The MLP-NN implemented in the Matlab software was used here.

Optimal MLP-NN architecture

One challenge for using an artificial neural network is to identify the best architecture for a specific application. Indeed, it is not an easy task. A strategy to address this issue is to formulate the searching for a good NN

topology as an optimization problem [6]. The objective function searches for the simplest neural network topology (lowest number of neurons, and faster convergence) with the best performance (lowest errors). The goal is to minimize a functional described by a weighted average of two square difference terms (E_{train},(E_{gen}): reference values and the ANN output for two datasets, one of them representing the data for training (E_{train}) and another one the NN results after the learning phase (E_{gen}), and a penalty term. The cost function is given by [7]:

$$J = penalty \times \left(\frac{\rho_1 \times E_{\text{train}} + \rho_2 \times E_{\text{gen}}}{\rho_1 + \rho_2} \right), \tag{2}$$

where $\rho_1 = 1$ and $\rho_2 = 0.1$ – see Carvalho et al. [7][1]; $penalty$ is neural network complexity meajurement, taking into account the number of neurons in the hidden layer(s) (n_{neurons}) and the number of epochs up to the convergence (n_{epochs}). The penalty term is expressed by:

$$penalty = c_1 e^{(n_{\text{neurons}})^2} + c_2(n_{\text{epochs}}) + 1, \tag{3}$$

where $c_1 = 1$ and $c_2 = 0.1$ are free parameters.

The optimization problem is solved by the meta-heuristic MPCA (multi-particle collision algorithm) [8], calculating a neural topology with lower difference between NN-output and reference datasets with the simplest arrangement.

2.2 SVM regression (SVR): concepts and approach

The Support Vector Regression (SVR) model is formulated from the Support Vector Machines (SVMs). In this matter, the SVM algorithm is based on the Statistical Learning Theory and VC dimension [9]; besides, it is a binary algorithm that can solve nonlinear problems and generalizes well to unseen data according to the VC dimension theory [10]. The SVM was largely developed at AT&T Bell Laboratories by Vapnik and co-workers [11–13]. Since then, it has been used for recognition tasks, image processing, classification problems, diagnosis, proagnosis, and others. This algorithm builds an optimum hyperplane to reach the best separation between two classes using a transformation from the input space to a space of high dimensionality, where a kernel function is applied to deal with nonlinearity. SVM can be generalized to become applicable for *regression* problems.

As in the SVM concept, SVR is also characterized by the use of sparse solution, kernels, VC control of the margin and the number of *support vectors*. Although less popular than SVM, SVR has been proven to be an effective tool in real-value function estimation. As a supervised-learning approach, SVR trains using a symmetrical loss function, which equally penalizes high and low misestimates. Using Vapnik's ε-insensitive approach, a flexible tube of minimal radius is formed symmetrically around the estimated function, such that the absolute values of errors less than a certain threshold ε are ignored both above and below the estimate. In this manner, points outside the tube are penalized, but those within the tube, either above or below the function, receive no penalty. One of the main advantages of SVR is that its computational complexity does not depend on the dimensionality of the input space. Additionally, it has excellent generalization capability, with high prediction accuracy [14].

From a training data set $g = \{(x_1, y_1), (x_2, y_2), ..., (x_i, y_i)\}$, using the SVR technique means to find the object function $min_w \dfrac{1}{2}||\omega||^2$ subject to Equation 4, where $||\omega||$ is the magnitude of the normal vector to the surface that is being approximated by this minimization.

$$|y_i - \langle \omega, x_i \rangle - b| < \varepsilon \tag{4}$$

where x_i is a training sample with labeled value as y_i. The inner product plus intercept $\langle \omega, x_i \rangle + b$ is the prediction $f(x_i)$ (technically known as the multivariate regression) for that sample, b is a bias, and ε is the free threshold parameter that represents the radius of the tube located around the regression function (see Figure 2).

The minimization task involves the minimization of a empirical loss function (or empirical risk, which is the prediction error of the training outputs) and obtaining as small a ω as possible, using the training set g. The applied loss function is the *ε-insensitive loss function* given by Equation 5.

$$C(f(x) - y) = |f(x) - y| - \varepsilon, \;\; for\; |f(x) - y| \geq \varepsilon, \;\; otherwise\; C = 0 \tag{5}$$

The SVR's tube has the goal of covering the maximum data points as possible, as shown in Figure 2. The size of the stated excess positive and negative deviations are depicted by ξ and ξ^*, which are called *slack* variables. Outside of the ($-\varepsilon$, ε) region, the slack variables assume non-zero values [15].

[1] However, the user can define different values.

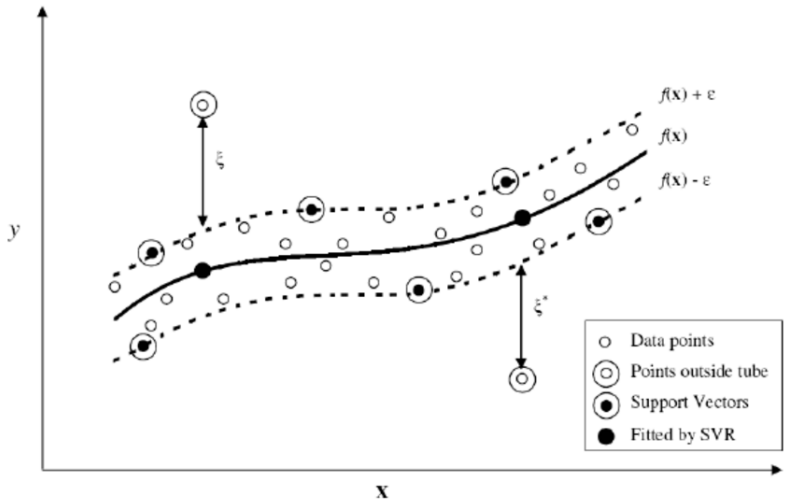

Figure 2. An illustration of the flexible tube generated by the SVR technique using ε-insensitive loss function.

Then, once we have discussed the concept and mathematical modeling of the SVR briefly, we can go further to its application in this paper's proposal. First, the SVR training phase is aborded; next, we explain the prediction phase.

SVR modeling: training phase

The SVR training phase was implemented from a binarized dataset of 208-frame samples of 9 pixels each. The three kernel functions available on the *Matlab* software for the used *fitrsvm* function were tested: Gaussian or Radial Basis Function (RBF), Polynomial of 3^{rd} order and Linear kernels. Besides, to compare the generalization error of the models, in this case, the generalization error is the out-of-sample Mean-Squared Error (MSE) generated from cross-validated models (through 5-fold cross-validation) using regression partitioned SVM; the MSE was computed from each kernel application.

SVR modeling: prediction phase

At last, we used the trained models to predict the response values from a binarized dataset of 88-frame samples of 9 pixels each. The respective response values were compared to a labeled dataset of 88 samples, where each sample corresponds to the correct answer; i.e., if the label is '1' then, the right answer for that respective frame belongs to an edge pattern; otherwise, it belongs to a non-edge pattern. For this step, the *predict* function was applied in the *Matlab* software. It is important to highlight that the datasets for training and prediction are independently and identically distributed (i.i.d.).

3 Results and discussion

As mentioned before, the edge extraction is the key point for the computational effort of the methodology for UAV positioning based on image processing. The procedure is based on to identify if the pixel belong or not to the border of an object inside of image. In order to characterize the pixel, a set of pixels around to the pixel to be identified are used to decide the type of pattern the pixel to belong. Figure 3-(a) shows the edge pattern samples (upper part with 24 examples) and non-edge (lower part with 2 samples). Figure 3-(b) displays the georeferenced satellite image. For our expriment, the UAV's flight is around the exagon in the center of the image. For performing the UAV position, several procedures are executed: image catched by the UAV, scaling and rotation for direction matching between UAV and satellite images, image gray scale, image binaritazion, segmentation of images, and finally the convolution between treated images. The job here is to identify the type of edges in the images by machine learning strategies. Neural networks have been explored to this characterization – see [2, 3, 16, 17]. However, our team always is investigating new approaches to achieve better performance for our methodology.

The UAV image acquisition is obtained at same time interval during the flight, where all image processing procedures are executed. In this paper, three kernels (Gaussian, Linear, Polynomial) for SVR method are evaluated for edge extraction. The results are compared with the segmentation obtained from MLP-NN, with the architecture determined as an optimal solution by the MPCA metaheurisc. Here, only the accuracy of both schemes are verified.

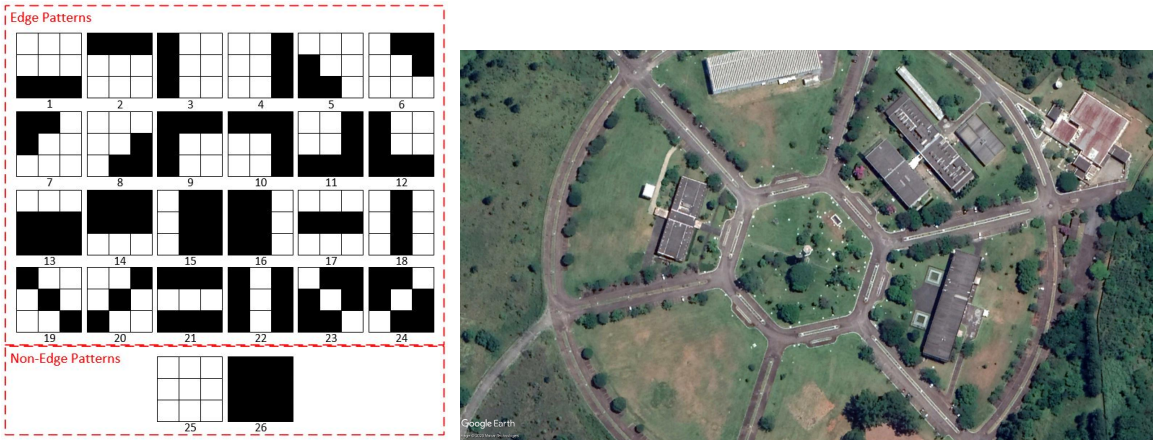

Figure 3. (a) Image patterns of edge and non-edge for the training phase input; (b) Satellite image.

3.1 ANN-MLP algorithm

Two standard methods for edge extractration are the Sobel's Canny's algorithms, as employed by Conte and Doherty [1]. Our innovation for the technology to UAV autonomous navigation was to apply artificial intelligence techniques for the image edge identification. The multilayer perceptron is a supervised machine learning scheme, and its configuration was determined by the MPCA. The MLP-NN toplogy is described in Table 1. The neural network was employed by edge identification, and it was effective.

Table 1. MLP-NN architecture determined by the MPCA metaheuristic.

Type of paramter	MLP-NN
# neurons – input layer	9
# neurons – hidden layer	18
# neurons – output layer	1
Activation function	hyperbolic tangent
Learning rate (η)	0.73
Momentum (α)	0.85
Max. training epochs	15000

3.2 SVR algorithm

From the SVR technique application, we made a comparison among the kernel functions as explained before, and the Gaussian showed the best performance with a MSE of 0.0146. In contrast, the Linear and Polynomial kernel functions obtained 0.2801 and 0.0586, respectively. Figure 4 shows the prediction phase (i.e., testing phase) results from a dataset of 88 pixels. Thus, we can analyze the Gaussian kernel function behaves the same as the observed data function, i.e., as it was supposed to behave, with 99.98% of accuracy. Besides, the response time of the SVR prediction phase of the 88 pixels was 0.745ms in the processor IntelCore i7-6500 CPU @ 2.50GHz - 64bits.

A comparison of success rate for SVR and NN is shown in Table 2. Both methodologies have a good performance and the differences ate not significant statisticaly.

Table 2. Success rate for training and testing considering different maching learning approaches.

Phase	SVR-G (%)	SVR-L (%)	SVR-P (%)	MLP-NN (%)
Training	99.99993	69.363578	99.450339	99.999412
Testing	99.18	72.31	91.37	89.98

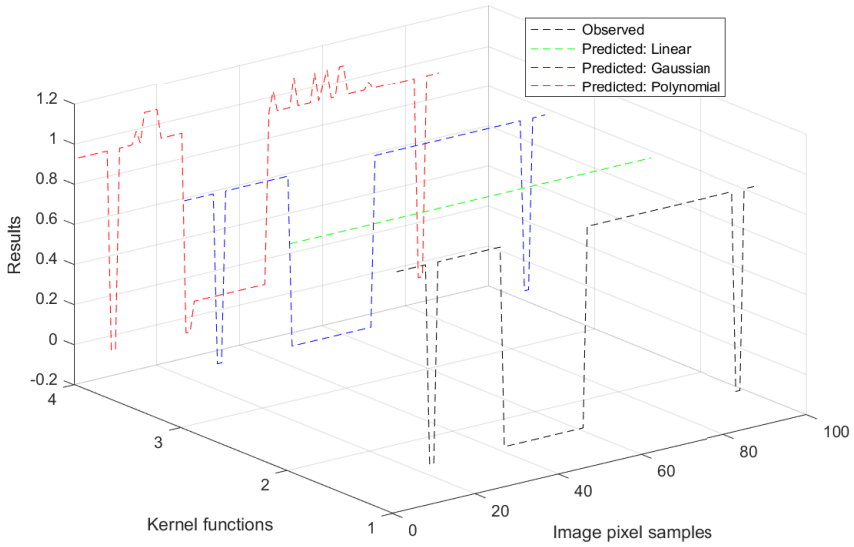

Figure 4. Graph of the SVR prediction results.

4 Conclusions

The support vector regression algorithm was applied and proved to be very effective for image edge extraction, with high degree of accuracy. Table 2 shows the degree of success of the evaluated methods. Clearly, the SVR using Gaussian (SVR-G) and polynomial (SVR-P) kernels presented higher degree of accuracy during the training phase than the linear kernel (SVR-L). However, the SVR-G was significantly superior than SVR-P in the testing phase.

In brief, the SVR-G shows good performance for edge extraction. This technique is strongly competitive with other schemes, such as Sobel and Canny's algorithms, as well with the MLP neural network.

Acknowledgements. The authors would like to thank the Brazilian agencies for research support. Author HFCV thanks to the National Council for Scientific and Technological Development (CNPq, Portuguese) for the research grant (CNPq: 312924/ 2017-8).

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors.

References

[1] G. Conte and P. Doherty. An integrated uav navigation system based on aerial image matching. In *IEEE Aerospace Conference*, pp. 1–10, 2008.

[2] J. R. Braga, H. F. Campos Velho, G. Conte, P. Doherty, and E. H. Shiguemori. An image matching system for autonomous uav navigation based on neural network. In *14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 1–6, Phuket, Thailand. IEEE, 2016.

[3] V. A. M. F. Torres, B. R. A. Jaimes, E. S. Ribeiro, M. T. Braga, E. H. Shiguemori, H. F. Campos Velho, L. C. B. Torres, and A. P. Braga. Combined weightless neural network FPGA architecture for deforestation surveillance and visual navigation of UAVs. *Engineering Applications of Artificial Intelligence*, vol. 87, n. 1, pp. 103227, 2020.
[4] C. A. O. Silva, G. A. M. Goltz, E. H. Shiguemori, C. L. De Castro, H. F. Campos Velho, and A. P. Braga. Image matching applied to autonomous navigation of unmanned aerial vehicles. *International Journal of High Performance Systems Architecture*, vol. 6, n. 4, pp. 205–212, 2017.
[5] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1998.
[6] S. B. M. Sambatti, J. A. Anochi, E. F. P. Luz, E. H. Shiguemori, A. R. Carvalho, and H. F. Campos Velho. Automatic configuration for neural network applied to atmospheric temperature profile identification. In *EngOpt: International Conference of Engineering Optimization*, pp. Paper code 406 (http://engopt.org/12/authors/optimization.html), Rio de Janeiro, Brazil. COPPE/UFRJ. Session: Optimization and Surrogates, 2012.
[7] A. R. Carvalho, M. F. Ramos, and A. A. Chaves. Metaheuristics for the feedforward artificial neural network (ANN) architecture optimization problem. *Neural Comput Applic*, vol. 20, pp. 1273–1284, 2011.
[8] E. F. P. Luz, J. C. Becceneri, and H. F. Campos Velho. A new multi-particle collision algorithm for otimization in a high-performance environment. *Journal of Computacional Interdisciplinary Sciences*, vol. 1, pp. 1–7, 2008.
[9] V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition [in Russian]*. Akademie-Verlag, Nauka, Moscow, 1974.
[10] A. J. Smola and B. Scholkopf. A tutorial on support vector regression. *Statistics and Computing*, vol. 14, n. 1, pp. 199–222, 2004.
[11] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Annual Conference on Computational Learning Theory*, Haussler D. (Ed.), pp. 144–152, Pittsburgh, PA. ACM, 1992.
[12] I. Guyon, B. Boser, and V. Vapnik. Automatic capacity tuning of very large vc-dimension classifiers. In *Advances in Neural Information Processing Systems 5*, Hanson S.J., Cowan J.D., and Giles C.L. (Eds.), pp. 147–155. Morgan Kaufmann Publishers, 1993.
[13] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, vol. 20, n. 1, pp. 273–297, 1995.
[14] M. Award and R. Khanna. *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Appress, Berkeley - CA, 2015.
[15] S. K. Lahiri and K. C. Ghanta. The support vector regression with the parameter tuning assisted by a differential evolution technique: Study of the critical velocity of a slurry flow in a pipeline. *Chemical Industry and Chemical Engineering Quarterly*, vol. 14, n. 3, pp. 191–203, 2008.
[16] G. A. M. Goltz, E. H. Shiguemori, and H. F. Campos Velho. UAV position estimation by image processing using neural networks. In *X Congresso Brasileiro de Inteligência Computacional*, pp. 9–17, Fortaleza, Brazil, 2011.
[17] W. Silva, E. H. Shiguemori, N. L. Vijaykumar, and H. F. . Campos Velho. Estimation of UAV position with use of thermal infrared images. In *International Conference on Sensing Technology (ICST-2015)*, pp. 211–217, Auckland, New Zealand: School of Engineering and Advanced Technology – Massey University, 2015.