Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXXVII, págs. 2033-2043 (artículo completo)
A. Cardona, L. Garelli, J.M. Gimenez, P.A. Kler, S. Márquez Damián, M.A. Storti (Eds.)
Santa Fe, 5-7 Noviembre 2019

DRONE AUTONOMOUS NAVIGATION BY HARDWARE IMAGE PROCESSING

José Renato G. Braga[a], Haroldo F. de Campos Velho[a], Elcio H. Shiguemori[b] and Patrick Doherty[c]

[a]*Laboratório de Computação e Matemática Aplicada (LABAC), Institudo Nacional de Pesquisas Espaciais (INPE), São José dos Campos (SP), Brasil,* jgarciabraga@gmail.com / haroldo.camposvelho@inpe.br, *http://www.lac.inpe.br*

[b]*Instituto de Estudos Avançados (IEAv), Departamento de Ciência e Tecnologia Aeroespacial, São José dos Campos (SP), Brasil,* elcio@ieav.cta.br, *http://www.ieav.cta.br/*

[c]*Department of Computer and Information Science (IDA), Linköping University, Linköping, Sweden,* patrick.doherty@liu.se, *https://liu.se/en*

Keywords: UAV autonomous navigation, Visual odometry, Computer vision, FPGA, Non-extensive particle filter.

Abstract. Our approach for autonomous navigation is to apply image processing for estimating the drone position. Two techniques are employed: visual odometry and computer vision. Edge detection is one important step for computer vision and it is performed by neural network implemented on FPGA. After image segmentation, a correlation between the satellite image – or reference image – and the image obtained by the drone is computed. Finally, the positioning by visual odometry and computer vision are combined using a new formulation of particle filter, called *non-extensite particle filter*. Our results show better results in comparison with other edge identification procedures, implying a more precise trajectory correction.

1 INTRODUCTION

The use of Unmanned Aerial Vehicle (UAV) has found many fileds of applications, such as agriculture (Psirofonia et al., 2017) and cattle (Chamoso et al., 2014), rescue services (Naranjo et al., 2016), environmental (Messinger and Silman, 2016), surveillance (Motlagh et al., 2017), just to mention some.

One important issue in the UAV research is the autonomous navigation, where signals from a Global Navigation Satellite System (GNSS) with Inertial Navigation System (INS) are commonly employed. However, the GNSS signal may fail, one needs to resort to alternative techniques to the mentioned one. Image processing-based schemes offer vaible solutions for UAV positioning.

Visual odometry (VO) and computer vision (CV) are two approaches based on image processing which can be applied for the UAV autonomous navigation (Conte and Doherty, 2009). A formulation for computer vision uses image edge identification catched by the UAV and satellite. Satellite data is a geo-referenced image. Edge extraction is performed by an artificial neural network (ANN) (Haykin, 1998). An automatic topology for the ANN can be found by solving an optimization problem (Anochi and Campos Velho, 2014; Anochi et al., 2015). The correlation between the segmented images is calculated to compute the UAV location. Here, the ANN is implemented on a hardware device: FPGA (Field Programmable Gate Array).

The two positioning schemes – VO and CV – are combined using a new version of the Particle Filter (PF): the non-extensive particle filter (NExt-PF). The distrubution derived from the Tsallis' thermstatistics (Tsallis, 1988, 1999) is used as the likelihood operator for the new PF. The new Bayesian filter can quantify the uncertanty of the trajectory estimated by data fusion from the image processing procedures.

2 AUTONOMOUS NAVIGATION BY IMAGE PROCESSING

In this section, two approaches applied for UAV autonomous navigation, visual odometry (VO) and computer vision (CV), are briefly presented.

2.1 Visual Odometry

A method for finding the UAV location on the basis of the previous *drone* position and orientation is the Visual odometry (VO) (Nister et al., 2004; Scaramuzza and Friedrich, 2011). Only the monocular VO is employed in this paper for the UAV positioning of outdoor movement. In its classical application, the VO consists of extracting interest points and trcking them in the image sequence. By matching the interest points between two successsive drone images, taken after a period of time (Δt), one can estimate the UAV position.

Four procedures are needed to carry out the visual odometry:

1. *Image Sequence*,
2. *Detecting Points*,
3. *Matching between Points*,
4. *Movement*.

For the *Image Sequence* step, two images are captured by drone at the two instants $t - \Delta t$ and t. The time period Δt is selected for allowing a large ovelapping between the two captured images. The *Detecting Points* step determines the interest points for the images. The data

structure for the interest points is storaged in an attribute vector – the *descriptor* procedure. The SURF (Speeded Up Robust Features) is a well known descriptor (Bay et al., 2008). *Matching between Points* is another step applied to match the interest points using an attribute vector. Some similarity metric is used for the matching. Finally, the *Movement* step calculates the UAV location from the corresponding pairs of the interest points. We use the eight point algorithm to compute the fundamental matrix F which for all pairs of corresponding interest points (x, x') is such as:

$$(x')^{\mathrm{T}}\, F\, x = 0\,. \tag{1}$$

The Singular Value Decomposition (SVD) of F can then be used to determine the UAV motion. Thus, we have

$$\mathrm{SVD}(F) = K^{\mathrm{T}}\, R\, [t]_x\, K^{-1} \tag{2}$$

where K is the matrix of the intrinsic sensor parameters linked to the vehicle, R is the rotation matrix, and $[t]_x$ is the representation of the cross product of the translation vector.

2.2 Computer Vision

Figure 1 shows the procedure used by the computer vision (CV) formulation. The *reference image* is a georeferenced satellite image, and the *aerial image* is the one catched by the drone. Several steps are used in the CV formulation. First of all, the images are mapped into a gray scale. After that, a median filtering process is applied (Gonzalez and Woods, 2017). Then, the multi-layer perceptron (MLP) neural network (Haykin, 1998) is used to extract edges from the images.

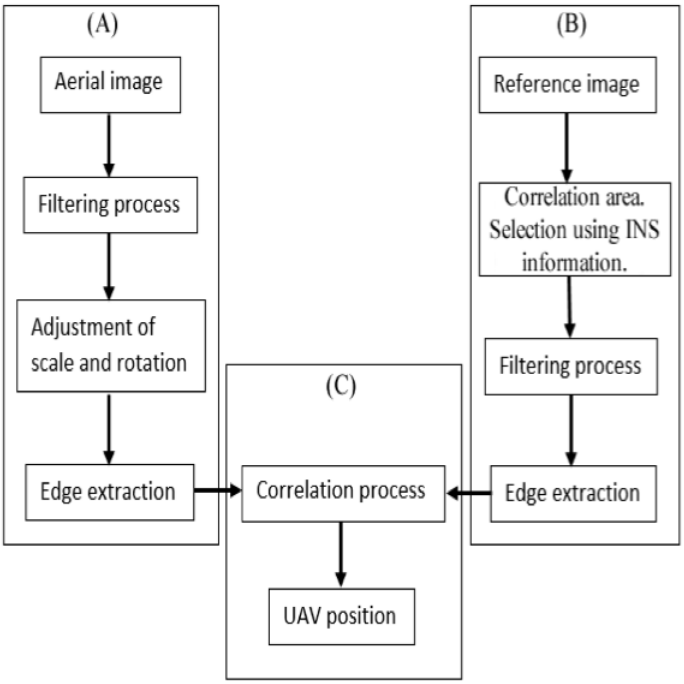


Figure 1: Sumarizing the computer vision UAV positioning system.

The MLP is a supervised feedforward neural network. The MLP-NN has an input layer – receiving data by the user –, one or more hidden layers – where artificial neurons are computational units –, and one output layer – this last layer can or cannot be composed of computational

neurons. The learning process for the MLP-NN has two phases. In the first phase, the input information is propagated up to the output layer. The error between the target values – used for training – and the neural network output is computed and back-propagated for updating the weight values: this is the *back-propagation algorithm* for identifying the neuron connection weights (Haykin, 1998). The binary patterns for edge and non-edge for a 3×3 pixel window is shown in Figure 2.

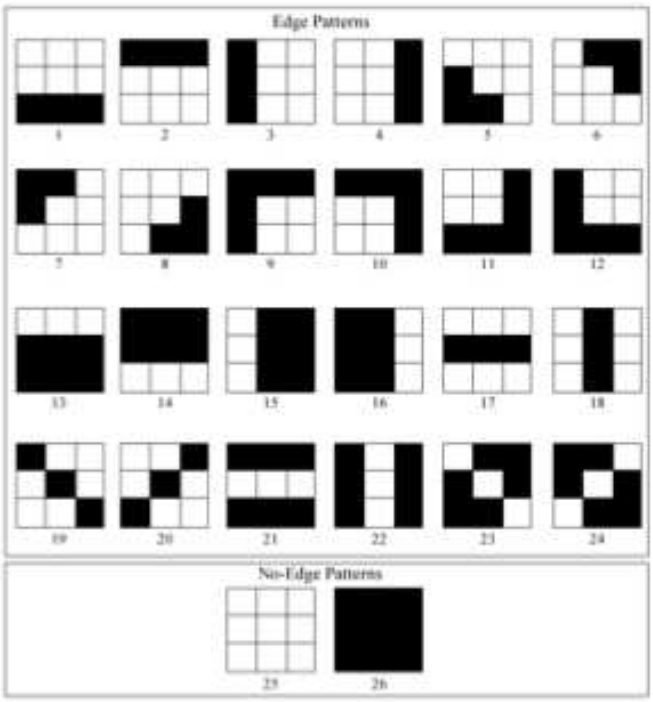


Figure 2: Edge and non-edge patterns for MLP-NN learning phase.

It is not easy to find a good architecture for the ANN. The best architecture to the supervised neural network can be found by minimizing a functional (Anochi and Campos Velho, 2014) by meta-heuristic methods such as Multi-Particle Collision Algorithm (MPCA) (Luz et al., 2000). The MPCA is a meta-heuristic inpsired from phenomena occuring inside of a nuclear reactor. During a neutron traveling, it can have a collision with an atomic nucleus. After collision, the neutron can be absorbed or scattered. In the MPCA formalism, a set of candidate solutions are generated for each iteration, where the *absorption* means a better generated solution will be adopted, otherwise the *scattering* operator is activated – the candidate solution will be searching in a distant location in the search space.

2.3 Neural network implemented on FPGA

During the development of the computer vision (CV) strategy for UAV positioning, the greater computational effort was verified for the edge identification procedure with MLP-NN. Other edge extractors can also be used, such as Sobel or Canny algorithms (Davies, 1990), but the two cited algorithms for edge extraction produce less precise UAV position. However, Sobel and Canny's operators performed 4 and 5 UAV position estimation per second, respectively. The edge extraction by MLP-NN with optimal topology perform only one UAV position estimation at each 6 seconds.

The MLP-NN produced a more precise UAV position estimation, but it was the slowest algorithm among the studied edge extractors. The artificial neural networks can be implemented in software or on hardware (Omondi and Rajapakse, 2006). One idea to speed up the edge extraction by MLP-NN was to use a co-processor – FPGA (Field Programmable Gate Array).

There are 512 binary patterns considering the mentioned 3×3 (pixel matrix) window for edge and non-edge – see Figure 2. Therefore, the results for edge detection by MLP-NN were codified using *Look Up Table* (LUT) strategy on FPGA for enhancing the computational performance. The LUT has a binary index for the inputs ranging to zero up to 511. The output value from the LUT can be "0" (non-edge) or "1" (edge).

The hardware consisted of a computer RaspBerry Pi Model B-1 (processador ARM1176JZF-S: single core 32-bits, 700 MHz, memory 512MB SDRAM (Synchronous Dynamic Random Access Memory), 2 USB (Universal Serial Bus) gates, video gate HDMI (High-Definition Multimedia Interface), GPU (Graphics Processing Unit) Broadcom VideoCore IV 250 MHz, and LOGI PI board with FPGA Xilinx Spartan 6 LX9. The link between the RaspBerry CPU and the LOGI PI board is performed by *Serial Peripheral Interface* (SPI), a protocol developed for communication of microprocessors and connected devices. The RaspBerry computer and LOGI PI board are shown in Figure 2.3

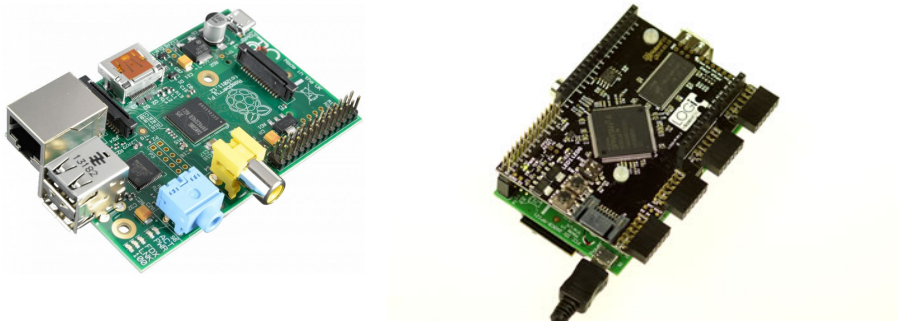


Figure 3: Right: Raspbeery Pi computer. Left: LOGI PI board with FPGA embedded.

3 DATA FUSION BY NON-EXTENSIVE PARTICLE FILTER

Particle filters or sequential Monte Carlo methods can be used to estimate the probability density function (PDF) by sampling some candidate solutions – *particles* – with associated weights (Gordon et al., 1993). The estimation with *Particle Filter* (PF) computes a posterior distribution from resampling the ensemble obtained by multiplying likelihood operator and the prior distribution. As the PF does not require assumptions of linearity or Gaussianity, it is applicable to general nonlinear problems.

Two important properties are verified by the PF: the Bayes' rule, and the Markov property. The Bayes' rule for conditional probability gives:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \,. \tag{3}$$

The probability $P(B)$ can be considered as a normalization factor. The Markov's process then is identified by:

$$p(w_n|w_{n-1}, \ldots, w_2, w_1) = p(w_n|w_{n-1}) \,. \tag{4}$$

For the current application, the distribution w_n represents the vector with the entries being the distributions associated to the UAV position estimations by VO and CV: the data fusion. The algorithm for the PF implementation can be summarized below.

1. Define the prior distribution:

$$\left\{ w_{0|n-1}^{(i)} \right\}_{i=0}^{N} \sim p_{w_0}(w_0)$$

here, the initial distribution is a Gaussian one, with zero mean and $\sigma^2 = 5$: $p_{w_0}(w_0) = \mathcal{N}(0, 5)$);

2. Calculate:

$$r_n^{(i)} = p(y_n | w_{n|n-1}) = p_{\text{et}}(y_n - h(w_n, t_n))\, p_{w_n}(w_n)$$

where y_n express observations, being $h(.)$ the observation operator, and p_{et} representing the likelihood operator:

$$p_{\text{et}}(z) = \begin{cases} \mathcal{N}_z(0,1) & \text{(Gaussian distribution, commonly applied)} \\ \mathcal{T}_z(0,\sigma_q) & \text{(Tsallis' distribution, applying NEx-PF)} \end{cases}$$

the standard deviation σ_q is defined in Section 3.1, and the inovation z is computed by:

$$z = y_n - h(w_n, t_n)\ .$$

3. Normalization:

$$\tilde{r}_n^{(i)} = \frac{r_n^{(i)}}{\sum_{j=1}^{M} r_n^{(j)}}\ ;$$

4. Resampling: remove M particles with low probability:

$$\text{If}: \ \Pr\{w_{n|n}^{(i)} = w_{n|n-1}^{(j)}\} = \tilde{q}_n^{(j)} \leq w^{\text{Ref}}, \ \ i = 1, \ldots, M\ ;$$

where w^{Ref} is a reference probability, with $M < N$.
Resampling:

- Generate M ordered numbers: $u_k = M^{-1}\left[(k-1) + \tilde{u}\right]$, with: $\tilde{u} \sim U(0,1)$ (uniform distribution),
- Resampled particles are obtained by producing m_i copies of particle $w^{(j)}$:

$$\begin{aligned} m_j \ &= \ \text{number of} \ \ u_k \\ & \qquad u_k \in \left[\sum_{s=1}^{j-1} \tilde{r}_n^{(s)}, \sum_{s=1}^{j} \tilde{r}_n^{(s)} \right] \end{aligned}$$

5. Time up-dating: compute the new particles:

$$w_{n+1|n}^{(i)} = f(w_{n|n}^{(i)}, t_n) + \mu_n\ , \quad \text{with:} \ \ \mu_n \in \mathcal{N}_z(0,1)$$

being $f(.)$ the dynamical system, and: $w_{n+1|n}^{(i)} \sim p(w_{n+1|n}^{(i)} | w_{n|n}^{(i)})$, being $i = 1, 2, \ldots, N$;

6. Set: $t_{n+1} = t_n + \Delta t$, and go to step-2.

3.1 Non-extensive Likelihood Operator

A generalized thermostatistics has been proposed by C. Tsallis (Tsallis, 1988) by introducing the non-extensive entropy:

$$S_q = \frac{k}{q-1}\left[1 - \sum_{i=1}^{N} p_i^q\right] \tag{5}$$

where p_i is a probability of the state-i, and q is the *non-extensivity* parameter. In thermodynamics, the parameter k is called the Boltzmann's constant. Tsallis' entropy reduces to the the usual Boltzmann-Gibbs-Shanon formula in the limit $q \to 1$.

Similar to the extensive entropy, the equiprobability condition implies in a maximum value for the non-extensive entropy function, leading to a special type of distributions (Tsallis, 1999). Three expressions for the Tsallis' distribution are described below.

$\mathbf{q > 1}$:

$$p_q(x) = \alpha_q^+ \left[1 - \frac{1-q}{3-q}\left(\frac{x}{\sigma}\right)^2\right]^{-1/(q-1)} \tag{6}$$

$\mathbf{q = 1}$:

$$p_q(x) = \frac{1}{\sigma}\left[\frac{1}{2\pi}\right]^{1/2} e^{-(x/\sigma)^2/2} \tag{7}$$

$\mathbf{q < 1}$:

$$p_q(x) = \alpha_q^- \left[1 - \frac{1-q}{3-q}\left(\frac{x}{\sigma}\right)^2\right]^{1/(q-1)} \tag{8}$$

where $\alpha_q^{\pm}$ are given by:

$$\begin{aligned}
\alpha_q^+ &= \frac{1}{\sigma}\left[\frac{q-1}{\pi(3-q)}\right]^{1/2} \frac{\Gamma\left(\frac{1}{q-1}\right)}{\Gamma\left(\frac{3-q}{2(q-1)}\right)}, \\
\alpha_q^- &= \frac{1}{\sigma}\left[\frac{1-q}{\pi(3-q)}\right]^{1/2} \frac{\Gamma\left(\frac{5-3q}{2(1-q)}\right)}{\Gamma\left(\frac{2-q}{1-q}\right)}, \\
\sigma^2 &= \frac{\int_{-\infty}^{+\infty} x^2 [p_q(x)]^q dx}{\int_{-\infty}^{+\infty} [p_q(x)]^q dx}.
\end{aligned}$$

The distributions above applies if $|x| < \sigma[(3-q)/(1-q)]^{1/2}$, otherwise $p_q(x) = 0$. For distribitions with $q < 5/3$, the standard central limit theorem applies, implying that if p_i is written as a sum of M random independent variables, when $M \to \infty$, the probability density function for p_i is the normal (Gaussian) distribution. However, for $5/3 < q < 3$ the Levy-Gnedenko's central limit theorem applies, resulting for $M \to \infty$ in the Lévy distribution as the probability density function for the random variable p_i. The index in such Lévy distribution is $\alpha = (3-q)/(q-1)$ (Tsallis, 1999).

4 NUMERICAL RESULTS

The UAV images were obatined by helicopter RMAX (Yamaha Motor Company), used for testing in the Linköpin University (Sweden) – see Figure 4. The helicopter flew with average speed of 3 ms^{-1} and about 60 m over the surface (altitude). The UAV camera captures

image from the Nadir with frequency of 25 Hz, resolution of 0.12 m/pixel with 288×360 pixels — pixel represents an area ~ 1540 m^2. The flight extention was about 1 km, and 1443 images/points were catched during the experiment.

Figure 4: Drone: Helicopter RMAX Yamaha.

The SURF algorithm was employed to identify the interest points in the visual odometry, and the RANSAC (RANdom SAmple Consensus) (Fischler and Bolles, 1961) was used for removing false corresponding points. A supervised neural network was applied for edge identification from the objects in the scene, according to the patterns shown in Section 2.2. Table 1 shows the MLP-NN architecture determined by the MPCA meta-heuristic method.

MLP-NN features	parameter
Neurons in the input layer	9
Neurons in the output layer	1
Number of hidden layers	1
Neurons in the hidden layer	18
Activation function	tanh
Rate of momentum	0.85
Learning rate	0.73

Table 1: MLP-NN architecture computed by the MPCA meta-heuristic.

The neural network described in Table 1 was implemented in software (CPU: RaspBerry Pi) and on hardware (FPGA: Xilinx Spartan-6). The FPGA is imbedded in the LOGI PI board connected to the RaspBerry Pi. The time for execution for different algorithms for edge extraction is shown in Table 2. The FPGA implementation was able to reduce the computational effort for the MLP-NN processing. However, the execution time using MLP-NN by FPGA is more than 7 times slower than the Canny algorithm implemented in software (CPU). Indeed, the processing time inside the FPGA is much faster than the time shown in Table 2, almost the whole time is spended moving data from the CPU-FPGA and returning the result FPGA-CPU.

Algorithm	Time (seconds)
Sobel – CPU	0.083
Canny – CPU	0.074
Optimal MLP-NN – CPU	1.684
MLP-NN by LUT – FPGA	0.587

Table 2: Execution time for edge detection algorithms in the Raspberry PI with LOGI PI (FPGA).

The data fusion for the UAV positioning combining the visual odometry and computer vision was carried out by the nox-extensive particle filter. A parameteric study was done to identify the best value for the non-extensivity parameter q. Numerical simulations were executed with 100 sampled values in the interval $q \in [0, 3]$. The initial set of 1000 particles was worked with random values from a Gaussian distribution with zero mean and variance equal to one ($\mathcal{N}(0, 1)$). Our numerical simulations indicates $q = 2.57$ (Braga et al., 2018).

Table 3 shows the average error for the experimental flight. It is clear the improvement by using data fusion. However, the application of NExt-PF for data fusion with non-extensivity parameter $q = 2.57$ produced a better result than standard particle filter. Our results show the impact of the likelihood operator in the final UAV position estimation. The NExt-PF was more precise than standard PF and/or considering visual odometry or computer vision individually.

Method	Average error
VO	6.7755
CV	4.4458
NExt-PF: $q = 1.00$	3.5313
NExt-PF: $q = 2.57$	2.8339

Table 3: Average error for *drone* positioning using different methods for image edge extraction.

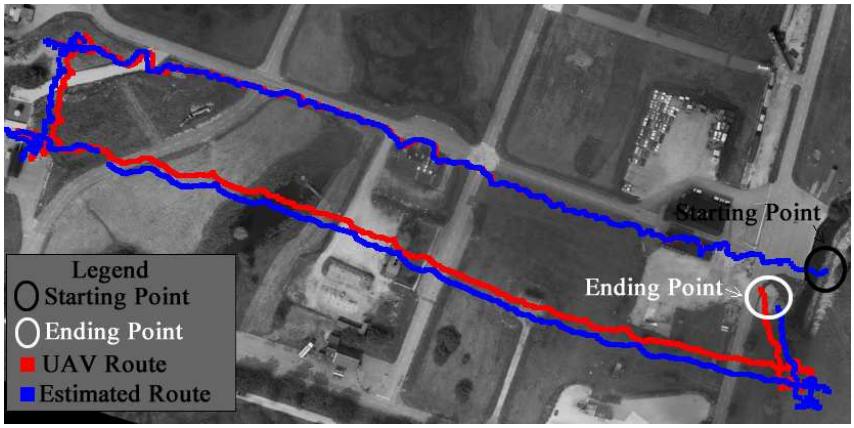
The trajectory executed by the UAV is shown in red color in Figures 4a – showing the trajectory estimation by visual odomatry (VO) – and 4b – displaying the the trajectory estimation by non-extensive entropy with $q = 2.57$. The starting point is marked with black circle, and the ending point is marked with white circle. Clearly, the VO strategy shows a drift error (cumulative error), and the trajectory correction with NExt-PF is a better strategy.

5 FINAL REMARKS

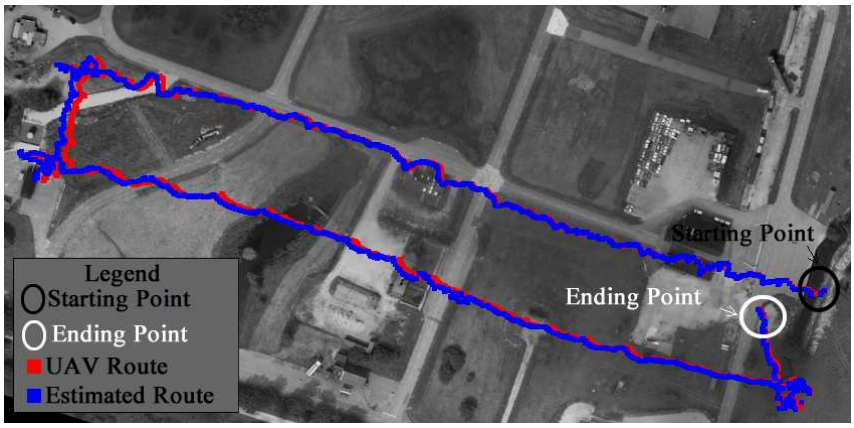
Image processing was employed for *drone* autonomous navigation. Visual odometry and computer vision were implemented and a data fusion approach combining the two techniques was applied using the non-extensive particle filter. The use of particle filters allows one to compute the *confidence interval* (Braga et al., 2018), by calculating the uncertainties associated to the present estimation problem.

Our computer vision implementation requires edge identification for convoluting reference and UAV segmented images. The edge identification has influence to determine the UAV position – see Table 3. A better positioning estimation was obtained with data fusion using optimal neural network for edge extraction. However, the neural extractor was the slowest procedure on software implementation.

The neuro-computer implementing the optimal MLP-NN was carried out on FPGA. The FPGA processing was effective to reduce the CPU processing time about 35%. But, the total FPGA execution time, including *processing time* and *data transfer time*, is greater than process-



(a)



(b)

Figure 5: Drone true (red color) and estimated (blue color) trajectories with: (a) estimation by VO, (b) estimation by NExt-PF with $q = 5.47$.

ing time spent by less precise Sobel or Canny algorithms. Our result shows the direction for the technology to be employed, by using a hardware encapsulating CPU and FPGA in the same chip, as found in the ZYBO ZYNQ 7000 system[1].

ACKNOWLEDGEMENTS

The authors acknowledges CNPq, Capes, and Fapesp, Brazilian research support agencies.

REFERENCES

Anochi J. and Campos Velho H.C. Optimization of feedforward neural network by multiple particle collision algorithm. In *FOCI IEEE Symposium on Foundations of Computational Intelligence*, pages 128–134. 2014.

Anochi J., Campos Velho H.F., Furtado H.C.M., and Pacheco E.L. Self-configuring two types of neural networks by mpca. *Journal of Mechanics Engineering and Automation*, 5:23–36, 2015.

Bay H., Ess A., Tuytelaars T., and Gool L.V. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110:346–359, 2008.

Braga J.R.G., Campos Velho H.F., and Shiguemori E.H. Determining the trajectory of un-

[1] https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf

manned aerial vehicles by a novel approach for the particle filter. *Mecanica Computacional*, 36:683–692, 2018.

Chamoso P., Raveane W., Parra V., and A. G. *UAVs applied to the counting and monitoring of animals. Ambient Intelligence-Software and Applications*. (Eds: Ramos C., Novais P., Nihan C., Corchado Rodríguez: Advances in Intelligent Systems and Computing), 2014.

Conte G. and Doherty P. Vision-based unmanned aerial vehicle navigation using geo-referenced information. *EURASIP Journal on Advances in Signal Processing*, 1:308–387, 2009.

Davies E. *Machine Vision: Theory, Algorithms and Practicalities*. Academic Press, 1990.

Fischler M.A. and Bolles R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24:381–395, 1961.

Gonzalez R.C. and Woods R.E. *Digital Image Processing*. Pearson, 4 edition, 2017.

Gordon N.J., Salmond D.J., and Smith A.F.M. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F (Radar and Signal Processing*, 140:107–113, 1993.

Haykin S. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1998.

Luz E.F.P., Becceneri J.C., and Campos Velho H.F. A new multiparticle collision algorithm for optimization in a high performance environment. *Journal of Computational Interdisciplinary Sciences*, 1:03–09, 2000.

Messinger M. and Silman M. Unmanned aerial vehicles for the assessment and monitoring of environmental contamination: an example from coal ash spills. *Environmental Pollution*, 218:889–899, 2016.

Motlagh N.H., Bagaa M., and T. T. Uav-based iot platform: a crowd surveillance use case. *IEEE Communications Magazine*, 55:128–134, 2017.

Naranjo J.E., Clavijo M., Jimenez F., Gomen O., Rivera J.L., and Anquita M. Autonomous vehicle for surveillance missions in off-road environment. In *IEEE Intelligent Vehicles Symposium - Vol. IV*, pages 98–103. 2016.

Nister D., Naroditsky O., and Bergen J. Visual odometry. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. 2004.

Omondi A.R. and Rajapakse J.C. *FPGA implementations of neural networks*. Springer, 2006.

Psirofonia P., Samaritakis V., Eliopoulos P., and Potamitis I. Use of unmanned aerial vehicles for agricultural applications with emphasis on crop protection: three novel case-studies. *Journal of Agricultural Science and Technology*, 5:30–39, 2017.

Scaramuzza D. and Friedrich F. Visual odometry part I: The first 30 years and fundamentals. *EEE Robotics and Automation Magazine*, 18:80–92, 2011.

Tsallis C. Possible generalization of boltzmann-gibbs statistics. *Journal of Statistical Physics*, 52:479–487, 1988.

Tsallis C. Nonadditive entropy and nonextensive statistical mechanics - an overview after 20 years. *Brazilian Journal of Physics*, 39:337–356, 1999.