Ground Vehicle Detection Through Aerial Images Taken by a UAV

Alberto Pérez, Pablo Chamoso, Víctor Parra, Antonio Juan Sánchez
Computer and Automation Control Department, Faculty of Science
University of Salamanca
Salamanca, Spain
{alberto.pgarcia, chamoso, parra, anto}@usal.es

***Abstract*—Advantages in the application of intelligent approaches, such as the conjunction of artificial vision and Unmanned Aerial Vehicles (UAV), have been recently emerging. This paper presents a system capable of detecting ground vehicles through aerial images taken by a UAV in real time. In addition, the system offers the possibility to autonomously guide the UAV to keep track of a vehicle that has been previously detected.**

Keywords—Unmanned Aerial Vehicle; Convolutional Neural Networks; Vehicle detection; Vehicle tracking; Multi-Agent Systems

I. INTRODUCTION

UAV systems offer an increasing and varied number of possibilities for applications in the professional world. The importance of this increase, in addition to the evolution of technology and the reduction of costs, is due to the emergence of multi-rotor systems, which offer high performance when capturing aerial images due to their high stability when making motions in either axis of space.

Consequently, multi-rotors offer suitable characteristics to address the issue proposed in this case study, which consists of identifying vehicles from the onboard camera to track them in autonomously.

To achieve these two objectives, tracking and visual recognition and detection techniques are applied in combination with UAV control. These will be discussed throughout this article.

The most relevant functionality the system offers consists of the possibility of visually identifying potential targets (cars). As with any visual analysis application, information processing in (almost) real time is a difficult problem to solve due to the large size of the input data. Numerical representation of a picture will vary considerably depending on subtle changes in illumination, camera conditions or many other factors.

Also, shadows cast by the objectives and the fact that both the cars and the multi-rotor are in motion, complicate the task. These conditions may be overwhelming for most traditional image processing techniques, such as Chan Vese segmentation[9], which has excelled in other image separation tasks; however, it is not applicable to the type of issue to be addressed in this case study.

ANNs have been used on numerous occasions to solve many real-life problems in conjunction with MAS. In [15][14][13][16] [5][17][7][11][12], the reader can have a quick look of the use of different types of neural networks for achieving real-life goals.

Convolutional Neural Networks (CNN)[8][18]were introduced as a general solution to image recognition problems with variable inputs, such as the presented case. A CNN consists of a multi-layer Artificial Neural Network (ANN) with a built-in feature extraction process and translational tolerance of the input image space. Therefore, this type of network is capable of accurately identifying images of a target object among cluttered background noise. This is achieved by learning the distinctive features that characterize the class the object belongs to, regardless of the relative position at which it appears in the input image sample.

Once the identification problem has been solved, the complex problem of tracking one of the vehicles already identified is presented. To do this, tracking techniques are used beginning with the image analysis previously performed. The vehicle trajectory is predicted and, when the movement of the target is known, it is translated to the multi-rotor using the designed autonomous navigation system.

This article is structured as follows: the next section describes the general background of vehicle detection and tracking, multi-rotors and CNN. The third section is a system overview where the system components and the visual recognition techniques used are described in detail. Finally, the fourth section presents the results and conclusions obtained.

II. BACKGROUND

There are three converging areas in this study: vehicles detection and tracking, multi-rotor systems such as hardware tools used as part of the work, and image processing techniques used for carrying out the analysis of the images obtained by the UAV, in this case Convolutional Neural Network technique. This section describes the current state of the art of each technique. The next section explains how they have been used in this study.

A. Vehicles detection and tracking

The problem of real time vehicle optical detection and tracking is not an emerging problem as it has existed for quite

some time. Many authors have undertaken studies for years [10], even addressing various multiple tracking vehicles [2].

However, these existing studies are based on images taken from almost the same height as the level of the target to be analyzed. Nowadays this detection can also be performed from an aerial vehicle that does not require a large expenditure to acquire an air vehicle, as discussed below.

The case study is interesting due to the functionality obtained by performing the analysis from an aerial vehicle, although the process itself is more complex.

This interest centers on the potential application of this study to multiple scenarios, such as the persecution of suspicious vehicles. Escorting a vehicle may also be useful for the aerial view of the land surrounding it, since this will be a much broader view than that obtained from the ground.

Computationally, the main difficulty of the task of tracking vehicles lies in the detection phase. This is because once they have been detected, tracking them is a relatively simple task by comparison.

At the execution level, the case study may involve certain risks produced by external elements to the system, as might be the case of wiring, tunnels, bridges or trees. For this reason, the scenario in which the system was tested was very specifically selected.

B. *Multi-rotor systems such as hardware tools used as part of the work*

The commercial availability of multi-rotors allows the task to be carried out simply and somewhat cheaply. This type of UAV allows the inclusion of an air camera that, thanks to the stabilization system, collects images of high quality and low distortion.

The possibilities of applying multi-rotors in the professional world have increased in recent years because of these advances in technology, especially aerial imaging.

Aerial imaging provides detailed images and quick monitoring of areas, making it a very efficient way to solve the problem proposed in this study.

In addition, these advances also provide high processing capability when processing navigation algorithms for the UAV to move completely autonomously.

The electronic component that governs the processing of all the information available to the multi-rotor and turns it into movement has evolved greatly. This case study uses Single Board Computers (SBC), equipped with several cores to process simultaneous tasks at high speed. It is also possible to connect all kind of devices in order to add functionality to the system, as discussed in later sections.

As a result, the multi-rotor is able to make decisions about its own motions (in either axis) on the fly. This will allow the tracking task to be relatively simple.

C. *Convolutional Neural Network*

The layer architecture for the CNN used will vary according to the application. However, the count of output neurons in the final layer will always be associated with two output classes: one for the object of interest, and the other one for background noise. Hence, every time the network is executed over an image patch, it will output two values, each of which will be interpreted as the confidence level with which the network believes the corresponding class correctly describes the analyzed sample.

The CNN is trained with data manually collected from previous multi-rotor test flights and later, artificially augmented. The augmentation consists of increasing the number of training data available by applying a series of transformations to translate, rotate and scale each manually collected example. Such a process can yield up to 40 artificial samples for every original image patch, thus giving the network a lot more data to train with. The training data is separated into two sets for each of the classes, target (vehicles) and background (mainly road).

The network is then trained with the prepared data sets through the stochastic gradient descent method for back-propagation [3], which offers an optimal route to minimizing the classification error for this type of highly mutable training data

III. SYSTEM OVERVIEW

A. *Multi-rotor system*

The multi-rotor architecture to obtain images consists of three main parts: hardware, software and communication protocol.

The hardware part refers to the system running the software and is present at both ends of the communication.

At one end is the UAV, a multi-rotor with 6 engines and 6 arms bearing an SBC with a700MHz processor and 512Gb of Random Access Memory (RAM), capable of running a Linux distribution. The connectivity of the SBC allows connecting a camera via Ethernet and a Wi-Fi antenna via Universal Serial Bus (USB); it also gets the information that Speed Electronic Controllers (ESCs) need to control the multi-rotor motors.

The IP camera remains fixed in the structure of the multi-rotor, without pan/tilt motions, and its zoom is not used. It offers a resolution of 1280X720 pixels and allows obtaining its frames through a CGI (Common Gateway Interface) stream. It also has night vision, but this mode has not yet been used, so the case study focuses only on lighted conditions.

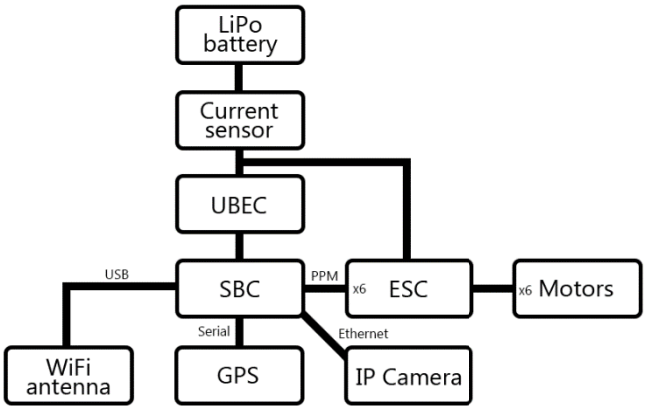

Fig. 1. Schema of the electronic components

At the other end of the communication there is an access point whose potency depends on the distance to be achieved; it can cover distances of up to 3 kilometers. This access point will connect the SBC UAV control computer with a high performance laptop with a USB gamepad connected for manual control or autonomous flight (waypoints, tracking).

With regard to the software, SBC runs a piece of software that reads, processes and delivers telemetry. It is responsible for carrying out the calculations for the behavior and stability of the multi-rotor.

The computer runs a piece of software specifically developed (in Java) to control the multi-rotor both manually and autonomously.

It can be controlled manually by using a gamepad as the input device. Moreover, the automatic flight mode allows different options: to establish a set of points that the multi-rotor processes in the configured manner (speed, height, etc.); to determine a perimeter entering its points and tour the area enclosed by this perimeter so that the shortest route is automatically traced for the area to be covered according to the settings; and to indicate on the image the vehicle to follow and try to follow it.

That software, therefore, is also able to detect vehicles in real time, based on the processing of the image captured by the camera and the application of CNN techniques described throughout this article. Thus, the detection and tracking process is isolated from the stabilization process performed by the SBC from the multi-rotor itself.

The user can see all the information from the sensors that the multi-rotor sends in real time, such as information relating to power consumption, intensity and quality of the Wi-Fi signal, image and video (Fig. 3). This information will generate log files in XML format that will serve to make a more accurate and detailed analysis of the captured images, as it will provide additional useful information such as height and global position.

B. Multi-Agent System

The use of Multi-Agent Systems (MAS) provides a mechanism that allows individual units called agents to perform tasks concurrently. These agents can be software units that undertake simple tasks to reach a common goal of the overall system.

In the multi-rotors used for this case study, the use of multi-agent systems provides a number of important advantages. First, it isolates the task of multi-rotor stabilization with respect to the tasks relating to the calculation of the power with which each motor rotates (associated to the desired motion). It also implies a decentralization of the complete system, so trajectory calculations tasks, based on a complex analysis such as the vehicle detection being treated, can be performed in the control computer with much greater processing capability than the multi-rotor SBC. Finally, it provides great modularity and scalability, which makes it possible to add new functionality in a single multi-rotor or even the use of several multi-rotors in the system with one single control computer.

In addition, the MAS provides a communication system that allows each of the agents to exchange information following the protocol defined by the platform.

There are multiple platforms when making a practical application related to these MAS, such as JADE (Java Agent DEvelopment framework)[1] or PANGEA (Platform for Automatic Construction of Organizations of Intelligent Agents) [23] [4][6][21][22]. The latter has been used for the deployment and communication of agents that will perform the full functionality needed for the case study.

PANGEA is based on the IRC protocol. With the use of this platform, described in [24], the information transmitted in PANGEA is also encapsulated in Mavlink [19]communication protocol based frames. This protocol has been modified to optimize their headers (since there is redundant information to be enclosed within the IRC command). Its packages are defined as indicated by the following structure:

- **Payload length** (1 byte): first byte of the packet. It is the length of the Payload field. Range: from 0 to 255.
- **Packet sequence** (1 byte): second byte of the packet. It represents the count of the sequence. The range goes from 0 to 255, starting at 0 and restarting it to 0 again after it reaches 255.
- **Message ID** (1 byte): third byte of the packet. This is the identification of the message type. It determines the format of the payload. The range is from 0 to 255 and there are 180 messages, so 10 new messages can be defined.
- **Payload** (n bytes): from the fourth byte to the byte n+4 of the packet. Here the information data to be transmitted is codified by following the format that the message ID field defines.

- **CRC** (2 bytes): the last two bytes of the packet, after the first n+4 bytes. They contain a check-sum of the packet to avoid errors.

In addition, over 18 new types of communication messages (field 'Message ID') have been defined in the Mavlink protocol, optimizing to the bit level each of the bytes encapsulating the Payload field.

C. Runtime operation

Object recognition is carried out at runtime by analysing individual frames received from an on-board camera. A sliding window approach is taken to sequentially analyse small, adjacent, and overlapping image patches positioned in a grid pattern over the frame. The trained network evaluates each of these image patches and the output values are recorded.

This process is repeated at three scales. The middle scale is chosen to approximate the relative size at which the target objects are expected to be seen – a value which can be calculated trigonometrically based on the current flight altitude of the UAV. The two remaining scales are set at 85% and 115% of the middle scale window size. These additional scales provide supplementary information, which later helps to boost the readings obtained at each grid coordinate.

For every window analyzed, the output values of the CNN are processed with the softmax activation function described in Equation 1, where y_i is the resulting value of the network for output neuron i. This transformation results in a probability-like value $P(i)$, which for any given class i estimates the likelihood that the analyzed window belongs to it.

$$P(i) \equiv \sigma(y, i) = \frac{e^{y_i}}{\sum_{n=0}^{N} e^{y_n}} \quad (1)$$

Equation 2 defines L_{xys} as the likelihood value that the analyzed window at grid coordinate *(x;y)* and scale s belongs to the target class. Combining these values over the entire grid for every x, y and s will lead to a discrete 2D probability distribution over the original input frame, which will indicate the positions at which a target object has been detected.

Boosting the values at each coordinate through a simple nearest-neighbor clustering algorithm, where the likelihood value for each point in the grid is increased by strong adjacent readings, can further enhance this distribution. This process is detailed in Equation 3.

$$L_{xys} \equiv P(target|x, y, s) \quad (2)$$

$$B_{xy} = \sum_{i=x-1}^{x+1} \sum_{j=y-1}^{y+1} \sum_{s=0}^{2} \begin{cases} L_{ijs} & if\ L_{ijs} \geq 1/2 \\ 0 & if\ L_{ijs} < 1/2 \end{cases} \quad (3)$$

The boosted value B_{xy} can then pass through a threshold, determined ad hoc for the particular application, to finally produce a quantized representation of the probability distribution.

An additional benefit of boosting involves the elimination of most false positives, occurrences from which neural networks are never exempt, but which, fortunately, often tend to appear without any neighboring support, thus disappearing after applying this process. Likewise, the use of this mechanism will make the system more tolerant towards false negatives, as these will not have such a significant impact on the final result since the network often finds reinforcing values at adjacent positions.

D. Visual recognition

For this application, the network utilized follows an architecture given by 64x64-18C7-MP4-96C5-MP2-4800L-2, wherein there are two convolutional feature extraction stages and one hidden linear layer. The training data is prepared as previously described, with the target class consisting of individual vehicle samples. A small subset of this data set can be seen in Fig. 2.

Fig. 2.A subset of the training data used for the CNN, where the dataset is divided into two classes: background (first and second rows) and target (third and fourth rows)

E. Vehicles tracking

Vehicle tracking is the next problem to be addressed. Once the target is detected in the image, an analysis is made of the trajectory that it has traced, in an attempt to simulate the motion of the multi-rotor itself.

This multi-rotor motion is performed only in the longitudinal and vertical axes. Displacements in the transverse axis produced by the vehicle are counteracted using a combination of yaw (rotation over the vertical axis) and pitch (forward in the longitudinal axis). This enables the camera to point in the direction of motion, which is required to follow the vehicle.

These motions are translated into values in a range from 1000 to 2000 and are associated with each of the possible motions that can be performed with the multi-rotor: roll, pitch, yaw and throttle. The scale of associated values corresponds to the most common values of the remote control obtained from analog signals depending on the duration of the pulses. Along these lines, the value 1500 is associated with the absence of motion, associating each of the two sub-intervals into which the interval is divided as follows: interval (1000, 1499) is associated with a direction of the axis. For example, in the case of throttle, the direction is downwards. Interval (1501, 2000) is associated with the opposite direction (upwards for the same

example). In both directions, the farthest value to the absence of motion (1500) results in stronger or fast motions, whereas as the value approaches the other ends (close to 1500), it results in softer or slow motions.

Following this proposed scheme, the multi-rotor will perform motions trying to focus at all times on the vehicle marked as a target in the captured image. To this end, the image size is divided by associating a one thousand-subdivisions interval for both image height and image width. This corresponds to the number of values enclosed by the range of motion values 1000 to 2000. With this division and contrasting analyzed trajectories with the corresponding motion, it is possible to obtain the value of motion to perform and send it to the multi-rotor, which will then translate it to the corresponding rotational speed of the engines to trace that motion and keep it in a stable position.

Calculations are repeated at a frequency high enough to produce motions with certain fluency and to make sure the multi-rotor does not move abruptly due to the completion of major corrections. This frequency is set to 5 times per second, releasing the machine from the heavy detection calculation, even though it would have been optimal, at least, 8 times per second.

IV. Results and Conclusions

The work was successfully tested in controlled scenarios. The neural network was also trained to test the reliability of the system. It is therefore necessary to test the entire system in scenarios with the external elements under control (as with these elements, the viability of the system is severely affected) in order to continue testing its efficiency.

Moreover, the tracking algorithm works well for vehicle speeds below 50kmh due to limitations, both physical (the multi-rotor is not able to acquire the required speed in non-ideal conditions, because of, for example, unexpected gusts of wind) and algorithmic, as the vehicle may disappear from the camera lens.

In those cases where the vehicle disappears from the image, during a sequence of frames associated with two seconds of time, the multi-rotor rises itself to cover a larger area with the camera lens. However, during those two seconds of uncertainty, the multi-rotor keeps the average motion from the corresponding trajectory with the frames of the last 6 seconds as a mechanism to keep following the vehicle.

In case of loss of the vehicle, the multi-rotor remains stable in its final position and enters manual pilot mode to await new orders. In the absence thereof, and if its battery was running low, it would return to a point marked as safe landing point.

Fig. 3. Screenshot of the developed software during a flight where the yellow car (in the white frame) is the objective to track.

As for vehicle identification system, it should be noted that one of the added benefits of boosting involves removing most false positives, occurrences from which neural networks are never exempt. Luckily, these occurrences often tend to appear without any neighboring support, thus disappearing after applying this process. Likewise, the use of this mechanism will make the system more tolerant towards false negatives, as these will not have such a significant impact on the final result since the network often finds reinforcing values at adjacent positions.

The CNN can be trained surprisingly fast, reaching a plateau for the training criterion (MSE on classification) after only a few SGD epochs. A summary of the CNN training results is given in Table I: Confusion matrix of the visual classification CNN trained with the augmented sample data over 10 SGD epochs.

TABLE I. Confusion matrix of the visual classification CNN trained with the augmented sample data over 10 SGD epochs.

Training Data Classification 12,000 Samples			
Class	Vehicle	Road	Error
Vehicle	**5,602**	398	93.4%
Road	193	**5,807**	96.8%
Global			**95.1%**
Testing Data Classification 3,000 Samples			
Class	Vehicle	Road	Error
Vehicle	**1,341**	159	89.4%
Road	62	**1,438**	95.9%
Global			**92.65%**

Despite the existence of up to 10% error in one of the situations, this error is calculated over a single frame, whereas during the performance of a tracking task it is possible to analyze up to 12 frames per second. For this reason, the error will be considerably reduced.

We expect this to be the case since the system provides some "memory" when performing the analysis. Moreover, it is known that in successive frames, the probability of finding the

target vehicle is much higher in the vicinity of its previous position. Although the above situations are taken into account, if one of the frames is not detected, the error does not accumulate and can be fixed in the next frame. Therefore, given the high speed at which the images are received, it will hardly influence the motion produced by the multi-rotor for the vehicle tracking.

ACKNOWLEDMENT

This work has been carried out by the project Sociedades Humano-Agente: Inmersión, Adaptación y Simulación. TIN2012-36586-C03-03. Ministerio de Economía y Competitividad (Spain). Project co-financed with FEDER funds.

REFERENCES

[1] Bellifemine, F., Poggi, A., & Rimassa, G. (2001). Developing multi-agent systems with JADE. In Intelligent Agents VII Agent Theories Architectures and Languages (pp. 89-103). Springer Berlin Heidelberg.

[2] Betke, M., Haritaoglu, E., & Davis, L. S. (1996, September). Multiple vehicle detection and tracking in hard real-time. In Intelligent Vehicles Symposium, 1996., Proceedings of the 1996 IEEE (pp. 351-356). IEEE.

[3] Bottou, L.: Stochastic learning. In: O. Bousquet, U. von Luxburg (eds.) Advanced Lectures on Machine Learning, Lecture Notes in Artificial Intelligence, LNAI 3176, pp. 146–168. Springer Verlag, Berlin (2004)

[4] CI Pinzón, J Bajo, JF De Paz, JM Corchado. S-MAS: An adaptive hierarchical distributed multi-agent architecture for blocking malicious SOAP messages within Web Services environments. Expert Systems with Applications 38 (5), 5486-5499

[5] DI Tapia, A Abraham, JM Corchado, RS Alonso. Agents and ambient intelligence: case studies. Journal of Ambient Intelligence and Humanized Computing 1 (2), 85-93. 2010.

[6] DI Tapia, S Rodríguez, J Bajo, JM Corchado. FUSION@, a SOA-based multi-agent architecture. International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008), 99-107. 2008.

[7] DI Tapia, RS Alonso, JF De Paz, JM Corchado. Introducing a distributed architecture for heterogeneous wireless sensor networks. LNCS 5518, pp. 116-123. 2009.

[8] Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics 36(4), 193–202 (1980). DOI 10.1007/BF00344251

[9] Getreuer, P.: Chan-Vese Segmentation. Image Processing On Line 2012 (2012). DOI 10.5201/ipol.2012.g-cv

[10] Goerick, C., Noll, D., & Werner, M. (1996). Artificial neural networks in real-time car detection and tracking applications. Pattern Recognition Letters, 17(4), 335-343.

[11] J Bajo, JF De Paz, S Rodríguez, A González. Multi-agent system to monitor oceanic environments. Integrated Computer-Aided Engineering 17 (2), 131-144. 2010.

[12] JA Fraile, J Bajo, JM Corchado, A Abraham. Applying wearable solutions in dependent environments. Information Technology in Biomedicine, IEEE Transactions on 14 (6), 1459-1467. 2011.

[13] JF De Paz, S Rodríguez, J Bajo, JM Corchado. Mathematical model for dynamic case-based planning. International Journal of Computer Mathematics 86 (10-11), 1719-1730. 2009.

[14] JM Corchado Rodríguez. Redes Neuronales Artificiales: un enfoque práctico. Vigo : Servicio de Publicacións da Universidade de Vigo, 2000.

[15] JM Corchado, C Fyfe. Unsupervised neural method for temperature forecasting. Artificial Intelligence in Engineering 13 (4), 351-357. 1999.

[16] JM Corchado, J Bajo, JF De Paz, S Rodríguez. An execution time neural-CBR guidance assistant. Neurocomputing 72 (13), 2743-2753. 2009.

[17] JM Corchado, B Lees. Adaptation of cases for case based forecasting with neural network support. Soft computing in case based reasoning, 293-319. 2001.

[18] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to docu-ment recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998). DOI 10.1109/5.726791

[19] Meier, L., Camacho, J. F., Godbolt, B., Goppert, J., Heng, L., & Lizarraga, M. (2009). Mavlink: Micro air vehicle communication protocol.

[20] S Rodríguez, B Pérez-Lancho, JF De Paz, J Bajo, JM Corchado. Ovamah: Multiagent-based adaptive virtual organizations. Information Fusion, 2009. FUSION'09. 12th Interna-tional Conference on, 990-997. 2009.

[21] S Rodríguez, Y de Paz, J Bajo, JM Corchado. Social-based planning model for multiagent systems. Expert Systems with Applications 38 (10), 13005-13023. 2011.

[22] S Rodriguez, V Julián, J Bajo, C Carrascosa, V Botti, JM Corchado. Agent-based virtual organization architecture. Engineering Applications of Artificial Intelligence 24 (5), 895-910. 2011.

[23] Zato, C., Sanchez, A., Villarrubia, G., Rodriguez, S., Corchado, J. M., & Bajo, J. (2012, May). Platform for building large-scale agent-based systems. In 2012 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS), May (pp. 17-18).

[24] Zato, C., Villarrubia, G., Sánchez, A., Bajo, J., & Corchado, J. M. (2013). PANGEA: A New Platform for Developing Virtual Organizations of Agents. International Journal of Artificial Intelligence TM, 11(A13), pp.93-102