Visual Navigation for UAV with Map References Using ConvNets

Fidel Aznar(✉), Mar Pujol, and Ramón Rizo

Departamento de Ciencia de la Computación e Inteligencia Artificial,
Universidad de Alicante, San Vicente del Raspeig/Sant Vicent del Raspeig, Spain
{fidel,mar,rizo}@dccia.ua.es

Abstract. In this paper, a visual system for helping unmanned aerial vehicles navigation, designed with a convolutional neural network, is presented. This network is trained to match on-board captured images with several previously obtained global maps, generating actions given a known global control policy. This system can be used directly for navigation or filtered, combining it with other aircraft systems. Our model will be compared with a classical map registration application, using a Scale-Invariant Feature Transform (SIFT) key point extractor. The system will be trained and evaluated with real aerial images. The results obtained show the viability of the proposed system and demonstrate its performance.

1 Introduction

Unmanned aerial vehicle (UAV) navigation is an active research area. There are some situations, places and even devices, where visual perception is the best option for navigation. To develop visual navigation systems is complex because there are many factors that affect perception. However, given various assumptions this complexity can be reduced. In this paper we will assume that we have a prior record of recent images of the area to be overflown. Moreover, the path to be developed by the UAV is known given the UAV position and a global control policy. Thus, we firstly need the UAV to be able to locate itself on a previously obtained global map.

Most state of the art approaches rely on global localization based on visual matching between current view and available georeferenced satellite/aerial images [1–4] using feature detection. For example, in [2] geo-referenced is aided by Google Maps. Feature detectors and descriptors, that exploit the self-similarity of the images, are paired to establish the correspondence between the on-board image and the map. Subsequently, template matching using a sliding window approach is confined in the search region predicted by inter-frame motion obtained from optical flow. In [4] the matching is based on scale-invariant feature transform (SIFT) features and the system estimates the position of the UAV and its altitude on the base of the reference image.

This work has been supported by the Spanish Ministerio de Economia y Competitividad, project TIN2013-40982-R. Project co-financed with FEDER funds.

O. Luaces et al. (Eds.): CAEPIA 2016, LNAI 9868, pp. 13–22, 2016.
DOI: 10.1007/978-3-319-44636-3_2

After the incredible success of deep learning in the computer vision domain, there has been much interest in applying Convolutional Network (ConvNet) features in robotic fields such as visual navigation. There are several papers related to visual matching using convNets. For example, [5] shows how to learn directly from image data a general similarity function for comparing image patches. In [6] the effectiveness of convNets activation features for tasks requiring correspondence is studied. This paper claims that convNet features localize at a much finer scale than their receptive field sizes. They can be used to perform intra-class alignment as well as conventional hand-engineered features, and that they outperform conventional features in keypoint prediction.

This article describes a navigational aid system for UAVs based on the registration of perceptions on a previous map using convNets. A Convolutional Network will be trained to generate actions for every visual perception given a global motion plan. Our intention is to provide a useful navigation system for drones that can be combined or filtered with other aircraft systems.

The organization of the paper is as follows. Firstly, a global navigation policy for a specific map will be defined. Secondly, we will describe the system and the tests to be performed with real aerial maps. Next, we will introduce a convolutional network to develop visual navigation. Finally, we will validate our system with real aerial images and will compare it with a classical visual matching strategy related to [4].

2 Global Navigation Map

Our motion model is based on building a global motion plan, defined specifically for the task to be developed. For this purpose we have defined a potential function $U(\boldsymbol{x})$ so that the global navigation plan will be defined by its gradient. There are many studies and different alternatives to define potential functions that behave desirably as feedback motion plan. We use a potential function U quadratic with distance. This function allows us to calculate the potential field for a given space point $\boldsymbol{x}$. The system presented here does not depend on a specific potential function and will not be deeply discussed here.

$$U(\boldsymbol{x}) = \alpha \sum_{i=1}^{N} q_i \frac{\boldsymbol{x} - \boldsymbol{x_i}}{\|\boldsymbol{x} - \boldsymbol{x_i}\|},\ A(\boldsymbol{x}) = \cos^{-1}\left(\frac{-\nabla U(\boldsymbol{x}) \cdot \boldsymbol{a}}{\|\nabla U(\boldsymbol{x})\|\|\boldsymbol{a}\|}\right)$$

We are interested in developing an interface where the user could touch different areas adding repulsion or attraction forces. Therefore, we will use N particles. For each particle i we will define its intensity q_i (that can be attractive or repulsive) and position $\boldsymbol{x_i}$, where α is a normalization term. More specifically, we will focus on the angle of the gradient of this potential function, $A(\boldsymbol{x})$, as we are only interested in the direction of the aircraft. This direction is obtained through a reference vector $\boldsymbol{a}$.

In Fig. 1(a), an aerial image of the University of Alicante campus is presented. All the introduced particles for generating the potential field are shown, where

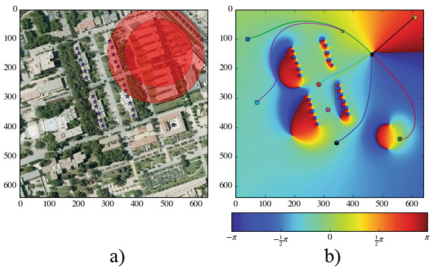

Fig. 1. (a) Aerial image of the University of Alicante campus with the attractive particles (circles) and repulsors (stars) (b) $A(\boldsymbol{x})$ function of the previous map, where colour represents the navigation angle in radians. Seven different routes developed with this map for different starting positions are also shown

circles are attractive forces and stars repulsive ones. The radius of the particle represents its intensity. In Fig. 1(b) the $A(\boldsymbol{x})$ function, calculated for the previous particles is presented. A simulation of movement using this map is also presented, for seven different starting positions, where a circle represents the initial position and a star the final one. For this simulation, we have iterated 500 times, adjusting the vehicle angle using $A(\boldsymbol{x})$ function with a translational velocity equal to one meter per iteration.

3 Test Design

For the development of our task we require the UAVs to be equipped with a compass and a barometer (both are very common sensors for even low cost drones). The usefulness of the barometer is to ensure a uniform altitude for capturing the images. This is a key advantage of this type of vehicle, because we can reduce and even eliminate the need to extract multi-scale features (they can fly at the same altitude as the map was obtained). Moreover, a compass is needed for registering the images in an independent point of view.

As was discussed above, the presented system allows us to have several points of attraction and repulsion placed on a global visual map. The perceptions of a UAV will be used to determine the action to be proposed by the system given a registration process (carried out internally by the network). Therefore, we could use the system in several ways: to create reactive sensors to avoid or direct the drone to different areas, such as the work presented in [7] or to provide support for visual navigation, as will be used in this paper.

To accomplish this task we need a visual global map. We will use five orthophotos of the same area of the campus, taken at different time of the day of different years. These photos are sufficiently different (different shadow areas, camera types, changes in vegetation...) to test the robustness of the system. The first four images (Fig. 2(a) will be used as global map for network training. One last image (Fig. 2(b) will be used as validation.

Fig. 2. (a) Different images of the same zone of the campus taken in several flights used for training. Capture years are (from left to right) 2002, 2005, 2007 and 2009. Different shadow orientations indicate different day times captures. (b) Image used for test taken in 2012. A perceptual window of size $w = 32$ is represented

These images will be reduced by 80 % to decrease the amount of computation and memory needed by the system. Once reduced, the perceptual window represented in Fig. 2(b) will correspond to 32×32 pixels. Although there is a noticeable data reduction we have determined that there is sufficient information to make a smooth visual navigation, as we will see shortly. With the same philosophy we have discretized the global navigation map with 20 possible angles (the allowed turns that can develop the aircraft for each input image).

4 ConvNet Navigation

Recent progress in the computer vision and machine learning community has shown that the features generated by Convolutional Networks (convNets) outperform other methods in a broad variety of visual recognition, classification and detection tasks. ConvNets have been demonstrated to be versatile and transferable, i.e. even although they were trained on a very specific target task, they can be successfully deployed for solving different problems and often even outperform traditional hand engineered features [8].

In this section we will provide a convNet network model for developing our navigation task. We will also discuss the training strategy followed for network convergence. Is worth highlighting that we have used Theano Library (http://deeplearning.net/software/theano/) for the implementation of our models.

4.1 Proposed Model

As previously discussed, convolutional networks have several features that make them particularly suitable for working with real images. We propose to use the network model presented in Fig. 3 for this task. The first four layers are responsible to extract visual features while the last one, the softmax layer, is responsible to select from desired action for the input. All the internal layers utilize Parametric Rectified Linear Units (PReLU) [9], because they have shown greater results in network convergence and generalization.

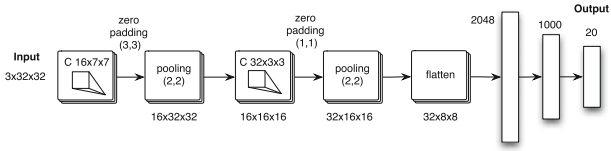

Fig. 3. Proposed convNet model. The first four layers are responsible to extract visual features while the last one, the softmax layer, is responsible to select from desired action for the input.

Layers responsible for the visual scene analysis will serve mainly for two purposes: the convolution step, where a fixed-size window runs over the image defining a region of interest, and the processing step, that uses the pixels inside each window as input for the neurons that, finally, perform the feature extraction from the region. This iterative process results in a new image (feature map), generally smaller than the original one. However, in our case this filter will be extended with zeros (zero padding) to generate more uniform filters for subsequent phases. After each convolutional layer, there are pooling layers that were created in order to reduce the variance of features by computing the max operation of a particular feature over a region of the image. This process ensures that the same result can be obtained, even when image features have small translations or rotations, being very important for object classification and detection.

Finally, the network processing units lose their spatial notion, lining up in a fully connected layer. All these 2048 neurons will be connected to another full connected layer of 1000 neurons, ending with the final classification layer of 20 neurons (the 20 allowed turns that can develop the aircraft for each input image). More specifically, we must learn 1061164 parameters including weights, bias and the PReLU coefficients for this network.

The most common classifier layer is the softmax function, also called normalized exponential. It is a generalization of the multinomial logistic function that generates a K-dimensional vector of real values, which represents a categorical probability distribution:

$$P(y_i|\boldsymbol{Z};\boldsymbol{W}) = \frac{e^{\boldsymbol{Z}^T\boldsymbol{W}_i}}{\sum_j e^{\boldsymbol{Z}^T\boldsymbol{W}_j}}$$

That can be interpreted as the (normalized) probability assigned to all the network outputs given the image $\boldsymbol{z}_i$ of the sample vector $\boldsymbol{Z}$ and parameterized by $\boldsymbol{W}$. One of the reasons for discretizing the action map is to be able to use a classification cost function, because regression problems require different cost functions (such as Mean Squared Error, MSE) much more difficult to converge. Intuitively, regression cost functions require very fragile and specific properties from the network to output exactly one correct value for each input.

4.2 Training Process

Network training involves finding the set of weights that minimize the classification error of the network. We have developed several tests, increasing the number of internal layers, the number of convolutional filters and their size. The more balanced network is presented in Fig. 3. In order to avoid overfitting several techniques have been tested, such as batch normalization layers, dropout or L2/L1 weight decay penalty. Our final network architecture uses a L2 penalty of 0.006 and a dropout factor of 0.5 for fully connected layers. Batch normalization layers achieved without L2 or dropout, better training accuracy values but do not generalize well on our test set. Therefore, the final cost $\mathcal{C}$ of the network to be minimized is presented, where N is the number of samples, λ is the intensity of the L2 penalty and k, l are weight iterators:

$$\underset{\boldsymbol{W}}{\operatorname{argmin}}\, \mathcal{C}(\boldsymbol{W}) \text{ where } \mathcal{C}(\boldsymbol{W}) = \frac{1}{N}\sum_{i} -\log P(y_i|\boldsymbol{Z};\boldsymbol{W}) + \frac{1}{2}\lambda \sum_{k}\sum_{l} \boldsymbol{W}_{k,l}^2$$

It should be noted that the process of generating training data for the four global image maps has big memory requirements. We have to extract patches from the images presented in Fig. 2 with a window size of $3 \times 32 \times 32$. For each global image 47045 patches must be extracted. Therefore, we calculate, for each phase of training, the patch extraction in an online way. More specifically, we extract random patches (24000 per epoch) for each of the training maps (6000 for each map) in order to train the network.

This training process involve to calculate the global navigation map of Sect. 2 and extract, for each random patch, the action that the UAV must develop. Because both maps have the same size and represent the same space, is trivial to obtain which action must be taken given a specific position. In this case, to train the network, we have taken the action found in the global motion map at the central position of the patch. In this way, the network has been trained to generate the required output (the turn angle extracted from our global navigation map) for every presented patch.

5 Results

As can be seen in Fig. 4, this network is able to learn the actions to be emitted depending on the perception of the drone (developing internally a global matching process with the four training maps). We have observed that the network has emitted the correct action for the 88 % of the presented images and the 75 % for the test set. It is important to underline that all the test images are extracted for a map not previously seen, captured three years later that the last training image set (even taken in a different time of the day). For this work, the training dataset extracted from Fig. 1(a) is not extended artificially (which could further increase the generalization ability of the network).

Previously, we have stated that for this application to work, recent global maps must be taken. But as we have seen, the network is able to generalize even without these conditions.

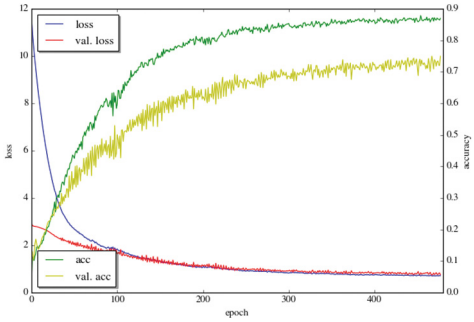

Fig. 4. Evolution of network error and accuracy for the training and validation data for each epoch. At every epoch 24000 random patches are presented to the network (6000 for each training map)

Is worth highlighting that once the network is trained we have calculated a mean runtime for prediction of 1.36 ms for an Intel core i7. This is the time required to calculate, given a patch input, the turn to be developed by the aircraft.

5.1 Comparing with a SIFT Registration Application

As has been discussed above, most global localization applications based on visual matching use feature detectors such as SIFT, Features from accelerated segment test (FAST) or Oriented FAST and Rotated BRIEF (ORB). In order to compare our system with this approximation we have implemented a global registration application based on [4].

As presented in Fig. 5 we use a SIFT extractor to obtain both, the robot perception (patch) features and the global map features. More specifically, we have used the SIFT key extractor provided by the openCV library (http://opencv.org). Once these features are extracted we perform a matching process using FANN algorithm [10] to determine which patch features correspond to

Fig. 5. (a) Matching of a drone perception (patch) through the map using SIFT. (b) Distance between real position and estimated position for 100 random patches of size 32×32 where a dot represents the real position and a cross the estimated one. (c) Same test with a window size of 64×64 and represented over the discretized global action map

the global map features. In Fig. 5(a), the matching results for one patch of size 32×32 pixels is presented. In Fig. 5(b) the distance between the actual position and the estimated one for 100 random patches is showed. In Fig. 5(c) the distance of real position an the predicted one is observed for 100 random patches of size 64×64. The predicted and real positions of the patches are plotted on the discretized motion map (20 angle levels) to observe the error on the emitted action.

To calculate a measure of performance for this algorithm, in order to compare it with our system, we obtain the predicted position of a patch and calculate the corresponding action (discretized angle), counting the correct predicted angles regards our global policy angle. We perform this test for several global maps sizes, increasing the perception size in a proportional way.

The following table shows the percentage of correct predicted actions by applying this algorithm to 10,000 randomly selected perceptions of size w in a previously presented global map. We also include the accuracy rate obtained by our convNet. It must be underlined that the train accuracy of SIFT algorithm is calculated based on patch extraction and matching from the 2009 map presented in Fig. 2(a). The test accuracy for this approximation is obtained calculating the matching process from the previous map over the 2012 test map presented in Fig. 2(b).

Base algorithm	Perception size (w in pixels)	Map size	Training accuracy (as decimal)	Test accuracy (as decimal)
Sift	32	128	0.35	0.15
Sift	64	256	0.61	0.13
Sift	160	640	0.80	0.11
ConvNet	32	128	0.88	0.75

5.2 Using the System for Reactive Visual Navigation

Finally, several trajectory predicted by the network for various selected start positions are shown in Fig. 6. The trajectories showed in Fig. 6(a) are developed using the four training maps. As can be seen, most of the predicted movements make the vehicle to reach the zones of attraction. However, there are some circumstances where this is not the case: there are zones where the combination of repulsion and attraction forces could nullify the potential field. In addition, some zones of the map combined with the movement policy could suffer from visual ambiguity. These problems could cause that the predicted trajectory may not match the global movement policy.

As it was presented before, the network is able to generalize and emit correct actions from not previously seen images. We have used our network to predict reactive trajectories using a not previously seen map, obtaining good results as can be seen in Fig. 6(b).

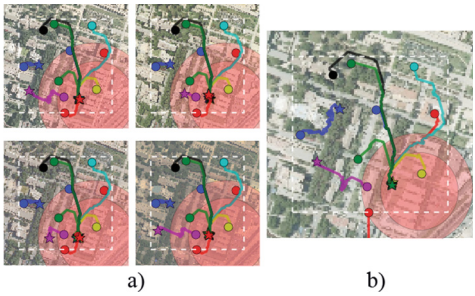

Fig. 6. Movement of ten aircraft navigating through the training and test maps. The initial positions (circles) and final (starts) are shown. The simulation is carried out for 120 iterations, with a velocity vector size of 1 m per iteration

Is worth highlighting that the results presented here are obtained without any filtering (it is a reactive navigation) using only the most likely angle defined by $P(y_i|\boldsymbol{Z};\boldsymbol{W})$, discarding the rest of information of this distribution. Obviously, including these two factors could substantially improve the robustness of the aircraft navigation.

6 Discussion

This paper presents an aid system for UAV navigation using the registration of perceptions through a global map. A convolutional neural network, to extract the essential features of the global visual maps given a global movement policy is used. The resulting model is able to generalize and work with no previously seen images, captured with different sensors and with changes in light conditions. Although this application does not require multi-scale matching (the UAV will fly at the same altitude of the global map), it easy to extend it, extracting multi scale patches for the network training process.

The accuracy of the network for training maps is higher than the obtained by matching features using SIFT key point extractors (even with larger maps and window perception size). Only using a much bigger global map (80 % larger) we could achieve a similar accuracy for training global maps. One of the reasons for the convNet proper functioning is that the network is able to extract the essential features of the map with respect to the global navigation policy. Thus, the more complex areas force the network to devote more resources to find visual indicators that characterize it. This explains its better performance in this task compared with specialized key point extractors. Moreover, the generalization of this network is far better that the key point matching approach, although we have discarded so much visual information to reduce the amount of resources required for network training.

In this approximation we have tested our system using real aerial maps. A coherent set of global maps (or previous perceptions) with a global control policy must be provided so that the production of two distinct actions for the

same perception will be avoided (an expansion of the perception window or a filtering step can smooth this problem). Although this network has been shown generic enough for working with new images we must ensure that the set of images that define the global map will be generic enough so that the network will be able to draw general characteristics.

As future plans we will test this system in a UAV vehicle, combining this sensor with the other aircraft modules to develop navigation and security tasks. Our next step will be to integrate this system within a low cost module that could be used to perform visual swarm robotics behaviours with low cost UAV.

References

1. Nogueira, K., Miranda, W.O., Dos Santos, J.A.: Improving spatial feature representation from aerial scenes by using convolutional networks. In: 2015 28th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 289–296. IEEE (2015)
2. Shan, M., Charan, A.: Google map referenced UAV navigation via simultaneous feature detection and description. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2015)
3. Cesetti, A., Frontoni, E., Mancini, A., Zingaretti, P., Longhi, S.: A vision-based guidance system for UAV navigation and safe landing using natural landmarks. In: Valavanis, K.P., Beard, R., Oh, P., Ollero, A., Piegl, L.A., Shim, H. (eds.) Selected Papers from the 2nd International Symposium on UAVs, pp. 233–257. Springer, Heidelberg (2009)
4. Cesetti, A., Frontoni, E., Mancini, A., Ascani, A., Zingaretti, P., Longhi, S.: A visual global positioning system for unmanned aerial vehicles used in photogrammetric applications. J. Intell. Robot. Syst. **61**, 157–168 (2011)
5. Long, J.L., Zhang, N., Darrell, T.: Do convnets learn correspondence? In: Advances in Neural Information Processing Systems, pp. 1601–1609 (2014)
6. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4353–4361 (2015)
7. Mejias, L., Fitzgerald, D.L., Eng, P.C., Xi, L.: Forced Landing Technologies for Unmanned Aerial Vehicles: Towards Safer Operations. In-Tech, Rijeka (2009)
8. Sunderhauf, N., Shirazi, S., Dayoub, F., Upcroft, B., Milford, M.: On the performance of convnet features for place recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4297–4304. IEEE (2015)
9. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)
10. Muja, M., Lowe, D.G.: Fast approximate nearest neighbors with automatic algorithm configuration. VISAPP (1) **2**, 331–340 (2009)