SVM classification of moving objects tracked by Kalman filter and Hungarian method

Gábor Szűcs, Dávid Papp, Dániel Lovas

Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Magyar Tudósok krt. 2., H-1117, Budapest, Hungary,

szucs@tmit.bme.hu, pappdavid27@gmail.com,
lovas.daniel@simonyi.bme.hu

Abstract. Fishery video data often require laborious visual analysis, therefore a video-based fish identification challenge is announced in the LifeCLEF campaign for automatic fish categorization and enumeration. We have elaborated a complex system to detect, classify and track objects (fishes) in underwater video by examining each image frame of it. For the detection process we used background subtraction and morphologic methods, and then our solution calculated bounding boxes based on object contours. We used Kalman filter to track the moving objects, but an additional matching method was required to pair the objects in consecutive time periods because of many fishes. We used Hungarian method for this matching problem. We categorized the detected fishes with C-SVC classifier, as an advanced SVM (Support Vector Machine) classifier. The classifier used high level descriptors, which are based on the extracted SURF vectors in each object. For optimization the C-SVC classifier we conducted a preliminary test, and we used the best parameters for teaching the classifier. We predicted the fish species in the official test video set, and our predictions were evaluated officially by NCS (Normalized Counting Score).

Keywords: SVM method, fish classification, tracking, Kalman filter, Hungarian method

1 Introduction

The analysis of video data usually requires very time-consuming and expensive input by human observers, and this is true for underwater videos as well, although the statistics of data collection would be very useful for exploratory applications, in particular for fisheries and biological areas. This analytical "bottleneck" greatly restricts the use of the powerful video technologies and demands effective methods for automatic content analysis to enable proactive provision of analytical information; and in order to solve this problem a challenge is announced in the LifeCLEF [1] campaign of ImageCLEF [2].

In this challenge two datasets (training data set with ground truth and a test set) were released for the video-based fish identification task [3]. The goal was to automatically count fish per species in video segments (e.g., video X contains N_1 instances of fish of species 1, ..., N_n instances of fish species n).

We have divided the problem into subtasks: object detection, classification and tracking, where the objects were the fishes; and we have implemented a video analysis system to solve these tasks. The applied methods and our solution are described below.

2 Object detection, classification and tracking system

Object classification and tracking are different tasks, but both of them based on object detection (fish detection) in images of videos. We have implemented fish detection in OpenCV in such way that the bounding boxes of detected fishes are stored as small images with corresponding information (actual timestamp, identifier, etc). The common subtask in both of the problems is mapping, i.e. interconnection of bounding box images and fish identifiers (these identifiers are generated for only distinguishable aim), because the results of the mapping can be used for classification and tracking as well. Thus the bounding box images are input for classification method, which estimates the species of fishes. The consecutive images with common fish identifiers can be classified into different species; therefore the final decision of classification in our solution is based on majority voting.

2.1 Detection of many fishes

The one of the challenges in the fish detection was the observation of many objects in an image. For object detection at first we have used background subtraction [4], in order to separate the foreground from background. The most common morphological methods (erosion, dilation and the combination of them, the closing) have been applied in order to get smooth and solid edges [5]. After this smoothing an algorithm for contours of objects has been applied, that is evolved by Suzuki and Abe [6]. Using the contours our solution calculates the bounding boxes and the object centres. Some of detected

objects were too small to substantively use in classification, hence those objects that were smaller then 15x15 pixel were filtered out (deleted).

2.2 Fish tracking with Kalman filter and Hungarian method

After the detection in our solution Kalman filter [7][8] has been used to track objects in three steps: (i) initialization, and after that there is a cycle process with (ii) prediction and (iii) correction.

Initialization: at the first frame, where any detection was, the Kalman filter was initialized; and for every detected object an identity number and a confidence value (CFV) were attached.

Prediction: In this step a prediction was made by Kalman filter on each detected object (using the calculated object centre) to forecast the future position of the investigated fish.

Correction: After prediction the new detections (in next frame of the video) give the measurements (which are used in the comparison of the measurements with predictions). These measurements were used for correct the Kalman filter objects. In order to reach the best tracked-measured coupling we applied the Hungarian method [9][10], completed with a restriction that we removed those objects that not belong to a new measurement. We present this applied method below.

Let v_i be a tracked object, where $i = 0...k$ and let w_j be a newly detected object where $j = 0...l$. Before our solution used the Hungarian method a M^{kxl} matrix was calculated, where M[i,j] denotes the Euclidean distance (measured in pixels in the image) of v_i and w_j objects. The rows and columns with higher elements than a given threshold were removed to prevent false matching. If a row corresponding to v_i was removed from M, then we lowered the confidence value (CFV) of that particular tracking. On the other side removing w_j means that we should track this object, i.e. probably this is a new object (a new Kalman filter has been created for this object), because this is far from all others.

At this point the Hungarian method were performed on M which resulted the optimal object tracking (v,w) pairs, i.e. minimal sum of distances.

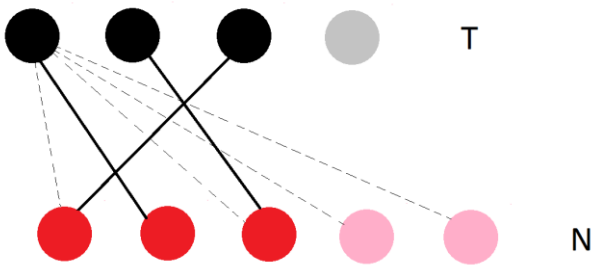

Fig. 1. Mechanism of Hungarian method

The Fig. 1. illustrates the mechanism of the applied Hungarian method, where the v_i objects are representing by a set of black and grey vertices (T) and w_j object by red and pink ones (N). The thick edges connect the matched (v,w) pairs, whereas grey and pink colour denote those nodes that have no pair. After the Hungarian method the matched object pairs will be the input of the correction phase of Kalman filter (black vertices are tracked objects and the red ones are the measurements).

2.3 Fish classification

2.3.1 Elaboration of image descriptors

The first part of the classification process is the representation of each image based on their visual content. This consists of three steps: (i) feature detection, (ii) feature description, (iii) image description as usual phases in computer vision.

Feature detection and description: Lots of different feature types can be detected in an image, e.g. corners, edges, ridges, as “interesting” part of an image. In our solution we have used Fast-Hessian Detector [11] to determine the “key points” in each image, and SURF (Speed Up Robust Features) [11] descriptor to extract local information at each key point. The SURF is based on Lowe’s SIFT (Scale Invariant Feature Transform) [12][13] with the expectation to be faster and more robust. Both of these methods are widely used in practice and in theoretical works (as well) with some possible further devel-

opment; but we have chosen SURF, because in videos the processing speed is more important. A SURF descriptor vector belongs to only one interesting point of an image, but an image possesses many feature descriptor vectors, which should be aggregated into an image descriptor.

Image description: The next step of creating the representation is the completion of high-level representation of each image. We have applied BoW (bag-of-words) model [14][15] for this purpose, where images are treated as text documents. According to this, "visual words" (so called "codewords") in images need to be defined from feature descriptors. The whole set of codewords gives the codebook (similarly to dictionary in text tasks). To determine the codebook we clustered the SURF descriptors with K-means [16] algorithm, and the resulting cluster centers will represent the codewords, since a centroid represents similar feature descriptors. We have experimented with different cluster sizes (*k*=5000 and *k*=10000), these are discussed later. At this point, the codebook with k codewords was available for further calculations, which can be considered as a concise representation of the whole image set. According to the codebook the next step is to create a descriptor that specifies the distribution of the visual codewords in any image, called high-level descriptor. We have built histograms as high-level descriptors for each image:

- Let $H = [0^{(1)}, 0^{(2)}, \dots, 0^{(k)}]$ be the initial histogram of the r^{th} image, where k denotes the size of the codebook (each element represent a codeword in *H*).
- We performed 1NN (1-nearest neighbour) algorithm for each S_i^r SURF descriptor to find the closest codeword (based on Euclidean distance), then the corresponding element of *H* was incremented by 1, where *i* cycles through the descriptors created for the r^{th} image

2.3.2 Training the classifier

For the classification task we have divided the labelled image set into two subsets: training and test set. We used the first one to create the codebook, and train the classifiers, and the latter for preliminary testing. The histograms were created for each training image, then we performed a variation of SVM (Support Vector Machine), the C-SVC (C-support vector classification)

[17][18] with linear kernel function to train the classifier (classification model). The SVM is basically a binary linear classifier, thus in order to extend it to a number of classified categories, the one-against-all technique was used. During this method a binary classifier was created for each category in the training set. We have executed kFold cross-validation technique before the preliminary test to determine the parameter of the C-SVC classifier. (After the training, the codebook was already available.)

2.4 Preliminary test of classification

For the maximization the goodness of classification part we have conducted a preliminary test, as validation phase of the learning process. The labelled image set was divided into train set (11221 images) and preliminary test set (11220 images). The training phase was executed by different parameters according to the number of codewords sizes (5000 and 10000), the number of dimensions of SURF vectors (64 and 128), and the number of SURF vectors in an image (80, 200, and 500). Besides the accuracy we have measured the speed of the algorithm as well, and the results can be seen in Table 1 and 2.

Table 1. Results of the preliminary testing at 5000 codewords

	number of dimensions: 128		number of dimensions: 64	
	run time (s)	accuracy	run time (s)	accuracy
vectors: 80	N.A.	0.259	N.A.	0.253
vectors: 200	9744	0.644	8672	0.595
vectors: 500	11863	**0.663**	9406	0.629

After the preliminary testing at 5000 codewords the case of 10000 codewords was testing with only 200 and 500 SURF vectors, because 80 vectors has resulted in very poor accuracy.

Table 2. Results of the preliminary testing at 10000 codewords

	number of dimensions: 128		number of dimensions: 64	
	run time (s)	accuracy	run time (s)	accuracy
vectors: 200	12532	0.611	11179	0.227
vectors: 500	15791	0.625	13927	0.381

The best result of the preliminary testing is the 0.663 at case of 128 dimensions, 500 vectors and 5000 codewords. The run time of the training phase was long, but the largest part (92-94%) of this was the SVM teaching, which was important for good accuracy, while the other parts are fast (creating codewords: 3-6%, histogram calculation: less, than 1%); furthermore the test phase was also quick (10 minutes). The run time values were measured on a PC (with Intel Core i7-4770K processor, 16 GB RAM and SSD).

For the prediction of official test we have chosen the SVM model with the best parameters (128 dimensions, 500 vectors and 5000 codewords), and we have submitted 2 prediction files (as results of 2 runs). The used method of the runs was the same (as described above), only small difference was between the two submissions. The first run (BME TMIT RUN1) contains false detections, because of the glances (blinks) and the low level threshold in detection (higher level threshold could avoid these); while at the second run (BME TMIT RUN2) these detections were filtered out, and corresponding predictions in the crucial videos were deleted.

3 Evaluation

3.1 Evaluation metrics

In the official evaluation the normalised counting score is measured (instead of accuracy as in our preliminary testing). The counting score (CS) is defined as can be seen in Equation (1), where d is the difference between the number of occurrences in the run (per species) and the number of occurrences in the ground truth (Ngt).

$$CS = e^{-\frac{d}{Ngt}} \tag{1}$$

The precision (Pr) is defined as Pr= TP/(TP+FP) with TP and FP being, respectively, the true positives and the false positives. The normalised counting score (NCS) is defined as NCS = CS x Pr.

3.2 Final official results

Our final official results for each fish species can be seen in Table 3., where the occurrences of fish species in the test set and the NCS (Normalized Counting Score) results are presented.

Table 3. Occurrences and NCS (Normalized Counting Score) results at fish species

Fish species ID (identifier)	Occurrences in the test set	BME TMIT RUN1	BME TMIT RUN2
1	93	0.00	0.05
2	129	0.03	0.12
3	517	0.21	0.29
4	1876	0.07	0.19
5	0	1.00	1.00
6	1317	0.42	0.58
7	24	0.08	0.19
8	1985	0.43	0.59
9	5016	0.02	0.05
10	0	0.85	0.85
11	118	0.02	0.09
12	1531	0.34	0.38
13	0	0.99	1.00
14	700	0.00	0.03
15	187	0.83	0.86

In cases, there were no occurrences of a given species and in the runs no fish were observed of those species, CS = 1 (since d = 0) and Pr was equal to 1. It can be seen in Table 3 that our solution has perfectly found this absence at two species (among three fish species: 5, 10, 13).

The aggregated official results of different fish species can be seen in Table 4. (the last column is the product of previous ones), and we can conclude that the second run of our submissions was better.

Table 4. Our final official results

Run identifier	Counting score	Precision	Normalized Counting Score
BME TMIT RUN1	0.62	0.44	0.27
BME TMIT RUN2	0.67	0.51	0.34

4 Conclusion

We have elaborated a complex system to detect and track objects (fish) in underwater video by examining each image frame of it. For the detection process we used background subtraction and morphologic methods, and then our solution calculated bounding boxes based on object contours. The first time we detect an object, we assign a Kalman filter to it, which is able to predict the possible location of that object in the next frame. We likely detect that particular object (the same fish) in the next frame also, but we should not apply a new Kalman filter. To deal with this, we used Hungarian method to pair the existing Kalman filters with the new "candidate" ones. Then we erased the candidate Kalman filters with matching pair, and kept the single ones. This way we were able to track the detected objects. We also categorized the detected fishes with C-SVC classifier; however this required representing the objects based on visual information. For this purpose, our system calculated SURF descriptors for each object, and then clustered them with K-means algorithm. Our solution built histograms for each fish based on the resulting cluster centres (according to BoW model), and these histograms were the input of the C-SVC. For optimization the C-SVC classifier we conducted a preliminary test, and we used the best parameters for teaching the classifier. We predicted the fish species in the official test video set based on our implemented model, and the number of occurrences of each fish species was enumerated. At the official evaluation (by the second submission) we reached 0.34 value of NCS (Normalized Counting Score).

References

1. Joly, A., Müller, H., Goeau, H., Glotin, H, Spampinato, C., Rauber, A., Bonnet, P., Vellinga, W. P., Fisher, B.: LifeCLEF 2015: multimedia life species identification challenges, *Proceedings of CLEF 2015* (2015)
2. Villegas, M., Müller, H., Gilbert, A., Piras, L., Wang, J., Mikolajczyk, K., Herrera, A. G. S., Bromuri, S., Amin, M. A., Mohammed, M. K., Acar, B., Uskudarli, S., Marvasti, N. B., Aldana, J. F., García, M. M. R.: General Overview of ImageCLEF at CLEF2015 Labs. *Lecture Notes in Computer Science*, Springer (2015)
3. Spampinato, C., Fisher, B and Boom, B.: LifeCLEF Fish Identification Task 2015, *CLEF working notes 2015* (2015)
4. KaewTraKulPong P. and Bowden, R.: An Improved Adaptive Background Mixture Model for Real- time Tracking with Shadow Detection*,* In Proc. *2nd European Workshop on Advanced Video Based Surveillance Systems*, AVBS01. Sept (2001)
5. Fisher, R., Perkins, S., Walker, A., Wolfart, E.: *Mathematical Morphology,* (2003) *http://homepages.inf.ed.ac.uk/rbf/HIPR2/matmorph.htm*
6. Suzuki, S. and Abe, K.,: Topological Structural Analysis of Digitized Binary Images by Border Following. *Computer Vision, Graphics, and Image Processing*, 30(1), 32-46. (1985)
7. Kalman, R. E.: A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1), 35-45. (1960)
8. Welch, G. F.: Kalman Filter. *Computer Vision: A Reference Guide*, 435-437. (2014)
9. Kuhn, H. W.: The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2), 83-97. (1955)
10. Frank, A.: On Kuhn's Hungarian method—a tribute from Hungary. *Naval Research Logistics (NRL)*, 52(1), 2-5. (2005)
11. Bay, H. and Tuytelaars, T. and Van Gool, L.: SURF: Speeded Up Robust Features, *9th European Conference on Computer Vision*, (2006)
12. Lowe, D.: Object recognition from local scale-invariant features. In: *ICCV* (1999)
13. Lowe, D. G.: Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision*, 60, 2, pp. 91-110, (2004)
14. Fei-Fei, L., Fergus, R., & A. Torralba, A.: Recognizing and Learning Object Categories, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, (2007)
15. Lazebnik, S., Schmid, C. and Ponce, J.: Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, Proceedings of the *IEEE Conference on Computer Vision and Pattern Recognition*, New York, Vol. 2, pp. 2169-2178 (2006)
16. MacQueen, J.: Some methods for classification and analysis of multivariate observations, Proceedings of the *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1, pp. 281-297 (1967)
17. Boser, B., Guyon, I., Vapnik, V.: A Training Algorithm for Optimal Margin Classifier, Proc. of the *5th Annual ACM Workshop on Computational Learning Theory*, pp. 144-152 (1992)
18. Cortes, C., Vapnik, V.: Support-vector networks*, Machine Learning*, Vol. 20, No. 3, pp. 273-297 (1995)