To appear in MTS/IEEE OCEANS Conference Aberdeen, 2017. DOI: 10.1109/OCEANSE.2017.8084691

Visual Fish Tracking: Combining a Two-Stage Graph Approach with CNN-Features

Jonas Jäger, Viviane Wolff and
Klaus Fricke-Neuderth
Fulda University of Applied Sciences
Fulda, Germany
{*jonas.jaeger, viviane.wolff, klaus.fricke-neuderth*}
@et.hs-fulda.de

Oliver Mothes and
Joachim Denzler
Friedrich Schiller University Jena
Jena, Germany
{*oliver.mothes, joachim.denzler*}
@uni-jena.de

***Abstract*—Video based underwater monitoring is used by a growing number of marine researchers to observe changes in the ecosystem over time. Especially the analysis of water organisms such as fish gives important information about the health-state of our oceans. Videos of fish communities can be collected in a large scale but the manual analysis by human experts is slow and expensive. Therefore, automatic methods for fish video analysis are required to make use of all collected data. In this paper we investigate computer vision approaches for visual multi-object tracking of fish in their unrestricted natural environment. Our focus is on a two-stage graph approach that uses activations of a convolutional neural network (CNN) to model object appearance. Evaluation results show that the examined two-stage graph tracking approaches have a higher tracking accuracy and are much faster than state of the art in fish tracking.**

1. Introduction

Underwater imaging systems such as autonomous underwater vehicles (AUVs [1]) or baited remote underwater video systems (BRUVs [2]) allow marine researchers to collect large amounts of video data on aquatic organisms like fish. These data provide valuable information about the ecosystem since fish communities reflect the impact of effects such as pollution, global warming, overfishing or habitat loss. The biggest issue with such data is that human experts can only analyze a small fraction of the collected video footage. To make use of all the available data, automatic methods for visual fish video analysis are required to generate powerful statistics that could be the basis for a more sustainable fishery management. The goal of fish video analysis is to count and classify each fish in a video frame. It is also important to keep track of hiding fish since the total number of present individuals matters. For pure fish species classification there are already astonishing results in literature. Qin et al. [3] utilize deep learning techniques to achieve a classification accuracy of 98.57% on a dataset [4] with 27370 images of 23 fish species created within the Fish4Knowledge project [5]. More difficult seems to be the detection and tracking problem because fish are visually very well adjusted to their habitat and swim in erratic pattern within a 3D space. Li et al. [6] report a fish detection precision of 82.7% on a dataset with 12 species from ImageCLEF. They use the Faster R-CNN deep learning approach of Ren et al. [7] and fine-tune it for fish detection.

The following sections present the investigation of a tracking-by-detection approach that integrates CNN Features within a two-stage graph framework. Fish appearance is extracted from convolutional neural networks that already have been successful applied to fish classification and detection problems. We also try other features like HOG [8] or RGB histograms.

2. Related Work

An algorithm that was designed to track fish in their unconstrained environment was proposed by Spampinato et al. [9]. Their tracking process is based on a covariance model to describe detected objects. Tracks are generated by a *Template Matching* approach by comparing the covariance matrices of the current object with candidate objects. The method is evaluated on static videos from the Fish4Knowledge project with hand-labeled ground-truth tracks. For evaluation, the metric of Bashir and Porikli [10] was adapted. The authors method achieved an average performance of 94% and outperformed the Camshift algorithm.

Another fish specific multi-object tracking method was developed by Chuang et al. [11]. In contrast to the work of [9], this method was designed for moving camera scenarios. The method is based on deformable multiple kernels. Detected objects are described by a color histogram, a texture histogram and a histogram of oriented gradients (HOG). Tracking is realized by the estimation of kernel motion with the mean-shift algorithm. The method is evaluated on videos collected by remotely operated underwater vehicles or cameras towed by a vessel. All videos show fish in their natural environment. The authors found that their method is faster than the graph-based algorithm of the authors of [12] but their tracking error is higher.

Berclaz et al. [12] formulate multi-object tracking as a graph theoretical problem [13], [14], [15], [16]. In this approach a Directed Acyclic Graph is used by tiling image

frames of a sequence. Detections based inside of the tiles are used as node layers of the graph and a k-shortest-path algorithm extracts correct object paths. Consequently, only one object can exist in a single tile which can evoke problems with object occlusions. Furthermore, the runtime of the offline tracker is very dependent on the resolution of the tile grid. In our graph-based tracking approach based on the method of [16] two graphs are used to overcome both disadvantages of occlusions and runtime.

The contribution of this paper is an investigation how well appearance features work in a two-stage graph-based tracking framework. We also adapted the method of [16] to fish tracking, which shows best performance of all tested approaches in our setup.

3. Deep Tracking-by-Detection

Our method combines the two-stage graph-based tracking approach of Mothes and Denzler [16] with CNN features [17] for object appearance modeling. The first stage extracts segments of robust object trajectories in an acyclic directed graph by finding shortest paths. CNN features are incorporated by defining the visual difference of two detections as the L2-norm of their feature vectors. In the second stage a similar graph of tracklets is formulated and tracks are extracted by finding again the shortest paths.

3.1. Deep Features

Our feature representation is based on the activations of a specific layer in a convolutional neural network (CNN). Figure 1 shows a schematic illustration of such a neural network. It consists of different types of layers that are connected in hierarchical order. Each layer in a network contains a specific number of neurons that will respond to the output of the previous layer.

The CNN in Figure 1 contains two main building blocks of a CNN architecture - convolutional and fully-connected layers. *Conv1* to *Conv5* indicate convolutional layers and *FC6* to *FC7* fully-connected layers. Neurons in a convolutional layer are organized as filter-kernels that respond to specific patterns in an image.

During the training process a CNN learns a world model based on the dataset that is used. In case of the ImageNet [18] dataset that was used to train our models a world representation is based on 1000 different object classes (e.g. fish, car, human, ...).

After training, an input image can be feed into the network and all neurons will react based on the learned model. Donahue et al. [17] found that these neural activations are powerful generic feature representations even if the net was not trained with images in the domain where features should be extracted for. They also mentioned that the feature representation can be further improved by fine-tuning a pretrained model to a specific domain, which is the SeaCLEF [19] dataset in our case. Therefore a much smaller dataset can be used than for the initial training. Zeiler and Fergus [20] show that in the first layer low level features

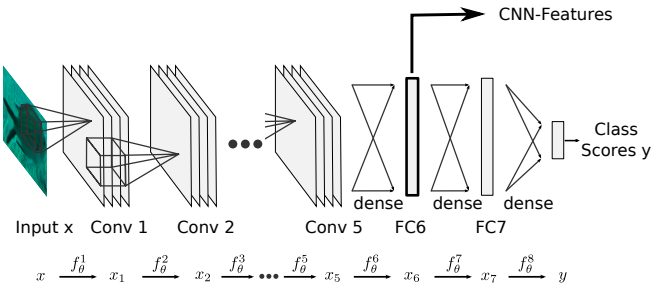

Figure 1. Schematic illustration of a convolutional neural network (CNN) in the style AlexNet [21] architecture. *Conv1* to *Conv5* indicate convolutional layers. *FC6* and *FC7* are fully-connected layers.

such as color or edge representations are learned and in the later stages higher level features can be observed.

For our tracking method we feed cropped bounding boxes that mark a fish inside a video frame into the network. The bounding boxes are provided as ground-truth information by the used dataset. As network architectures we used AlexNet [21] and VGG-19 [22] for our experiments. Both models have been pretrained on the ImageNet [18] dataset. We also adapted AlexNet to our fish problem by fine-tuning the model with the provided species sample images of the SeaClef dataset. See Section 4.1 for more information on the used dataset.

3.2. Graph-based Tracking

To find the motion trajectories of moving objects through the whole sequence, a reliable tracking algorithm is necessary. In the following section a two-staged graph-based tracking approach based on [16] is used for associating the detection results of Section 3.1. In the first stage the algorithm searches matching detections in different frames, which have strong similar properties, like appearance and position, and extract them to small trajectories (*Tracklet Extraction*). In the second stage, then the so-called tracklets are linked together to whole object trajectories (*Tracklet Linking*) with a similar strategy.
Using two stages for tracking has the effect that we can handle long-term occlusions and accelerate the whole tracking process instead of using a single stage like the approach of [12].

Tracklet Extraction

To find matched detections in single frames we formulate a Directed Acyclic Graph (DAG) $\boldsymbol{\mathcal{G}}$ where every node is represented by a detection hypothesis $\boldsymbol{h}_{t,i}$ of Section 3.1. Detection hypotheses are defined by $\boldsymbol{H} = \{\boldsymbol{H}_0, ..., \boldsymbol{H}_T\}$ with $\boldsymbol{H}_t = \{\boldsymbol{h}_{t,0}, ..., \boldsymbol{h}_{t,K_t}\}$ where $\boldsymbol{h}_{t,i}$ represents the i^{th} detection hypothesis of frame t in our sequence. Additionally, a source node $\boldsymbol{h}_{source}$ and a sink node $\boldsymbol{h}_{sink}$ are added to the DAG $\boldsymbol{\mathcal{G}} = (\boldsymbol{H}, \boldsymbol{E}, \boldsymbol{d}), \boldsymbol{E} \subseteq \boldsymbol{H} \times \boldsymbol{H}$. Both are fully connected to all other nodes of the DAG. All other nodes $\boldsymbol{h}_{t,i}$ are connected to $\boldsymbol{h}_{t+\Delta t,j}$. The graph $\boldsymbol{\mathcal{G}}$ is

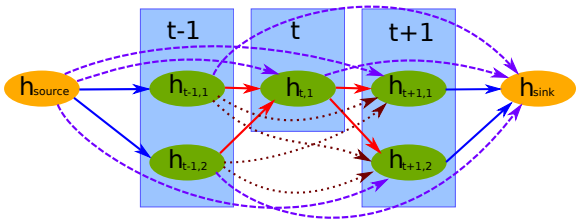

Figure 2. Tracklet Extraction - Initially, in the first stage the tracking algorithm generates a DAG of detection hypotheses and extracts a number of detection nodes of neighboring frames to create short confident paths (tracklets) with a high matching probability.

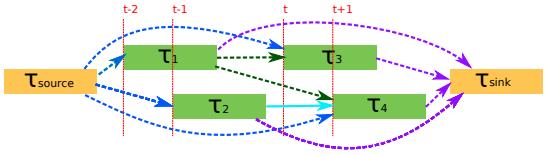

Figure 3. Tracklet Linking - In the second stage the extracted tracklets of the algorithms first stage with similar properties are linked to each other to whole object trajectories.

illustrated in Figure 2. The single edge weights between nodes of the DAG are defined in the following non-negative edge cost function $d : \boldsymbol{H} \times \boldsymbol{H}$ with P detection features:

$$d(\boldsymbol{h}_{t,i}, \boldsymbol{h}_{t+\Delta t,j}) = \sum_{p=0}^{P} \alpha_p \cdot d_p(\boldsymbol{h}_{t,i}, \boldsymbol{h}_{t+\Delta t,j}), \quad r.t. \quad \Delta t > 0 \tag{1}$$

where d_p are sub-weights of the detection features scaled by a factor α_p which regularizes the influence of the individual tracking priors. As detection features the temporal distance, the euclidean distance of the detections or appearance dissimilarities can be used. To extract only reliable path segments in the first stage, thresholds θ_{min} and θ_{max} must be defined for every sub-weight such that: $\theta_{min} \leq d_p \leq \theta_{max}$. Subsequently, we set all edges which do not match this precondition to infinity.

Now, applying iteratively a shortest path algorithm like *Dijkstra* [23] to $\boldsymbol{\mathcal{G}}$ finds reliable segments of object trajectories. After extraction, every weights of the shortest path in the graph is set to infinity to prevent multi-extractions. All extracted tracklets are then used in the second stage of the tracking approach.

Tracklet Linking

The confident paths segments of the first stage are now linked together to whole object trajectories. The extracted tracklet hypotheses $\boldsymbol{H}' = \{\boldsymbol{\tau}_0, ..., \boldsymbol{\tau}_{K'}\}$ build another DAG $\boldsymbol{\mathcal{G}}' = (\boldsymbol{H}', \boldsymbol{E}', \boldsymbol{d}'), \boldsymbol{E}' \subseteq \boldsymbol{H}' \times \boldsymbol{H}'$ illustrated in Figure 3. Every tracklet hypothesis τ_i represented as node in $\boldsymbol{\mathcal{G}}'$ is connected to the virtual source node τ_{source} and sink node τ_{sink} and is fully connected to tracklet nodes beginning in the following frames of the sequence. The non-negative cost function $d' : \boldsymbol{H}' \times \boldsymbol{H}'$ with

$$d'(\boldsymbol{\tau}_{t,i}, \boldsymbol{\tau}_{t+\Delta t,j}) = \sum_{p=0}^{P} \beta_p \cdot d'_p(\boldsymbol{\tau}_{t,i}, \boldsymbol{\tau}_{t+\Delta t,j}) \quad r.t. \quad t_{max}(\boldsymbol{\tau}_i) < t_{min}(\boldsymbol{\tau}_j) \tag{2}$$

defines the single edge weights of $\boldsymbol{\mathcal{G}}'$ again with single prior sub-costs d'_p like in Equation 1 of the algorithms first stage. These sub-costs are based on the detection features of $t_{max}(\boldsymbol{\tau}_i)$ and $t_{min}(\boldsymbol{\tau}_j)$ and were scaled by the factor β_p. Similar, like in the algorithms first stage, the P sub-costs d'_p can be computed by using the detection features of the single detections inside the tracklets. Here, the mean of appearances or the mean velocities–based on the spatial position–of the single tracklets can be used. Finally, again a shortest path algorithm is applied to the graph $\boldsymbol{\mathcal{G}}'$ and iteratively the single shortest paths are extracted. In order to prevent the extraction of the same path, all weights of an extracted path are set to infinity.

3.3. Incorporating a CNN appearance model

As a detection features for the graph-based tracking approach of Section 3.2 , we incorporate the extracted CNN features of our detections of Section 3.1 as appearance model. As described in Section 3.1 a CNN feature vector $\mathbf{x}$ is obtained by extracting the neuronal activations of a specific layer in a convolutional neural network. We define an appearance measure as the Euclidean distance of L_2 normalized CNN features. In addition to the appearance measure we use a temporal distance as edge cost between two detections in the graphs. When detection one is in frame n and detection two in frame $n + m$, the temporal distance is defined as $n - (n + m) - 1$.

As preconditions we use a maximum search radius in 2D space, a maximum change in object size and a maximum number of frames that can be skipped in order to assign a detection to a track.

4. Experiments

To evaluate our algorithm we used the CLEAR MOT metrics proposed by Bernardin and Stiefelhagen [24] for multi-object tracking scenarios. We further used a subset of the SeaCLEF 2016 fish video dataset [19] to compare our approach to other trackers.

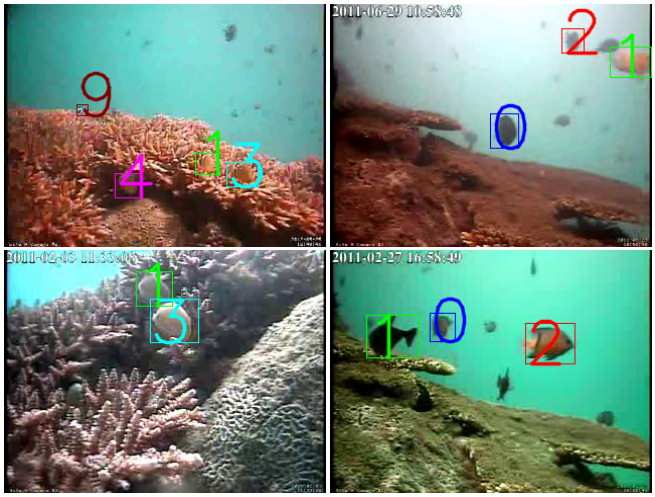

Figure 4. SeaCLEF dataset - Example frames with bounding boxes and track ids.

TABLE 1. ANALYSIS OF USED FISH DATASET IN TERMS OF FRAMES PER VIDEO, TRACKS PER VIDEO, DETECTIONS PER VIDEO AND MEAN DETECTIONS PER FRAME.

Video name	frames	tracks	detections	detections(pf)
3939..	290	54	318	1.8
e5ba..	552	32	519	1.2
07ba..	287	20	247	1.3
9c33..	287	18	225	1.4
7cad..	243	4	65	1.2
5996..	274	41	216	1.6
390b..	275	101	693	2.7
9a6b..	282	59	687	2.6
0b21..	612	14	316	1.7
2a69..	845	10	286	1.0
mean	395	35	357	1.7

4.1. Data

The SeaCLEF subset (see also [25]) consists of more than 20000 training images of 15 fish species and ten videos annotated with bounding boxes. Each annotation was agreed by two expert annotators. Additional ground truth track annotations have been added by the authors of this paper.

Figure 4 shows example frames from four different videos of the SeaCLEF dataset. It presents coral reef species in their natural environment. The video footage was recorded by the Fish4Knowledge [5] project using static underwater cameras. Special challenges in this dataset are the low resolutions of $320x240$ pixel and $640x480$ pixel and the changing lightning conditions in the scenes. A more detailed analysis of available ground-truth data for the used fish dataset can be seen in Table 1.

4.2. Implementation Details

The implementation of our two-stage graph algorithm is based on Python using the igraph [26] library. We further use the caffe [27] framework for CNN fine-tuning and to extract CNN features. All experiments were performed on a linux machine with the following hardware setup: Intel Xeon CPU E5-2620 v3 2.4Ghz, 32GB of RAM and a Geforce TitanX 12GB. For CPU computations a single core was used.

GPU hardware requirements are strongly dependent on the CNN architecture that is used for feature extraction. When using AlexNet we were also able to run our algorithm on a Quadro K620M with 2GB GPU memory.

Our algorithm depends on 14 different hyperparameters. This parameters have been determined empirical with the characteristics of the fish dataset in mind.

TABLE 2. TRACKING PERFORMANCE IN TERMS OF CLEAR MOT AND RUNTIME ON THE DESCRIBED SUBSET OF SEACLEF [19] FISH VIDEOS. EACH ROW REPRESENTS THE MEAN PERFORMANCE OF A SPECIFIC ALGORITHM FOR ALL 10 VIDEOS OF THE DATASET. OUR METHOD IS TSG-CNN.

Method	MOTA	MOTP	IDSW	time
TSG-J [14]	29.96	59.70	0.3	1.66s
GMMCP [15]	31.44	83.35	0.5	40.36s
Berclaz [12]	63.40	100.00	4.1	4h 21m
TSG-Combined	87.6	100.00	2.7	2m 42s
TSG-CNN-Alex	87.6	100.00	2.7	2m 19s
TSG-CNN-VGG	87.6	100.00	2.7	4m 30s
TSG-CNN-AlexFish	87.6	100.00	2.4	2m 36s
TSG-M [16]	87.75	100.00	2.0	4s

4.3. Comparison to other Trackers

Table 2 shows experimental results of the tracking performance in terms of CLEAR MOT and runtime. As input for all trackers ground-truth detections have been used.

TSG-J and TSG-M indicate the two-stage graph based approach of Jiang et al. [14] and the refined method of Mothes and Denzler [16] at which our work TSG-CNN is based on. The suffix in TSG-CNN describes the CNN model that was used for feature extraction. TSG-CNN-Alex indicates that AlexNet was used and AlexFish is the fine-tuned version adapted to the fish dataset. In both nets features were extracted from layer $Conv5$. In case of VGG-Net we used features from layer $Conv5_4$. TSG-Combined in Table 2 means that we combined temporal, spatial and CNN distances to calculate the edge cost in the graph to describe the relationship between detections. So we combined the TSG-CNN approach with TSG-M. As precondition for the TSG experiments in Table 2 we used a maximum search radius of 10% in 2D space related to the image size. For TSG-M we used the same python base implementation as for TSG-CNN. The hyperparameters for the TSG experiments in this section are fixed for all TSG approaches.

TSG-J completely missed all tracks in 4 out of 10 videos which results in low MOTA/MOTP values and also in a low number of id switches (IDSW). For our experiments with TSG-J we used the original C++ implementations provided by the authors. We also tested the GMMCP tracker of Dehghan et al. [15] and the graph based approach of Berclaz

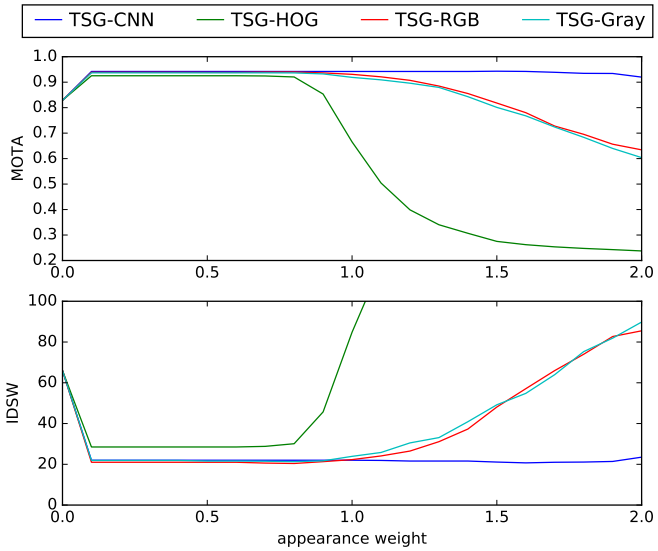

Figure 5. Performance of two-stage graph tracking for different features depending on the appearance weight. CNN, HOG, Gray-Histogram and RGB-Histogram. CNN features have been extracted from layer $Conv5$ in AlexNet.

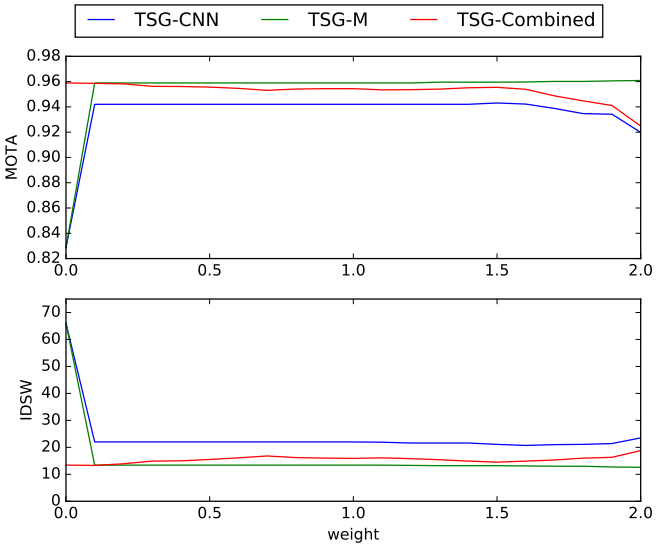

Figure 6. Performance of two-stage graph tracking when comparing TSG-M, TSG-CNN (AlexNet, $Conv5$) and the combination of both. Results for TSG-M depend on the spatial weight. TSG-CNN and TSG-Combined depend on the appearance weight. For TSG-Combined the spatial weight is fixed to 1.0.

et al. [12]. For the GMMCP tracker we used the original Matlab implementation provided by the authors. In the case of Berclaz a C++ reimplementation is used. The GMMCP tracker seems to have difficulties when applied to fish tracking scenarios and has low performance in terms of CLEAR MOT. While the method of Berclaz works much better than GMMCP it has a lower accuracy, more id switches and is much slower compared to our approach.

When comparing TSG-CNN-Alex with TSG-CNN-VGG it seems that deeper architectures for feature extraction do not improve the tracking performance but need more processing time. Only the fine-tuned TSG-CNN-AlexFish method has slightly less id switches.

Best performance in our experiments showed the method of Mothes and Denzler [16] TSG-M. It has a slightly higher accuracy than the TSG-CNN algorithms and less id switches while it is much faster. The combination of the TSG-M and TSG-CNN approach (TSG-Combined) results in no improvement of the tracking performance. This indicates that spatial distance of detections is equal or better suited to assign detections to tracks than CNN features. This also means that appearance features might be useful in scenarios where spatial distance is no good indicator for track assignment.

4.4. CNN vs other features

We where also interested in the question how other features work in the two-stage graph-based tracking framework. Therefore, HOG features, Grayscale histograms and RGB histograms have been tested in addition to CNN features to describe detected objects.

Figure 5 shows results for MOTA and IDSW depending on the appearance weight which controls the influence of the appearance measure to the overall edge cost between two detections or tracklets in the graph. The precondition of a maximum search radius has been set to infinity in order to see only the influence of the appearance features to the tracking performance. This results in a higher MOTA but also in a higher number of id switches, that can be mainly controlled by the choice of an appropriate max search radius.

When looking at Figure 5 we see that CNN features work best and HOG features have worst performance. RGB and Grayscale histograms have almost same performance as CNN features when choosing the right value for the appearance weight. This indicates that the appearance of individual fish can be described equally well by CNN features and RGB or Grayscale histograms.

4.5. Spatial vs CNN distance

Figure 6 shows a comparison of our TSG-CNN method, TSG-M [16] and the combination of both. Results for TSG-M depend on the spatial weight which controls the influence of the spatial distance between two detections to the tracking performance. TSG-CNN and TSG-Combined depend on the appearance weight. For TSG-Combined the spatial weight is fixed to 1.0. As in Section 4.4 the maximum search radius is set to infinity.

We see that TSG-M has best accuracy and lowest number of id switches. The combined method has lower performance than TSG-M but is better than TSG-CNN. Thus, spatial distance is better suited to find objects for a track than an appearance measure for the problem of fish tracking.

5. Conclusion and Further Work

We presented a multi-object tracking method that combines a two-stage graph-based approach with a CNN appearance model. The experiments show that our method TSG-CNN has similar performance as the pure graph-based approach of Mothes and Denzler [16] TSG-M but is much slower. Both two-stage graph approaches clearly outperform the method of Berclaz et al. [12], which was reported as the best fish tracking method in the work of Chuang et al. [11]. Our experiments in Section 4.5 indicate that the spatial distance is more important than an appearance measure for the two-stage graph tracking performance, but appearance could be used in scenarios where the spatial distance is not a good indicator for related detections. We also find in Section 4.4 that RGB-histograms have almost the same descriptive power as CNN features to describe individual fish in the two-stage graph.

In future it would be interesting to automatically finetune the 14 hyperparamerters of the two-stage graph method to the data. This could improve the performance of all presented TSG approaches. For research in automatic fish video analysis in general it would be useful to have a large dataset for evaluation that consists of ground-truth annotations of bounding boxes with species label and the associated track.

References

[1] G. Trimble, "A multiprocessor system for auv applications," in *Unmanned Untethered Submersible Technology, Proceedings of the 1987 5th International Symposium on*, vol. 5, Jun 1987, pp. 208–219.

[2] T. J. Langlois, E. S. Harvey, B. Fitzpatrick, J. J. Meeuwig, G. Shedrawi, and D. L. Watson, "Cost-efficient sampling of fish assemblages: comparison of baited video stations and diver video transects," *Aquatic Biology*, vol. 9, no. 2, pp. 155–168, 2010.

[3] H. Qin, X. Li, J. Liang, Y. Peng, and C. Zhang, "Deepfish: Accurate underwater live fish recognition with a deep architecture," *Neurocomputing*, vol. 187, pp. 49 – 58, 2016, recent Developments on Deep Big Vision.

[4] B. J. Boom, P. X. Huang, J. He, and R. B. Fisher, "Supporting ground-truth annotation of image datasets using clustering," in *Proceedings of the 21st International Conference on Pattern Recognition, ICPR 2012, Tsukuba, Japan, November 11-15, 2012*, 2012, pp. 1542–1545.

[5] R. B. Fisher, Y.-H. Chen-Burger, D. Giordano, L. Hardman, and F.-P. Lin, *Fish4Knowledge: Collecting and Analyzing Massive Coral Reef Fish Video Data*, 2016.

[6] X. Li, M. Shang, J. Hao, and Z. Yang, "Accelerating fish detection and recognition by sharing cnns with objectness learning," in *OCEANS 2016 - Shanghai*, April 2016, pp. 1–5.

[7] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893.

[9] C. Spampinato, S. Palazzo, D. Giordano, I. Kavasidis, F.-P. Lin, and Y.-T. Lin, "Covariance based fish tracking in real-life underwater environment," 2012, pp. 409–414.

[10] F. Bashir and F. Porikli, "Performance evaluation of object detection and tracking systems," in *In PETS*, 2006.

[11] M. C. Chuang, J. N. Hwang, J. H. Ye, S. C. Huang, and K. Williams, "Underwater fish tracking for moving cameras based on deformable multiple kernels," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, pp. 1–11, 2016.

[12] J. Berclaz, E. Turetken, F. Fleuret, and P. Fua, "Multiple object tracking using k-shortest paths optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[13] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[14] X. Jiang, D. Haase, M. Krner, W. Bothe, and J. Denzler, "Accurate 3d multi-marker tracking in x-ray cardiac sequences using a two-stage graph modeling approach," in *Computer Analysis of Images and Patterns*, vol. 8048, 2013, pp. 117–125.

[15] A. Dehghan, S. Modiri Assari, and M. Shah, "Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[16] O. Mothes and J. Denzler, "Anatomical landmark tracking by one-shot learned priors for augmented active appearance models," in *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2017.

[17] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," *CoRR*, vol. abs/1310.1531, 2013.

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[19] SeaCLEF. (2016) Seaclef 2016. [Online]. Available: http://www.imageclef.org/lifeclef/2016/sea

[20] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, 2014, pp. 818–833.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[23] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[24] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, no. 1, p. 246309, 2008.

[25] J. Jäger, E. Rodner, J. Denzler, V. Wolff, and K. Fricke-Neuderth, "Seaclef 2016: Object proposal classification for fish detection in underwater videos," in *Working Notes of CLEF 2016 - Conference and Labs of the Evaluation forum*, 2016, pp. 481–489.

[26] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal*, vol. Complex Systems, p. 1695, 2006.

[27] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.