Neurocomputing 342 (2019) 137–144

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Continuous convolutional object tracking in developmental robot scenarios

Stefan Heinrich*, Peer Springstübe, Tobias Knöppler, Matthias Kerzel, Stefan Wermter

Department of Informatics, Knowledge Technology, Universität Hamburg, Germany

ARTICLE INFO

Article history:
Received 13 July 2018
Revised 5 October 2018
Accepted 9 October 2018
Available online 4 February 2019

Keywords:
Object tracking
Convolutional neural networks
Developmental robotics

ABSTRACT

Tracking arbitrary objects in natural environments is a challenging task in visual computing. A central problem is the need to adapt to changing appearances under strong transformation and occlusion. We propose a tracking framework that utilises the strength of Convolutional Neural Networks to create a robust and adaptive model of the object from training data produced during tracking. An incremental update mechanism provides increased performance and reduces the computational costs for training during tracking, allowing for robust real-time tracking with state-of-the-art performance. Together with optimisations for deploying the framework on humanoid robots and distributed devices, this shows its viability for research in developmental robotics on questions around infant cognition or active exploration.

1. Introduction

The process of tracking the changing position of an object is a very fundamental problem in the fields of computer vision as well as artificial intelligence and has been studied for a long time [1–3]. It is an important task in the area of intelligent devices and robotics, as tracking objects visually is often a prerequisite for complex tasks. For example, a developmental robot can learn the affordances of a manipulated object by correctly keeping track of this object's change, while for an autonomous car keeping track of other vehicles and pedestrians in the streets can help planning a route and avoid collisions [4]. However, the visual tracking of arbitrary objects in a video stream is still a difficult task because the objects' appearances may change over time [5].

To overcome these challenges, an object tracker needs a mechanism for identifying and extracting robust features from a video stream. It also needs a flexible method for learning a model of the object's representation using a scarce supply of training data and adapting it dynamically over time. Previous attempts to provide such functionality include integrating Convolutional Neural Networks (*CNNs*) for feature extraction and learning the object's visual representation. For instance, the Fully Convolutional Network based Tracker (*FCNT*) [6] uses the convolutional part of VGG16 [7] to extract visual features from a video stream and trains another small CNN on those features. In this approach, both the final and a penultimate convolutional layer were used as outputs since their features contribute different discriminating abilities. Thus, the object tracking adopts object feature models that resulted from training with huge object recognition databases. To improve the quality of the object representation, other algorithms include using pre-trained R-CNNs for feature extraction [8], training features in CNNs while tracking [3], or creating negative training samples by taking false locations around the correct position [9]. Nevertheless, these approaches suffer from weak update selection schemes, making them prone to using poor samples and at the same time particularly complex, preventing any real-time application. Other approaches that prepare adaptive silhouette models for specific shapes, such as humans, in order to provide robust tracking [10], need depth information that is in most cases not available.

In this paper, we comprehensively describe *HIOB*, a HIerarchical and modular OBject tracking framework[1], and provide profound insight into employing HIOB in developmental robot scenarios. HIOB is a CNN-based tracker that consist of highly optimised components for feature extraction and components that train and update specific object models during tracking conditioned by the tracking confidence. Since HIOB is supposed to provide robust tracking, particularly under occlusion and distortion conditions, for real-time

* Corresponding author.
E-mail addresses: heinrich@informatik.uni-hamburg.de (S. Heinrich), peer@garstig.org (P. Springstübe), tobias@knoeppler.net (T. Knöppler), kerzel@informatik.uni-hamburg.de (M. Kerzel), wermter@informatik.uni-hamburg.de (S. Wermter).

[1] We first introduced HIOB as a contribution to ESANN2018 [11].

https://doi.org/10.1016/j.neucom.2018.10.086

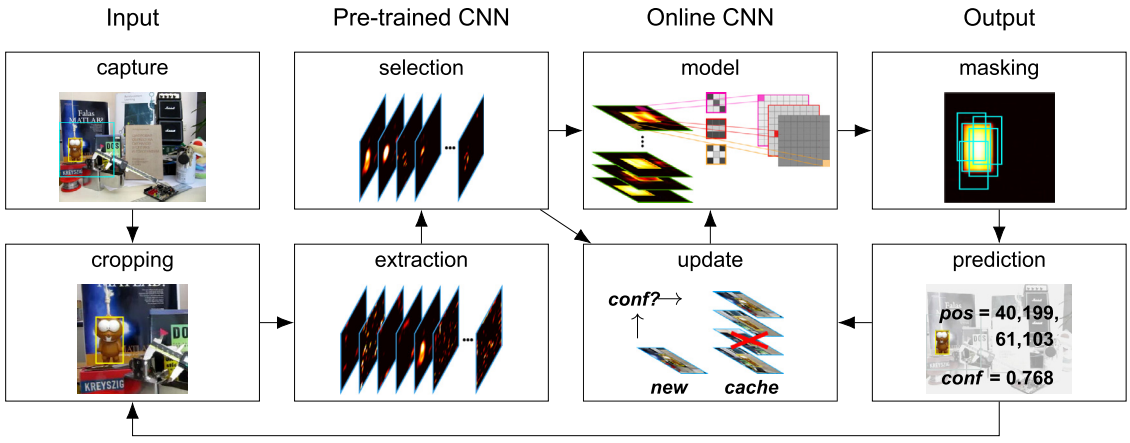


Fig. 1. Overview over the HIerarchical modular OBject tracking framework HIOB.

applications, it is optimised for performing well and at the same time remaining computationally efficient. This allows employing HIOB[2] in the developmental robotics scenario and thus enables such a robot to actively explore and learn in real environments and real time.

2. Hierarchical modular object tracking

Our proposed framework is an extension of the FCNT introduced by Wang et al. [6]. It combines the feature extraction capabilities of a pre-trained CNN (*pCNN*) with the flexibility of an online CNN (*oCNN*). This allows to rely on model features obtained from training on huge object recognition datasets, but also to quickly create and update specific object models during tracking. An overview of the framework is provided in Fig. 1.

2.1. Initialising the tracking

The tracking is initialised with a bounding box around the given position of the object in the first frame F^0. Next, the captured image is cropped to the region of interest (*ROI*) and scaled to a fixed size s, e.g. $s = 368 \times 368$, which introduces a trade-off between computational complexity and precision. This so-called *SROI* provides the input into the pCNN that consists of the convolutional part of the VGG16 [7]. In particular, the last convolutional layer *conv5_3* and an earlier layer *conv4_3* are used for feature extraction and selection, since they have been shown to describe different visual aspects well [6]. Here, other options are possible as well, such as using ResNet [12] or context-specific feature models, depending on the desired constraints for accuracy and speed as well as the computation framework. A target heat map regression analysis is used to identify those features that have the strongest relevance for locating the object and thus to reduce the number of features from 512 per output layer to 384. The selected filters are used as input for the oCNN that consists of two convolutional layers, of which the first layer includes 32 filters with a kernel size of 9×9 and connects with a concatenated ReLU to the second layer that comprises a single 5×5 filter. Finally, the oCNN is trained to produce the prediction mask in the form of a 2D-Gaussian, cropped to the object's bounding box on a 46×46 array. Since the object is arbitrary and not known a priori, this bounding box is the first and only data point.

[2] The complete implementation of the HIOB framework is available at https://github.com/kratenko/HIOB.

2.2. Tracking

During tracking, for each following frame F^t, $t > 0$ the captured image is cropped around the last known position of the object, scaled, and fed as input to the pCNN, similar to the initialisation step. The resulting 2×384 features are used as input to the oCNN to produce a 46×46 mask for predicting the most probable areas of the object's position. This mask is normalised to values within [0, 1] and integrated into a mask M^t representing the position in the full-sized captured image. On this prediction mask, a large number c of candidates for a new bounding box is created by altering the last predicted bounding box randomly, according to a Gaussian distribution. The number c is again a trade-off between computation effort and precision, but a large number of, e.g., $c = 1{,}000$ candidates is reasonable because the creation and evaluation can be done in parallel, similar to the calculation of filter kernels. For each candidate X_n^t a confidence value $conf_n^t$ is calculated, with $A(X_n^t)$ being the size of X_n^t in pixels and $M^t(j)$ the probability predicted by the oCNN for pixel j:

$$conf_n^t = \frac{\sum_{j \in X_n^t} M^t(j)}{A(X_n^t)} \tag{1}$$

The candidate with the highest confidence is used as the predicted position X^t and its confidence is used as that prediction's confidence $conf^t$. If no candidate is rated above the threshold $min\_conf = 0.1$, the previous position is kept ($X^t = X^{t-1}$) and the confidence is set to $conf^t = 0$.

After each prediction, based on an update strategy, the sample is considered for updating the model. In this case, the sample is appended to a FIFO cache, storing the last 10 selected samples, and a single update iteration of the model is executed in its current configuration. Independent of the course of the tracking, the sample obtained from the initial frame is always kept in the cache because it is the only reference guaranteed to be of good quality. This way the oCNN adapts to any morphology change in the appearance of the object.

2.3. Update Strategies

Deciding which samples are used for updating the model is critical for the success of the tracking. The algorithm must be able to adapt to changes in appearance to create a solid model, but at the same time must avoid samples of poor quality as they can corrupt the model. Ideally, the updates should provide all occurring perspectives and object appearances under occlusion and motion. However, it is not necessarily possible to inform the algorithm of a perspective that is particularly *good*, because it cannot grasp

the full object's characteristic a priori. As good heuristics, updates should be triggered by a prediction with a low confidence $conf^t$, which indicates a change in appearance. Utilising a lower bound for the confidence of samples prevents poor quality data from distorting the model, e.g., because the tracker clearly lost the object for a moment. As another heuristics, updates should also get enforced after a certain number of frames δ_t without changes to the model, to ensure that it is kept up to date. In this work, we explore six update strategies in detail in order to understand the impact of these heuristics:

- *None* – the model remains as trained for the initial bounding box. No further updates are executed on the model, thus the strategy comes at no computational cost but has no opportunity to reflect changes, and serves as a baseline.
- *Full* – update on every frame, thus maximising the available training data by including all changes over time. This strategy explores the opposite extreme to *None*.
- *Static* – update after a fixed number of $\delta_t = 20$ frames, to reduce the training effort and attempting to still include reasonable changes of the object's morphology in an uninformed manner.
- *Dynamic* – update when the appearance is considerably different to the trained model, measured by the confidence $conf^t$ being lower than 0.4, to include changes in an informed way.
- *Low Confidence Combined (LCC)* – update in case of low confidence $conf^t \leq 0.4$ combined with enforcing updates for $\delta_t = 20$, based on Wang et al. [6].
- *High Gain Combined (HGC)* – update high gain cases: low but not too low confidence $0.2 \leq conf^t \leq 0.4$, avoiding poor samples, enforced for $\delta_t = 20$.

3. HIOB for developmental robotics

To utilise an object tracker, such as HIOB, in real-world tasks in general and on a developmental robot in particular, we must take the constraints and the context of real-world tracking into consideration. In order to allow for real-time processing, we also preliminarily studied the trade-offs as mentioned above and enhanced the HIOB framework implementation and deployment.

3.1. Conditions and context of object tracking

In developmental robotics research, the goal is to study human cognitive functions in settings that resemble the conditions of human infants interacting in natural environments [13]. These conditions include *embodied* interaction, thus interacting with natural motor and sensing capabilities of an infant as well as integrating the multi-modal sensations within active perception [14]. For our tracking, this means that we need to be able to rely on standard RGB vision, because young infants are still developing a sense for depth based on stereo vision, as well as on fast, thus real-time processing.

The embodied interaction in natural environments also introduces particular properties of the context. Infants, and thus developmental robots, often interact with objects by grabbing, shaking, and scooting them, and are in the process of developing object permanence despite introducing strong motion and motion blur, as well as partial or full occlusion [15]. Thus, our tracking must be able to handle these contexts well in order to be feasible for a broad range of research questions such as active perception, language acquisition, and higher-order planning.

3.2. Enhancing HIOB for real-time tracking

In HIOB, therefore, we took great care to realise the framework for real-time applications on a robot that is equipped with RGB vision. Overall, the framework is implemented in Python and Tensorflow[3], which allows for accessible prototyping and at the same time particularly fast processes for redundant computations within CUDA stream processors. Notable optimisations have been done in the feature extraction, feature selection and prediction candidate generation and evaluation. Additionally, we wrapped the frame-capturing and computing processes as (optional) ROS[4] nodes, in order to allow an easy deployment for a range of robot devices and computers[5].

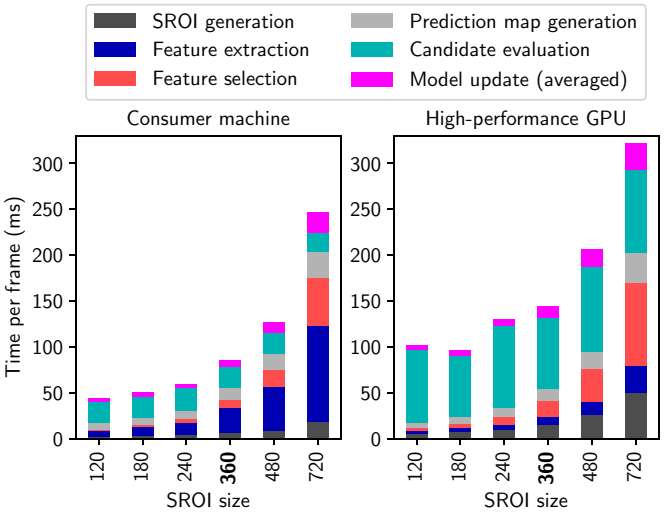


Fig. 2. Preliminary evaluation of computation times for HIOB components after optimisation (FPS = 1000/ms).

For a preliminary evaluation, we tested HIOB for live tracking on our robot platform under different SROI sizes s, in order to gather insights into computational effort under realistic conditions. We used the vision from the robot and ran the calculations on either a consumer machine or a high-performance GPU server[6]. Fig. 2 presents the overall and component-wise computation times over both systems. A notable outcome is that the fully tensor-graph-based feature extraction heavily benefits from the GPU as expected, while the feature selection and candidate evaluation, which are both mixtures from parallel python calls and tensor computations, still gain from good CPU core speed. Overall this test shows that HIOB can get operated with stable 11–17 frames per second (*FPS*) for reasonable SROI sizes on HD resolution.

3.3. NICO-object interaction data

A recent example of a developmental robot is the Neuro-Inspired COmpanion *NICO*, which is used in research on multi-modal human-robot interaction and neuro-cognitive modelling [16]. NICO includes two HD RGB cameras and interaction capabilities of a 3.5-year-old child. In interactions with objects, the robot's hands are naturally introducing huge occlusions, making the tracking of object manipulation effects from the visual perspective difficult.

To employ NICO as a test-bed, we recorded 60 object-hand interactions, with frame rates as described above. The recordings include different push, pull, grasp, and lift actions on a broad range

[3] https://www.tensorflow.org/.
[4] https://www.ros.org/.
[5] Prepared Docker image (https://www.docker.com/) using ROS: https://github.com/theCalcaholic/HIOB-ROS.
[6] Consumer machine: NVidia GTX 970, Intel i5-6600k; High-performance GPU: NVidia TitanXp, Intel Xeon E4-2620.

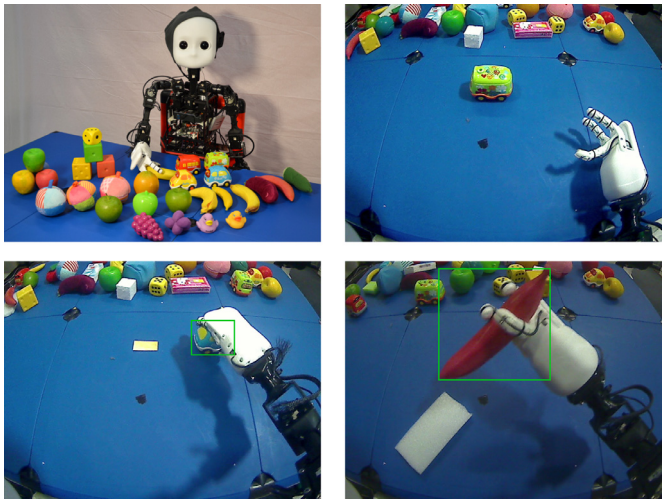


Fig. 3. The NICO robot in the object interaction setup and representative examples of its vision and tracked objects under occlusion (bottom-left) and with difficult shape and texture (bottom-right).

of toy objects that show diverse behaviour when interacted with, such as rolling away, bouncing on the table, or changing their morphology. Accordingly, we manually annotated all recordings to capture the object shape by an exact rectangle (see Fig. 3).

4. Evaluation and analysis

On the one hand, we want to measure the performance of our framework in a comparable fashion on challenging datasets and live benchmarks. In particular, we need to measure the strength of the modular approach and the update strategies on a broad range of scene characteristics. On the other hand, we want to analyse in depth, how the mechanisms of online updates impact on the tracking and preservation of a suitable object model.

4.1. Performance

For the performance tests, we used the extension of the established Online Object Tracking Benchmark (*OOTB*) by Wu et al., which includes 100 tracking sequences with up to 3872 frames and metrics for evaluation [17]. Here, the precision metric measures the Euclidean distance between the predicted bounding box position and the ground truth, while the success metric measures the intersection in proportion to the union and thus the overlap between predicted bounding box and ground truth.

Also, we participated in the Princeton Tracking Benchmark (*PTB*), which includes data and tests that are not publically available for framework or model optimisation [5]. Finally, we measured the tracking on our NICO-object interaction dataset to compare the performance and success results on HD vision and restricted frames per seconds with the other benchmarks.

4.1.1. Online object tracking benchmark

We used the OOTB to evaluate the six update strategies. Table 1 provides the results on individual challenges included in the benchmark. Strategy *None* illustrates how HIOB performs without adaptation of the model during tracking. The poor performance by the *Full* strategy shows that more updates are not necessarily an improvement. Periodic updates with the *Static* fall short if an object was occluded, out of view, or blurred, while updating with the *Dynamic* strategy tends to update to an extent that the model is no longer able to provide good predictions.

LCC produces comparable results in both frameworks, FCNT and HIOB, as plotted in Fig. 4. The tracking of HIOB is much smoother compared to FCNT because it uses a single training step on updates while FCNT reinitialises its network and executes 50 training steps for every update. An analysis of failed trackings shows that the model is often disrupted by poor quality samples in the training data. These errors are amplified by the generation of additional erroneous predictions. *HGC* avoids this corruption by discarding samples of very low confidence. The result is a higher tracking performance with even fewer model updates. A significant performance increase can be seen for tracking sequences that include an occlusion of the object or motion blur, which is likely to produce erroneous training samples (compare Table 1).

4.1.2. Princeton tracking benchmark

On the PTB[7], the HIOB framework ranks third out of 15 recent frameworks on the RGB challenge (not using depth information). It achieved the best results for the “animal” (72.5%) and “rigid” (78.2%) target types and ranks overall among the top three in all categories. A reason for not being able to compete with the top two for the “human” (53.1%) target type seems to be the fact that these frameworks were particularly optimised for shapes like human poses by using RGB-D information [10]. The results also indicate that HIOB is particularly good in non-occlusion (84.5%) and still quite good in occlusion cases (52.9%) that appear to be a major difficulty for all frameworks.

4.1.3. NICO scenario tracking test

Using the NICO object interaction data we evaluated different SROI sizes restricted by the frame per second rate that was obtained in the preliminary speed comparison. Fig. 5 presents the precision and success results, indicating that an SROI of $s = 360 \times 360$ provides the best trade-off even for this resolution, even for notable cases of motion. We also found that an extreme change of the object size is the most likely case for a failed tracking, whereas occlusion and motion cases are usually accurate. Another finding was a notable difficulty in tracking very thin objects with a poor texture. Both observations, as well as the measured success values, indicate that the bounding box was not fitting the object well, although overall the tracking went well.

4.2. Case studies

The improved strategy HGC and the technical realisation was developed by analysing cases of particularly difficult object morphology and of failed trackings. We studied many cases from the categories above and in the following will present the most representative and insightful ones.

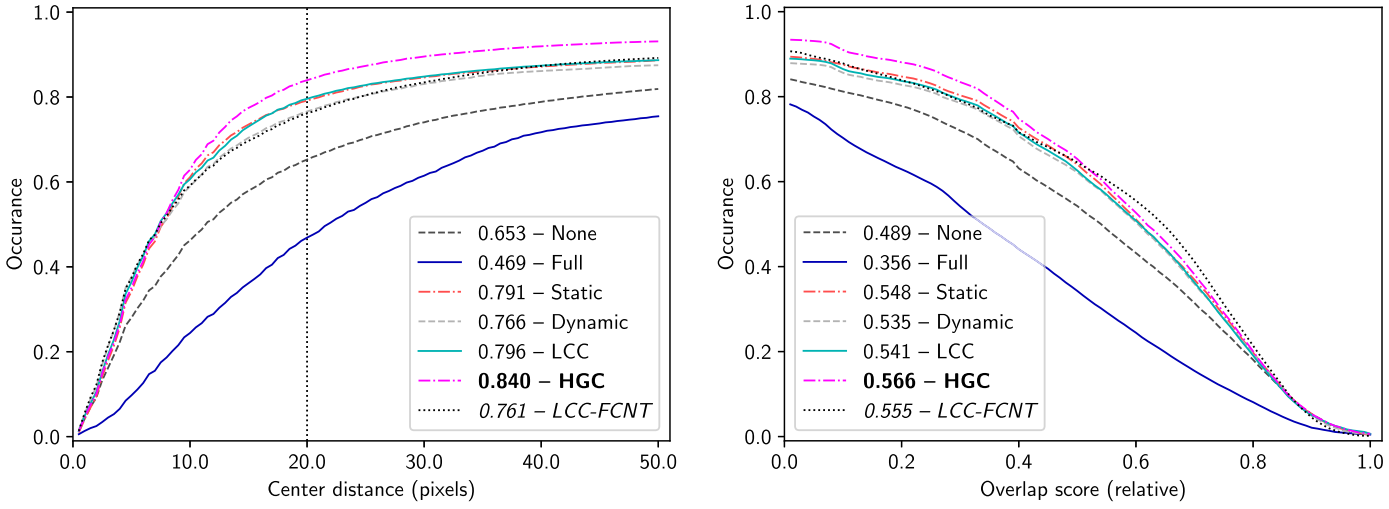
4.2.1. Motion in the OOTB

A common issue in the OOTB is fast motion and motion blur because it severely disrupts the model for the baseline update strategies. Fig. 6 illustrates how poor quality samples result in model corruption. In Fig. 6b, rapid camera movement and the resulting motion blur produces a misplaced prediction. Because of the low confidence, an update is executed, training the model to predict a position behind the tracked person (Fig. 6c). With the HGC strategy, the updates are prevented until a less blurry image in a later frame, seen in Fig. 6d, produces a better training sample that gradually updates the model to recognise the person in a blurry image.

[7] Current results: http://tracking.cs.princeton.edu/eval.php.

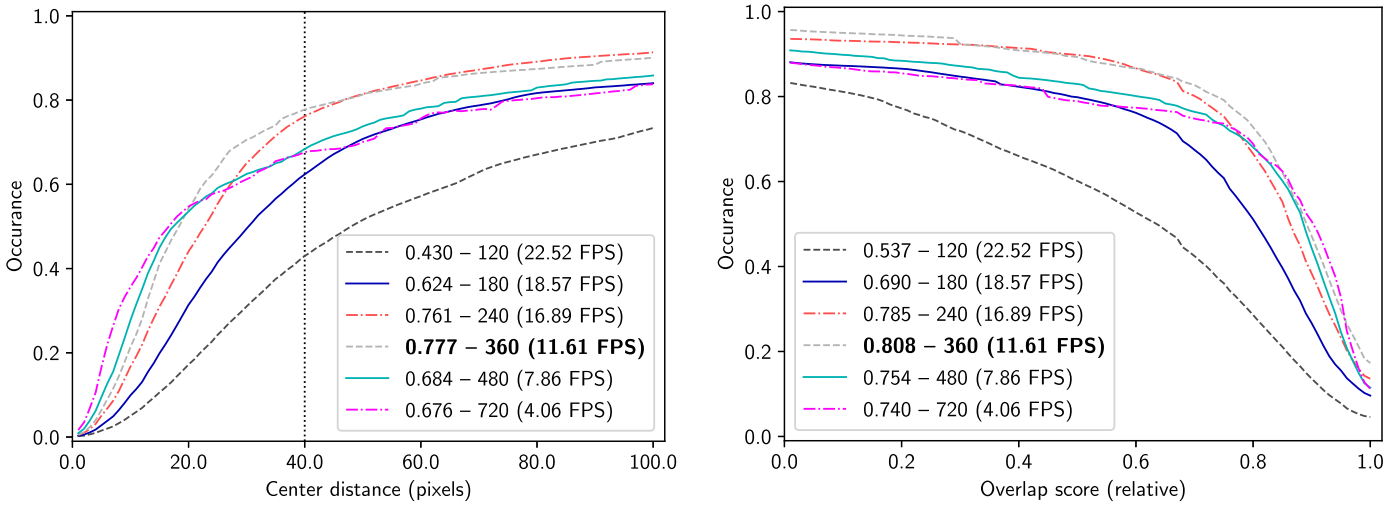
Table 1
Precision (left) and success (right) on individual attributes over different update strategies in HIOB.

Attribute	None	Full	Static	Dynamic	LCC	HGC	None	Full	Static	Dynamic	LCC	HGC
All	0.653	0.469	0.791	0.766	0.796	0.840	0.489	0.356	0.548	0.535	0.541	0.566
Background clutter	0.579	0.477	0.748	0.693	0.750	0.790	0.424	0.363	0.518	0.494	0.503	0.519
Deformation	0.561	0.452	0.716	0.620	0.646	0.733	0.441	0.306	0.513	0.440	0.456	0.506
Fast motion	0.598	0.547	0.718	0.790	0.768	0.794	0.513	0.419	0.576	0.598	0.609	0.625
Motion blur	0.599	0.471	0.808	0.646	0.720	0.847	0.549	0.401	0.604	0.584	0.595	0.681
Low resolution	0.719	0.575	0.838	0.941	0.948	0.936	0.365	0.183	0.432	0.432	0.432	0.423
Occlusion	0.599	0.410	0.735	0.681	0.724	0.800	0.464	0.328	0.512	0.494	0.500	0.553
Out of view	0.608	0.344	0.712	0.662	0.773	0.814	0.505	0.326	0.572	0.591	0.613	0.642
Scale variation	0.624	0.378	0.776	0.739	0.772	0.845	0.448	0.292	0.505	0.488	0.490	0.534



(a) Precision (value: rating up to 20 pixels). (b) Success (value: area under curve).

Fig. 4. Comparison of different update strategies in the HIOB framework versus the FCNT framework on the OOTB using the conventional metric as described in [17], with 20 pixels as the threshold for "good". The LCC strategy was tested in the original FCNT implementation from Wang et al. [6], and replicated within HIOB; HGC presents our proposed high-gain strategy.



(a) Precision (value: rating up to 40 pixels). (b) Success (value: area under curve).

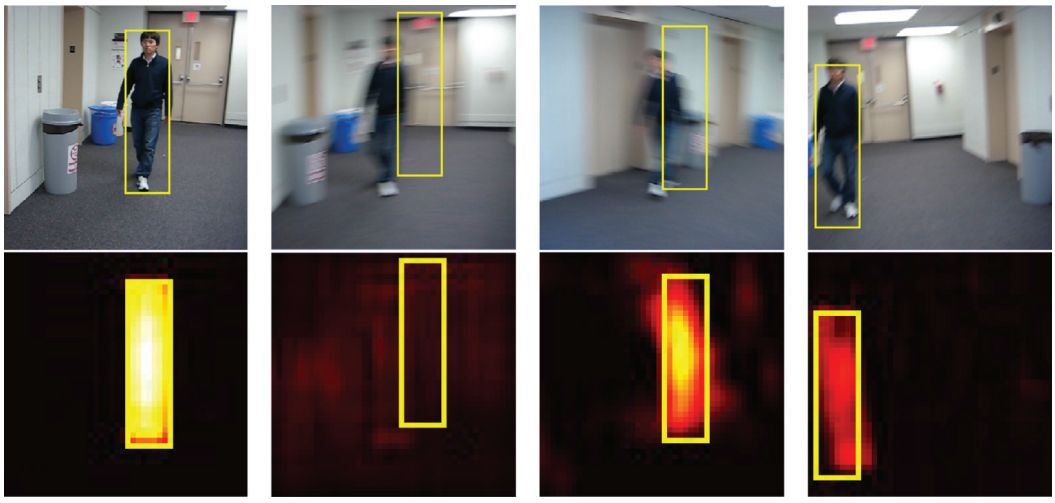
Fig. 5. Comparison of different SROI sizes in the HIOB frameworks on the NICO dataset, using 40 pixels as the threshold for "good".

4.2.2. Occlusion in the OOTB

Another difficulty was occlusion and distraction, such that the object or a large portion of it is occluded or even a second similar object occurs in the same region. In Fig. 7 we present how the high gain strategy leads to a continuous tracking despite a similar objects moving in front of the tracked object. Although the frames between the full visibility of the tracked object (Fig. 7b–c) include strong occlusion, the tracking does not get distracted.

4.2.3. Motion, occlusion, and size in the NICO scenario

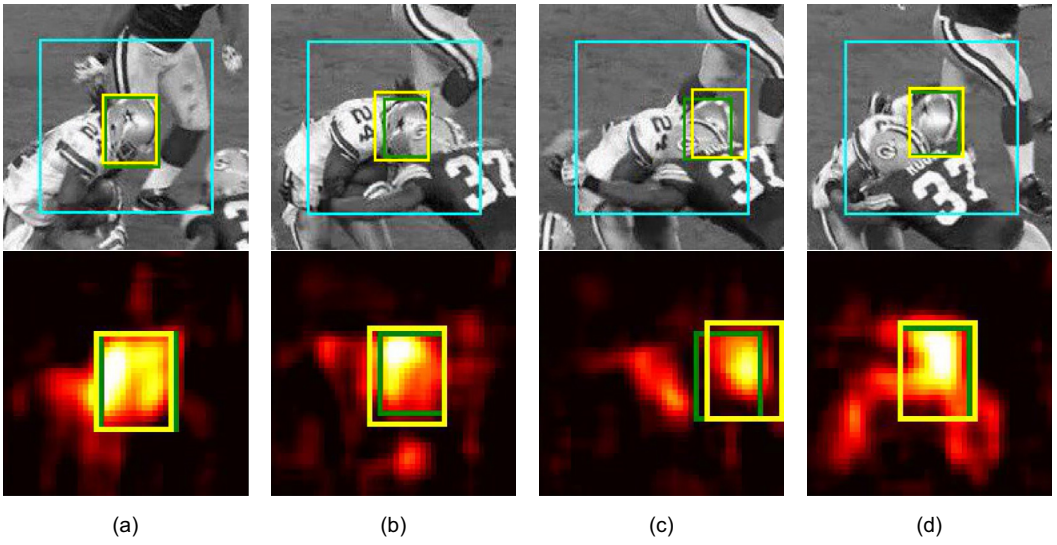
In the developmental robot scenario with NICO, occlusion was usually a strong factor in the first frames, where the robot manipulated the object. Depending on the characteristic of the object (e.g., heavy with large friction versus round), different motions were observed showing the tracking remained accurate. However, in extreme cases like fast falling thin and simple-textured objects, the tracking could get lost (see Fig. 8a–b). The previously observed



The LCC update strategy increasingly leads to predictions of low confidence, while HGC only utilises samples with a high gain:
a) Initial frame shows a solid prediction.
b) Misplaced prediction caused by motion blur. Update executed on prediction with *conf* = 0.
c) The model was trained to predict the position next to the object.
d) The HGC strategy waits for a less blurry image to execute an update.

Top: ROI with prediction in yellow; Bottom: corresponding prediction mask.

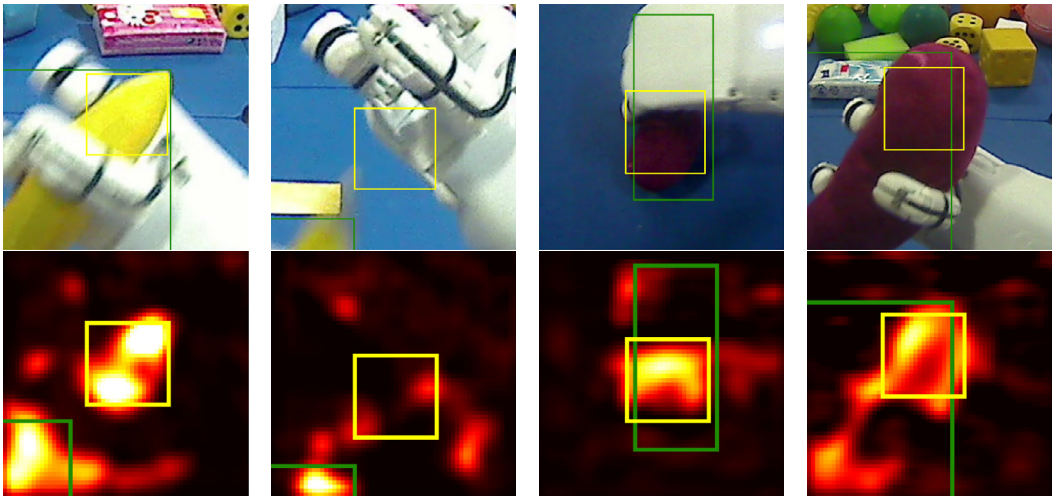
Fig. 6. Illustration of update strategies in HIOB under motion and motion blur.



Sample shows a particularly difficult scene in low quality and grayscale:
a) Second instance of tracked object (helmet of a football player) enters ROI.
b–c) Second instance strongly occludes and distracts from tracked object.
d) Tracking with HGC remains on desired object.

Top: larger frame section with ROI in cyan, prediction in yellow, and ground truth in green;
Bottom: corresponding prediction mask (most probable areas of the object's position), again with prediction in yellow and ground truth in green.

Fig. 7. Representative example of HIOB performing under occlusion and distraction.



Examples of trackings where the size changed quickly w.r.t. the initial frames.
a–b) Object with simple texture was moved towards the camera but suddenly dropped out of the hand fast and got lost because the change between two frames was too drastic and not captured within the SROI.
c–d) Strongly occluded object was correctly tracked despite model updates with small parts only.

Top: ROI with prediction in yellow and ground truth in green;
Bottom: corresponding prediction mask.

Fig. 8. Representative examples of HIOB operating in extreme cases for size change plus motion or occlusion in the NICO scenario.

cases of huge size differences led to strongly reduced overlaps to the ground truths and thus to a deterioration of the learned model because of updates that include only parts of the objects. Nevertheless, if the texture was not too simple HIOB was able to handle these cases mostly well since the selected updates provided a good abstraction of the object (see Fig. 8c–d). This shows that the NICO scenario introduces unforeseen conditions and the need for a better online adaptation of the bounding box.

5. Discussion

We presented HIOB, a hierarchical modular object tracking framework, based on the FCNT algorithm by Wang et al. [6]. Integrating a pre-trained CNN with an on-line CNN allows running cheaper update steps for specific object models on top of a strong feature extraction. The flexibility and effectiveness of the approach show a good performance on recent benchmarks and at the same time allows for tracking in real-world and real-time settings.

5.1. Update strategy for tracking under occlusion and motion

The strategy of updating the on-line CNN in cases of high gain improved the performance notably, because it combines the strengths of time-based and change-based samples, but mitigates negative effects. While updating only after a certain number of frames reduces the computational costs during the other frames [6], it is clearly inadequate for fast changing object appearances. A low confidence indicates that the model is inaccurate for the current appearance, but could also mean that the object is heavily occluded or subject to strong motion blur. In these cases, the especially low confidence would likely cause poor samples that could deteriorate the model. Thus, in our high-gain update strategy (HGC), these samples are avoided, showing a strong impact on these occlusion and motion cases, but still handling different morphology well.

5.2. Tracking in real time for developmental robot settings

With the flexible and modular design, HIOB is suited to integrate different components for, e.g., feature extraction or candidate generation, depending on needs for accuracy versus speed. Novel models can be included similarly easy as novel realisations of previously expensive tensor computations. For real-world approaches, such as in developmental robot scenarios [13], these modules, sensor-dependent parameters like the SROI size, and context-dependent characteristics such as candidate numbers or the target FPS, can be shaped for a good compromise. HIOB yields top results in current benchmarks like OOTB and PTB but can provide a similar performance in our NICO-object interaction setting. Here, the SROI size is particularly important because of the increased computational costs in relation to the camera resolution and the scene characteristics, but with a good balance, high precision and success rates would be observed.

5.3. Future work

In order to keep the computation time low and at the same time maintain or improve the precision of the tracking we can enhance the framework further in two directions. On the one hand, we can adopt other recent and upcoming concepts for feature extractions and candidate generation. For example, including residual connections in the pre-trained layers has been shown to result in similar or higher accuracies on complex object recognition tasks while remaining computationally cheap. Networks such as ResNet-50 or Inception-v4 have been shown to provide very good object models with a reduced computational complexity [12,18]. As another example, candidates can be generated by actually modelling the object movement and detected occlusions [19,20].

On the other hand, the masking and prediction can get advanced in order to more precisely reflecting the current object shape. Currently, the bounding box is determined by choice, which is given during initialisation of the tracking, in order to keep the costs low by avoiding additional heuristics. This is fine for most objects, nevertheless, for extreme transformation such as scale in perspective or rotation of long and slim objects, the model can deteriorate. Thus, a viable solution is to create additional candidate locations by slightly rotating as well as increasing or decreasing each considered object location bounding box and evaluate its score similar to the other candidates. These candidates would not mean additional complex computations but yield feasible estimates for changes over time.

6. Conclusion

A continuous model for object tracking, such as our proposed HIOB, can achieve a performance comparable to models that are constantly reinitialised when the model utilises a smart update strategy. Our suggested strategy prevents disrupting the model by excluding poor quality data samples and simultaneously reduces the need for redundant updates. Since all components have been optimised for efficient and effective computation, HIOB is capable of real-time tracking on normal consumer computers. Overall, this leads to a significant improvement in the tracking performance and thus opens up applications that demand fluent and robust tracking. This makes HIOB particularly interesting for the developmental robotics community, where robots are simulating the development of infants' cognitive capabilities, including active perception, language acquisition, and higher-order planning.

Acknowledgements

The authors gratefully acknowledge partial support from the German Research Foundation (DFG) under project CML (TRR 169) and from the NVIDIA Corporation. We also thank Erik Strahl for the support in working with the NICO robot.

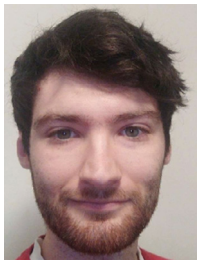
References

[1] Z. Kalal, K. Mikolajczyk, J. Matas, Tracking-learning-detection, IEEE Trans. Pattern Anal. Mach. Intell. 34 (7) (2012) 1409–1422.
[2] B. Babenko, M.H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1619–1632.
[3] H. Li, Y. Li, F. Porikli, DeepTrack: Learning discriminative feature representations online for robust visual tracking, IEEE Trans. Image Process. 25 (4) (2016) 1834–1848.
[4] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[5] S. Song, J. Xiao, Tracking revisited using RGBD camera: unified benchmark and baselines, in: Proceedings of the ICCV, 2013, pp. 233–240.
[6] L. Wang, W. Ouyang, X. Wang, H. Lu, Visual tracking with fully convolutional networks, in: Proceedings of the ICCV, 2015, pp. 3119–3127.
[7] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556.
[8] S. Hong, T. You, S. Kwak, B. Han, Online tracking by learning discriminative saliency map with convolutional neural network, in: Proceedings of the ICML, 2015, pp. 597–606.
[9] X. Zhou, L. Xie, P. Zhang, Y. Zhang, Online object tracking based on cnn with metropolis-hasting re-sampling, in: Proceedings of the ICMR, 2015, pp. 1163–1166.
[10] S. Hannuna, M. Camplani, J. Hall, M. Mirmehdi, D. Damen, T. Burghardt, A. Paiement, L. Tao, DS-KCF: a real-time tracker for RGB-D data, J. Real-Time Image Process. online (2016) 1–20.
[11] P. Springstübe, S. Heinrich, S. Wermter, Continuous convolutional object tracking, in: Proceedings of the 26th ESANN, 2018, pp. 73–78.
[12] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE CVPR, 2016, pp. 770–778.
[13] A. Cangelosi, M. Schlesinger, Developmental Robotics: From Babies to Robots, The MIT Press, Cambridge, US, 2015.

[14] D. Vernon, C.V. Hofsten, L. Fadiga, A Roadmap for Cognitive Development in Humanoid Robots, Vol. 11 of Cognitive Systems Monographs, Springer Science & Business Media, Cambridge, US, 2011.
[15] K.C. Soska, K.E. Adolph, S.P. Johnson, Systems in development: motor skill acquisition facilitates three-dimensional object completion, Dev. Psychol. 46 (1) (2010). 129
[16] M. Kerzel, E. Strahl, S. Magg, N. Navarro-Guerrero, S. Heinrich, S. Wermter, NICO - neuro-inspired companion: a developmental humanoid robot platform for multimodal interaction, in: Proceedings of the IEEE RO-MAN, 2017, pp. 113–120.
[17] Y. Wu, J. Lim, M.H. Yang, Online object tracking: a benchmark, in: Proceedings of the IEEE CVPR 2013, 2013, pp. 2411–2418.
[18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE CVPR, 2016, pp. 2818–2826.
[19] K. Meshgi, S.i. Maeda, S. Oba, H. Skibbe, Y.z. Li, S. Ishii, An occlusion-aware particle filter tracker to handle complex and persistent occlusions, Comput. Vis. Image Underst. 150 (2016) 81–94.
[20] N. An, S.Y. Sun, X.G. Zhao, Z.G. Hou, Remember like humans: visual tracking with cognitive psychological memory model, Int. J. Adv. Robot. Syst. 14 (1) (2017) 1–9.

Stefan Heinrich received his Diplom (German MSc) in computer science and cognitive psychology from the University of Paderborn, and his PhD in Computer Science from the Universität Hamburg, Germany. He is a postdoctoral research associate at Knowledge Technology, Universität Hamburg in the international collaborative research centre Crossmodal Learning (TRR-169). His research interest is located in between artificial intelligence, cognitive psychology, and computational neuroscience. Here, he aims to explore computational principles in the brain to foster our fundamental understanding of the brain's mechanisms but also to exploit them in developing intelligent systems.

Peer Springstübe studied computer science, physics, and philosophy at the Universität Hamburg, where he graduated with his Diplom (German MSc) in Computer Science. His studies and research focused on embedded systems, object tracking, and convolutional neural networks. Currently, he is pursuing an industry career with a focus on intelligent embedded systems. His research interests include intelligent behaviour and the emergence of knowledge in autonomous, intelligent systems.



Tobias Knöppler received his BSc in Computer Science at the University of Hamburg where he was working on object tracking for robotic applications. He is pursuing an industry career with a focus on innovative technologies. His research interests include neural networks, artificial intelligence, agent-oriented programming and intelligent assistive technologies.

Matthias Kerzel received his Diplom (German MSc) and PhD in computer science from the Universität Hamburg, Germany. He currently works as a postdoctoral research associate at Knowledge Technology, Universität Hamburg in the context of the international collaborative research centre Crossmodal Learning (TRR-169). His research interests include artificial intelligence and developmental, humanoid robotics. In these areas, he currently focuses on neurocognitive models for deep reinforcement learning and the autonomous development of interactive multimodal perception and sensorimotor abilities through interaction with the environment.

Stefan Wermter is Full Professor at the Universität Hamburg, Germany, and Director of the Knowledge Technology Institute. His main research interests are in the fields of neural networks, hybrid systems, neuroscience-inspired computing, cognitive robotics and natural communication. He has been the general chair for the International Conference on Artificial Neural Networks 2014. He is an associate editor of the journals 'Transactions on Neural Networks and Learning Systems', 'Connection Science', and 'International Journal for Hybrid Intelligent Systems' and he is on the editorial board of the journals 'Cognitive Systems Research', 'Cognitive Computation' and 'Journal of Computational Intelligence'. Currently, he serves as co-coordinator of the international collaborative research centre on Crossmodal Learning (TRR-169) and is the coordinator of the European Training Network SECURE on safety for cognitive robots.