




Universitat Autònoma de Barcelona

ADVERTIMENT. L'accés als continguts d'aquesta tesi queda condicionat a l'acceptació de les condicions d'ús establertes per la següent llicència Creative Commons:  http://cat.creativecommons.org/?page_id=184

ADVERTENCIA. El acceso a los contenidos de esta tesis queda condicionado a la aceptación de las condiciones de uso establecidas por la siguiente licencia Creative Commons:  <http://es.creativecommons.org/blog/licencias/>

WARNING. The access to the contents of this doctoral thesis it is limited to the acceptance of the use conditions set by the following Creative Commons license:  <https://creativecommons.org/licenses/?lang=en>



**Universitat Autònoma
de Barcelona**

Towards End-to-End Networks for Visual Tracking in RGB and TIR Videos

A dissertation submitted by **Lichao Zhang** at Universitat Autònoma de Barcelona to fulfil the degree of **Doctor of Philosophy**.

Bellaterra, September 12, 2019

Director	Dr. Joost van de Weijer Centre de Visió per Computador Universitat Autònoma de Barcelona (UAB)
Co-Directors	Dr. Abel Gonzalez-Garcia Centre de Visió per Computador Dr. Fahad Shahbaz Khan Department of Electrical Engineering Linköping University
Thesis Committee	Dr. Juan Carlos SanMiguel Electronic Technology and Communications Department Universidad Autónoma de Madrid (UAM) Dr. Daniel Ponsa Computer Science Department Universitat Autònoma de Barcelona (UAB) Dr. Michael Felsberg Department of Electrical Engineering Linköping University Dr. Jordi Gonzalez Centre de Visió per Computador Universitat Autònoma de Barcelona (UAB) Dr. Javier Vázquez Corral Departament de Tecnologies de la Informació i les Comunicacions Universitat Pompeu Fabra



This document was typeset by the author using \LaTeX 2 ϵ .

The research described in this book was carried out at the Centre de Visió per Computador, Universitat Autònoma de Barcelona. Copyright © 2019 by **Lichao Zhang**. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the author.

ISBN: 978-84-945373-1-8

Printed by Ediciones Gráficas Rey, S.L.

Acknowledgements

The universe has been an ocean of unpredictable truth and knowledge. Fortunately I have the honour to explore the mysteries inside of it even though what I have researched is only a drop of it. I feel the excitement when I discover some real truth happened in the real world. During my PhD adventure, I met many great and kind people. With their help, I could go through this wonderful journey successfully.

Firstly, I want to thank my supervisor Joost van de Weijer. I learned a lot from him, especially from his attentiveness and concentration towards research. Thanks to his patience and guidance, I could pass through the bottleneck occurred in the previous two years and keep my project moving forward smoothly. Sincerely, he gave me a lot of freedom in pursuing my goal in academic research. The interesting thing that impressed me the most is his exact predictions on the research trends. This was proven specially useful for the first two research ideas which provided successful and interesting directions of work. Furthermore, whenever I have achieved progress for my projects, I have always gotten valuable advice after meeting with him and move one more step.

For my co-supervisor, Fahad Shahbaz Khan, my most sincere thanks too. The most impressive thing is that he could always provide positive affirmation and useful comments. When we met on ECCV, we had a live meeting about the reviews of TIP with Martin present. I still remember his astuteness and exact judgement on how to solve the hard questions proposed by the reviewers. Another thing was that once he set a SOTA for the UpdateNet, which helped me realize what a good paper should be like for a top conference.

I also extend my thanks to Martin Danelljan. I really admire his pure manner both in academic and personal kindness. As he is well-known and in the top of the tracking community, every cooperation with him gives me the feeling that I can achieve anything. That is really cool and spiritually motivating! When we were preparing the TIP, his help for the motion feature significantly improved the performance of our tracker. When I wanted to participate in the VOT challenge, I was motivated by his work, and he was generous to share his insights.

I also have to largely thank my co-supervisor, Abel Gonzalez-Garcia, who has been an ideal supervisor at my side. His positive attitude always encourages me and makes me feel better about my research, as well as gets me back on track

whenever I drifted. When discussing difficult problems, he always seems to have clever ideas which inspire me and give me something to think. His objective and positive guidance can generate power to move on. One Saturday at CVC, that was being a bit difficult for my project and I had a meeting with him about some exciting ideas. The discussion with him led to the usage of skip connection for the UpdateNet and also inspired me to improve my work.

The inspiration and support from my colleagues, Yaxing Wang, Xialei Liu and Lu Yu have motivated me to push myself to be better. Thanks to Marc Masana, who often took us to many interesting places to experience the local life in Spain when we first arrived. To the people who came to CVC the same year as me and have been doing their PhD in our group. Lei Kang has helped me a lot with coding, especially with python, but mainly in my personal life. I enjoy the time when we play basketball and have dinner together on Friday after work as we both live in Cerdanyola.

Lastly but most importantly, I want to thank my family for their unlimited support and understanding. My mother Jinlian Wei, my father Xinfu Zhang, my two sisters, Minting Zhang and Liting Zhang. My journey would have never been the same without you.

Abstract

As a fundamental research topic, visual tracking plays an important role in computer vision. It has been widely applied in many fields, including autonomous driving, navigation, and robotics. The target of visual tracking is to estimate the trajectory of an object in a sequence of images, where the object is selected manually in the first frame. Tracking is regarded as a difficult task because real-world videos exhibit a large range of variations. In recent years end-to-end training of deep learning methods has dominated tracking research. Visual tracking can be applied to different modalities, such as RGB and thermal infrared (TIR).

In this thesis, we identify several problems of current tracking systems. The lack of large-scale labeled datasets hampers the usage of deep learning, especially end-to-end training, for tracking in TIR images. Therefore, many methods for tracking on TIR data are still based on hand-crafted features. This situation also happens in multi-modal tracking, e.g. RGB-T tracking. Another reason, which hampers the development of RGB-T tracking, is that there exists little research on the fusion mechanisms for combining information from RGB and TIR modalities. One of the crucial components of most trackers is the update module. For the currently existing end-to-end tracking architecture, e.g. Siamese trackers, the online model update is still not taken into consideration at the training stage. They use no-update or a linear update strategy during the inference stage. While such a hand-crafted approach to updating has led to improved results, its simplicity limits the potential gain likely to be obtained by learning to update.

To address the data-scarcity for TIR and RGB-T tracking, we use image-to-image translation to generate a large-scale synthetic TIR dataset. This dataset allows us to perform end-to-end training for TIR tracking. Furthermore, we investigate several fusion mechanisms for RGB-T tracking. The multi-modal trackers are also trained in an end-to-end manner on the synthetic data. To improve the standard online update, we pose the updating step as an optimization problem which can be solved by training a neural network. Our approach thereby reduces the hand-crafted components in the tracking pipeline and sets a further step in the direction

of a complete end-to-end trained tracking network which also considers updating during optimization.

Extensive experiments on several benchmark datasets from the RGB, TIR and RGB-T modalities demonstrate the effectiveness of our proposed methods. Specifically, synthetic TIR data is effective for end-to-end training, our fusion mechanisms outperform the single modality counterparts, and our update network outperforms the standard linear update.

Key words: *computer vision, deep learning, visual tracking, multi-modal tracking, end-to-end training, learning to update*

Resumen

Siendo un tema de investigación fundamental, el seguimiento visual juega un importante papel en la visión por computador. Se ha aplicado extensamente en diversos campos, incluyendo la conducción autónoma, navegación y robótica. El objetivo del seguimiento visual es estimar la trayectoria de un objeto en una secuencia de imágenes, habiendo seleccionado manualmente el objeto en el primer fotograma del video. El seguimiento se considera una tarea compleja debido a la gran gama de variaciones que presentan los videos del mundo real. En los últimos años, los métodos de aprendizaje profundo entrenados de extremo-a-extremo han dominado la investigación sobre seguimiento. El seguimiento visual se puede aplicar a diferentes modalidades tales como RGB o infrarrojo térmico (TIR).

En esta tesis, identificamos varios problemas de los sistemas de seguimiento actuales. La falta de conjuntos de datos etiquetados a gran escala dificulta el uso del aprendizaje profundo, especialmente en relación al entrenamiento de extremo-a-extremo para el seguimiento de imágenes TIR. Por lo tanto, numerosos métodos para el seguimiento en TIR todavía se basan en representaciones diseñadas manualmente. Esta situación también ocurre en el seguimiento multimodal, como por ejemplo, el seguimiento en RGB-T. Otra razón que dificulta el desarrollo del seguimiento RGB-T es que existe poca investigación sobre los mecanismos de fusión para combinar imágenes de modalidades RGB y TIR. Por otra parte, uno de los componentes más importantes de la mayoría de los seguidores es el módulo de actualización. En las arquitecturas de seguimiento de extremo-a-extremo actuales, como por ejemplo los seguidores Siameses, la actualización en línea del modelo no se tiene en cuenta durante la etapa de entrenamiento. Suelen utilizar una estrategia de actualización lineal durante la etapa de inferencia, o no actualizan el modelo en absoluto. A pesar de los positivos resultados obtenidos mediante esta actualización diseñada a mano, su simplicidad limita la ganancia potencial que se podría obtener al aprender a actualizar de manera automática.

Para abordar la escasez de datos para el seguimiento TIR y RGB-T, proponemos la traducción de imagen-a-imagen para generar un conjunto de datos TIR sintéticos

a gran escala. Este conjunto de datos nos permite realizar un entrenamiento de extremo-a-extremo para el seguimiento TIR. Además, investigamos varios mecanismos de fusión para el seguimiento RGB-T. Los seguidores multimodales también reciben entrenamiento de extremo-a-extremo sobre los datos sintéticos. Para mejorar la actualización en línea estándar, planteamos la tarea de actualización como un problema de optimización que puede resolverse mediante el entrenamiento de una red neuronal. Por lo tanto, nuestro enfoque reduce los componentes diseñados a mano en el proceso de seguimiento y da un paso más en la dirección de una red de seguimiento entrenada de extremo-a-extremo que incluye la actualización durante la optimización.

Extensos experimentos en varios conjuntos de datos de referencia de las modalidades RGB, TIR y RGB-T demuestran la eficacia de los métodos propuestos. Específicamente, los datos sintéticos de TIR son efectivos para el entrenamiento de extremo-a-extremo, nuestros mecanismos de fusión superan a los equivalentes de modalidad única, y nuestra red de actualización supera a la actualización lineal estándar.

Palabras clave: *visión por computador, aprendizaje profundo, seguimiento visual, seguimiento multimodal, entrenamiento de extremo-a-extremo, aprendizaje de la actualización*

Resum

Sent un tema de recerca fonamental, el seguiment visual juga un important paper en la visió per computador. S'ha aplicat extensament en diversos camps, incloent la conducció autònoma, navegació i robòtica. L'objectiu del seguiment visual és estimar la trajectòria d'un objecte en una seqüència d'imatges, havent seleccionat manualment l'objecte en el primer fotograma del vídeo. El seguiment es considera una tasca complexa a causa de la gran quantitat de variacions que presenten els vídeos del món real. En els últims anys, els mètodes d'aprenentatge profund entrenats d'extrem a extrem han dominat la investigació sobre seguiment. El seguiment visual es pot aplicar a diferents modalitats com ara RGB o infraroig tèrmic (TIR).

En aquesta tesi, identifiquem diversos problemes dels sistemes de seguiment actuals. La manca de conjunts de dades etiquetats a gran escala dificulta l'ús de l'aprenentatge profund, especialment en relació a l'entrenament d'extrem a extrem per al seguiment d'imatges TIR. Per tant, nombrosos mètodes per al seguiment en TIR encara es basen en representacions dissenyades manualment. Aquesta situació també ocorre en el seguiment multimodal, com per exemple, el seguiment en RGB-T. Una altra raó que dificulta el desenvolupament del seguiment RGB-T és que existeix poca investigació sobre els mecanismes de fusió per combinar imatges de modalitats RGB i TIR. D'altra banda, un dels components més importants de la majoria dels seguidors és el mòdul d'actualització. En les arquitectures de seguiment d'extrem a extrem actuals, com ara els seguidors Siameses, l'actualització en línia del model no es té en compte durant l'etapa d'entrenament. Solen utilitzar una estratègia d'actualització lineal durant l'etapa d'inferència, o bé no actualitzen el model en absolut. Malgrat els positius resultats obtinguts mitjançant aquesta actualització dissenyada manualment, la seva simplicitat limita el guany potencial que es podria obtenir en aprendre a actualitzar de manera automàtica.

Per abordar l'escassetat de dades per al seguiment TIR i RGB-T, proposem la traducció d'imatge a imatge per generar un conjunt de dades TIR sintètiques a gran escala. Aquest conjunt de dades ens permet realitzar un entrenament d'extrem a extrem per al seguiment TIR. A més, investiguem diversos mecanismes de fusió

per al seguiment RGB-T. Els seguidors multimodals també reben entrenament d'extrem a extrem sobre les dades sintètiques. Per millorar l'actualització en línia estàndard, plantegem la tasca d'actualització com un problema d'optimització que es pot resoldre mitjançant l'entrenament d'una xarxa neuronal. Per tant, el nostre enfocament redueix els components dissenyats a mà en el procés de seguiment i fa un pas més en la direcció d'una xarxa de seguiment entrenada d'extrem a extrem que inclou l'actualització durant l'optimització.

Extensos experiments en diversos conjunts de dades de referència de les modalitats RGB, TIR i RGB-T demostren l'eficàcia dels mètodes proposats. Específicament, les dades sintètiques de TIR són efectives per a l'entrenament d'extrem a extrem, els nostres mecanismes de fusió superen els equivalents de modalitat única, i la nostra xarxa d'actualització supera l'actualització lineal estàndard.

Paraules clau: *visió per computador, aprenentatge profund, seguiment visual, seguiment multimodal, entrenament d'extrem a extrem, aprenentatge de la actualització*

Contents

Abstract (English/Spanish/Catalan)	iii
List of figures	xv
List of tables	xxi
1 Introduction	1
1.1 Towards End-to-End Training for Visual Tracking in RGB and TIR Videos	3
1.1.1 Synthetic Data Generation for End-to-End Thermal Infrared Tracking	3
1.1.2 Multi-Modal Fusion for End-to-End RGB-T Tracking	4
1.1.3 Learning the Model Update for Siamese Trackers	5
1.2 Objectives and Approach	6
1.2.1 Synthetic Data Generation for End-to-End TIR Tracking	7
1.2.2 Multi-Modal Fusion for End-to-End RGB-T Tracking	8
1.2.3 Learning the Model Update for Siamese Trackers	10
2 Visual Tracking	13
2.1 Visual Trackers	13
2.1.1 Optimization Based Trackers	14

2.1.2	Siamese Network Based Trackers	17
2.1.3	Optimization and Siamese Network Based Trackers	20
2.2	Datasets	22
2.2.1	RGB Datasets	22
2.2.2	TIR Datasets	25
2.2.3	RGB-TIR Datasets	25
3	Synthetic Data Generation for End-to-End Thermal Infrared Tracking	29
3.1	Introduction	29
3.2	Related Work	31
3.2.1	DCF Tracking	31
3.2.2	TIR Tracking	33
3.2.3	Image-to-Image Translation	34
3.3	Method Overview	35
3.4	Deep Learning Features for Correlation Filter Tracking	36
3.4.1	Correlation Filter Tracking	37
3.4.2	Efficient Convolution Operators	38
3.5	Generating TIR images	40
3.5.1	Image-to-Image Translation Methods	41
3.5.2	Datasets	42
3.5.3	Implementation Details	44
3.5.4	TIR Image Translation Quality	44
3.6	Experimental Results	45
3.6.1	Datasets	45

3.6.2	Evaluation Measures and Protocol	46
3.6.3	Implementation Details	47
3.6.4	Network Layers	49
3.6.5	Network Architectures	49
3.6.6	Results on Real and Generated Data	50
3.6.7	Adding Motion Features	52
3.6.8	State-of-the-Art Comparison	53
3.6.9	TIR Data Attributes Analysis	55
3.7	Conclusion	56
4	Multi-Modal Fusion for End-to-End RGB-T Tracking	57
4.1	Introduction	57
4.2	Related Work	59
4.2.1	Single Modality Tracking	59
4.2.2	Modality Fusion Tracking	61
4.3	Baseline RGB Tracker	61
4.4	End-to-End Multi-Modal Tracking	64
4.4.1	Multi-Modal Fusion for Tracking	64
4.4.2	RGB-T Data Generation	65
4.5	Experiments	66
4.5.1	Generating the Training RGB-T Dataset	67
4.5.2	Evaluation Datasets and Protocols	67
4.5.3	Implementation Details	68
4.5.4	Analysis of Fusion Mechanisms	69

4.5.5	VOT-RGBT2019 Dataset	71
4.5.6	RGBT210 Dataset	71
4.5.7	Attribute Analysis on RGBT210 Dataset	72
4.6	Conclusions	72
5	Learning the Model Update for Siamese Trackers	73
5.1	Introduction	73
5.2	Related Work	75
5.3	Updating the Object Template	77
5.3.1	Standard Update	78
5.3.2	Learning to Update	79
5.3.3	Tracking Framework With UpdateNet	79
5.3.4	Training UpdateNet	80
5.4	Experiments	81
5.4.1	Training Dataset	81
5.4.2	Evaluation Datasets and Protocols	82
5.4.3	Implementation Details	82
5.4.4	Ablation Study	83
5.4.5	Analysis on Representation Update	85
5.4.6	Generality and Tracking Speed	86
5.4.7	Fine-tuning the Linear Update Rate	87
5.4.8	Comparison With Other Updating Strategies	88
5.4.9	LaSOT Dataset	89
5.4.10	TrackingNet Dataset	90

5.5	Conclusions	91
5.6	Difference With Offline Weighted Fusion	91
5.7	Visualization of Updating Templates	92
5.8	Change Rate for Update	93
6	Conclusions and Future directions	95
6.1	Conclusions	95
6.2	Future Directions	97
A	Publications	99
A.1	Scientific Articles	99
A.2	Contributed Code and Models	99
	Bibliography	112

List of Figures

- 1.1 **The pipeline of tracking.** Given the target state in the first frame, the tracker predicts the target locations and bounding boxes in subsequent frames. 2

- 1.2 **Visualization of the imaging difference between ‘RGB’ and ‘TIR’ for the same scene.** On the upper half, we show three exemplar videos from the RGB modality and we show their corresponding videos from TIR modality on the lower half. In the left example, the dog is covered with the shadow from the pole in the RGB modality, but in the TIR modality, this shadow disappears and the target is much clearer than that in RGB modality. In the center example, the human is also discriminative as his temperature is much higher and the shadow from the car disappears in the TIR image. Conversely, the RGB images can also provide complementary benefits over the TIR images. For example, in the right example, the appearance of the person in the TIR image is vague and is heavily influenced by the surrounded tree. While in the RGB modality, the person is clearly distinguished from the surroundings as the color and shape information is very discriminative. 4

- 1.3 **Visualization of failure samples during tracking with no-update (a) and linear update (b).** In (a), tracking is starting from left to right, where the aim is to track the gymnast. In the center and right frame, the tracking prediction is misaligned due to the complex appearance changes of the target. In (b), the target is the athlete. The tracker cannot adapt to the complex scenarios, e.g. fast motion and background clutter, as the update mechanism is a linear function. In both aforementioned cases, the tracker cannot predict the accurate location both with no-update or simple linear update. 6

1.4	Overview of our synthetic data generation and end-to-end TIR tracking. In the pipeline, firstly we use image to image translation models to generate abundant synthetic TIR videos. Then use these labeled TIR videos to train an end-to-end TIR tracker, detailed in chapter 3.	7
1.5	Overview of our end-to-end RGB-T tracking. In the pipeline, we use technique developed in Chapter 4 to generate large-scale paired synthetic RGB-T datasets to train an end-to-end RGB-T tracker. We also investigate three different fusion mechanisms in different parts of the tracker, including pixel-level, feature-level and response-level fusion as depicted in the upper part of the pipeline.	9
1.6	Overview of the tracking pipeline with <i>UpdateNet</i>. In the pipeline, we focus on the update component for the RGB tracker. We propose to use a convolutional neural network to update the object template (Chapter 5), which is more robust in complex scenarios during tracking than the linear update.	10
2.1	Fully-convolutional Siamese architecture. Figure from (Bertinetto et al., 2016b).	17
2.2	Main framework of Siamese-RPN. It includes feature extraction network and region proposal network. Figure from (Li et al., 2018b).	19
2.3	The tracking architecture of DiMP. It includes feature extraction and model predictor. Figure from (Bhat et al., 2019).	20
2.4	Selected frames from the datasets of OTB2013, OTB100, VOT, LaSOT, TrackingNet, and GOT-10k respectively.	23
2.5	Selected frames from the datasets of VOT-TIR2015 and VOT-TIR2017.	25
2.6	Selected frames from the datasets of VOT-RGBT 2019 and RGBT210.	26
3.1	Qualitative comparison of our approach trained on generated data only (red) with baseline ECO (Danelljan et al., 2017) (green) on the <i>quadrocopter2</i>, <i>car2</i> and <i>garden</i> videos. The ground truth bounding box is provided in yellow. Owing to the synthetic TIR data our model is able to follow the object successfully in case of out-of-plane rotation, partial occlusion and scale changes.	32

3.2	Average activation of filters from the first layer of pre-trained AlexNet on the test set of KAIST for RGB and TIR images.	36
3.3	Overview of our approach. (a) Image-to-image translation component (proposed by Isola et al. (2017)) for generating a large labeled synthetic TIR tracking dataset. We use blue dashed line to represent the baseline RGB training model and the green dashed line represents our proposed synthetic data training model. After the translation of RGB data to TIR data, we acquire enough suitable data for end-to-end training networks for TIR tracking. (b) Two-branch architecture for training the network to obtain adaptive features for TIR tracking (proposed by Valmadre et al. (2017)). The optimal correlation filter is computed in the discriminative correlation filter layer (DCFL) for the image processed in the upper branch. This filter is then applied on the image in the bottom branch.	37
3.4	Results for the two image translation methods considered: pix2pix and CycleGAN. The video frames are taken from the test set of KAIST, and have not been seen during training.	39
3.5	Histogram of the gradient magnitude for real and synthetic TIR data computed on the test set of KAIST (Hwang et al., 2013). For comparison we have also added the gradient magnitude histogram for grayscale images from which the synthetic dataset has been generated.	45
3.6	The EAO on VOT-TIR2017 (Kristan et al., 2017a) when using deep features extracted from different layers.	47
3.7	The EAO on VOT-TIR2017 (Kristan et al., 2017a) when using deep features extracted from different networks. Our synthetic data can benefit general networks for fine-tuning.	48
3.8	The success plot of one-pass evaluation (OPE) on the the VOT-TIR2017 benchmark (Kristan et al., 2017a). We show the AUC score of each tracker in the legend. The best results are obtained when using both real and generated data.	51
3.9	Performance of our tracker (generated+real) on VOT-TIR2017 (Kristan et al., 2017a) for different percentages of synthetic data. The leftmost point indicates using only real data.	52

3.10 **Qualitative comparison of our approach trained on generated and real data with state-of-the-art trackers.** CREST, TCNN, EBT and DSLT are tested on the some challenging sequences, *excavator*, *jacket*, *mixed_distractors*, *garden*, *quadrocopter2*, *boat2*, *bird* and *trees2* in VOT-TIR2017 (Kristan et al., 2017a). Yellow dashed bounding box means Groundtruth and red solid bounding box is Ours. The last two rows show failure cases of our tracker. 54

3.11 **Attribute-based comparison of our trackers with state of-the-art methods on VOT-TIR2017.** We show expected overlap measure for four attributes: camera motion, dynamics change, motion change, occlusion, size change, and others. Our trackers provide consistent improvements in case of camera motion, motion change, occlusion and others, compared to existing methods. 55

4.1 **Qualitative comparison between ‘mfDiMP’ and ‘DiMP’.** Two exemplar videos from RGB modality and TIR modality on the top and bottom separately, where DiMP performs on each of them with single modality input. Our *mfDiMP* can effectively track the object by fusing both modalities. 58

4.2 **Overview of our multi-modal fusion framework on feature-level.** We input images from RGB and TIR modalities to the feature extractor separately. Then we fuse deep features from different blocks of the backbone. Fused features from block3 and block4 are input to IoU modulation and IoU predictor. Fused features from block4 are input to model predictor to predict final response map. 62

4.3 **Precision plot and success plot by comparing our mfDiMP with the top-10 trackers on RGBT210 dataset (Li et al., 2017b)** We can see our mfDiMP outperforms DiMP with an absolute gain of 6.7% and 4.2% in terms of precision rate and success rate respectively. 68

5.1 **Qualitative comparison between model updates.** We learn to update the model template using *UpdateNet*. When combined with Siamese trackers such as SiamFC (Bertinetto et al., 2016b), our learned updating strategy can be effectively adapted to current circumstances, unlike the simple linear update commonly used. 74

5.2 **Overview of our tracking framework with UpdateNet.** (Left) The on-line update of the object template is performed by UpdateNet, which receives as input the initial ground-truth template, last accumulated template and current predicted template, and outputs updated accumulated template. (Right) Training of UpdateNet using the distance to the ground-truth object template on next frame. 77

5.3 **Visualization of accumulated templates for SiamFC.** ‘Frame’ column shows the search region and ground-truth box used to extract the templates, whose top four channels we show in ‘GT’. ‘No-update’ presents the response map resulting from applying the initial template to the search region. For ‘Linear’ and ‘UpdateNet’ strategies we also show their accumulated templates. 84

5.4 **Change rate between contiguous frames.** We present individual results for two example videos (top, middle) and average results for all videos in VOT2018. 86

5.5 **EAO vs. speed on VOT2018.** We compare our UpdateNet combined with two different Siamese trackers against the state-of-the-art methods. UpdateNet can substantially improve the tracking performance without significantly affecting the speed. 87

5.6 **EAO performance on VOT2018.** We compare our method with the state-of-the-art methods on VOT2018. Our proposed approach achieves superior performance. 88

5.7 **The linear update rate evaluation for DaSiamRPN and SiamFC on VOT2018 (Kristan et al., 2018).** The x-axis is the update rate values. The y-axis is the EAO scores on VOT protocol (Kristan et al., 2018). The red and pink dashed lines are our UpdateNet performances with DaSiamRPN and SiamFC, respectively. 89

5.8 **Evaluation on LaSOT testing set.** Normalized precision and success plots following the OPE protocol II. 90

5.9 **Training of the learned fusion weights offline.** 92

5.10 **Visualization accumulated and ground-truth templates for SiamFC.**
The first column shows the search region and the ground-truth box.
'GT' shows top four channels of the real template extracted from the
ground-truth box. For each update strategy ('Linear' and 'UpdateNet')
we show the accumulated templates and the resulting response map
when applied to the search region, respectively. 94

5.11 **Change rate between contiguous frames.** We present additional re-
sults for six example videos in VOT2018. 94

List of Tables

2.1	Comparison of several datasets in three modalities for visual tracking.	22
3.1	Datasets used for training the image-to-image translation models. We test all models using a subset of three videos from the official test set of KAIST (Hwang et al., 2013).	43
3.2	Datasets used for training the tracker, using real TIR data or generated TIR data from RGB images.	46
3.3	Comparison of different tracker variants with and without motion features. Results are on the VOT-TIR2017 benchmark (Kristan et al., 2017a) with ResNet-50 (He et al., 2016) as base network. Boldface indicates the best results. In both cases, the best results are achieved when combining both real and generated TIR data.	50
3.4	Comparison with state-of-the-art trackers on VOT-TIR2017 (Kristan et al., 2017a). Boldface indicates the best results. The results are reported in terms of expected average overlap (EAO), robustness (failure rate) and accuracy. Our proposed tracker significantly outperforms the state-of-the-art by achieving an EAO score of 0.436.	53

4.1 **Fusion mechanisms analysis on VOT-RGBT2019 (VOT challenge, 2019).** We evaluate several fusion mechanisms at different levels of DiMP (Bhat et al., 2019). The results are reported in terms of EAO, normalized weighted mean of accuracy (A), and normalized weighted mean of robustness score (R). We explicitly show the input modality for each component of the tracker. Here, ‘RGB’ and ‘TIR’ are the single modality, ‘RGB/TIR’ means each modality input separately, ‘RGB+TIR’ means that both modalities are input simultaneously, and ‘RGBT’ indicates fused features from both modalities used in the remaining of network. Finally, ‘ft’ means fine-tuning and ‘ $\times 10$ ’ means a higher learning rate for fine-tuning. The best results are highlighted in bold font. 66

4.2 **State-of-the-art comparison on VOT-RGBT2019 dataset.** Our mfDiMP improves the baseline tracker DiMP with an absolute gain of 6.4% in terms of EAO. The best results are highlighted in bold font. 70

4.3 **Attribute-based Precision Rate and Success Rate (PR/SR %) on RGBT210 dataset with several trackers.** These trackers include popular RGB trackers such as ECO and CSR, recent multi-modal fusion tracker like CMRT and SGT, and also extended RGB-T trackers from KCF and CFnet. Our tracker surpasses almost all the trackers over all the attributes. 70

5.1 **Ablation study on VOT2018 (Kristan et al., 2018).** We present several update strategies for SiamFC (Bertinetto et al., 2016b). The results are reported in terms of EAO, normalized weighted mean of accuracy (A), and normalized weighted mean of robustness score (R). ‘Skip’ column indicates the origin of the skip connection, if any. Here, K is the number of stages UpdateNet is trained for. 83

5.2 **Results for other updating strategies on VOT2016.** DSiam (Guo et al., 2017) and MemTrack (Yang & Chan, 2018) use SiamFC as base tracker. The best two results are highlighted in red and blue fonts, respectively. 88

5.3 **State-of-the-art comparison on TrackingNet.** Our UpdateNet significantly improves DaSiamRPN (Zhu et al., 2018) with an absolute gain of 3.4% and 3.9%, in terms of precision and success. The best two results are highlighted in red and blue fonts, respectively. 90

1 Introduction

Videos play a crucial role in the world. They are of the utmost importance in many industrial applications, such as autonomous driving, video surveillance, robotics, and traffic control. They are also widely present in everyday life, where people watch movies or matches, conduct video chat and play video games. One of the most important fields of video usage is in the social media: for example 300 hours of video are uploaded to YouTube every minute. YouTube receives more than 90 PB of video data every year, and has more than 7 billion videos (Youtube Statistics, 2019).

As a consequence of this enormous amount of data, video understanding is one of the main research lines of computer vision. Videos provide more information than still images as they add a temporal component through which motion and other information can be exploited. There are many tasks on video understanding such as action recognition, visual tracking, and optical flow. Among them, visual tracking, which follows the location of a selected object across frames, plays a fundamental role, and also has a wide range of applications for the real-world, including navigation, surveillance, robotics, traffic control, autonomous driving, and augmented reality. In general visual tracking involves two parts: it aims to calculate the target location using a classifier and it estimates the target bounding box using a target estimation component given its first frame state, as depicted in Figure 1.1.

Nowadays, deep learning has achieved huge success and has unleashed a revolution in computer vision. It has had a significant impact on almost all computer vision research directions, including image classification (He et al., 2016), image segmentation (Chen et al., 2017), object detection (Redmon et al., 2016), super-resolution (Dong et al., 2015), etc. The success of deep learning is mainly due to two factors: the development of new accessible hardware (GPUs) and the presence of large labelled datasets. One of the main advantages of deep learning is its capability to learn end-to-end, meaning that optimal discriminative representations are learned jointly with final classifiers.

In recent years, visual object tracking, especially tracking in color videos (re-

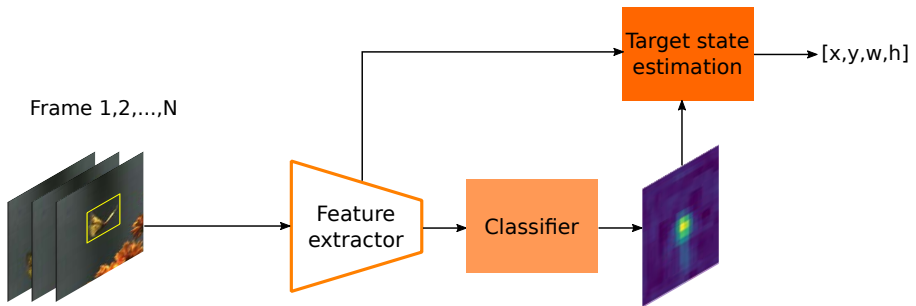


Figure 1.1 – **The pipeline of tracking.** Given the target state in the first frame, the tracker predicts the target locations and bounding boxes in subsequent frames.

ferred to as RGB tracking) has undergone profound changes (Bertinetto et al., 2016b; Bolme et al., 2010; Danelljan et al., 2017, 2019, 2014a, 2016b, 2014b; Henriques et al., 2015; Li et al., 2018b; Lukezic et al., 2017; Zhu et al., 2018). Researchers have mainly focused on RGB tracking as large datasets are available (Kristan et al., 2016a; Valmadre et al., 2018; Wu et al., 2015). The training of end-to-end networks was found to be difficult for visual tracking, and the introduction of deep learning for visual tracking was later than in many of the other research fields of computer vision. Only recently have appeared some end-to-end trackers (Bertinetto et al., 2016b; Valmadre et al., 2017) for RGB tracking. However, these trackers still contain some elements which are not optimized in an end-to-end sense, such as the online update. Training end-to-end trackers is one of the research themes of this thesis.

Another development in recent years is the wider accessibility of multi-modal data. In the tracking community special attention has been paid to the thermal infrared (TIR) modality because of its robustness to bad environmental conditions, e.g. low illumination, rain, and smog. Videos from TIR modality can provide complementary information to the information present in RGB videos. As a result RGB-T tracking is also becoming a promising direction in visual tracking. In this thesis, we investigate several aspects of TIR and multi-modal tracking.

This thesis research was performed in the turbulent period in which deep learning was introduced to RGB tracking and TIR tracking was receiving growing research attention. We identified three challenges for tracking which we describe in more detail in the following.

1.1 Towards End-to-End Training for Visual Tracking in RGB and TIR Videos

Despite the astounding development of visual tracking, there are still many open research questions. In this thesis, we focus on the two different modalities and their combination in visual tracking, i.e. RGB tracking, TIR tracking, and RGB-T tracking, aiming to achieve end-to-end training. We also propose to replace the 'hand-crafted' model update by an update which is automatically learned.

1.1.1 Synthetic Data Generation for End-to-End Thermal Infrared Tracking

Visual tracking aims to estimate a trajectory of an object through a video based on only one bounding box annotation at the beginning of the sequence. This tracking mechanism also applies to TIR tracking, with the only difference that TIR tracking uses the TIR images instead of the RGB images. The importance of TIR tracking increases as improvements of thermal infrared sensors in resolution and quality are currently being developed. The advantage of thermal images is that they are not influenced by illumination variations and shadows, for example the left and right examples in Figure 1.2. Also objects can be distinguished from the background as the background is normally colder, as depicted in the center and right examples in Figure 1.2. In addition, thermal infrared tracking can be used in total darkness, where visual cameras have no signal.

As far as we know, there are still no deep neural networks specifically trained for TIR tracking. As a consequence, the usage of hand-crafted features remains dominant for TIR tracking. For example, at present, the leading TIR trackers still employ hand-crafted features in their models. The winner (Yu et al., 2017) of VOT-TIR2017 (Kristan et al., 2017a) challenge employs HOG and motion features. Furthermore, there are other trackers (Felsberg et al., 2015; Zhu et al., 2016) with top-performance which are based on hand-crafted features. The outperforming results of these methods show that hand-crafted features are still a better choice for tracking on TIR modality.

Generally the deep neural networks are trained on a large amount of labeled training data, which allows the network to learn discriminative features for the classifier. Despite its astounding success, the impact of deep learning on generic TIR tracking has been limited. One of the key issues when employing deep features for TIR tracking is the unavailability of large-scale labeled TIR tracking data for training. Therefore, data scarcity is the crucial issue hampering end-to-end training applied on TIR tracking.



Figure 1.2 – **Visualization of the imaging difference between ‘RGB’ and ‘TIR’ for the same scene.** On the upper half, we show three exemplar videos from the RGB modality and we show their corresponding videos from TIR modality on the lower half. In the left example, the dog is covered with the shadow from the pole in the RGB modality, but in the TIR modality, this shadow disappears and the target is much clearer than that in RGB modality. In the center example, the human is also discriminative as his temperature is much higher and the shadow from the car disappears in the TIR image. Conversely, the RGB images can also provide complementary benefits over the TIR images. For example, in the right example, the appearance of the person in the TIR image is vague and is heavily influenced by the surrounded tree. While in the RGB modality, the person is clearly distinguished from the surroundings as the color and shape information is very discriminative.

1.1.2 Multi-Modal Fusion for End-to-End RGB-T Tracking

RGB-T tracking aims to predict the states of the object in videos by fusing RGB and TIR modalities (corresponding to the visible and thermal infrared spectrum data respectively), given the initial ground-truth bounding box. Normally, RGB images have the advantage that they contain high-frequency shape and texture information, which provides discriminative information for describing the objects, as shown in Figure 1.2. TIR images are sometimes superior in the sense that the image quality is more robust under unfavourable scenarios, such as adverse illumination, rain and smog. This is because the thermal sensors capture the image pixels depending on the temperature of the objects. Therefore, tracking in general can be significantly improved with the benefit of the complementary data information from different

modalities. Thus RGB-T tracking comes into existence accordingly. Specifically, images from the RGB modality and images from the TIR modality can compensate each other's deficiencies during tracking. Due to these advantages, some research works have focused on the RGB-T tracking (Li et al., 2016, 2018c, 2017b, 2018d).

But currently, there exists relatively little research on multi-modal tracking compared with RGB tracking. Among them, most are still using sparse representations, normally based on hand-crafted features for multi-modal tracking (Li et al., 2016, 2017a, 2018d; Liu & Sun, 2012). These approaches utilize the pixel intensity as the feature representation, which seriously limits their applicability for handling complex scenarios.

For comparison, some researchers design a baseline RGB-T tracker by extending a normal single modality tracker to a multi-modal fusion tracker (Li et al., 2018d). This extension is done by directly concatenating the features from RGB modality and TIR modality into a single vector, which is input to the tracker. In these extended RGB-T trackers, there are some using deep features as input. But they still use off-the-shelf features (Simonyan & Zisserman, 2014) which are pre-trained from training datasets for other tasks. So far, there is still no research on how to implement end-to-end training for RGB-T tracking. We identify two reasons for this. First, it is not obvious in what part of the tracking pipeline the fusion should be done. Ideally, we should fuse the information of the different modalities in such a way that it allows for optimal end-to-end training. Second, data scarcity of multi-modal tracking data is a major obstacle to end-to-end training. Currently, there are no large-scale aligned multi-modal datasets for training. These two issues, i.e. no specific fusion scheme and a lack of data, limit the progress of end-to-end training for multi-modal tracking.

1.1.3 Learning the Model Update for Siamese Trackers

Visual tracking has undergone a profound development during these years. Recently, the Siamese network based trackers have drawn much attention. These Siamese trackers formulate the visual object tracking problem as learning a general similarity map by cross-correlation between the feature representations learned for the target template and the search region. To ensure tracking efficiency, the offline learned Siamese similarity function is normally fixed during the running time. However, appearance changes are often large and failing to update the template could lead to early failure of the tracker, as can be seen in the exemplar video in the upper half of Figure 1.3. In such scenarios, it is important to adapt the model to the current target appearance. To accommodate this problem, more recent Siamese trackers (Li et al., 2018b; Wang et al., 2018; Zhu et al., 2018) have implemented a simple linear update strategy using a running average with a fixed learning rate (Stauffer &

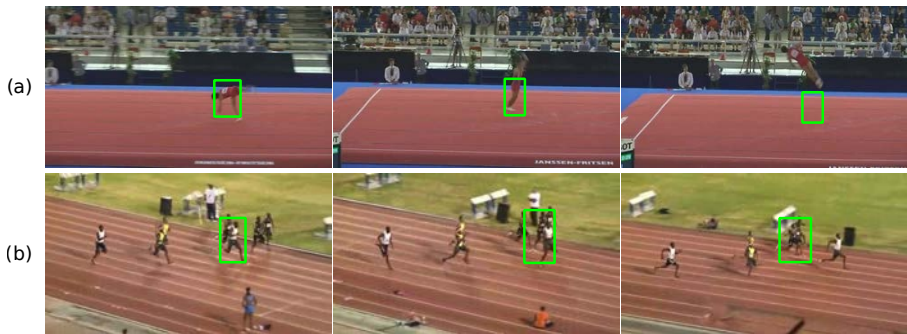


Figure 1.3 – **Visualization of failure samples during tracking with no-update (a) and linear update (b).** In (a), tracking is starting from left to right, where the aim is to track the gymnast. In the center and right frame, the tracking prediction is misaligned due to the complex appearance changes of the target. In (b), the target is the athlete. The tracker cannot adapt to the complex scenarios, e.g. fast motion and background clutter, as the update mechanism is a linear function. In both aforementioned cases, the tracker cannot predict the accurate location both with no-update or simple linear update.

Grimson, 1999).

While template averaging provides a simple means of integrating new information, it has several severe drawbacks. Firstly, the tracker cannot recover from object drift. Partially, this is caused by the fact that it loses access to the initial template, which is the only real template without doubt on the sequence. Secondly, the linear update function is constrained to a simple weighted combination of previous appearance templates. This severely limits the flexibility of the update mechanism, important when the target undergoes complex appearance changes. Considering more complex combination functions is expected to improve results.

Thus with a simple linear update, the tracker cannot adapt the object appearance changes in complex scenarios, see for example the video in the lower half of Figure 1.3. The development of more complex update schemes is therefore expected to improve tracking performance.

1.2 Objectives and Approach

Above we have indicated two challenges in TIR tracking and RGB-T tracking. We also analyze the drawbacks of linear update in RGB tracking. In this dissertation,

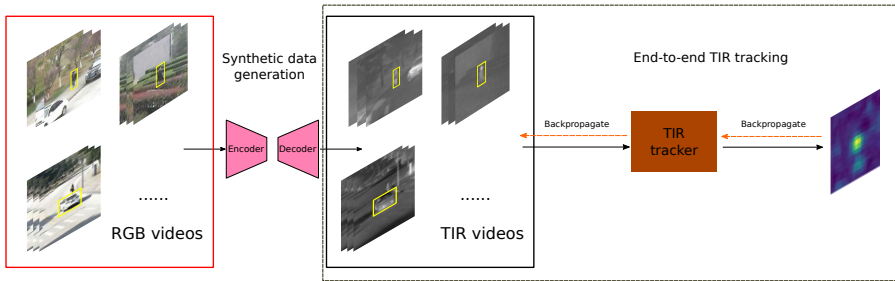


Figure 1.4 – **Overview of our synthetic data generation and end-to-end TIR tracking.** In the pipeline, firstly we use image to image translation models to generate abundant synthetic TIR videos. Then use these labeled TIR videos to train an end-to-end TIR tracker, detailed in chapter 3.

we aim to tackle these issues in order to advance end-to-end training for visual tracking in RGB and TIR videos. In section 1.2.1, we introduce our method which accommodates the issue of data-scarcity for TIR tracking. This data generation technique and end-to-end training method will be described in Chapter 3. As the issues of no specific fusion scheme and lack of data limit the progress of end-to-end multi-modal training, we will briefly explain our methods for addressing them in section 1.2.2 and will present the comprehensive approaches and experiments in Chapter 4. In section 1.2.3, we introduce our update mechanism for updating the template during tracking, which improves the tracking performance compared with the linear update. The motivation and training for the novel update mechanism will be described in Chapter 5.

1.2.1 Synthetic Data Generation for End-to-End TIR Tracking

Recently, Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) have attracted much attention as they can generate images which are indistinguishable from real images. A GAN learns a discriminative model that tries to classify whether the output image from a generative model is real or fake, while simultaneously training the generative model to fool the discriminator. Later on, image-to-image translation networks (Isola et al., 2017) have extended the GAN model and have experienced fast progress. An image-to-image translation network is able to generate samples from complex image distributions conditioned on an input image. As a consequence, they can learn a translation model between domains *from* paired images. Furthermore, these translation networks are extended to learn mappings between domains from unpaired images (Zhu et al., 2017). This work uses cycle

consistency: transferring an image to another domain and then transferring it back to the initial domain, where the second generated image should be the same as the original image. These generative networks can be used to generate large-scale synthetic data to compensate training datasets in data-scarce domains, such as depth and TIR images. In this work, we investigate on how to effectively transfer labeled RGB videos to synthetic TIR videos along with the labels by using image-to-image translation networks.

End-to-end training is generally expected to improve results, and is therefore also a desired objective for TIR tracking. However, one of the main issues, which is obstructing end-to-end training applied on TIR tracking, is the lack of large-scale labeled training TIR datasets. Inspired by the successful application of translation networks in other fields (Chen et al., 2017; Dong et al., 2015; He et al., 2016; Redmon et al., 2016), we propose to use the translation models to abundantly generate synthetic TIR data from available labeled RGB data. We explore both the usage of paired and unpaired image translation models for this purpose and find the optimal one as our final generation model. Through this generation model, we can obtain a large-scale labeled dataset of synthetic TIR sequences by transferring from corresponding RGB sequences. With the access of the large-scale TIR training dataset, we can implement end-to-end training for TIR tracking. To the best of our knowledge, we are the first to train end-to-end networks for TIR tracking.

Extensive evaluations on the latest TIR tracking challenge (Kristan et al., 2017a) verifies the efficiency of our different models trained on synthetic TIR datasets. One of the more important results shows that a tracker trained on only synthetic data can outperform the tracker trained on available real TIR data.

1.2.2 Multi-Modal Fusion for End-to-End RGB-T Tracking

RGB-T tracking has drawn much attention, as jointly using RGB and TIR images provides rich information which can assist trackers in complex tracking scenarios as depicted in Figure 1.2. But still now, most of the works on RGB-T tracking use sparse hand-crafted representations since these can effectively suppress the noise present in TIR images (Li et al., 2016, 2018c, 2017b, 2018d). Furthermore, in benchmarks (Li et al., 2018c, 2017b), some trackers using off-the-shelf deep features appear. For comparison with other trackers, they design some RGB-T baseline trackers by directly concatenating the feature representations from RGB and TIR modalities. Because these deep features are trained from the training datasets for other tasks such as image classification in computer vision, the objects cannot be described in optimal way during tracking. In conclusion, there is still no work which investigates end-to-end training for RGB-T tracking. We identify two reasons for this: the first one is that there is no specific end-to-end framework for fusing the RGB and TIR

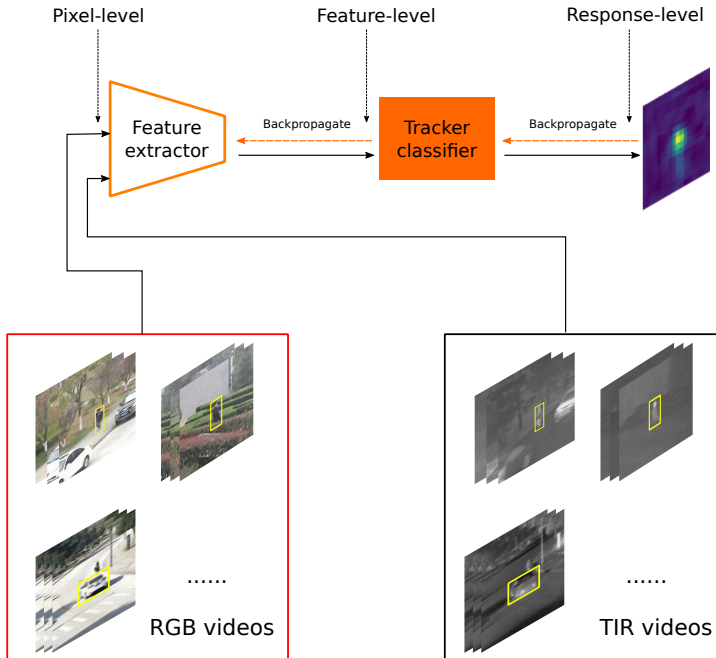


Figure 1.5 – **Overview of our end-to-end RGB-T tracking.** In the pipeline, we use technique developed in Chapter 4 to generate large-scale paired synthetic RGB-T datasets to train an end-to-end RGB-T tracker. We also investigate three different fusion mechanisms in different parts of the tracker, including pixel-level, feature-level and response-level fusion as depicted in the upper part of the pipeline.

modalities; the second one is that there are no large-scale training datasets.

We propose two methods to address the two issues. First, we investigate how to effectively fuse multi-modal data in different parts of the tracker, and find the one which can make optimal use of information from both modalities. We propose three end-to-end multi-modal fusion mechanisms, consisting of pixel-level fusion, feature-level fusion and response-level fusion. Second, we use the technique developed in Chapter 3 to generate large-scale paired synthetic RGB-T datasets. These datasets allow us to train a multi-modal tracking in an end-to-end manner.

Comprehensive experiments have been conducted to analyze the different fusion mechanisms on VOT-RGBT2019 dataset. We obtain the best results when fusing at the feature-level of the tracker. With this fusion mechanism we achieve the state-of-the-art performance on RGBT210 dataset.

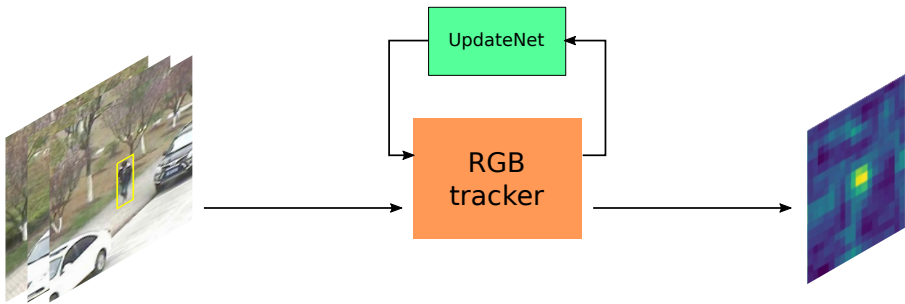


Figure 1.6 – **Overview of the tracking pipeline with *UpdateNet*.** In the pipeline, we focus on the update component for the RGB tracker. We propose to use a convolutional neural network to update the object template (Chapter 5), which is more robust in complex scenarios during tracking than the linear update.

1.2.3 Learning the Model Update for Siamese Trackers

The principle of the Siamese trackers is to learn the similarity between an object appearance template and a feature representation of a search region and finally predict a response map by cross-correlation. They extract the feature representations for the object and the search region through a deep convolutional neural network which is trained offline on a large RGB dataset. Such a deep similarity learning strategy has shown to provide discriminative representations for visual tracking. During the online tracking phase, the object template is initialized in the first frame and is then fixed for matching with the search region in the following frames. Later on, some researchers proposed to use a linear update strategy to adapt the object change with a small and fixed update rate (Valmadre et al., 2017). This update strategy accommodates the issue of large appearance changes in complex scenarios to a certain degree. But there exist several drawbacks for the linear update, as this strategy assumes a constant rate of appearance change across all frames in the video, as well as across different videos. In practice, the tracking situations can be various as it depends on a complex combination of external factors such as motion, blur and background clutter. Thus, the update requirements for the object template are greatly varied. As a result, a simple linear update is normally inadequate to cope with changing update requirements and is unable to generalize to all potentially encountered situations. Also, excessive reliance on the initial template results in an inability to recover from tracking failures when catastrophic drift happens.

To address this issue, we propose to learn the object template update itself in Chapter 5 of this thesis. Our learned update strategy utilizes object and background

information, and is thus adaptive to the present circumstances of each particular situation. In our approach, the predicted template is calculated as an update function of the initial ground-truth template, the accumulated template from all previous frames, and the template at the predicted object location in the current frame. Hence, the new accumulated template contains an effective historical summary of the object appearances across all the frames in a video. More specifically, the update function is implemented with a convolutional neural network, which we call *UpdateNet*. This is a compact network that can be applied to any Siamese tracker to enhance its online update capabilities while maintaining its efficiency properties. Furthermore, the complex structure makes it able to effectively learn the nuances of template update and be adaptive to various tracking situations. Extensive experiments on four benchmark datasets (VOT2016, VOT2018, LaSOT, and TrackingNet) demonstrate that the proposed approach significantly improves the performance of the trackers with respect to the linear update or no-update. As a result, UpdateNet achieves competitive results in these datasets.

2 Visual Tracking

The aim of this chapter is to present the general context and overall background of the topics which are covered in the next chapters. We will first provide an overview of trackers and then discuss the existing datasets.

2.1 Visual Trackers

Visual object tracking is a fundamental subject in computer vision whose applications range from autonomous driving to robotics and video analysis. The main idea is to estimate the trajectory of an object in a sequence of images, given that the tracker is initialized at the object region in the first frame. Then the model of the object appearance is learned online during tracking. Such knowledge can be used in multiple scenarios. For example, first, tracking pedestrians in a surveillance context to perform video analysis and conflict detection. Second, tracking objects of interest, e.g. cars and cyclist, to estimate their trajectories and take a navigational decision accordingly. Both of them indicate that tracking is an important aspect of visual perception in robotics and autonomous systems, where the extraction of high-level information from the camera sensor can be helpful to assist the human in tracking decisions.

There are two prevalent paradigms for visual tracking in recent years. One is the optimization based trackers, also called Discriminative Correlation Filter (DCF) trackers (Bolme et al., 2010; Danelljan et al., 2014a, 2015a,b, 2014b; Galoogahi et al., 2013; Henriques et al., 2012). The other paradigm is the Siamese network based trackers (Bertinetto et al., 2016b; Li et al., 2018b; Valmadre et al., 2017; Zhu et al., 2018).

Since Bolme et al. (2010) proposed the Minimum Output Sum of Squared Error (MOSSE) model, the formulation framework of the DCF was defined and it also brings about a majestic development for DCF in visual tracking. Later on, Galoogahi et al. (2013) extended it to multi-channel DCF where the HOG and SIFT features can be used. Meanwhile, Henriques et al. (2012) extended it by adding a kernel technique, which defines the inner product between two samples in some

high-dimensional feature space. Moreover, Danelljan et al. (2014b) integrated the multi-dimensional feature maps, e.g. color feature (Van De Weijer et al., 2009), into DCF, which is computationally efficient while maintaining high discriminative power. Additionally, Danelljan et al. (2014a) proposed to learn a scale correlation filter, parallel with the translation correlation filter, for scale estimation. To further improve the correlation filter, Danelljan et al. (2015b) introduced a spatial regularization component in the DCF formulation, and improved the performance by a large margin. Kiani Galoogahi et al. (2017) proposed to use the mask to crop the correlation filter. This can extract much more realistic negative samples for training correlation filter, thus effectively alleviating the issue of boundary effect for DCF trackers. Lastly, convolutional features are also used to learn the correlation filter by Danelljan et al. (2015a).

The other paradigm is based on Siamese networks. Bertinetto et al. (2016b) proposed to learn a similarity metric, i.e. Siamese network, offline for conducting the tracking procedure by template matching. Then, Bertinetto et al. (2016b) extended it by adding a correlation filter component on the exemplar branch for final cross-correlation with the feature map of the search region. However this results in limited improvement. Later on, Li et al. (2018b) introduced the region proposal network (RPN) from object detection (Ren et al., 2015) to Siamese tracking. This uses a classification branch and a regression branch for the tracking prediction. This training framework, called SiamRPN, significantly improved the performance of SiamFC. Based on SiamRPN, Zhu et al. (2018) used large amounts of diverse categories of positive pairs and semantic negative pairs, along with data augmentation and an incremental learning technique to learn distractor-aware Siamese networks for accurate tracking.

2.1.1 Optimization Based Trackers

The DCF based trackers have undergone prosperous development in recent years. In this section, we introduce the typical correlation filter. Then we describe Efficient Convolution Operators (ECO) (Danelljan et al., 2017) which is an advanced DCF based tracker and achieved high-performance in several datasets (Kristan et al., 2016a; Wu et al., 2013, 2015). We use it as the correlation filter for our experiments in Chapter 3.

DCF (Henriques et al., 2015): To discriminate the target appearance from the background, the discriminative correlation filter learns a linear correlation filter f . Then the filter f is used to predict the target location. The desired filter f is

calculated by minimizing the following least squares objective:

$$E(f) = \left\| \sum_{d=1}^D f^d \star x^d - y \right\|^2 + \lambda \sum_{d=1}^D \|f^d\|^2. \quad (2.1)$$

Here \star denotes the circular correlation. x^d denotes feature maps of training samples x , where d indexes the channel, $d \in \{1, \dots, D\}$. y is the desired regression target which is a Gaussian-shaped function. λ is a regularization weight to control overfitting. By periodically shifting the sample x to a circulant matrix X , $X = C(x)$ where C is a cyclic shift operator, the correlation filter f can be calculated with a closed-form solution at all cyclic shifts.

There are many beneficial properties for the circulant matrix. One of them is that the circulant matrix can be made diagonal by the Discrete Fourier Transform (DFT). Thus, the closed-form solution of f can be transferred into Fourier domain, where the process is efficient and versatile. In addition, the storage and computation are reduced by several orders of magnitude. As a consequence of the above advantages, the DCF framework achieves excellent performance and is therefore broadly applied in the tracking community.

The filter is calculated with:

$$\hat{f}^d = \frac{\hat{x}^d \hat{y}^*}{\sum_{d=1}^D \hat{x}^d (\hat{x}^d)^* + \lambda}, \quad (2.2)$$

where \hat{x}^d is the Discrete Fourier Transform (DFT) of input sample x^d , \hat{y}^* denotes the complex conjugate of the DFT $\mathcal{F}(y)$, and f^d denotes the DFT of the generating vector, $\hat{f}^d = \mathcal{F}(f^d)$. From now on, we will always use a hat $\hat{\cdot}$ as shorthand for the DFT of a vector.

Due to the circulant property, the correlation filter generates a large amount of shifted patches of the foreground target as the negative training samples. While these shifted synthetic samples are not truly representative of negative patches in real-world scenes. Thus, the DCF tracker is generally affected by circular boundary effects.

ECO (Danelljan et al., 2017): The tracker uses a pre-trained model to extract the deep features for the follow-up components of the tracker. The model is trained on an image classification dataset. It learns a projection matrix P to reduce dimensionality of the deep features. This technique effectively alleviated the problem of computational complexity and overfitting for the trackers with multi-feature fusion.

The model filter f is learned based on a set of M training samples $\{x_j\}_1^M$ and

corresponding target maps $\{y_j\}_1^M$. The label y_j is the desired target score, which is defined as a periodically repeated Gaussian function centered at the sample location (Danelljan et al., 2016b; Henriques et al., 2015). Each training sample consists of multiple feature layers $x_j^d \in \mathbb{R}^{N_d \times N_d}$, where N_d is the spatial resolution of layer $d \in \{1, \dots, D\}$. These feature layers contain both shallow and deep features of varying resolutions, specifically for the first and fifth convolutional layers in the VGG-M network (Chatfield et al., 2014). The tracker predicts the object location using the target score operator, defined as below:

$$S_{f,P}\{x\} = \sum_{d=1}^D f^d * PJ_d\{x^d\}. \quad (2.3)$$

Here, the detection score function of the target is predicted by $S_{f,P}\{x\}$. The feature-wise operator J_d is used to interpolate the sample x to a continuous domain. It performs in Fourier domain by using a cubic spline kernel, see more details in (Danelljan et al., 2016b). The projection matrix P can effectively reduce the dimensionality of the feature space.

To learn the detection score operator, we use the following least squares loss,

$$E(f) = \sum_{j=1}^M \alpha_j \|S_{f,P}\{x_j\} - y_j\|^2 + \sum_{d=1}^D \|w f^d\|^2 + \lambda \|P\|_F^2. \quad (2.4)$$

Here, the filter f are regularized by w . The spatial regularization weight function w is employed to mitigate the effects of periodic repetition (Danelljan et al., 2015b). Each sample x_j is weighted by α_j . The label functions y_j are set to Gaussian functions centered at the target location.

An equivalent loss is obtained by using Parseval's formula as follows:

$$E(f) = \sum_{j=1}^M \alpha_k \|\widehat{S_{f,P}\{x_j\}} - \hat{y}_j\|^2 + \sum_{d=1}^D \|\hat{w} * \hat{f}^d\|^2 + \lambda \|P\|_F^2. \quad (2.5)$$

Here $\hat{\cdot}$ denotes the Fourier coefficients. The projection matrix P and the filter f are jointly trained by the Gauss-Newton method in the first frame. In subsequent frames, the resulting normal equations are efficiently solved using the method of Conjugate Gradients (Nocedal & Wright, 2006), assuming a fixed P . For more details, we refer to (Danelljan et al., 2017, 2016b).

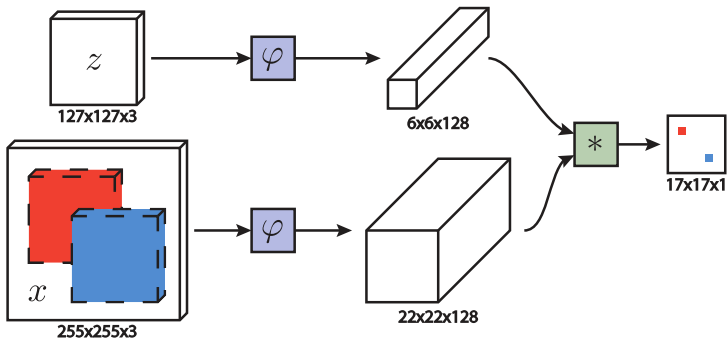


Figure 2.1 – **Fully-convolutional Siamese architecture.** Figure from (Bertinetto et al., 2016b).

2.1.2 Siamese Network Based Trackers

Siamese approaches learn a feature embedding network and a similarity metric offline and during inference, the response map is obtained by the cross-correlation of template and search region. Cross-correlation is a measure of similarity of two templates using dot product. In this section, firstly we introduce the original Siamese tracker, i.e. SiamFC (Bertinetto et al., 2016b). Then we introduce its high-quality variant, i.e. DaSiamRPN (Zhu et al., 2018). We use both of them as our baseline trackers equipped with UpdateNet in Chapter 5.

SiamFC (Bertinetto et al., 2016b): The tracker uses a Siamese network (two-stream architecture) for finding the optimal matched template with an end-to-end training manner, as depicted in Figure 2.1. One stream extracts the object template’s features based on an *exemplar* image that contains the object to be tracked. The other stream receives as input a large *search region* in the target image. The two outputs are cross-correlated to generate a response map of the search region. Many trackers (He et al., 2018; Li et al., 2018b; Valmadre et al., 2017; Wang et al., 2018; Zhang et al., 2018; Zhu et al., 2018) have extended the SiamFC architecture for tracking. The Siamese-based trackers have gained popularity since they provide a good trade-off between computational speed and tracking performance.

They use a deep convolutional neural network, formulated as φ , to extract the feature representations. Then the template of the object and the feature of search region are represented as $T = \varphi(z)$ and $S = \varphi(x)$, respectively. Finally they use a cross-correlation layer to compute the response map as: $f(z, x) = T * S + b\mathbf{1}$, where $b\mathbf{1}$ denotes a signal which uses value b in every position. The Siamese network is

trained on positive and negative pairs, with a logistic loss as below:

$$l(y, v) = \log(1 + \exp(-yv)) \quad (2.6)$$

where v is a real-valued response map which is obtained by a cross-correlation layer $v = f(z, x)$. $y \in \{+1, -1\}$ is the ground-truth label. The pairs are obtained from a dataset of annotated videos by extracting exemplar and search images that are centred on the target. The images are extracted from two frames of a video that both contain the object and are at most T frames apart. The fully-convolutional network is trained with the loss of a score map (Eq. (2.6)) by using the pairs that consist of an exemplar image and a larger search image. During the inference stage, the Siamese tracker performs efficiently as no online learning. While these can easily result in the fact that they often struggle at the problem of target classification and do not explicitly account for distractors, on the other hand.

SiamRPN (Li et al., 2018b): This tracker takes inspiration of the region proposal network (RPN) from object detection (Ren et al., 2015), by using correlation feature map of the two branches for proposal extraction. The RPN happens after the Siamese network, where there exist two branches, one is for classification and the other for regression, as depicted in Figure 2.2. The RPN uses the channels of features maps to represent the foreground, background and four coordinates of the five anchors. The outputs of the two branches are obtained by cross-correlation separately. The template branch of the Siamese network is used to represent the target appearance, which is encoded into feature map of RPN to discriminate foreground from background.

Assuming that $\varphi(z)$ and $\varphi(x)$ are the output feature representations of the Siamese network, where z is the exemplar template, and x is the search region. The correlation is computed on both the classification branch and the regression branch:

$$A_{w \times h \times 2k}^{cls} = [\varphi(x)]_{cls} \star [\varphi(z)]_{cls} \quad (2.7)$$

$$A_{w \times h \times 4k}^{reg} = [\varphi(x)]_{reg} \star [\varphi(z)]_{reg} \quad (2.8)$$

The template feature maps $[\varphi(z)]_{cls}$ and $[\varphi(z)]_{reg}$ are used as kernels and \star denotes the cross-correlation operation.

When training the network with several anchors, they employ the loss function used in Faster R-CNN (Ren et al., 2015). The classification loss L_{cls} is cross-entropy loss over two classes (foreground *vs.* background). The loss is defined as: $L_{cls} = -(y \log(p) + (1 - y) \log(q))$, where p and q denote the predictions for the foreground and background, $y \in \{+1, -1\}$ is the ground-truth label.

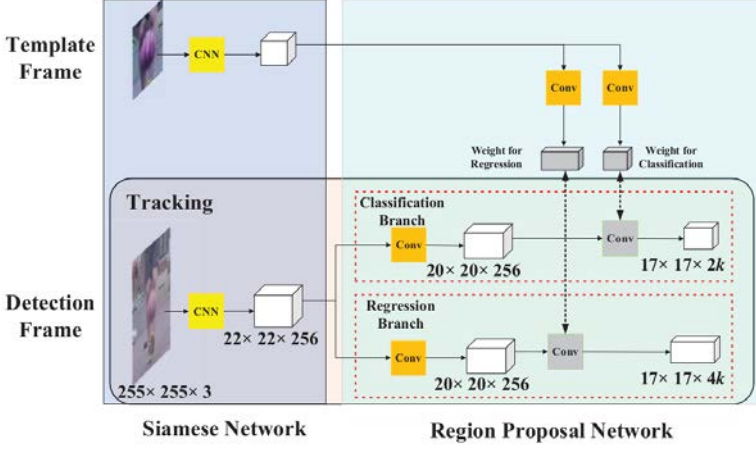


Figure 2.2 – **Main framework of Siamese-RPN.** It includes feature extraction network and region proposal network. Figure from (Li et al., 2018b).

They adopt smooth L_1 loss with normalized coordinates for regression. Let A_x, A_y, A_w, A_h denote the center point and shape of the anchor boxes and T_x, T_y, T_w, T_h denote those of ground truth boxes. The normalized distance is:

$$\delta_0 = \frac{T_x - A_x}{A_w}, \delta_1 = \frac{T_y - A_y}{A_h}, \delta_2 = \ln \frac{T_w}{A_w}, \delta_3 = \ln \frac{T_h}{A_h} \quad (2.9)$$

Then they pass through smooth L_1 loss which is formulated as:

$$\text{smooth}_{L_1}(x, \sigma) = \begin{cases} 0.5\sigma^2 x^2, & |x| < \frac{1}{\sigma^2} \\ |x| - \frac{1}{2\sigma^2}, & |x| \geq \frac{1}{\sigma^2} \end{cases} \quad (2.10)$$

Then the regression loss is calculated as below:

$$L_{reg} = \sum_{i=0}^3 \text{smooth}_{L_1}(\delta_i, \sigma) \quad (2.11)$$

The final loss function is calculated by combining the classification loss and regression loss:

$$\text{loss} = L_{cls} + \lambda L_{reg}, \quad (2.12)$$

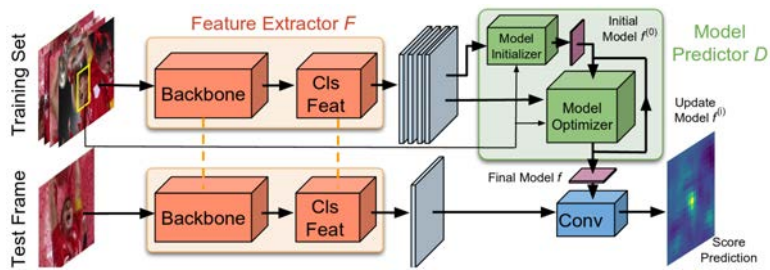


Figure 2.3 – **The tracking architecture of DiMP.** It includes feature extraction and model predictor. Figure from (Bhat et al., 2019).

where λ is the trade-off value for balancing classification task and regression task.

DaSiamRPN (Zhu et al., 2018) observes that the features of Siamese trackers are less discriminative from the non-semantic backgrounds which are usually considered as distractors. They investigate that this is mainly due to the imbalance of the non-semantic background and semantic distractors in the training data. To tackle this issue, they propose an effective sampling strategy to control the distribution and make the model focus on the semantic distractors. During inference of on-line tracking, a novel distractor-aware module is designed to perform incremental learning, which can explicitly suppress distractors.

2.1.3 Optimization and Siamese Network Based Trackers

In this section, we introduce a tracker consisting of both the Optimization part and Siamese network part, called *DiMP* (Bhat et al., 2019) as the Figure 2.3. We use this tracker in Chapter 4.

DiMP (Bhat et al., 2019): To enable the tracker to be trained completely end-to-end, they embed the online learning of the target model into the tracker itself by learning a powerful discriminative filter. DiMP consists of the three components: feature extractor, model predictor, and IoU-Net (for target estimation as also used in Jiang et al. (2018)). With this carefully designed end-to-end network and an effective optimization method, they achieve superior performance by setting a new state-of-the-art on several recent RGB tracking datasets (Fan et al., 2018; Huang et al., 2018; Kristan et al., 2018; Muller et al., 2018; Wu et al., 2015).

Optimization part. The model predictor is an optimization component in the DiMP tracker. It is followed by the feature extractor and predicts the final optimized

filter f . The model predictor consists of the model initializer and model optimizer. The model initializer is to provide a reasonable initial estimate of the filter, i.e. $f^{(0)}$. The model optimizer is to solve the final model f with the steepest descent (SD). The model predictor uses a set of S_{train} to train the discriminative model $f = D(S_{train})$. The classification loss is calculated as:

$$L_{cls} = \frac{1}{N_{iter}} \sum_{i=0}^{N_{iter}} \sum_{(x,c) \in S_{test}} \left\| l(x * f^{(i)}, z_c) \right\|^2, \quad (2.13)$$

where z_c is a Gaussian shape function centered on the target c . l is a hinge loss. The final response map is obtained by $s = x * f, x \in S_{test}$.

Siamese network part. In DiMP, both of the feature extractor and IoU-Net use Siamese architecture, as two streams of images are fed to the reference branch and test branch separately. The backbone feature extractor F normally employs the ResNet-18 or ResNet-50 (He et al., 2016), pre-trained on ImageNet (Russakovsky et al., 2015), to extract the deep feature representations for the follow-up implementation models. Specifically, the deep features are extracted for the model predictor and IoU-Net. The feature extractor F is shared and is fine-tuned during the end-to-end training. When fine-tuning, the input data for F is a pair of sets (M_{train}, M_{test}) . Each set $M = \{(I_j, b_j)\}_{j=1}^{N_{frames}}$ contains images I_j along with their object bounding box b_j . The images are passed through the feature extractor F , and the train set $S_{train} = \{(x_j, c_j)\}$ is obtained by $x_j = F(I_j)$, where c_j is the center coordinate of b_j .

Next, IoU-Net model is used to predict the IoU between the deep feature x and a bounding box candidate B as in the ATOM tracker (Danelljan et al., 2019). ATOM aims to use the IoU network to predict the target estimation instead of using multi-scale search. IoU-Net is also a Siamese network and has two branches, i.e. *IoU modulation* and *IoU predictor*. The first one calculates the modulation vector from the reference image, and the second branch predicts the IoU values from the test image. Both of the branches are added with convolutional layers. Then they are followed by Precise ROI Pooling (PrPool) and fully connected layer g . The reference branch outputs a precomputed vector c to modulate the feature of the test image. The IoU is calculated in terms of x and B as follows,

$$IoU(B) = g(c(x_0, B_0) \cdot z(x, B)), \quad (2.14)$$

where, x_0, B_0 and x, B are from the reference and test images separately, and z is the feature after PrPool layer in test branch.

Modality	Dataset	Sequences	Frames
RGB	OTB100 (Wu et al., 2015)	98	58,610
	VOT2018 (Kristan et al., 2018)	60	21,356
	LaSOT (Fan et al., 2018)	1,400	3.52M
	TrackingNet (Muller et al., 2018)	30,643	14,431,266
	GOT-10k (Huang et al., 2018)	10,000	1.5M
TIR	VOT-TIR2015 (Felsberg et al., 2015)	20	11.3K
	VOT-TIR2017 (Kristan et al., 2017b)	25	18.5K
RGB-T	VOT-RGBT2019 (VOT challenge, 2019)	60	20,083
	RGBT210 (Li et al., 2017b)	210	210K

Table 2.1 – Comparison of several datasets in three modalities for visual tracking.

2.2 Datasets

In this section, we introduce the datasets for the different modalities for visual tracking. They are divided into RGB datasets, TIR datasets and RGB-T datasets. There are several standard tracking datasets for each modality as described in Table 2.1, where we categorize several datasets into their respective modalities and show their sequences numbers, frame numbers, and frame rate. For example, in the RGB modality, there are OTB (Wu et al., 2013, 2015), VOT (Kristan et al., 2016a), LaSOT (Fan et al., 2018), TrackingNet (Muller et al., 2018) and GOT-10k (Huang et al., 2018). Among them, OTB and VOT are the typical tracking datasets, which appeared at the early stage as a unified evaluation benchmark for tracking. And also there are the recent proposed large-scale training datasets, e.g. LaSOT, TrackingNet and GOT-10k, which can provide abundant data for training deep neural networks for tracking. In the TIR modality, there are mainly two datasets, i.e. VOT-TIR2015 (Felsberg et al., 2015) and VOT-TIR2017 (Kristan et al., 2017b). In the RGB-T modality, the two popular datasets, i.e. VOT-RGBT2019 (VOT challenge, 2019) and RGBT210 (Li et al., 2017b), are used for evaluating multi-modal tracking.

2.2.1 RGB Datasets

In this section, we introduce several popular RGB datasets. We present six exemplar frames from these RGB datasets as shown in Figure 2.4.

OTB dataset (Wu et al., 2013, 2015): This dataset consists of the OTB2013 (Wu et al., 2013) and OTB100 datasets (Wu et al., 2015). OTB2013 has 51 sequences containing more than 29,000 frames in total. OTB100 has 98 sequences containing



Figure 2.4 – Selected frames from the datasets of OTB2013, OTB100, VOT, LaSOT, TrackingNet, and GOT-10k respectively.

more than 58,610 frames in total. Their main protocol is *One Pass Evaluation (OPE)*, in which tracking continues after failure and thus only the ground-truth from the absolute initial frame is used. The evaluation metrics are the *precision plot* and the *success plot*. Precision is measured by the center location error, calculated by the average Euclidean distance between the center location predicted by the tracker and the ground-truth location. The precision plot shows the percentage of frames whose estimated location is within the given precision threshold for different threshold values, and the representative score is the precision taking 20 pixels as threshold. On the other hand, success is measured through the intersection over union (IoU) between the predicted bounding box and the ground-truth. The success plot shows the ratio of bounding boxes whose IoU is higher than a given threshold. Trackers are then ranked in terms of Area Under the Curve (AUC) of the success plots.

VOT dataset (Kristan et al., 2016a): At present, seven editions of the challenge have been opened for visual object tracking challenge (VOT) (Kristan et al., 2016a), i.e. from VOT2013 to VOT2019. In VOT2015, there are 60 RGB sequences for the main challenge. In VOT2018, the long-term challenge is added requiring the trackers to determine when the target has disappeared and re-detect the target after losing the target. The main VOT protocol, the supervised experiment, establishes that when the evaluated tracker fails, i.e. when the overlap with the ground-truth is below a given threshold, it is re-initialized in the correct location five frames after the failure. Since VOT2015, the main evaluation measure used to rank the trackers is Expected Average Overlap (EAO), which is a combination of accuracy (A) and robustness (R).

While the unsupervised experiment conducts the OPE similar as the OTB protocol.

LaSOT dataset (Fan et al., 2018): To benefit object tracking using deep learning, LaSOT is proposed to present large-scale high-quality videos including a training and testing subset. Thus deep trackers trained on the training subset can be evaluated on the testing subset. LaSOT consists of two protocols, the protocol I evaluates the trackers across whole videos including the training and testing subsets. The protocol II, only evaluates on the testing subset. LaSOT is a much larger and more challenging dataset which includes long-term sequences. The total LaSOT dataset contains 1400 sequences with 3.52M frames. The testing subset, split from the whole LaSOT dataset, has 280 sequences with 690K frames. The LaSOT dataset (Fan et al., 2018) follows the OPE criterion of OTB (Wu et al., 2013). Besides precision plot and success plot, LaSOT also uses *normalized precision plot* to counter the situation that target size and image resolution have large discrepancies for different frames and videos, which heavily influences the precision metric.

TrackingNet dataset (Muller et al., 2018): This is a large-scale tracking dataset consisting of videos in the wild. TrackingNet is built to address the issue of data-hungry algorithms for deep tracking. It contains a large variety of frame rates, resolutions and object classes. It has a total of 30,643 videos split into 30,132 training videos and 511 testing videos, with an average of 471 frames. The testing videos have a similar distribution as the training set which is carefully selected from Youtube-BoundingBoxes dataset (Real et al., 2017). TrackingNet uses precision, normalized precision and success as the evaluation metrics.

GOT-10k dataset (Huang et al., 2018): It is an abbreviation of Generic Object Tracking Benchmark. GOT-10k dataset has over 10,000 video segments, covering 563 classes of real-world moving objects and more than 80 motion patterns, amounting to a total of over 1.5 million manually labeled bounding boxes. It also provides additional supervision in terms of attribute labels such as ratio of object visible or type of motion. The GOT-10k is split into unified training, validation and testing subsets. The training set contains 9,335 videos (1,403,359 frames), with 480 object classes and 69 motion classes. The validation set consists of 180 videos segments, with 150 object classes and 15 motion classes. The testing set consists of 180 videos segments, with 84 object classes and 32 motion classes, allowing trackers to carry out effective evaluation. GOT-10k chooses the widely used average overlap (AO) and success rate (SR) as the evaluation metrics.



Figure 2.5 – Selected frames from the datasets of VOT-TIR2015 and VOT-TIR2017.

2.2.2 TIR Datasets

From 2015, the VOT challenge introduced the VOT-TIR dataset as a sub-challenge. We present three exemplar frames from the TIR datasets as shown in Figure 2.5.

VOT-TIR2015 dataset (Felsberg et al., 2015): This is the first standard TIR tracking benchmark which provides the dataset and toolkit to fairly evaluate TIR trackers. The dataset contains 20 TIR image sequences, with an average sequence length of 563 frames. It consists of six kinds of challenges such as dynamics change, occlusion, camera motion, motion change, size change, and empty. The VOT-TIR2015 dataset follows the standard VOT protocol (including the EAO evaluation) for testing.

VOT-TIR2017 dataset (Kristan et al., 2017b): It consists of 25 TIR sequences, with an average sequence length of 740 frames. Their resolutions range from 305×225 to 1920×480 pixels. VOT-TIR2017 is more challenging than VOT-TIR2015. There are 11 global attributes (per-sequence), including: blur, dynamics change, temperature change, object motion, size change, camera motion, background clutter, aspect ratio change, object deformation, scene complexity, neutral. There are 6 local attributes (per-frame), including: occlusion, dynamics change, motion change, size change, camera motion, neutral. They follow the evaluation protocol as in the VOT-TIR 2015.

2.2.3 RGB-TIR Datasets

Integrating RGB and TIR (called RGB-T in the thesis) spectrum data has been proven to be effective in boosting the performance of visual trackers. The integration of RGB and TIR also allows tracking target objects in both the conditions of daytime and night time. RGB and TIR information complement each other and contribute to visual tracking in different aspects. Recently, two popular RGB-T datasets, i.e.



Figure 2.6 – Selected frames from the datasets of VOT-RGBT 2019 and RGBT210.

VOT-RGBT 2019 (VOT challenge, 2019) and RGBT210 (Li et al., 2017b), are used as the benchmarks for the evaluation of RGB-T trackers. We present three pairs of exemplar frames from the RGB-TIR datasets in Figure 2.6.

VOT-RGBT 2019 dataset (VOT challenge, 2019): In VOT2019, two new multi-channel datasets, i.e. VOT-RGBD and VOT-RGBT, are introduced for multi-modal tracking. This is because of that the multi-modal fusion information can definitely help visual tracking in the real world. There are 60 public pairs of RGB and TIR sequences for VOT-RGBT 2019 dataset. The total frame pairs reach about 20.1K. There are five attributes labeled for each sequence of the 60 pairs for both modalities. The list of attributes is: camera motion, dynamics change, occlusion, size change, and motion change. They adopt four metrics, i.e. Accuracy (A), Robustness (R), Expected Average Overlap (EAO), and Frames Per Second (FPS) for evaluating the performance of trackers. Among them, the main metric for the final ranking of trackers is normally EAO.

RGBT210 dataset (Li et al., 2017b): It consists of 210 RGB-T videos, each of them containing a pair of RGB video and TIR video. It totally contains about 210K frames and the maximum number of frames of each video pairs reaches 8K. Moreover, they annotate each frame with an exact bounding box of the target for both modalities. The videos in this dataset also cover various environmental challenges, such as raining, night, and cold days. To analyze the attribute-based performance of different

trackers, RGBT210 dataset annotates 12 attributes, including No Occlusion (NO), Partial Occlusion (PO), Heavy Occlusion (HO), Low Illumination (LI), Low Resolution (LR), Thermal Crossover (TC), Deformation (DEF), Fast Motion (FM), Scale Variation (SV), Motion Blur (MB), Camera Moving (CM), and Background Clutter (BC). In addition, they set five widely used metrics for evaluating the performance of RGB-T trackers. These metrics are Precision Rate (PR), Success Rate (SR), Accuracy, Robustness, and Expected Average Overlap (EAO).

3 Synthetic Data Generation for End-to-End Thermal Infrared Tracking *

3.1 Introduction

Visual tracking aims to estimate a trajectory of an object through a video based on only one bounding box annotation at the beginning of the sequence. Tracking is important for applications in surveillance (Emami et al., 2012), video understanding (Renoust et al., 2016) and robotics (Liu et al., 2012). One of the main challenges of tracking is the limited data problem: the tracker should be able to track an object based on only a single annotated bounding box. The success of a tracker is therefore very dependent on the quality of the discriminative features which are used by the tracker.

Recently, specialized tracking subproblems have emerged. Among these is the field of tracking in thermal infrared (TIR) images, whose importance is further increasing due to improvements of thermal infrared sensors in resolution and quality (Felsberg et al., 2015, 2016; Kristan et al., 2017a). The advantage of thermal images is that they are not influenced by the illumination variations and shadows, and objects can be distinguished from the background as the background is normally colder. In addition, thermal infrared tracking can be used in total darkness, where visual cameras have no signal. Considering these advantages, thermal infrared tracking has a wide range of applications in car and pedestrian surveillance systems as well as various defense systems (Gade & Moeslund, 2014).

In recent years, Discriminative Correlation Filter (DCF) based methods (Bolme et al., 2010; Danelljan et al., 2017; Henriques et al., 2015) have shown to provide excellent tracking performance on existing benchmarks (Kristan et al., 2016a; Mueller et al., 2016; Wu et al., 2015). The DCF based trackers learn a correlation filter from example patches to discriminate between the target and background appearance. Further, the DCF based framework efficiently utilizes all spatial shifts of the training samples by exploiting the properties of circular correlation to train and apply a discriminative classifier in a sliding window fashion. Lately, the DCF based framework has been significantly advanced by employing high-dimensional

*This chapter is based on a publication in IEEE Transactions on Image Processing 2018

visual features (Danelljan et al., 2016a, 2014b; Henriques et al., 2015), powerful learning methods (Danelljan et al., 2017; Song et al., 2017b), reducing boundary effects (Danelljan et al., 2015b), and accurate scale estimation (Danelljan et al., 2014a). Due to their superior performance in RGB tracking, some of these methods have also been applied with success to TIR (Danelljan et al., 2017, 2015b).

Recently, deep learning has revolutionized the field of computer vision significantly advancing the state-of-the-art in many applications (Krizhevsky et al., 2012). Generally the deep networks are trained on a large amount of labeled training data. Despite its astounding success, the impact of deep learning on generic visual tracking (RGB data) has been limited. One of the key issues when employing deep features for tracking is the unavailability of large-scale labeled tracking data for training. Further, the tracking model is desired to be learned using a single labeled frame. Therefore, most existing deep learning based DCF trackers (Danelljan et al., 2017, 2016b; Ma et al., 2015a) employ deep features pre-trained on the ImageNet dataset (Russakovsky et al., 2015) for image classification task. Other approaches (Song et al., 2017b; Valmadre et al., 2017) have investigated the integration of DCF in a deep network by adapting the end-to-end philosophy, but did not result in major improvements over features from pre-trained networks.

Even more than for RGB tracking, introducing deep learning to TIR tracking is hampered by the absence of large datasets. The datasets which are available for thermal infrared videos are relatively small. Moreover, there is no ImageNet counterpart of infrared still images on which a large network could be pre-trained. Therefore, the usage of hand-crafted features remains dominant for TIR tracking. For instance, the top three trackers in VOT-TIR2017 (Kristan et al., 2017a, 2016a) are still exploiting hand-crafted features. The winner (Yu et al., 2017) of VOT-TIR2017 challenge employs HOG (Dalal & Triggs, 2005) and motion features. Further, the other top-performing methods (Felsberg et al., 2015; Zhu et al., 2016) are based on hand-crafted features. The success of these methods show that hand-crafted features are still the best choice for TIR tracking.

Deep learning has also resulted in fast progress in generative models which are able to generate samples from complex image distributions (Goodfellow et al., 2014). These models have been further extended to image-to-image translation models (Isola et al., 2017) which allow to learn mappings between image domains. A further extension of this work allows to learn mappings between unpaired domains (Zhu et al., 2017), which is based on the observation that transferring an image to another domain and then transferring it back to the first domain should result in the same image which was provided as input. One of the more interesting applications of these generative networks is that they can be used to construct synthetic datasets of small data domains, such as TIR. In this work, we show that labeled data from RGB can be translated to TIR data as well as the labels.

In this work we tackle the key limited-data problem for TIR-tracking by utilizing recent developments in image-to-image translation methods (Isola et al., 2017; Zhu et al., 2017). The idea is to automatically transfer RGB tracking videos to the TIR domain. We can then automatically transfer the labels from these RGB videos to the synthetic TIR videos. The resulting data can then be used to extract discriminative deep features for the TIR domain. The advantage is that we can generate the TIR counterpart of the available RGB tracking datasets which are much larger compared to the current TIR tracking datasets. The main contributions of this chapter are:

- We address the scarcity of labeled data for TIR tracking. Therefore, we propose a framework which transfers RGB data to synthetic TIR data. The labels available for the RGB data are also transferred to the TIR data, resulting in a large synthetic TIR data set for tracking.
- We are the first to perform end-to-end training for TIR tracking, showing that this can significantly improve results (see Table 3.3). We also show that a tracker trained on only synthetic data can outperform trackers trained on available labeled TIR data (see Fig. 3.1).
- We perform extensive evaluations on the latest TIR tracking challenge (Kristan et al., 2017a) verifying the efficiency of our different models trained on synthetic TIR datasets. We show that when combined with motion features our method obtains state of the art on the TIR tracking challenge.

The rest of the Chapter is organized as follows. In section 3.2 we briefly discuss related work. In section 3.3 we introduce the standard correlation filter and the current end-to-end deep correlation filter. In section 3.4 and section 3.5 we describe the prevalent generative adversarial networks and present our generated synthetic tracking videos. In section 3.6 we present our experiments on standard thermal infrared tracking dataset. In section 3.7 we conclude our work and plan our further research.

3.2 Related Work

3.2.1 DCF Tracking

In recent years, discriminative correlation filter (DCF) based tracking methods have shown excellent performance in terms of accuracy and robustness on benchmark tracking datasets (Kristan et al., 2016a; Wu et al., 2015). The DCF based trackers aim at learning a correlation filter in an online fashion from example image patches to discriminate between the target and background appearance. The seminal work

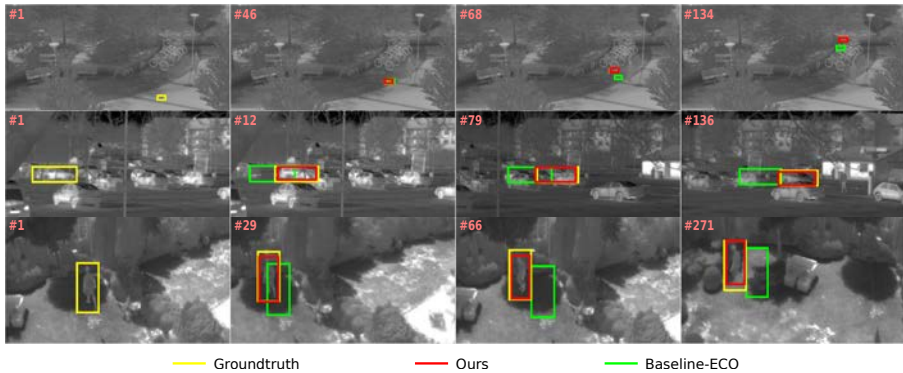


Figure 3.1 – **Qualitative comparison of our approach trained on generated data only (red) with baseline ECO (Danelljan et al., 2017) (green) on the *quadrocopter2*, *car2* and *garden* videos.** The ground truth bounding box is provided in yellow. Owing to the synthetic TIR data our model is able to follow the object successfully in case of out-of-plane rotation, partial occlusion and scale changes.

of Bolme et al. (2010) was restricted to a single feature channel (grayscale image). Later, the DCF framework was extended to use multi-dimensional hand-crafted features (Danelljan et al., 2014b; Galoogahi et al., 2013; Henriques et al., 2015), such as HOG (Dalal & Triggs, 2005) and Color Names (Van De Weijer et al., 2009). Some of the recent advances in DCF frameworks can be attributed to reducing boundary effects (Danelljan et al., 2015b), robust scale estimation (Danelljan et al., 2014a), integrating context (Mueller et al., 2017a), and adding a long-term memory component (Ma et al., 2015b).

Even after more than five years of flourishing, discriminative correlation filter based tracking is still the mainstream in single object tracking. Recent modifications on DCF include: Mueller et al. (2017a) sample four context patches around the target and incorporate these to regularize the regression function, which has the same effect as hard negative mining. Lukezic et al. (2017) enlarge the search region and improve tracking of non-rectangular objects by using spatial maps to restrict the correlation filter. They also give the learned filter adaptive channel-wise weights, which improves the quality of the filter. Kiani Galoogahi et al. (2017) use a mask to crop the object in the spatial domain and get a new closed-form solution of the correlation filter in the Fourier domain by embedding the mask matrix into the formulation. This yields significantly more shifted examples unaffected by boundary effects.

Compared to hand-crafted features (e.g. HOG (Dalal & Triggs, 2005), intensity

and Color Names (Van De Weijer et al., 2009)), deep CNN features significantly improve the robustness of the tracker against geometric variations (Danelljan et al., 2015a), resulting in a significant improvement of the performance. This is mainly caused by the high discriminative power of deep features, since CNNs are trained on large datasets such as ILSVRC2012 (Russakovsky et al., 2015). Later, Ma et al. (2015a) propose to encode the target appearance on several convolutional layers and each layer has a corresponding correlation filter. This hierarchical architecture locates targets by maximizing the response of each layer with different weights. They find an optimized position in a coarse to fine way. Directly using different layers may not take full advantage of the CNN features because of the discrete distribution of features. To exploit the continuity between different layers of networks, Danelljan et al. (2016b) propose to learn a convolution operator in the continuous spatial domain called CCOT. As CCOT is very slow, Danelljan et al. (2017) propose to factorize the convolution operator to reduce the dimensions of feature maps. Then they use GMM to generate samples, which significantly accelerate the tracker, enabling the tracker to run in real-time, while still maintaining the same or higher accuracy.

3.2.2 TIR Tracking

Currently, the leading TIR trackers still employ hand-crafted features in their models. Yu & Yu (2018) propose structural learning on dense samples around the object. Their tracker uses edge and HOG features and transfers them into the Fourier domain, to obtain a real-time tracker. Later they extend this work, called DSLT (Yu et al., 2017), by integrating HOG (Dalal & Triggs, 2005) and motion features. With this tracker they won the VOT-TIR2017 challenge (Kristan et al., 2017a). Another TIR tracker, called EBT (Zhu et al., 2016), uses edge features to devise an objectness measure specific for each instance. This enables the generation of high quality object proposals and the use of richer features. Concretely, for each proposal they extract a 2640-dimensional histogram feature as well as a 5-level pyramid computed from the intensity channel. They achieve the runner-up position in the VOT-TIR2017 challenge. SRDCFir (Felsberg et al., 2015) extends the SRDCF (Danelljan et al., 2015b) tracker for TIR data by adding motion features. SRDCF is a DCF-based tracker that introduced a spatial regularization function to penalize those DCF coefficients that reside outside the target region, which mitigates the damaging boundary effects present in the traditional DCF.

Another branch for TIR tracking combines the input TIR data with the visual modality, concretely with the image intensity given as a grayscale image. For example, Li et al. (2016) propose an adaptive fusion scheme to incorporate information from grayscale images and TIR videos during tracking. Similarly, the approach by Li

et al. (2017a) samples a set of patches around the object and extracts a joint sparse representation in both grayscale and TIR modalities.

The usage of generating other modalities was pioneered by Hoffman et al. (2016). They used generation of depth data to improve classification on the abundant-data modality (RGB), whereas we use data generation as a source of labeled data for the scarce-data modality (TIR). Xu et al. (2017) use a network which generates TIR images to pre-train the weights. These weights are then applied in a network which is used on RGB data with the aim to improve tracking of pedestrians. Other than them, we use the generation of TIR data for data augmentation; we create large synthetic labeled data sets of TIR data to be able to train end-to-end features for TIR data.

3.2.3 Image-to-Image Translation

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) have achieved promising results in several tasks such as image generation (Denton et al., 2015), image editing (Perarnau et al., 2016), and representation learning (Salimans et al., 2016). The conditional variants of GANs (Mirza & Osindero, 2014) enable to condition the image generation on a selected input variable, for example, an input image. In this case, the task becomes image-to-image translation, and this is the variant we use here. The general method of Isola et al. (2017), pix2pix, was the first GAN-based image-to-image translation work that was not designed for a specific task (e.g. colorization (Zhang et al., 2016b)). The architecture is based on an encoder-decoder with skip connections (Ronneberger et al., 2015) and it is trained using a combination of two losses: a conditional adversarial loss (Goodfellow et al., 2014) and a loss that maps the generated image close to the corresponding target image. This method achieves excellent results, but requires matching pairs of training images, which limits the applicability of the model as such data might not be easily accessible. In order to overcome this limitation, Zhu et al. (2017) extended this model to the case in which paired data is not available. Their method, called CycleGAN, relies on the assumption that mapping an image from the input domain to the target and then back to the input (i.e. the cycle) should result in the identity function. Based on this, they add a cycle consistency loss that enforces the correct reconstruction of the input image resulting of the composition of the two mappings. They demonstrate the effectiveness of their method for multiple tasks such as edges to real images or photo enhancement. In this chapter, we use image-to-image translation to generate a synthetic large-scale TIR tracking dataset from a labeled RGB dataset, with the goal of learning better deep features for tracking.

Related to our work is the research (Berg et al., 2018) where they convert TIR images to RGB images. This task is more difficult than the task which we address in this

chapter, because the TIR modality contains less information than the RGB modality. Berg et al. (2018) proposed two Convolutional Neural Networks based methods which colorize TIR images to plausible visual luminance and chrominance. These methods are fully automatic. They do not require any user input, pre-processing, and post-processing, while they are still robust to image pair misalignment. Extensive qualitative and quantitative experiments on the public dataset demonstrate the proposed methods generate perceptual effects.

3.3 Method Overview

We aim to train end-to-end deep features for tracking in TIR data. However, to train effective deep features for TIR data, we need a large dataset of labeled TIR videos. Unfortunately, the amount of labeled TIR data is very scarce. To the best of our knowledge, only BU-TIV dataset (Wu et al., 2014) contains a considerable amount of labeled TIR videos, but most of them depict only one object class (pedestrian). Therefore, most state of the art TIR tracking methods are still based on hand-crafted features (Felsberg et al., 2015; Yu et al., 2017; Zhu et al., 2016). On the other hand, there are vast amounts of RGB videos labeled for tracking (Kristan et al., 2016a; Wu et al., 2015). One solution could therefore be to use the pre-trained features which are optimal for tracking in RGB data for TIR data. However, this is unlikely to be optimal because TIR and RGB data differ significantly.

To illustrate the difference in nature of RGB and TIR data we measure the average activation of the 96 filters of the first layer of a pre-trained AlexNet on the KAIST dataset. This dataset contains both RGB and TIR images of the same scenes (a similar study for depth images has been performed by Song et al. (2017a)). The pre-trained network is trained to recognize objects in RGB images (i.e. on ImageNet). We present the results in Fig. 3.2. The graph shows the average activation of the filters in descending order. When applied to data which is similar to that on which the network is trained, the average activations tend to produce a uniform distribution. This can be seen for RGB images where most filters yield the same average activation and only a few filters differ from this pattern. When we perform the same experiment on TIR data the pattern changes. We can now observe clear differences between filters which have a higher average activation and filters which have lower average activation. This shows that these filters are probably not optimal for TIR tracking. When we look at the exact filters which have low and high activation, we see that low frequency patterns (blobs and edges) are prevalent for the TIR data, whereas high-frequency filters are seldom found in TIR data. This is not surprising since most textures, responsible for most of the high-frequency content of images, do not appear in TIR data. In conclusion, given the different nature of the image

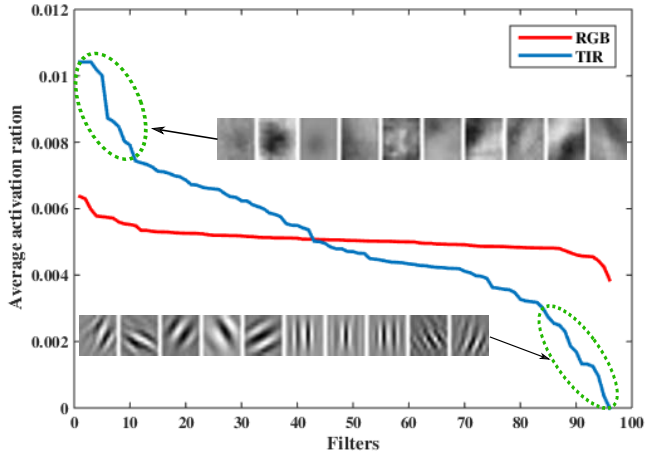


Figure 3.2 – Average activation of filters from the first layer of pre-trained AlexNet on the test set of KAIST for RGB and TIR images.

statistics of RGB data and TIR data it is probable that a network which is trained on TIR data would outperform a network trained on RGB data.

In this chapter, we aim to address the problem of data scarcity of labeled videos for tracking in TIR data. We do this by exploiting the vast amount of labelled RGB videos which are available, in combination with recent advances in image-to-image translation techniques. We will use these image-to-image translation models to transfer large labeled RGB datasets to synthetic TIR dataset together with the tracking annotations. As a result we now have a large labeled synthetic TIR dataset. We use this synthetic TIR dataset to train end-to-end deep networks to obtain optimal TIR features for tracking. Then we plug the optimal TIR feature model into a state-of-the-art tracker. Here we use ECO (Danelljan et al., 2017). An overview of our method is provided in Fig. 3.3. In the following section we detail the various parts of our algorithm.

3.4 Deep Learning Features for Correlation Filter Tracking

In this section, we introduce the standard correlation filter and the current end-to-end deep correlation filter. Then we describe Efficient Convolution Operators

3.4. Deep Learning Features for Correlation Filter Tracking

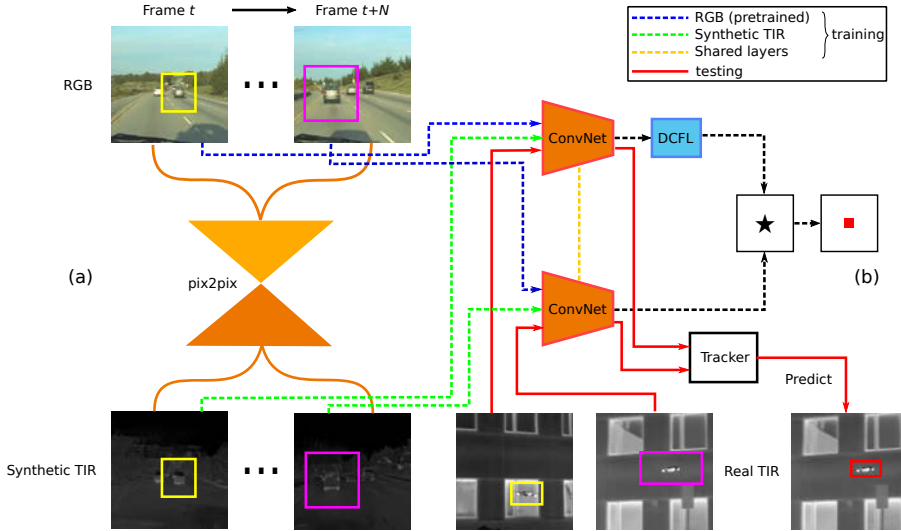


Figure 3.3 – **Overview of our approach.** (a) Image-to-image translation component (proposed by Isola et al. (2017)) for generating a large labeled synthetic TIR tracking dataset. We use blue dashed line to represent the baseline RGB training model and the green dashed line represents our proposed synthetic data training model. After the translation of RGB data to TIR data, we acquire enough suitable data for end-to-end training networks for TIR tracking. (b) Two-branch architecture for training the network to obtain adaptive features for TIR tracking (proposed by Valmadre et al. (2017)). The optimal correlation filter is computed in the discriminative correlation filter layer (DCFL) for the image processed in the upper branch. This filter is then applied on the image in the bottom branch.

(ECO) (Danelljan et al., 2017) which we will use as the correlation filter for our experiments.

3.4.1 Correlation Filter Tracking

The conventional discriminative correlation filters (DCF) formulation (Henriques et al., 2015), learns a linear correlation filter f that discriminates the target appearance from the background. The target location is predicted by applying the correlation filter to a sample feature map. The desired filter f can be obtained by

minimizing a least squares objective:

$$E(f) = \left\| \sum_{d=1}^D f^d * x^d - y \right\|^2 + \lambda \sum_{d=1}^D \|f^d\|^2. \quad (3.1)$$

Here $*$ denotes circular correlation. x^d denotes feature maps of training samples x , where the layer $d \in \{1, \dots, D\}$. f^d denotes channel d of filter f . y is the regression target and λ is a regularization weight to control over-fitting. A closed-form solution is obtained in the Fourier domain,

$$\hat{f}^d = \frac{\hat{x}^d \hat{y}^*}{\sum_{d=1}^D \hat{x}^d (\hat{x}^d)^* + \lambda}. \quad (3.2)$$

Where the \hat{y}^* denotes the complex conjugate of the discrete Fourier transform $\mathcal{F}(y)$.

Recently, researchers have proposed several methods for end-to-end training of features for tracking: CFNet (Valmadre et al., 2017), DCFNet (Wang et al., 2017), and CFCF (Gundogdu & Alatan, 2017). All use a two-branch Siamese network, of which one branch is used to compute the optimal correlation filter, which is applied in the other branch to obtain the response map (see Fig. 3.3). Both branches share the weights of their convolutional layers. Training is performed with paired images from same video. It backpropagates the gradients through the discriminative correlation filter layer (DCFL) with a closed-form solution (Valmadre et al., 2017). Surprisingly, trackers based on end-to-end training only slightly outperformed off-the-shelf features. It should be noted that all end-to-end trackers train on RGB datasets, mainly on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC15) (Russakovsky et al., 2015), and no results for end-to-end tracking on other modalities like TIR are available.

In this chapter, we use the end-to-end CFNet training procedure proposed by Valmadre et al. (2017). This method obtains stable and fast network training due to their Fourier domain implementation of the discriminative correlation filter layer. Other than them we will apply it to TIR tracking. Since current available datasets for TIR tracking are rather small, we propose in the next section our approach to generating synthetic TIR tracking data from labeled RGB tracking data.

3.4.2 Efficient Convolution Operators

Previously, we have explained how to train end-to-end features for tracking. These features can be used in different discriminative correlation filter methods. In our work we use the Efficient Convolution Operator (ECO) (Danelljan et al., 2017)

3.4. Deep Learning Features for Correlation Filter Tracking

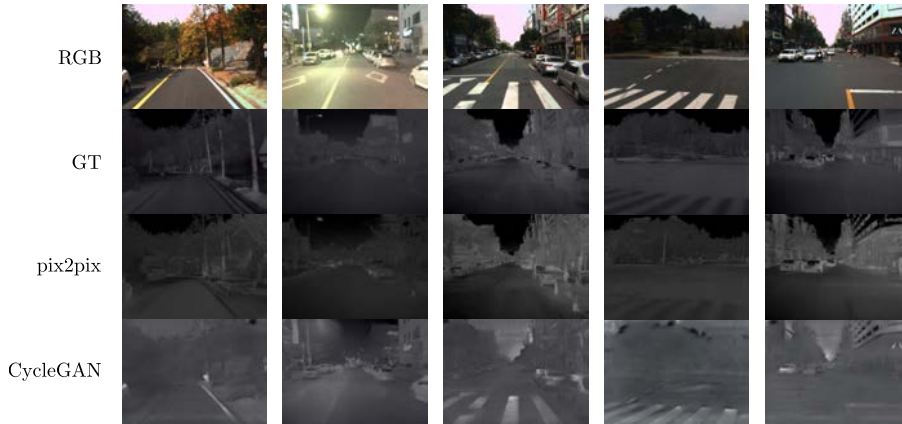


Figure 3.4 – **Results for the two image translation methods considered: pix2pix and CycleGAN.** The video frames are taken from the test set of KAIST, and have not been seen during training.

method, shown to obtain state-of-the-art results while being computationally efficient. However, its original implementation is based on features extracted from a pre-trained CNN model trained on the ImageNet 2012 classification dataset (Rusakovsky et al., 2015). Even though these features are extracted from a model which is trained for image classification, ECO obtains excellent results for tracking. In our experiments we combine ECO with the end-to-end trained features for TIR tracking.

The ECO tracker aims at combining shallow and deep features by learning a multi-channel continuous convolution filter in a joint optimization scheme across all feature channels. Furthermore, it learns a projection matrix, to reduce the dimensionality of high-dimensional features. Here we briefly describe the training and inference procedures applied in ECO.

ECO learns the target model parameters based on a set of training samples $\{x_j\}_1^M$ and corresponding labels $\{y_j\}_1^M$. The label function y_j consists of the desired target scores at all spatial locations in the corresponding training sample x_j . It is defined as a periodically repeated Gaussian function centered at the sample location (Danelljan et al., 2016b). Each training sample contains multiple feature layers $x_j^d \in \mathbb{R}^{N_d \times N_d}$, where N_d is the spatial resolution of layer $d \in \{1, \dots, D\}$. These feature layers correspond to both shallow and deep features of varying resolutions.

The tracker predicts the target location using the target score operator, defined as

$$S_{f,P}\{x\} = \sum_{d=1}^D f^d * PJ_d\{x^d\}. \quad (3.3)$$

Here, x is the input sample and f is the learned filter that predicts the detection score function $S_{f,P}\{x\}$ of the target. The sample x is first interpolated to the continuous domain using the operator J_d , by applying a cubic spline kernel in the Fourier domain (see the method by Danelljan et al. (2016b) for more details). The projection matrix P is then applied to reduce the dimensionality of the feature space.

The detection score operator is learnt via minimization of a least squares objective,

$$E(f) = \sum_{j=1}^M \alpha_j \|S_{f,P}\{x_j\} - y_j\|^2 + \sum_{d=1}^D \|w f^d\|^2 + \lambda \|P\|_F^2. \quad (3.4)$$

Here, the projection and filter are regularized by a constant λ . The spatial regularization weight function w is employed to mitigate the effects of periodic repetition (Danelljan et al., 2015b). Each sample x_j is weighted by α_j , based on a learning rate parameter γ . The label functions $\{y_j\}_1^M$ are set to Gaussian functions centered at the target location.

Using Parseval's formula an equivalent loss is obtained as,

$$E(f) = \sum_{j=1}^M \alpha_k \|\widehat{S_{f,P}\{x_j\}} - \hat{y}_j\|^2 + \sum_{d=1}^D \|\hat{w} * \hat{f}^d\|^2 + \lambda \|P\|_F^2. \quad (3.5)$$

Here $\hat{\cdot}$ denotes the Fourier coefficients. We learn the projection matrix P jointly with the filter f in the first frame by applying Gauss-Newton and adopting the Conjugate Gradient method (Nocedal & Wright, 2006) for each iteration. In subsequent frames, the resulting normal equations are efficiently solved using the method of Conjugate Gradients, assuming a fixed P . For more details, we refer to the trackers Danelljan et al. (2017, 2016b).

3.5 Generating TIR images

In this section we discuss image-to-image translation methods and compare them for the task of transferring RGB to synthetic TIR data.

3.5.1 Image-to-Image Translation Methods

We use two different image-to-image translation methods to transform labeled RGB videos into labeled TIR videos. First, we use pix2pix (Isola et al., 2017), which requires paired training data. Therefore, we need matching frames in both RGB and TIR, which we can obtain from multispectral video datasets such as KAIST (Hwang et al., 2013). Second, we use CycleGAN (Zhu et al., 2017), an extension on pix2pix that can be trained from unpaired data. As a consequence, any videos in the RGB and TIR modalities can be used to train CycleGAN. Despite the higher availability of unpaired training data, we expect the weaker supervision of CycleGAN to result in synthesized TIR images of lower quality. In this section, we present both translation methods and experimentally confirm this intuition. In later sections, we generate TIR data using only pix2pix given its empirically superior performance.

Both methods are based on Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) conditioned on input images. GANs consist of two networks, generator G and discriminator D , that compete against each other. The generator tries to generate samples that resemble the original data distribution, whereas the discriminator tries to detect whether samples are real or have been generated by G . When the GAN architecture is conditioned on an input image, the task becomes image-to-image translation. In our case, the input image is a color frame from an RGB video and the target is the matching frame in the TIR modality.

Paired - pix2pix. pix2pix (Isola et al., 2017) is an effective, task-agnostic method that can be applied to translate between many domain pairs, including maps to satellite pictures, edge maps to real pictures, or grayscale images to color images. The generator is based on an encoder-decoder architecture with skip connections (U-Net proposed by Ronneberger et al. (2015)). The discriminator is a convolutional PatchGAN (Li & Wand, 2016), which classifies each local image patch independently, making it especially suited for modifying textures or styles.

Let x be an image from the input domain X and y an image from the target domain Y . In pix2pix, both the generator and discriminator are conditioned on the input image x . The conditional GAN objective function is defined as

$$\begin{aligned} \mathcal{L}_{cGAN}(G, D) = & \mathbb{E}_{x,y}[\log D(x, y)] \\ & + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))], \end{aligned} \quad (3.6)$$

where z is a random noise vector used as input for the generator. Additionally, pix2pix also includes an L1 loss to increase the sharpness of the output images

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]. \quad (3.7)$$

The final objective function is the weighted sum of these two losses. Following the original adversarial training (Goodfellow et al., 2014), G tries to minimize this final objective while D tries to maximize it:

$$G^* = \operatorname{argmin}_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G). \quad (3.8)$$

We translate an RGB video to TIR by applying pix2pix independently for each video frame. The original model of Isola et al. (2017) achieves mild stochasticity in its outputs by keeping the dropout layers at test time, which are normally used only during training. In our case, this is not only unnecessary but also damaging, as it makes the output video less stable. For this reason, we only use dropout layers during training.

Unpaired - CycleGAN. Paired data might be hard to come by for particular tasks including RGB to TIR conversion, as the amount of paired videos in both modalities is rather limited. Zhu et al. (2017) present CycleGAN, a method for learning to translate between image domains when paired examples are not available. The main idea consists in adding a cycle consistency loss, based on the assumption that mapping an image $x \in X$ to domain Y and back to X should leave it unaltered. For this reason, besides the classic generator $G: X \rightarrow Y$, CycleGAN also learns a generator to perform the inverse mapping $F: Y \rightarrow X$. The method is then trained with a weighted combination of an unconditional adversarial loss (Goodfellow et al., 2014) and the cycle consistency loss in both directions

$$\begin{aligned} \mathcal{L}_{cyc}(G, F) = \mathbb{E}_x[\|F(G(x)) - x\|_1] \\ + \mathbb{E}_y[\|G(F(y)) - y\|_1], \end{aligned} \quad (3.9)$$

See the method (Zhu et al., 2017) for more details. As in pix2pix, we apply CycleGAN independently per frame, and we remove the dropout layers at test time to generate a more stable video output.

3.5.2 Datasets

We consider multiple datasets for training our image translation methods, spanning the two presented supervision levels: paired and unpaired. Table 3.1 details the number of images of all the considered datasets. Among the paired datasets, the biggest and most relevant is KAIST Multispectral Pedestrian Dataset (Hwang et al., 2013), which contains a significant amount of aligned images in the RGB and TIR modalities, captured from a moving vehicle in different urban environments and under different lighting conditions. We follow the official data split (Hwang et al.,

Type	Dataset	Number of images	
		RGB	TIR
Paired	KAIST (Hwang et al., 2013)	50,184	50,184
	CVC-14 (González et al., 2016)	8,473	8,473
	OSU Color Thermal (Davis & Keck, 2005)	8,545	8,545
	VAP Trimodal (Palmero et al., 2016)	5,924	5,924
	Bilodeau (Bilodeau et al., 2014)	7,821	7,821
	LITIV2012 (Torabi et al., 2012)	6,325	6,325
	total	87,088	87,088
Unpaired	VOT2016 (Kristan et al., 2016b)	21,455	-
	VOT2017 (Kristan et al., 2017a)	4,049	-
	OTB (Wu et al., 2015)	58610	-
	ASL (Portmann et al., 2014)	-	6,490
	Long-term (Gade et al., 2013)	-	47,423
	InfAR (Gao et al., 2016)	-	46,121
	total	84,114	100,034

Table 3.1 – **Datasets used for training the image-to-image translation models.** We test all models using a subset of three videos from the official test set of KAIST (Hwang et al., 2013).

2013) as in Isola et al. (2017) and use all the frames from training videos for training. We evaluate both image translation methods using 3 randomly left out videos from the test set, amounting to 5,728 images. Train and test sets have no videos in common.

Other paired datasets include images of people captured under different conditions: pedestrians during day or night (CVC-14 (González et al., 2016)), static cameras at a busy intersection (OSU Color-Thermal Database (Davis & Keck, 2005)) or in different positions and zooms (LITIV2012 dataset (Torabi et al., 2012), interactions in indoor scenes with controlled lighting settings (VAP Trimodal People Segmentation Dataset (Palmero et al., 2016)), or moving in different planes (Bilodeau et al., 2014). This amounts to a total of 87K image pairs.

We use all paired datasets to train both pix2pix and CycleGAN. Additionally, we collect an RGB-TIR unpaired dataset as extra training data for CycleGAN. As RGB data we include all the sequences from VOT2016 (Kristan et al., 2016b), VOT2017 (Kristan et al., 2017a), and OTB (Wu et al., 2015). As TIR data we include the TIR images from ASL (Portmann et al., 2014), Long-term (Gade et al., 2013), and InfAR (Gao et al., 2016). This amounts to a total of about 230K images, almost $5\times$ more images than the paired training dataset.

3.5.3 Implementation Details

We train all networks in pix2pix and CycleGAN from scratch, initializing the weights from a Gaussian distribution with zero mean and standard deviation of 0.02. We use the same network architectures as in the original papers (Isola et al., 2017; Zhu et al., 2017). As the method (Isola et al., 2017), we apply random jittering by slightly enlarging the input image and then randomly cropping back to the original size. We train pix2pix for 10 epochs, with batch size 4 and learning rate 0.0002. CycleGAN is trained for 3 epochs, batch size 2 and learning rate 0.0002. Note that both models are trained for an equivalent number of iterations given the size of their training sets.

3.5.4 TIR Image Translation Quality

In order to test the two image translation methods considered we select a random subset of the test set of KAIST (Hwang et al., 2013), amounting to about 10% of the entire dataset. We translate the RGB videos into TIR using pix2pix or CycleGAN, and then compute the Euclidean distance of the translations with the TIR ground-truth images. Finally, we average the distance for all frames. pix2pix obtains an average distance of 35.3, whereas CycleGAN obtains 69.5. This demonstrates the superiority of pix2pix for this task, showing how a paired training signal is more valuable than the unpaired counterpart, despite the bigger training dataset of the latter. Fig. 3.4 shows a qualitative comparison of both approaches. We can observe how the translated images using pix2pix are clearly superior to those translated by CycleGAN. Moreover, they look remarkably similar to the ground-truth TIR images, confirming the validity of the proposed data augmentation approach. Therefore, we select pix2pix as our method to generate TIR tracking data from RGB videos.

In addition we compare the statistics of the image gradients of real TIR data and synthetic TIR data generated by pix2pix. The histogram of the gradient magnitude for both datasets on the test set of KAIST is provided in Fig. 3.5. We have also added the gradient magnitude of the grayscale images from which the synthetic dataset is generated. The results show that the gradient magnitude of the synthetic data closely follows that of the real data. Only small variations can be seen for low magnitude gradients. The similarity of the image statistics of real and synthetic data suggests that trackers trained on the synthetic data could also be successful.

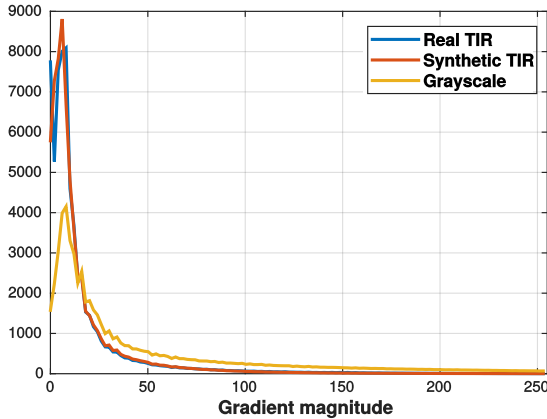


Figure 3.5 – **Histogram of the gradient magnitude for real and synthetic TIR data computed on the test set of KAIST (Hwang et al., 2013).** For comparison we have also added the gradient magnitude histogram for grayscale images from which the synthetic dataset has been generated.

3.6 Experimental Results

3.6.1 Datasets

We train several versions of our tracker using both real and generated TIR tracking data, summarized in Table 3.2. As real TIR data we use BU-TIV (Wu et al., 2014), ASL (Portmann et al., 2014), and OTCBVS (Davis & Keck, 2005). The predominant class in these datasets is human/pedestrian, although BU-TIV (Wu et al., 2014) includes some vehicles and ASL (Portmann et al., 2014) also contains animals like cat and horse. We select all those sequences that include annotated bounding boxes around the objects, leading to a total of 375K bounding boxes from 34K images. On the other hand, we generate synthetic TIR tracking data using the RGB videos from VOT2016 (Kristan et al., 2016b), VOT2017 (Kristan et al., 2017a), and OTB (Wu et al., 2015), which are standard tracking benchmarks used by the community. In total, we obtain 168 TIR videos with tracking annotations by translating the original RGB frames using pix2pix and transferring the corresponding bounding box annotations. The total number of bounding boxes is $4.5\times$ greater than in the real TIR images. Furthermore, the generated TIR videos contain a wider variety of object classes than the real TIR videos. This increases the generality of the learned deep features. In both cases, we leave out around 10% of videos during training as validation set.

Type	Dataset	Videos	Images	Bounding-boxes
Real	BU-TIV (Wu et al., 2014)	5	23,393	34,7291
	ASL (Portmann et al., 2014)	13	6,490	7,773
	OTCBVS (Davis & Keck, 2005)	4	4861	19,944
	Total	22	34,744	375,008
Generated	VOT2016 (Kristan et al., 2016b)	60	21,455	21,455
	VOT2017 (Kristan et al., 2017a)	10	4,049	4,049
	OTB (Wu et al., 2015)	98	58,610	58,610
	Total	168	84,114	84,114

Table 3.2 – **Datasets used for training the tracker, using real TIR data or generated TIR data from RGB images.**

We evaluate our TIR tracker on the VOT-TIR2017 dataset (Kristan et al., 2017a), which is identical to VOT-TIR2016 dataset (Felsberg et al., 2016) as the 2016 edition of this benchmark was far from being saturated. It contains 25 TIR videos of varying image resolution, with an average sequence length of 740 frames adding up to a total of 13,863 frames. Each sequence has been manually annotated with exactly one bounding box per frame around a particular object instance. There is a wide variety of object classes, including pedestrian, animals such as rhino or bird, and vehicles like quadcopter or car. Moreover, the dataset includes extra annotations in the form of attributes, either at frame level (e.g. camera motion, occlusion) or at the sequence level (e.g. blur, background clutter). This test dataset has no videos in common with the RGB modality of VOT2016-17 used for training.

3.6.2 Evaluation Measures and Protocol

We follow the measures and evaluation protocol proposed by the VOT-TIR2017 benchmark (Felsberg et al., 2016). The two primary measures are accuracy (A) and robustness (R), which have been shown to be highly interpretable and only weakly correlated (Čehovin et al., 2016). Accuracy is computed as the overlap (intersection over union) between the predicted track region and the ground-truth bounding box, averaged over frames. The VOT protocol establishes that when the evaluated tracker fails, i.e. when the overlap is below a given threshold, it is re-initialized in the correct location five frames after the failure. In order to reduce the positive bias introduced by this protocol, the accuracy measure ignores the first ten frames after the re-initialization when computing the average overlap. Robustness measures the number of times the tracker fails for each sequence and then takes the average over all sequences. These two measures are conflated into a third, the Expected Average Overlap (EAO), which is the main measure used to rank the trackers. The

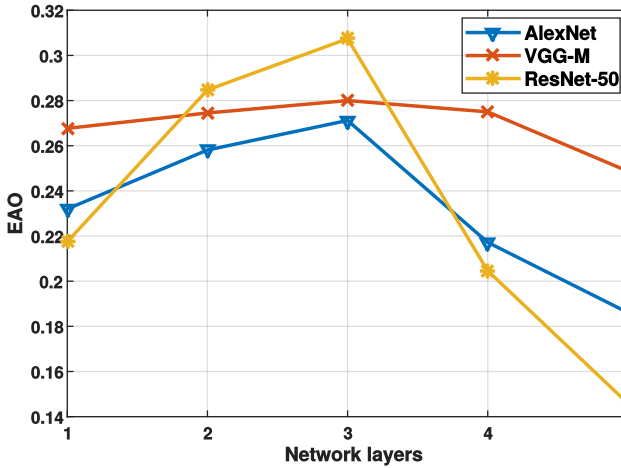


Figure 3.6 – The EAO on VOT-TIR2017 (Kristan et al., 2017a) when using deep features extracted from different layers.

EAO estimates the expected average overlap of a tracker for a particular sequence of a fixed, short length. We refer the reader to the paper (Kristan et al., 2016a) for more details.

Besides the standard VOT metrics, we also report results following the One-Pass Evaluation (OPE) protocol originally proposed by Wu et al. (2015). The most standard evaluation metric used with this protocol is success rate. For each frame in the test video, we compute the overlap between the predicted track and the ground-truth bounding box. A predicted track is considered successful if its overlap with the ground-truth is above a particular threshold. We obtain a success plot by evaluating the success rate at different overlap thresholds. Conventionally, the Area Under the Curve (AUC) of the success plot is reported as a summary measure. Note how this protocol does not reset the tracker in case of failure. We use the VOT toolkit (Kristan et al., 2017a) to compute the measure and plot the results.

3.6.3 Implementation Details

We train CFNet following the method (Valmadre et al., 2017). We perform tests with three different networks as base model: AlexNet (Krizhevsky et al., 2012), VGG-M (Chatfield et al., 2014), and ResNet-50 (He et al., 2016). As the method (Valmadre et al., 2017), we reduce the total stride of the networks from 16 to 4 by changing the stride of the first and second pooling layers from 2 to 1 in AlexNet, and that

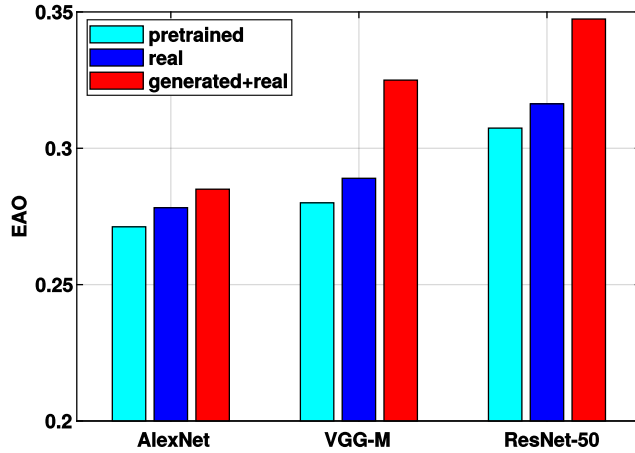


Figure 3.7 – The EAO on VOT-TIR2017 (Kristan et al., 2017a) when using deep features extracted from different networks. Our synthetic data can benefit general networks for fine-tuning.

of second convolutional and pooling layers in VGG-M. This allows us to obtain bigger feature maps, which benefits the correlation filters. For fairness, we apply this modification to all trained models. As training input data for the network, we randomly pick object regions from pairs of images from the same video. Specifically we crop a centered region on the object of approximately twice the object’s size, and resize it to 125×125 pixels. We use Stochastic Gradient Descent (SGD) with momentum of 0.9 and weight decay of 0.0005 to fine-tune the network, which is pre-trained for image classification on ILSVCR12 (Russakovsky et al., 2015). The learning rate is decreased logarithmically at each epoch from 10^{-4} to 10^{-5} . The model is trained for 50 epochs with mini-batches of size 128.

For the baseline tracker ECO (Danelljan et al., 2017) (Fig. 5.2, blue dashed lines), we use the recommended settings (‘OTB_DEEP_settings’) detailed in the code provided by the authors (ECO, 2017). ECO is an RGB tracker, so we have adapted the following parameters given the different nature of TIR data. Following the method (Nam & Han, 2016; Park & Berg, 2018) we use the feature map of the third convolutional layer as the input of the correlation filter, (convolutional block in case of ResNet-50). We confirm these results in the following section. We reduce the learning rate used to update the correlation filter from the 0.009 used for RGB data to 0.003. A smaller learning rate is more suitable for TIR data, as TIR images have less detailed information than RGB, for example lacking texture, and thus

the object appearance remains more stable during tracking. In order to optimally leverage the learned CNN features, we do not add the dimensionality reduction step at the output of each layer as the method (Danelljan et al., 2017). ECO uses this to increase the tracker’s efficiency, which is not a priority in our work.

Our trained models are available at https://github.com/zhanglichao/generatedTIR_tracking.

3.6.4 Network Layers

Our tracker uses deep features from a particular network layer. Previous works (Nam & Han, 2016; Park & Berg, 2018) selected mid-level features from the third convolutional layer as optimal for tracking in RGB videos. Here, we validate this choice for TIR data by analyzing the performance of the selected tracker across all layers for the three networks considered. We perform these experiments using only pre-trained features, i.e., we do not fine-tune the networks for tracking. Fig. 3.6 presents the tracking performance measured by EAO on VOT-TIR2017 (Kristan et al., 2017a) as a function of the network layer. Trackers that use features extracted from the third layer offer best results. Thus we select this features for the remainder of the paper.

3.6.5 Network Architectures

In this section, we experiment with all three base networks with different types of data used for training. All models use ECO (Danelljan et al., 2017) as base tracker, in some cases with the adaptations detailed in section 3.4. We consider two baselines, ‘pretrained’ and ‘real’. The first baseline uses features from the corresponding CNN pre-trained for the image classification task. On the other hand, ‘real’ is also fine-tuned using real TIR tracking datasets (sec. 3.6.1). Our tracker (‘generated + real’) combines both real TIR and synthesized from RGB with pix2pix for the fine-tuning process. Fig. 3.7 presents these results. For all base networks, fine-tuning helps when learning effective features for tracking. This shows that the generated data is complementary to the available real data, making the generated data beneficial even when a good amount of real data is available. Moreover, the gain granted by fine-tuning the network is significantly higher when augmenting the training dataset with our generated TIR data. The performance boost is especially remarkable for higher capacity models such as ResNet-50, since networks with more parameters require more data to train. For all following experiments, we use ResNet-50 as base network for the trackers.

Tracker	without motion features			with motion features		
	EAO	A	R	EAO	A	R
handcrafted	0.235	0.60	2.74	0.361	0.62	1.12
pretrained	0.307	0.62	2.00	0.381	0.69	1.06
real	0.316	0.62	2.01	0.409	0.67	1.24
generated	0.321	0.63	2.00	0.419	0.65	0.83
generated → real	0.331	0.61	1.76	0.429	0.63	0.82
generated + real	0.347	0.63	1.68	0.436	0.65	0.80

Table 3.3 – **Comparison of different tracker variants with and without motion features.** Results are on the VOT-TIR2017 benchmark (Kristan et al., 2017a) with ResNet-50 (He et al., 2016) as base network. Boldface indicates the best results. In both cases, the best results are achieved when combining both real and generated TIR data.

3.6.6 Results on Real and Generated Data

We now present more detailed results for different configurations of our ECO tracker with ResNet-50. We include another baseline (‘handcrafted’) that employs hand-crafted features as it is prevalent in TIR tracking (Felsberg et al., 2015; Yu et al., 2017; Zhu et al., 2016), and thus has not been trained using data. The variant called ‘generated’ is fine-tuned using only our generated TIR data with pix2pix. Finally, we consider another way of combining real and generated data to train the model, ‘generated → real’, which besides uses a two-step fine-tuning mechanism, first using generated data and then real data. This is opposed to ‘generated + real’, which fine-tunes using both real and generated data simultaneously without distinction. Table 3.3 presents the results for all these models using metrics EAO, A, and R on the VOT-TIR2017 dataset (Kristan et al., 2017a).

First, we observe how the use of deep features is fundamental for the success of this tracker, given the low accuracy of the hand-crafted model. Simply using pre-trained features already provides a significant improvement in terms of EAO. Fine-tuning the model on real data brings further benefits. Interestingly, fine-tuning only on the generated data with pix2pix results in better performance than fine-tuning on the real data; with an EAO of 0.289 on real data and 0.300 on generated data. This demonstrates our intuition that having great amounts of diverse data is very relevant when learning specialized deep features for TIR tracking. Finally, simultaneously using both real and generated data to fine-tune the network results in our best model. Moreover, training without distinguishing between the two types of data leads to better results, as opposed to a more complex two-stage fine-tuning

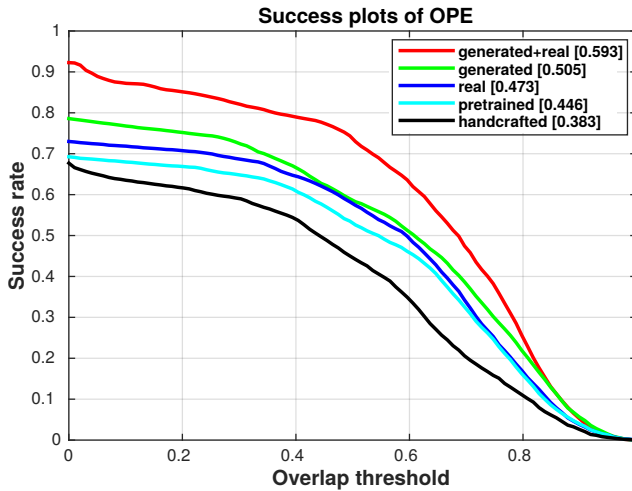


Figure 3.8 – The success plot of one-pass evaluation (OPE) on the the VOT-TIR2017 benchmark (Kristan et al., 2017a). We show the AUC score of each tracker in the legend. The best results are obtained when using both real and generated data.

process.

We present results using the OPE evaluation metric in Fig. 3.8. Also under this metric, hand-crafted features show a clearly inferior performance compared to deep features. Simple pre-trained deep features obtain higher success rates, especially for mid-range overlap thresholds. Fine-tuning on real data gives the tracker a small boost, and when fine-tuning using our generated data, the performance is further improved. Finally, the best performance is achieved when fine-tuning using both types of data simultaneously.

Finally, we analyze the performance of our generated + real tracker for different amounts of generated TIR data. Fig. 3.9 shows EAO as a function of the percentage of synthetic TIR data in the total training set. Interestingly, increasing the amount of synthetic data monotonically improves the tracker performance. The rightmost point, which corresponds to using all our generated data (90% of the training set), does not seem to be saturated, and thus additional generated data could bring an even further performance boost.

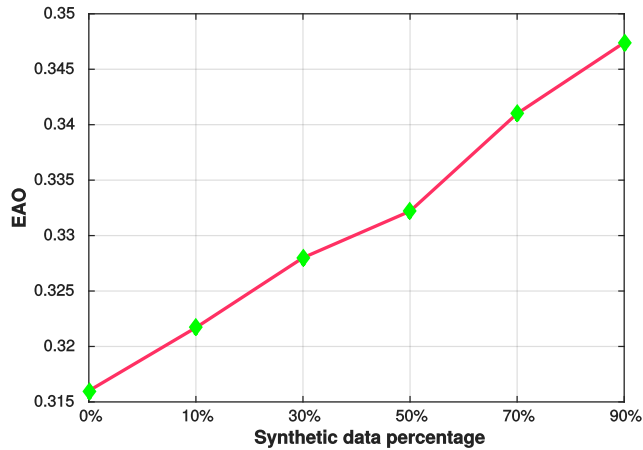


Figure 3.9 – Performance of our tracker (generated+real) on VOT-TIR2017 (Kristan et al., 2017a) for different percentages of synthetic data. The leftmost point indicates using only real data.

3.6.7 Adding Motion Features

As detailed in the paper (Felsberg et al., 2015), the use of hand-crafted motion features can substantially improve tracking performance for TIR data. Following the implementation of the SRDCFir tracker (Felsberg et al., 2015), we compute motion features by thresholding the absolute pixel-wise difference between the current and the previous frame. We then use this motion mask as an extra feature channel.

Table 3.3 presents the results of our trained models when motion features are used alongside deep features. We can see how motion features provide significant performance improvements to all models. Furthermore, the conclusions drawn in the previous experiment still hold. The models trained with generated data outperform both the pre-trained model and the model trained with real data only. Finally, the model trained with a combination of generated and real data achieves an impressive performance, surpassing other methods.

We show a qualitative comparison baseline ECO (pretrained) and ours (generated) in Fig. 3.1. In challenging cases (e.g. third row), the improved deep features learned using our generated TIR data lead to a tracking model that is accurate while robust to various problematic conditions such as occlusion, scale change, and out-of-plane rotation.

Tracker	EAO	A	R
CREST	0.215	0.56	4.13
CSRDCF	0.248	0.57	3.49
TCNN	0.287	0.62	2.79
SRDCFir	0.364	0.63	1.10
EBT	0.368	0.44	0.82
DSLTL	0.401	0.60	0.91
Ours	0.436	0.65	0.80

Table 3.4 – **Comparison with state-of-the-art trackers on VOT-TIR2017 (Kristan et al., 2017a)**. Boldface indicates the best results. The results are reported in terms of expected average overlap (EAO), robustness (failure rate) and accuracy. Our proposed tracker significantly outperforms the state-of-the-art by achieving an EAO score of 0.436.

3.6.8 State-of-the-Art Comparison

Here, we compare our best model with the three top TIR trackers in the VOT-TIR2017 challenge (Kristan et al., 2017a), i.e. DSLTL (Yu et al., 2017), EBT (Zhu et al., 2016), and SRDCFir (Felsberg et al., 2015). We also include in our comparison the best CNN-based tracker in VOT-TIR2016, TCNN (Nam et al., 2016). Additionally, we compare with recently introduced CF-based (CSRDCF by Lukezic et al. (2017)) and spatial CF-based (CREST by Song et al. (2017b)) trackers. These trackers have shown excellent performance on VOT (Kristan et al., 2016a) and OTB (Wu et al., 2015) RGB datasets.

Table 3.4 shows the comparison of our best model (generated+real) including motion mask with the state-of-the-art methods in literature on the VOT-TIR2017 benchmark (Kristan et al., 2017a). Among the existing methods, SRDCFir and EBT achieve EAO scores of 0.364 and 0.368 respectively. An EAO score of 0.287 is achieved by the TCNN tracker. The recently introduced CREST and CSRDCF trackers achieve EAO scores of 0.215 and 0.248 respectively. The current state-of-the-art on this dataset is the DSLTL tracker with an EAO score of 0.401. Our tracker significantly outperforms DSLTL by setting a new state-of-the-art with an EAO score of 0.436. Our approach also achieves superior performance in terms of accuracy and obtains second best results in terms of robustness. We further analyze the robustness of our tracker and found our approach to have promising improvements with respect to robustness in all videos except *trees2*, compared to EBT.

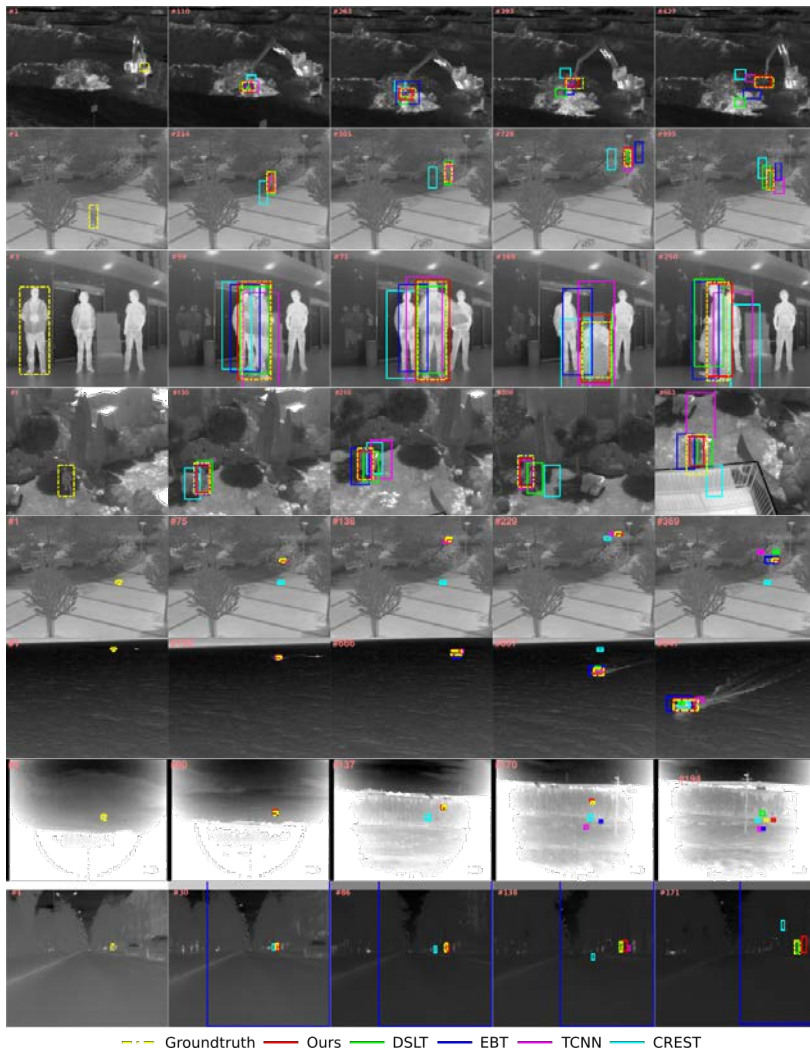


Figure 3.10 – **Qualitative comparison of our approach trained on generated and real data with state-of-the-art trackers.** CREST, TCNN, EBT and DSLT are tested on the some challenging sequences, *excavator*, *jacket*, *mixed_distractors*, *garden*, *quadrocopter2*, *boat2*, *bird* and *trees2* in VOT-TIR2017 (Kristan et al., 2017a). Yellow dashed bounding box means Groundtruth and red solid bounding box is Ours. The last two rows show failure cases of our tracker.

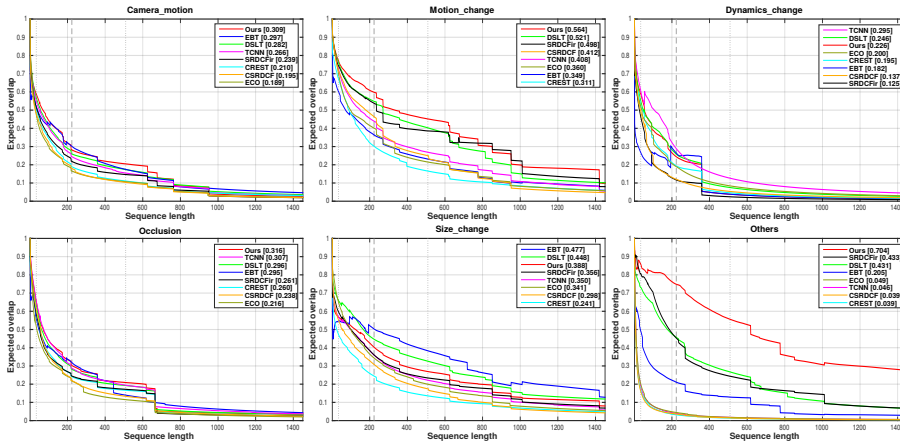


Figure 3.11 – Attribute-based comparison of our trackers with state-of-the-art methods on VOT-TIR2017. We show expected overlap measure for four attributes: camera motion, dynamics change, motion change, occlusion, size change, and others. Our trackers provide consistent improvements in case of camera motion, motion change, occlusion and others, compared to existing methods.

Fig. 3.10 shows a qualitative comparison of our tracker with state-of-the-art methods. Our tracker follows the target object more accurately and is robust to challenging conditions such as scale change and occlusion. Among existing methods, DSLT also provides improved tracking performance but struggles with accurate target localization. The proposed TIR-specialized deep features learned through abundant generated TIR data enable precise target localization, leading to superior tracking results. Finally, the last two rows of the figure show two example cases in which our tracker fails. In the first case, the object is rather tiny and lies on a cluttered background region, which increases the probability of confusing the tracked object with the background. In the second case, there is a considerable scale change combined with heavy occlusion, leading to a poor estimation of the object extent and the corresponding tracking failure.

3.6.9 TIR Data Attributes Analysis

In order to provide a more detailed analysis of the results, we present in Fig. 3.11 the per-attribute performance comparison of our tracker and several state-of-the-art methods. The attributes are: camera motion, dynamics change, motion change, occlusion, size change, and others. Each attribute plot indicates the expected

overlap for every tracker as a function of the sequence length, computed on a particular subset of videos annotated with the corresponding data attribute. For most attributes, including the challenging scenarios of heavy camera motion, motion change, and occlusion, our proposed tracker outperforms state-of-the-art trackers. This consistent improvement on challenging attributes is likely due to specialized discriminative features, learned specifically for TIR tracking. In case of dynamics change, both TCNN and DSLT provide superior tracking performance. The TCNN tracker (Nam et al., 2016) can accurately match object proposals due to a tree structure encompassing multiple CNNs. The DSLT tracker (Yu et al., 2017) also uses dense proposals and structural learning classifier. In case of size change, the EBT tracker (Zhu et al., 2016) and DSLT provide superior results. In this attribute, our approach provides the third best results by outperforming trackers such as SRDCFir and TCNN. Overall, our approach achieves best results on 4 out of 6 attributes.

3.7 Conclusion

In this Chapter, we have proposed a method to generate synthetic TIR data from RGB data. We use recent progress on image to image translation models for this purpose. The main advantage of this is that we can generate a large dataset of labeled TIR sequences. This dataset is far larger than datasets with real labeled sequences which are currently available for TIR tracking. These larger datasets allow us to perform end-to-end training for TIR features. To the best of our knowledge we are the first to train end-to-end features for TIR tracking. We show that our features trained on the synthetic data outperform other features for TIR tracking, including features which are computed by fine-tuning a network on real TIR sequences. In addition, we show that a combination of both real and generated data leads to a further improvement. Once we combine our feature with the motion feature we obtain state of the art results on the VOT-TIR2017.

4 Multi-Modal Fusion for End-to-End RGB-T Tracking *

4.1 Introduction

As an important task in computer vision, visual object tracking, especially RGB tracking (Bertinetto et al., 2016b; Bolme et al., 2010; Danelljan et al., 2017, 2019, 2014a, 2016b, 2014b; Henriques et al., 2015; Li et al., 2018b; Lukezic et al., 2017; Zhu et al., 2018), has undergone profound changes in recent years. Researchers mainly focus on RGB tracking as large datasets are available (Kristan et al., 2016a; Valmadre et al., 2018; Wu et al., 2015). However, RGB tracking obtains unsatisfactory performance in bad environmental conditions, e.g. low illumination, rain, and smog. It was found that thermal infrared sensors provide a more stable signal for these scenarios. Therefore, RGB-T tracking has drawn more research attention recently (Li et al., 2016, 2018c, 2017b, 2018d).

As multi-modal data, i.e. from the RGB and TIR modalities, can provide complementary information for tracking, multi-modal tracking is a promising research direction. Images from the RGB modality have the advantage that they contain high-frequency texture information and provide rich representations for describing objects. Images from the TIR modality have the advantage that they are not influenced by illumination variations and shadows. Moreover, objects with elevated temperature can be distinguished from the background as the background is normally colder. Therefore, fusing the information from multi-modal data could benefit the tracker because it can exploit the complementary information of the modalities to improve tracking performance.

There exists relatively little research on multi-modal tracking (Li et al., 2016, 2017a, 2018d; Liu & Sun, 2012; Wu et al., 2011). Most of these works are still using the sparse representation, normally with the hand-crafted features, for multi-modal tracking (Li et al., 2016, 2017a, 2018d; Liu & Sun, 2012). Later on in the tracker (Li et al., 2018d), for comparison they design some baseline RGB-T trackers by extending the single modal tracker to a multi-modal tracker. This extension is done by

*This chapter is based on a publication in Proceedings of the IEEE International Conference on Computer Vision Workshops 2019

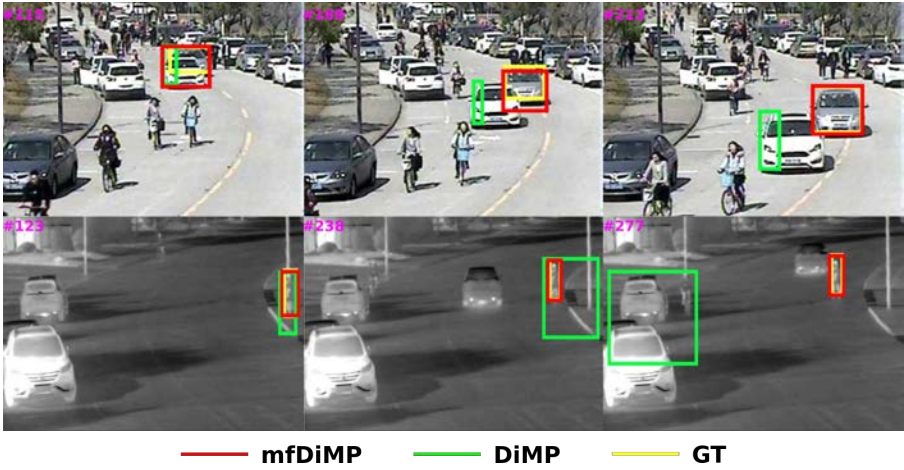


Figure 4.1 – **Qualitative comparison between ‘mfDiMP’ and ‘DiMP’.** Two exemplar videos from RGB modality and TIR modality on the top and bottom separately, where DiMP performs on each of them with single modality input. Our *mfDiMP* can effectively track the object by fusing both modalities.

directly concatenating the features from the RGB and the TIR modalities into a single vector, which is then fed into the tracker. They also use some deep features for concatenation, but they are still off-the-shelf features pre-trained for other tasks. Therefore, there is still no previous work which investigates end-to-end training. We mention two main reasons for this: first, it is not obvious in what part of the tracking pipeline the fusion should be done. Ideally, we should fuse the information of the different modalities in such a way that it allows for optimal end-to-end training. Second, data scarcity of multi-modal tracking data is a major obstacle to end-to-end training. Currently there are no large-scale aligned multi-modal datasets. These two issues, i.e. no specific fusion scheme and lack of data, limit the progress of end-to-end multi-modal training.

To tackle this problem, in this chapter we investigate how to effectively fuse multi-modal data in an end-to-end training manner, which could make optimal use of information from both modalities (see Figure 4.1). We propose three end-to-end multi-modal fusion architectures, consisting of pixel-level fusion, feature-level fusion and response-level fusion. Here we use as a baseline tracker the RGB tracker DiMP (Bhat et al., 2019). To ensure that the proposed fusion tracker can be trained in an end-to-end manner, we also generate large-scale paired synthetic RGB-T datasets with the same method as proposed in Chapter 3. We perform extensive experiments

on two commonly used benchmarks for RGB-T tracking: VOT-RGBT2019 (VOT challenge, 2019) and RGBT210 (Li et al., 2017b). Our multi-modal fusion tracker sets a new state-of-the-art on both datasets, achieving an EAO score of 0.391 on VOT-RGBT2019 and AUC score of 55.5% on RGBT210.

This chapter is organized as follows. In section 4.2, we discuss the successful single modality tracking methods of recent years and the situation of current multi-modal tracking. In section 4.3, we introduce the baseline tracker and analyze its components. In section 4.4, we describe the proposed methods and formulations for the fusion of multi-modal tracking and provide the synthetic data for end-to-end training. In section 4.5, we present our extensive experiments on the VOT-RGBTIR2019 dataset and RGBT210 dataset. Finally, in section 4.6, we conclude our work and propose future research directions.

4.2 Related Work

4.2.1 Single Modality Tracking

Most current tracking algorithms focus on RGB images (Bertinetto et al., 2016b; Bolme et al., 2010; Danelljan et al., 2017, 2019, 2014a, 2016b, 2014b; Henriques et al., 2015; Li et al., 2018b; Lukezic et al., 2017; Zhu et al., 2018), although several approaches track in the TIR modality instead (Li et al., 2018d; Yu & Yu, 2018; Yu et al., 2017; Zhang et al., 2019a). Despite the development of deep learning in many computer vision tasks, object tracking continued to use hand-crafted features during the first stage of deep learning (Danelljan et al., 2017, 2016b; Ma et al., 2015a; Nam & Han, 2016; Song et al., 2018). Later on, some trackers (Danelljan et al., 2017, 2016b; Ma et al., 2015a) pioneered in the involvement of deep features in tracking by using the pre-trained models for an image classification task (Russakovsky et al., 2015). The main reasons for only using pretrained models were the lack of large-scale training datasets and the difficulty of designing a suitable end-to-end training framework for tracking. Bertinetto et al. (2016b) proposed to train a network end-to-end by using a video object detection dataset (Russakovsky et al., 2015). Recently, several large-scale tracking datasets (Fan et al., 2018; Huang et al., 2018; Muller et al., 2018; Valmadre et al., 2018), e.g. GOT-10k (Huang et al., 2018), have been released with millions of images and various categories for training. Therefore, some current tracking approaches (Bhat et al., 2019; Danelljan et al., 2019; Li et al., 2018a) perform end-to-end training by leveraging these large-scale datasets.

RGB trackers. Bertinetto et al. (2016b) proposed to use a fully-convolutional architecture to learn a similarity metric offline, i.e. a Siamese network. After training, the Siamese network is deployed for online tracking with high efficiency. To learn

attention on the cross correlation, Wang et al. (2018) include additional attention components in the Siamese network and learn the spatial and channel weights for the *exemplar* model. Li et al. (2018b) utilize a proposal network to estimate the score maps and bounding boxes using two branches, which provides more accurate object scales than the traditional multi-resolution scale estimation. Later it is extended to use deeper and wider networks achieving significant improvement (Li et al., 2018a).

An alternative approach to Siamese networks is correlation filter (CF) based tracking (Bolme et al., 2010; Danelljan et al., 2014a, 2015b, 2016a, 2014b; Galoogahi et al., 2013; Henriques et al., 2015; Kiani Galoogahi et al., 2017; Lukezic et al., 2017; Ma et al., 2015b; Mueller et al., 2017a; Zhang et al., 2014, 2016a, 2019b), which has occupied top positions for many years given its discriminative abilities and efficient tracking speed. The core part of CF trackers is the calculation of a filter that is later applied to detect the object in the search region of next frame. The calculation is performed in the Fourier domain, which makes it highly efficient. To overcome the issue of boundary effect in correlation filter in tracking, Danelljan et al. (2015b) proposed to regularize the filter with a Gaussian window and Kiani Galoogahi et al. (2017) proposed to use a mask formulated in the correlation filter. Some CF trackers (Danelljan et al., 2017, 2015a, 2016b; Ma et al., 2015a) also benefited from pre-trained convolutional features. CFNet (Valmadre et al., 2017) added end-to-end training by formulating CF as one layer of the network, although this only gives a marginal gain with respect to the baseline model, SiamFC (Bertinetto et al., 2016b). Park & Berg (2018) proposed to learn an initial model for the correlation filter offline, accelerating the convergence speed for the filter optimization.

TIR trackers. Contrarily to RGB tracking, most of the top performing TIR trackers still use hand-crafted features in their models. For example, SRDCFir(Felsberg et al., 2015) extends the SRDCF(Danelljan et al., 2015b) tracker for TIR data by combining motion features with hand-crafted visual features, e.g. HOG(Dalal & Triggs, 2005), color names(Van De Weijer et al., 2009), intensity, etc. EBT(Zhu et al., 2016) uses edge features to devise an objectiveness measure that generates high quality object proposals. Yu & Yu (2018); Yu et al. (2017) propose structural learning on dense samples around the object, using edge and HOG features (Dalal & Triggs, 2005), transferred to the Fourier domain for efficiency. In Chapter 3, we propose using an end-to-end trainable deep network. They generate a large-scale TIR tracking dataset for training from existing RGB tracking datasets. They use a current image translation approach (Isola et al., 2017) to synthesize a large amount of TIR images from RGB and they transfer the corresponding object annotations. By training the network with this data, they achieve state-of-the-art results in TIR tracking. Following this idea, we obtain a large-scale RGB-T dataset that enables the use of deep learning for RGB-T tracking.

4.2.2 Modality Fusion Tracking

Fusing the RGB and TIR modalities is a promising direction. Some RGB-T trackers once have been proposed. Conaire et al. (2008) proposed to efficiently combine visible and thermal features by fusing the outputs of multiple spatiogram trackers, which is a derivation from mean-shift type algorithm (Birchfield & Rangarajan, 2005). Wu et al. (2011) used a sparse representation for the target template by concatenating RGB and TIR image patches. Similarly, Liu & Sun (2012) also use a sparse representation by minimizing the coefficients from each modality. However, these methods provide sub-optimal fusions as both modalities contribute equally, while in practice one modality may have more valuable information than the other. Li et al. (2016, 2017a) addressed this with an adaptive fusion scheme to integrate visual and thermal information in the sparse representation by introducing weights to balance the contribution of each modality. In order to limit the effect of background clutter during tracking, Li et al. (2018d) introduced a ranking between the two modalities, which is taken into account in the used patch-based features. They effectively avoided background effects by using the learned features with a structured SVM.

As far as we know, all of the current RGB-T approaches use hand-crafted features, which significantly limits their tracking performance. Although several RGB-T tracking datasets (Li et al., 2016, 2018c, 2017b) have been recently released, they are only for testing purposes only and are not large enough for training a deep learning based RGB-T tracker. We propose adapting a deep RGB tracker for RGB-T by exploring different types of modality fusion, and performing end-to-end training with partly synthesized RGB-T data.

4.3 Baseline RGB Tracker

In this section, we describe the architecture of the tracker we have selected for our multi-modal tracking experiments. We use the Discriminative Model Prediction (DiMP) tracker (Bhat et al., 2019), which was originally proposed for single modality tracking.

Discriminative Model Prediction. DiMP (Bhat et al., 2019) proposed an end-to-end trainable tracking architecture, capable of learning a powerful discriminative filter by embedding the online learning of the target model into itself. DiMP consists of the following components: feature extractor, model predictor, and target estimation network (IoU-Net (Jiang et al., 2018)). With these carefully designed components and an effective optimization method, they achieve excellent performance on RGB tracking by setting a new state-of-the-art on several RGB tracking

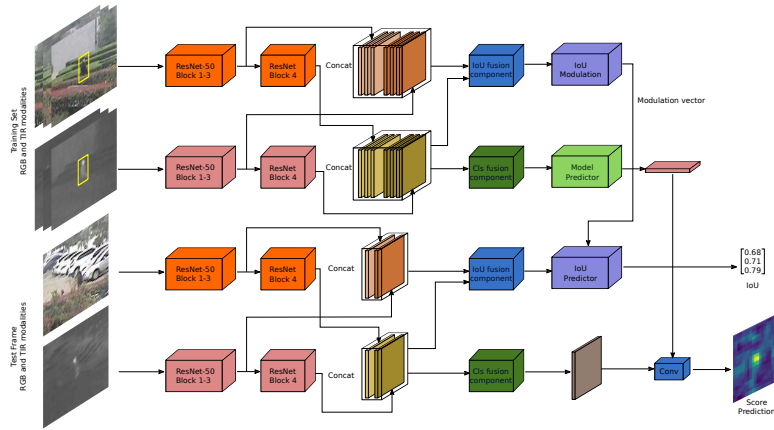


Figure 4.2 – **Overview of our multi-modal fusion framework on feature-level.** We input images from RGB and TIR modalities to the feature extractor separately. Then we fuse deep features from different blocks of the backbone. Fused features from block3 and block4 are input to IoU modulation and IoU predictor. Fused features from block4 are input to model predictor to predict final response map.

datasets (Fan et al., 2018; Huang et al., 2018; Kristan et al., 2018; Muller et al., 2018; Wu et al., 2015).

Feature extractor. The backbone feature extractor F normally aims to extract the deep feature representations for the follow-up implementation models. Here, specifically in DiMP (Bhat et al., 2019), the deep representations are extracted for the model predictor and target estimation network.

DiMP (Bhat et al., 2019) employs the ResNet-18 and ResNet-50 architectures, which is trained on ImageNet, as the backbone feature extractors for DiMP-18 and DiMP-50 separately. They implement fine-tuning the backbone for the end-to-end training. After an analysis on the impact of different feature blocks in DiMP (Bhat et al., 2019), they use the features from block3 and block4 for IoU-Net, and only from block4 for the classifier. The feature extractor F is shared and only performed on a single image patch per frame.

For training the feature extractor F , they input data for F with a pair of sets (M_{train}, M_{test}) . Each set $M = \{(I_j, b_j)\}_{j=1}^{N_{frames}}$ contains images I_j along with their object bounding box b_j . The target model is predicted by using M_{train} and then evaluated on the test frames M_{test} . M_{train} and M_{test} are constructed by sampling N_{frames} frames for both from first and second halves of the segment respectively.

They pass the images through the feature extractor F , and obtain the train set $S_{train} = \{(x_j, c_j)\}$, where $x_j = F(I_j)$, and c_j is the center coordinate of the box b_j .

Model predictor and response map. The Model predictor is to obtain the final optimized filter f , which consists of model initializer, which is a convolutional layer followed by a precise ROI pooling (Jiang et al., 2018), and model optimizer, which is to solve the final model f by the steepest descent (SD). The model filter f is solved by using multiple samples in S_{train} , which happens in the model initializer. The input of the model predictor is a set of S_{train} , and obtain the model f by training on the model predictor: $f = D(S_{train})$. Then the filter f is evaluated on the test samples S_{test} and finally classification loss for offline training is computed as:

$$L_{cls} = \frac{1}{N_{iter}} \sum_{i=0}^{N_{iter}} \sum_{(x,c) \in S_{test}} \left\| l(x * f^{(i)}, z_c) \right\|^2. \quad (4.1)$$

Where, z_c is a Gaussian function centered as the target c . $f^{(0)}$ is the output of model initializer. The response map can be calculated as: $s = x * f, x \in S_{test}$.

Bounding box estimation. DiMP uses an IoU-Net based architecture from the ATOM tracker (Danelljan et al., 2019). The function of the IoU-Net model is to predict the IoU between the deep feature x of an image and a bounding box candidate B . Bounding box estimation is then performed by maximizing the IoU prediction. The network has two branches, one is the *IoU modulation* for calculating the modulation vector from reference image, and the other branch is *IoU predictor* for predicting the IoU values from test image. Then the reference branch is added with a convolutional layer, while the test branch is added with two convolutional layers as it dominates the IoU prediction. Both of them then are followed by PrPool (Precise ROI Pooling) (Jiang et al., 2018) and a fully connected layer. Here the interaction between the two branches is that a precomputed vector in the reference branch is used to modulate the feature representation of the test image via channel-wise correlation. The IoU is predicted in terms of the bounding box B as follows:

$$IoU(B) = g(c(x_0, B_0) \cdot z(x, B)) \quad (4.2)$$

Where, x_0, B_0 are from the reference image, and x, B are from the test image. z is the feature representation after PrPool layer in test branch. g is the IoU predictor with three fully connected layers. c is a modulation vector.

4.4 End-to-End Multi-Modal Tracking

There are two main issues when extending state-of-the-art RGB trackers to multi-modal data such as RGB-T. First, a fusion component is not considered as a native design component for the RGB tracker architecture, since the tracker only considers a single modality as input. Therefore, when extending to multi-modal data these trackers must be equipped with a fusion strategy. Second, the lack of large-scale paired RGB-T training datasets complicates training of end-to-end representations, which have shown to significantly improve results for RGB tracking. To tackle the former, we investigate how to effectively fuse multi-modal data for tracking, aiming to make the best use of all available data modalities, in this case, RGB and TIR. To tackle the latter, we ensure that the proposed multi-modal tracker can be trained in an end-to-end manner by generating a large-scale paired synthetic RGB-T dataset, similarly to the method proposed in Chapter 3.

In this section, we first comprehensively explain our three end-to-end multi-modal fusion architectures, including pixel-level fusion, feature-level fusion and response-level fusion. We also explain how we apply the method in Chapter 3 to generate a large multi-modal data set.

4.4.1 Multi-Modal Fusion for Tracking

In this subsection, we investigate three different mechanisms for multi-modal fusion, with the aim to find the optimal fusion architecture. We start the fusion work on the input of the network (pixel-level). Then we explore the fusion on the intermediate of the network. Li et al. (2018d) extended some RGB trackers by concatenating the RGB and TIR features into a single vector, and then used them as off-the-shelf features for the classifier of the trackers. In contrast, we fuse end-to-end features which are prepared for both the model predictor and target estimation network (feature-level). Moreover, we explore fusion on the final response maps of the network (response-level). Our fusion mechanisms all benefit from end-to-end training. In the experiments (section 4.5), we evaluate and analyze all these fusion strategies and obtain the optimal one.

Pixel-level fusion. The first modality fusion we consider is at the input of the network. We propose to fuse the RGB and TIR images by directly concatenating the images along the channel direction and then inputting the fused RGB-T image to the feature extractor. To complete this fusion, we extend the filter size of the first layer in feature extractor from $7 \times 7 \times 3 \times 64$ to $7 \times 7 \times 4 \times 64$. The images which are input to the feature extractor should be concatenated as following: $I^F = [I^V | I^T]$. Here I^V is the image from RGB modality, I^T is the image from TIR modality, and

finally the fused image is I^F .

Feature-level fusion. To delay the fusion to a more semantically aware stage of the network, we evaluate the effectiveness of fusion in the intermediate of the network architecture. Concretely, we implement the fusion after the feature extractor, i.e. fusing the deep feature representations from the RGB and TIR modalities. We pass the RGB and TIR images through the feature extractor separately and extract features from both modalities independently. Then we concatenate the features from each modality and feed them into the IoU predictor and model predictor. Because of this, the representation is more expressive for the IoU predictor and also more discriminative for the model predictor. The framework of our proposed method for multi-modal fusion on feature-level is shown in Figure 4.2. We concatenate the feature representations, output from the feature extractors. The feature concatenation can be expressed as intuitive syntax: $x^F = [x^V|x^T]$. Here, x^V is the features from the RGB modality, x^T is the features from the TIR modality, and x^F is the fused features.

Response-level fusion. To evaluate the effectiveness of ensemble of independently trained trackers on each modality, we perform the multi-modal fusion on the final part of the training architecture in DiMP (Bhat et al., 2019), i.e. response-level fusion. For the response-level fusion, we use a pair of feature extractors and model predictors, for passing each images from RGB and TIR modalities separately. Finally we sum together their response maps to get the fused response map. But we use only one IoU-Net component which only passes single modality, so there are two cases for training the whole network. One is using RGB modality to fine-tune the IoU-Net and the other is to use TIR modality instead. For the fusion on the response map, assuming that we obtain two response maps, s^V from RGB modality and s^T from TIR modality. Then we sum them together to calculate the fused response map, expressed as: $s^F = s^V + s^T$.

4.4.2 RGB-T Data Generation

The lack of large-scale paired RGB-T training datasets also hampers end-to-end tracking in RGB-T datasets. We use the same method in Chapter 3, which proposes to use image-to-image translation methods to generate synthetic TIR data for tracking. In that chapter we show that such data improves results for end-to-end trained TIR trackers. Here we explain how we generated the training data aiming for fine-tuning the pre-trained DiMP models. We take advantage of a normal RGB training dataset for RGB tracking, and then generating the TIR images by a well-trained image-to-image translation model (Isola et al., 2017). With the above steps, we obtain an aligned synthetic RGB-T training dataset for RGB-T tracking. As a result, our proposed fusion architectures (see section 4.4.1) can also benefit from

Fusion level	Feature extractor	IoU-Net	Model predictor	Response map	EAO (↓)	A (↓)	R (↓)
Single modality	RGB	RGB	RGB	RGB	0.327	0.586	0.345
	TIR	TIR	TIR	TIR	0.331	0.584	0.332
	TIR	TIR (ft)	TIR (ft)	TIR	0.336	0.587	0.331
	TIR (ft)	TIR	TIR (ft)	TIR	0.339	0.589	0.329
	TIR (ft)	TIR (ft)	TIR (ft)	TIR	0.341	0.590	0.328
Pixel-level	RGB+TIR (ft)	RGBT (ft)	RGBT (ft)	RGBT	0.345	0.552	0.281
Response-level	RGB/TIR (ft)	RGB (ft)	RGB/TIR (ft)	RGB+TIR	0.342	0.546	0.309
	RGB/TIR (ft)	TIR (ft)	RGB/TIR (ft)	RGB+TIR	0.349	0.554	0.291
Feature-level	RGB/TIR (ft)	RGB (ft)	RGB+TIR (ft)	RGBT	0.346	0.545	0.266
	RGB/TIR (ft)	TIR (ft)	RGB+TIR (ft)	RGBT	0.359	0.564	0.243
	RGB/TIR (ft)	RGB+TIR (ft)	RGB (ft)	RGB	0.354	0.563	0.276
	RGB/TIR (ft)	RGB+TIR (ft)	TIR (ft)	TIR	0.366	0.601	0.261
	RGB/TIR (ft)	RGB+TIR (ft)	RGB+TIR (ft)	RGBT	0.389	0.605	0.224
	RGB/TIR (ft,×10)	RGB+TIR (ft)	RGB+TIR (ft)	RGBT	0.391	0.615	0.228

Table 4.1 – **Fusion mechanisms analysis on VOT-RGBT2019 (VOT challenge, 2019)**. We evaluate several fusion mechanisms at different levels of DiMP (Bhat et al., 2019). The results are reported in terms of EAO, normalized weighted mean of accuracy (A), and normalized weighted mean of robustness score (R). We explicitly show the input modality for each component of the tracker. Here, ‘RGB’ and ‘TIR’ are the single modality, ‘RGB/TIR’ means each modality input separately, ‘RGB+TIR’ means that both modalities are input simultaneously, and ‘RGBT’ indicates fused features from both modalities used in the remaining of network. Finally, (‘ft’) means fine-tuning and (‘×10’) means a higher learning rate for fine-tuning. The best results are highlighted in bold font.

end-to-end training. We hope that this allows us to see a similar performance gain as has been observed for RGB tracking.

Specifically as input data for multi-modal tracking, the elements in each set of the training data (M_{train}, M_{test}) are $M = \{(I_j^V, I_j^T, b_j)\}_{j=1}^{N_{frames}}$. Here, I_j^T represents the j -th synthetic TIR image generated from the aligned RGB image I_j^V . b_j is their identical bounding box.

4.5 Experiments

In this section, we provide a comprehensive evaluation of the proposed tracker mfDiMP on two benchmarks, VOT-RGBT2019 (VOT challenge, 2019) and RGBT210 (Li et al., 2017b), and describe all implementation and evaluation details.

4.5.1 Generating the Training RGB-T Dataset

We use the recent Generic Object Tracking Benchmark (GOT-10k) (Huang et al., 2018) to train our fused modality networks. GOT-10k has over 10,000 video segments, covering 563 classes of real-world moving objects and more than 80 motion patterns, amounting to a total of over 1.5 million manually labeled bounding boxes. It also provides additional supervision in terms of attribute labels such as ratio of object visible or type of motion. We employ GOT-10k’s training set, which contains 9,335 videos (1,403,359 frames), with 480 object classes and 69 motion classes. We refrain from using the set of 1000 prohibited videos listed in the VOT challenge website (VOT challenge, 2019), and so we train our model with the remaining 8,335 videos (1,251,981 frames).

With this reduced version of GOT-10k RGB dataset, we generate a large-scale RGB-T dataset by synthesizing paired TIR images using an image-to-image translation approach, as the work in Chapter 3. Specifically, we select pix2pix (Isola et al., 2017) for image-to-image translation given its superior performance. To train the pix2pix model, we use a total of 87K pairs of aligned images in the RGB and TIR modalities, depicting several different scenarios. These images are carefully collected and arranged from many current existing RGB-TIR datasets as depicted in Chapter 3. We train the pix2pix model using the default settings described in Chapter 3. After training, we use it to transfer the selected RGB videos in GOT-10k (Huang et al., 2018) to synthetic TIR videos, along with the labels.

4.5.2 Evaluation Datasets and Protocols

VOT-RGBTIR2019 dataset (VOT challenge, 2019) contains 60 public testing sequences, with a total of 20,083 frames. It is used as the most recent edition of the VOT challenge. We follow the VOT protocol, which establishes that when the evaluated tracker fails, i.e. when the overlap with the ground-truth is below a given threshold, it is re-initialized in the correct location five frames after the failure. The main evaluation measure used to rank the trackers is Expected Average Overlap (EAO), which is a combination of accuracy (A) and robustness (R). We compute all results using the provided toolkit (VOT challenge, 2019).

RGBT210 dataset (Li et al., 2017b) contains 210 highly-aligned public RGB and TIR video pairs for testing, with 210K frames in total and a maximum of 8K frames per sequence pair. There are a total of 12 representative attributes, such as camera moving, large scale variations and environmental challenges, which are annotated for each video. These facilitate attribute-sensitive evaluation analyses. We compare our results with other trackers using the provided toolkit (RGB-T dataset, 2018). We use precision plot and success plot to evaluate the trackers.

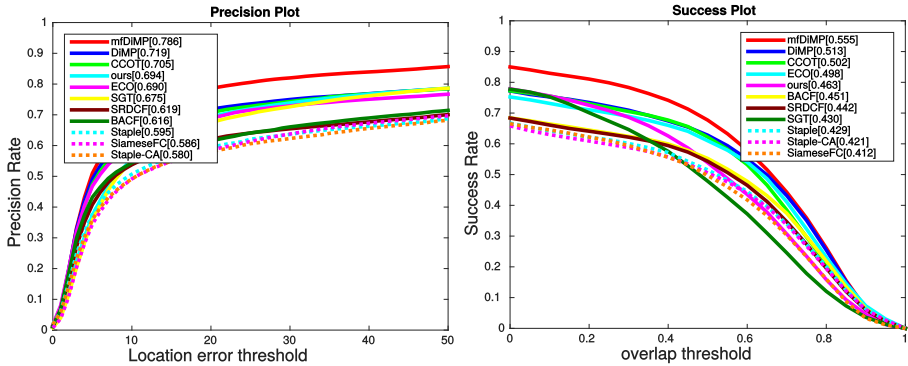


Figure 4.3 – **Precision plot and success plot by comparing our mfDiMP with the top-10 trackers on RGBT210 dataset (Li et al., 2017b)** We can see our mfDiMP outperforms DiMP with an absolute gain of 6.7% and 4.2% in terms of precision rate and success rate respectively.

4.5.3 Implementation Details

We use DiMP (Bhat et al., 2019) as our base tracker with ResNet-50 (He et al., 2016) as backbone network. The base architecture of DiMP is pre-trained on several large-scale RGB training datasets (Fan et al., 2018; Huang et al., 2018; Lin et al., 2014; Muller et al., 2018). To test the single modality versions, we simply input the images from either modality as in traditional RGB trackers (Bertinetto et al., 2016b; Danelljan et al., 2017; Henriques et al., 2015). RGB images have 3 image channels while TIR images have 1 channel, and so the pixel-level fusion uses 4-channel images. For the feature-level fusion, we concatenate the convolutional features after the feature extractor. Finally, for the response-level fusion we add together the final confidence maps independently predicted by the RGB and TIR modalities.

We use separate, modality-specific feature extractors for the response-level fusion and feature-level fusion. As hyperparameters for fine-tuning our architecture, we use the default values used to train each component in DiMP (Bhat et al., 2019), which have been carefully set and described by the authors in section 3.2 of the work (Bhat et al., 2019). We keep the default learning rates as for each component in the DiMP model and then decrease them by collaboratively multiplying a small gain learning rate, i.e. 0.001, and for two feature extractors with the learning rate of 0.01 and 0.001 separately for TIR modality and RGB modality. We do this here as considering that the feature extractor for RGB modality was pre-trained with large-scale RGB datasets already, and less RGB features need to be learned. On the

other hand, the feature extractor of TIR modality needs to catch up with that of RGB modality in terms of learning, hence we set a higher learning rate for fine-tuning the feature extractor of the TIR modality.

As a result of the stochastic nature of DiMP, the tracker generates different results for every run. Following the procedure employed in the work (Bhat et al., 2019), we compute default 15 runs of our mfDiMP tracker for vot-toolkit and 5 for RGBT210 dataset. Then obtain the stable result finally by averaging the various multi-run results.

4.5.4 Analysis of Fusion Mechanisms

In this section, we present an analysis to see where is the best to fuse the modalities in DiMP as shown in Table 4.1. This table is an extensive evaluation of all considered fusions under different configurations. We start with a comprehensive evaluation of the baseline tracker DiMP (Bhat et al., 2019) for single modality. We present several configurations in the upper part of Table 4.1. First, the ‘Single modality’ with the RGB modality or TIR modality in the first two rows, is the original DiMP (pre-trained for RGB) using either RGB or TIR images during online tracking. It shows how using TIR images provides a higher result. For the next three rows, we fine-tune the feature extractor and/or IoU-Net with synthetic TIR images. It shows fine-tuning the single modality network improves the pre-trained networks with an absolute gain of 1%. In the lower part of Table 4.1, we analyzed the effectiveness of each fusion mechanism for DiMP (Bhat et al., 2019), which we discuss in detail in the remainder of this section.

Pixel-level fusion. First, we evaluate the pixel-level fusion. From Table 4.1, we can observe that pixel-level fusion improves the performance of baseline tracker from 0.331 to 0.345 with an absolute gain of 1.4%. The images from the RGB modality and TIR modality have complementary information. Therefore, using the fused images to train the network end-to-end, can help the model to learn better deep feature representations, which in turn improves tracking performance.

Response-level fusion. The fusion now takes place in the final response map, which belongs to the classifier. The signals from both modalities pass through the classifier separately and both compute the response map. Then we sum together the two response maps, obtaining the final fused response map. Meanwhile for the IoU-Net, there is only a single modal input. Here we use two strategies, one uses the features from RGB modality to feed into IoU-Net and the other uses TIR features instead. Both fusion mechanisms enhance the tracking performance, and using the TIR modality for IoU-Net outperforms using RGB, achieving scores of 0.349 and 0.342, respectively. The results show that fusion on response-level obtains the same

	ECO (Danelljan et al., 2017)	SiamFC (Bertinetto et al., 2016b)	DaSiamRPN (Zhu et al., 2018)	ATOM (Danelljan et al., 2019)	DiMP (Bhat et al., 2019)	mfDiMP
EAO(!)	0.265	0.254	0.324	0.318	0.327	0.391
A (!)	0.580	0.594	0.604	0.575	0.586	0.615
R (!)	0.480	0.533	0.482	0.374	0.345	0.228

Table 4.2 – **State-of-the-art comparison on VOT-RGBT2019 dataset.** Our mfDiMP improves the baseline tracker DiMP with an absolute gain of 6.4% in terms of EAO. The best results are highlighted in bold font.

	ECO (Danelljan et al., 2017)	CSR (Lukezic et al., 2017)	CMRT (Li et al., 2018d)	SGT (Li et al., 2017b)	CNN+KCF+ RGBT (Henriques et al., 2015)	CFnet+ RGBT (Valmadre et al., 2017)	mfDiMP
No Occlusion	87.7/64.3	68.1/45.2	86.1/59.4	82.4/50.7	63.7/42.9	69.7/52.2	88.9/67.3
Partial Occlusion	72.2/52.5	52.7/36.6	77.1/52.2	75.4/48.3	56.0/36.4	57.2/38.4	84.0/60.1
Heavy Occlusion	58.3/41.3	37.1/24.3	54.6/34.8	53.1/34.1	36.6/25.9	39.3/27.3	68.4/45.8
Low Illumination	66.6/45.6	47.3/31.1	71.4/46.4	71.6/44.7	52.8/34.5	49.8/33.6	77.1/53.7
Low Resolution	64.1/38.1	46.0/23.1	64.7/37.4	65.8/37.5	54.6/32.5	45.2/27.7	69.2/43.6
Thermal Crossover	82.1/58.8	43.2/29.3	65.8/43.0	64.9/40.7	49.6/33.2	42.8/29.4	76.5/55.2
Deformation	61.2/45.0	44.7/33.0	65.2/45.8	65.3/45.9	44.8/34.4	48.9/35.2	77.7/56.6
Fast Motion	58.2/39.2	42.6/25.0	58.8/34.9	58.0/33.1	37.1/24.1	36.5/23.0	76.7/52.6
Scale Variation	74.5/55.4	53.3/37.5	72.5/49.2	67.4/41.7	50.3/32.6	56.7/40.6	82.2/59.5
Motion Blur	67.8/49.9	34.7/23.8	58.4/40.5	58.6/39.6	30.4/22.0	30.3/22.4	72.5/51.2
Camera Moving	61.7/45.0	38.9/27.4	60.0/41.9	59.0/40.7	36.2/27.0	37.2/27.9	75.3/53.8
Background Clutter	52.9/35.2	38.4/23.7	58.3/35.6	58.6/35.5	42.3/28.4	43.7/28.1	71.5/45.7
ALL	69.0/49.8	49.1/33.0	69.4/46.3	67.5/43.0	49.3/33.1	51.8/36.0	78.6/55.5

Table 4.3 – **Attribute-based Precision Rate and Success Rate (PR/SR %) on RGBT210 dataset with several trackers.** These trackers include popular RGB trackers such as ECO and CSR, recent multi-modal fusion tracker like CMRT and SGT, and also extended RGB-T trackers from KCF and CFnet. Our tracker surpasses almost all the trackers over all the attributes.

effectiveness as pixel-level fusion. For both of them, the fusion takes place in the extra part of the network, which is easier to implement and only fewer variants are evaluated. Next we move to the feature-level fusion and consider several fusion variants in detail.

Feature-level fusion. We consider inputting fused feature representations into two different DiMP components, i.e. model predictor and IoU-Net. In the former case, only the model predictor receives fused features, whereas the remaining component (IoU-Net) still uses features from a single modality. In both cases, we improve over single modality tracking, and the version with TIR for IoU-Net obtains a better result with 0.359, an absolute gain of 2.8% with respect to the TIR fine-tuned best model in single modality. In the latter case, we use fused features for IoU-Net. We obtain the highest score, 0.366, when using TIR features for the other component (the model predictor). It demonstrates that using fused features for IoU-Net outperforms using fused features for the model predictor. We attribute it to that IoU-net is a more

complex network than the model predictor, where the fused feature representations can be more effective to prompt IoU-Net to its ultimate capacity. Finally, we use fused features to feed both the model predictor and IoU-Net, which significantly improves the result to 0.389 with a substantial absolute gain of 5.8%. Considering that the feature extractor is pre-trained on RGB images, it is natural to assume that the feature extractor for TIR images needs a stronger training signal. For this reason, we propose a variant in which the feature extractor for the TIR modality has a higher learning rate ($\times 10$). This variant achieves 0.391, which is the best result and significantly outperforms the best single modality tracker with a big jump.

From Table 4.1, we can see that fusion on feature-level with end-to-end training, provides distinguished improvements for the performance of tracker. Specifically, fusion of the feature representations for both of model predictor and IoU-Net achieves the best result on VOT-RGBT2019 dataset (VOT challenge, 2019). In the following sections, we select this best performing variant as our final tracker, which we call *mfDiMP*.

4.5.5 VOT-RGBT2019 Dataset

In this section, we evaluate our *mfDiMP* on the VOT-RGBT2019 dataset in terms of EAO in Table 4.2. We compare with several high-quality RGB trackers including ECO (Danelljan et al., 2017), ATOM (Danelljan et al., 2019), DiMP (Bhat et al., 2019), SiameseFC (Bertinetto et al., 2016b), DaSiamRPN (Zhu et al., 2018), which also input with RGB modality as their natural designs. The single modality baseline tracker DiMP (Bhat et al., 2019), which shows dominant performances on various RGB datasets (Fan et al., 2018; Huang et al., 2018; Kristan et al., 2018; Muller et al., 2018), also achieves excellent results on VOT-RGBT2019. By using our multi-modal fusion with end-to-end training, we improve DiMP by an absolute gain of 6.4% in terms of EAO. This significant improvement demonstrates that our selected fusion mechanism is effective for maximally exploiting the multi-modal nature of the given images.

4.5.6 RGBT210 Dataset

We evaluate *mfDiMP* on the recent RGBT210 dataset (Li et al., 2017b) using their two evaluation metrics (Figure 4.3). We compare against the top-10 trackers on this dataset, including CCOT (Danelljan et al., 2016b), ECO (Danelljan et al., 2017), CMRT (Li et al., 2018d), BACF (Kiani Galoogahi et al., 2017), SRDCF (Danelljan et al., 2015b), SGT (Li et al., 2017b), Staple (Bertinetto et al., 2016a), Staple-CA (Mueller et al., 2017b), SiameseFC (Bertinetto et al., 2016b). We can see how our tracker significantly outperforms the second best tracker (DiMP) with an absolute gain

of 6.7% and 4.2%, in terms of precision rate and success rate respectively. As a result, mfDiMP achieves a new state-of-the-art also on this dataset, bringing further evidence to the advantages of end-to-end training for multi-modal tracking in terms of accurate object localization.

4.5.7 Attribute Analysis on RGBT210 Dataset

There are totally 12 kind of different attributes as in the RGBT210 datasets (Li et al., 2017b). We analyze the performance of our method on these attributes in terms of precision rate and success rate (PR/SR %) in Table. 4.3. We compare with some popular RGB trackers such as ECO (Danelljan et al., 2017), CSR (Lukezic et al., 2017). We also compare with the state-of-the-art RGB-T trackers on this dataset, e.g. CMRT (Li et al., 2018d) and SGT (Li et al., 2017b), and some extended RGB-T trackers from KCF, CFnet as in the work (Li et al., 2018d). The further comparison on specific scenarios, proves the robustness and generality of our mfDiMP on RGB-T tracking. Our tracker outperforms most trackers on all the attributes. Specifically in the attributes such as partial occlusion, low illumination, deformation, fast motion, camera moving, and background clutter, mfDiMP achieves a significant gain of about 10% in terms of Success Rate (SR), compared with the second best.

4.6 Conclusions

Most of the multi-modal trackers are still using hand-crafted features, or simple off-the-shelf features. In this Chapter we investigate how to effectively fuse multi-modal data in an end-to-end training manner, which could make optimal use of information from both modalities. We propose three end-to-end multi-modal fusion architectures, consisting of pixel-level fusion, feature-level fusion and response-level fusion. To ensure that the proposed fusion tracker can be trained in an end-to-end manner, we also generate large-scale paired synthetic RGB-T datasets. We perform extensive experiments on two recent benchmarks for RGB-T tracking: VOT-RGBT2019 (VOT challenge, 2019) and RGBT210 (Li et al., 2017b). The results show that the proposed fusion tracker does significantly improve the performance of the baseline tracker with respect to single modality. As a consequence, our end-to-end multi-modal fusion tracker sets new state-of-the-art results on both datasets.

5 Learning the Model Update for Siamese Trackers *

5.1 Introduction

Generic visual object tracking is the task of predicting the location of a target object in every frame of a video, given its initial location. Tracking is one of the fundamental problems in computer vision, spanning a wide range of applications including video understanding (Renoust et al., 2016), surveillance (Emami et al., 2012), and robotics (Liu et al., 2012). It is a highly challenging task due to frequent appearance changes, various types of occlusions, the presence of distractor objects, and environmental aspects such as motion blur or illumination changes.

Currently, there are two prevalent tracking paradigms: Siamese tracking methods (Bertinetto et al., 2016b; Li et al., 2018b; Wang et al., 2018; Zhu et al., 2018) and tracking-by-detection methods (Bolme et al., 2010; Danelljan et al., 2017, 2016b; Henriques et al., 2015; Nam & Han, 2016; Zhang et al., 2014, 2019b). In this work, we consider Siamese trackers, since they provide competitive accuracy while achieving impressive computational efficiency. The basic principle of these trackers is to match an object appearance template with a corresponding feature representation of the search region in the test frame. The features for the object template and the search region are acquired through a deep neural network trained offline on a large dataset. Such a training strategy has shown to provide excellent visual descriptors for the tracking task (Bertinetto et al., 2016b; Zhu et al., 2018).

In the original Siamese tracker (Bertinetto et al., 2016b), the object template is initialized in the first frame and then kept fixed during the remainder of the video. However, appearance changes are often large and failing to update the template could lead to early failure of the tracker. In such scenarios, it is important to adapt the model to the current target appearance. To accommodate this problem, more recent Siamese trackers (Li et al., 2018b; Wang et al., 2018; Zhu et al., 2018) have implemented a simple linear update strategy using a running average with a fixed learning rate (Stauffer & Grimson, 1999). This strategy assume a constant rate

*This chapter is based on a publication in Proceedings of the IEEE International Conference on Computer Vision 2019



Figure 5.1 – **Qualitative comparison between model updates.** We learn to update the model template using *UpdateNet*. When combined with Siamese trackers such as SiamFC (Bertinetto et al., 2016b), our learned updating strategy can be effectively adapted to current circumstances, unlike the simple linear update commonly used.

of appearance change across all frames in the video, as well as across different videos. In practice, the update requirements for the object template greatly vary for different tracking situations, which depend on a complex combination of external factors such as motion, blur, or background clutter. Therefore, a simple linear update is often inadequate to cope with changing updating needs and to generalize to all potentially encountered situations. Moreover, this update is also constant in all spatial dimensions, which does not allow for localized partial updates. This is especially damaging in situations such as partial occlusions, where only a certain part of the template needs to be updated. Finally, excessive reliance on the initial template may suffer from catastrophic drift and the inability to recover from tracking failures.

In this Chapter, we propose to *learn* the target template update itself. Our learned update strategy utilizes target and image information, and is thus adaptive to the present circumstances of each particular situation. In our approach, the updated template is computed as a function of (i) the initial ground-truth template, (ii) the accumulated template from all previous frames, and (iii) the feature template at the predicted object location in the current frame. Hence, the new accumulated

template contains an effective historical summary of the object’s current appearance, as it is continually updated using the most recent information while being robust due to the strong signal given by the initial object appearance. More specifically, the aforementioned template update function is implemented as a convolutional neural network, *UpdateNet*. This is a compact model that can be combined with any Siamese tracker to enhance its online updating capabilities while maintaining its efficiency properties. Furthermore, it is sufficiently complex to learn the nuances of effective template updating and be adaptive enough to handle a large collection of tracking situations.

We evaluate UpdateNet by combining it with two state-of-the-art Siamese trackers: SiamFC (Bertinetto et al., 2016b) and DaSiamRPN (Zhu et al., 2018). Through extensive experiments on common tracking benchmarks, such as VOT2018 (Kristan et al., 2018), we demonstrate how our UpdateNet provides enhanced updating capabilities that in turn result in improved tracking performance (see Figure 5.1). We also present results in the recent LaSOT dataset (Fan et al., 2018), which is substantially more challenging as it contains abundant long-term sequences. Overall, we propose an efficient model to learn how to effectively update the object template during online tracking and that can be applied to different existing Siamese trackers.

5.2 Related Work

Tracking Frameworks. Most existing tracking methods are either based on tracking-by-detection or employ template matching. Object trackers based on tracking-by-detection pose the task of target localization as a classification problem where the decision boundary is obtained by online learning a discriminative classifier using image patches from the target object and the background. Among the tracking-by-detection approaches, discriminative correlation filter based trackers (Danelljan et al., 2017, 2016b; Henriques et al., 2015; Zhang et al., 2019b) have recently shown excellent performance on several tracking benchmarks (Kristan et al., 2018, 2016b; Wu et al., 2013, 2015). These trackers learn a correlation filter from example patches of the target appearance to discriminate between the target and background appearance.

The other main tracking framework is based on template matching, typically using Siamese networks (Bertinetto et al., 2016b; Held et al., 2016; Li et al., 2018b; Valmadre et al., 2017; Wang et al., 2018; Zhu et al., 2018), that implement a similarity network by spatial cross-correlation. Bertinetto et al. (2016b) proposed a Siamese tracker which is based on a two-stream architecture. One stream extracts the object template’s features based on an *exemplar* image that contains the object to be tracked. The other stream receives as input a large *search region* in the target image.

The two outputs are cross-correlated to generate a response map of the search region. Many trackers have extended the SiamFC architecture (He et al., 2018; Li et al., 2018b; Valmadre et al., 2017; Wang et al., 2018; Zhang et al., 2018; Zhu et al., 2018) for tracking. The Siamese-based trackers have gained popularity since they provide a good trade-off between computational speed and tracking performance. However, most of these approaches struggle to robustly classify the target especially in the presence of distractors due to no online learning. In this work, we analyze the limitations of Siamese trackers regarding the update of the template model and propose a solution to address them.

Updating the object template. Most trackers either use simple linear interpolation to update the template in every frame (Bolme et al., 2010; Choi et al., 2018; Danelljan et al., 2015b, 2016b; Henriques et al., 2015; Kiani Galoogahi et al., 2017) or do not update the initial template at all (Bertinetto et al., 2016b; Li et al., 2018b; Wang et al., 2018; Zhu et al., 2018). Such update mechanisms are insufficient in most tracking situations, as the target object may suffer appearance changes given by deformations, fast motion, or occlusion. Moreover, fixed update schedules also result in object templates that are more focused on recent frames (Danelljan et al., 2016a), while forgetting the historical appearances of the object. To address this issue, Danelljan et al. (2016a,b) propose to include a subset of historic frames as training samples when calculating the current correlation filter, which leads to better results than the traditional linear frame-by-frame update. Nonetheless, storing multiple samples in memory results in increased computation and memory usage, which in turn heavily decreases the tracking speed. The ECO tracker (Danelljan et al., 2017) tries to alleviate this problem by modelling the distribution of training samples as a mixture of Gaussians, where each component represents a distinct appearance. This significantly reduces required storage and, combined with a conservative update strategy (only every five frames), leads to increased tracking efficiency. Even with more previous samples, the correlation filter is still updated by averaging the filters of their corresponding samples (as still linear interpolation update).

Recently, Yang & Chan (2018) employed a Long Short-Term Memory (LSTM) to estimate the current template by storing previous templates in memory during on-line tracking, which is computationally expensive and a rather complex system. Choi et al. (2017) also uses a template memory but uses reinforcement learning to select one of the stored templates. This method fails to accumulate information from multiple frames. The meta-tracker (Park & Berg, 2018) extends the initialization of the target model in the first frame by a pre-trained approach, but still need a linear update in on-line tracking. Yao et al. (2018) propose to learn the updating coefficients for CF trackers using SGD offline. While the solution for

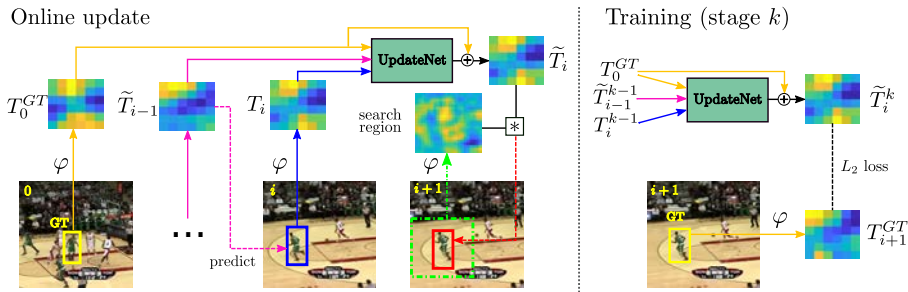


Figure 5.2 – **Overview of our tracking framework with UpdateNet.** (Left) The online update of the object template is performed by UpdateNet, which receives as input the initial ground-truth template, last accumulated template and current predicted template, and outputs updated accumulated template. (Right) Training of UpdateNet using the distance to the ground-truth object template on next frame.

correlation filter is still the hand-crafted manner, and these coefficients are fixed without updating during tracking.

To adapt to the object variations, Guo et al. (2017) propose to compute a transformation matrix with respect to the initial template through regularized linear regression in the Fourier domain. Since only the initial template is considered when estimating the transformation, this approach ignores the historical object variations observed during tracking, which may result important for a smoother adaptation of the exemplar template. Moreover, they compute the transformation matrix as a closed-form solution on the Fourier domain, which suffers from issues related to the boundary effect (Kiani Galoogahi et al., 2015). Our work instead uses a powerful yet easily trainable model to update the object template based not only the first frame, but also on the accumulated template using all previous frames, leveraging the observed training data. Furthermore, our UpdateNet is trained to *learn* how to perform an effectively update the object template, based on the observed training tracking data.

5.3 Updating the Object Template

In this section, we present our approach to learn how to update the object template during online tracking. We start by revisiting the standard update mechanism in tracking and identifying its drawbacks. Then, we introduce our formulation to overcome them and describe our model and training procedure in detail. The focus of this chapter is on Siamese trackers. Note, however, that our approach is not

limited to Siamese trackers and the same formulation could be applied to other types of trackers, e.g. DCF (Danelljan et al., 2017, 2016b; Henriques et al., 2015).

5.3.1 Standard Update

Several recent tracking approaches (Bolme et al., 2010, 2009; Henriques et al., 2015; Li et al., 2018b; Valmadre et al., 2017; Wang et al., 2018; Zhu et al., 2018) use a simple averaging strategy to update the object appearance model given a new data sample. This strategy dates from early tracking methods (Stauffer & Grimson, 1999) and has long been the standard for online updating given its acceptable results and in spite of its limitations. The template is updated as a running average with exponentially decaying weights over time. The choice of exponential weights yields the following recursive formula for updating the template,

$$\tilde{T}_i = (1 - \gamma)\tilde{T}_{i-1} + \gamma T_i. \quad (5.1)$$

Here, i is the frame index, T_i is the new template sample computed using only the current frame, and \tilde{T}_i is the accumulated template. The update rate γ is normally set to a fixed small value (e.g. $\gamma = 0.01$) following the assumption that the object's appearance changes smoothly and consistently in consecutive frames. In DCF trackers (Bolme et al., 2010; Henriques et al., 2015), T corresponds to the correlation filter. In Siamese trackers, instead, T is the object appearance template extracted from a particular frame by a fully convolutional feature extractor. While the original SiamFC tracker (Bertinetto et al., 2016b) does not perform any model update, more recent Siamese trackers (Bertinetto et al., 2016b; Li et al., 2018b; Wang et al., 2018; Zhu et al., 2018) have adopted (5.1) to update their templates.

While template averaging provides a simple means of integrating new information, it has several severe drawbacks:

- It applies a constant update rate for every video, despite the possibly different updating necessities caused by multiple factors such as camera motion. Even within the same video, the required update on the object template may dynamically vary at different times.
- The update is also constant along all spatial dimensions of the template, including the channel dimension. This prevents updating only part of the template, which is desirable under partial occlusions, for example.
- The tracker cannot recover from drift. Partially, this is caused by the fact that it loses access to the appearance template T_0 , which is the only template which is without doubt on the object.
- The update function is constrained to a very simple linear combination of previous appearance templates. This severely limits the flexibility of the update mechanism, important when the target undergoes complex appearance

changed. Considering more complex combination functions is expected to improve results.

5.3.2 Learning to Update

We address the drawbacks listed above by proposing a model that learns an adaptive update strategy. Since the focus of this chapter is on Siamese trackers, T represents here the object appearance template. To address the limitations of simple template averaging, we propose to learn a generic function ϕ that updates the template according to,

$$\tilde{T}_i = \phi(T_0^{GT}, \tilde{T}_{i-1}, T_i). \quad (5.2)$$

The learned function ϕ computes the updated template based on initial ground-truth template T_0^{GT} , the last accumulated template \tilde{T}_{i-1} and the template T_i extracted from the predicted target location in the current frame. In essence, the function updates the previous accumulated template \tilde{T}_{i-1} by integrating the new information given by the current frame T_i . Therefore, ϕ can be adapted to the specific updating requirements of the current frame, based on the difference between the current and accumulated templates. Moreover, it also considers the initial template T_0^{GT} in every frame, which provides highly reliable information and increases robustness against model drift. The function ϕ is implemented as a convolutional neural network, which grants great expressive power and the ability to learn from large amounts of data. We call this neural network *UpdateNet* and describe it in detail in the following section.

5.3.3 Tracking Framework With UpdateNet

We present here the structure of UpdateNet and describe how it is applied for online tracking. Figure 5.2 (left) presents an overview of our adaptive object update strategy using UpdateNet with a Siamese tracker. We extract deep features from image regions with a fixed fully convolutional network φ , employing the same feature extractor as in the SiamFC tracker (Bertinetto et al., 2016b). We extract T_0^{GT} from the ground-truth object location in the initial frame (number 0 in Figure 5.2). In order to obtain T_i for the current frame, we use the accumulated template from all previous frames \tilde{T}_{i-1} to predict the object location in frame i (dashed purple line) and extract the features from this region (solid blue line). Note that \tilde{T}_{i-1} corresponds to the output of UpdateNet for the previous time step, not shown here for conciseness. We concatenate the extracted features T_0^{GT} and T_i with the accumulated features \tilde{T}_{i-1} to form the input of UpdateNet. This input is then

processed through a series of convolutional layers (sec. 5.4.3) and outputs the predicted new accumulated template \tilde{T}_i . For the first frame, we set T_i and \tilde{T}_{i-1} to T_0^{GT} as there have not been any previous frames.

The only ground-truth information used by UpdateNet is the given object location in the initial frame, all other inputs are based on predicted locations. Hence, T_0^{GT} is the most reliable signal on which UpdateNet can depend to guide the update. For this reason, we employ a residual learning strategy (He et al., 2016), where UpdateNet learns how to modify the ground-truth template T_0^{GT} for the current frame. This is implemented by adding a skip connection from T_0^{GT} to the output of UpdateNet. This approach still takes into account the set of historical appearances of the object for updating, but pivots such update on the most accurate sample. We have also experimented with adding skip connections from other inputs as well as no residual learning at all (see sec. 5.4).

5.3.4 Training UpdateNet

We train our UpdateNet to predict the target template in the coming frame, i.e. the predicted template \tilde{T}_i should match the template T_{i+1}^{GT} extracted from the ground-truth location in the next frame (Figure 5.2, right). The intuition behind this choice is that T_{i+1}^{GT} is the optimal template to use when searching for the target in the next frame. In order to achieve this, we train UpdateNet by minimizing the Euclidean distance between the updated template and the ground-truth template of the next frame, defined as

$$\mathcal{L}_2 = \|\phi(T_0^{GT}, \tilde{T}_{i-1}, T_i) - T_{i+1}^{GT}\|_2. \quad (5.3)$$

In the remainder of this section we describe the procedure employed to generate the training data and introduce a multi-stage training approach for UpdateNet.

Training samples. In order to train UpdateNet to minimize (5.3), we need pairs of input triplets $(T_0^{GT}, \tilde{T}_{i-1}, T_i)$ and outputs T_{i+1}^{GT} that reflect the updating needs of the tracker during online application. The object templates of the initial frame T_0^{GT} and target frame T_{i+1}^{GT} can be easily obtained by extracting features from ground-truth locations in corresponding frames. In case of the current frame's template T_i , however, using the ground-truth location represents a seldom encountered situation in practice, for which the predicted location in the current frame is very accurate. This unrealistic assumption biases the update towards expecting very little change with respect to T_i , and thus UpdateNet cannot learn a useful updating function. Therefore, we need to extract T_i samples for training by using an imperfect localization in the i -th frame. We can simulate such situation by using the accumulated template \tilde{T}_{i-1} , ideally presenting localization errors that occur during online tracking.

Multi-stage training. In theory, we could use the accumulated template \tilde{T}_{i-1} output by UpdateNet. However, this would force the training to be recurrent, making the procedure cumbersome and inefficient. To avoid this, we split our training procedure into sequential stages that iteratively refine UpdateNet. In the first stage, we run the original tracker on the training dataset using the standard linear update

$$\tilde{T}_i^0 = (1 - \gamma) \tilde{T}_{i-1}^0 + \gamma T_i^0, \quad (5.4)$$

which generates accumulated templates and realistically predicted locations for each frame. We set the update rate γ to the recommended value for the tracker. This corresponds to a first approximation to likely inputs for UpdateNet during tracking inference, albeit with the less sophisticated linear update strategy. In every posterior training stage $k \in \{1, \dots, K\}$, we use the UpdateNet model trained in the previous stage to get accumulated templates and object location predictions as follows

$$\tilde{T}_i^k = \phi_i^k \left(T_0^{GT}, \tilde{T}_{i-1}^{k-1}, T_i^{k-1} \right). \quad (5.5)$$

Such training data samples closely resemble the expected data distribution at inference time, as they have been output by UpdateNet. We investigate a suitable value for the total number of stages K in the experimental section (sec. 5.4).

5.4 Experiments

5.4.1 Training Dataset

We use the recent Large-scale Single Object Tracking (LaSOT) (Fan et al., 2018) to train our UpdateNet. LaSOT has 1,400 sequences in 70 categories, which amounts to a total of 3.52M frames. Each category contains exactly twenty sequences, making the dataset balanced across classes. It also provides longer sequences that contain more than 1,000 frames (2,512 frames on average) in order to satisfy the current long-term trend in tracking. We used the official training and test splits, which preserve the balanced class distribution. In fact, we only employ a subset containing 20 training sequences from 20 randomly selected categories, with a total of 45,578 frames. We have found experimentally that this suffices to learn an effective updating strategy, and that additional data brings only a small performance boost while increasing training time.

5.4.2 Evaluation Datasets and Protocols

We evaluate results on standard tracking benchmarks: VOT2018/16 (Kristan et al., 2016a), LaSOT (Fan et al., 2018) and TrackingNet (Muller et al., 2018).

VOT2018/16 (Kristan et al., 2016a). VOT2018 dataset has 60 public testing sequences, with a total of 21,356 frames. It is used as the most recent edition of the VOT challenge. The VOT protocol establishes that when the evaluated tracker fails, i.e. when the overlap with the ground-truth is below a given threshold, it is re-initialized in the correct location five frames after the failure. The main evaluation measure used to rank the trackers is Expected Average Overlap (EAO), which is a combination of accuracy (A) and robustness (R). We also use VOT2016 (Kristan et al., 2016b) for comparison purposes, which has 10 different sequences with VOT2018 (Kristan et al., 2018). We compute all results using the provided toolkit (Kristan et al., 2018).

LaSOT (Fan et al., 2018). LaSOT is a much larger and more challenging dataset including long-term sequences. Following recent works that use this dataset (Fan & Ling, 2018; Li et al., 2018a), we report results on protocol II, i.e. LaSOT testing set. The testing subset has 280 sequences with 690K frames in total. LaSOT dataset (Fan et al., 2018) follows the OPE criterion of OTB (Wu et al., 2013). It consists of precision plot which is measured by the center location error, and success plot which is measured through the intersection over union (IoU) between the predicted bounding box and the ground-truth. Besides precision plot and success plot, LaSOT also uses normalized precision plot to counter the situation that target size and image resolution have large discrepancies for different frames and videos, which heavily influences the precision metric. We use success plot and normalized precision plot to evaluate the trackers in the article. We use their code (Fan et al., 2018) to create all plots.

TrackingNet (Muller et al., 2018). This is a large-scale tracking dataset consisting of videos in the wild. It has a total of 30,643 videos split into 30,132 training videos and 511 testing videos, with an average of 470,9 frames. It uses precision, normalized precision and success as evaluation metrics.

5.4.3 Implementation Details

We use SiamFC (Bertinetto et al., 2016b) and DaSiamRPN (Zhu et al., 2018) as our base trackers, and the backbone Siamese network adopts the modified AlexNet. We do not perform any changes except for in the updating component. The original implementation of SiamFC did not perform any object update. We borrow a linear update rate from CFNet (Valmadre et al., 2017) as $\gamma = 0.0102$ for templates generating in the training stage 1. We use the original version of DaSiamRPN, which does not employ any update strategy. We analyze the effect of the linear update

Update for SiamFC	Skip	EAO (\uparrow)	A (\uparrow)	R (\downarrow)
Linear	-	0.188	0.50	0.59
UpdateNet ($K = 1$)	-	0.205	0.48	0.58
UpdateNet ($K = 1$)	T_i	0.207	0.47	0.57
UpdateNet ($K = 1$)	\tilde{T}_{i-1}	0.214	0.49	0.58
UpdateNet ($K = 1$)	T_0^{GT}	0.250	0.50	0.53
UpdateNet ($K = 2$)	T_0^{GT}	0.257	0.51	0.50
UpdateNet ($K = 3$)	T_0^{GT}	0.262	0.52	0.49

Table 5.1 – **Ablation study on VOT2018 (Kristan et al., 2018)**. We present several update strategies for SiamFC (Bertinetto et al., 2016b). The results are reported in terms of EAO, normalized weighted mean of accuracy (A), and normalized weighted mean of robustness score (R). ‘Skip’ column indicates the origin of the skip connection, if any. Here, K is the number of stages UpdateNet is trained for.

rate in the tracking performance in sec. 5.4.7. To train UpdateNet, we set a group of templates, including T_0^{GT} , \tilde{T}_{i-1} , T_i and T_{i+1}^{GT} as the input. They are all sampled sequentially from a same video. It is noteworthy that \tilde{T}_{i-1} and T_i are generated by the real tracking procedure, while T_0^{GT} and T_{i+1}^{GT} are ground-truth templates. We store all training object templates on disk, extracted using either linear/no update (stage $k = 1$) or a previous version of UpdateNet ($k > 1$). Let the template size be $H \times W \times C$. UpdateNet is a two-layer convolutional neural network: one $1 \times 1 \times 3 \cdot C \times 96$ convolutional layer, followed by a ReLU and a second convolutional layer of dimensions $1 \times 1 \times 96 \times C$. For SiamFC, $H = W = 6$ and $C = 256$, whereas DaSiamRPN $C = 512$. In the first stage, the weights are initialized from scratch and the learning rate is decreased logarithmically at each epoch from 10^{-6} to 10^{-7} . In next stage, the weights are initialized by the best model from the last stage, and the learning rate is decreased logarithmically at each epoch from 10^{-7} to 10^{-8} . We train the model for 50 epochs with mini-batches of size 64. We use Stochastic Gradient Descent (SGD) with momentum of 0.9 and weight decay of 0.0005.

5.4.4 Ablation Study

We start our evaluation by ablating our approach on different components in order to analyze their contribution to the final performance. Table 5.1 shows the results using VOT2018 (Kristan et al., 2018) dataset under the EAO measure. In the middle of

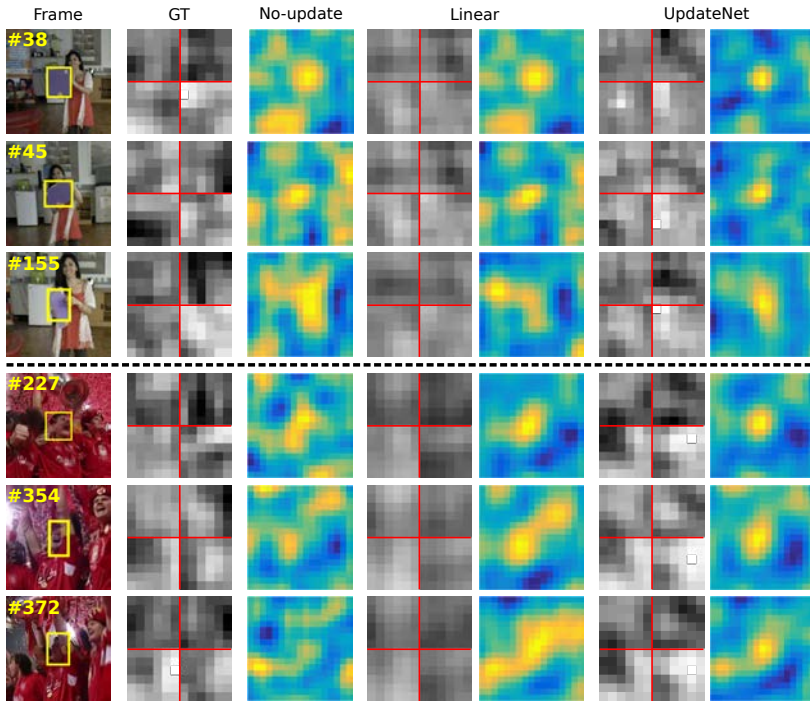


Figure 5.3 – **Visualization of accumulated templates for SiamFC.** ‘Frame’ column shows the search region and ground-truth box used to extract the templates, whose top four channels we show in ‘GT’. ‘No-update’ presents the response map resulting from applying the initial template to the search region. For ‘Linear’ and ‘UpdateNet’ strategies we also show their accumulated templates.

the table, it shows that updating the object template with the first stage of UpdateNet results beneficial when it is residually trained with respect to T_0^{GT} , as the learned update strategy is grounded on a reliable object sample. Moreover, our multi-stage training further increases the performance achieved by UpdateNet, reaching a total improvement of 7.4% with respect to the original SiamFC with no update. For the remainder of the chapter, we use UpdateNet trained on 3 stages and with skip connection from T_0^{GT} .

5.4.5 Analysis on Representation Update

This section attempts to provide insights regarding the performance improvements achieved by UpdateNet. Siamese networks are trained to project the images into a feature space in which spatial correlation is maximal. Update strategies operate on the learned features, possibly interfering with their correlation abilities and potentially damaging the tracking performance. In order to study the interference of the update strategies on the features we visualize the accumulated templates of SiamFC for both linear update and UpdateNet in Figure 5.3. We also include the ground-truth template extracted from the annotated bounding-box. For each template we show the feature maps of the four most dynamic channels in the ground-truth template, arranged in a 2×2 grid. For comparison reasons, the accumulated templates are generated using the ground-truth object locations instead of the predicted locations during tracking. Moreover, next to each accumulated template we also show the response map generated when correlating the template with the search region. We observe several interesting properties that support the performance gains seen in practice. First, the accumulated templates using UpdateNet resemble the ground-truth more closely than those in linear update (see e.g. highlight on bottom-right channel for frame 38, first example). Second, response maps tend to be sharper on the object location for UpdateNet, which shows how our strategy does not negatively interfere with the desired correlation properties of the learned features. Finally, the accumulated templates of the linear update change at a very slow rate and are clearly deficient in keeping up with the appearance variation exhibited in videos.

In order to further study this observation, we propose quantifying the change rate between templates of contiguous frames. For each $i \in \{1, \dots, N\}$ we compute the average difference in the template as $\delta_i = \frac{1}{|E|} \sum_E |T_i - T_{i-1}|$, where N is the number of frames in a video and the sum runs over each element of the feature maps (e.g. $E = 6 \times 6 \times 256$). We present the results in Figure 5.4. The bottom row contains the average change rate δ of all 60 videos in VOT2018 (Kristan et al., 2018). It is clear that the linear update strategy cannot deliver the updating rate required by the change in the features of the ground-truth template. UpdateNet, on the other hand, provides a much more adaptive strategy that is substantially closer in magnitude to the change rate of the ground-truth template. The top and middle rows also show the change rate of the two individual sequences in Figure 5.3, 'book' and 'soccer1'. We can see UpdateNet mimics the real template in high change periods, as indicated by the high correlation on their extremes. This leads to predicting better response map as shown in Figure 5.3.

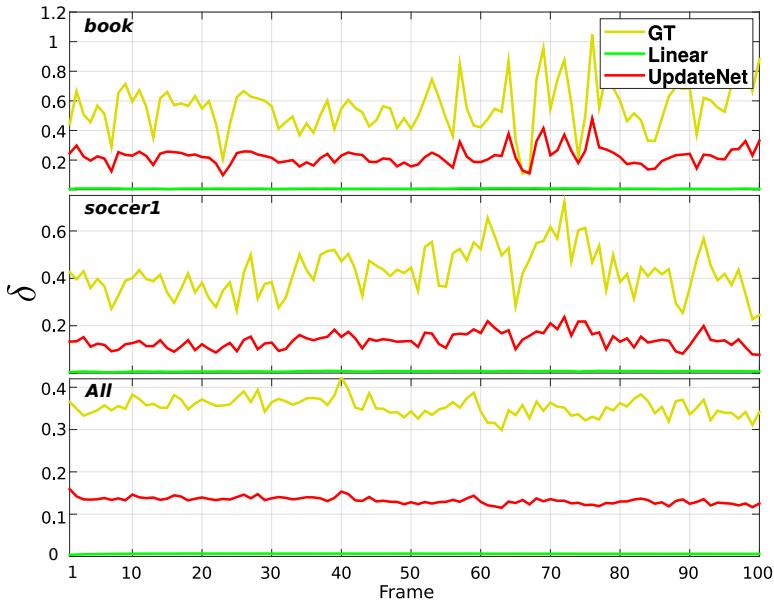


Figure 5.4 – **Change rate between contiguous frames.** We present individual results for two example videos (top, middle) and average results for all videos in VOT2018.

5.4.6 Generality and Tracking Speed

In this section, we evaluate the generality of our UpdateNet by applying it to other Siamese trackers as shown in Figure 5.5. It presents results on VOT2018 in terms of EAO with respect to the tracking speed. We measure tracking speed in frames per second (FPS) and use a logarithmic scale on its axis. We observe that we improve on Siamese trackers, e.g. SiamFC (Bertinetto et al., 2016b) and DaSiamRPN (Zhu et al., 2018) by adding a very small temporal overhead. Finally the top-performance trackers are shown in Figure 5.6. We compare with trackers including DRT (Sun et al., 2018a), DeepSTRCF (Li et al., 2018e), LSART (Sun et al., 2018b), R_MCPF (Zhang et al., 2019b), SRCT (Lee & Kim, 2018), CSRDCF (Lukezic et al., 2017), LADCF (Xu et al., 2018), MFT (Kristan et al., 2018), UPDT (Bhat et al., 2018) and ATOM (Danelljan et al., 2019) among others as in the challenge report (Kristan et al., 2018). Among the top trackers, our approach achieves superior performance while maintaining a very high efficiency. Further, our tracker obtains a performance relative gain of 2.8% over the base tracker DaSiamRPN (Zhu et al., 2018).

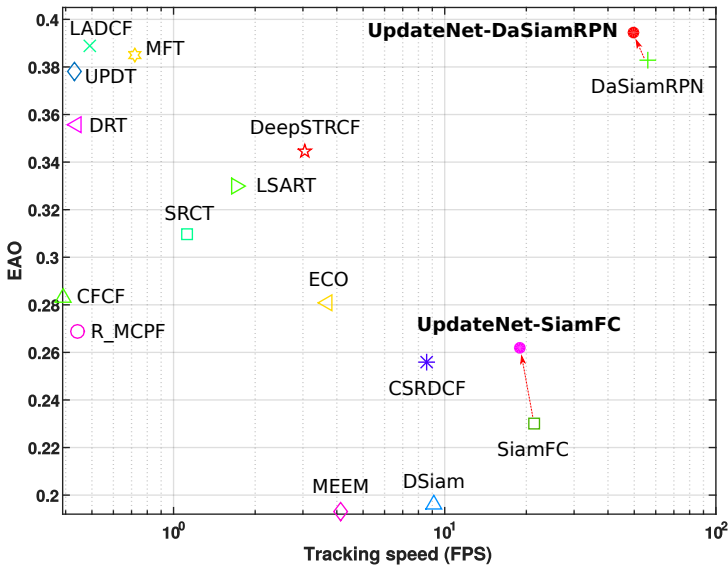


Figure 5.5 – **EAO vs. speed on VOT2018**. We compare our UpdateNet combined with two different Siamese trackers against the state-of-the-art methods. UpdateNet can substantially improve the tracking performance without significantly affecting the speed.

5.4.7 Fine-tuning the Linear Update Rate

The linear update in the previous section uses the update rate for SiamFC recommended by the authors (Valmadre et al., 2017) ($\gamma = 0.0102$) and for DaSiamRPN from the original tracker (Zhu et al., 2018) ($\gamma = 0$). We now investigate whether the linear update strategy can bring higher performance gains when fine-tuning the update rate on the test set. We test several update rates uniform sampled from the $[0, 0.2]$ interval. Figure 5.7 shows the performance of the linear update for DaSiamRPN (light green) and SiamFC (dark green). The red dashed line at the top and pink dashed line in the middle are the performance of our UpdateNet applied on DaSiamRPN and SiamFC respectively. We can see for SiamFC the peak performance is indeed achieved between 0.01 and 0.05. For DaSiamRPN, the original tracker with no-update performs best, which proves that for more complex Siamese trackers trained off-line, on-line linear update may even damage the performance. This shows that even a fine-tuned linear update cannot improve its results any further. Moreover, our UpdateNet outperforms all update rate values without requiring any

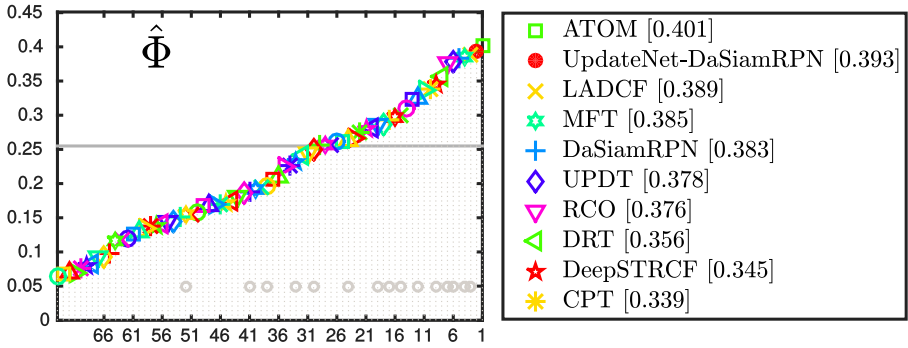


Figure 5.6 – **EAO performance on VOT2018.** We compare our method with the state-of-the-art methods on VOT2018. Our proposed approach achieves superior performance.

	DSiam	MemTrack	SiamFC		DaSiamRPN	
	(Guo et al., 2017)	(Yang & Chan, 2018)	Linear	UpdateNet	Linear	UpdateNet
EAO	0.181	0.273	0.235	0.289	0.439	0.481
A	0.492	0.533	0.529	0.543	0.619	0.610
R	2.934	1.441	1.908	1.320	0.262	0.206

Table 5.2 – **Results for other updating strategies on VOT2016.** DSiam (Guo et al., 2017) and MemTrack (Yang & Chan, 2018) use SiamFC as base tracker. The best two results are highlighted in red and blue fonts, respectively.

manual fine-tuning. Despite the need of a higher update rate for some videos, we can see how the performance continuously and rapidly decreases as the update rate increases, evidencing the unsuitability of a fixed and general update rate for all videos.

5.4.8 Comparison With Other Updating Strategies

Some recent approaches (Guo et al., 2017; Yang & Chan, 2018) proposed alternative updating strategies for Siamese trackers. Table 5.2 presents a comparison with DSiam (Guo et al., 2017) and MemTrack (Yang & Chan, 2018) on VOT2016, as the work (Yang & Chan, 2018) only reports results on this VOT edition (see Figure 5.5 for DSiam results on VOT2018). Our UpdateNet leads to a more effective update and higher tracking performance, while also being applicable to different Siamese

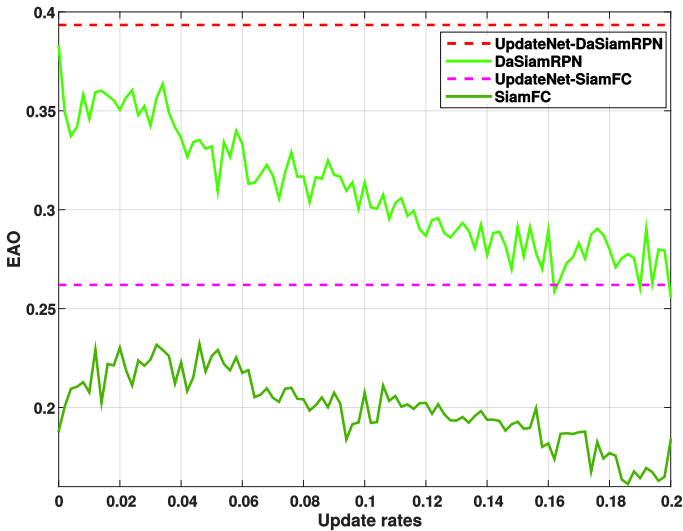


Figure 5.7 – **The linear update rate evaluation for DaSiamRPN and SiamFC on VOT2018 (Kristan et al., 2018)**. The x-axis is the update rate values. The y-axis is the EAO scores on VOT protocol (Kristan et al., 2018). The red and pink dashed lines are our UpdateNet performances with DaSiamRPN and SiamFC, respectively.

architectures. Despite the already excellent performance of DaSiamRPN, UpdateNet brings an improvement of 4.2%, reaching state-of-the-art. Moreover, our approach yields a substantial absolute gain of 5.6% in terms of robustness, which is a common weak point of Siamese trackers.

5.4.9 LaSOT Dataset

We test our model on the recent LaSOT dataset (Fan et al., 2018). Since long-term sequences are common in LaSOT, the updating component of the tracker is crucial, as more sudden variations may appear and object appearance may depart further from the input object template. We show the top-10 trackers, including MDNet (Nam & Han, 2016), VITAL (Song et al., 2018), StructSiam (Zhang et al., 2018), DSiam (Guo et al., 2017), SINT (Tao et al., 2016), STRCF (Li et al., 2018e), ECO (Danelljan et al., 2017), SiamFC (Bertinetto et al., 2016b) and DaSiamRPN (Zhu et al., 2018) in Figure 5.8. The results are presented following the official protocol. We can see how UpdateNet enhances the updating capabilities of DaSiamRPN and leads to a significant performance boost on all measures. As a result, our tracker

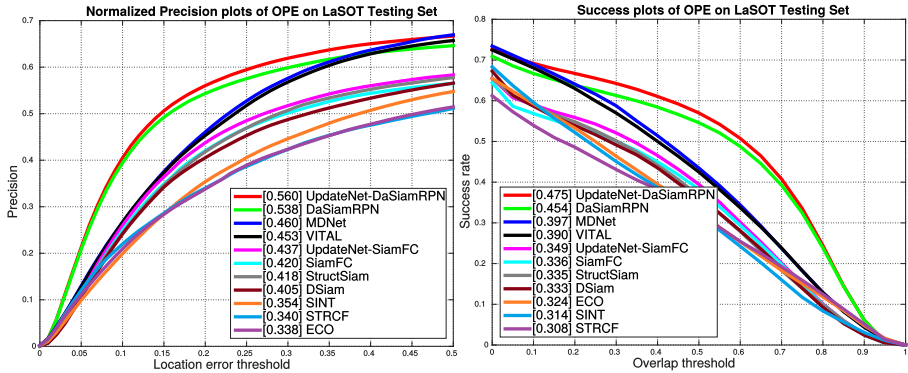


Figure 5.8 – Evaluation on LaSOT testing set. Normalized precision and success plots following the OPE protocol II.

	ATOM (Danelljan et al., 2019)	ECO (Danelljan et al., 2017)	CFNet (Valmadre et al., 2017)	MDNet (Nam & Han, 2016)	SiamFC Linear	UpdateNet	DaSiamRPN Linear	UpdateNet
Precision (%)	64.8	49.2	53.3	56.5	53.3	53.1	59.1	62.5
Norm. Prec. (%)	77.1	61.8	65.4	70.5	66.3	67.4	73.3	75.2
Success (%)	70.3	55.4	57.8	60.6	57.1	58.4	63.8	67.7

Table 5.3 – State-of-the-art comparison on TrackingNet. Our UpdateNet significantly improves DaSiamRPN (Zhu et al., 2018) with an absolute gain of 3.4% and 3.9%, in terms of precision and success. The best two results are highlighted in red and blue fonts, respectively.

with learned update strategy surpasses all state-of-the-art trackers on this dataset. This brings further evidence to the advantages of adaptive update strategy in terms of accurate object localization.

5.4.10 TrackingNet Dataset

We evaluate our UpdateNet-DaSiamRPN on the testing set of TrackingNet (Muller et al., 2018) using their three evaluation metrics (Table 5.3). Compared with DaSiamRPN, our UpdateNet+DaSiamRPN obtains absolute gains of 3.4%, 1.9% and 3.9% in terms of precision, normalized precision and success. UpdateNet leads to a significant performance improvement on all three metrics. This shows how a learning the model update can greatly benefit Siamese trackers on several datasets and under different measures.

5.5 Conclusions

Siamese trackers usually update their appearance template using a simple linear update rule. We identify several shortcomings of this linear update and propose to learn the updating step as an optimization problem. We employ a neural network, coined UpdateNet, that learns how to update the current accumulated template given the appearance template of the first frame, the current frame, and the accumulated template of the previous step. The proposed UpdateNet is general and can be integrated into all Siamese trackers. Comparable results on four benchmark datasets (VOT2016, VOT2018, LaSOT, and TrackingNet) show that the proposed approach to updating does significantly improve the performance of the trackers with respect to the standard linear update (or with respect to no-update at all).

5.6 Difference With Offline Weighted Fusion

Another possible update mechanism that improves on the linear update could be learning an offline weighted fusion of three templates (the input of UpdateNet). In order to demonstrate the benefits of more sophisticated updating mechanisms, we propose the following experiment.

UpdateNet uses a convolutional neural network that leverages previous templates to predict an accumulated template that is similar to the real one. Therefore, the fusion mechanism implemented by UpdateNet is more sophisticated than a simple offline weighted fusion, which depends on the actual input features and can be adapted accordingly. In order to confirm this, we propose here the following experiment to compare UpdateNet to an offline weighted fusion. We express the template update as a weighted linear combination $\tilde{T}_i = \alpha_{init} T_0^{GT} + \alpha_{accu} \tilde{T}_{i-1} + \alpha_{curr} T_i$. We initialize the three weights to 0, 0.9898, and 0.0102 respectively following the default settings for the linear update. Then, we train these weights with the same training process as UpdateNet until convergence, as shown in Figure 5.9.

Next we compare the EAO results on VOT2018. SiamFC with an offline weighted fusion achieves *0.198*, which is a little higher than the baseline linear update (*0.188*) but much lower than *0.262* achieved with UpdateNet. These results show that our UpdateNet is significantly better than an offline weighted fusion. We mainly attribute the superior results of UpdateNet to the following reasons. The offline weighted fusion learns a high value for α_{curr} . In this case, the tracker is likely to succumb to drift when the current template is not reliable. Instead of excessively relying on the current template, UpdateNet can benefit from all the input templates due to the representation bottleneck in the channel dimension. Furthermore, UpdateNet includes a non-linearity which allows it to better adjust to the non-linear

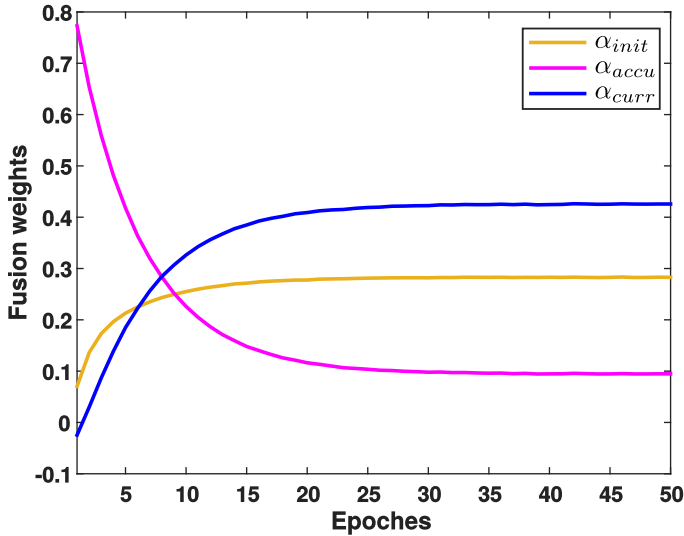


Figure 5.9 – Training of the learned fusion weights offline.

variations, such as rotation and object motion. Thus, yielding more expressive and reliable representations for the predicted template.

5.7 Visualization of Updating Templates

We provide additional accumulated templates of SiamFC for both linear update and UpdateNet in Figure 5.10 (similar to Figure 5.3 in the chapter). By visualizing more exemplar videos, we can see that UpdateNet learns templates which are more similar to the ground-truth and predicts more accurate response maps for cross-correlation. For visualization ease, we add a red cross ‘+’ to split the four channels of the template feature. The four channels are the most dynamic channels in the ground-truth template for the corresponding video. We select them as follows. For each $j \in \{1, \dots, C\}$ we compute the average difference in the template as $\Delta_j = \frac{1}{|N|} \sum_N \frac{1}{|A|} \sum_A |T_i^{GT} - T_{i-1}^{GT}|$, where N is the number of frames in a video and the sum runs over the spatial area of each channel of the feature maps (e.g. $A = 6 \times 6$). We select largest 4 channels in terms of Δ_j .

We can observe multiple interesting behaviors in Figure 5.10. Firstly, the accumulated templates using UpdateNet resemble the ground-truth more closely than those with linear update, see e.g. the bottom-right channel (106) in frame 125 of A ,

and the bottom-right channel (121) in frame 42 of F , where the same highlighted region appears for both UpdateNet and the ground-truth. The template of the linear update, instead, does not resemble the ground truth for either of these two sequences. In general, the accumulated templates from UpdateNet are almost as dynamic as the ground-truth templates, meaning that our UpdateNet can adapt to the template change in a video much better than linear update, which changes very slowly. Secondly, we can see in the cross-correlation response map how UpdateNet better predicts the object location, while linear update predicts many spurious peaks on the response map and the true peak in the center is less sharp, see e.g. frame 115 in example C with an additional peak, frame 76 in example C and frame 106 in example D with blurred peaks, frame 88 in example A and frame 7 in example B with multiple peaks, among others. To summarize, Figure 5.10 clearly shows that our strategy does not negatively interfere with the desired correlation properties of the learned features, on the contrary, it helps by adaptively updating the templates. On the other hand, the accumulated templates of the linear update change at a very slow rate and are inefficient in keeping up with the appearance variation exhibited in videos.

5.8 Change Rate for Update

In addition to Figure 5.4 in the chapter, we here provide similar results for the sequences shown in Figure 5.11 of this section. We calculate the change rate δ between templates of contiguous frames and show the results in Figure 5.11. Our UpdateNet provides an adaptive update strategy that is close to the change rate of the real template, while linear update can only offer a constant change rate. The change rate for UpdateNet follows the same trends as the ground-truth, see for example the high correlation with the high peaks in e.g. frame 50 in ‘soldier’, frame 60 in ‘butterfly’ and frame 61 in ‘blanket’. This leads to predicting better response maps as shown in Figure 5.10.

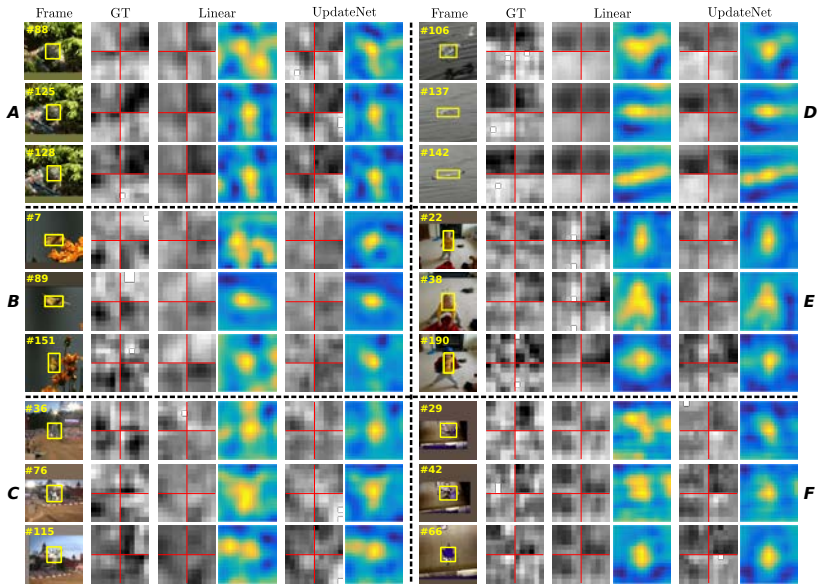


Figure 5.10 – **Visualization accumulated and ground-truth templates for SiamFC.** The first column shows the search region and the ground-truth box. ‘GT’ shows top four channels of the real template extracted from the ground-truth box. For each update strategy (‘Linear’ and ‘UpdateNet’) we show the accumulated templates and the resulting response map when applied to the search region, respectively.

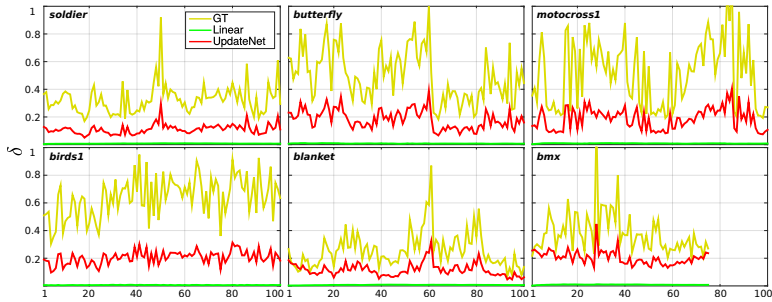


Figure 5.11 – **Change rate between contiguous frames.** We present additional results for six example videos in VOT2018.

6 Conclusions and Future Directions

This thesis aims at improving visual tracking in different modalities, including RGB modality, TIR modality and multi-modal RGB-T setting. First we summarize main conclusions of the thesis. Finally, we discuss possible directions for future work.

6.1 Conclusions

In this thesis, we investigate visual tracking in various modalities. Based on an analysis of tracking literature we identified three shortcomings of tracking existing systems:

- **Data-scarcity of TIR tracking:** The lack of large labeled datasets hampers the usage of convolutional neural networks for tracking in thermal infrared (TIR) images. Therefore, most state of the art methods on tracking for TIR data are still based on hand-crafted features.
- **End-to-end training for RGB-T tracking:** The RGB-T tracking predominantly uses hand-crafted methods for visual tracking, while these have limited performance. Deep learning methods are not yet applied into this field.
- **The model-update in RGB tracking:** In general, the appearance template in Siamese approaches is linearly combined with the accumulated template from the previous frame, resulting in an exponential decay of information over time. While such an approach to updating has led to improved results, its simplicity limits the potential gain likely to be obtained by learning to update.

In this thesis, we have explored three deep learning methods to address these issues. In Chapter 3, we propose to use image-to-image translation models. These models allow us to translate the abundantly available labeled RGB data to synthetic TIR data. We explore both the usage of paired and unpaired image translation models for this purpose. These methods provide us with a large labeled dataset of synthetic TIR sequences, on which we can train end-to-end optimal features for

tracking. To the best of our knowledge we are the first to train end-to-end features for TIR tracking. We perform extensive experiments on VOT-TIR2017 dataset. We show that a network trained on a large dataset of synthetic TIR data obtains better performance than one trained on the available real TIR data. Combining both data sources leads to further improvement. In addition, when we combine the network with motion features we outperform the state of the art with a relative gain of over 10%, clearly showing the efficiency of using synthetic data to train end-to-end TIR trackers.

In Chapter 4, we propose an end-to-end tracking framework for fusing the RGB and TIR modalities in RGB-T tracking. Our baseline tracker is DiMP (Discriminative Model Prediction), which employs a carefully designed target prediction network trained end-to-end using a discriminative loss. We analyze the effectiveness of modality fusion in each of the main components in DiMP, i.e. feature extractor, target estimation network, and classifier. We consider several fusion mechanisms acting at different levels of the framework, including pixel-level, feature-level and response-level. Our tracker is trained in an end-to-end manner, enabling the components to learn how to fuse the information from both modalities. As data to train our model, we generate a large-scale RGB-T dataset by considering an annotated RGB tracking dataset (GOT-10k) and synthesizing paired TIR images using an image-to-image translation approach. We perform extensive experiments on VOT-RGBT2019 dataset and RGBT210 dataset, evaluating each type of modality fusing on each model component. The results show that the proposed fusion mechanisms improve the performance of the single modality counterparts. We obtain our best results when fusing at the feature-level on both the IoU-Net and the model predictor, obtaining an EAO score of 0.391 on VOT-RGBT2019 dataset. With this fusion mechanism we achieve the state-of-the-art performance on RGBT210 dataset.

In Chapter 5, we propose to replace the hand-crafted update function with a method which learns to update. We use a convolutional neural network, called UpdateNet, which given the initial template, the accumulated template and the template of the current frame aims to estimate the optimal template for the next frame. The UpdateNet is compact and can easily be integrated into existing Siamese trackers. We demonstrate the generality of the proposed approach by applying it to two Siamese trackers, SiamFC and DaSiamRPN. Extensive experiments on VOT2016, VOT2018, LaSOT, and TrackingNet datasets demonstrate that our UpdateNet effectively predicts the new target template, outperforming the standard linear update. On the large-scale TrackingNet dataset, our UpdateNet improves the results of DaSiamRPN with an absolute gain of 3.9% in terms of success score.

6.2 Future Directions

We have identified several directions for future work. Large networks are known to provide excellent feature embedding (Bhat et al., 2019; Li et al., 2018a), but are often impractical for real-life applications. Network distillation has been successfully applied to improve image classification (Hinton et al., 2015). Normally it uses a teacher-student setup in which a teacher (large) network is used to guide a small student network. This is done by using a loss function that minimizes cross-entropy between the outputs of the student and teacher network for classification. As is well known in tracking, real-time tracking is an essential requirement. As now there are deep trackers which use many large network, achieving high-quality performance, but with high efficiency cost. Therefore, as a next step we plan to use network distillation for training efficient small networks which can significantly improve the tracking performance compared to directly training the student network on the data.

Recently, multi-modal tracking is attracting more and more attentions in the tracking community. In this thesis, we have made an effort to improving the RGB-T tracking and achieved significant tracking performance with the utility of synthetic TIR data generation. Next step, we will extend this method to other modalities, e.g. the RGB-D tracking which is also an important subject on VOT2019 challenge (VOT challenge, 2019). Similar as the RGB-T tracking, the hand-crafted trackers are still dominating the RGB-D tracking. We will propose specific and effective end-to-end training mechanism for RGB-D tracking, in order to benefit the RGB-D tracking with deep learning. For the implementation, we will refer to the technique in the paper (Li et al., 2019) and train an effective depth data generator. Then we apply it to generate large scale paired RGB-D data for training end-to-end RGB-D trackers.

Visual object tracking (VOT) and semi-supervised video object segmentation (VOS) have the identical task that both of them require estimating the position of an arbitrary target specified in the first frame of a video. Recently, more attention is paid on the VOS (Johnander et al., 2019) and also multi-task for VOT and VOS (Wang et al., 2019). There are certainly many relations between the two tasks considering their same track flow. We will make some efforts on the combination of the two tasks by referring to the works (Johnander et al., 2019; Wang et al., 2019). We will also investigate more video tasks, e.g. video object recognition, which is similar to visual tracking. The benefits of co-training of multi-tasks could be: on the one hand, we can improve the performance of visual tracking by introducing similar tasks. On the other hand, we can use one unified framework to solve multiple visual tasks.

A Publications

A.1 Scientific Articles

- **Lichao Zhang**, Abel Gonzalez-Garcia, Joost van de Weijer, Martin Danelljan, Fahad Shahbaz Khan. "Synthetic Data Generation for End-to-End Thermal Infrared Tracking." In *IEEE Transactions on Image Processing (TIP)*, 28.4 (2018): 1837-1850.
- **Lichao Zhang**, Martin Danelljan, Abel Gonzalez-Garcia, Joost van de Weijer, Fahad Shahbaz Khan. "Multi-modal fusion for end-to-end RGB-T tracking." In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2019.
- **Lichao Zhang**, Abel Gonzalez-Garcia, Joost van de Weijer, Martin Danelljan, Fahad Shahbaz Khan. "Learning the Model Update for Siamese Trackers." In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.
- Yaxing Wang, **Lichao Zhang**, Joost van de Weijer. "Ensembles of generative adversarial networks." In *Workshop on Adversarial Training, NIPS*, Barcelona, Spain, 2016.
- Lu Yu, **Lichao Zhang**, Joost van de Weijer, Fahad Shahbaz Khan, Yongmei Cheng, and C. Alejandro Parraga. "Beyond eleven color names for image understanding." In *Machine Vision and Applications*, 29.2 (2018): 361-373.

A.2 Contributed Code and Models

- **Synthetic TIR Data Generator**
https://github.com/zhanglichao/generatedTIR_tracking
- **End-to-end Multi-modal Tracking Models**

https://github.com/zhanglichao/end2end_rgb_tracking

- **Siamese Trackers with UpdateNet**

<https://github.com/zhanglichao/updatenet>

Bibliography

- Berg, A., Ahlberg, J., & Felsberg, M. (2018). Generating visible spectrum images from thermal infrared. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshops*.
- Bertinetto, L., Valmadre, J., Golodetz, S., Miksik, O., & Torr, P. H. (2016a). Staple: Complementary learners for real-time tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 1401–1409).
- Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., & Torr, P. H. (2016b). Fully-convolutional siamese networks for object tracking. In *ECCV workshop*.
- Bhat, G., Danelljan, M., Van Gool, L., & Timofte, R. (2019). Learning discriminative model prediction for tracking. *CoRR*, *abs/1904.07220*.
- Bhat, G., Johnander, J., Danelljan, M., Shahbaz Khan, F., & Felsberg, M. (2018). Unveiling the power of deep tracking. In *European Conference on Computer Vision (ECCV)*.
- Bilodeau, G.-A., Torabi, A., St-Charles, P.-L., & Riahi, D. (2014). Thermal-visible registration of human silhouettes: A similarity measure performance evaluation. *Infrared Physics & Technology*, *64*, 79–86.
- Birchfield, S. T., & Rangarajan, S. (2005). Spatiograms versus histograms for region-based tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, (pp. 1158–1163). IEEE.
- Bolme, D. S., Beveridge, J. R., Draper, B. A., & Lui, Y. M. (2010). Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Bolme, D. S., Draper, B. A., & Beveridge, J. R. (2009). Average of synthetic exact filters. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Čehovin, L., Leonardis, A., & Kristan, M. (2016). Visual object tracking performance measures revisited. *Transactions on Image Processing*, *25*(3), 1261–1274.

- Chatfield, K., Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*.
- Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4), 834–848.
- Choi, J., Jin Chang, H., Fischer, T., Yun, S., Lee, K., Jeong, J., Demiris, Y., & Young Choi, J. (2018). Context-aware deep feature compression for high-speed visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Choi, J., Kwon, J., & Lee, K. M. (2017). Visual tracking by reinforced decision making. *CoRR, abs/1702.06291*.
- Conaire, C. Ó., O'Connor, N. E., & Smeaton, A. (2008). Thermo-visual feature fusion for object tracking using multiple spatiogram trackers. *Machine Vision and Applications*, 19(5-6), 483–494.
- Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Danelljan, M., Bhat, G., Khan, F. S., & Felsberg, M. (2017). Eco: efficient convolution operators for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Danelljan, M., Bhat, G., Khan, F. S., & Felsberg, M. (2019). Atom: Accurate tracking by overlap maximization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Danelljan, M., Häger, G., Khan, F., & Felsberg, M. (2014a). Accurate scale estimation for robust visual tracking. In *BMVA British Machine Vision Conference (BMVC)*.
- Danelljan, M., Hager, G., Shahbaz Khan, F., & Felsberg, M. (2015a). Convolutional features for correlation filter based visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*.
- Danelljan, M., Hager, G., Shahbaz Khan, F., & Felsberg, M. (2015b). Learning spatially regularized correlation filters for visual tracking. In *IEEE International Conference on Computer Vision (ICCV)*.
- Danelljan, M., Hager, G., Shahbaz Khan, F., & Felsberg, M. (2016a). Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Danelljan, M., Robinson, A., Khan, F. S., & Felsberg, M. (2016b). Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision (ECCV)*.
- Danelljan, M., Shahbaz Khan, F., Felsberg, M., & Van de Weijer, J. (2014b). Adaptive color attributes for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 1090–1097).
- Davis, J. W., & Keck, M. A. (2005). A two-stage template approach to person detection in thermal imagery. In *Application of Computer Vision, 2005. WACV/MOTIONS'05 Volume 1. Seventh IEEE Workshops on*.
- Denton, E. L., Chintala, S., Fergus, R., et al. (2015). Deep generative image models using a laplacian pyramid of adversarial networks. In *Annual Conference on Neural Information Processing Systems (NIPS)*.
- Dong, C., Loy, C. C., He, K., & Tang, X. (2015). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2), 295–307.
- ECO (2017). <https://github.com/martin-danelljan/ECO>.
- Emami, A., Dadgostar, F., Bigdeli, A., & Lovell, B. C. (2012). Role of spatiotemporal oriented energy features for robust visual tracking in video surveillance. In *IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*.
- Fan, H., Lin, L., Yang, F., Chu, P., Deng, G., Yu, S., Bai, H., Xu, Y., Liao, C., & Ling, H. (2018). Lasot: A high-quality benchmark for large-scale single object tracking. *CoRR*, abs/1809.07845.
- Fan, H., & Ling, H. (2018). Siamese cascaded region proposal networks for real-time visual tracking. *CoRR*, abs/1812.06148.
- Felsberg, M., Berg, A., Hager, G., Ahlberg, J., Kristan, M., Matas, J., Leonardis, A., Čehovin, L., Fernandez, G., Vojir, T., et al. (2015). The thermal infrared visual object tracking vot-tir2015 challenge results. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*.
- Felsberg, M., Kristan, M., Matas, J., Leonardis, A., Pflugfelder, R., Häger, G., Berg, A., Eldesokey, A., Ahlberg, J., Čehovin, L., et al. (2016). The thermal infrared visual object tracking vot-tir2016 challenge results. In *European Conference on Computer Vision (ECCV)*.

- Gade, R., Jørgensen, A., & Moeslund, T. B. (2013). Long-term occupancy analysis using graph-based optimisation in thermal imagery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Gade, R., & Moeslund, T. B. (2014). Thermal cameras and applications: a survey. *Machine vision and applications*, 25(1), 245–262.
- Galoogahi, H. K., Sim, T., & Lucey, S. (2013). Multi-channel correlation filters. In *IEEE International Conference on Computer Vision (ICCV)*.
- Gao, C., Du, Y., Liu, J., Lv, J., Yang, L., Meng, D., & Hauptmann, A. G. (2016). Infar dataset: Infrared action recognition at different times. *Neurocomputing*, 212, 36–47.
- González, A., Fang, Z., Socarras, Y., Serrat, J., Vázquez, D., Xu, J., & López, A. M. (2016). Pedestrian detection at day/night time with visible and fir cameras: A comparison. *Sensors*, 16(6), 820.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Annual Conference on Neural Information Processing Systems (NIPS)*.
- Gundogdu, E., & Alatan, A. A. (2017). Good features to correlate for visual tracking. *arXiv preprint arXiv:1704.06326*.
- Guo, Q., Feng, W., Zhou, C., Huang, R., Wan, L., & Wang, S. (2017). Learning dynamic siamese network for visual object tracking. In *IEEE International Conference on Computer Vision (ICCV)*.
- He, A., Luo, C., Tian, X., & Zeng, W. (2018). A twofold siamese network for real-time object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Held, D., Thrun, S., & Savarese, S. (2016). Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision (ECCV)*.
- Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2012). Exploiting the circulant structure of tracking-by-detection with kernels. In *European conference on computer vision*, (pp. 702–715). Springer.

- Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2015). High-speed tracking with kernelized correlation filters. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 37(3), 583–596.
- Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- Hoffman, J., Gupta, S., & Darrell, T. (2016). Learning with side information through modality hallucination. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Huang, L., Zhao, X., & Huang, K. (2018). Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *CoRR*, abs/1810.11981.
- Hwang, S., Park, J., Kim, N., Choi, Y., & Kweon, I. S. (2013). Multispectral pedestrian detection: Benchmark dataset and baseline. *Integrated Comput.-Aided Eng*, 20, 347–360.
- Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Jiang, B., Luo, R., Mao, J., Xiao, T., & Jiang, Y. (2018). Acquisition of localization confidence for accurate object detection. In *European Conference on Computer Vision (ECCV)*, (pp. 784–799).
- Johnander, J., Danelljan, M., Brissman, E., Khan, F. S., & Felsberg, M. (2019). A generative appearance model for end-to-end video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 8953–8962).
- Kiani Galoogahi, H., Fagg, A., & Lucey, S. (2017). Learning background-aware correlation filters for visual tracking. In *IEEE International Conference on Computer Vision (ICCV)*.
- Kiani Galoogahi, H., Sim, T., & Lucey, S. (2015). Correlation filters with limited boundaries. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Cehovin Zajc, L., Vojir, T., Hager, G., Lukezic, A., Eldesokey, A., & Fernandez, G. (2017a). The visual object tracking vot2017 challenge results. In *Proceedings of the International Conference on Computer Vision Workshops*.

- Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Cehovin Zajc, L., Vojir, T., Hager, G., Lukežic, A., Eldesokey, A., et al. (2017b). The visual object tracking vot2017 challenge results. In *Proceedings of the IEEE International Conference on Computer Vision*, (pp. 1949–1972).
- Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Zajc, L. Č., Vojir, T., Bhat, G., Lukežič, A., Eldesokey, A., et al. (2018). The sixth visual object tracking vot2018 challenge results. In *ECCV workshop*.
- Kristan, M., Matas, J., Leonardis, A., Vojir, T., Pflugfelder, R., Fernandez, G., Nebehay, G., Porikli, F., & Čehovin, L. (2016a). A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11), 2137–2155.
- Kristan, M., Pflugfelder, R., Leonardis, A., Matas, J., Porikli, F., Cehovin, L., Nebehay, G., Fernandez, G., Vojir, T., Gatt, A., et al. (2016b). The visual object tracking vot2016 challenge results. In *Proceedings of the European Conference on Computer Vision Workshops*.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Annual Conference on Neural Information Processing Systems (NIPS)*.
- Lee, H., & Kim, D. (2018). Salient region-based online object tracking. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, (pp. 1170–1177). IEEE.
- Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., & Yan, J. (2018a). Siamrpn++: Evolution of siamese visual tracking with very deep networks. *CoRR, abs/1812.11703*.
- Li, B., Yan, J., Wu, W., Zhu, Z., & Hu, X. (2018b). High performance visual tracking with siamese region proposal network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Li, C., Cheng, H., Hu, S., Liu, X., Tang, J., & Lin, L. (2016). Learning collaborative sparse representation for grayscale-thermal tracking. *Transactions on Image Processing*, 25(12), 5743–5756.
- Li, C., Liang, X., Lu, Y., Zhao, N., & Tang, J. (2018c). Rgb-t object tracking: benchmark and baseline. *CoRR, abs/1805.08982*.
- Li, C., Sun, X., Wang, X., Zhang, L., & Tang, J. (2017a). Grayscale-thermal object tracking via multitask laplacian sparse representation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(4), 673–681.

- Li, C., & Wand, M. (2016). Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision (ECCV)*.
- Li, C., Zhao, N., Lu, Y., Zhu, C., & Tang, J. (2017b). Weighted sparse representation regularized graph learning for rgb-t object tracking. In *Proceedings of the 25th ACM international conference on Multimedia*, (pp. 1856–1864). ACM.
- Li, C., Zhu, C., Huang, Y., Tang, J., & Wang, L. (2018d). Cross-modal ranking with soft consistency and noisy labels for robust rgb-t tracking. In *European Conference on Computer Vision (ECCV)*, (pp. 808–823).
- Li, F., Tian, C., Zuo, W., Zhang, L., & Yang, M.-H. (2018e). Learning spatial-temporal regularized correlation filters for visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Li, Z., Dekel, T., Cole, F., Tucker, R., Snavely, N., Liu, C., & Freeman, W. T. (2019). Learning the depths of moving people by watching frozen people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 4521–4530).
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, (pp. 740–755). Springer.
- Liu, H., & Sun, F. (2012). Fusion tracking in color and infrared images using joint sparse representation. *Science China Information Sciences*, 55(3), 590–599.
- Liu, L., Xing, J., Ai, H., & Ruan, X. (2012). Hand posture recognition using finger geometric feature. In *International Conference on Pattern Recognition (ICPR)*.
- Lukezic, A., Vojír, T., Zajc, L. C., Matas, J., & Kristan, M. (2017). Discriminative correlation filter with channel and spatial reliability. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Ma, C., Huang, J.-B., Yang, X., & Yang, M.-H. (2015a). Hierarchical convolutional features for visual tracking. In *IEEE International Conference on Computer Vision (ICCV)*.
- Ma, C., Yang, X., Zhang, C., & Yang, M.-H. (2015b). Long-term correlation tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

- Mueller, M., Smith, N., & Ghanem, B. (2016). A benchmark and simulator for uav tracking. In *European Conference on Computer Vision (ECCV)*.
- Mueller, M., Smith, N., & Ghanem, B. (2017a). Context-aware correlation filter tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Mueller, M., Smith, N., & Ghanem, B. (2017b). Context-aware correlation filter tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 1396–1404).
- Muller, M., Bibi, A., Giancola, S., Alsubaihi, S., & Ghanem, B. (2018). Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *European Conference on Computer Vision (ECCV)*.
- Nam, H., Baek, M., & Han, B. (2016). Modeling and propagating cnns in a tree structure for visual tracking. *arXiv preprint arXiv:1608.07242*.
- Nam, H., & Han, B. (2016). Learning multi-domain convolutional neural networks for visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Nocedal, J., & Wright, S. J. (2006). Numerical optimization 2nd.
- Palmero, C., Clapés, A., Bahnsen, C., Møgelmoose, A., Moeslund, T. B., & Escalera, S. (2016). Multi-modal rgb–depth–thermal human body segmentation. *International Journal of Computer Vision*, 118(2), 217–239.
- Park, E., & Berg, A. C. (2018). Meta-tracker: Fast and robust online adaptation for visual object trackers. *arXiv preprint arXiv:1801.03049*.
- Perarnau, G., van de Weijer, J., Raducanu, B., & Álvarez, J. M. (2016). Invertible conditional gans for image editing. In *Advances in Neural Information Processing Systems 2016 Workshop on Adversarial Training*.
- Portmann, J., Lynen, S., Chli, M., & Siegwart, R. (2014). People detection and tracking from aerial thermal views. In *International Conference on Robotics and Automation (ICRA)*.
- Real, E., Shlens, J., Mazzocchi, S., Pan, X., & Vanhoucke, V. (2017). Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 5296–5305).

- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 779–788).
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Annual Conference on Neural Information Processing Systems (NIPS)*.
- Renoust, B., Le, D.-D., & Satoh, S. (2016). Visual analytics of political networks from face-tracking of news video. *IEEE Transactions on Multimedia*, 18(11), 2184–2195.
- RGB-T dataset (2018). <https://sites.google.com/view/ahutracking001>.
- Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3), 211–252.
- Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. In *Annual Conference on Neural Information Processing Systems (NIPS)*.
- Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Song, X., Herranz, L., & Jiang, S. (2017a). Depth cnns for rgb-d scene recognition: Learning from scratch better than transferring from rgb-cnns. In *AAAI Conference on Artificial Intelligence (AAAI)*.
- Song, Y., Ma, C., Gong, L., Zhang, J., Lau, R. W., & Yang, M.-H. (2017b). Crest: Convolutional residual learning for visual tracking. In *IEEE International Conference on Computer Vision (ICCV)*.
- Song, Y., Ma, C., Wu, X., Gong, L., Bao, L., Zuo, W., Shen, C., Lau, R. W., & Yang, M.-H. (2018). Vital: Visual tracking via adversarial learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Sun, C., Wang, D., Lu, H., & Yang, M.-H. (2018a). Correlation tracking via joint discrimination and reliability learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Sun, C., Wang, D., Lu, H., & Yang, M.-H. (2018b). Learning spatial-aware regressions for visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Tao, R., Gavves, E., & Smeulders, A. W. (2016). Siamese instance search for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Torabi, A., Massé, G., & Bilodeau, G.-A. (2012). An iterative integrated framework for thermal-visible image registration, sensor fusion, and people tracking for video surveillance applications. *Computer Vision and Image Understanding*, 116(2), 210–221.
- Valmadre, J., Bertinetto, L., Henriques, J., Vedaldi, A., & Torr, P. H. (2017). End-to-end representation learning for correlation filter based tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Valmadre, J., Bertinetto, L., Henriques, J. F., Tao, R., Vedaldi, A., Smeulders, A. W., Torr, P. H., & Gavves, E. (2018). Long-term tracking in the wild: A benchmark. In *European Conference on Computer Vision (ECCV)*, (pp. 670–685).
- Van De Weijer, J., Schmid, C., Verbeek, J., & Larlus, D. (2009). Learning color names for real-world applications. *Transactions on Image Processing*, 18(7), 1512–1523.
- VOT challenge (2019). www.votchallenge.net/vot2019.
- Wang, Q., Gao, J., Xing, J., Zhang, M., & Hu, W. (2017). Dcfnet: Discriminant correlation filters network for visual tracking. *arXiv preprint arXiv:1704.04057*.
- Wang, Q., Teng, Z., Xing, J., Gao, J., Hu, W., & Maybank, S. (2018). Learning attentions: residual attentional siamese network for high performance online visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Wang, Q., Zhang, L., Bertinetto, L., Hu, W., & Torr, P. H. (2019). Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 1328–1338).
- Wu, Y., Blasch, E., Chen, G., Bai, L., & Ling, H. (2011). Multiple source data fusion via sparse representation for robust visual tracking. In *14th International Conference on Information Fusion*, (pp. 1–8). IEEE.

- Wu, Y., Lim, J., & Yang, M.-H. (2013). Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Wu, Y., Lim, J., & Yang, M.-H. (2015). Object tracking benchmark. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 37(9), 1834–1848.
- Wu, Z., Fuller, N., Theriault, D., & Betke, M. (2014). A thermal infrared video benchmark for visual analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*.
- Xu, D., Ouyang, W., Ricci, E., Wang, X., & Sebe, N. (2017). Learning cross-modal deep representations for robust pedestrian detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Xu, T., Feng, Z.-H., Wu, X.-J., & Kittler, J. (2018). Learning adaptive discriminative correlation filters via temporal consistency preserving spatial feature selection for robust visual tracking. *CoRR, abs/1807.11348*.
- Yang, T., & Chan, A. B. (2018). Learning dynamic memory networks for object tracking. In *European Conference on Computer Vision (ECCV)*.
- Yao, Y., Wu, X., Zhang, L., Shan, S., & Zuo, W. (2018). Joint representation and truncated inference learning for correlation filter based tracking. In *European Conference on Computer Vision (ECCV)*.
- Youtube Statistics (2019). <https://merchdope.com/youtube-stats>.
- Yu, X., & Yu, Q. (2018). Online structural learning with dense samples and a weighting kernel. *Pattern Recognition Letters*, 105, 59–66.
- Yu, X., Yu, Q., Shang, Y., & Zhang, H. (2017). Dense structural learning for infrared object tracking at 200+ frames per second. *Pattern Recognition Letters*, 100, 152–159.
- Zhang, K., Zhang, L., Liu, Q., Zhang, D., & Yang, M.-H. (2014). Fast visual tracking via dense spatio-temporal context learning. In *European Conference on Computer Vision (ECCV)*.
- Zhang, L., Bi, D., Zha, Y., Gao, S., Wang, H., & Ku, T. (2016a). Robust and fast visual tracking via spatial kernel phase correlation filter. *Neurocomputing*, 204, 77–86.
- Zhang, L., Gonzalez-Garcia, A., van de Weijer, J., Danelljan, M., & Khan, F. S. (2019a). Synthetic data generation for end-to-end thermal infrared tracking. *Transactions on Image Processing*, 28(4), 1837–1850.

- Zhang, R., Isola, P., & Efros, A. A. (2016b). Colorful image colorization. In *European Conference on Computer Vision (ECCV)*.
- Zhang, T., Xu, C., & Yang, M.-H. (2019b). Learning multi-task correlation particle filters for visual tracking. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 41(2), 365–378.
- Zhang, Y., Wang, L., Qi, J., Wang, D., Feng, M., & Lu, H. (2018). Structured siamese network for real-time visual tracking. In *European Conference on Computer Vision (ECCV)*.
- Zhu, G., Porikli, F., & Li, H. (2016). Beyond local search: Tracking objects everywhere with instance-specific proposals. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*.
- Zhu, Z., Wang, Q., Li, B., Wu, W., Yan, J., & Hu, W. (2018). Distractor-aware siamese networks for visual object tracking. In *European Conference on Computer Vision (ECCV)*.