SINGLE AND MULTI-OBJECT VIDEO TRACKING USING LOCAL AND DEEP ARCHITECTURES

A Thesis presented to

the Faculty of the Graduate School

at the University of Missouri

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

NOOR MUSHREQ ABDELHAMEED AL-SHAKARJI

Dr. Kannappan Palaniappan, Thesis Supervisor

MAY 2022

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

SINGLE AND MULTI-OBJECT VIDEO TRACKING USING LOCAL AND DEEP ARCHITECTURES

presented by Noor Mushreq Abdelhameed Al-Shakarji,
a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Kannappan Palaniappan

Dr. Jeffrey Uhlmann

Dr. Filiz Bunyak

Dr. Prasad Calyam

Dr. Bimal Balakrishnan

ACKNOWLEDGMENTS

First and foremost, I would like to praise and thank God, the Almighty, who has granted me countless blessings, knowledge, opportunity, and great supporters, so that I have been finally able to accomplish the thesis.

I would like to show my greatest appreciation to my supervisor, **Dr. Kannappan Palaniappan**, whose expertise and knowledge were invaluable brilliant ideas. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level. I am thankful for your continuous support to build my confidence to get great opportunities.

I would also like to thank **Dr. Filiz Bunyak** for her guidance to make my work go the extra mile. Thank you for all the support has shown me. You always give me a valuable, robust solutions, and suggestions for any question I ask. Your kindness never ends.

Special thanks to my committee members, **Dr. Jeffrey Uhlmann, Dr. Prasad Calyam, and Dr. Bimal Balakrishnan** who have had a great impact on my progress. Your suggestions were valuable to enrich my research.

Also, I would like to thank my husband, **Ahammd**, who has been a constant source of support and encouragement during the challenges of graduate school. You are my life-long companion. Nothing has value without you. I am truly thankful for having you in my life. Also, I will never forget to thank my two eyes, my kids, **Yaqeen** and **Roya** for their kindness and patience during my study journey.

Finally, I would like to thank my big family, (my grandmother, father, mother, brothers, and sisters) for their support and prayers for me during the compilation of this dissertation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Moving object tracking is a fundamental computer vision task with a wide variety of real-life applications ranging from surveillance and autonomous systems to biomedical video analysis. A robust, accurate, scalable, and high-performance multi-object tracking (MOT) system of sized objects, requires novel approaches for visual appearance adaptation and generalized learning to handle challenging cases including object shape and viewpoint invariance, illumination invariance, complex object dynamics, clutter in the scene, partial or full occlusions and degraded environments. In this dissertation, our tracking system develops two pipelines and a fusion mechanism to provide precise trajectory information for detecting moving objects of interest and trajectories for object behavior and activity-based scene understanding. For single object trajectory estimation, we extended the recognition and feature fusion based single object tracking framework called Likelihood of Features Tracker (LoFT) with color attributes, scale selection, and kernelized correlation filter modules, to improve object appearance description, adaptation to scale changes, and localization of the target within the search window. For multi-object trajectories, we propose a time-efficient detect-track-and-predict system based on a novel three-step cascaded data association scheme. M2Track combines a fast hybrid spatial distance-based gating short-term data association, a robust tracklet-linking stage using discriminative and deep-learning-based object appearance models, with an explicit occlusion handling module relying not only on motion patterns but also on environmental context including the presence of potential occluders, and other foreground and background objects in the scene. Experimental results on different international challenge benchmarks, tasks, and datasets ranging from wide aerial motion imagery and full-motion video to biomedical microscopy videos demonstrate the robustness and efficiency of our pipelines that reach state-of-the-art performance.

Chapter 1

Introduction

Technologies in computer vision and machine intelligence have served our lives for decades. Producing meaningful, contextual information from the real world is not an easy task. For many years, researchers attempt to adapt and mimic human capabilities for automated purposes. Computer vision involves capturing, processing, and analyzing real-world images and videos to accomplish a certain goal, which becomes more complicated and more in demand with the recent advances in sensor technologies for both cameras and platforms. Some of these technologies involve robust video compression, summarization, or activity analysis and recognition.

One of the most common and challenging tasks in a computer vision framework is object tracking. There is a wide variety of applications in this field including visual surveillance, video summarization, sports video analysis, and biomedical video analysis. There is no specific way to classify object tracking methods. Different literatures classify them depending on different aspects (e.g., number of objects, detection objects of interest approaches, or different object representations). Figure 1.1 elaborates some of these categories. Object tracking aims to extract the motion trajectory of the object(s) of interest in a video sequence. For single object tracking, tracking approach can be categorized in various ways (e.g., detection-based [5, 6] versus recognition-based [7, 8]; or according to how the objects are localized within the search region, as generative[9, 10] versus discriminative methods [11, 12]). For multi-object tracking, the most popular categories are tracking-by-detection [13, 14, 15, 16, 17, 18], and tracking-by-model evolution [19, 20, 21]. Tracking-by-detection consists of a detection module that localizes the objects of interest in a frame, and an association module that links these detected objects in time, maintaining their identity and producing object trajectories. Tracking-by-model evolution involves an initialization step to locate the objects of interest on the first frame followed by a per-object model evolution in time by using deformable models such as active contours to keep track of individual object states (position, motion, shape, and orientation) in the following frames.

Single object tracking is the process of estimating the locations of an object over time and is usually divided into three modules:

- Initialization: generating an object model by extracting the features from the predefined target of interest from the first frame.
- Localization: detecting the object on the following frames by finding the most relevant match with the object model.
- Update: updating the object model with the new features of the new target located in the new frame.

The most challenges with the three modules are how to choose: the optimal features,

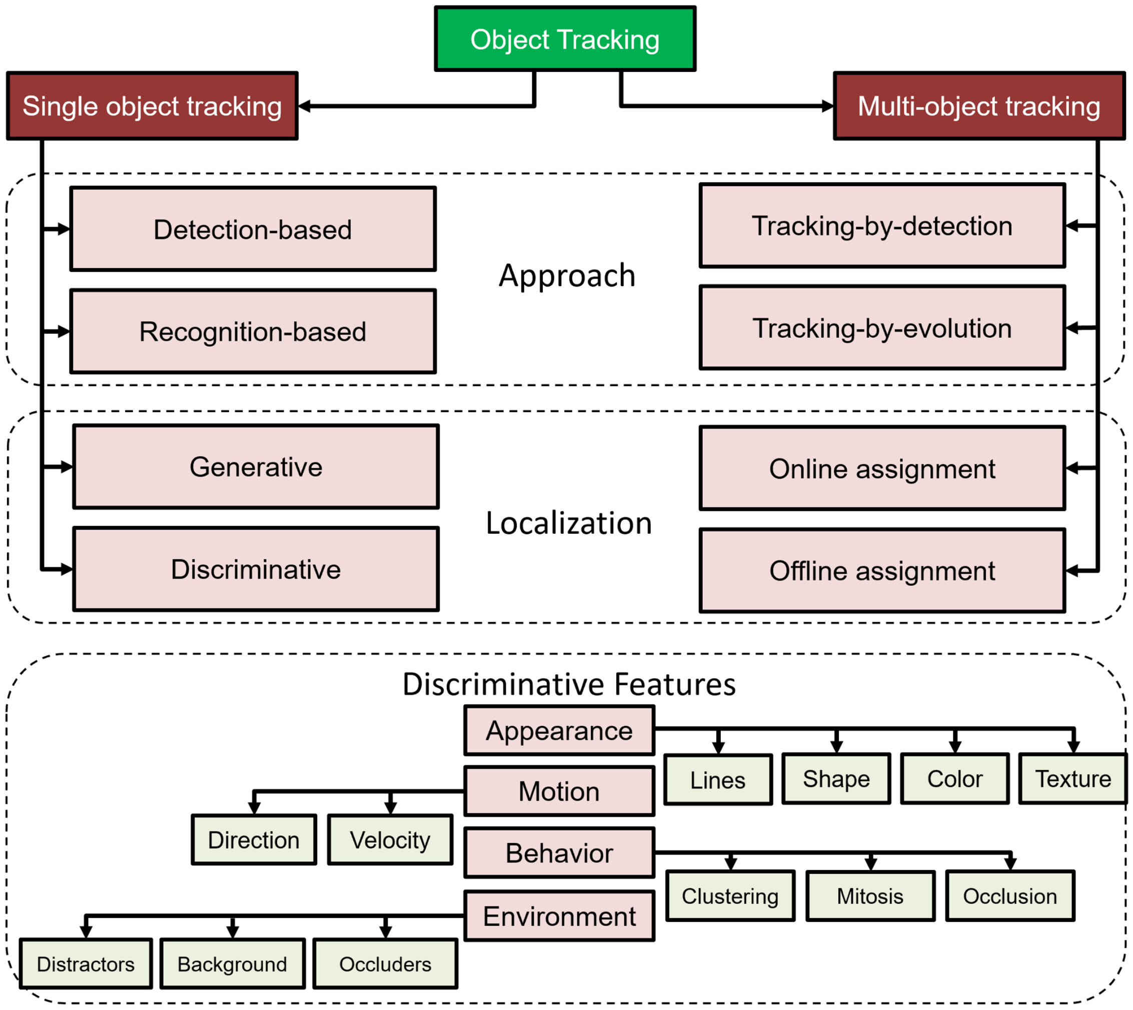


Figure 1.1: An overview of different categories and techniques for object tracking that can be used to build up a robust object tracking system.

the optimal matching approach, and the optimal update strategies. There are a lot of previous works with different strategies and methodologies for single object tracking. Some work used regular machine learning and computer vision approaches [22, 23, 24, 25, 26, 27] or used deep learning approaches [28, 29] in terms of initialization, localization and matching.

In [22], the Sum of Template And Pixel-wise LEarners (Staple) tracker was introduced. Staple combines complementary cues in a ridge regression framework using two image patch representations that are sensitive to complementary factors to learn a model online that is inherently robust to both color changes and deformations using a combination of a Correlation Filter using HoG features and a global color histogram.

In [28], Convolutional Features for Correlation Filters (CFCF) tracker was proposed. CFCF learns a discriminative convolution operator as its tracking model and poses the learning problem in the continuous spatial domain. The tracker employs a fully convolutional neural network (CNN) model, HoG, and Color Names (CN) as matching features.

Zhang et al. [23], introduced robust tracking via Multiple Experts using Entropy Minimization (MEEM). MEEM uses an online SVM with a re-detection based on the entropy of the score function restoration scheme, which allows a tracker to evolve backward to undo undesirable model updates. The tracker creates an example of experts by storing historical snapshots while tracking. The tracker can restore the snapshots when needed by the best of these experts selected by using an entropy minimization criterion.

Correlation filter-based tracking is a discriminative method in which a filter based on the object's appearance is used to estimate the most likely target location in the search window by distinguishing the target from its surrounding background. Convolution of the search window image with the filter is performed in the Fourier domain as element-wise multiplications to reduce complexity. The used filters are updated online to adapt to object appearance changes. Since their first use in visual tracking [12] [30], correlation filter-based tracking has been extended in multiple ways. [11, 31] improved adaptation to scale changes; while [22] incorporated histogram-based ridge regression learning to improve robustness to fast deformation and shape variations. Kernelized Correlation Filter (KCF) tracker that was introduced by /cite was the most popular correlation filter-based tracker. KCF operates on simple HoG features and Color Names (CN). The KCF tracker is equivalent to a Kernel Ridge Regression trained with thousands of sample patches around the object at different translations. It implements multi-thread multi-scale support, sub-cell peak estimation, and replacing the model update by linear interpolation with a more robust update scheme

Learning Adaptive Discriminative Correlation Filter on low-dimensional manifold (LADCF) tracker [25] was proposed. LADCF utilizes an adaptive spatial regularizer to train low-dimensional discriminative correlation filters. Adaptive spatial regularization and temporal consistency are combined in an objective function. Robustness is further considered by integrating HOG, Color Names, and ResNet-50 features.

Lukezic et al. [26] introduced the Discriminative Correlation Filter with Channel and Spatial Reliability (CSR-DCF). The spatial reliability map adapts the filter support to the part of the object suitable for tracking which overcomes both the problems of circular shift enabling an arbitrary search region size and the limitations related to the rectangular shape assumption. The reliability is estimated from the properties of the constrained least-squares solution. The channel reliability scores were used for weighting the per-channel filter responses in localization.

Likelihood of Features Tracking (LoFT) [27] was proposed. A recognition-based single target tracker that relies on the fusion of multiple complementary features. For each feature, a likelihood map is estimated by comparing the target feature histogram to search region sliding window histograms. Individual features perform differently depending on the target characteristic and environmental conditions during tracking.

Fusing different features enable the adaptation of the tracker to dynamic environment changes and target appearance variabilities.

Multi-object tracking (MOT) aims to locate multiple objects in a scene, maintain their identities in time, and form motion trajectories for further analysis. MOT divides into two steps:

- Detection step: the targets in each video frame are localized;
- Association step: the detected targets are assigned and connected to existing trajectories.

The detection step is datasets dependent. The optimal approach chosen depends on the context, the nature, and the availability of ground truth. Either trained or pre-trained models using transfer learning are used as deep learning detection-based approaches [32, 33, 34, 35]. While supervised [36, 37, 38], or unsupervised [39, 40, 41] can be used as machine learning-based approaches. The Association step is the core of multi-object tracking. The process is to find the correspondence matching between a set of new detections and the existing ones and identify the new objects entering the scene and treat them as new tracks. Association algorithms can be performed by gathering information from the most recent time step (i.e., past frames) which is called online mode [42, 43], whereas in offline (batch mode), the information from the entire video including past and future frames is used. Online mode tends to be sensitive to detection errors, and produce fragmented tracklets [44, 45]. While the offline mode has longer and more reliable trajectories. Some applications require to be used online such as surveillance systems.

Most of the multi-object trackers and approaches [46, 47, 48, 49, 50, 51, 52, 53] share the same concepts in finding the optimal association approach for assignment, the optimal optimization approach to implement the association approach, and the optimal features used for matching which can be machine learning features or deep learning features.

Pirsiavash et al. [47] proposed the Globally-optimal greedy (GOG) tracker. GOG formulates the multi-object tracking problem as the integer linear program (ILP). The model is based on the min-cost flow network, It allows to handle long sequences with a large number of objects, even in complex scenarios with long-term occlusion of objects.

In [46], Visual Intersection Over Union Tracker (VIOU) was proposed. VIOU is based on the IOU tracker and improved by track continuation using the 'visual tracker' if no detection is available. If a valid detection can be associated with the track again, the 'visual tracking' is stopped, and the tracker reduces to the original IOU tracker.

Online and Real Time Tracking with the GMPHD Filter using Group Management and Relative Motion Analysis (GMMA) tracker was proposed by [48]. GMMA tracker is an online multiple-objects tracking framework with a two-stage data association strategy with the Gaussian mixture probability hypothesis density (GM-PHD) filter. GMMA also includes an occlusion handling method based on group management and motion analysis.

In [49], the Continuous Energy Minimization for Multi-target tracking (CEM) tracker was introduced. CEM is an offline multi-object tracking algorithm as minimization of continuous energy overall target locations and all frames of a time window. Motion and interaction of all objects of interest in the scenes are represented by a suitable energy function.

The Simple Online and Real-time Tracking (SORT) tracker introduced by [50], is an online and real-time tracking-by-detection tracker that uses object positions and sizes for both motion estimation and data association. Then, It was extended to Simple online and real-time tracking with a deep association metric (DeepSORT) by integrating appearance information to improve the performance of SORT. DeepSORT [51] combines appearance information to track objects through long-term occlusions with fewer identity switches. An offline pre-training stage is required to learn a deep association metric on a large scale person re-identification dataset.

In [52], The deep affinity network (DAN) tracker was introduced. DAN is a deep neural network tracker that explicitly learns the affinity between objects over time. It is trained to predict the optimal linear assignment using ground truth assignment matrices as supervision. Visual features are first extracted from a VGG network and then processed by DAN to output a matrix of soft assignments, which finally are stitched into tracks using the Hungarian algorithm.

Shuai et. al. [53] proposed the Siamese Multi-Object Tracking (SiamMOT) tracker. SiamMOT is a region-based multi-object tracking network that detects and associates object instances simultaneously. The Siamese tracker models the motion of instances across frames, and it is used to temporally link detection in online multi-object tracking

Choosing the optimal approaches, which is the goal of this dissertation, is to develop a robust, accurate, and high-performance moving object tracking to provide comprehensive information about objects' movement for further use with different computer vision problems; Perform a persistent and adaptive tracking system under different challenges, scenarios, and demands; Design image appearance-based detections and associations tracker that is scalable to thousands of detections or measurements, running near real-time, fit to the variety of object size and density, invariant to rigid and deformable objects, work with cross-domain objects, for instance, aerial, ground-based, or biological videos.

1.1 Problem Statement

Visual object tracking is an integral part of many computer vision applications ranging from video surveillance to biomedical image analysis. In general, the success of these applications depends on robust and accurate detection and tracking under challenging conditions such as changing object appearance, scale and illumination variations, shadows, partial and full occlusions, the existence of distractors, and camera motion.

For single-object tracking, shape and appearance challenges, rapid object displacement, camera motion, and object scale changes are the most common challenge. For multi-object tracking, the performance of the tracking-by-detection methods is greatly influenced by the performance of the object detection methods and object-to-track association. False positives (false detections), false negatives (undetected objects), under-segmentation (merging of neighboring objects), and over-segmentation (object fragmentation) are major sources of detection error during the tracking process. Recent advances in object detection [54, 55, 56] are important for the success of the tracking task. Object-to-track association is the process of assigning detected objects to already existing tracks over time. Association ambiguities may lead to trajectory fragmentation, and identity switches, and may cause drift under occlusion [57, 58]. Moreover, the number of objects in the scene can cause a problematic issue in terms of complexity, since the number of objects is directly proportional to the complexity of implementation (e.g., the number of assigned hypotheses).

A robust visual moving object tracking system needs to generalize across different datasets. The nature of benchmark datasets can be one of the most important factors in tracking system performance. Different datasets have different characteristics, scalability, and requirements that need to be met. Wide aerial motion imaginary (WAMI) datasets have different characteristics and challenges compared to regular full-motion videos. WAMI videos suffer from extreme camera motion, low frame rate, and small object sizes. While for full-motion video, frequent object deformations, rapid scale and appearance changes, shadow, and illumination artifacts. While in biomedical fields, cell tracking datasets suffer from frequent object deformations, non-distinct appearance, low videos quality (contrast, resolution, etc.), and image acquisition artifacts. Figure 1.2 shows different datasets that have been used in our implementations. Our goal is to focus on developing a robust and fast-moving object tracking system to meet different dataset requirements.

1.2 Research Objective

Moving objects' behavior and scene understanding usually involve many computer vision problems, e.g., object detection, tracking, and behavior recognition. The objective of this dissertation is to develop a robust, accurate, and high-performance moving object(s) tracking system for different computer vision tasks. We are focusing on many computer vision problems to serve our tracking system. And that will be integrated to have a comprehensive full scene understanding. Staring from object

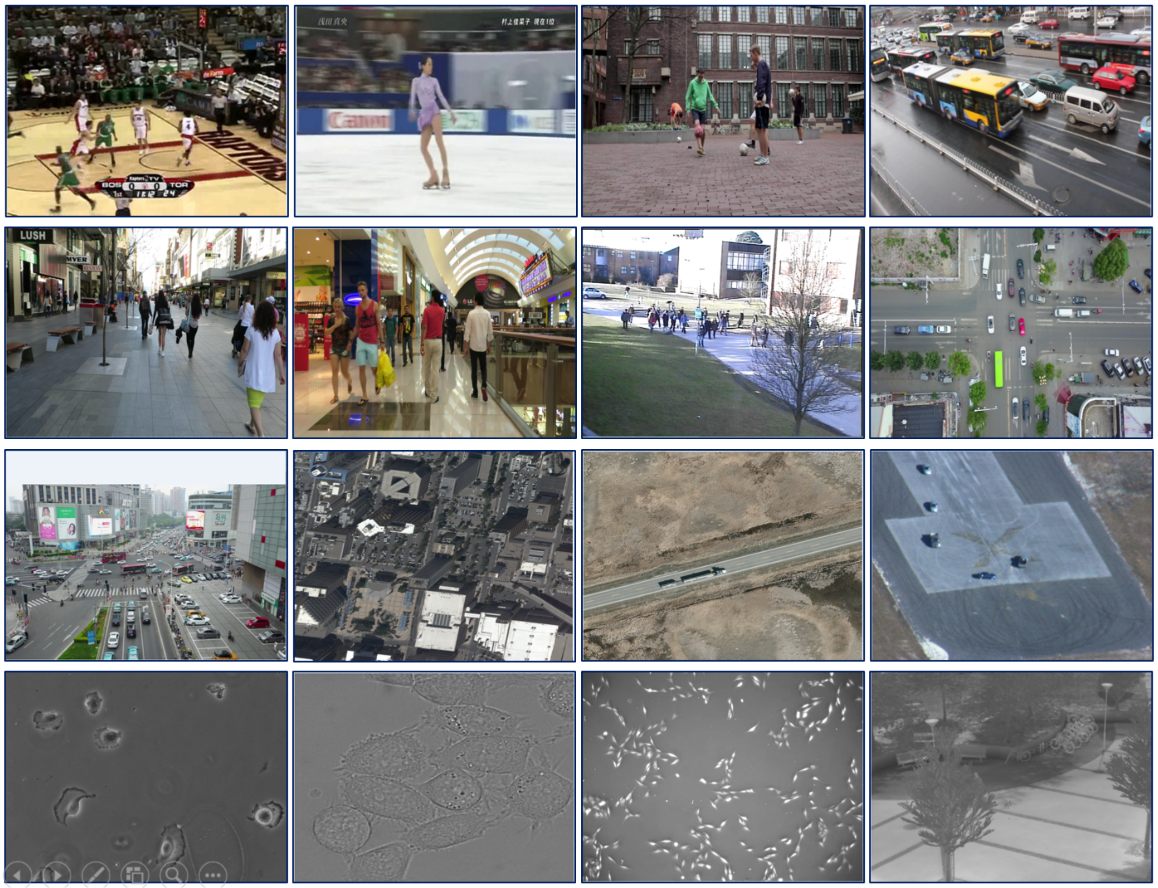


Figure 1.2: Different benchmarks for different purposes. The description for each benchmark from the upper left: 1st, 2nd, 3rd images are from the VOT benchmark [59] for single object tracking. The 4th image is from the UA-DETRAC benchmark [60] for multi-object detection and tracking. The 5th and 6th images are from the MOT16 [4] for multi-object tracking. The 7th is from the PETS09 benchmark [61] for person counting and density estimation, multi-object tracking, and event recognition. The 8th and 9th images are from the VisDrone benchmark[62] for multi-object tracking. The 10th image is from Albuquerque urban aerial imagery ABQ video [1] for single and multi-object tracking, vehicle detection, and 3D building reconstruction. The 11th and 12th images are from the VIRAT benchmark [63] for action recognition and tracking. The 13th,14th, and 15th images are from cell detection and tracking challenge [64]. The 16th image is from the VOT-IR benchmark for single object tracking in IR [59]

detection, tracking ends with semantic scene understanding for better analysis. Our tracking system has two pipelines:

- Single object tracker: recognition-based single object tracking system that relies on the fusion of multiple sources of information about the target and its environment to perform robust single object tracking that is invariant to occlusion, shadow, illumination changes, scale changes, and object deformation.

- Multi-object tracker: a lightweight, efficient modular design, a time-efficient detection-based multi-object tracking system that uses a multi-step cascaded data association scheme to ensure time efficiency by reducing the hypotheses through the steps, while preserving tracking accuracy by having discriminative object appearance models and retrieving trajectories after occlusion events

The main objective of our pipelines is to develop a robust, accurate, and high-performance moving object tracking system for full-motion/real-world video and wide aerial motion imagery as well as biomedical and microscopy videos. Our developed system's main objective is to perform persistent tracking under object appearance changes, objects similarity, object entering and leaving the scene, camera motion, background noises, occlusion, scenarios, environmental changes, etc.

1.3 Dissertation Layout

Our contribution to this dissertation has two folds. The first is to provide precise trajectory information for an individual object of interest using the single object tracking pipeline, and the second fold is to track all objects in the scene and produce their trajectories using the multi-object tracking pipeline.

1.3.1 Single Object Tracking Contributions:

Our group developed a Likelihood of Features Tracking (LoFT) system [27, 65, 66] that fuses multiple sources of information about the target and its environment to perform robust single object tracking. LoFT is a recognition-based target tracking system initially designed for vehicle tracking in low frame rate wide-area motion imagery. LoFT, shown in Figure 1.3, uses a rich feature set describing intensity, edge, shape, and texture information. Target to search window feature comparison is performed using cross-correlation and sliding window histogram differencing using an efficient integral histogram computation scheme. The process produces a likelihood map for each individual feature. Different features are more sensitive or more robust against different target characteristics and environmental conditions. Fusing different feature likelihood maps enables the adaptation of the tracker to scene dynamics and target appearance variabilities. We extended and improve LoFT by incorporating:

- **C-LoFT: extension of LoFT with color attributes:** we extended the original LoFT framework with color information to boost the performance of the tracker, particularly for video sequences where foreground and background have distinct color signatures. We used our extension of the Color Names (CN) feature scheme instead of RGB color for lower computational cost and to ensure robustness and invariance to shadow and illumination changes.

- **An operation mode decision unit to activate color use:** while color introduces rich information for object description, color can affect the tracker negatively, especially, when there is no adequate color dissimilarity between the tracked object and its surrounding region. C-LoFT uses an adaptive tracking scheme and uses

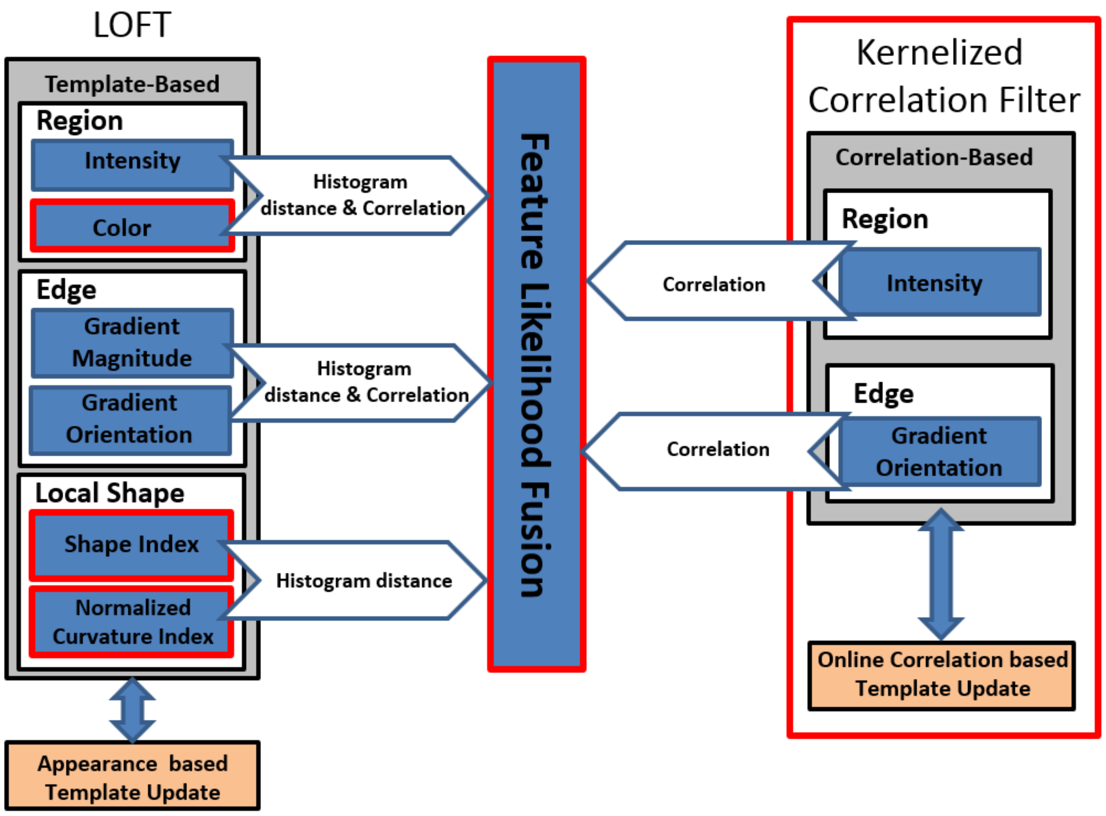

LOFT
Template-Based
Region
Intensity
Color
Edge
Gradient Magnitude
Gradient Orientation
Local Shape
Shape Index
Normalized Curvature Index
Appearance based Template Update
Histogram distance & Correlation
Histogram distance & Correlation
Histogram distance
Feature Likelihood Fusion
Correlation
Correlation
Kernelized Correlation Filter
Correlation-Based
Region
Intensity
Edge
Gradient Orientation
Online Correlation based Template Update

Figure 1.3: Single object tracking pipeline. The red borders show our contributions.

color depending on the availability and reliability of the color attribute. We use Bhattacharyya distance [67] to measure color dissimilarity between the target and the surrounding area.

- **Frame level max-pooling scale selection:** we extended C-LoFT by proposing a novel frame-level scale selection. CS-LoFT, the proposed frame-level scale selection, ensures scale coherence in the selected likelihood map (all pixels corresponding to the same scale), robustness against local outliers, and adaptation to scale changes caused by object motion, camera motion, or zoom.

- **A multi-dimensional kernelized correlation filter module:** we extend the original LoFT framework with a kernelized correlation filter (KCF), a discriminative module, to strengthen LoFT by enabling robust and efficient localization of targets and strength adaptation to environmental changes. KC-LoFT also incorporates an online template update scheme to prevent drifts and to enable robust handling of partial or full occlusions. Online template updating includes learning filter parameters and updating the template for each frame, if necessary, by including positive and negative examples within the search window using ridge regression. We utilized Peak-to-sidelobe ratio (PTSR) [12] which evaluates the discrimination power of a likelihood peak to avoid updating the correlation-based template and regression parameters during occlusion and to prevent the correlation likelihood map to be fused with the remaining maps when it is unreliable. During occlusion events, PTSR helps suspend correlation template update until the object appears again.

- **Implemented on different datasets and participated in different challenges:** we participated in the VOT16 challenge [68], implemented on VOT15 [59], VOT16 [68] benchmarks, and ABQ WAMI dataset [1] to evaluate our contribution.

1.3.2 Muli-object Tracking Contributions

Our proposed multi-object tracker is named Multi-cue Multi-target Tracker M2Track. M2Track is a lightweight, efficient modular design multi-object tracking system Figure 1.4. M2Track is a Detection-Tracking-Prediction-based multi-object tracking system for sized video objects with three-level data association, shown in Figure 1.5. M2Track combines a fast spatial distance only short term data association, a robust tracklet linking step using discriminative object appearance models, and an explicit occlusion handling module relying not only on the motion of tracked objects' patterns but also on environmental constraints such as the presence of potential occluders in the scene. M2Track cascaded modular design ensures high performance in terms of time efficiency, tracking accuracy, robustness to environmental challenges, and invariance to camera acquisition types.

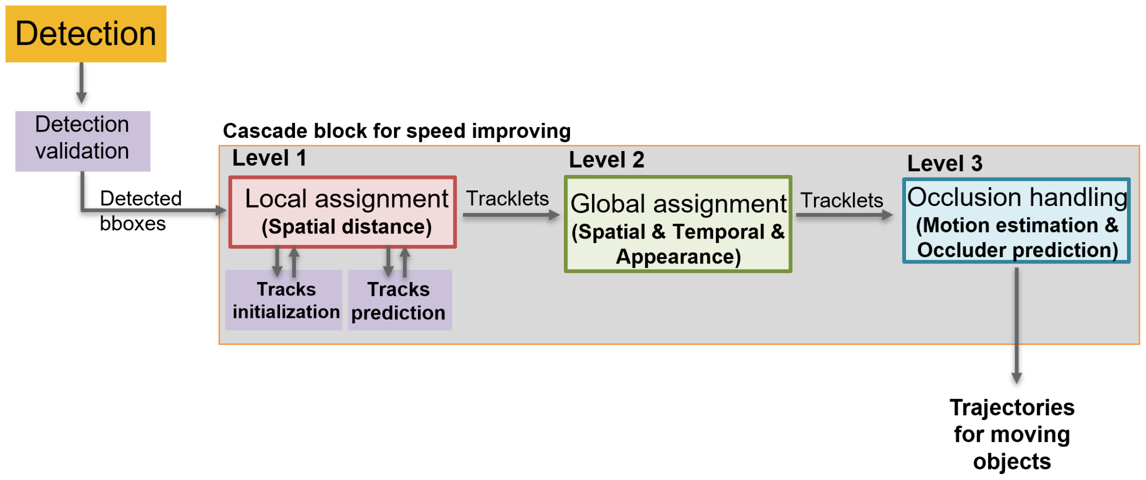

Detection
Detection validation
Detected bboxes
Cascade block for speed improving
Level 1
Local assignment (Spatial distance)
Tracks initialization
Tracks prediction
Tracklets
Level 2
Global assignment (Spatial & Temporal & Appearance)
Tracklets
Level 3
Occlusion handling (Motion estimation & Occluder prediction)
Trajectories for moving objects

Figure 1.4: Multi-cue Multi-target Tracker M2Track, a multi-step cascaded data association multi-object tracking pipeline.

The main contributions of M2Track pipeline are:

- **Multi-cue object detection:** detection step in our pipeline is datasets dependent. The datasets and their context affect the instance/object detection methods that are used. For instance; adaptive k-means [69] was utilized; Deep learning object detection using either trained or pre-trained models using transfer learning has been used to fit with different dataset requirements and demands (i.e. Faster RCNN [32], Mask RCNN [70], YOLO [33], DMNet [35]); Motion-based detection using flux tensor spatio-temporal operator [71, 72] was exploited; Multi-cue object detection

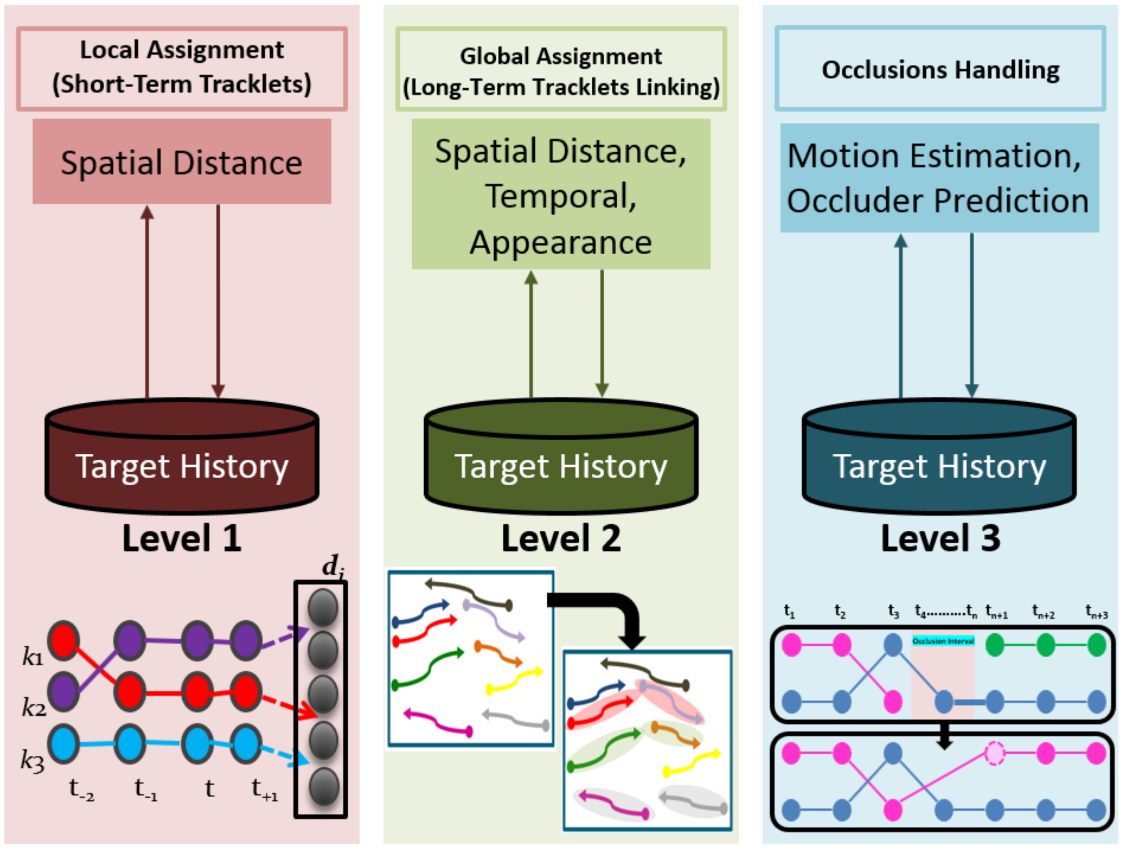

Local Assignment (Short-Term Tracklets)
Spatial Distance
Target History
Level 1
Global Assignment (Long-Term Tracklets Linking)
Spatial Distance, Temporal, Appearance
Target History
Level 2
Occlusions Handling
Motion Estimation, Occluder Prediction
Target History
Level 3
d_j
k_1
k_2
k_3
t_{-2} t_{-1} t t_{+1}
t_1 t_2 t_3 t_4.........t_n t_{n+1} t_{n+2} t_{n+3}
Occlusion Interval

Figure 1.5: The three-step cascaded data association scheme

was proposed [73], which fuses both appearance and motion-based detections in a complementary manner using deep learning combined with flux tensor spatio-temporal filtering. The proposed multi-cue detection is able to detect moving objects with high precision and recall, while filtering out false positives such as stopped objects, through intelligent fusion.

- **Multi-object tracking cascade with multi-step data association:** we proposed a tracking-by-detection approach using a three-level cascaded data association: start with short-term (level1) spatial data association using linear assignment with the Hungarian optimization algorithm [74]. Followed by a long-term (level2) trajectory-level tracklet linking module to robustly re-link fragmented tracklets. Robust re-linking of trajectories depends on several constraints for filtering unfeasible tracklets by taking into account tracklets' spatial, kinematics, and the history of objects' motion. For the occlusion handling module (level3), Kalman filter prediction, tracking objects' motion patterns, and environmental constraints (i.e., surrounding objects and potential occluders) clues are used to support the occlusion module for robust occlusion event detection. Figure 1.5 illustrates the proposed three levels data association scheme.

- **Local assignment short-term tracklets (level1):** low-cost local data association operating at object level on consecutive frames relying only on spatial distance information. The local assignment is used to have a time-efficient short-term but reliable tracklet generator which represents the backbone of the trajectories for further processes (e.g., re-linking, correction, switching ID). Hungarian optimization algorithm [74] is utilized to assign the detected objects in the current frame with the tracks, that are already generated from previous frames. A hybrid cost matrix for frame-to-frame during data association is used to be invariant to wide cross-domain datasets. In this step during the assignment, birth, death, appearance, and the disappearance of objects in the scene are handled.

- **Track position prediction:** Kalman filter [75] with a constant velocity model is used to predict the new positions of the tracked objects using their past trajectories. Velocity vectors for each target are stored to be used in the following steps for tracklets re-linking and occlusion handling. The process involves prediction and update steps. First, the positions of the targets are estimated from their current states, then Kalman filter updates the target states using the detections from the current frame.

- **Hybrid cost matrix:** A hybrid cost matrix during frame-to-frame data association is used to be invariant to wide cross-domain datasets. Three types of special information have been exploited. Centroid distance matching is helpful with objects that have large movement displacement due to low frame rates (i.e., moving objects on wide aerial imagery). The second uses bounding boxes overlapping matching that is helpful with objects that have low displacement movement. (i.e., cells with very slow motions). And, mask overlapping matching which is helpful with objects that have an irregular shape or fast deformable nature.

- **Global assignment long-term tracklets linking (level2):** the global data association is a trajectory-level tracklet linking module that robustly re-links fragmented tracklets. Linking objects in this level are treated as trajectories rather than detected objects to reduce the number of hypotheses and reduce computational cost. Robust re-linking of trajectories depends on several constraints for filtering unfeasible tracklets by taking into account tracklets spatial, kinematics, and the history of objects' motion. Besides that, robust and discriminative object appearance models (i.e., novel color correlation cost matrix, object texture, deep feature descriptor using Siamese network) are used.

- **Robust appearance model for long-term tracklet linking:** the appearance models of tracked objects provide powerful information to refine the set of candidate matches. Our appearance model can discriminate different tracked objects while being invariant against factors such as illumination, shadow, etc. We propose an appearance model that combines shape and texture information using HoG descriptor [76] with object color attributes using our novel color correlation cost matrix. And, Deep features for matching similarity using a trained Siamese network [77]. During matching, mean square error (MSE) is used to compute the distance between HoG descriptors for two potential matching. Earth mover distance [78] is used to compute the distance between two color histograms. For deep features, the comparison is done by calculating the Sigmoid function for each object to find the probability of the similarity between two objects.

- **Occlusions Handling:** to recover the tracker from miss detecting resulting from detection algorithms errors or occluding behind other objects in the scene, we utilize an occlusion handling module that relays on Kalman filter prediction, tracked objects' motion patterns, and environmental constraints (i.e. presence of potential occluders) to build occlusion confidence map. Occlusion confidence maps predicted the locations of the tracked objects after the occlusion event ends. Occlusion confidence maps are generated by combining the tracklet's motion model, occlusion confidence, and motion prediction range.

- M2Track is a detection-to-track tracker that exploited image appearance-based detection and associations approach using objects' visual appearance, masks, and bounding boxes. M2Track is scalable to thousands of detections and runs in near real-time for its efficiency designed in a modular system. M2Track handles rigid and deformable objects by taking into account the outcome of detection variations (i.e., object coordinates, object bounding boxes, object masks). M2Track has been integrated into different approaches and applications:

  – Moving vehicle detection and tracking: on wide aerial imagery datasets (i.e., ABQ [1], CLIF [2]), drones-based videos (VisDrone [79], UAVDT [80]), and fixed-camera videos (UA-DETRAC [60], FPSS [81]).

  - Cell segmenting, detection and tracking (i.e., cell tracking with mitosis detection dataset CTMC [82], cell segmentation and tracking challenge dataset [64]).
  - Video annotation and video analysis (i.e., biomedical cell annotation, surveillance moving object annotation).

- M2Track shows its high performance by participating in different international multi-object tracking challenges. M2Track has outperformed or comparable performance compared to the different state-of-the-art participated trackers including: UA-DETRAC [60], MOT16 [4], PETS09 [61], VisDrone[62], cell segmentation and tracking challenge [64], and CTMC [82] dataset)

Algorithms and methods that are presented in this dissertation are published as our contributions in [83, 84, 85, 86, 87, 88, 89, 90, 73, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104]. Chapter 2 explains our extension and contributions to single object tracking and shows some practical implementation on different datasets. Chapter 3 presents M2Track, the multi-object tracking pipeline. Chapter 4 presents the challenges, implementation, and experimental analysis on aerial imagery datasets in terms of high and low altitude video acquisitions. Chapter 5 describes the implementation of our multi-object tracking pipeline on biomedical video benchmarking. Chapter 5 shows our implementations and experimental results on the ground-based/ real-world videos. Followed by the conclusions for our contribution and discussion of future directions.

Chapter 2

Single Object Tracking Pipeline

2.1 Overview

Visual object tracking is one of the most essential problems in computer vision. Visual object tracking is the process of estimating the locations of an object over time. Most of the visual object tracking algorithms follow a similar pipeline that can be divided into three modules: initialization, localization, and update. Initialization is the process of generating an object model by extracting the features from the predefined target of interest from the first frame. That model is used to localize and detect the target on the subsequent frames. The localization module is the process of detecting the object on the following frames by finding the most relevant match with the object model created in the initialization module step. The update module is the process of updating the object model with a new feature of the new target located in the new frame. This step is very important to have a robust tracker to make the tracker adaptive to scene variations. Despite all the recent advancements in computer vision, visual tracking remains to be a challenging task because of real-world sequence challenges such as partial or full occlusions, background clutter, shadow, and other illumination artifacts.

Recently our group proposed a recognition-based single object tracking system LoFT (Likelihood of Features Tracking) [65, 27, 66] that relies on the fusion of multiple sources of information about the target and its environment to perform robust single object tracking. Originally LoFT was developed to track objects on wide aerial motion imagery (WAMI) acquired by an airborne camera sensor array where a large field-of-view undergoes persistent observation. These sequences involved small support regions for tracked objects and thus did not allow the use of detailed appearance descriptions. However, full-motion video datasets like OTB [105], ALOV [106], and VOT2016[59], include richer appearance information for tracked objects. These objects also undergo large-scale changes due to either object or camera motion, or camera zoom. In this chapter, we extend and improve LoFT by incorporating:

- Color information using our extension of *Color Name (CN)* scheme
- An operation mode decision unit to activate color use when there is an adequate color dissimilarity between the tracked object and its surrounding region
- Frame level max-pooling scale selection
- A discriminative module using kernelized correlation filters to strengthen LoFT's adaptation to environment changes
- A multi-dimensional kernelized template (beside LoFT's original multi-feature

template) to ensure comprehensive localization of the target in different scenarios

- A decision module to adaptively update and fuse kernelized template

2.2 Likelihood of Features Tracking (LoFT) Framework

Likelihood of Features Tracking (LoFT) [65, 27, 66] consists of three main modules: (i) target modeling, (ii) likelihood fusion; and (iii) appearance update. *Target modeling* models the appearance of the tracked object using a rich feature set including intensity, edge, shape, and texture information. Gradient magnitude, gradient orientation, intensity, shape, and curvature indices (derived from eigenvalues of the Hessian matrix) are some of the features used. Figure 1.3 describes the features that LoFT uses. LoFT uses a recognition-based target localization approach. For each feature in the feature set, a likelihood map is estimated by comparing the target's feature histogram to sliding window feature histograms within the search region. Efficient sliding window histogram computation is performed using integral histogram-based implementation [107]. *Likelihood fusion* combines likelihood maps from different features into a single likelihood map. The maximum likelihood location in this map is used for target localization. Likelihood fusion enables the adaptation of the tracker to dynamic environment changes and target appearance variabilities. Each feature performs differently depending on the target characteristic and environmental situations during tracking. Two weighting schemes are considered, Variance Ratio (VR) [108] for histogram-based features, and Distractor Index [66] for correlation-based features. *Appearance update* enables dynamic appearance adaptation by maintaining and updating a single template by calculating affine changes in the target to handle orientation and scale changes [27].

2.3 C-LoFT: Extension of LoFT with Color Attributes

Most modern trackers ignore color information and rely purely on features derived from grayscale information [109, 110, 111]. Grayscale images are sometimes sufficient to produce reasonably good tracking results for a lower computational cost. However rich chromatic information can be very valuable for object tracking. Various methods have been proposed to incorporate color information into trackers such as simple color space transformations in [112, 113] or color histograms in [114]. More recently, well-performing trackers have been further improved with the incorporation of color information. In [115] Centinet et al. combined color histograms with Minimum Output Sum of Squared Error (MOSSE) correlation filter [12] as a robust descriptor for object tracking. Danelljan et al. [116] explored the contribution of color in tracking-by-detection frameworks and extended CSK tracker [117] with various color incorporation approaches. While color introduces rich information for object description, color measurements can vary significantly over an image sequence due to variations in illumination, shadows, and specular reflections. Invariance to these factors has been studied in image classification [117, 113], action recognition [117], and tracking [116].

2.3.1 Adaptive Color Use with Color Names (CN) Feature

We extend the original LoFT framework with color information using *Color Names (CN)* feature [118]. The linguistic study described in [119] shows that the English language has eleven basic color terms: black, blue, brown, gray, green, orange, pink, purple, red, white and yellow. [118] explores this concept to generate CN (Color Names) map. *Color Names (CN)* feature associates the RGB color model with linguistic color labels. We use the mapping provided by [118] to map RGB color models to 11 color names. The process reduces the 256^3 RGB color space to 11 color names space. The goal is to boost the performance of the tracker, particularly for video sequences where foreground and background have distinct color signatures. Color information is used in the tracking process only when the tracked object and its surroundings have distinct color features. We use Bhattacharyya distance [120] to measure color similarity between the target and the surrounding area. Let H_{obj} and H_{bg} be $11 - bin$ color names histograms for tracked object and surrounding background respectively. Discrete probability densities $p(i)$ for the object, and $q(i)$ for the background, are computed by normalizing each histogram by corresponding number of elements in it:

$$p(i) = \frac{H_{obj}(i)}{n_{obj}} \tag{2.1}$$

$$q(i) = \frac{H_{bg}(i)}{n_{bg}} \tag{2.2}$$

Bhattacharyya distance between object versus background color distributions is computed as:

$$S = \sum_{i=1}^{11} \sqrt{p(i)q(i)} \tag{2.3}$$

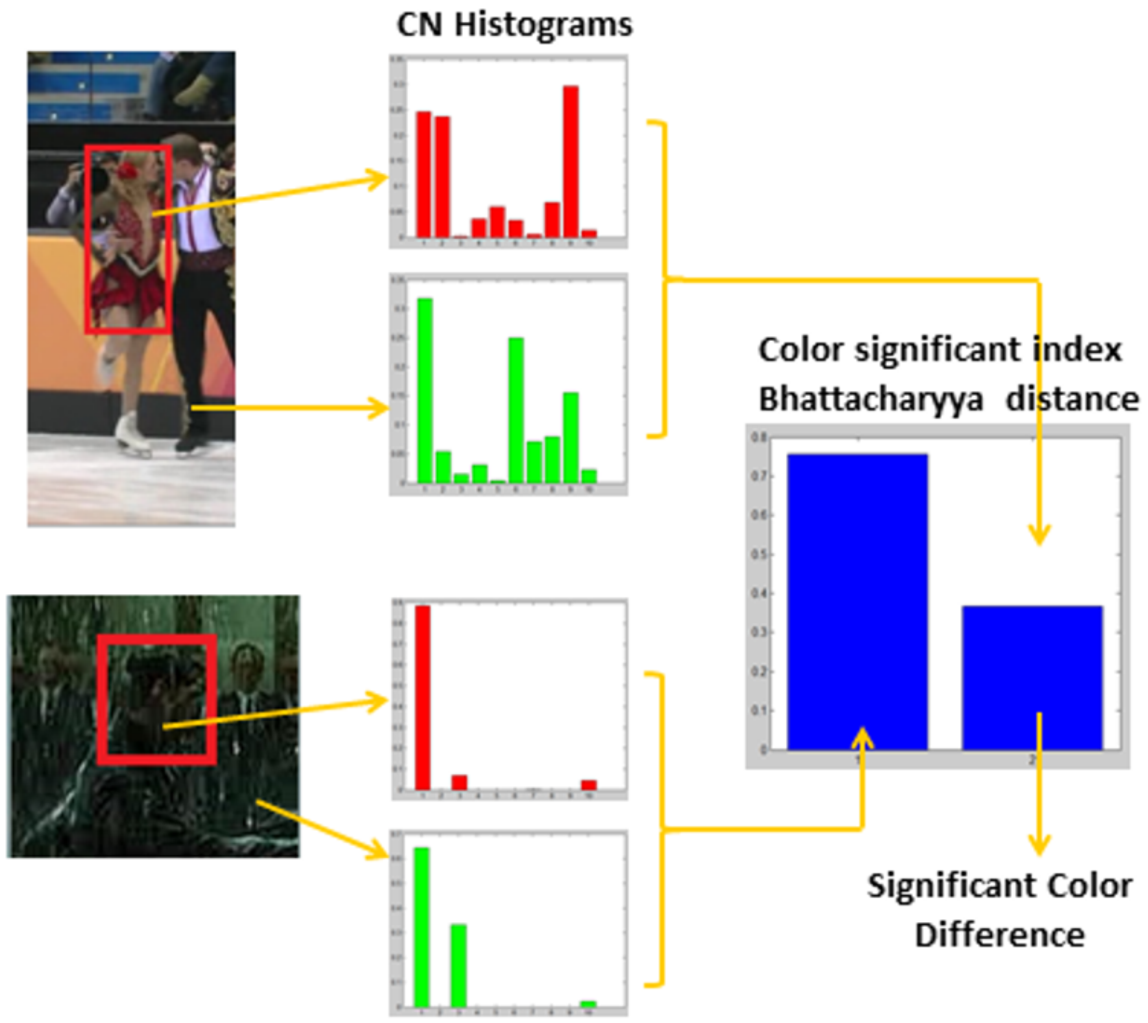


Figure 2.1: Two examples from different sequences of VOT15 showing different values of S (similarity measures) between target and surrounding region histograms.

Inclusion (or exclusion) of color features is determined for each sequence based on the value of S. Figure 2.1 shows the process of calculating S for two different sequences. Low values of S correspond to discriminant color information (dissimilar object versus background color distribution). Figure 2.2 illustrates the incorporation of color information in C-LoFT.

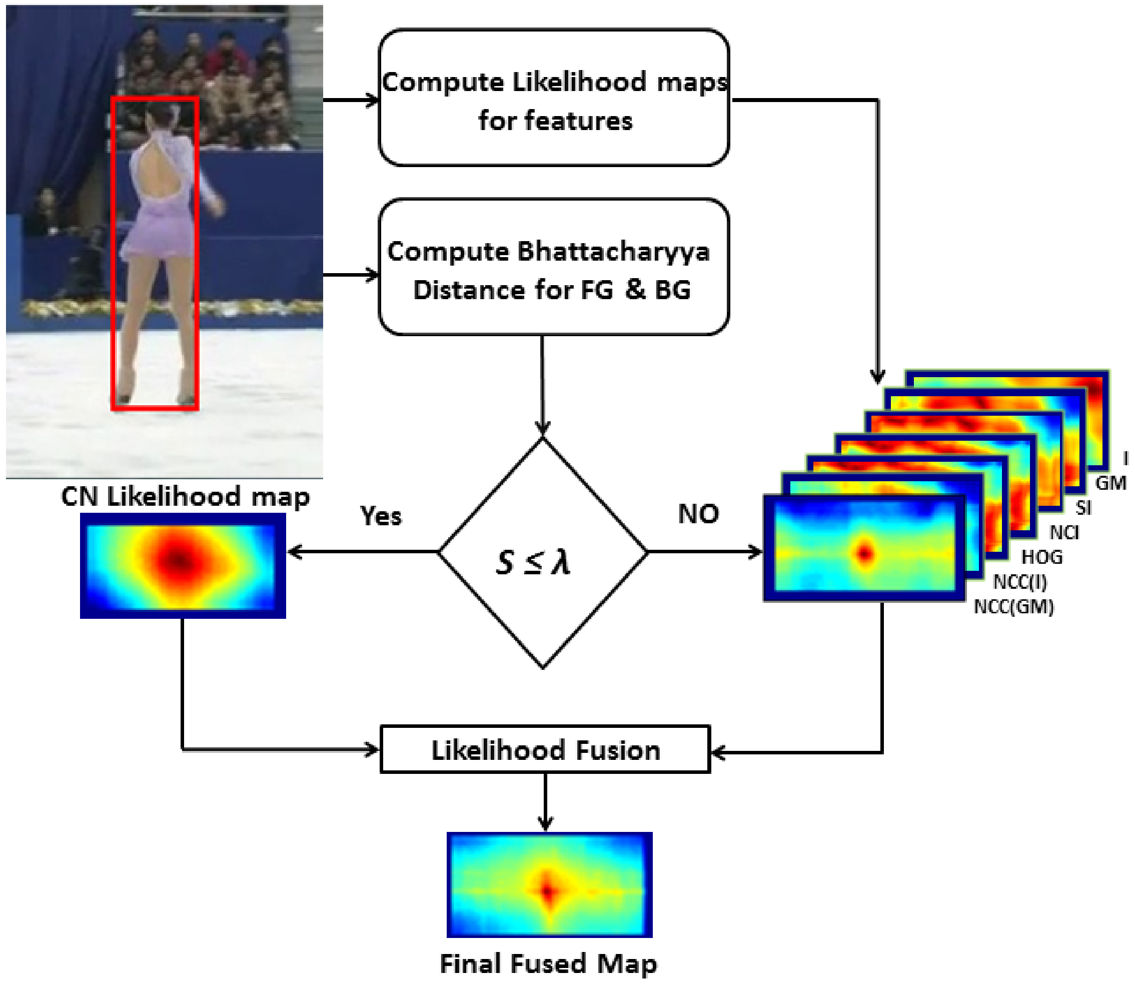


Figure 2.2: Incorporation of color information in C-LoFT. In which the likelihood map of I represents: Intensity histogram, GM: Gradient magnitude histogram, SI: Shape index histogram, NCI: Normalize curvature index histogram, HoG: Histogram of gradient, NCC(I): Normalized cross-correlation for intensity, and NCC(GM): Normalized cross correlation for Gradient.

2.4 CS-LoFT: Scale Space Tracking

When tracked objects move along the camera axis, their sizes in the video change. Many trackers such as CSK[117], or MOSSE[12] track the objects of interest at a single scale and ignore possible changes in size. This implies poor performance in sequences with significant scale variations. Incorporation of multi-scale templates as in DSST (Discriminative Scale-Space Tracking) [31] improves tracking performance through scale changes.

CS-LoFT extends C-LoFT with multi-scale processing. Scale changes not only alter the size of the tracked objects but also affect the scale of their texture. LoFT features that are most sensitive to scale changes are local shape features *local shape index* (Eq.2.6) and *normalized curvature index* (Eq.2.7) [66] derived from eigenvalues of Hessian matrix. The Hessian matrix H describes the second-order structure of local intensity variations around each image point $L(x; y)$,

$$H(x,y;\sigma) = \begin{bmatrix} L_{xx}(x,y;\sigma) & L_{xy}(x,y;\sigma) \\ L_{xy}(x,y;\sigma) & L_{yy}(x,y;\sigma) \end{bmatrix} \tag{2.4}$$

where $L_{xy} = \frac{\partial^2 L}{\partial x \partial y}$, $L(x, y; \sigma)$ refers to the Gaussian smoothed intensity image

$$L(x,y;\sigma) = g(x,y;\sigma) * L(x,y) \tag{2.5}$$

g is the Gaussian kernel, and σ is the scale. Local shape index (Eq. 2.6) and normalized curvature index (Eq. 2.7) are derived from the eigenvalues $\lambda_1 \geq \lambda_2$ of Hessian matrix H.

$$\phi(x,y) = \tan^{-1} \frac{\lambda_2(x,y)}{\lambda_1(x,y)} \tag{2.6}$$

$$\theta(x,y) = \tan^{-1} \frac{((\lambda_1(x,y)^2) + (\lambda_2(x,y)^2))^{\frac{1}{2}}}{1 + L(x,y)} \tag{2.7}$$

2.4.1 Frame Level Max-pooling Scale Selection

We propose a novel frame level scale selection as described in Algorithm 1. First local shape (Eq. 2.6) and normalized curvature (Eq. 2.7) indices are computed at multiple scales. Sliding window template matching is performed between search window and tracked object template; and, likelihood maps $\mathcal{L}_{\sigma_i}$ are obtained for each scale σ_i. Then, pixelwise maximum likelihood is computed as:

$$\mathcal{L}_{max}(x, y) = \underset{\sigma_i}{max}(\mathcal{L}_{\sigma_i}(x, y)) \tag{2.8}$$

Finally, the best scale σ^* is then selected as the scale that minimizes the mean square error between the pixelwise maximum likelihood $\mathcal{L}_{max}$ and each $\mathcal{L}_{\sigma_i}$.

$$\sigma^* = \underset{i}{argmin}(\sum_{x,y}(\mathcal{L}_{max} - \mathcal{L}_{\sigma_i})) \tag{2.9}$$

The described frame level max-pooling scale selection approach is applied to the shape index and normalized curvature index separately. Figure 2.3 and Algorithm 1 describe frame level max-pooling scale selection approach. Likelihood maps corresponding to selected scales are fused with likelihood maps for the remaining features. Proposed frame level scale selection ensures (i) scale coherence in the selected likelihood map (all pixels corresponding to the same scale); and (ii) robustness against local outliers.

Algorithm 1 Frame level max-pooling scale selection

```
Input : $I_{SW}, I_T$: search window and object template
Output : $\sigma^*, \mathcal{L}^*$: best scale and corresponding likelihood map
1: for each scale i do
2:     $L(x, y; \sigma_i) \leftarrow g(x, y; \sigma_i) * I(x, y)$
3:     Compute Hessian matrix $H(x, y; \sigma)$ Eq. 2.4 and its eigenvalues $\lambda_1$,$\lambda_2$
4:     Compute local shape features $\phi(x, y)$ Eq. 2.6, $\theta(x, y)$ Eq. 2.7
5:     $\mathcal{L}_{\sigma_i} \leftarrow$ sliding window template matching between search window & object
       template
6: end for
7: Pixelwise maximum likelihood:$\mathcal{L}_{max}(x, y) \leftarrow \underset{\sigma_i}{max}(\mathcal{L}_{\sigma_i}(x, y))$
8: Best frame level scale:$\sigma^* \leftarrow \underset{i}{argmin}(\sum_{x,y}(\mathcal{L}_{max} - \mathcal{L}_{\sigma_i}))$
```

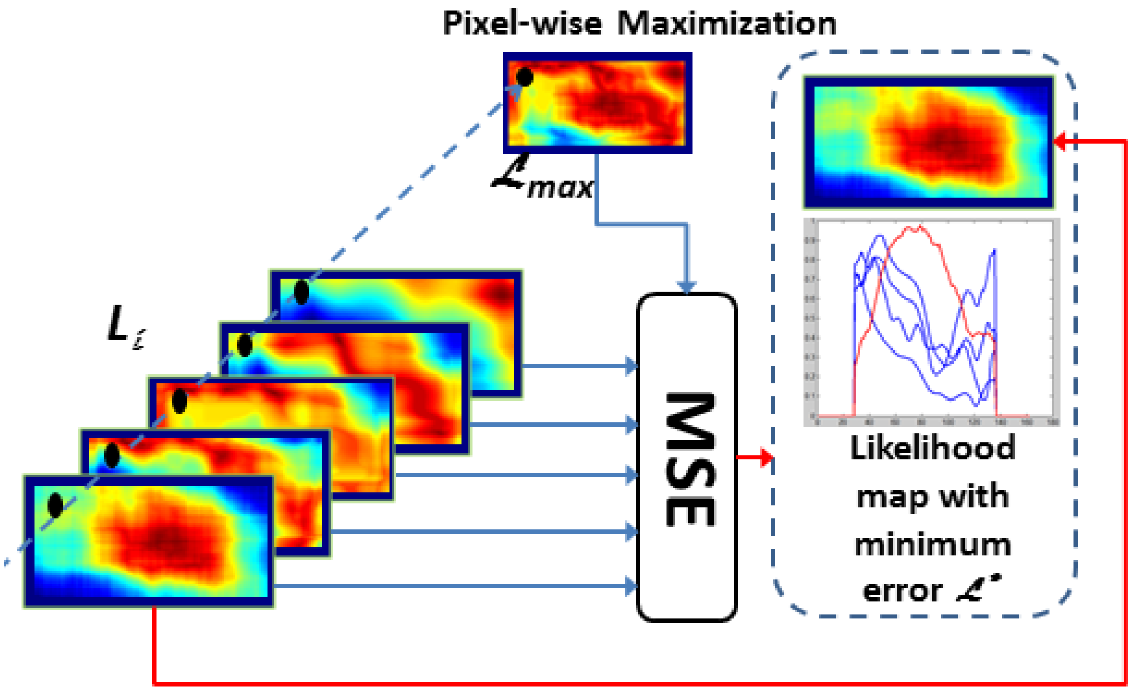


Figure 2.3: Frame level max-pooling scale selection approach.

2.5 KC-LoFT: Likelihood of Features Tracking with Kernelized Correlation

Correlation filter-based tracking is a discriminative tracking method in which a filter based on an object's appearance is used to estimate the most likely target location in the search window by distinguishing the target from its surrounding background. Convolution of the search window image with the filter is performed in Fourier domain as element-wise multiplications to reduce complexity. The used filters are updated online to adapt to object appearance changes. Since their first use in visual tracking [12] [30], correlation filter-based tracking has been extended in multiple ways. [11][31] improved adaptation to scale changes; while [22] incorporated histogram-based ridge regression learning to improve robustness to fast deformation and shape variations.

2.5.1 Kernelized Correlation Filter Module

Likelihood of Features Tracking (LoFT) system[27] fuses multiple sources of information about target and its environment to perform robust single object tracking. Target (template), to search window feature comparison is performed using cross-correlation and sliding window histogram differencing using an efficient integral histogram computation scheme. The process produces a likelihood map for each individual feature. LoFT appearance adaptation scheme maintains and updates a single template by calculating affine changes in the target to handle orientation and scale changes [27]. LoFT target template update is performed continuously to ensure accurate target localization. However continuous template update has two main problems: i) drift when target template is updated with non-target features during partial or full occlusions. ii) delayed recovery after occlusion events (many update steps are needed to relearn target appearance).

The original LoFT framework is extended with kernelized correlation filter (KCF) [30] module. The contribution on adding this module are:

1. A discriminative module to strengthen LoFT's adaptation to environmental changes.

2. A multi-dimensional kernelized template (beside LoFT's original multi-feature template) to ensure comprehensive localization of the target in different scenarios.

3. A decision module to adaptively update and fuse kernelized template. The new decision module uses peak-to-sidelobe ratio (PTSR) criterion to avoid fusing any irrelevant response from the kernelized correlation filters to the other LoFT feature maps and to prevent the update of the regression parameters and correlation-based template during occlusion or other cases of sudden appearance change.

KC-LoFT kernelized correlation module uses two features, HoG, and intensity, to localize the target within the search window. The two features are stacked to form a single vector x that is then used in the kernelized correlation filter (KCF) scheme. The goal of the correlation filter is to minimize the square error over the sample x and the expected target y using the ridge regression loss function defined as follow

$$\min_{w} \sum_{i}^{n} (f(x_i) - y_i)^2 + \lambda \|w\| \tag{2.10}$$

where n is the length of the feature vector and λ controls the regularization parameter w over the linear combination of samples $f(x) = w^T x$. Following the description and derivatives on [30] regression learning parameter, α can be learned using

$$\hat{\alpha} = \frac{\hat{y}}{\hat{k}^{xx} + \lambda} \tag{2.11}$$

where $\hat{k}^{xx}$ is a correlation kernel. Gaussian kernel is used as follow

$$k^{xx'} = \exp(-\frac{1}{\sigma^2}(\|x\|^2 + \|x'\|^2) - 2F^{-1}(\hat{x} \odot \hat{x}'^*)) \tag{2.12}$$

Where $\hat{x}$ is the DFT of x, $\hat{x}'^*$ is the complex-conjugate of $\hat{x}'$, $\odot$ is an element-wise multiplication, and $\hat{}$ is the Discrete Fourier Transform. To detect the position of the object, a new patch z (search window) is cropped from the location estimated from Kalman filter. The response is found according to the correlation between the previously learned template $\tilde{x}$ and new patch z.

$$\hat{f}(z) = (\hat{k}^{\tilde{x}z})^* \odot \hat{\alpha} \tag{2.13}$$

The steps of the target detection and online training process are described in Algorithm 2, where θ is the learning rate assumed to be 0.1.

2.5.2 KC-LoFT: Integration of Modules (Online Updating, Fusion)

KC-LoFT integrates the discriminative KCF module to the LoFT framework to enable robust and efficient localization of targets. Algorithm 2 describes the steps for using

Algorithm 2 Target detection and training process using correlation filter module

```
Input : $P_{f-1}, z, \alpha_{f-1}, \tilde{x}$: predicted position, current patch (search region), dual space
    coefficient, the previous updated correlation-based template
Output : $P_f, \alpha_f, \tilde{x}_{new}$: estimated new position, the updated dual space coefficient,
    updated correlated-based template
1: Calculate the kernel $k^{\tilde{x}z} = \exp(-\frac{1}{\sigma^2}(\|\tilde{x}\|^2 + \|z\|^2) - 2F^{-1}(\hat{\tilde{x}} \odot \hat{z}^*))$
2: Calculate the response $\hat{f}(z) = (\hat{k}^{\tilde{x}z})^* \odot \hat{\alpha}_{f-1}$
3: Find the new position $P_f$ from the maximum response $\hat{f}(z)$
4: Crop new patch $x_{new}$ with $P_f$ as a center of the region
5: Update the template $\tilde{x}_{new} = \theta\tilde{x} + (1-\theta)x_{new}$
6: Calculate $\hat{\alpha}_{new} = \frac{\hat{y}}{\hat{k}^{\tilde{x}x_{new}} + \lambda}$
7: Update dual space coefficient $\alpha_f = \theta\alpha_{f-1} + (1-\theta)\alpha_{new}$
```

correlation filter module. KC-LoFT also incorporates an online template update scheme to LoFT to prevent drifts and to enable robust handling of partial or full occlusions. Figure 1.3 illustrates modules involved in the proposed KC-LoFT pipeline.

The correlation filter learns filter parameters and updates the template for each frame by including positive and negative examples within the search window using ridge regression. KC-LoFT fuses the responses from LoFT features with the correlation response from the correlation module to generate a final fused probability map that is used to localize the target. Because the correlation template update happens during the processing of every frame, an online update on the correlation-based template has a faster response to appearance changes than the appearance-based template. The response $\hat{f}(z)$ from the correlation filter module is fused with the other appearance-based template likelihood maps from LoFT to generate a final likelihood map to localize the expected position of the target. Peak-to-sidelobe ratio (PTSR) likelihood map evaluation criterion is used to avoid updating the correlation-based template and regression parameters during occlusion and to prevent the correlation likelihood map to be fused with the remaining maps when it is unreliable. Peak-to-

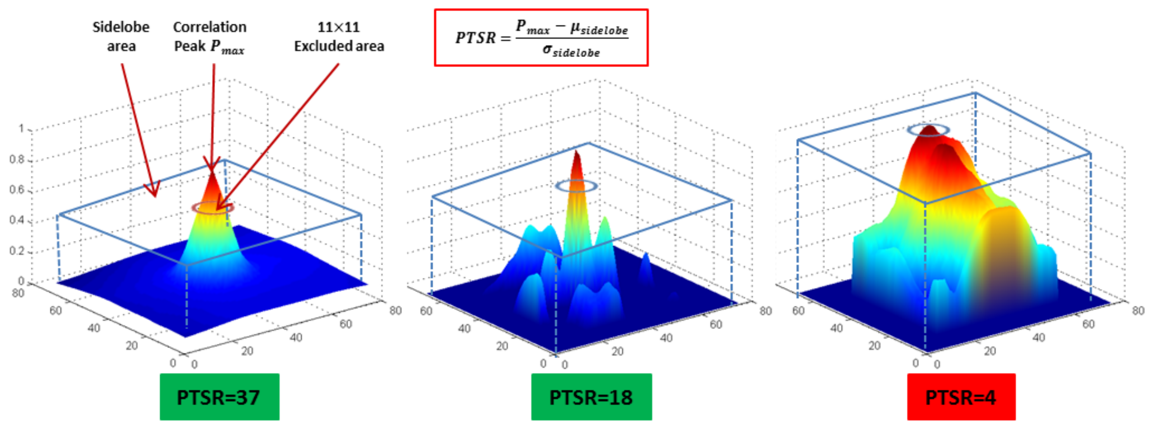


Figure 2.4: Sample likelihood maps and associated peak-to-sidelobe ratio values. Lower PTSR values indicate occlusions or other sudden appearance change scenarios and suspend the template update process.

sidelobe ratio [12] evaluates the discrimination power of a likelihood peak as follow:

$$PTSR = \frac{P_{max} - \mu_{sidelobe}}{\sigma_{sidelobe}} \tag{2.14}$$

where P_{max} is the value of the maximum response, $\mu_{sidelobe}$ and $\sigma_{sidelobe}$ are the mean and standard deviation of the likelihood map values except an 11×11 area around the maximum peak. Figure 2.4 shows sample likelihood maps and associated PTSR measurements.

During occlusion events, PTSR helps suspend correlation template update until the object appears again while the appearance-based template needs to be updated. When the object reappears after an occlusion event, it will be hard to localize the object again with the appearance-based template. Suspension of the template update process is important to preserve the target template through occlusion events. Figure 2.5 illustrates KC-LoFT template update and likelihood fusion processes.

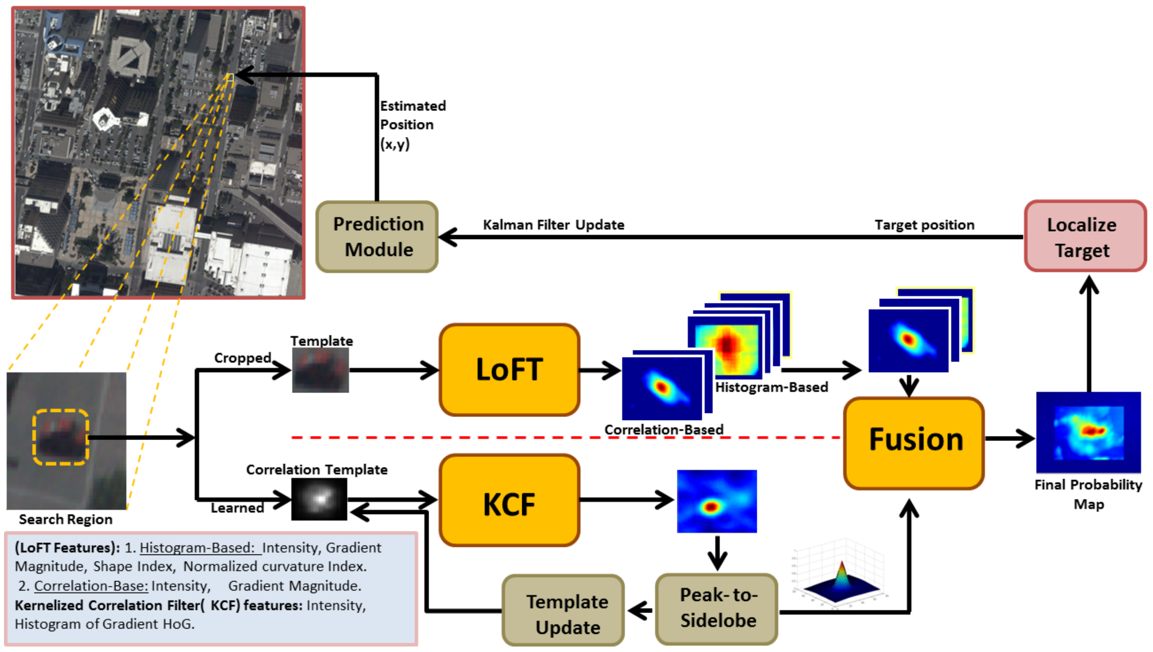


Figure 2.5: KC-LoFT template update and likelihood fusion pipeline.

2.6 Experimental Results for C-LoFT, CS-LoFT, and KC-LoFT

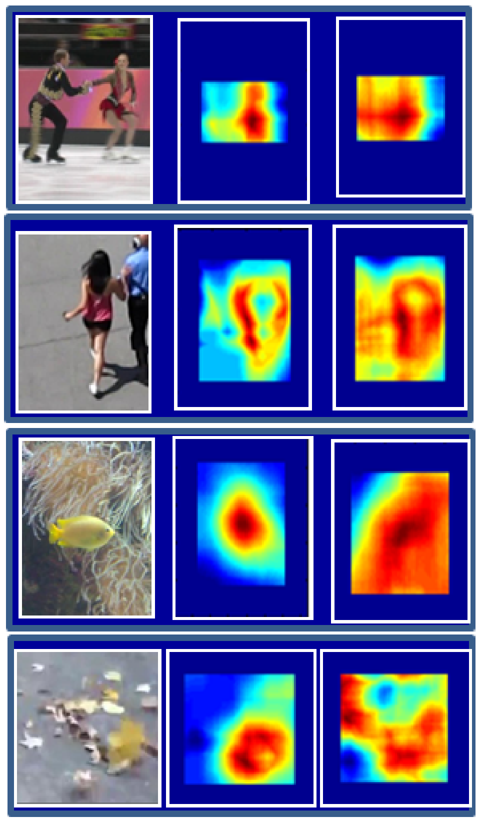
We have evaluated our contribution on VOT2015 benchmark dataset [121] and on ABQ dataset [1]. VOT15 contains 60 sequences with several visual attributes and challenges including: illumination changes, scale variations, motion change, camera motion, and occlusions. While ABQ is a wide aerial motion imagery video that suffers from extreme camera motion, low frame rate, and small object sizes. For evaluation, we adopted, *Accuracy* and *Robustness*, the two VOT15 evaluation metrics that are considered on the VOT challenge and described on their websites [122]. *Accuracy* measures how well the bounding box predicted by the tracker overlaps with the ground truth bounding box. *Robustness* measures how many times the tracker loses the target during tracking. A failure is indicated when the overlap measure becomes zero [121].

2.6.1 Color and Scale Features Extension Evaluation

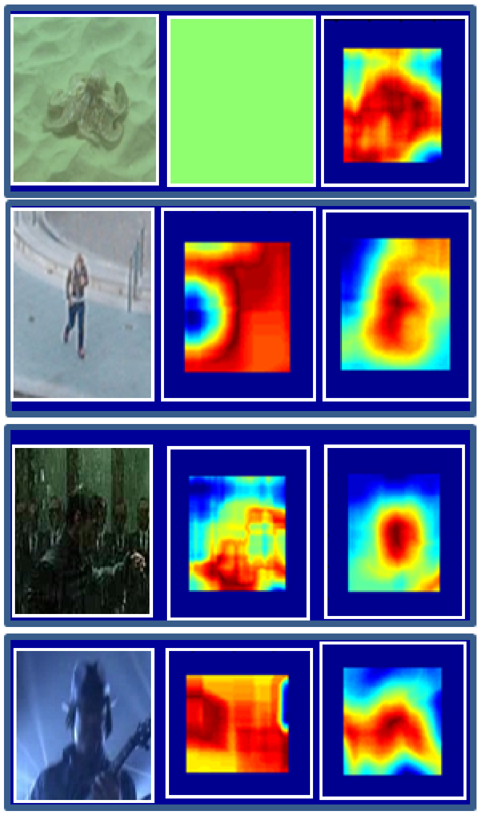
C-LoFT and CS-LoFT were implemented on VOT15 dataset. Figures 2.6 and 2.7 shows intermediate results related to color and scale processing in CS-LoFT. CS-LoFT switches to color tracking mode only when the tracked objects and their surroundings have different color features (figure 2.6a), otherwise relies on intensity (figure 2.6b). Figure 2.7 shows how much the proposed frame level max-pooling scale selection method preserves spatial context while pixel-wise max-pooling over scales generates distractors. Figure 2.8 shows the average robustness comparison between LoFT tracker with only CN feature, baseline LoFT, and CS-LoFT trackers. The comparison shows the significance of using color information and scale selection. Also, it shows that color information alone will not be helpful for a reliable tracker. Figure 2.9 shows detailed evaluation of LoFT and CS-LoFT on the all 60 video sequences. CS-LoFT improves the performance of the original LoFT tracker in terms of both *accuracy* and *robustness* by 12% and 21% respectively. Performance is particularly improved for the sequences that have significant color distractor and rapid scale changes.

2.6.2 Kernelized Correlation Module Evaluation

For KC-LoFT evaluation, ABQ dataset was used to evaluate the extended LoFT with the kernelized correlation filter module. For fair evaluation and comparison, some of the best non-deep learning correlation-based trackers whose codes are available publicly were used. Since 2013, VOT challenge group [122] has been organizing



(a)



(b)

Figure 2.6: Color versus intensity. (a) Tracked objects and their surroundings have different color features; (b) Tracked objects and their surroundings have similar color but different intensity features. Columns left to right: original image, likelihood map obtained using color names (CN) feature, likelihood map obtained using intensity feature. On (a), the tracker fuses the likelihood map for color while on (b) the tracker does not.

single object tracking challenges for selected full-motion video datasets. Table 2.1 shows VOT2015[121] and VOT2016[68] ranks of LoFT with some other state-of-the-art trackers used in this evaluation. All the listed trackers perform better than LoFT on these full-motion videos datasets. However, the nature of benchmark datasets can be one of the most important factors in tracker evaluation. LoFT and KC-LoFT are trackers developed for aerial wide aerial motion imagery. Wide aerial motion

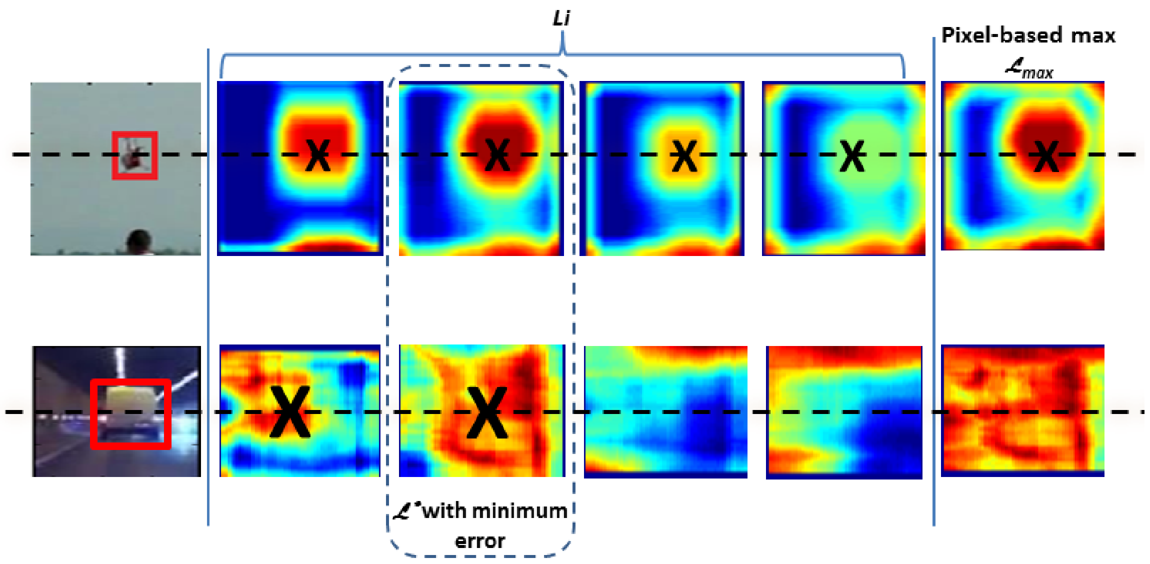


Figure 2.7: Frame level max-pooling best scale selection versus pixel-based max-pooling. The third column: the frame level $\mathcal{L}^*$ over all scales (best selection), while the sixth column: $\mathcal{L}_{max}$ pixel-based max-pooling result.

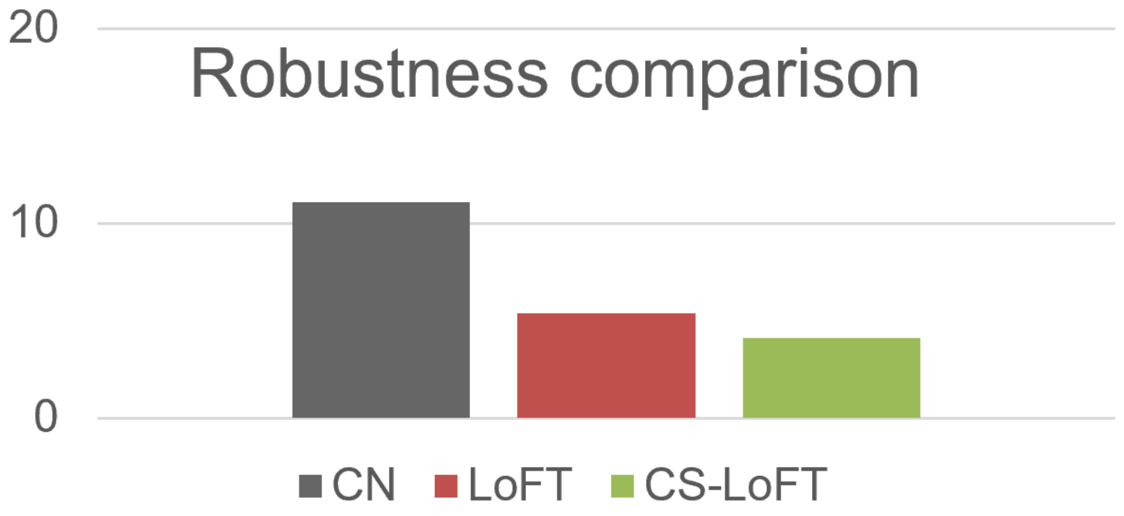


Figure 2.8: The average robustness comparison between LoFT tracker with only CN feature, baseline LoFT, and CS-LoFT tracker.

imaginary (WAMI) datasets have different characteristics and challenges compared to the regular full-motion videos. For tracker performance evaluation, we have used

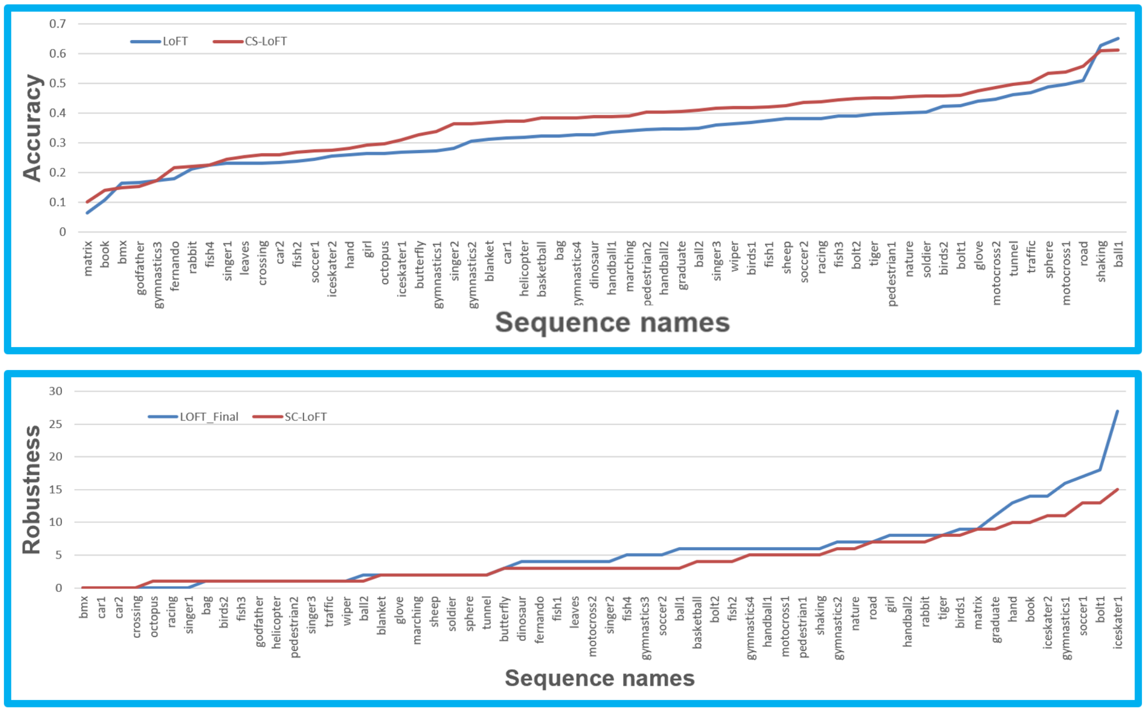


Figure 2.9: Performance evaluation on VOT2015 dataset: (Upper) accuracy, (Lower) robustness. Sequences are reordered by robustness and accuracy values respectively.

the same two metrics described in the previous section. Table 2.2 summarizes the tracking performances for the proposed KC-LoFT tracker and other state-of-the-art trackers on ABQ dataset. KC-LoFT increases both the accuracy and robustness of

Table 2.1: VOT2015 and VOT2016 rank the best non-deep learning trackers whose codes were made public. A lower rank is better. '-' means the tracker did not participate in the challenge. The ranks and results are described in detail on VOT15 [121] and VOT16 [68] challenge results reports.

Tracker Name	Rank on VOT2015	Rank on VOT2016
LoFT	#60	#68
STAPLE[22]	-	#5
SAMF [11]	#8	#24
DSST [31]	#38	#43
SRDCF [123]	#4	#17

Table 2.2: Tracker performance comparison on ABQ wide aerial motion imagery dataset.

Tracker Name	Rank↓	Accuracy↑	Robustness↓
SAMF[11]	2.5	0.63	0.91
KC-LoFT	2.5	0.59	0.81
SRDCF[123]	3	0.48	0.70
LoFT	3.5	0.54	0.85
DSST[31]	4	0.61	1.78
STAPLE[22]	5.5	0.49	1.18

the original LoFT tracker (by 9.6% and 5.1% respectively) and produces better or comparable results compared to the state-of-the-art trackers from the VOT2015[121] and VOT2016[68] challenges according to **Rank** metric which is the average ranking for evaluation metrics, accuracy, and robustness, used in this evaluation. Figure 2.10 shows sample tracking results for two cars. Trajectory color represents the number of re-initialization after tracker failures. The lower number of trajectory colors indicates tracker robustness. In both cases, KC-LoFT tracks the selected cars without any failures or restarts, while for the first car LoFT, DSST, and Staple require one or more restarts, and for the second car, all the trackers except KC-LoFT require one or more restarts.

2.7 Single Object Tracking Conclusion

We presented a recognition-based object tracking framework that extends Likelihood of Features Tracker (LoFT) with color attributes, scale selection, and kernelized correlation filter module producing CS-LoFT, KC-LoFT. The addition of color attributes improves object and background appearance description. Color information is incorporated through CN scheme. Automated feature scale selection adapts the tracker to

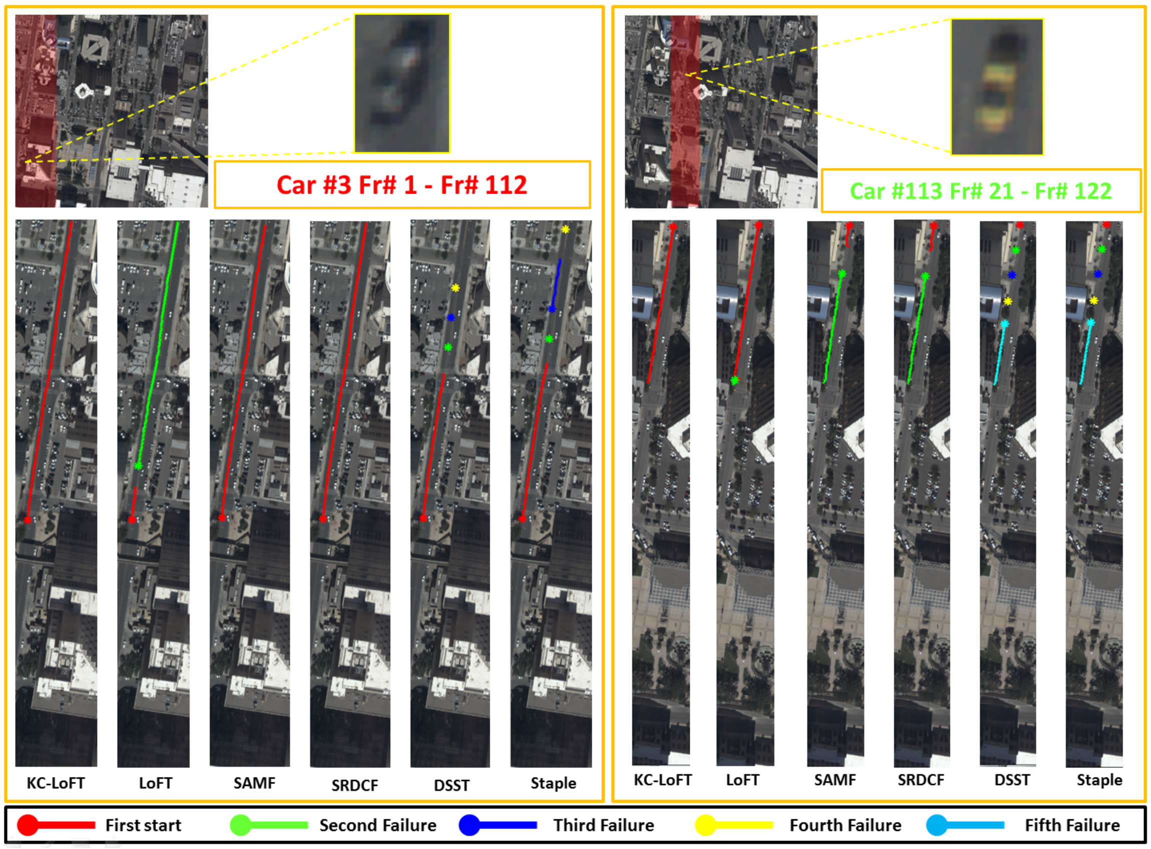


Figure 2.10: Comparison of tracking results in terms of robustness. Trajectory color represents the number of re-initialization after tracker failures. The lower number of trajectory colors indicates higher tracker robustness.

scale changes caused by object motion, camera motion, or zoom. The proposed scale selection method ensures scale coherence in the likelihood map and robustness against local outliers. Experimental results show improved performance, particularly for the sequences that have significant color distractors and rapid scene changes. KC-LoFT extends the Likelihood of Features Tracker (LoFT) framework with the kernelized correlation filter (KCF) scheme to better localize the target in a search window. KC-LoFT combines the LoFT strengths such as multiple complementary features robust to environmental conditions and shape deformations, with the KCF strength such as online template and parameter update and robustness to target shifts and rotations. KC-LoFT overcomes the LoFT and KCF limitations by improving the performance of the correlation filter with object location and search window prediction, and by suspending updates during occlusion events to keep the correlation-based template and parameters reliable. The use of appearance-based template helps KC-LoFT to better localize object during deformation, and online update helps adapt to environmental changes. Experimental results on wide aerial motion imagery show improved performance in terms of accuracy and robustness compared to LoFT and better or comparable results compared to other state-of-the-art trackers.

Chapter 3

Multi-Object Tracking Pipeline

3.1 Overview

Multi-object tracking (MOT) aims to locate multiple objects in a scene, maintain their identities in time, and form motion trajectories for further analysis. Multi-object tracking divides into two steps: *Detection step*, in which targets in each video frame are localized; *Association step*, where detected targets are assigned and connected to existing trajectories. The association process can be performed online [124, 125, 126] by only using information gathered from past frames, or offline (batch mode) [127, 128, 129] by exploiting information from the whole video including past and future frames. Some applications (i.e., online video surveillance, navigation, autonomous driving, etc.) require the use of online approaches because of their real-time nature, others can use online or offline approaches.

The performance of the tracking-by-detection methods is greatly influenced by the performance of the object detection methods used. False positives (false detections), false negatives (undetected objects), under-segmentation (merging of neighboring objects), and over-segmentation (object fragmentation) are major sources of tracking errors. Recent advances in object detection [54, 55, 56] are thus important for the success of the tracking task. Association links the detected objects in time. Tracking-by-detection frameworks formulate tracking as an object-to-track assignment problem. Local data association performs assignment, also can be called online association, considering information between adjacent frames [42, 43], whereas global data association, also called offline association, takes multiple frames into account [44, 45]. The local assignment is more sensitive to detection errors, tends to produce short fragmented trajectories (tracklets), and may cause drift under occlusion [57, 58]. Association is done by comparing object descriptors with suitable cost functions (similarity or dissimilarity measures). Features used to describe and compare objects can vary. The selection of discriminative features is very important in order to reduce association ambiguities. Features used in tracking should be able to discriminate different objects while being robust to factors such as illumination, viewpoint, pose, etc. changes. Appearance similarity of the detected objects is a challenge for the association process since it may cause matching ambiguities. It is important to integrate additional information such as scene structures [130], target context [131, 132, 133], or target state prediction [134, 135, 136] to resolve association ambiguities. However, these additional processes adversely affect the computational cost of tracking.

Our pipeline for multi-object tracking is called Multi-cue Multi-object Tracker (M2Track). M2Track is a tracking-by-detection framework proposed where results from an object detector are used as input to the tracker. M2Track consists of two main steps. *Detection step* which is datasets dependent in our pipeline, and *Association step*: a multi-step cascade data association applied on detected objects to ensure time efficiency and preserve accuracy through the steps, and many other intermediate steps that are shown in Figure 1.4 and will be discussed in details in next sections.

3.2 Multi-cue Object Detection

Advances in deep learning methods, sensor and GPU computing, sensor technologies, and training data collected for recent AI challenges [137, 138, 139, 140] have led to significant performance improvements in object detection accuracy and time efficiency. Researchers have used motion-based [141, 142, 56] or appearance-based approaches [143, 144] to address the challenges of object detection. Others have combined motion and appearance-based approaches for more robust performance [145, 146, 147].

Object detection is at the core of this section. For our multi-object tracking pipeline, a number of object detection approaches have been used. Selecting the detection approach depends on the dataset's nature. Supervised or unsupervised learning object detection (i.e., adaptive K-means), motion-based detection using flux tensor spatio-temporal operator [71, 72], deep learning object detection (i.e., Faster RCNN [32], Mask RCNN [70], YOLO [33], DMNet [35]) using trained or pre-trained models with transfer learning were implemented and utilized to fit with different dataset requirements and context. Additionally, our proposed work, which will be our goal in this section, a multi-cue object detection [73] has been used to detect the objects of interest to be tracked later in the further steps.

Despite the improvements, particularly on ground-based video analysis, moving object detection remains a challenging task in wide area motion imagery (WAMI) collected by drones. WAMI data presents unique challenges for video data fusion, object detection, and object tracking. Some of these challenges are illustrated in Figure 3.1 (a-c) such as small object sizes, object shape distortions, fast motion due to low frame rates, high densities of similar objects (i.e., parking lots full of cars, busy intersections with hundreds of vehicles and pedestrians), parallax, and, partial or full occlusions which can drastically affect object detection and tracking performance.

There are different sources of motion in an airborne vehicle tracking scenario including:

1. Motion of the drone platform itself
2. Motion of objects (i.e., vehicles and pedestrians) in the scene
3. Motion induced by parallax due to buildings and other tall structures in the scene

The platform motion can be eliminated by applying an efficient video registration technique to stabilize the video frames using [148]. Followed by a motion detection algorithm to identify moving objects, However, classification of real moving objects from parallax induced motions is a very challenging task in WAMI airborne video analysis.

In this section, we introduce a novel pipeline, shown in Figure 3.2, to identify true moving objects despite spurious detections by fusing deep appearance-based object detection with spatio-temporal tensor-based motion detection. The pipeline

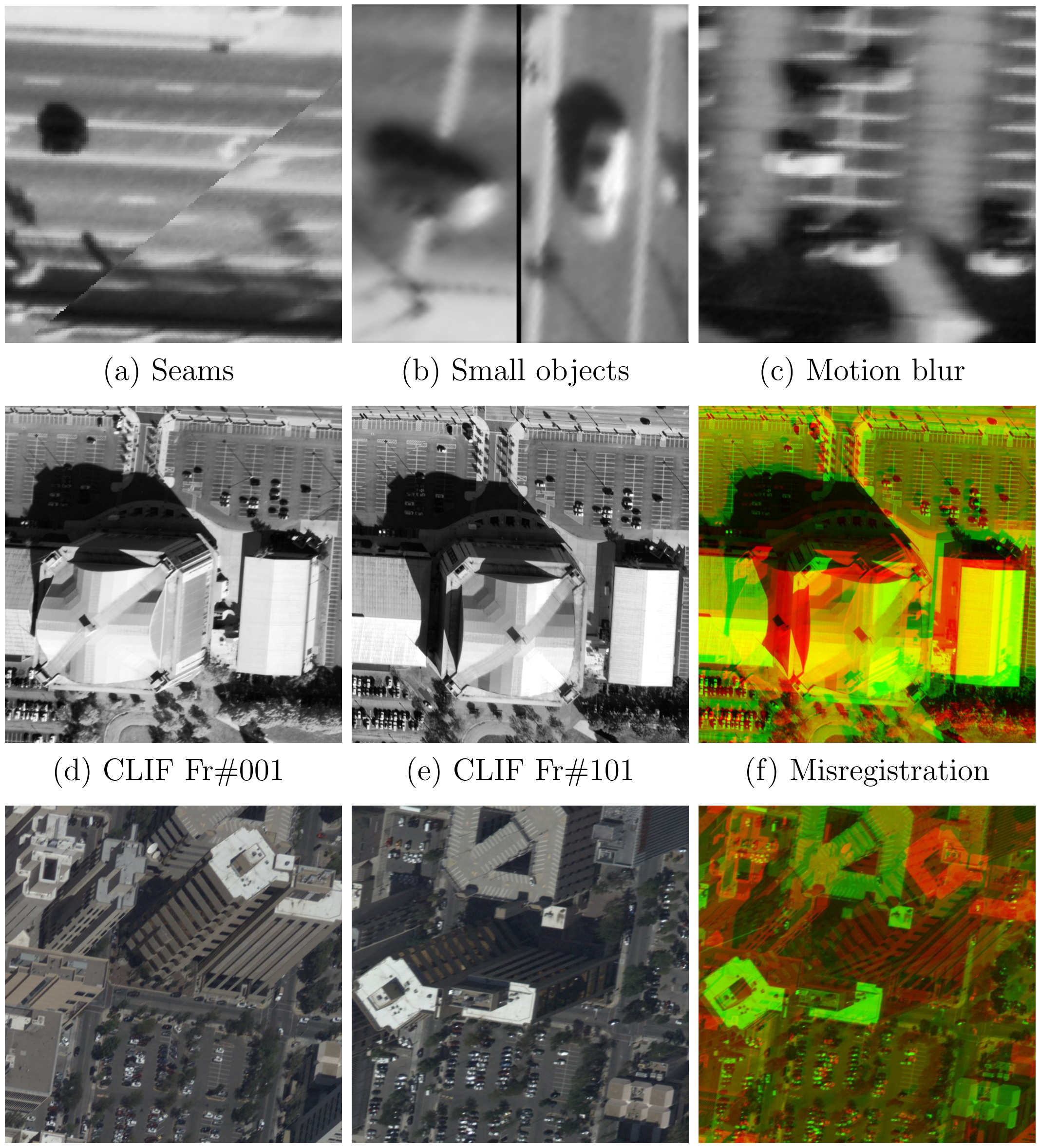


Figure 3.1: Illustration of challenges in WAMI: (a) Seams in multi-camera stitching from georegistration errors. (b) Small vehicle objects with drastic appearance change due to relative viewpoint. (c) Uncorrected motion blur. (d), (e) Two frames from CLIF 2007. (f) Composite pseudocolor image with (d) in red and (e) in green channels, showing ground-plane misalignment. (g), (h) Two frames from ABQ 2013 dataset. (i) Composite image showing parallax of tall buildings.

combines complementary appearance and motion information. Appearance-based detections are obtained using YOLO (You Only Look Once) [149] deep learning-based object detection system trained with vehicle image patches from aerial imagery. Motion detection is performed using a robust 3D (2D + time) tensor-based approach extending [150].

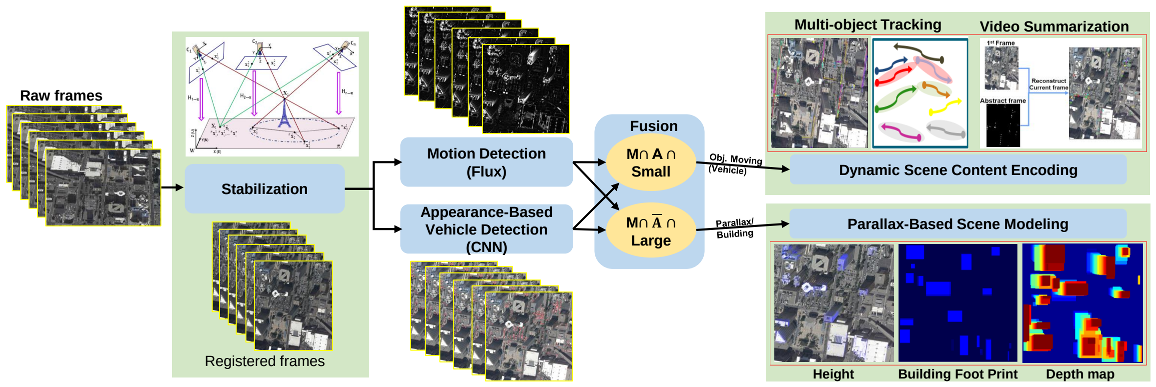


Figure 3.2: Multi-cue moving vehicle detection pipeline using motion, appearance, and shape information from detections at different stages. In the first stage, the aerial video is georegistered and stabilized using [148], in the second stage motion-based flux detection is fused with appearance-based YOLO detections.

The proposed moving vehicle detection system, for airborne WAMI, consists of three main modules: (1) tensor-based motion detection, (2) appearance-based vehicle detection, and (3) decision fusion. These modules combine computer vision methods for motion detection with machine learning approaches (deep learning for appearance-based detection) and rely on complementary appearance and motion information. Beyond moving vehicle detection, which is the main focus of this section, the proposed hybrid and multi-cue system also helps in detection of other scene structures such as high-rise buildings that is useful in scene understanding.

3.2.1 Tensor-based Motion Detection

This section describes the tensor-based motion detection module used in the proposed multi-cue pipeline. Structure tensors for images and video are a matrix representation of partial derivative information [150]. They allow both orientation estimation and image structure analysis with applications in image processing and computer vision. 2D structure tensors have been widely used in edge/corner detection and texture analysis, and 3D structure tensors have been used in low-level motion estimation and segmentation [151, 152].

The 3D structure tensor matrix $\mathbf{J}(\mathbf{x})$ for the spatiotemporal volume centered at $\mathbf{x}$ can be written in matrix form, without the positional terms shown, for clarity, as Eq. 3.1.

$$\mathbf{J} = \begin{bmatrix} \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial x} d\mathbf{y} & \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial y} d\mathbf{y} & \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial t} d\mathbf{y} \\ \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial y}\frac{\partial \mathbf{I}}{\partial x} d\mathbf{y} & \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial y}\frac{\partial \mathbf{I}}{\partial y} d\mathbf{y} & \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial y}\frac{\partial \mathbf{I}}{\partial t} d\mathbf{y} \\ \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial t}\frac{\partial \mathbf{I}}{\partial x} d\mathbf{y} & \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial t}\frac{\partial \mathbf{I}}{\partial y} d\mathbf{y} & \int_{\mathbf{\Omega}} \frac{\partial \mathbf{I}}{\partial t}\frac{\partial \mathbf{I}}{\partial t} d\mathbf{y} \end{bmatrix} \tag{3.1}$$

The elements of $\mathbf{J}$ (Eq. 3.1) incorporate information relating to local, spatial, or temporal gradients. The trace of the structure tensor, $\mathbf{trace}(\mathbf{J}) = \int_{\mathbf{\Omega}} ||\nabla I||^2 d\mathbf{y}$ incorporates total gradient change information in space and time corresponding to *both* moving and non-moving edges of the image sequence, but fails to capture the nature of these gradient changes (i.e. spatial only versus temporal).

The flux tensor [72, 71], characterizes temporal variations in the optical flow field within a local 3D spatiotemporal volume, and is our extension to 3D structure tensors designed to detect only the moving structures without expensive eigenvalue decompositions. In the proposed pipeline, in order to prevent information loss due to isoluminance, we define the *color flux tensor*, $\mathbf{J_{FC}}(\mathbf{x})$, as an extension to the regular flux tensor computed as follows:

$$\begin{bmatrix} \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{xt})^2 & \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{xt}I_{yt}) & \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{xt}I_{tt}) \\ \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{yt}I_{xt}) & \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{yt})^2 & \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{yt}I_{tt}) \\ \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{tt}I_{xt}) & \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{tt}I_{yt}) & \sum\limits_{\Omega}\sum\limits_{I=R,G,B}(I_{tt})^2 \end{bmatrix} \tag{3.2}$$

where the following partial derivative notation is used:

$$\begin{aligned} I_x = \tfrac{\partial I}{\partial x}, \quad I_y = \tfrac{\partial I}{\partial y}, \quad I_t = \tfrac{\partial I}{\partial t}, \\ I_{xt} = \tfrac{\partial^2 I}{\partial x \partial t}, \quad I_{yt} = \tfrac{\partial^2 I}{\partial y \partial t}, \quad I_{tt} = \tfrac{\partial^2 I}{\partial t \partial t} \end{aligned} \tag{3.3}$$

The elements of the flux tensor (Eq. 3.2) incorporate information about temporal color gradient changes which leads to efficient discrimination between stationary and moving image features. The trace of the flux tensor matrix,

$$\mathbf{trace}(\mathbf{J_{FC}}) = \int_{\mathbf{\Omega}} ||\frac{\partial}{\partial t}\nabla \mathbf{I}||^2 d\mathbf{y} \tag{3.4}$$

can be directly used to classify moving and non-moving regions without expensive eigenvalue decompositions.

Both tensor formulations use spatio-temporal consistency efficiently, thus producing less noisy and more spatially coherent edge and motion evidence [151]. the use of tensor-based edge and motion estimation also allows natural extension to color image processing by taking into account vector nature of color data. Extending differential-based operations to color images is hindered by the multi-channel nature of color images. The derivatives in different channels can point in opposite directions, hence cancellation might occur by simple addition [153]. The use of tensor-based representation prevents these cancellation effects.

In the proposed system, color flux tensor is used to identify motion blobs. Since this module is applied after video stabilization module which compensates for camera motion, detected motion blobs predominantly correspond to moving vehicles or parallax caused by high-rise buildings. Both of these structures are of interest for video analytics. Moving vehicles to summarize dynamic content, parallax to summarize static content (buildings) captured by a video. Unfortunately, while successful in detecting these structures, tensor-based motion detection can not distinguish these structures from each other (Figure 3.3).

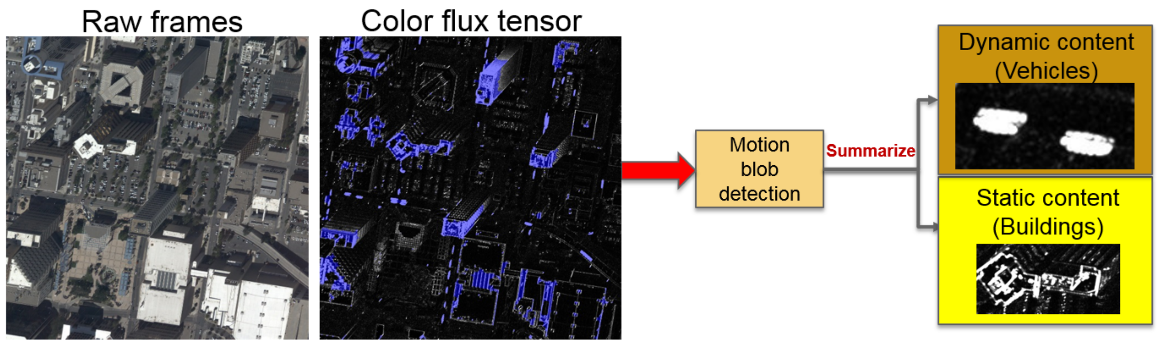


Figure 3.3: Summarization of moving cars and buildings using tensor-based motion detection result

3.2.2 Appearance-based Vehicle Detection using Deep Learning

Recently, deep learning approaches have revolutionized object detection. Faster R-CNN [32], YOLO [33], and SSD [34] are some of the state-of-the-art object detection methods. Deep learning-based object detectors can be divided into two main categories: region proposal based detectors (e.g. Faster R-CNN [32], R-CNN [154]), and single shot detectors (e.g. YOLO [33], and SSD [34]), which do not require a separate region proposal process, making them more computationally efficient. For instance, instead of region proposals, YOLO divides the input image into a grid of cells. Real-time moving object detection requires fast and accurate processing. YOLOv3 [149], an extended version of YOLO, is one of the fastest and most accurate object detections networks. It has 53 convolutional layers trained on ImageNet [155]. Then, 53 more layers are stacked to give the full 106 convolutional layers. YOLOv3 performs detection at three different scales by applying 1×1 detection kernels on feature maps of three different sizes at three different layers in the network. Detecting at different scales improves the detection of small objects compared to the previous versions. YOLOv3 was used since it has: (a) significant speed advantages over two-stage detectors while maintaining high detection accuracies, and (b) better generalization capabilities allowing the network to make reasonably accurate detections on unseen images visually different from the training data (Figure 3.4).

3.2.3 Robust Multi-cue Moving Vehicle Detection

The goal of the fusion-based multi-cue vehicle decision module is to fuse complementary information from two inherently different approaches to allow semantic classifi-

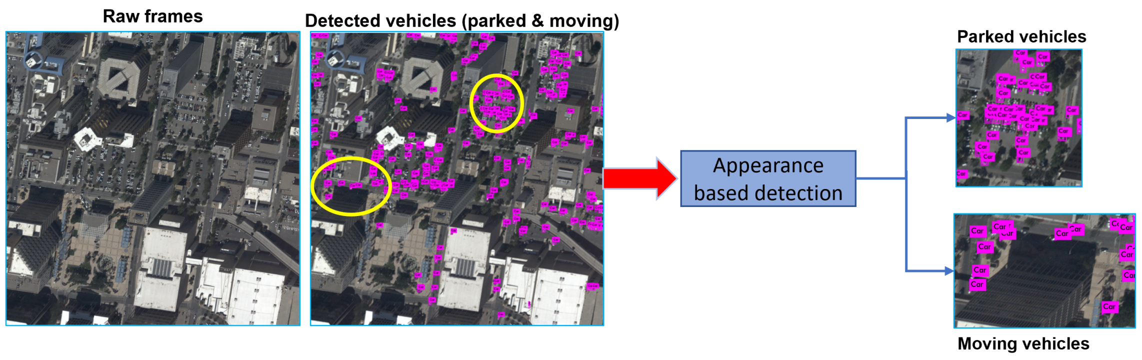

Raw frames
Detected vehicles (parked & moving)
Appearance based detection
Parked vehicles
Moving vehicles

Figure 3.4: Summarization of trackable vehicle objects (parked and moving)

cation of motion blobs, filter spurious detections, and boost overall vehicle detection accuracy. Tensor-based motion detection produces spatio-temporally coherent motion detection results robust to illumination changes and soft shadows due to its use of gradient based information. However, since the method relies on motion, it detects not only moving vehicles but also changes due to motion parallax caused by buildings (Figure 3.3). Appearance-based detection on the other hand returns only vehicles or other regions with appearances similar to vehicles, whether they are moving or stationary (i.e., parked cars). Stationary cars unnecessarily burden follow-up processes such as communication, tracking, and activity analysis. Unlike ground-based images, where objects with larger support regions have distinct appearance features, WAMI imagery consists of much smaller objects with less distinct features. When trained and tested on these smaller, less distinct objects, false-positives are also most likely compared to their counterparts in ground-based, higher resolution surveillance videos. Table 3.1 and Figure 3.5 show the detection categories in the proposed system.

During the fusion process, besides the moving and stationary vehicle category

Table 3.1: Fusion procedure for detecting moving vehicles and parallax-based buildings by combining motion (M) and appearance (A) information (see figure 3.5).

Motion (Flux)	**Appearance (Vehicle CNN)**	**Size**	**Detection Category**
1	1	any	Moving vehicle
0	1	any	Stationary vehicle or False (obj) detection
1	0	small	Other moving object or False (motion) detection
1	0	large	Motion parallax-based buildings

masks, an explicit building category mask is first generated as

$$Mask_{Building} = Mask_{Flux} \cap (1 - Mask_{YOLO}) \tag{3.5}$$

Building mask is then refined by first size based filtering to remove potential false detections, then by morphological operations, connected component labeling, and bounding box fitting (Figure 3.6). While single instance of building roof-top detection is enough to filter-out false vehicle detections. Aggregation of building roof-top detections in time, produces very valuable information regarding 3D scene structure, since spread of the detection instances is directly correlated with building height.

3.3 Detection, Validation and Filtering

The system uses the detection masks/bounding boxes produced from object detection algorithm frameworks. For each frame I_t of a given video sequence $V = \{I_1, I_2, ..., I_Q\}$ of length Q, there are set of N detected objects $D_t = \{d_{t,1}, d_{t,2}, ..., d_{t,N}\}$ where $d_{t,i}$ represents object i in frame I_t. Detection filtering is applied to eliminate potential false detections produced by the detector. Objects with low detection scores, or with undesired class are removed to reduce false positive objects and ensure reliable linking in the further steps.

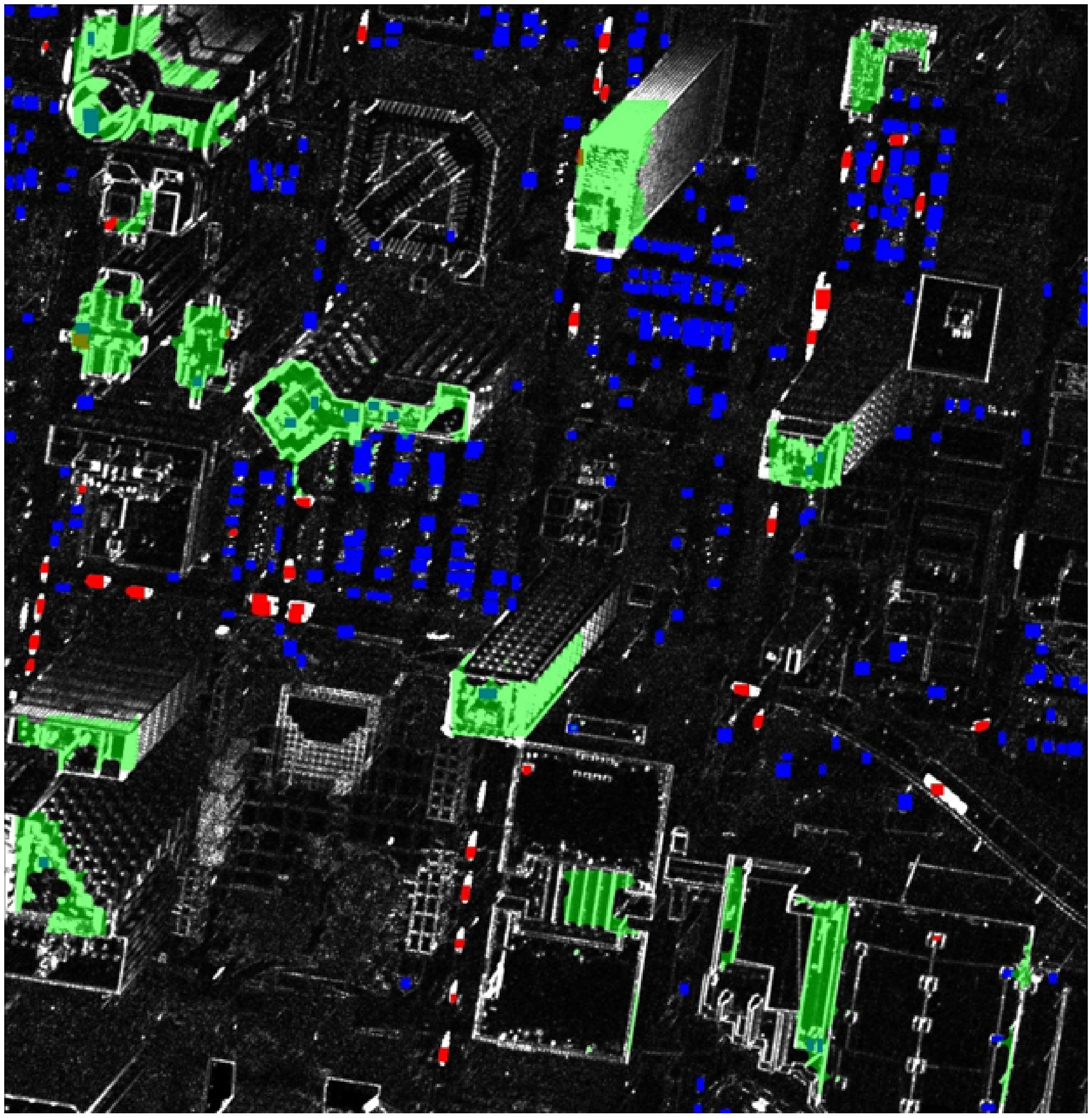


Figure 3.5: The detection categories in the proposed multi-cue detection system. See Table 3.1 for details. Red marker for moving vehicles, blue for the stationary vehicles of false object detection, gray for other moving objects or false motion detection, and green for motion parallax-based building.

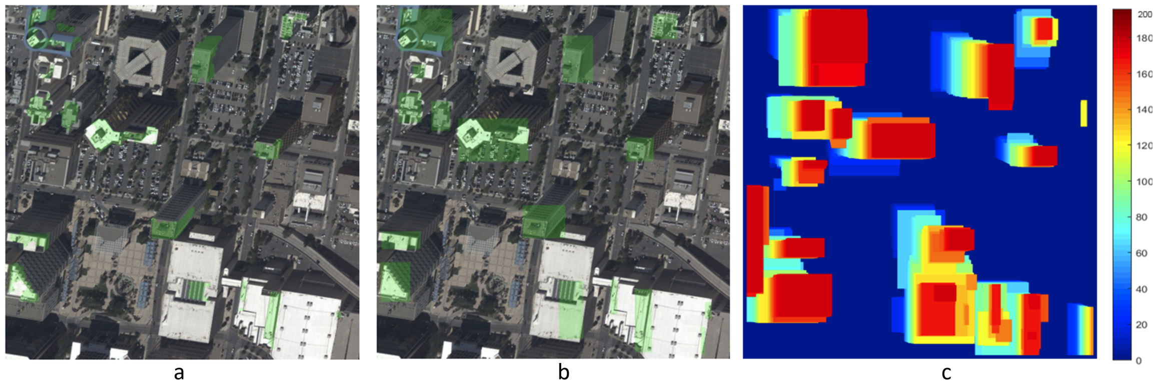


Figure 3.6: Building roof-top detection using flux-based motion parallax response. (a) Building parallax response, obtained fusing Flux tensor-based motion and YOLO-based vehicle appearance cues, overlaid on the original frame, (b) building roof-top bounding boxes for a single frame, obtained by post-processing output in (a), (c) per frame building roof-top detections aggregated in time where light blue indicates earlier instances, and red indicates later instances in the image sequence.

3.4 Track Initialization

When a new track is formed, four groups of information corresponding to the new track are initialized:

1. Detected object's location, width, height, detection score, detected class and appearance histogram. Each detected object $d_{t,i}$ is encoded with the vector $(d_{t,i}[x], d_{t,i}[y], d_{t,i}[w], d_{t,i}[h], s_{t,i}, c_{t,i}, A_{t,i})$, where the entries represent position, width, height, detection score, detected class, and object appearance histogram respectively.

2. Counters for age, start frame, end frame, visible frames, and invisible frames for the tracks;

3. Kalman filter parameters $KF_i^t = \{x_i^t, P_i^t\}$ where x_i^t represents state estimate for the object i at frame t and P_i^t represents associated covariance matrix.

4. Object velocity records.

3.5 Data Association Level 1: Short-term Local Association

Short-term local data association is the first data association stage in our system. In this stage, object detections $D_t = \{d_1, d_2, ..., d_n\}$ are assigned to the previously tracked objects $T_{t-1} = \{T_1, T_2, ..., T_m\}$. Where n is the number of the detected objects at frame t and m is the number of tracked objects at frame $t-1$. We assume that there is a matrix $C_t \in \mathbb{R}^{m,n}$, with entries $c_t^{ij} \in C$ representing the cost of assigning detection j to track i at time t. The goal is to find the optimal assignment of detections to tracks so that the total assignment cost is minimized using Munkres Hungarian algorithm [74]. Binary decision variables $b^{ij} \in \{0, 1\}$ are used to represent a detection-to-track assignment existence. Spatial distance is used for detection-to-track associations between consecutive frames.

$$\min_{b \in B} \sum_{i=1}^{m} \sum_{j=1}^{n} c_t^{ij} b^{ij} \tag{3.6}$$

with constrains

$$\sum_{i=1}^{m} b^{ij} = 1 \qquad j = 1, 2, ..., n, \quad \sum_{i=1}^{n} b^{ij} = 1 \qquad i = 1, 2, ..., m \tag{3.7}$$

$$c_t^{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{3.8}$$

Circular gating regions around the predicted track positions are used to eliminate highly unlikely associations, reduce the computational cost, and reduce false matches.

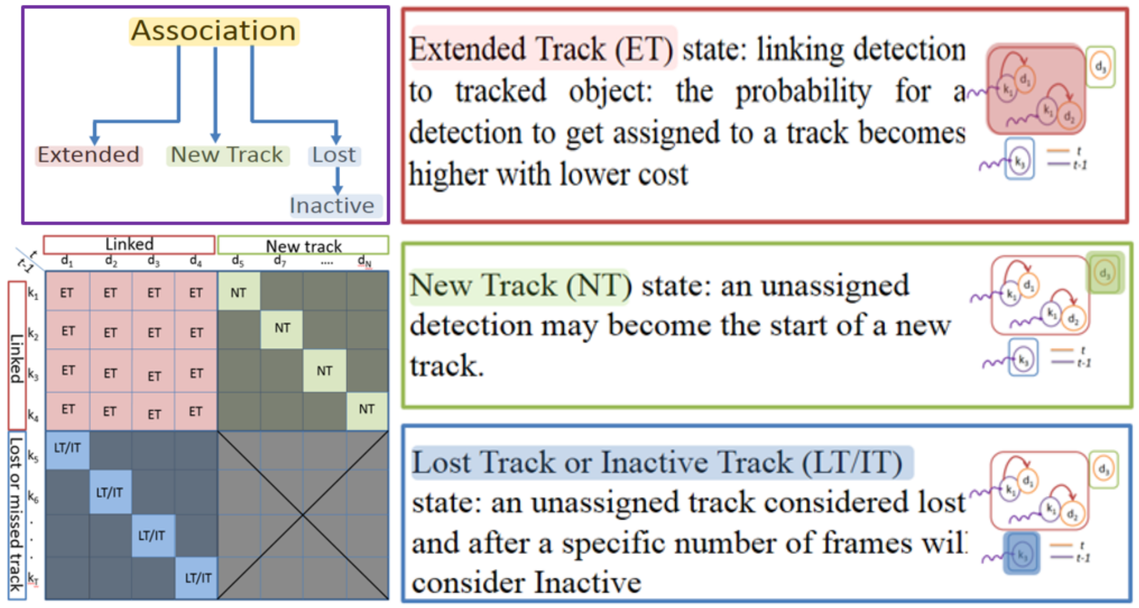

Association
Extended
New Track
Lost
Inactive
Linked
New track
Lost or missed track
ET
NT
LT/IT
Extended Track (ET) state: linking detection to tracked object: the probability for a detection to get assigned to a track becomes higher with lower cost
New Track (NT) state: an unassigned detection may become the start of a new track.
Lost Track or Inactive Track (LT/IT) state: an unassigned track considered lost and after a specific number of frames will consider Inactive

Figure 3.7: Association matrix at frame t with all possible states. The Association matrix is padded with dummy rows and columns to count for unsigned detections and tracks.

Minimization on the cost matrix results in the assignment $\mathbb{A}_t$ matrix containing the indices of the corresponding detection and track pairs. $\mathbb{A}_t$ determines track states for frame t. The four possible states $\{$*new track, extended track, lost track, inactive track*$\}$ are illustrated in Figure 3.7 and as described in Algorithm 3. State descriptions and associated parameter updates are summarized in Table 3.2.

Algorithm 3 Data Association Level 1: Short-term Local Association

```
Input: Detections at time t, $\mathcal{D}_t = \{d_1, d_2, ...., d_n\}$; active tracklet set $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_m\}$
Output: Updated active tracklet set $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_m\}$
   // Compute Assignment Cost Matrix
3  forall $d_i \in \mathcal{D}_t, \mathcal{T}_j \in \mathcal{T}$ do
4      $c_t^{ij} = \sqrt{(x_i - \hat{x}_j)^2 + (y_i - \hat{y}_j)^2}$
5  end
   // Determine Assignment Matrix
6  $\mathbb{A}_t$= MunkresHungarian($\mathcal{C}_t$)

   // Detected object not assigned to any active tracklet
7  forall $d_i$ where $\mathbb{A}_t(d_i, \mathcal{T}_{t-1}) = 0$ do
       // Create New Tracklet:  Init with detected object's properties
8      $\mathcal{T}_{t,k}.DStates \leftarrow d_i.States$
       $\mathcal{T}_{t,k}.Counters \leftarrow \{age = 1, VisibleFrames = 1,$
       $InvisibleFrames = 0, StartFrame = t, EndFrame = NULL\}$
       $\mathcal{T}_{t,k}.KF \leftarrow \{x_i^t, P_i^t\}$
9  end
   // Detected object matched to an active tracklet
10 forall $d_i$ where $\mathbb{A}_t(d_i, \mathcal{T}_{t-1,j^*}) = 1$ do
       // Extend Tracklet:  $\mathcal{T}_{t,j^*}$ extended with info from detection $d_i$.
11     $\mathcal{T}_{t,j^*}.DStates \leftarrow Update(\mathrm{T}_{t,j^*}.DStates, d_i.States)$
       $\mathrm{T}_{t,j^*}.Counters \leftarrow \{age{+}{+}, VisibleFrames{+}{+}\}$
       $\mathcal{T}_{t,j^*}.KF \leftarrow Update(\mathrm{T}_{t,j^*}.KF, d_i.States)$end
       // Tracklet not assigned to any detected object
12     forall $\mathcal{T}_{t-1,j}$ where $\mathbb{A}_t(\mathcal{D}_t, \mathcal{T}_{t-1,j}) = 0$ do
           // Maintain Lost Tracklet:  $\mathcal{T}_{t-1,j}$ is assigned to lost state.
13         $\mathcal{T}_{t,j}.Counters \leftarrow \{age{+}{+}, InvisibleFrames{+}{+}\}$
14     end
       // Terminate Inactive Tracklets:
15     forall $\mathcal{T}_{t-1,j}$ that spends $\lambda$ time steps in lost state do
16         $\mathcal{T}_{t,j^*}.Counters \leftarrow \{EndFrame = t\}$
           Save($\mathcal{T}_{t,j}$)
17     end
```

Table 3.2: State description and associated parameter updates.

State	Parameter Updates	Description
New Track	- Track ID - Kalman filter state $KF_i^t = \{x_i^t, P_i^t\}$ - Counters: age, visible frames, invisible frames - Appearance: CN color histogram & HoG descriptor	Detected objects not assigned to any existing tracks start new tracks.
Extended Track	- Kalman filter state $KF_i^t = \{x_i^t, P_i^t\}$ frames	Detected objects successfully matched to existing tracks extend those tracks.
Lost Track	- Counters: age, invisible frames	Tracks not assigned to any detected objects are assigned to lost state.
Inactive Track	Save - Track ID - Full trajectory (previous and last seen positions) - Appearance: CN color histogram & HoG descriptor - Counters: age, born/death frames	After spending λ time steps in lost state, unmatched tracks are terminated.

3.5.1 Hybrid Cost Matrix Generation

To produce better assignment between tracks and new detection. A hybrid method for generating a cost matrix for spatial information has been exploited depending on the nature of objects that need to be tracked. Three types of spatial information have been used for generating cost matrix C_t:

1. Centroid distance matching is helpful with objects that have large movement displacement due to low frame rates (i.e., moving objects on wide aerial imagery). c_t^{ij} represents the Euclidean distance between the centroid (x_i, y_i) of object i and (x_j, y_j) of object j and computed as

$$c_t^{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{3.9}$$

2. Bounding boxes overlapping matching that is helpful with objects that have low displacement movement. (i.e., cells with very slow motions). The value of c_t^{ij}

represents the intersection over union value between the bounding box area of object i and the bounding box area of object j and computed as

$$c_t^{ij} = \frac{|A_i \cup A_j| - |A_i \cap A_j|}{|A_i \cup A_j|} \tag{3.10}$$

3. Mask overlapping matching which is helpful with objects that have an irregular shape or fast deformable nature. The value of c_t^{ij} represents the intersection over union value between the mask area of object i and object j and computed as

$$c_t^{ij} = \frac{|A_i \cup A_j| - |A_i \cap A_j|}{|A_i \cup A_j|} \tag{3.11}$$

3.5.2 Track Position Prediction

The short-term data association step (Level 1), resolves the associations of the detected object positions D_t to the predicted track positions T_t at frame t. Kalman filter with a constant velocity model is used to predict the new positions of the tracked objects using their past trajectories. Velocity vectors for each target are stored to be used in the following steps. The process involves prediction and update steps. The first positions of the targets are estimated from their current states, then Kalman filter updates the target states using the detections from the current frame.

3.6 Data Association Level 2: Long-term Global Association with Appearance Model

Since the short-term association process considers only the information from consecutive frames, it can not recover from temporary detection or association problems that cause tracks to terminate early resulting in short tracklets rather than full tracks. Such of these problems are occlusions, false detections, matching ambiguities, etc. Further stages of our pipeline are used to improve the performance in these cases. Global data association is used to link fragmented tracklets to generate longer tracks. Global data association is an expensive process because it optimizes all possible hypotheses rather than only those on consecutive frames. In order to reduce computational cost while still resolving complex assignment cases, we perform global association at tracklet level rather than detection level using refinement process. Tracklet level global association offers two main advantages:

1. Reducing number of hypotheses thus reducing computational cost;
2. More information per hypothesis thus more reliable matching.

Re-linking on tracklet level reduces the number of hypotheses, supplies more information per hypothesis, reduces the computational cost, and ensures more robust matching. Given tracklet $\mathcal{T}_i$ and the set of match candidates $\mathcal{J}$ where: $\mathcal{J} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_k\} - \mathcal{T}_i$. The process is described in Algorithm 4. For a given tracklet $\mathcal{T}_i$, at most one tracklet that belongs to the set of candidates $\mathcal{J}$ is assigned to $\mathcal{T}_i$ if the set is not empty after refinement process. The refinement process filters out the infeasible hypotheses to reduce assignment space for the tracklet candidates. The refinement process starts by checking some conditions met with $\mathcal{T}_i$ to be refined from the candidate set $\mathcal{J}$. The refinement process relies on spatial distance, tracklet start and end times, motion directions, and appearance models. Once all potential match sets are refined, global data association determines the tracklet-to-tracklet associations by minimizing spatial and appearance distances.

Algorithm 4 Data Association Level 2: Forward Tracklet Linking

```
Input: Early terminated tracklet $\mathcal{T}_i$, set of match candidates $\mathcal{J} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_k\} - \mathcal{T}_i$
Output: $\mathcal{T}_j^*$ candidate tracklet for forward linking
18 forall $\mathcal{T}_j \in \mathcal{J}$ initialized on frame borders or other source positions do
19     $\mathcal{J} \leftarrow \mathcal{J} - \mathcal{T}_j$; // remove tracklets entering the field of view
20 end
21 forall $\mathcal{T}_j \in \mathcal{J}$ where $(\mathcal{T}_j.StartFrame < \mathcal{T}_i.EndFrame)$ do
22     $\mathcal{J} \leftarrow \mathcal{J} - \mathcal{T}_j$; // remove tracklets born before the death of $\mathcal{T}_i$
23 end
24 forall $\mathcal{T}_j \in \mathcal{J}$ where $(\mathcal{T}_j.StartXY - \mathcal{T}_i.EndXY) > D_{XY}$ do
25     $\mathcal{J} \leftarrow \mathcal{J} - \mathcal{T}_j$; // remove tracklets far from the last position of $\mathcal{T}_i$
26 end
27 forall $\mathcal{T}_j \in \mathcal{J}$ where $(\mathcal{T}_j.StartFrame - \mathcal{T}_i.EndFrame) > D_t$ do
28     $\mathcal{J} \leftarrow \mathcal{J} - \mathcal{T}_j$; // remove tracklets that started much later
29 end
   // dissimilar in appearance
30 forall $\mathcal{T}_j \in \mathcal{J}$ do
31     $D_{CN} = \text{dist}(\mathcal{T}_j.CN(First), \mathcal{T}_i.CN(Last)$ // color (CN distribution)
32     $D_{HoG} = \text{dist}(\mathcal{T}_j.HoG(First), \mathcal{T}_i.HoG(Last)$// shape/texture (HoG descriptor)
33     $D_{Siamese} = \text{dist}(\mathcal{T}_j.Siamese(First), \mathcal{T}_i.Siamese(Last)$// deep features/ (Siamese
           network descriptor)
34     if $(D_{CN} > T_{CN}).or.(D_{HoG} > T_{HoG}).or.(D_{Siamese} > T_{Siamese})$ then
35         $\mathcal{J} \leftarrow \mathcal{J} - \mathcal{T}_j$ // remove tracklets dissimilar in appearance
36     end
37 end
   // best match in terms of spatial and appearance distances
38 if $(\mathcal{J} \neq \emptyset)$ then
39     $\mathcal{T}_j^* = \underset{\mathcal{T}_j \in \mathcal{J}}{\text{argmin}} \quad \text{dist}_{XY,App}(\mathcal{T}_i, \mathcal{T}_j)$
40 end
```

3.6.1 Appearance Model for Tracklet Linking

Appearance description and matching are integral components of the data association process. Appearance descriptors used in tracking should be detailed enough to discriminate different tracked objects, while still being robust to external factors such as illumination changes, shadows, partial occlusions, pose, viewing angles, etc. A hand-crafted/engineered and learned deep feature representations to distinguish matched and unmatched pairs were combined to generate appearance model for the proposed tracker. In terms of hand-crafted/engineered feature description, we use histogram of oriented gradients (HOG) descriptor [76] and object color attributes using our novel color correlation cost matrix. And for learned deep features, the Siamese network is used for matching similarities between tracklet pairs.

HoG [76] is a widely used powerful descriptor that describes shape and texture through histogram of gradient orientations in local image regions. We record the HoG descriptor for all new tracks at the time of track initialization. Mean square error (MSE) is used to compute the distance between HoG descriptors:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(H_{K_i} - H_{K_j})^2 \tag{3.12}$$

where n is the number of HoG histogram bins, H_{K_i} and H_{K_j} is HoG histograms for current tracklet and tracklet potential candidate match respectively.

We incorporate color information through an extension of the CN (Color Names) model proposed in [118]. Linguistic study described in [119] shows that the English language has eleven basic color terms: black, blue, brown, gray, green, orange, pink, purple, red, white, and yellow. [118] explores this concept to generate CN (Color

Names) map. CN model associates RGB color values with linguistic color labels and reduces the 256^3 RGB color space to just 11^1 CN color space. The biggest challenge in color description is sensitivity to illumination. Illumination variations and shadows may alter the color (and intensity) of an object making the information not reliable (e.g. shadow may change blue to black, or yellow to brown). When performing color appearance comparison, it is important to consider the similarity of individual color codes and the likelihood of colors switching from one value to another (e.g., while blue, yellow, and orange are three distinct color names, the distance between blue and yellow is higher than the distance between yellow and orange, and transition likelihood from blue to yellow is lower compared to transition probability from yellow to orange).

We have built and used an 11×11 CN-to-CN color correlation weight matrix $\mathcal{W}_{CN}$ to account for similarity between different CN values during color distribution comparison. The elements of the color correlation matrix are computed as follows:

$$\mathcal{W}_{CN}(c_i, c_j) = 1 - 2 \times \underset{k}{max}\Big(\min \big(\mathcal{G}(k,i), \mathcal{G}(k,j)\big)\Big) \tag{3.13}$$

where c_i and c_j are two CN codes and $\mathcal{G}$ is the $2^{15} \times 11$ matrix describing the mapping from $32 \times 32 \times 32$ quantized RGB space to 11-valued CN space provided by [118]. In order to compare object color distributions, we use earth mover distance (EMD) [78]. EMD computes the minimum paid cost to transfer one histogram to another histogram. We use color correlation matrix $\mathcal{W}_{CN}$ as transfer cost. See Figure 3.8 for more details. Use of cross-bin color histogram distance computation using EMD scheme and color correlation matrix $\mathcal{W}$ described above decreases sensitivity to color changes caused by illumination variations and improves tracking results compared

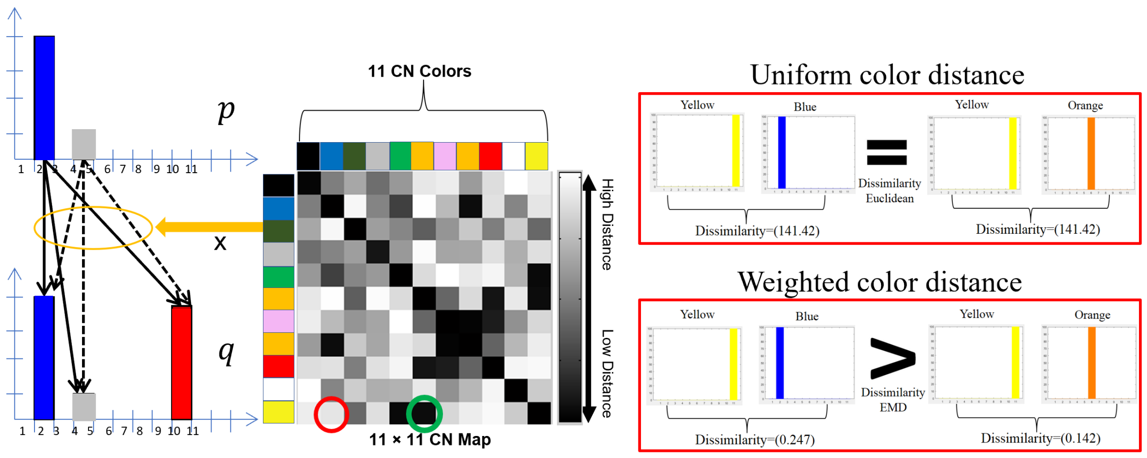


Figure 3.8: Color correlation cost matrix and earth mover distance computation.

to bin-to-bin color histogram comparison. Linking at this stage is exclusively for already partially tracked objects in the scene that suffered from early termination. For cases where objects were not detected for a period of time, because of partial or full occlusions, a separate occlusion handling stage is proposed.

In addition to the aforementioned hand-crafted feature descriptors, we use an offline trained Siamese network [77, 156] to select tracklet match candidates. A Siamese network is a type of deep learning network that uses two or more identical subnetworks that have the same architecture and share the same parameters and weights. We train the network to compare bounding boxes of two protentional tracklets matching. The network was trained to identify whether two bounding box observations are for the same object or not. The used Siamese network is originally for matching images of handwritten characters and the network architecture is illustrated in Figure 3.9. During the training phase, a pair of similar or dissimilar objects are passed through two identical deep learning branches that share the same parameters and weights. The outputs are fed to a loss layer that minimizes the distance between similar objects and maximizes the distance between dissimilar ones. The learned feature representation and matching function is used to find the most relevant tracklet from a candidate tracklet set. To compare two images, each image is passed through one of two identical subnetworks that share weights. The subnetworks convert each $105 \times 105 \times 1$ image to a 4096-dimensional feature vector. Images of the same object have similar 4096-dimensional representations. The output feature vectors from each subnetwork are combined through subtraction and the result is passed through a fully connected operation with a single output. A sigmoid operation converts this value to a probability between 0 and 1, indicating the network's prediction of whether the images are similar or dissimilar. The binary cross-entropy loss between the network prediction and the true label is used to update the network during training.

$$loss = -g \log(y) - (1 - g) \log(1 - y) \tag{3.14}$$

where g is a true label and it can be either 0 or 1, and y is the predicted label. The comparison between the features of two pairs is calculated using Sigmoid function. Sigmoid function is used to find the probability of the similarity between two bounding boxes of two pairs.

3.7 Data Association Level 3: Occlusions Handling

When a tracked object gets occluded, lack of detection causes its track to terminate. When the object later reappears, the system starts a new track with a new object id. Occlusion handling module aims to link these tracklets corresponding to the same object, preserving their object id through the occlusion events. Sample tracklets before

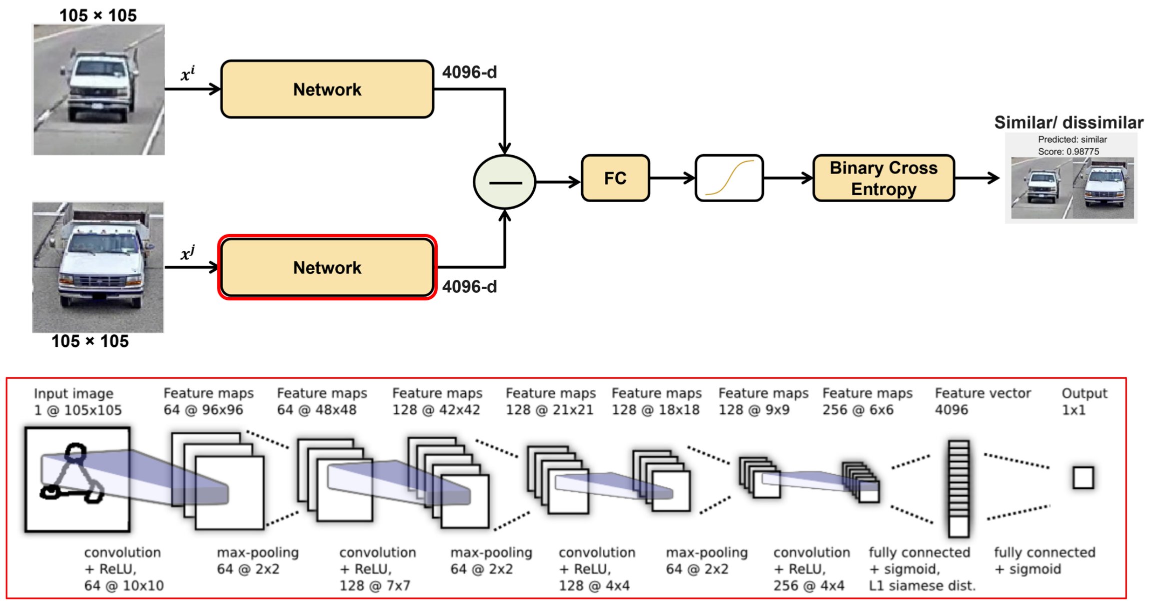


Figure 3.9: Siamese network deep features for matching.

and after occlusion handling process are shown in Figure 3.10. The proposed occlusion handling module extends the methods described in [157] with combined appearance-based and geometric-based constraints instead of geometric only matching. Occlusion handling process starts by identifying early terminated tracklets as pre-occlusion candidates, and late start tracklets as post-occlusion candidates. Tracks that exit the scene at known exit locations or sinks are excluded from the pre-occlusion candidate list. Tracks that enter the scene at known enter locations or sources are excluded from the post-occlusion candidate list. The occlusion handling module uses two types of information:

1. **Tracklets history** (i.e., initialization/termination times and positions, motion direction, etc. for tracklets in pre- and post-occlusion candidate lists).

2. **Scene information** (i.e., object detection masks to locate potential dynamic

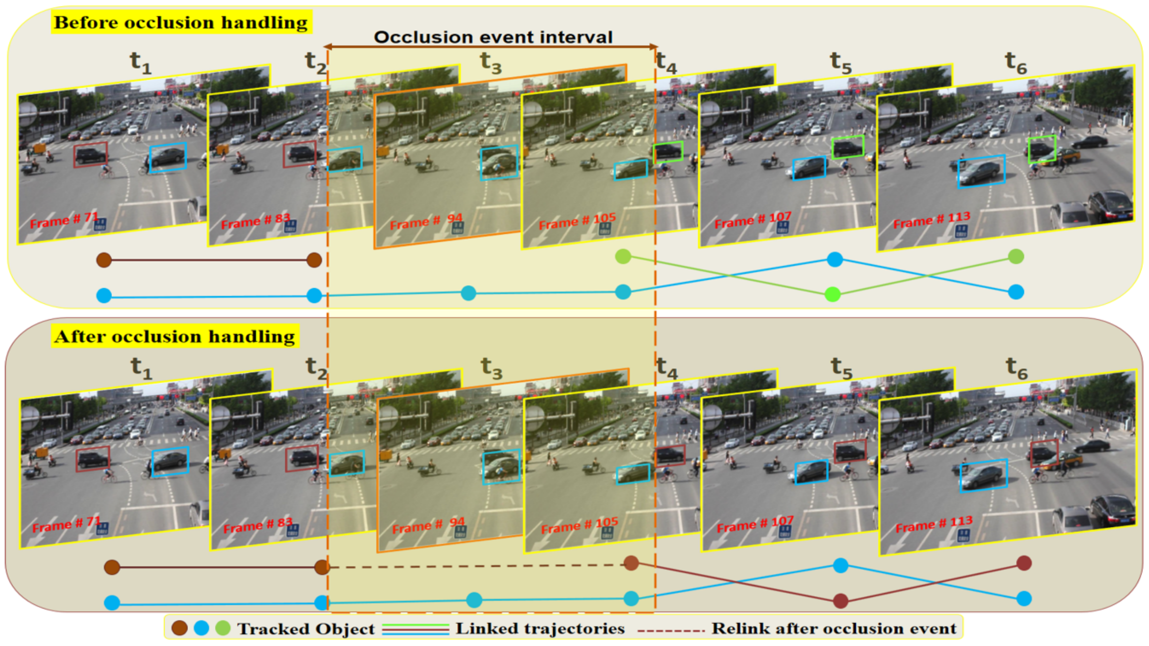


Figure 3.10: Sample tracklets before and after occlusion handling. The first row shows the trajectories after applying the first two data association stages from our pipeline (short-term and long-term data association). The second row shows relinked tracklets after applying occlusion handling module, the third data association stage.

occluders in the scene).

For each tracklet x in the pre-occlusion list, an occlusion confidence map $\eth$ is generated. These maps encode predicted locations of the tracked objects post occlusion. Occlusion confidence maps are generated by combining tracklet motion model $P_{T,x}$, occlusion confidence $P_{O,x}$, and motion prediction range $P_{M,x}$ for the tracked object for t subsequent frames as described in Eq. 3.15:

$$\eth^t(i) = P_{T,x}(i) \times P^t_{O,x}(i) \times P^t_{M,x}(i) \tag{3.15}$$

where i denotes pixel location and t denotes the maximum occlusion duration considered by the proposed system (an input parameter).

Tracklet motion model $P_{T,x}$ encodes inertial state of the tracked object using its motion history:

$$P_{T,x}(i) = \exp\left(- \frac{\left(\langle p_x, d\rangle - \|p_x\|\|d\|\right)^2}{2\sigma_1^2\|p_x\|^2\|d\|^2} \right) \tag{3.16}$$

where p_x is the predicted motion direction of trajectory x, which is computed as the mean of the motion direction history of x; d ($d = i - x_i$) is the spatial distance from pixel location i to the last seen location of x; and σ_1 is the motion direction variance.

Occlusion confidence $P_{O,x}$ measures the probability of a pixel being occluder/foreground versus background in t subsequent frames:

$$P_{O,x}(i) = \begin{cases} 1, & \text{if } i \in \text{detected \ regions (FG).} \\ 1-\alpha & \text{otherwise \ (BG).} \end{cases} \tag{3.17}$$

where $\alpha = [0, 1]$ is detection score reliability. Motion prediction range $P_{M,x}$ encodes predicted motion of the tracked object in t subsequent frames:

$$P_{M,x}(i) = \exp\left(- \frac{\|i - x_i\|^2}{2\sigma_2^2 t^2 p_x} \right) \tag{3.18}$$

where σ_2 is the motion variance within t frames. The combined confidence map $\mathfrak{d}^t$ is then used to find the most probable match from the post-occlusion list to the selected tracklet in the pre-occlusion list. The process is repeated for each tracklet in the pre-occlusion list. Figure 3.11 shows the intermediate results of the occlusion handling process.

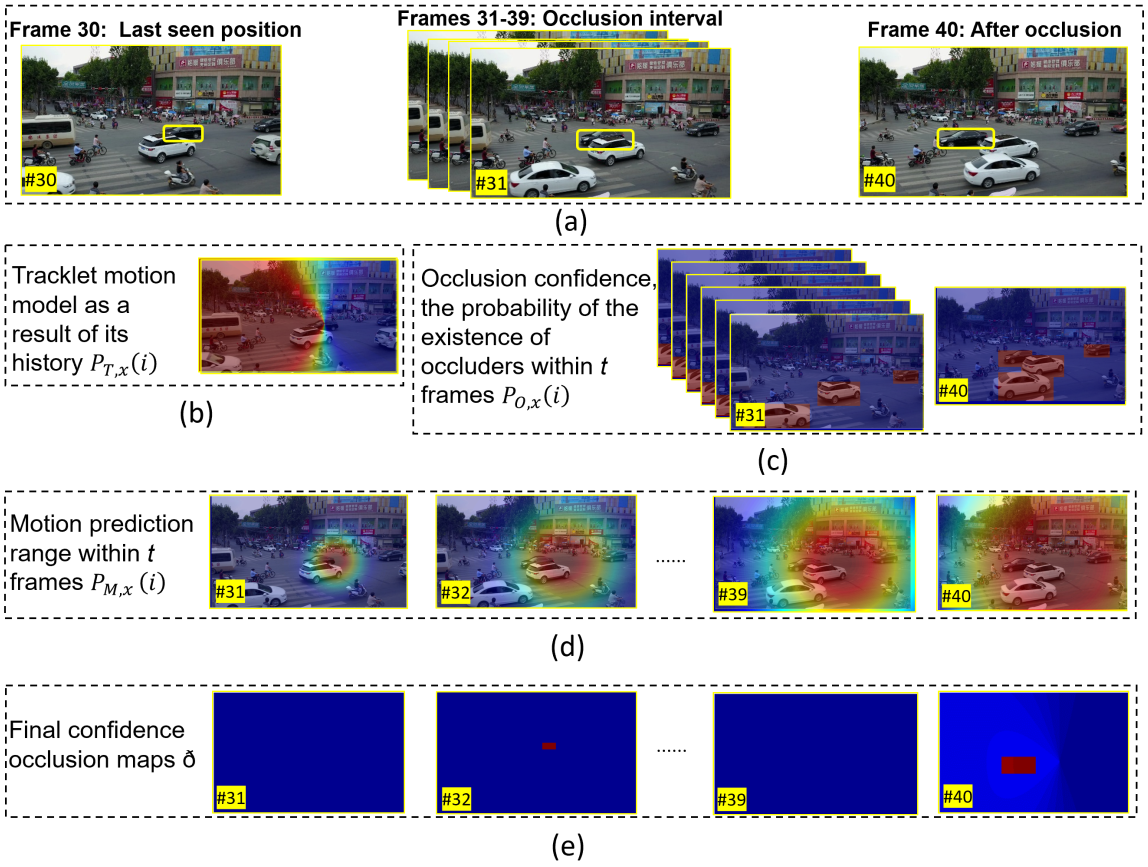

Frame 30: Last seen position
Frames 31-39: Occlusion interval
Frame 40: After occlusion
#30
#31
#40
(a)
Tracklet motion model as a result of its history $P_{T,x}(i)$
(b)
Occlusion confidence, the probability of the existence of occluders within t frames $P_{O,x}(i)$
#31
#40
(c)
Motion prediction range within t frames $P_{M,x}(i)$
#31
#32
......
#39
#40
(d)
Final confidence occlusion maps ð
#31
#32
......
#39
#40
(e)

Figure 3.11: Occlusion event handling process for a sample object from ViaDrone [62] dataset. The object is occluded from frame #31 to #39. A time interval ($t = 15$) is set to show how many frames are used for predicting the location of the object after occlusion event starts. (a) Video frames with bounding box on the selected object, last seen of the selected object, occlusion interval, and after occlusion. (b) Tracklet motion model $P_{O,x}$ that shows the inertial confidence of the object as a result from its history, the higher probability is toward the direction of object motion. (c) Occlusion Confidence $P_{O,x}$ the probability of potential occluders within t frames. (d) Object motion prediction $P_{M,x}$ within t frames. (e) The combined confidence $ð^t$ for the tracklets initiated on a specific frame.

3.8 M2Track Versions and Names

For efficient implementation with different datasets and challenges. M2Track has been written in Matlab and Python. M2Track has different pipeline components with different assigned names according to the purpose of using the tracker to fit different applications. Different M2Track versions are:

1. **M2Track-Lite**: Slim version of M2Track with only level1 data association (Matlab/ Python)

2. **M2Track-L2**: up to Level 2 from M2Track (Matlab/ Python)

3. **M2Track**: all three data association modules (Matlab)

Chapter 4

Multi-object Detection and Tracking for Aerial Videos

4.1 Overview

In recent years, there has been an exponential increase in detection and multi-object tracking applications due to advances in sensor technologies, and rapid advances in computational hardware in terms of both power and cost. These technological advances are leading to the emergence of new applications in different domains including aerial surveillance [158, 159], traffic monitoring [160, 161], urban planning [162, 163], precision agriculture [164, 165, 166], search and rescue [167, 168], and disaster relief [169]. Society is seeing a growing need for robust aerial imagery and video analytics capabilities to take full advantage of data fusion and to meet such application needs [170]. Novel methods, particularly those using artificial intelligence/machine learning (AI/ML), coupled with rapid advances in computational hardware (more powerful, lighter weight, lower energy, lower computing cost) are revolutionizing image processing, pattern recognition, and information fusion (e.g., WAMI fusion applications [171]). There is a growing need for robust video analytics capabilities to take full advantage of this data and address its applications' pressing needs. In this chapter, wide aerial videos will be explored as high and low altitude data for applying our object detection and tracking approaches to different acquisition methods to ensure our work robustness to different cross-domain data for different analytical computer vision applications.

4.2 Wide Aerial Motion Imagery/ High Altitude

Wide area motion imagery (WAMI) is characterized by large ground coverage of a few square miles, many objects of interest, and high-altitude oblique viewing geometries. WAMI platforms equipped with orientation sensors circle above a region of interest at a constant altitude, adjusting steadily the orientation of the camera array pointing to a narrow area of interest [172] within the region being imaged. Once georegistered and stabilized, these videos provide a virtual nadir (i.e., downward) view of the region being monitored [172] and enable large-scale surveillance and monitoring activity analysis applications for extended periods of time.

WAMI exploitation pipelines for object recognition and multi-target tracking have unique challenges such as large camera motion, low frame rate, small object sizes, multi-camera arrays, hundreds to thousands of moving objects per frame, oblique viewing angles, motion blur, parallax effects, shadows, etc., in addition to regular sensor resolution and weather challenges. Figure 3.1 shows some examples of these challenges.

This section briefly describes some high-altitude WAMI videos of interest. Table 4.1 gives a short description of sample publicly available WAMI datasets of interest. Two of these datasets are our focus, a multi-camera Columbus Large Imagery Format (CLIF) 2007 [2] dataset, and a single-camera Albuquerque, New Mexico (ABQ) 2013 [1] dataset. Video and annotation details for these datasets are summarized in Table 4.2 and further described below:

Table 4.1: Five WAMI dataset collections and their characteristics.

Name	Sensors	Scene	Ground-truth	Targets
UNICORN 2008 [173]	Visible (6-cameras) SAR	Wright-Patterson Air Force Base	Manual+GPS Partial (4 million labels)	Moving vehicles, radar reflectors, calibration targets
WPAFB 2009 [174, 175]	Visible (6-cameras) SAR	Wright-Patterson Air Force Base	1,537 stitched images; GT: 1/3 training; 1/3 self-test	All moving vehicles for two-thirds of the frames
MAMI 2013 [176, 177]	Aerial (5-color+1-gray) Ground (4-color) cameras	Wright-Patterson Air Force Base	Manual - Partial	Variety of objects in the scene
CLIF 2007 [2]	Visible (6-cameras)	Ohio State University campus	Manual (3,502,401labels)	All moving vehicles
ABQ 2013 [1]	Single camera	Albuquerque, NM Downtown	Manual	All moving vehicles in an ROI

Columbus Large Image Format (CLIF) 2007 Dataset: CLIF 2007 dataset [2] consists of several hours of imagery collected from a large format electro-optical (EO) platform by AFRL Sensors Directorate on October 2007 over the Ohio State University (OSU) campus. The data is collected using a matrix of six cameras at approximately 2 frames per second. Figure 4.1 shows samples of the matrix of the six cameras raw data, georegistered frame, and car samples.

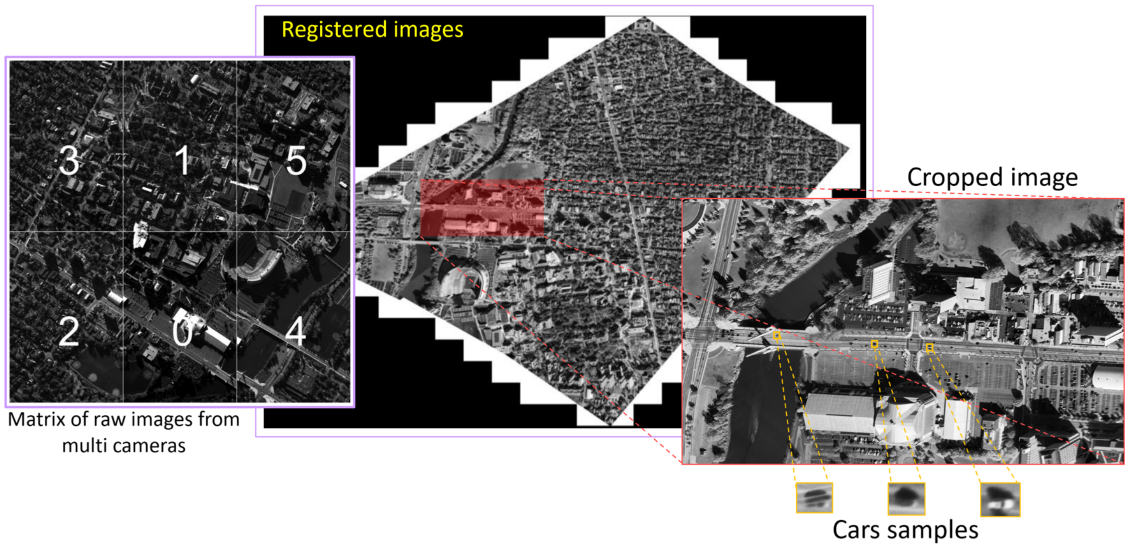


Figure 4.1: CLIF 2007 sample for raw image, georegistered, cropped, and moving car samples from the sequence

ABQ 2013 Dataset: Aerial urban imagery dataset collected by TransparentSky [1] using a large format camera mounted on a gimbal with on-board GPS and IMU, with a circular data collection flight path 1.5km above ground level over downtown Albuquerque, NM on September 3, 2013 [178, 179]. Imaging was done at a frame rate of 4Hz and 2.6km orbit radius. This dataset contains 1071 raw high-resolution images (6600×4400) with a nominal ground resolution of 25cm. Ground-truth for the dataset consists of manually marked bounding boxes and track IDs for all the moving vehicles (139 vehicle tracks in total) in a 2000×2000 region of interest extracted from 200 consecutive frames. Figure 4.2 shows samples of raw, georegistered ultra high-resolution images, and car samples.

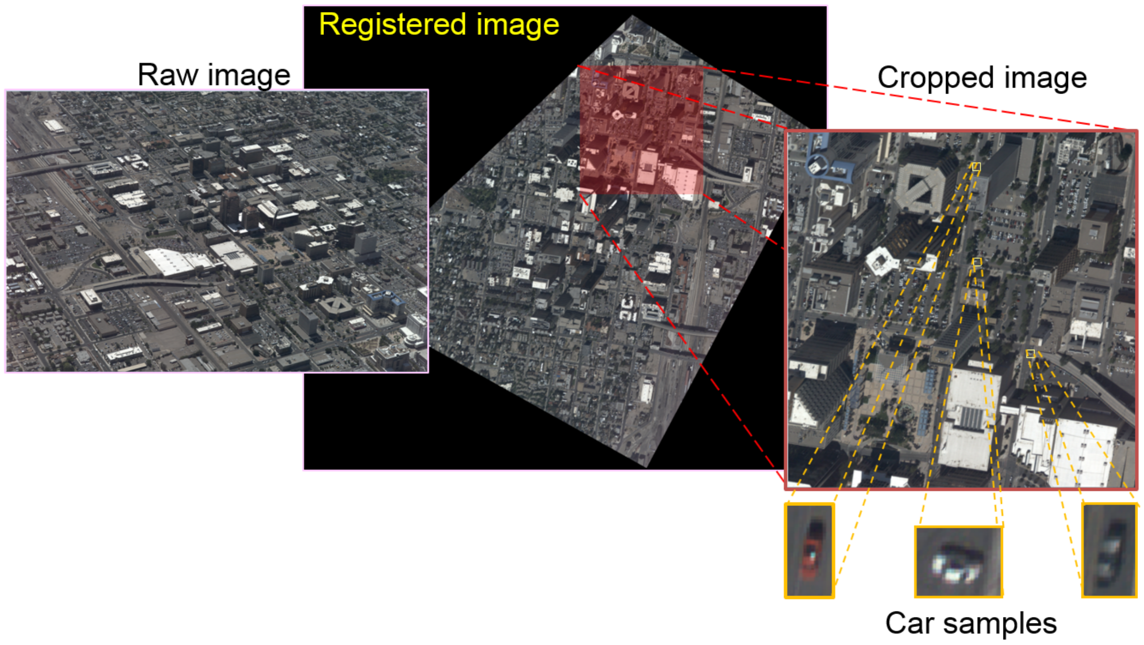


Figure 4.2: ABQ sample for raw image, georegistered, cropped, and moving car samples for the first frame of the sequence

Table 4.2: Video and annotation details for the ABQ 2013 [1] and CLIF 2007 [2] WAMI datasets.

Dataset	ABQ 2013 [1]	CLIF 2007 [2]
Frame per second	4	~ 2
Raw frame size	6600×4400 pixels	4016×2672 pixels
Registered frame size	$\approx$ 12000× 12000 pixels	31744×29696 pixels
ROI size with GT	2000×2000 pixels	Full frame
# Frames with GT	200	6,343
# Object instances	139	3,502,401

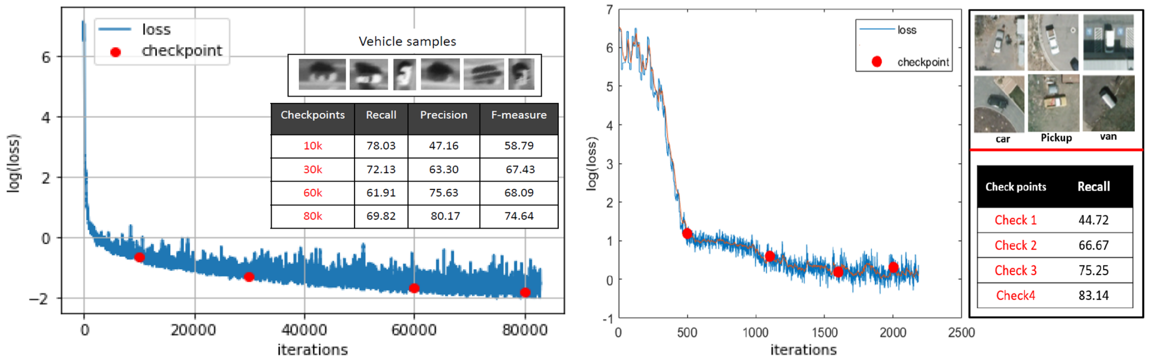
4.2.1 Moving Object Detection and Tracking

In this section, georegistered cropped images are used for both CLIF 2007 and ABQ datasets for our multi-object tracking pipeline. First, our proposed multi-cue object detection pipeline was applied to detect the moving cars on the scene. Then, the detected moving cars are tracked with our proposed detection-based multi-object tracking pipeline (M2Track).

Moving Object Detection for ABQ and CLIF 2007: multi-cue moving object detection was implemented on high altitude videos, in which, appearance-based WAMI vehicle detection using YOLO, tensor-based motion, and change detection Flux were fused to detect moving vehicles in the scenes.

For **appearance-based object detection**, single-stage detectors YOLOv3 [149] since it has significant speed advantages over two-stage detectors while maintaining high detection accuracy and having better generalization capabilities allowing the network to make reasonably accurate detections on unseen images visually different from the training data. YOLOv3 [149], an extended version of YOLO, is one of the fastest and most accurate object detections networks. It has 53 convolutional layers trained on ImageNet [155]. Then, 53 more layers are stacked to give the full 106 convolutional layers. YOLOv3 performs detection at three different scales by applying 1×1 detection kernels on feature maps of three different sizes at three different layers in the network. Detecting at different scales improves the detection of small objects compared to the previous versions which are beneficial for detecting small vehicles on WAMI.

We demonstrated appearance-based detection performance using two YOLO networks, one trained using the CLIF dataset [2], and one using the Vehicle Detection in Aerial Imagery (VEDAI) dataset [180]. VEDAI consists of 1200 satellite images collected during Spring 2012, over Utah, USA with an image resolution of 12.5cm $\times$ 12.5cm per pixel. Figure 4.3 shows training loss versus iterations trained using the CLIF dataset and a few image patches from the training set. During both training and inference, the very large WAMI images are partitioned into 500 × 500 non-overlapping image patches and fed to the network patch by patch. This process prevents loss of image resolution caused by image resizing, a critical problem for WAMI datasets with small targets.



Checkpoints	Recall	Precision	F-measure
10k	78.03	47.16	58.79
30k	72.13	63.30	67.43
60k	61.91	75.63	68.09
80k	69.82	80.17	74.64

Check points	Recall
Check 1	44.72
Check 2	66.67
Check 3	75.25
Check4	83.14

Figure 4.3: Loss for appearance training phase in YOLOv3 on CLIF 2007 (left), and ABQ (right) datasets. The red dots on the curves correspond to (recall, precision, and f-measure) values listed in the sub-figure for each dataset.

For **motion-based object detection**, it is important to robustly detect true object motion and structural changes in the scene as opposed to changes caused by artifacts such as illumination changes [178]. Flux tensor [71, 72, 56] is used to estimate and detects only the moving structures. Flux tensor is a 3D (2D+time) tensor-based approach that estimates and detects only the moving structures without expensive eigenvalue decompositions. The process detects the moving regions corresponding to real moving objects (vehicles), but also apparent motion caused by parallax of high-rise buildings. Building mask is then refined first by size-based filtering to remove potential false detections, then by morphological operations, connected component labeling, and bounding box fitting. While a single instance of building roof-top detection is enough to filter out false vehicle detections, aggregation of building roof-top detections in time produces very valuable information regarding the 3D scene structure shown in Figure 3.6 since the spread of the detection instances is directly correlated with building height. Figure 4.4 shows the intermediate results and the final result after applying the multi-cue detection pipeline on ABQ dataset.

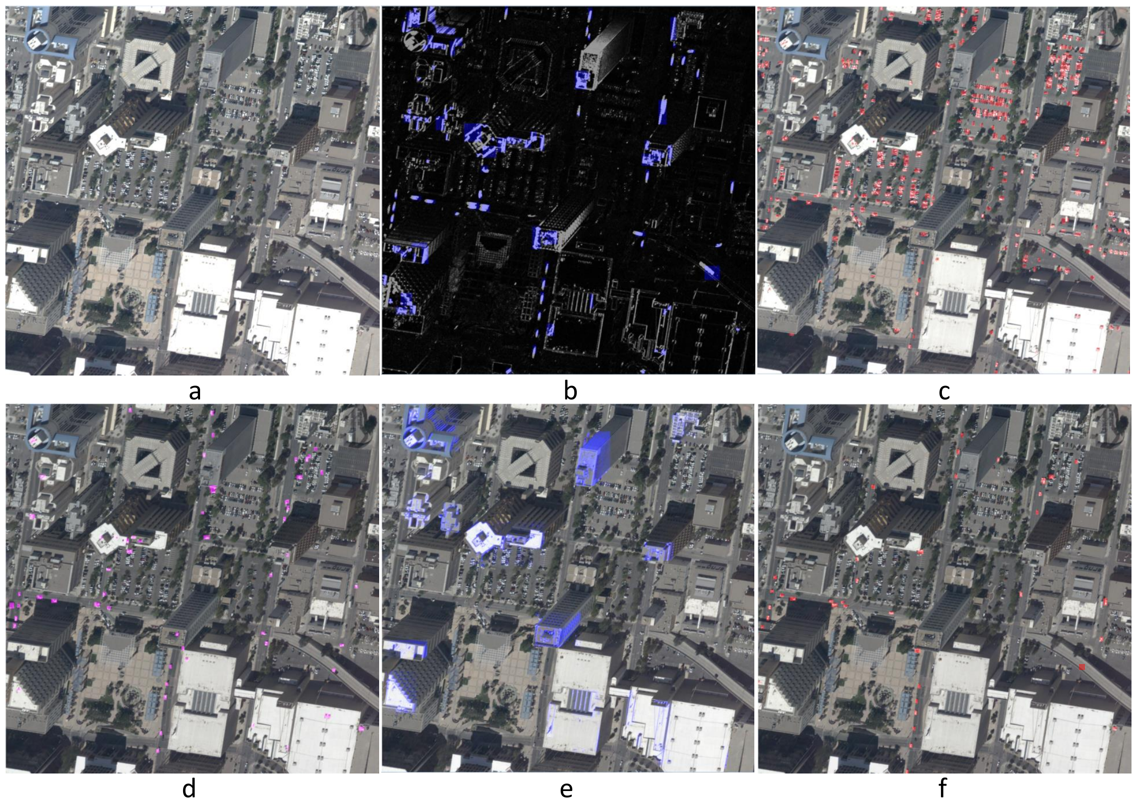


Figure 4.4: The intermediate results and the final result after applying multi-cue detection pipeline on ABQ dataset. a) Raw data, b) Motion mask overlaid on flux tensor motion-based detection. c) Appearance mask overlaid on the raw frame, the red overlaid masks represent all predicted vehicles (moved and parked) in the scene. d) Appearance-motion fusion result, some false positive appears on the top of the buildings. e) Buildings mask. f) The final result after filtering out false positives on the top of buildings.

For Object Tracking, we have tested and evaluated M2Track-Lite a slim version of our multi-cue multi-target tracker M2Track as described in Section 3.8. We applied the vehicle detection results from our multi-cue object detection pipeline on both ABQ and CLIF 2007 datasets. Figure 4.7 shows tracking results for both datasets.

4.2.2 Evaluation Metrics

Detection Metrics: Detection accuracy was evaluated on object-level measures. The results are quantitatively evaluated in terms of detection measures recall, precision (Eq. 4.1), and F-measure (Eq. 4.2), where GT, DT, and TP denote ground-truth, detection, and true prediction objects respectively.

$$Recall = \frac{\#TP}{\#GT}; \qquad Precision = \frac{\#TP}{\#DT} \tag{4.1}$$

$$Fmeasure = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{4.2}$$

MOT Metrics: Different evaluation metrics have been proposed for multi-object tracking [4, 181, 182]. For the evaluation of WAMI multi-object tracking results, we adopted Multi-Object Tracking (MOT) challenge evaluation metrics summarized below and described in [183, 182]. The toolkit for MOT benchmark evaluation provided in [184] was used to evaluate the WAMI tracking results. Below is a brief description of the MOT evaluation metrics used in this section:

1. **MOTA**: Multiple object tracking accuracy, the most popular metric, is calculated by combining three types of errors on a per-frame basis, and can be negative:

$$\text{MOTA} = 1 - \frac{\sum_t (FN_t + FP_t + IDS_t)}{\sum_t GT_t} \tag{4.3}$$

where t is the frame number, FN_t is the number of undetected (missed) objects (false negatives), FP_t is the number of extra detected objects (false positives), and GT_t is the number of ground-truth objects in frame t.

2. **IDS**: Identity switches, counts number of identity mismatches by considering the ID mapping in frame t and $t-1$. The IDS metric describes the number of times that the matched identity of a tracked trajectory changes.

3. **FRAG**: Fragmentation metric is the number of times that trajectories are fragmented. Both IDS and $FRAG$ metrics reflect the accuracy of tracked trajectories.

4. **MT**: Mostly tracked metric computes the percentage of trajectories (with respect to the number of ground-truth trajectories) tracked accurately for more than 80% of the trajectory duration.

5. **PT**: Partially tracked are cases not labeled as MT or ML.

6. **ML**: Mostly lost metric computes the percentage of trajectories tracked accurately for less than 20% of the trajectory duration. MT and ML metrics determine how much of the trajectories are recovered by the tracker.

4.2.3 Evaluation and Experimental Results

Georegistration accuracy: Inaccurate georegistration can result in both unstable ground-plane motion and motion from building parallax which makes filtering buildings more difficult and detecting vehicles inaccurate. Georegistration accuracy was assessed using four manually tracked points on the dominant ground plane for ABQ 2013 (4 Hz, 200 frames) and CLIF 2007 ($\sim$2 Hz, 100 frames) to quantify pixel drift errors. The mean and standard deviation of the translation errors ($\Delta x, \Delta y$, Euclidean distance) for each tracked point, averaged over all frames, due to shifts (drift) arising from georegistration errors are shown in Table 4.3. Figure 4.5 shows the drift with respect to the mean shift (left scatter plot) and with respect to adjacent frame pairs (right line plot). It is evident that CLIF has very large georegistration errors of more than one order of magnitude that is because the conventional homography estimation methods used for CLIF only use the information available from 2D feature correspondences and ignore (or unable) to utilize the 3D information in underlying scenes. CLIF has higher errors than the errors in ABQ that uses BA4S pose refinement [185, 148, 186] that utilizes the 3D information in underlying scenes. Outliers have up to 50-pixel shift error, with respect to mean position, and over 25-pixel frame-to-frame shift error in CLIF due to inaccuracies in multi-camera georegistration. The former reflects georegistration accuracy, while the latter is indicative of difficulties during vehicle detection and data association for tracking.

Table 4.3: Mean drift error in pixels for four points tracked in each WAMI sequence for 200 and 100 frames respectively in ABQ and CLIF, after georegistration using different methods.

Dataset	**Point A**	**Point B**	**Point C**	**Point D**	**Mean**	**StdDev**
ABQ-BA4S	0.5620	0.5095	0.4463	0.6607	0.5446	0.3599
CLIF-Conv	6.3970	4.5907	6.7544	6.3670	6.0273	5.0900

Detection Results: Figure 4.6 shows motion detection results on sample frames from two WAMI datasets ABQ 2013 single-camera WAMI, and AFRL CLIF 2007 multi-camera WAMI. For ABQ, there are motion responses from both moving objects and building structures due to motion parallax. This makes it difficult to use only the estimated motion areas to aid in detecting and tracking vehicles. Despite the fact that the ground plane in the video is well stabilized, by using 3D building struc-

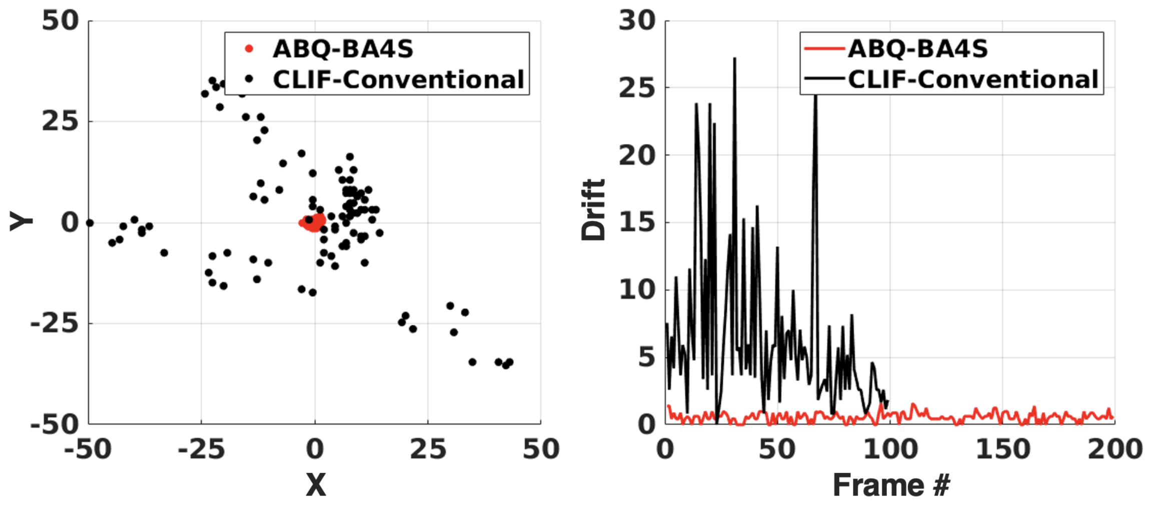


Figure 4.5: Drift in pixels for one manually tracked point in each WAMI sequence across 200 (ABQ) and 100 (CLIF) frames.

ture cues, the motion responses due to building motion parallax can be accurately filtered out. On the other hand, image transformation and mosaicing are another challenge in large-scale WAMI datasets. Inaccurate georegistration can result in both unstable ground-plane motion and motion from building parallax which makes filtering buildings more difficult and detecting vehicles inaccurate. The detection problem is further compounded by visible seams when mosaicing multiple cameras in WAMI frames, which can lead to motion detection failures, as shown in Figure 4.6 for AFRL CLIF dataset.

Table 4.4 shows detection performance for two moving vehicle detection approaches. Appearance-only using CNN-based detections (YOLOv3) are shown in Figure 4.4c and Figure 4.7b, and f. While, best recall is obtained by this approach, even lower precision (9.36%) for ABQ dataset is obtained because of the parked vehicles. Combining appearance and motion-based object detections generates promising results (Figure 4.4d) since parked vehicles get filtered out thanks to the motion mask from the flux tensor. Some false positives still remain due to vehicles parked on building rooftops. Explicit building detection through motion and appearance clues as described in Section 3.2.3, and used of it to further filter vehicles parked on rooftops results in the best precision (71.16%) and F-measure (64.41%) values for ABQ dataset and best precision (63.65%) and F-measure (72.09%) values for CLIF dataset.

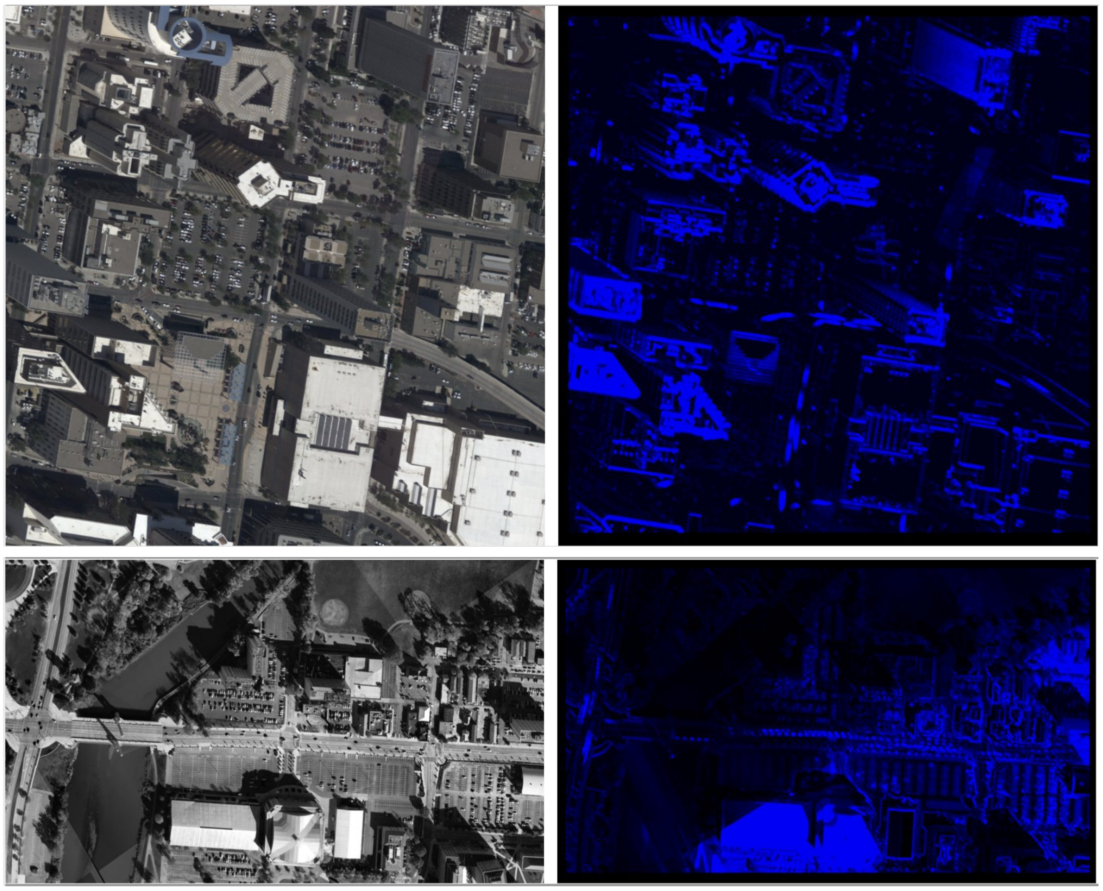


Figure 4.6: Motion detection results for sample images from two WAMI/ high-altitude datasets: Row 1 is ABQ 2013, and Row 2 is AFRL CLIF 2007 dataset. Col 1 shows the original images. Col 2 shows motion detection results using Flux and the detection response is shown in blue

MOT Results: Table 4.5 shows M2Track-Lite multi-object tracker performance on two sample high altitude WAMI datasets, single-camera ABQ 2013 [1] dataset with

Table 4.4: Precision, recall, and F-measure (in percent) for YOLO [149] and the proposed multi-cue moving vehicle detection pipeline for ABQ [1] and CLIF 2007 [2]

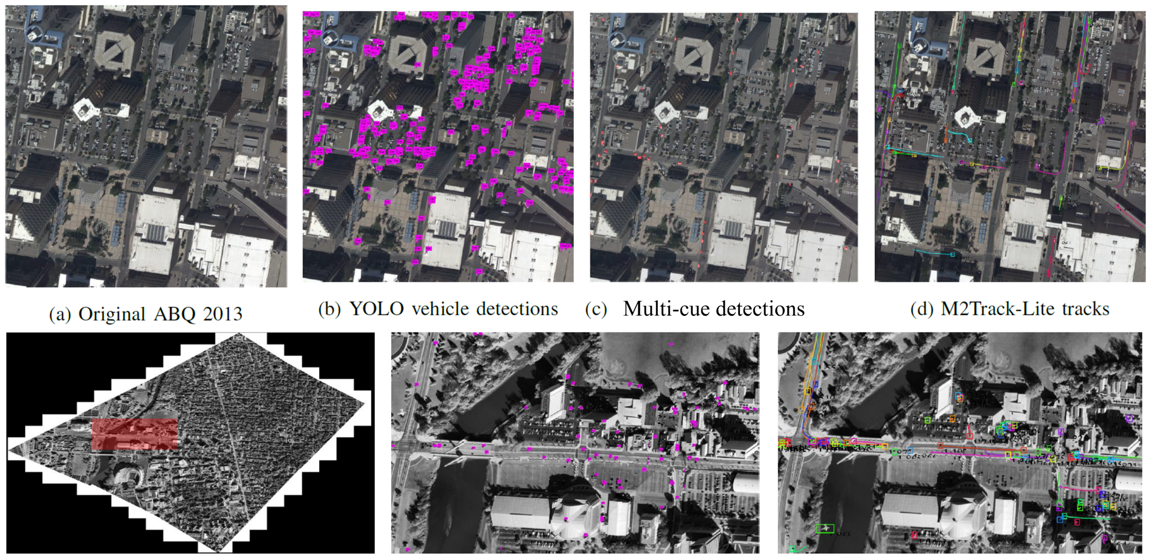
Datasets	Detected Alg.	#GT	#Detection	# Matches	Precision	Recall	F-measure
ABQ	YOLO [149]	8323	74368	6966	83.69	9.36	16.84
ABQ	Multi-cue (proposed)	8323	10066	5923	71.16	58.84	64.41
Cropped CLIF	YOLO [149]	7544	6550	5283	70.02	80.65	74.96
Cropped CLIF	Multi-cue (proposed)	7544	5778	4802	63.65	83.10	72.09

fairly accurate georegistration, and multi-camera AFRL CLIF 2007 [2] dataset with larger georegistration errors (Table 4.3, Figure 4.5) and visible inter-camera seams, Figure 4.6. Detection and tracking performance evaluations are also presented for a $5,000 \times 3,000$ region of interest (ROI) from the AFRL CLIF 2007 dataset positioned at $(x = 8750, y = 11220)$. This ROI was selected because the region includes multiple busy intersections and persistently stays in the field of view of the cameras resulting in longer ground-truth trajectories. Tracking results are presented for three types of detections corresponding to manually generated ground-truth (GT) detections, appearance-based deep network detections (YOLO [149]), and the proposed multi-cue detection approach based on the fusion of appearance-based and motion/change-based detections described in 3.2.

WAMI multi-object tracking performance heavily depends on object detection and georegistration accuracy. Figure 4.7 illustrates sample detection results (YOLO and multi-cue object detection (the proposed approach) and multi-object tracking results. Fusion of multiple cues provides better detections leading to improved tracking; using YOLO only versus the multi-cue detection approach, MOTA improves from -310 to 73 on ABQ (see Table 4.5). The impact of georegistration accuracy on multi-object detection and tracking was evaluated using CLIF 2007. When large georegistration errors are present, average of $\sim$6 pixels in CLIF, accurate estimation of motion and

Table 4.5: Tracking performance using ground-truth vehicle detections, YOLO, and multi-cue object detection (the proposed approach) in WAMI datasets ABQ 2013, and cropped $5,000 \times 3,000$ pixel region of interest (ROI) from AFRL CLIF 2007 (see Figure 4.7(e)). Evaluation metrics include Multi-object Tracking Accuracy (MOTA), Id Switches (IDS), Fragmentation (FRAG), Mostly Tracked MT), Partially Tracked (PT), Mostly Lost (ML), and GT-ID (Number of Tracks) as described in HOTA [183]. ABQ 4 Hz, 200 frames; CLIF $\sim$2 Hz, 100 frames.

Datasets	**Georegistration**	**Detector Alg**	**M2Track-Lite Tracker**							
Metrics			**MOTA↑**	**IDS↓**	**FRAG↓**	**MT↑**	**PT↑**	**ML↓**	**GT-ID**	**#Detections**
ABQ	BA4S	GT	99.70	24	0	139	0	0	139	8,323
ABQ	BA4S	YOLO [149]	-310.0	53	0	68	11	43	139	35,027
ABQ	BA4S	Multi-cue [73]	73.80	248	8	139	0	0	139	10,066
Full CLIF	Conventional	GT	90.42	3,559	676	885	1	0	886	12,413
Full CLIF	Conventional	YOLO[149]	40.30	184	46	571	0	0	571	17,574
Full CLIF	Conventional	Multi-cue [73]	30.30	1,058	45	551	15	5	571	16,338
Cropped CLIF	Conventional	GT	99.30	52	19	129	0	0	129	7,544
Cropped CLIF	Conventional	YOLO[149]	71.90	1054	281	100	25	4	129	6,550
Cropped CLIF	Conventional	Multi-cue [73]	68.30	622	264	73	48	8	129	5,778


(a) Original ABQ 2013 (b) YOLO vehicle detections (c) Multi-cue detections (d) M2Track-Lite tracks

(e) Original AFRL CLIF 2007 with ROI (f) YOLO vehicle detections in cropped region (g) M2Track-Lite tracks in cropped region

Figure 4.7: Sample detection and tracking results for ABQ 2013 [1] and AFRL CLIF 2007 [2] WAMI showing improvement in detection accuracy when appearance (YOLO) and motion (Flux) are combined.

change cues is adversely impacted, decreasing MOTA score from 71.9 for YOLO only, to 68.3 for multi-cue detection in the cropped CLIF 2007 and from 40.3 to 30.3 for the Full CLIF 2007 dataset (see Table 4.5). These results demonstrate that early upstream processing stages of WAMI video analytics pipelines, especially multi-camera georegistration accuracy, are crucial for accurate performance in small target detection and tracking.

On the other hand, M2Track was tested and compared with some state-of-the-art and the baseline trackers whose codes are available publicly online. The experiments were implemented on ABQ dataset. Our M2Track tracker outperforms other comparable trackers. M2Track takes into account spatial, appearance, and kinematic information to ensure reliable trajectory linking. While GOG [47] basically uses pairwise edges between network flow graphs to link trajectories resulting in ignoring kinematic constraints between observations. CMOT [42] is an online tracker that does not exploit the visual information from future frames which is important for increasing performance under unpredictable object displacement and camera motion. IHTLS [187] focuses on motion information to link the tracklets which leads to many identity switches, especially with WAMI datasets since the motion is quite challenging. SORT [50], the online tracker, approximates the inter-frame displacements of each object with a linear constant velocity model, which is independent of object categories and camera motion. Table 4.6 shows the multi-object tracking results for the selected baseline trackers and our M2Track-Lite tracker.

Table 4.6: Tracking performance comparison between four baseline trackers and our proposed tracker M2Track-Lite on ABQ dataset. The bounding box detections from the multi-cue object detection approach (proposed) were used for tracking.

Tracker	**MOTA ↑**	**MT ↑**	**PT ↑**	**ML ↓**	**GT**	**IDS ↓**	**FRAG ↓**
M2Track-Lite	37.80	139	0	0	139	248	8
GOG [47]	62.5	80	54	5	139	1500	428
CMOT [42]	66.3	70	39	30	139	165	56
IHTLS [187]	87.9	110	20	9	139	200	69
SORT [50]	54.1	26	93	20	139	419	310

4.3 Drone-base Aerial Imagery/ Low Altitude

Multi-object tracking for low-altitude aerial video analytics is exploited in this section. Low-altitude aerial videos are captured by Unmanned Aerial Vehicles (UAVs). Object detection and tracking implemented on UAVs have unique challenges, such as rapid scale changes, partial or full occlusions, rotation and camera jittering, scenes and camera view changes, high densities of similar objects, and shadow and illumination changes.

VisDrone2018 dataset [79] is one of the drone-based low altitude datasets that is used in our analytical implementation for our multi-object tracking pipeline. VisDrone2018 dataset [79] consists of 263 video clips with 179,264 frames and 10,209 static images. The set includes different scenarios taken across 14 different cities in China. The resolutions of video clips and static images are 3840×2160 and 2000×1500 respectively. The dataset was taken in different weather and light conditions. Ten categories are considered and annotated in VisDrone2018 dataset including pedestrian, person, car, van, bus, truck, motor, bicycle, awning-tricycle, and tricycle. Occlusion and truncation ratios are also provided. The dataset has many challenges (i.e., scale changes, high object density, rapid camera movements, viewpoint, and camera angle changes). The dataset was divided into training, validation, and testing set.

Since 2018, VisDrone team organizes challenge workshops. In 2018, A challenge "Vision Meets Drone Video Object Detection and Tracking" (VisDrone-VDT2018) was organized in conjunction with the European Conference on Computer Vision (ECCV 2018). Next year, the challenge workshop was organized in conjunction with International Conference on Computer Vision (ICCV 2019). The challenge focuses on four tasks; 1) object detection on static images; 2) object detection on videos;

3) single object tracking; 4) multi-object tracking. Each task has its own set of implementation and evaluation metrics. Multi-object tracking task is our focus.

4.3.1 Challenge Participation and Evaluation Metrics

We have tested and evaluated our M2Track tracker on VisDrone2018-test set that consists of 16 challenging real-world drone videos. Our tracker participated in 2018 and 2019 challenge workshops as task4A and task4B for multi-object tracking respectively as described below:

- **Task4A**:(without prior detection), for each object class, a list of bounding boxes, trajectory identifications, and confidence scores are required for evaluation. The evaluation protocol sorts the trajectories according to the average confidence detection score of each trajectory. Then, intersection over union overlapping with the ground truth according to specific thresholds will be considered for evaluation. Mean average precision **mAP** over different thresholds is calculated. Three thresholds are considered for evaluation are (0.25, 0.50, 0.75).

- **Task4B**: (with prior detection), the average rank of 10 metrics which are: **MOTA**, **MOTP**, **IDF1**, **FAF**, **MT**, **ML**, **FP**, **FN**, **IDS**, and **FM** is used to compare different algorithms. The Multi-Object Tracking Accuracy metric **MOTP** is:

$$MOTP = \frac{\sum_{t,i} d_{t,i}}{\sum_t m_t} \tag{4.4}$$

  where m_t denotes the number of matches in frame t and $d_{t,i}$ is the bounding box overlap of target i with its assigned ground truth object. **MOTP** is the measure of localization precision that shows how well the algorithm is. **IDF1** is identification metric computes the matching between ground-truth trajectories and predicted trajectories in term of three metric, **IDTP** True Positive ID, **IDFP** False Positive ID, and **IDFN** False Negative ID. **IDF1** values can range between 0 and 100%:

$$\mathbf{IDF1} = \frac{2IDTP}{2IDTP + IDFP + IDFN} \tag{4.5}$$

The **FAF** metric shows the average number of false alarms per frame. **FP**, **TP**, and **FN** count are determined by thresholding the intersection over union (**IoU**) between predicted and target bounding boxes using a threshold $d = 50\%$. The remaining metrics are described in previous section 4.2.2.

4.3.2 Evaluation and Experimental Results

M2Track-L2 participated in VisDrone2018 challenge 2018 on task4B. M2Track-L2 combines the first two levels from M2Track tracker pipeline, short-term local association, and long-term global association. We incorporate Faster R-CNN [188] object detection output to be input to M2Track-L2 tracker. The results of the comparison between our tracker to the other participants are shown in Table 4.7. Our rank was the fourth out of six participants. The table is described in detail in the workshop challenge report [138].

M2Track participated in VisDrone2018 challenge 2019 on task4A. M2Track uses three-level cascaded data association (local, global, and occlusion handling). We incorporate Faster R-CNN [188] object detection output to be input to M2Track

Table 4.7: Multi-object tracking results with prior object detection in each frame on VisDrone-VDT2018 testing set. The submitted algorithms are ranked based on the average rank of the ten metrics. The performance of the trackers is presented according to [138]

Method	**Rank↓**	**MOTA↑**	**MOTP↑**	**IDF1↑**	**FAF↓**	**MT↑**	**ML↓**	**FP↓**	**FN↓**	**IDS↓**	**FRAG↓**
V-IOU[46]	2.7	40.2	74.9	56.1	0.76	297	514	11838	74027	**265**	**1380**
TrackCG [189]	2.9	**42.6**	74.1	**58.0**	0.86	323	395	14722	68060	779	3717
GOG [47]	3.2	36.9	**75.8**	46.5	**0.29**	205	589	**5445**	86399	354	1090
M2Track-L2 [101]	3.8	35.8	75.6	45.1	0.39	211	550	7298	85623	798	2042
Ctrack [189]	3.9	30.8	73.5	51.9	1.95	**369**	**375**	36930	**62819**	1376	2190
FRMOT [138]	4.0	33.1	73.0	50.8	1.15	254	463	21736	74953	1043	2534

tracker. The results of the comparison between our tracker to the other participants are shown in Table 4.8. The table is described in detail in the workshop challenge report [190].

Table 4.8: Multi-object tracking results on the VisDrone-MOT2019 testing set. The challenge focus on five object categories (i.e., pedestrian, car, van, bus, and truck). The performance of the trackers is presented according to [190]

Method	**AP**	**AP@0.25**	**AP@0.50**	**AP@0.75**	**APcar**	**APbus**	**APtrk**	**APped**	**APvan**
DBAI-Tracker [191]	43.94	57.32	45.18	29.32	55.13	44.97	42.73	31.01	45.85
TrackKITSY [191]	39.19	48.83	39.36	29.37	54.92	29.05	34.19	36.57	41.20
Flow-Tracker [192]	30.87	41.84	31.00	19.77	48.44	26.19	29.50	18.65	31.56
HMTT[191]	28.67	39.05	27.88	19.08	44.35	30.56	18.75	26.49	23.19
TNT-DRONE	27.32	35.09	26.92	19.94	38.06	22.65	33.79	12.62	29.46
GGDTRACK	23.09	31.01	22.70	15.55	35.45	28.57	11.90	17.20	22.34
Ctrack	16.12	22.40	16.26	9.70	27.74	28.45	8.15	7.95	8.31
CMOT	14.22	22.11	14.58	5.98	27.72	17.95	7.79	9.95	7.71
IITD-DeepSort	13.88	23.19	12.81	5.64	32.20	8.83	6.61	18.61	3.16
T&D-OF	12.37	17.74	12.94	6.43	23.31	22.02	2.48	9.59	4.44
M2Track	10.09	14.95	9.41	5.92	18.98	17.86	4.86	5.20	3.58
VCLDAN	7.50	10.75	7.41	4.33	21.63	0.00	4.92	10.94	0.00
GOG	6.16	11.03	5.30	2.14	17.05	1.80	5.67	3.70	2.55
TBD	5.92	10.77	5.00	1.99	12.75	6.55	5.90	2.62	1.79
CEM	5.70	9.22	4.89	2.99	6.51	10.58	8.33	0.70	2.38
H2T	4.93	8.93	4.73	1.12	12.90	5.99	2.27	2.18	1.29
IHTLS	4.72	8.60	4.34	1.22	12.07	2.38	5.82	1.94	1.40
SGAN	2.54	4.87	2.06	0.69	10.42	0.00	0.00	2.27	0.00
OS-MOT	0.16	0.18	0.18	0.13	0.00	0.00	0.71	0.00	0.09

For testing the contribution of appearance-based deep feature matching on the Level2 global data association of M2Track, Siamese network [77, 156] to select tracklet match candidates was used. The network was trained to identify whether two bounding box observations of two fragmented tracklets are for the same object or not. AI-City Challenge dataset/ vehicle re-Identification track [137] was used for training. The network trained on (36,935 images for 1,879 cars) and evaluated on (18,290 images for 798 cars) each car taken from a single camera with an average accuracy of (93.7%) on the validation data using 30 mini-batches on 19,000 iterations. For testing, we used the trained weight on VisDrone2018-Validation set to test the similarity between fragmented tracklets on the level2 global data association module. Table 4.9 shows the comparison results for M2Track tracker without deep features matching, M2Track tracker with deep features matching for tracklet linking, and M2Track-L2 with using only Siamese network features for matching. The tested trackers use YOLOv3 [149] detection approached pretrained on ImageNet [155] and implemented on VisDrone2018-Validation set. The table shows using the Siamese network for matching tracklets improves the performance. However, using only deep features by excluding the other M2track module components affects the tracking performance. M2Track's modules accumulate for best performance and still important to have them all together.

Table 4.9: The comparison results for different versions of M2Track. **M2Track**: tracker without deep features matching, **M2track+Siamese**: M2Track with deep features matching for tracklet linking, and **M2Track-L2/Siamese**: M2Track-L2 with using only Siamese network deep features for matching. The trackers use YOLOv3 [149] detection approached trained on ImageNet [155] and implemented on VisDrone2018-Validation set

Tracker	**MOTA↑**	**MOTP↑**	**IDS↑**	**FRAG↓**	**FP↓**	**FN↓**	**GT**	**MT↑**	**PT**	**ML↓**
M2Track	71.3	28.4	285	69	4520	804	125	114	6	5
M2track+Siamese	73.5	27	300	68	4092	775	125	115	4	6
M2Track-L2/Siamese	31.4	30	218	27	12808	369	125	116	5	5

Chapter 5

Cells Segmentation, Detection and Tracking for Biomedical Videos

5.1 Overview

The capacity of cells to interact with and exert forces on their environment and alter their shape as they move [193] is essential to many biological processes including the cellular immune response to infections [194], embryonic development [195], wound healing [196] and tumor growth [197]. Detecting cell shape and their changes over time as cells navigate the microenvironment is essential for understanding the multiple mechanisms guiding and regulating cell motility [198]. It is important for biomedical research and medical diagnosis applications to study the behavior movement for individual cell formation, mitosis, lineage, density, etc. Manually locating and tracking cells is labor-intensive, and subject to different opinions from domain knowledge experts during validation. Automatic tracking methods are therefore required to analyze microscopy videos in order to achieve accurate motion and behavior analysis.

Over several decades, many classical computer vision methods and pipelines have been developed for automated cell detection, segmentation, and tracking [196, 199, 200, 201]. More recently, various models have been developed for cell boundary prediction to handle segmentation of touching cells [202, 203, 202, 203, 204, 205]. However, accurate cell analysis under different protocols, imaging modalities, and cell types remains challenging due to experimental variability, low signal-to-noise ratios, touching or overlapping cells, indistinct deforming boundaries, particularly in high cell density cases, agile, unpredictable motion of individual cells, and dynamic interactions between cells. In this chapter, two benchmarks for cell detection and tracking were used to apply our multi-object tracking pipeline. Cell Segmentation and Tracking Benchmark (CTC) offers a dataset of different cell types using different imaging modalities, and Cell Tracking with Mitosis Detection Benchmark (CTMC-v1) features a single imaging modality with different cell types.

5.2 Cell Segmentation and Tracking Benchmark

Automated methods and pipelines are needed to perform microscopy video analysis for image data acquired during live-cell studies. Particularly to segment, track, and characterize cells to accelerate scientific discovery and clinical adoption.

Cells segmentation and tracking benchmark (CTC) [206] consists of 2D and 3D time-lapse video sequences of fluorescent counterstained nuclei or cells moving on top or immersed in a substrate. The benchmark consists of 13 different datasets (8 for (2D) and 5 for (3D)). They can be either contrast-enhancing or fluorescence microscopy recordings of live cells and organisms. Table 5.1 shows the names and brief description for each datasets. Each dataset consists of two training and two testing videos. The training videos were provided with annotations, gold annotation (containing human-made reference annotations but not for all cells), and silver annotation (containing computer-generated reference annotations). The benchmark has different challenges: 1) Different appearances between datasets; 2) Low contrast between foreground and background; 3) The benchmark was taken in different light conditions and different image acquisition environments; 4) The ground-truth annotations for the training set are not fully provided for gold annotations and not accurate for silver annotations.

We participated in ISBI 2021 CTC-6 challenge, with over thirty teams reporting results on the CTC website [206] which is updated monthly. Not every method reported results for all datasets in the benchmark. We evaluated our pipeline on the eight 2D datasets for cell segmentation and tracking.

5.2.1 Cell Segmentation and Tracking Pipeline

We propose an end-to-end pipeline for accurate cell segmentation and tracking as shown in Figure 5.1. The goal is to localize and track different cell types in time-lapse video sequences. The pipeline consists of two main modules: cell segmentation and cell tracking modules. The cell segmentation module DMNet is designed to precisely localize and segment different cells, and the multi-cell tracking module M2Track uses a multi-step data association approach to efficiently track cells across frames.

Table 5.1: The name of cell segmentation and tracking challenge benchmark (CTC). The last columns show whether our algorithm is being implemented on videos or not

Datset name	Type	Description	Used?
BF-C2DL-HSC	2D	Mouse hematopoietic stem cells in hydrogel microwells	Yes
BF-C2DL-MuSC	2D	ouse muscle stem cells in hydrogel microwells	Yes
DIC-C2DH-HeLa	2D	HeLa cells on flat glass	Yes
Fluo-C2DL-MSC	2D	Rat mesenchymal stem cells on a flat polyacrylamide substrate	Yes
Fluo-N2DH-GOWT1	2D	GFP-GOWT1 mouse stem cells	Yes
Fluo-N2DL-HeLa	2D	HeLa cells stably expressing H2b-GFB	Yes
PhC-C2DH-U373	2D	Glioblastoma-astrocytoma U373 cells on a polyacrylamide substrate	Yes
PhC-C2DL-PSC	2D	Pancreatic Stem Cells on a Polystyrene substrate	Yes
Fluo-C3DH-A549	3D	GFP-actin-stained A549 Lung Cancer cells embedded in a Matrigel matrix	No
Fluo-C3DH-H157	3D	GFP-transfected H157 Lung Cancer cells embedded in a matrigel matrix	No
Fluo-C3DL-MDA231	3D	MDA231 human breast carcinoma cells infected with a pMSCV vector including the GFP sequence, embedded in a collagen matrix	No
Fluo-N3DH-CE	3D	C.elegans developing embryo	No
Fluo-N3DH-CHO	3D	Chinese Hamster Ovarian (CHO) nuclei overexpressing GFP-PCNA	No

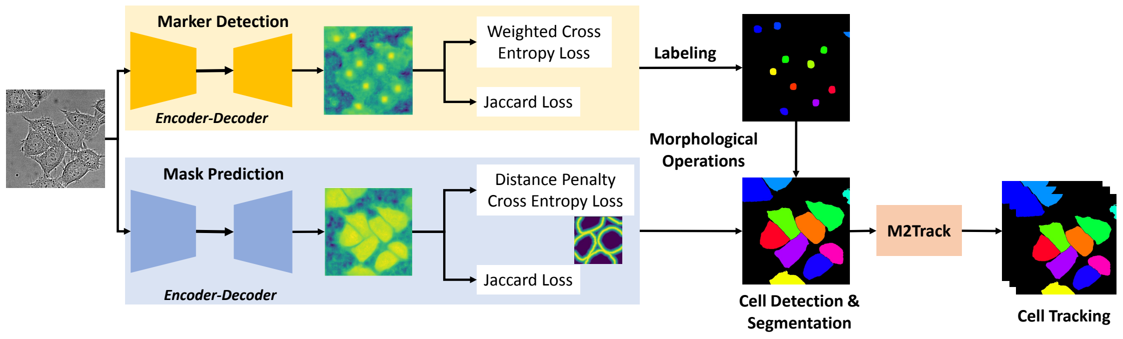


Figure 5.1: Overall pipeline with two streams DMNet for cell segmentation and M2Track for tracking.

For the cell segmentation module, the task is defined to find the segmentation mask of each cell. There are two streams in the proposed DMNet, one stream is designed for cell marker detection, and the other is designed for cell mask prediction, as shown in Figure 5.1.

Marker Detection Stream The marker-based loss function $L_{\text{marker}}(\cdot)$ is computed pixels with respect to the labeled marker annotations using a soft Jaccard and weighted cross-entropy loss functions:

$$L_{\text{marker}} = \alpha L_{\text{Jaccard}}(\cdot) + \beta L_{\text{wce}}(\cdot) \tag{5.1}$$

where α and β are used to balance the Jaccard loss L_{Jaccard} and weighted cross-entropy loss L_{wce}. The Jaccard loss is,

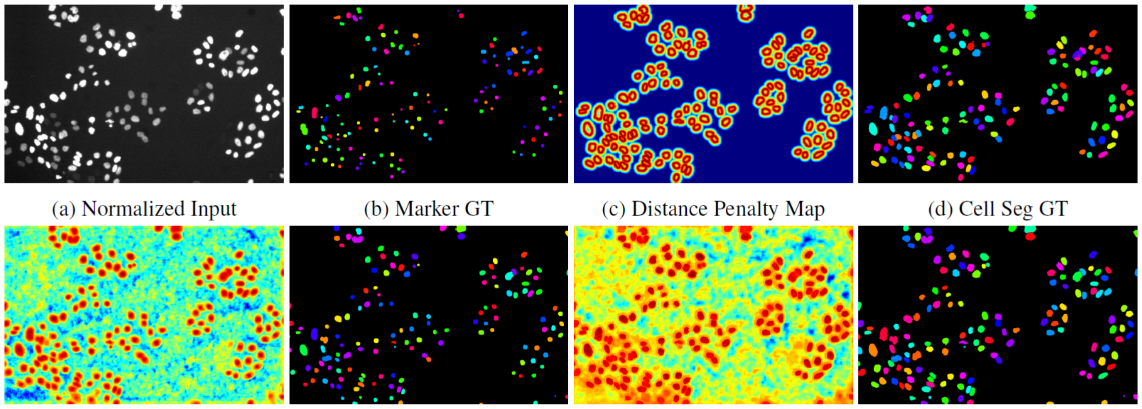
$$L_{\text{Jaccard}} = \frac{1}{N} \sum_{k=1}^{N} \frac{-\mathbf{y}_k \hat{\mathbf{y}}_k}{\mathbf{y}_k + \hat{\mathbf{y}}_k - \mathbf{y}_k \hat{\mathbf{y}}_k} \tag{5.2}$$

since the distribution of marker and non-marker pixels is highly biased, we use a class balanced cross-entropy loss, which is defined as:

$$L_{\text{wce}} = -\lambda_{-} \sum_{\mathbf{y}_k(i,j) \in Y_{-}} \log(1 - \hat{\mathbf{y}}_k) - \lambda_{+} \sum_{\mathbf{y}_k(i,j) \in Y_{+}} \log(\hat{\mathbf{y}}_k) \tag{5.3}$$

where each prediction map in the mini-batch of marker detection stream is $\hat{\mathbf{y}}_k$, of size $R \times C$, $\hat{\mathbf{y}}_k \in (0, 1)$ denotes a predicted marker map (see Figure 5.2 (e)), $\mathbf{y}_k$ is the groundtruth mask (see Figure 5.2 (b), $\mathbf{y}_k$ is the binarized version of it). $\lambda_{+} = \frac{|Y_{+}|}{(|Y_{+}|+|Y_{-}|)}$, $\lambda_{-} = \frac{|Y_{-}|}{(|Y_{+}|+|Y_{-}|)}$ balance the marker/non-marker pixels to control the weight of positive over negative samples.

For Mask Prediction Stream, the loss function L_{mask} is computed pixelwise with respect to the labeled mask segmentation annotations using a a soft Jaccard and


(a) Normalized Input (b) Marker GT (c) Distance Penalty Map (d) Cell Seg GT

(e) Marker Prediction (f) Labeled Marker Prediction (g) Mask Prediction (h) Cell Seg Prediction

Figure 5.2: Illustration of intermediate results in the DMNet workflow for cell segmentation: (a) Normalized raw input image, (b) Marker Ground-Truth for the supervision of the marker detection stream, (c) Distance penalty map in L_{dist}, (d) Cell segmentation ground-truth showing cells with tracking ids (binarized version provides supervision for mask prediction stream), (e) Marker prediction output from marker detection stream, (f) Labeled predicted markers after thresholding and connected component labeling, (g) Mask prediction output from mask prediction stream, and (h) Cell segmentation prediction using mask and marker, after splitting cells using marker guided morphological watershed algorithm.

distance penalized cross-entropy loss functions as:

$$L_{\text{mask}} = \alpha L_{\text{Jaccard}}(\cdot) + \beta L_{\text{dist}}(\cdot) \tag{5.4}$$

The L_{dist} is defined as:

$$L_{dist} = -\frac{1}{N} \sum_{k=1}^{N} \sum_{i=1}^{R} \sum_{j=1}^{C} (1 + \phi(i, j)) L_{ce} \tag{5.5}$$

where L_{ce} is the cross-entropy loss, describe as:

$$L_{ce} = \mathbf{m}_k(i, j) \log \hat{\mathbf{m}}_k(i, j) + (1 - \mathbf{m}_k(i, j)) \log(1 - \hat{\mathbf{m}}_k(i, j)) \tag{5.6}$$

Here each prediction map in the mini-batch of mask prediction stream is $\hat{\mathbf{m}}_k$ (see Figure 5.2 (g)), of size $R \times C$. The cross-entropy loss is modified by a distance penalty map ϕ, which inverses and normalizes the distance transform map D. The Euclidean distance transform map is computed as:

$$D^2(i,j) = \sum_i^R \sum_j^C (x(i,j) - b(i,j))^2 \tag{5.7}$$

where $b(i,j)$ is the location of a background pixel (value 0) that is closest to corresponding input points $x(i,j)$, where edge pixels of cells are 0, and remaining pixels are 1. Figure 5.2 (c) shows an example distance penalty map ϕ.

During the inference, both markers and masks are generated, and then the morphological operation watershed [207] is applied to the split cell mask guided by the generated markers. For each stream, the same Convolutional Neural Network (CNN) structure HRNet [208, 52] as the CNN model is used to learn the marker localization and mask prediction map since HRNet encodes rich representations of low-resolution and high-resolution information.

For the cell tracking module, the detected cells estimated by the DMNet segmentation module are used as input to the tracking module. The tracking module is a multi-step cascade data association pipeline shown in Figure 5.3,

The cascade data association has two steps: first, short-term tracking which is frame-to-frame data association and matching using mask intersection over union (IOU) score. Followed by the second step, long-term tracking, which is called the global data association step connects cells at the track level using spatial and temporal clues to re-link fragmented tracklets.

Short-Term Tracking: Short-term data association step, optimizes the associa-

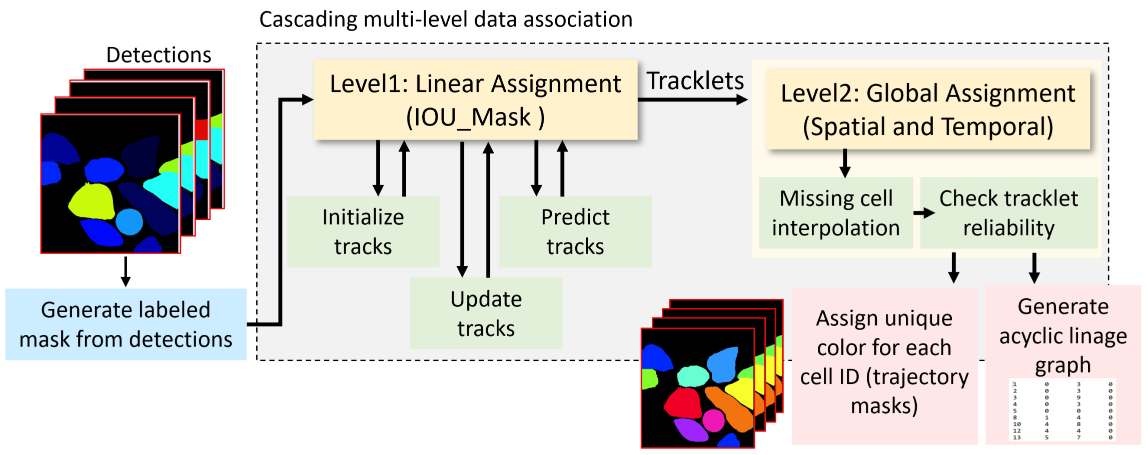

Cascading multi-level data association
Detections
Generate labeled mask from detections
Level1: Linear Assignment (IOU_Mask)
Initialize tracks
Update tracks
Predict tracks
Tracklets
Level2: Global Assignment (Spatial and Temporal)
Missing cell interpolation
Check tracklet reliability
Assign unique color for each cell ID (trajectory masks)
Generate acyclic linage graph

Figure 5.3: M2Track with intersection-over-union mask overlap matching for multi-cell tracking-by-detection. The two major modules are Level 1 for managing frame-to-frame linear assignments between detected cells and handling the entering and exiting cells, and Level 2 for tracklet linking to re-link missing or occluded cells.

tions of current detected cells D^t at frame t to the predicted track T^{t-1} at frame $t-1$, where the set of detections, $D^t = \{d_1, d_2, ..., d_N\}$ is assigned to the previously tracked objects $T^{t-1} = \{T_1, T_2, ..., T_M\}$, and T^{t-1} is the set of predicted cell trajectories from previous cell motion history computed using Kalman filter with constant velocity model, N is the number of the detected cells at frame t, M is the number of tracked cells at frame $t-1$. Mask IOU score is used to assign detection-to-track between following frames by minimizing a cost matrix using Munkres Hungarian algorithm [74] as:

$$\min_{b \in B} \sum_{i=1}^{m} \sum_{j=1}^{n} c^{ij} b^{ij} \tag{5.8}$$

where c_t^{ij} is an i row to j column entry on cost matrix representing the cost of assigning detection j to tracklet i at time t and its value represents the IOU between the area of i and j detect masks as:

$$c_t^{ij} = \frac{|A_i \cup A_j| - |A_i \cap A_j|}{|A_i \cup A_j|} \tag{5.9}$$

with constraints,

$$\sum_{i=1}^{m} b^{ij} = 1 \qquad j = 1, 2, ..., n;$$
$$\sum_{j=1}^{n} b^{ij} = 1 \qquad i = 1, 2, ..., m.$$

Circular gating regions around the predicted track positions are used to eliminate highly unlikely associations to reduce the computational cost, and to reduce false matches. Pairs of detection and tracks represent the results of minimum optimization. For each individual cell, a (one out of four) status (new track, linked track, lost track, and dead track) is assigned according to the assignment process. Since this step considers only information from consecutive frames, having false detections, occlusions, and matching ambiguities causes track fragmentation. A further step is important to improve the performance.

Long-Term Tracking: Problems during object detection or data association process result in implicit fragmentation of cells. Long-term tracking is used to re-link fragmented trajectories to produce longer tracks. Using information across long video segments can make this process expensive. Optimizing hypotheses at the track level rather than the object level reduces the computational cost of data assignment by gating uncertain hypotheses. Spatial distances and temporal information are used for filtering.

5.2.2 Evaluation Metrics

For evaluation the performance of the algorithms, qualified overall measurements for segmentation OP_{CSB} and tracking accuracy OP_{CTB} are used. The segmentation accuracy measurement OP_{CSB} composes of segmentation metric **SEG**, and detection metric **DET**, and tracking accuracy measurement OP_{CTB} composes of segmentation metric **SEG**, and tracking metric **TRA** were:

$$OP_{CSB} = 0.5 \times (SEG + DET) \tag{5.10}$$

$$OP_{CTB} = 0.5 \times (SEG + TRA) \tag{5.11}$$

where **SEG** based on the Jaccard similarity index J of the sets of pixels of matching objects:

$$SEG = J(GT, R) = \frac{GT \cap R}{GT \cup R} \tag{5.12}$$

where R denotes the set of pixels belonging to its matching segmented object. and GT the set of pixels belonging to ground truth. A ground truth object GT and a segmented object R are considered matching if the following condition holds

$$|GT \cap R| > 0.5.|GT| \tag{5.13}$$

The J is computed for each GT object in a video. It must always be in the [0,1] interval, where 1 means perfect match and 0 means no match. For **DET** metric, showed how accurately each given object has been identified, is computed as:

$$DET = 1 - min(AOGM_D, AOGM_{D_0})/AOGM_{D_0} \tag{5.14}$$

it is based on a comparison of the nodes of acyclic oriented graphs representing objects in both the GT and algorithm's outputs as described in [209]; Where $AOGM_D$ is the cost of transforming a set of objects provided by the algorithm into the set of GT objects; $AOGM_{D0}$ is the cost of creating the set of GT nodes from scratch (i.e., have an empty set of results). The normalization ensures that **DET** metric always falls in the [0,1] interval, with higher values corresponding to better detection performance. While the minimum operator in the numerator prevents from having a negative value when it is cheaper to create the GT set of objects from scratch than to transform the results set of objects into the GT one.

For tracking metric **TRA**, it is calculated by computing the number of steps required to match the ground-truth and predicted trajectory graphs:

$$TRA = 1 - \frac{\min\left(AOGM, AOGM_0\right)}{AOGM_0} \tag{5.15}$$

where $AOGM$ is the cost for generating a graph for the predicted trajectories. $SOGM_0$ is the cost for creating a predicted trajectory graph from scratch (i.e., assuming empty tracking results).

5.2.3 Evaluation and Experimental Results

Implementation Details: input images are pre-processed to enhance contrast using a z-score mapping. During training, the marker detection stream is trained with supervision using ground-truth of tracking markers, and the segmentation mask is supervised by silver-truth of annotations. Both the marker localization and mask prediction streams were trained on eight 2D datasets (see Tables 5.2, 5.3, 5.4) and five 3D datasets.

Table 5.2: DMNet cell segmentation performance ($\mathbf{OP_{CSB}}$) on CTC-6 of March 2021. All reported results are from the CTC Challenge website [206]. The first row is OP_{CSB} and the second row is ranking compared to other submitted algorithms. Not all methods reported results for all datasets which are shown as NA. Since CALT-US did not report results for Fluo-C2DL-MSC we provide two sets of Rankings – 8 datasets and 7 datasets for equivalent comparison. Rank Sum is the sum of all the ranks across cell types. DMNet consistently outperforms other methods on 2D cell segmentation.

Dataset	BF-C2DL -HSC	BF-C2DL -MuSC	DIC-C2DH -HeLa	Fluo-C2DL -MSC	Fluo-N2DH -GOWT1	Fluo-N2DL -HeLa	PhC-C2DH -U373	PhC-C2DL -PSC	Avg	Rank Sum(8,7)
KIT-Sch-GE [210]	0.905	0.878	0.850	0.686	0.895	0.938	0.927	0.859	0.893	
	1/14	1/14	14/27	6/32	20/42	11/40	15/30	1/33		69,63
PURD-US [211]	0.745	0.678	0.703	0.478	0.915	0.943	0.940	0.790	0.816	
	13/14	14/14	19/27	24/32	14/42	6/40	11/30	13/33		114,90
CALT-US [212]	0.901	0.852	0.925	–	0.948	0.915	0.959	0.703	0.886	
	2/14	4/14	1/27	–	3/42	18/40	1/30	25/33		NA,54
DMNet (Ours)	0.835	0.860	0.864	0.602	0.939	0.954	0.949	0.826	0.890	
	10/14	3/14	12/27	17/32	5/42	1/40	6/30	6/33		**60,43**

Table 5.3: DMNet cell tracking performance ($\mathbf{OP_{CTB}}$) on CTC-6 of March 2021. The first row is OP_{CTB} and the second row is ranking compared to other submitted algorithms. All reported results are from the CTC Challenge website [206]. Unreported results are shown as NA. DMNet consistently outperforms other methods on 2D cell tracking.

Dataset	BF-C2DL -HSC	BF-C2DL -MuSC	DIC-C2DH -HeLa	Fluo-C2DL -MSC	Fluo-N2DH -GOWT1	Fluo-N2DL -HeLa	PhC-C2DH -U373	PhC-C2DL -PSC	Avg	Rank Sum
KIT-Sch-GE	0.901	0.872	0.848	0.683	0.894	0.938	0.925	0.855	0.865	
	1/10	1/10	8/20	3/26	13/35	10/33	10/24	1/26		47
PURD-US	0.716	0.670	0.684	0.479	0.914	0.941	0.939	0.783	0.766	
	9/10	10/10	13/20	18/26	10/35	6/33	8/24	9/26		83
CALT-US	NA	NA	NA	NA	NA	NA	NA	NA	NA	
	NA	NA	NA	NA	NA	NA	NA	NA		NA
DMNet (Ours)	0.828	0.849	0.854	0.591	0.939	0.953	0.947	0.821	0.848	
	6/10	**2/10**	6/20	12/26	**2/35**	**1/33**	**3/24**	4/26		**36**

When using 3D datasets, we used one frame or slice per volume with the most annotated labels for training. Input images are resized and then cropped for training. Resize scale factor for each dataset are: Fluo-C2DL-MSC:0.35, Fluo-C3DH-H157:0.35, Fluo-C3DL-MDA231:2, Fluo-N3DH-CE:0.5, Fluo-N3DH-CHO:0.6, PhC-

Table 5.4: Performance of DMNet segmentation and tracking pipeline on CTC-6 of March 2021. For each performance metric, the first row is the accuracy metric and the second row is the ranking compared to all other submitted algorithms (as of March 2021). The top three performances of DMNet by cell type are bolded.

Dataset	BF-C2DL -HSC	BF-C2DL -MuSC	DIC-C2DH -HeLa	Fluo-C2DL -MSC	Fluo-N2DH -GOWT1	Fluo-N2DL -HeLa	PhC-C2DH -U373	PhC-C2DL -PSC
OP_{CTB}	0.828	0.849	0.854	0.591	0.939	0.953	0.947	0.821
	6/10	**2/10**	6/20	12/26	**2/35**	**1/33**	**3/24**	4/26
OP_{CSB}	0.835	0.860	0.864	0.602	0.939	0.954	0.949	0.826
	10/14	**3/14**	12/27	17/32	5/42	**1/40**	6/30	6/33
DET	0.971	0.979	0.926	0.681	0.946	0.985	0.975	0.945
	8/14	**2/14**	13/27	17/32	10/42	10/40	16/30	9/33
SEG	0.699	0.742	0.802	0.522	0.931	0.923	0.923	0.708
	6/10	**2/10**	6/20	13/26	**1/35**	**1/33**	**3/24**	4/26
TRA	0.957	0.957	0.907	0.661	0.946	0.983	0.972	0.933
	4/10	4/10	9/20	12/26	7/35	10/33	11/24	8/26

C2DL-PSC:3, BF-C2DL-MuSC:0.75, BF-C2DL-HSC:0.75. We crop patches with the size of 256×256 from images in each dataset to train the networks, except BF-C2DL-HSC, BF-C2DL-MuSC which we crop patches of 512×512. Regular data augmentation strategies were used including rotation, flip, and scale from 0.8 to 1.5 for each sample. Hyperparameters are learning rate of 0.001 with Adam Optimizer for training both streams for 300 epochs with, $\alpha = 2.5$ and $\beta = 10$.

Comparison on CTC-6 Benchmark: DMNet+M2Track performance is compared with state-of-the-art methods on *Cell Segmentation* and *Cell Tracking Tasks*. Since not every method reported results for all 2D datasets, we show the three most competitive methods KIT-Sch-GE [210], PURD-US [211] and CALT-US [212], which have results for almost all eight 2D cell microscopy videos.

DMNet is robust and achieves state-of-the-art cell detection and segmentation performance on all eight 2D CTC-6 datasets. In Table 5.2, we compare our DMNet with the state-of-the-art methods on the CTC-6 challenge. We compute the rank sum of each method of $\mathbf{OP_{CSB}}$ on all the datasets. Because CALT-US did not report results on Fluo-C2DL-MSC, we put NA in that column, and compute the rank-sum on eight datasets and seven datasets respectively as shown in the last column of Table 5.2. Our DMNet achieves the best results on all the 2D datasets with a rank-sum of 60 (eight datasets) and 43 (seven datasets), which demonstrates the robustness and effectiveness on 2D cell segmentation. DMNet+M2Track is robust and achieves state-of-the-art cell tracking performance on all eight 2D CTC datasets. In Table 5.3, we compute the rank sum of each method of $\mathbf{OP_{CTB}}$ on all the datasets. Because CALT-US did not perform cell tracking, therefore it is empty in this table. Our DMNet + M2Track achieves the best rank-sum with 36, which demonstrates the robustness and effectiveness on 2D cell tracking task. Table 5.4 shows the ranks of our pipeline compared to the other participants on CTC-6. Our pipeline ranked in the top three on four out of the eight 2D cell type microscopy videos. Not every method provided results for all cell types, whereas DMNet+M2Track results are given for all videos.

Figure 5.4 shows the results of DMNet+M2Track segmentation and tracking pipeline for three cell types. The first column shows the Raw Input image for three cell types, which are typically low contrast, with dense, clustered cells and small object sizes. A z-score normalization is applied to the raw input image to remove outliers. The raw input is stretched to increase image contrast, as shown in the second column Normalized Input. We show the ground truth segmentation with tracking ids as Tracking GT in the third column. The fourth and fifth columns are the final marker detections and cell tracking predictions for all the cells in each frame of the video. We can clearly see in Figure 5.4 column (d) that DMNet accurately predicts

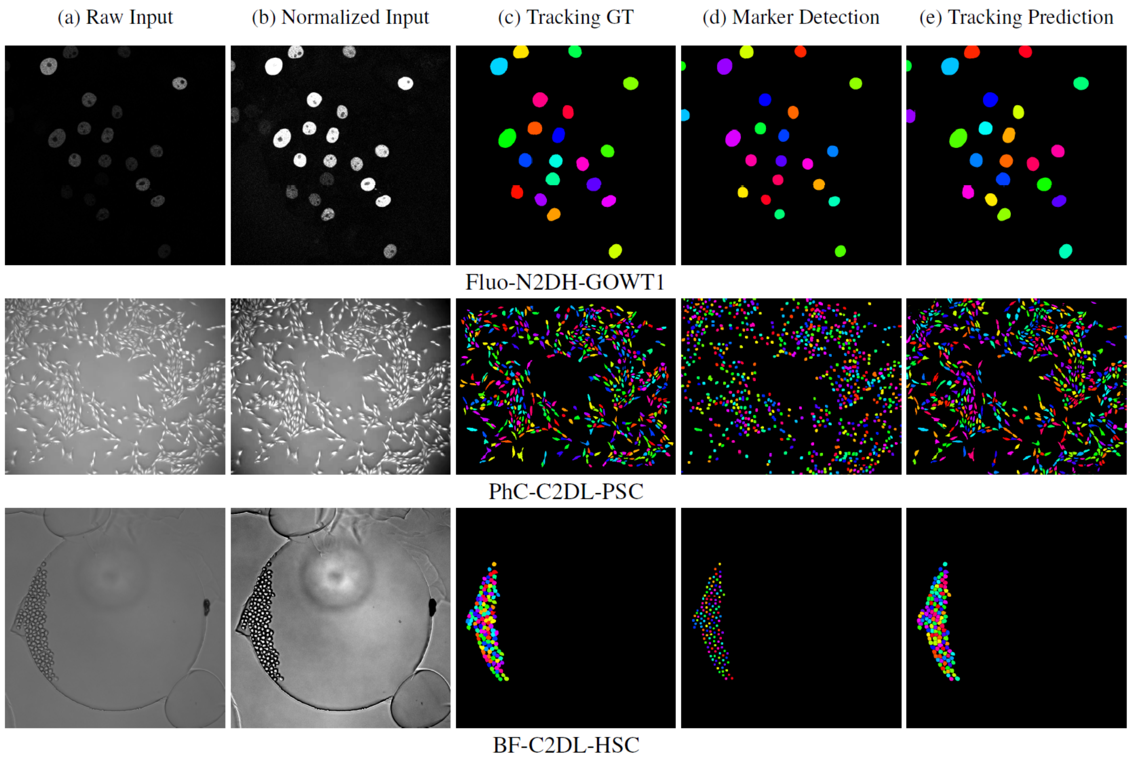


Figure 5.4: Visualization of DMNet+M2Track segmentation and tracking results for three cell types including Fluo-N2DH-GOWT1, PhC-C2DL-PSC, and BF-C2DL-HSC exhibiting a range of cell sizes and densities.

and separates the cell markers. Hence, using labeled markers as guidance for the watershed algorithm to split the predicted cell masks results in consistently satisfactory cell segmentation results.

5.3 Cell Tracking with Mitosis Detection Benchmark

CTMC-v1 is a human-annotated live-cell imaging dataset for cell tracking benchmark. CTMC-v1 dataset [82] provides fully manual annotations in the format of bounding boxes that are persistently tracked over time. Additionally, CTMC-v1 dataset features a single imaging modality with different cell types. The benchmark consists of 86 live-cell imaging videos (47 training + 39 testing). Videos represent 14 different cell lines of various shapes and sizes, with each cell annotated using bounding boxes with 400×320 frame resolution. Table 5.5 summarizes the statistics of the dataset and Figure 5.5 shows examples of the given data. This dataset is part of the multi-cell tracking with mitosis challenge [3].

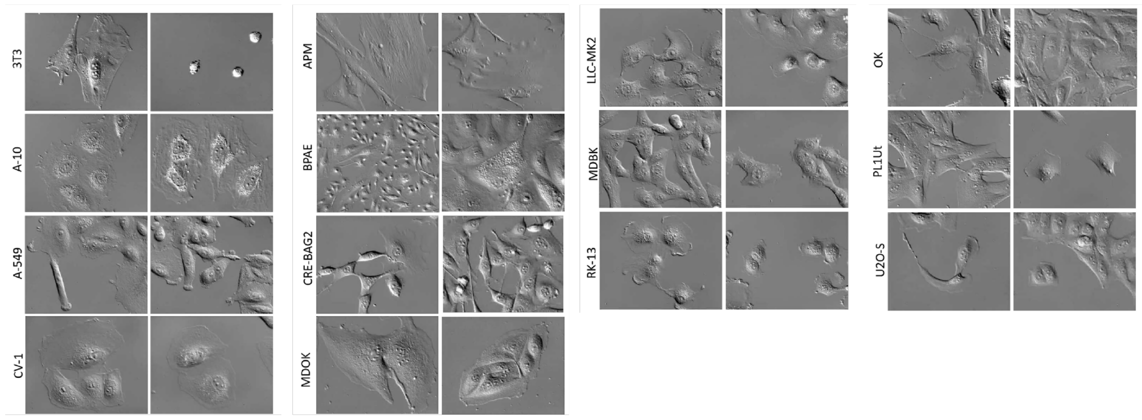


Figure 5.5: Example frames for each of the 14 cell types in the CTMC-v1 dataset.

CTMC-v1 benchmark has a number of challenges, such as a wide range of cell lines of various shapes and sizes, touching or overlapping cells, indistinct deformable boundaries, high cell density, unpredictable motion of individual cells, dynamic interactions between cells, cell mitosis, cell overlap, cell-to-cell interactions and shape variations, low contrast and resolution with unclear cell edges with MPEG compression artifacts. Additionally, each cell type provides an average of two sequences, both of which contain a significant amount of similar frames. This proposes a poorly distributed dataset despite its large scale, which, in turn, is a major cause of model over-fitting.

Table 5.5: Summary table that describes the CTMC-v1: cell tracking and mitosis challenge dataset.

Imaging modality	DIC: differential interference contrast microscopy
Number of sequences	86 (49 for training and 37 for testing, 4-10 sequences per cell type in total)
Number of cell types	14
Number of cells (tracks)	2,900 in total. The number of cells per sequence varies from 1-185
Number of frames	152,584 in total. The number of frames per sequence varies from 300 to 4,440.
Number of detections	2,045,834 in total. The number of detections per sequence varies from 1,196 to 172,074
Cell density	13 cells per frame on average. Density per cell type ranges from 3.42 to 27.88
Mitosis events	457 mitosis events. Ranges from 2 to 115 per cell type

5.3.1 Cell Segmentation and Tracking Pipeline

We propose an end-to-end pipeline for accurate cell detection and tracking. The goal is to localize and track different cell types in CTMV-v1 benchmark. The pipeline consists of two main modules: first, cell detection module EDF-YOLOv3, a model-agnostic detection ensemble framework (EDF) that shows great robustness to over-fitting, second, a multi-cell tracking module, by employing M2Track algorithm to link cell detections and track cell lineage.

For cell detection, Yolov3 was adopted to detect the cells on the benchmark. We train 14 folds; in each fold, one cell type out for validation was left and the remaining 13 cell types were used for training. By excluding a whole cell type from training and using it for validation, we trained our base models to generalize to new cell types. The utilized YOLOv3 model has initialized with pre-trained weights from the ImageNet [155, 213] dataset. The final prediction layer is modified for a single "cell" class; therefore it is initialized randomly using Kaiming He initialization [214]. All training in this work uses the same hyper-parameters: the mini-batch size of 48, an input resolution of 416×416, a learning rate of 10^{-3}. We allow the training to run for a maximum of $500,000$ iterations and we multiply the learning rate by 0.1 at step $400,000$ and once more at step $450,000$. Figure 5.6 show an overview of the ensemble detection framework EDF-YOLOv3.

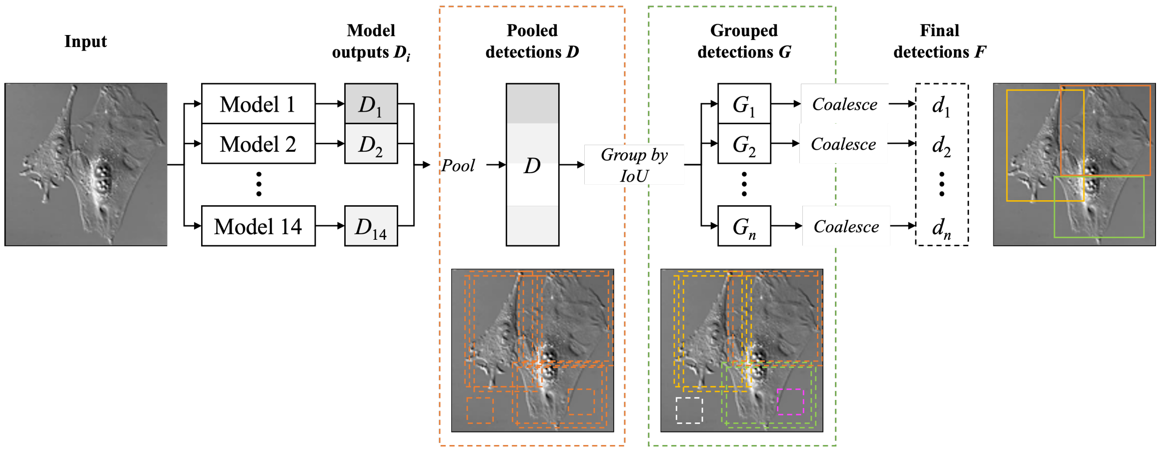


Figure 5.6: General overview of the proposed ensemble detection framework EDF-YOLOv3. 14 detection sets D_i are generated from 14 detection models (YOLOv3 in our experiments) trained on different subsets of the training dataset. These sets are pooled together in a single set D, then grouped together such that for each pair of detections in the same group $d_i, d_j \in G_k$, $\mathbf{IoU}(d_i, d_j) > 0.5$. Groups with fewer detections than half the number of models (i.e., 7) are discarded. The remaining groups are coalesced into a single detection by picking the detection with the highest confidence score.

For cell tracking, the detected cells from EDF-YOLOv3 are used as input to the tracking module. The tracking module is a multi-step cascade data association pipeline M2Track shown in Figure 5.7. M2Track performs cascade data association in two steps:

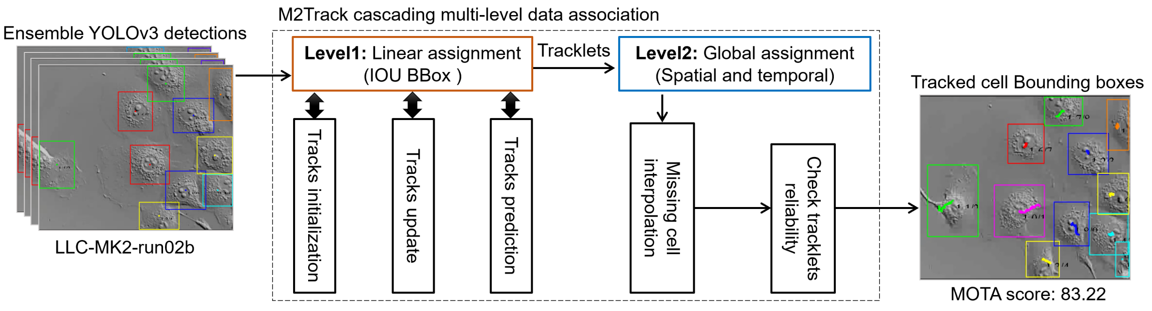

M2Track cascading multi-level data association
Ensemble YOLOv3 detections
Level1: Linear assignment (IOU BBox)
Tracklets
Level2: Global assignment (Spatial and temporal)
Tracked cell Bounding boxes
Tracks initialization
Tracks update
Tracks prediction
Missing cell interpolation
Check tracklets reliability
LLC-MK2-run02b
MOTA score: 83.22

Figure 5.7: An overview diagram of M2Track implemented on CTMC-v1 benchmark.

Short-term tracking: is a frame-to-frame data association and matching using bounding box intersection over union (**IoU**) score to generate cost matrix. The **IoU** cost matrix is used to optimize the associations of current N detected cells in D^t set at frame t to the M predicted tracks in T^{t-1} set at frame $t-1$, where the set of detecitons, $D^t = \{d_1^t, d_2^t, .., d_N^t\}$ is assigned to the previously tracked objects $T^{t-1} = \{T_1^{t-1}, T_2^{t-1}, .., T_M^{t-1}\}$, and T^{t-1} is the set of predicted cell trajectories previously tracked. Kalman filter [75] with constant velocity model is used to predict the motion of the cell in each frame. The optimization is implemented by minimizing the cost matrix using Munkres Hungarian algorithm [74] as:

$$\min_{b\in B} \sum_{i=1}^{m} \sum_{j=1}^{n} c^{ij} b^{ij} \tag{5.16}$$

where c_t^{ij} is an i row to j column entry on cost matrix representing the cost of assigning detection j to tracklet i at time t and its value represents the IOU between i and j bounding box detect cells as:

$$c_t^{ij} = \frac{|d_i \cup d_j| - |d_i \cap d_j|}{|d_i \cup d_j|} \tag{5.17}$$

and b is a binary assignment matrix to ensure the rows and columns of b are summed to 1, with constraints,

$$\sum_{i=1}^{m} b^{ij} = 1 \qquad j = 1, 2, ..., n;$$
$$\sum_{j=1}^{n} b^{ij} = 1 \qquad i = 1, 2, ..., m.$$

To reduce false matches, to eliminate low certainty association, and to reduce the computational cost, spatial circular gating regions around the predicted track positions are used. Association cost matrix decision assigns one out of four statuses (new track, linked track, lost track, and dead track) for each individual cell. Since this step considers only information from consecutive frames, having false detections, occlusions, and matching ambiguities causes track fragmentation. A further step is important to improve the performance.

Long-term tracking: the second step of M2Track, uses information across long video segments. Increasing the number of frames or the number of cells makes this process expensive. Our approach optimizes fragmented tracklets re-linking by connecting cells at the track level rather than object level using spatial and temporal clues. Cell interpolation is used to recover missing cells due to miss-detection or occlusion events.

5.3.2 Evaluation and Experimental Results

In order to show the difficulty of generalization, we report the results for the different methods on both the training and testing set in Table 5.6. The results for the hidden test sets were generated using the public benchmark [215, 216, 217, 64, 218]. Table 5.6 shows that a cell-specific model – that is trained on specific cell types – performs best on training data and worst on testing data. This is a clear indication that training a different model for each cell type yields over-fitting. In contrast, a general model (using different approaches) consistently results in worse performance in training and superior performance for the hidden test data.

Table 5.6: Summary result table showing the MOTA score for different proposed methods compared to the baseline. **YOLOv3 Model I**: Pool data together and select randomly 80% for training and 20% for validation (no ensemble). **YOLOv3 Model II**: Leave one sequence out from each cell type for validation (no ensemble). **YOLOv3 Model III**: Leave one cell type out for validation (no ensemble); in our experiments, the cell type that was left is 3T3. All of the methods below (excluding the baseline) use M2Track for tracking.

	Avg. MOTA per cell type		Avg. MOTA per sequence		Avg. MOTA per tracklet	
Method	train	test	train	test	train	test
Baseline (FasterRCNN + Tracktor) [82]	-	23.85	-	22.24	-	35.10
Cell-type specific YOLOv3	**90.37**	26.60	**89.98**	22.90	**88.92**	24.20
YOLOv3 Model I	73.10	35.40	75.60	34.60	75.30	37.50
YOLOv3 Model II	89.96	38.92	89.80	37.53	88.86	44.72
YOLOv3 Model III	63.73	33.70	66.33	34.04	65.64	42.34
EDF-YOLOv3 (ours)	80.83	**48.08**	81.41	**48.60**	77.74	**50.55**

It can also be observed how deep detection networks (in our case YOLOv3) are sensitive to the way data is split into training and validation. We can see that random splitting (YOLOv3 Model I) yields a baseline score of 37.50. Leaving one sequence out from each cell type for validation (YOLOv3 Model II) greatly improves generalizability; giving a MOTA score of 44.72. Similarly, leaving an entire cell type for validation (YOLOv3 Model III) can also yield to a comparable improvement, giving a MOTA score of 42.34. Finally, our ensemble detection framework applied to YOLOv3 (EDF-YOLOv3) produces a state-of-the-art MOTA score on the public leader board with a MOTA score of 50.55.

We conclude that leaving a cell type for validation to train a deep detection model

Table 5.7: Tracker performance comparison as reported on Cell Tracking with Mitosis Detection Challenge website [3].

Tracker	**Detection**	**MOTA↑**	**IDF1↑**	**TRA↑**	**MT↑**	**ML↓**	**FP↓**	**FN↓**	**IDS↓**	**FRAG↓**
M2Track	EDF-YOLOv3 (ours)	**50.6**	**56.7**	**52.51**	**428**	174	158,593	**309,015**	**2,118**	**10,453**
M2Track	DMNet [35]	39.0	41.0	36.95	219	317	**109,689**	466,667	3,142	20,829
Tracktor [82]	FasterRCNN [188]	35.1	50.5	51.95	427	**126**	299,941	312,368	4,318	25,992

can greatly boost the model's generalizability. Furthermore, applying this method 14 times with each of the cell types and combining the resulting models into one using our EDF method can produce the best cell detection framework on the CTMC-v1 dataset.

Since tracking performance is affected by the quality of the object detection algorithm used. Table 5.7 shows tracking performance as reported on Cell Tracking and Mitosis Detection Challenge website [3]. Our EDF-YOLOv3 method for detection combined with M2Track tracker outperforms in most of multi-object tracking metrics **MOTA**, **IDF1**, **TRA**, **MT**, **FN**, **IDS**, and **FRAG**. Our team achieved the top one rank on the Computer Vision for Microscopy Image Analysis (CVMI) workshop at CVPR 2021.

Chapter 6

Multi-object Tracking for Ground-based/ Real-world Videos

6.1 Overview

In recent years, with the increase accessing to social media networks, and the fact that cameras are not exclusive anymore to professional usage in real-world life. Computer vision approaches became more in demand, such as video enhancement and filtering, video registration, object recognition, object detection, and moving object tracking. Ground-based/ real-world videos are the most popular types of videos that are used for multi-object tracking. Especially, modern object tracking methods can be applied to real-time video streams of basically any camera. Therefore, performing object tracking requires feeding the individual frames to the tracking approach, and applying some optimization techniques such as frame skipping or parallelized processing to improve object tracking performance with the real-time video feed from one or multiple cameras. There are many challenges with using real-world videos for multi-object tracking such as frequent object deformations, rapid scale and appearance changes, shadow, and illumination artifacts, and fully or partially object occlusion.

6.2 UA-DETRAC Bechmark

UA-DETRAC is a challenging real-world multi-object detection and multi-object tracking benchmark [60]. UA-DETRAC, University of Albany DETection and tRACking (UA-DETRAC) benchmark dataset, consists of 100 video sequences (60 for training and 40 for testing). The videos are selected from over 10 hours of videos taken at 24 different locations, representing various common traffic types and conditions including urban highways, traffic crossings and T-junctions. The videos are recorded at 25fps, with a resolution of 960×540 pixels. The videos, 140,000 frames, were manually annotated for 8,250 vehicles in a total of 1.21 million labeled bounding boxes. Besides the bounding boxes, several attributes were annotated. 1) Vehicle category: car, bus, van, and others. 2) Weather: cloudy, night, sunny, and rainy. 3) Scale: according to the square root of vehicles' area, small, medium, and large vehicle size categories are used. 4) Occlusion ratio: the fraction of the vehicle bounding box being occluded is used to define occlusion ratio with no occlusion, partial occlusion, and heavy occlusion categories. 5) Truncation ratio: shows the degree of vehicle parts that are outside the frame. The UA-DETRAC benchmark is designed to be used with object detection and multi-object tracking applications. The UA-DETRAC dataset has videos with large variations and challenges in scale, pose and illumination, occlusion, and background clutter. In 2017 and 2018, Advanced Video and Signal based Surveillance

(AVSS) conference held a workshop on Advanced Traffic Monitoring, in conjunction with the International Workshop on Traffic and Street Surveillance for Safety and Security (IWT4S) on UA-DETRAC benchmark to evaluate the state-of-the-art object detection and multi-object tracking algorithms using evaluation protocol. The UA-DETRAC protocol considers object detection and tracking jointly to evaluate the performance of multi-object tracking systems.

6.2.1 Evaluation Metrics

In [60], UA-DETRAC evaluation protocol proposed a new evaluation metrics which are **PR-MOTA**, **PR-MOTP**, **PR-MT**, **PR-ML**, **PR-IDS**, **PR-FM**, **PR-FP**, and **PR-FN**. The new metrics are different than the classic metrics (i.e. **FP, FN, IDS, FM, MT, ML, MOTA, MOTP**) that are described in previous chapters. The proposed UA-DETRAC evaluation protocol considers object detection and tracking jointly for evaluation. The protocol considers the relation between object detection (recall, and precision) metrics and the classic object tracking metrics. According to the evaluation protocol for the challenge, final evaluations are computed by applying the competitive trackers on a set of object detection results obtained by different detection thresholds corresponding to different detection precision and recall levels. A robust object tracking method is expected to perform well even with detection score variation.

6.2.2 UA-DETRAC Benchmark Implementation and Experimental Results

We have tested and evaluated our M2Track tracker on UA-DETRAC-test set consisting of 40 challenging real-world traffic videos (10 videos for beginner level (easy), 30 videos for experienced level (medium and hard) as classified by [60]). Our tracker participated in 2017 and 2018 challenge workshops as described below:

Participating in AVSS17 challenge on UA-DETRAC set: using M2Track-L2 that combines the first two levels from M2Track tracker pipeline, short-term local association and long-term global association. We incorporate CompACT [54] object detection output to be input to M2Track-L2 tracker. The results of the comparison between our tracker to the other participants that use CompACT object detection algorithm are shown in Table 6.1. Our rank was the second out of seven participants in the easy level and the fifth out of eleven in the experience level. The table is described in detail in the workshop challenge report [139].

Table 6.1: Tracker performance comparison against AVSS2017 challenge participants on 40 video sequences for the beginner (easy)/ experienced (medium and hard) levels in various environmental conditions. " − " indicates the data is not available. The best two high ranks performance for each metric is marked in bold. The performance of the trackers is presented according to [139]. The arrows indicate whether the metric for the specific column is better when high or low.

Trackers	**Rank↓**	**PR-MOTA↑**	**PR-MOTP↑**	**PR-MT↑**	**PR-ML↓**	**PR-IDS↓**	**PR-FM↓**	**PR-FP↓**	**PR-FN↓**
GOG[47]	5/7	23.9/11.7	**47.4**/34.4	20.5/10.8	21.0/21.1	829.9/2571.2	776.2/2463.8	6276.5/25352.8	36738.3/145257.5
CEM[219]	6/9	8.1/4.5	44.2/33.2	3.8/2.6	40.9/34.5	**73.7/198.1**	**88.3/267.5**	**3236.0/9047.6**	60393.3/200703.1
IHTLS[187]	7/11	20.8/8.7	46.5/34.2	20.2/10.7	21.6/21.1	178.0/774.0	735.8/2835.9	10484.0/42814.2	37172.1/145188.5
H^{2}T[220]	5/6	21.8/10.1	44.0/33.6	21.7/11.5	21.7/20.3	162.9/687.8	**191.7/922.2**	10278.4/41193.8	36115.2/139703.2
CMOT[42]	3/3	22.5/10.3	45.9/33.4	**23.3**/12.6	20.0/19.7	40.7/**243.2**	254.1/1255.9	11424.4/45619.6	34134.9/**134568.6**
HGFT[221]	-/8	-/**12.1**	-/33.5	-/10.4	-/21.5	-/1927.5	-/2141.0	-/24160.0	-/145262.2
GM-PHD[222]	-/4	21.8/10.9	**47.6/35.0**	16.2/**15.1**	20.4/21.6	641.8/556.4	2038.5/1674.9	37963.0/29687.1	186043.9/147257.0
JTEGCTD[139]	1/1	**28.4/14.2**	47.1/34.4	23.1/**13.5**	**18.3/18.7**	**69.4**/415.3	260.6/1345.7	**5034.0**/26221.8	**33093/133867.4**
MTT[51]	-/10	-/12.0	-/**35.7**	-/7.7	-/23.2	-/814.7	-/3158.9	-/**14016.8**	-/156997.0
GMPHD-KCF[223]	-/2	-/12.0	-/33.8	-/10.8	-/**19.5**	-/648.8	-/1300.2	-/30518.1	-/140669.4
M2Track-L2(ours)	2/5	**25.2**/10.7	45.8/33.8	**23.8**/11.9	**15.8**/20.0	179.9/514.7	590.6/1705.5	10155.4/35624.7	**32742.9**/142110.0

Our tracking results were compared to ten MOT trackers participated on the challenge, including GOG [47], CEM [219], IHTLS [187], H^{2}T [220], GMPHD-KCF [223], GM-PHD[222], HGFT [221], MTT [51], JTEGCTD [139], and CMOT [42]. Our tracker outperforms the other trackers in term of **PR-MOTA**, **PR-MT**, **PR-ML**, and **PR-FN** in beginner level; and the results are comparable for the remaining metrics. Using **RANK** metric which is the average ranking for the all evaluation metrics, our tracker has scored the second place in beginner level and the fifth place for the experienced level.

Table 6.2 summarizes frame rates (in terms of frames per second) for our tracker and other state-of-the-art trackers provided in [139]. The frame rate for our tracker has been obtained by averaging frame rates on 40 sequences of UA-DETRAC-test set. Our tracker had outperformed other competitive trackers after GM-PHD[222] tracker. Our simple feature set combined with a two-step distance-only local and distance appearance combined global data association scheme allows high frame rates while still preserving the reliability and accuracy of our tracker.

Table 6.2: Average running speed (in FPS) of the object tracking algorithms on the UA-DETRAC-test benchmark. " − " indicates the data is not available, and × indicates the GPU is not used. The two highest trackers in terms of speed are shown in bold. The running speed performance of the trackers is presented according to [139]

Trackers	**Codes**	**CPU**	**RAM**	**Frequency**	**GPU**	**Speed**	**Rank**
GOG	Matlab	Intel i7-3520M	16GB	2.90GHz	×	389.51	3
CEM	Matlab	Intel i7-3520M	16GB	2.90GHz	×	4.62	8
IHTLS	Matlab	Intel i7-3520M	16GB	2.90GHz	×	19.79	7
H2T	C++	Intel i7-3520M	16GB	2.90GHz	×	3.02	11
CMOT	Matlab	Intel i7-3520M	16GB	2.90GHz	×	3.79	10
HGFT	Matlab	-	32GB	-	×	3.97	9
GM-PHD	C++	Intel i7-4770	16GB	3.50GHz	×	**947.24**	1
JTEGCTD	Matlab	Intel i7-3720QM	8GB	2.70GHz	×	60.38	4
MTT	Python, C++	Intel E5-2650	128GB	2.00GHz	TitanX	24.30	6
GMPHD-KCF	C++	Intel i7-4770	32GB	3.50GHz	×	24.60	5
M2Track-L2	Matlab	Intel i7-4720	16GB	2.60GHz	×	**471.28**	2

Participating in AVSS18 Challenge on UA-DETRAC dataset: We have

Table 6.3: Tracker performance comparison against AVSS2018 challenge participants on 40 video sequences for the beginner (easy)/ experienced (medium and hard) levels in various environmental conditions. " − " indicates the data is not available. The best two high ranks performance for each metric is marked in bold. The performance of the trackers is presented according to [93]. The arrows indicate whether the metric for the specific column is better when high or low.

Tracker	**Rank↓**	**PR-MOTA↑**	**PR-MOTP↑**	**PR-MT↑**	**PR-ML↓**	**PR-IDS↓**	**PR-FM↓**	**PR-FP↓**	**PR-FN↓**
GOGGOG[47]	3/6	**23.9**/11.7	**47.4/34.4**	20.5/10.8	21.0/21.1	829.9/2571.2	776.2/2463.8	6276.5/25352.8	36738.3/145257.5
JTEGCTD[139]	1/1	28.4/**14.2**	47.1/**34.4**	**23.1/13.5**	**18.3%/18.7%**	**69.4/415.3**	**260.6**/1345.7	**5034.0**/26221.8	**33093.8/133867.4**
DMC[224]	-/2	-/**14.6**	-/34.1	-/11.6%	-/20.6%	-/908.3	-/**1287.4**	-/**16056.7**	-/141463.2
GMMA[225]	-/4	-/12.3	-/34.3	-/10.8	-/21.0	-/627.5	-/2423.7	-/25577.4	-/**144148.9**
M2Track_L2 [100]	2/5	25.2/10.7	45.8/33.8	**23.8%/11.9%**	**15.8%/20.0%**	179.9/514.7	590.6/1705.5	10155.4/35624.7	**32742.9**/142110.0
M2Track [94]	2/3	**25.9**/12.1	**47.2/35.0**	15.0/7.7	20.6/24.8	**91.8/378.3**	**323.7/947.5**	**2485.2/8241.0**	38820.9/162937.6

tested and evaluated our tracking-by-detection multi-object tracker M2Track that uses three-level modules cascade for efficient data association (local, global, and occlusion handling). We incorporate CompACT [54] object detection output to be input to M2Track tracker. The results of comparison between our tracker to the other participants that use CompACT object detection algorithm are shown in Table 6.3. We denote M2Track tracker to be M2Track-L2, and M2Trck to differentiate between the two versions that participated in AVSS17 and AVSS18 respectively. M2Track-L2 uses two modules for data association (local, and global), while M2Track uses three-level modules for data association (local, global, and occlusion handling). Table 6.3 describes in detail the results that showed in the workshop challenge report [93].

Our tracking results were compared to four MOT trackers participant on the challenge, including GOG [47], JTEGCTD [139], DMC [224], GMMA[225], and our two level module M2Track-L2 tracker. Our tracker M2Track outperforms the other trackers in term of **PR-MOTA**, **PR-MOTP**, **PR-IDS**, **PR-FM**, and **PR-FP** in beginner level; and **PR-MOTP**, **PR-IDS**, **PR-DM**, and **PR-FP** in experienced level set; and the results are comparable for the remaining metrics. Using **RANK** metric, our tracker has scored the second place in beginner level and the third place for the experienced level.

In terms of extending M2Track-L2 to M2Track, as shown in Table 6.3, the performance of the tracker is improved in terms of identity switches **PR-IDS** and tracklet fragmentation **PR-FM**, due to the third level of our pipeline (occlusion handling), which links the fragmented tracklets. **PR-FP** metric also increased because the target bounding boxes are predicted and recovered during occlusion handling. Figure 6.1 shows sample results.

We have analyzed the contribution of different module components on our tracker to the overall performance by systematically enabling/disabling different components. We used UA-DETRAC-train set for evaluation. Our tests show that all components contribute to better-accumulated performance. Figure 6.2 shows the contribution of each tracker component evaluated on UA-DETRAC-train set. For example, in the experiment shown on the fifth bar in Figure 6.2, disabling the CN-to-CN color correlation weight matrix $\mathcal{W}_{CN}$ and assuming uncorrelated color codes, results in a drop in PR-MOTA metric from 0.84 to 0.8.

6.3 Multi-Object Tracking Benchmark (MOT16)

MOT16 benchmark [4] consists of 14 video sequences (7 for training and 7 for testing) with a total of 11235 frames. MOT2016 sequences were collected to create a framework for the standardized evaluation of multi-object tracking methods. The dataset focuses on pedestrian tracking in different scenarios. The sequences have different weather conditions, camera motion, crowded scenarios, and different viewpoints. Around 215,166 object detection have been annotated. The ground truth

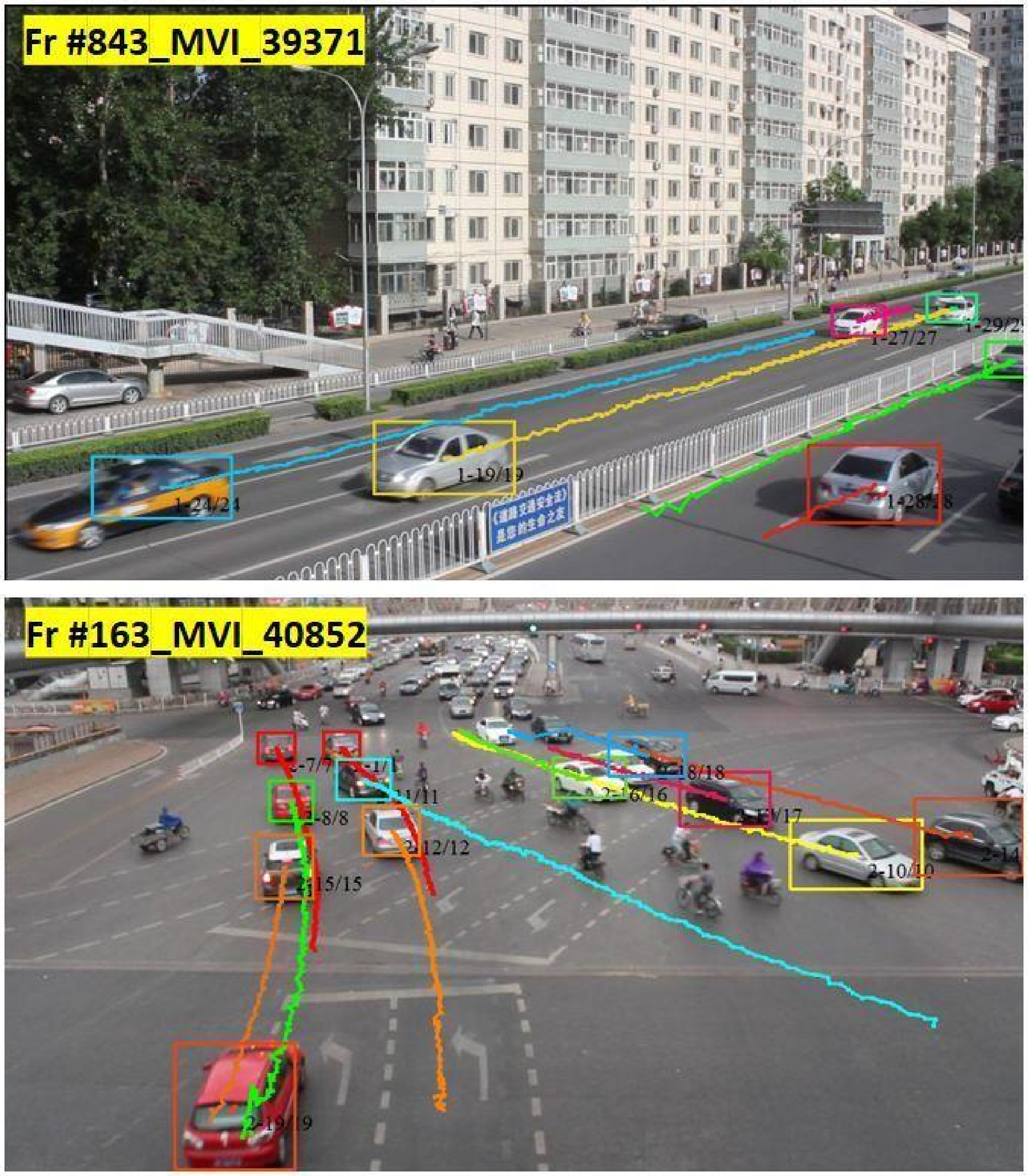


Figure 6.1: Sample track results from two different sequences from UA-DETRAC-test set.

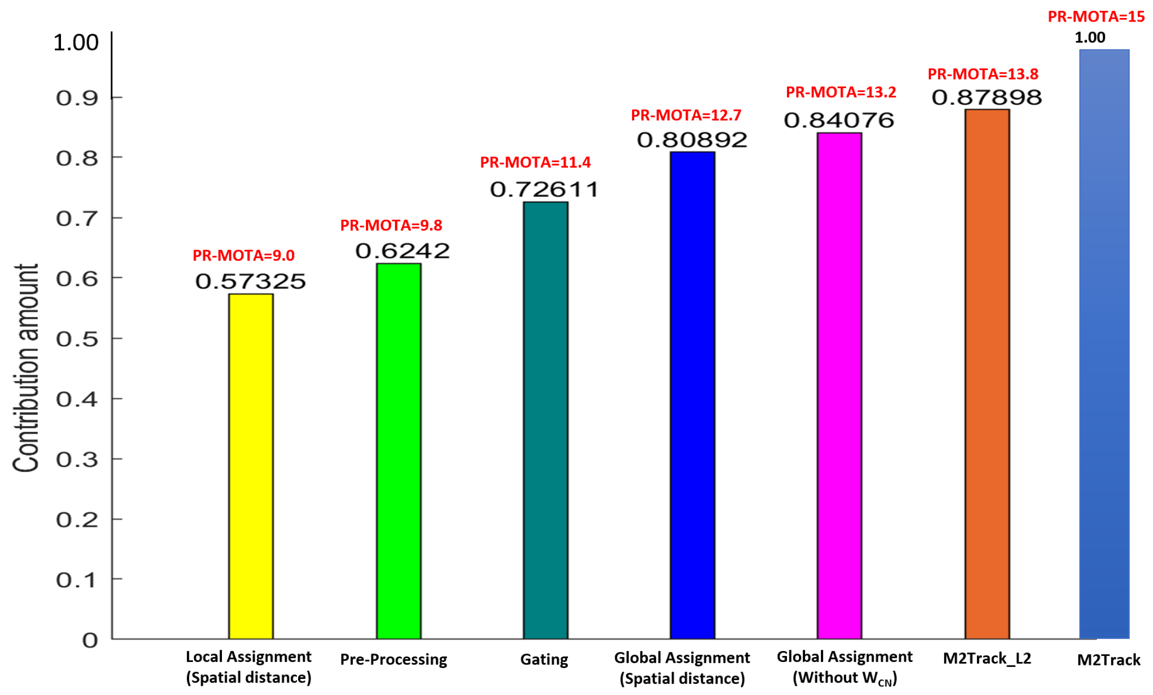


Figure 6.2: Contribution of each tracker component evaluated on UA-DETRAC-train set. PR-MOTA metric for each experiment compared to the Full Mode (M2Track) (higher is better).

was carefully annotated and multiple object classes beside pedestrians bounding boxes were provided (i.e. pedestrian, person on vehicle, car, bicycle, motorbike, non-motorized vehicle, static person, distractor, occluder, reflection). For each bounding box confidence score, class, and visibility ratio have been provided.

6.3.1 MOT16 Benchmark Implementation and Experimental Results

We have tested and evaluated our multi-object tracker on MOT16 benchmark testing set. We used two object detection methods, automatic detection deformable part-based model (DPM) [226] shown in Table 6.4 and YOLOv3 deep learning object detection [147] shown in Table 6.5. We use the metrics that are employed by the MOT evaluation toolkit [227], **MOTA, FP, FN, IDS, MOTP, MT, ML,IDS, and FM**.

Table 6.4: Our tracker performance comparison against MOT16 challenge trackers that are described on [4]. The results were implemented on 7 test video sequences using (DPM) [226] object detection method as an input to the trackers. The arrows indicate whether the metric for the specific column is better when high or low.

Tracker	**Rank↓**	**MOTA↑**	**MOTP↑**	**MT↑**	**ML↓**	**FP↓**	**FN↓**	**IDS↓**	**FM↓**
TBD	4	**33.7**	**76.5**	7.2	54.2	5804	112587	2418	2252
CEM	3	33.2	75.8	**7.8**	54.4	6837	114322	**642**	**731**
DP_NMS	3	32.2	**76.4**	5.4	62.1	**1123**	121579	927	944
SMOT	6	29.7	75.2	4.3	**47.7**	17426	**107552**	3108	4483
JPDA_M	5	26.2	76.3	4.1	67.5	3689	130549	**365**	**638**
M2Track_L2	2	**33.7**	75.0	7.24	49.8	6619	112133	2143	2199
M2Track	1	**36.6**	73.9	**29.0**	**15.8**	**3525**	**76653**	3600	3724

For Table 6.4, our results are compared with MOT16 challenge baseline trackers described on [4] using the same object detection for fare comparison. Our tracker ranks second place and outperforms the other trackers in terms of **FP, FN, IDS, MOTP, and FM**. For Table 6.5 we compare our two trackers, M2Track-L2 and M2Track, to show the importance of our modules. Using the occlusion handling module increases the accuracy of the tracker. **MOTA** increased from 33.7 to 36.6 and **MT** metrics increased from 55 to 220 since the trajectories preserved. **ML** metric decreased from 378 to 120 trajectories that are being lost. Figure 6.3 shows some samples of MOT16 challenge benchmark implemented on our tracker.

Table 6.5: Comparison between M2Track-L2 and M2Track. YOLOv3 object detection is used as an input to the trackers

Tracker	**MOTA↑**	**MOTP↑**	**MT↑**	**ML↓**	**FP↓**	**FN↓**	**IDS↓**	**FM↓**
M2Track-L2	**33.7**	**75**	55	378	6619	12133	**2143**	**2199**
M2Track	**36.6**	73.9	**220**	**120**	35259	76653	3600	3724

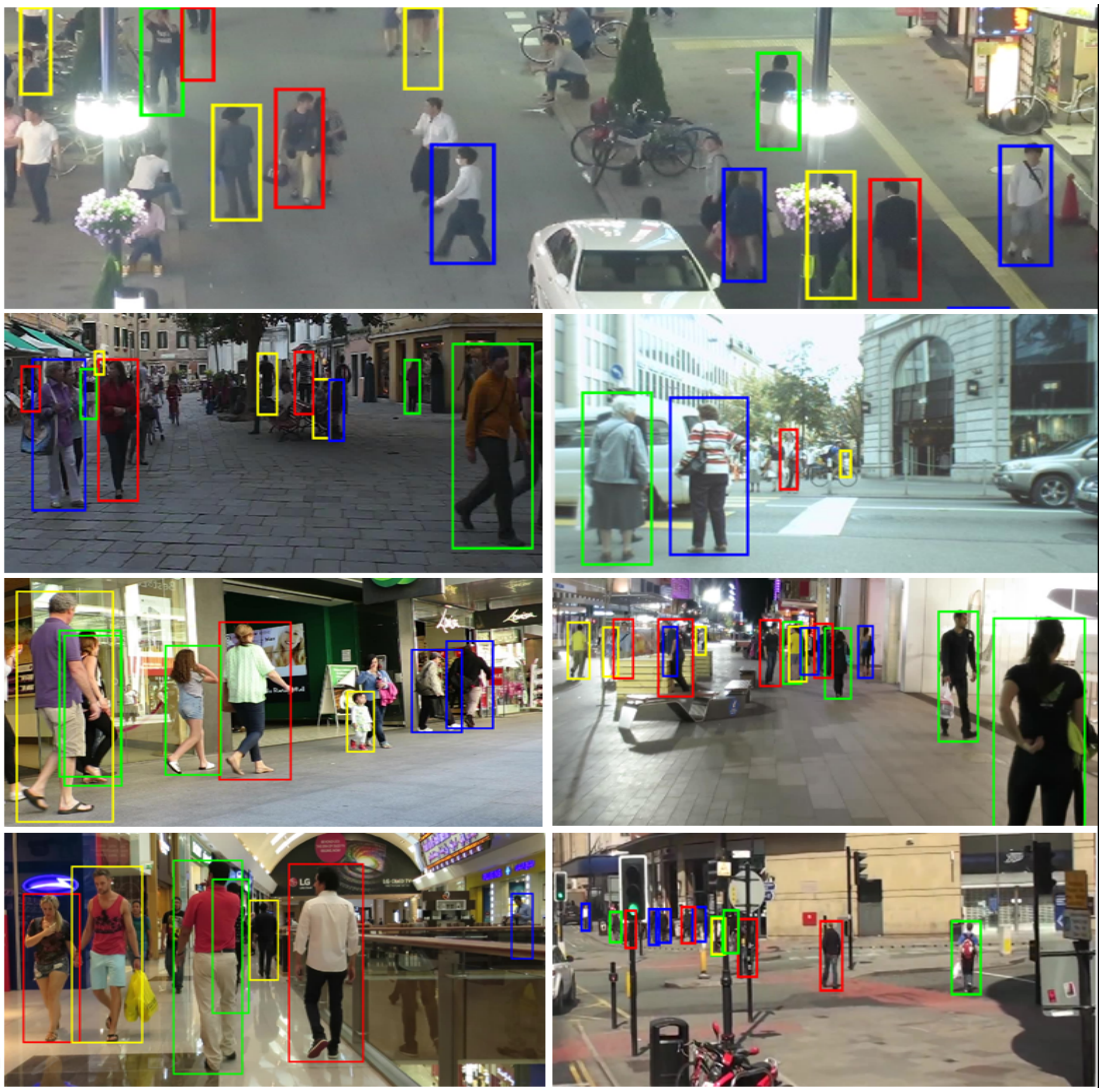


Figure 6.3: M2Track tracking results on the train sequences in the MOT16 benchmark. The color of the boxes represents the identity of the targets. The graph is best shown in color.

Chapter 7

Conclusions and Future Research

7.1 Conclusions

One of the most common and challenging tasks in a computer vision framework is object tracking. Visual object tracking applications range from video surveillance to biomedical image analysis. The objective of our work is to develop robust, accurate, and high-performance moving object tracking to provide comprehensive information about objects' movement for further use with different computer vision problems. Many computer vision problems have been utilized to serve our tracking pipeline to have a comprehensive full scene understanding. Staring from object detection, tracking ends with semantic scene understanding for better analysis. Our contribution to this thesis has two folds. The first is to provide precise trajectory information for an individual object of interest using the single object tracking pipeline, and the second fold is to track all objects in the scene and produce their trajectories using the multi-object tracking pipeline.

For the single object tracking: we presented a recognition-based object tracking framework that extends the Likelihood of Features Tracker (LoFT) with color attributes, scale selection, and kernelized correlation filter module producing CS-LoFT, KC-LoFT. The addition of color attributes improves object and background appearance description. The automated feature scale selection makes the tracker adapts to scale changes caused by object motion, camera motion, or zoom, and ensures scale coherence in the likelihood map and robustness against local outliers. Kernelized correlation filter (KCF) scheme is added to better localize the target in a search window. The online update helps adapt to environmental changes. Experimental results show improved performance, particularly for the sequences that have significant color distractors and rapid scene changes. Also, experimental results on wide-area motion imagery show improved performance in terms of accuracy and robustness compared to LoFT and better or comparable results compared to other state-of-the-art trackers.

For multi-object tracking: our pipeline M2Track tracker, (Multi-cue, Multi-object tracker), is detection-based multi-object tracking with three-step data association to ensure time efficiency, and tracking accuracy. M2Trck is light but efficient modular pipeline for large-scale object density and robust to wide cross-domain benchmarks by having a hybrid cost matrix for frame-to-frame data association; robust and discriminative object appearance model; novel color correlation cost matrix; deep feature model for trajectory matching; and occlusion handling module using (Kalman filter prediction, tracking objects' motion patterns, and considering the environmental constraints). M2Track can handle birth/death, and the appearance/disappearance of objects in the scene. M2Track is an image appearance-based detections and associations pipeline that achieved and fulfilled the requirements of any robust and reliable tracker since it is scalable to thousands of detections, running in near real-time, invariant to object size and density, rigid and deformable objects, and works with cross-domain objects (i.e., aerial, ground-based, biological, real-world, etc.)

M2Track outperforms or has comparable results with the state-of-the-art trackers and baselines in different challenges such as the top three rank on cell segmentation and tracking challenge in 2021 [206], the top one rank on the cell tracking with mitosis detection dataset challenge in 2021 [3], the top two rank in the UA-DETRAC challenge in 2017 and 2018 [101, 93], the top four at the vision meet drones challenge in 2018 [92], and others that are described on the experimental result sections on this dissertation. M2Track was integrated into/with different approaches and applications such as video analysis [84, 85], object detection and tracking in WAMI [85, 95, 73], biomedical cell segmenting and tracking [84], video annotation [87], and video compression [90, 73].

7.2 Future Research

For future work, M2Track pipeline can be improved and upgraded by adding new modules to support the accuracy and robustness to different scenarios. Develop a high-performance multi-object detection and tracking pipeline near real-time for wide aerial motion imagery that works with different environments on different camera acquisitions. Develop framework descriptor for object feature analysis (i.e., appearance-based of deep and non-deep features, motion kinematics, environmental constraints, and depth estimation) by collecting information and cues regarding objects of interest. Develop fusion modules that adaptively select suitable features to be used for trajectory matching. Exploiting the 3D world by tracking 3D moving objects on a reconstructed 3D model. Develop M3Track (Multi-camera, multi-cue, multi-object tracking) by considering object re-identification through multiple cameras, and global coordinates using stitching and mosaicking approaches.

Bibliography

[1] ABQ video. http://www.transparentsky.net.

[2] Todd Rovito, James Patrick, Steve Walls, Daniel Uppenkamp, Olga Mendoza-Schrock, Vince Velten, Chris Curtis, and Kevin Priddy. Columbus Large Image Format (CLIF) 2007 Dataset. https://github.com/AFRL-RY/data-clif-2007.

[3] Cell Tracking with Mitosis Detection Challenge. https://motchallenge.net/data/CTMC-v1/.

[4] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831v2*, 2016.

[5] I. Haritaoglu, D. Harwood, and L.S. Davis. W/sup4/: real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, 2000.

[6] J. Shin, S. Kim, S. Kang, S. Lee, J. Paik, B. Abidi, and M. Abidi. Optical flow-based real-time object tracking using non-prior training active feature model. *Real-Time Imaging*, 11(3):204–218, 2005.

[7] H. Possegger, T. Mauthner, and H. Bischof. In defense of color-based model-free tracking. *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, pages 2113–2120, 2015.

[8] J. Xiao, R. Stolkin, and A. Leonardis. Single target tracking using adaptive clustered decision trees and dynamic multi-level appearance models. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4978–4987, 2015.

[9] A.D. Jepson, D.J. Fleet, and T.F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, 2003.

[10] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2259–2272, 2011.

[11] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. *European Conference on Computer Vision Workshops (ECCV)*, LNCS 8926, 2014.

[12] D. S Bolme, J.R. Beveridge, B. A Draper, and Y.M. Lui. Visual object tracking using adaptive correlation filters. *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, pages 2544–2550, 2010.

[13] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[14] M.D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Online multiperson tracking-by-detection from a single, uncalibrated camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1820–1833, 2011.

[15] M. Ullah, F.A. Cheikh, and A.S. Imran. Hog based real-time multi-target tracking in bayesian framework. *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2016.

[16] Jean-Baptiste Lugagne, Haonan Lin, and Mary J Dunlop. DeLTA: automated cell segmentation, tracking, and lineage reconstruction using deep learning. *PLoS Computational Biology*, 16(4):e1007673, 2020.

[17] Martin Maška, Ondřej Daněk, Saray Garasa, Ana Rouzaut, Arrate Munoz-Barrutia, and Carlos Ortiz-de Solorzano. Segmentation and shape tracking of whole fluorescent cells based on the Chan-Vese model. *IEEE Transactions on Medical Imaging*, 32(6):995–1006, 2013.

[18] Daniel H Rapoport, Tim Becker, Amir Madany Mamlouk, Simone Schicktanz, and Charli Kruse. A novel validation algorithm allows for automated cell tracking and the extraction of biologically meaningful parameters. *PLOS One*, 6(11):e27315, 2011.

[19] Adel Hafiane, Filiz Bunyak, and Kannappan Palaniappan. Level set-based histology image segmentation with region-based comparison. *Proceedings Microscopic Image Analysis with Applications in Biology*, 2008.

[20] Likhitha Kolla, Michael C Kelly, Zoe F Mann, et al. Characterization of the development of the mouse cochlear epithelium at the single cell level. *Nature Communications*, 11(1):1–16, 2020.

[21] Adel Hafiane, Filiz Bunyak, and Kannappan Palaniappan. Evaluation of level set-based histology image segmentation using geometric region criteria. In *IEEE Int. Symp. on Biomedical Imaging (ISBI)*, pages 1–4, 2009.

[22] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. Torr. Staple: complementary learners for real-time tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1401–1409, 2016.

[23] J. Zhang, S. Ma, and S. Sclaroff. MEEM: robust tracking via multiple experts using entropy minimization. In *European Conference on Computer Vision*, pages 188–203, 2014.

[24] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2014.

[25] T. Xu, Z. Feng, X. Wu, and J. Kittler. Learning adaptive discriminative correlation filters via temporal consistency preserving spatial feature selection for robust visual object tracking. *IEEE Transactions on Image Processing*, 28(11):5596–5609, 2019.

[26] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan. Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, pages 6309–6318, 2017.

[27] R. Pelapur, S. Candemir, F. Bunyak, M. Poostchi, G. Seetharaman, and K. Palaniappan. Persistent target tracking using likelihood fusion in wide-area and full motion video sequences. In *IEEE International Conference on Information Fusion*, pages 2420–2427, 2012.

[28] M. Danelljan, A. Robinson, F. Shahbaz Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, pages 472–488, 2016.

[29] Z. Zhang and H. Peng. Deeper and wider siamese networks for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4591–4600, 2019.

[30] J.F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[31] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. *British Machine Vision Conference*, 2014.

[32] R. Girshick. Fast R-CNN. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[34] W. Liu et al. SSD: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, volume LNCS 9905, pages 21–37, 2016.

[35] Rina Bao, Noor M Al-Shakarji, Filiz Bunyak, and Kannappan Palaniappan. DMNet: dual-stream marker guided deep network for dense cell segmentation and lineage tracking. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3361–3370, 2021.

[36] Z. Ren, Z. Yu, X. Yang, M. Liu, Y.J. Lee, A. G. Schwing, and J. Kautz. Instance-aware, context-focused, and memory-efficient weakly supervised object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10598–10607, 2020.

[37] G. Cheng, J. Yang, D. Gao, L. Guo, and J. Han. High-quality proposals for weakly supervised object detection. *IEEE Transactions on Image Processing*, 29:5794–5804, 2020.

[38] J. Wang, J. Yao, Y. Zhang, and R. Zhang. Collaborative learning for weakly supervised object detection. *arXiv preprint arXiv:1802.03531*, 2018.

[39] E. Crawford and J. Pineau. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3412–3420, 2019.

[40] H. Tian, Y. Chen, J. Dai, Z. Zhang, and X. Zhu. Unsupervised object detection with lidar clues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5962–5972, 2021.

[41] E. Xie, J. Ding, W. Wang, X. Zhan, H. Xu, P. Sun, Z. Li, and P. Luo. DetCo: unsupervised contrastive learning for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8392–8401, 2021.

[42] S. Bae and K. Yoon. Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, pages 1218–1225, 2014.

[43] A. Andriyenko, K. Schindler, and S. Roth. Discrete-continuous optimization for multi-target tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2054–2068, 2016.

[44] B. Yang and R. Nevatia. Multi-target tracking by online learning a CRF model of appearance and motion patterns. *International Journal of Computer Vision*, 107(2):203–217, 2014.

[45] J. Xing, H. Ai, and S. Lao. Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1200–1207, 2009.

[46] E. Bochinski, T. Senst, and T. Sikora. Extending IOU based multi-object tracking by visual information. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2018.

[47] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1201–1208, 2011.

[48] Y. Song, K. Yoon, Y. Yoon, K. C. Yow, and M. Jeon. Online multi-object tracking with GMPHD filter and occlusion group management. *IEEE access*, 7:165103–165121, 2019.

[49] A. Milan, S. Roth, and K. Schindler. Continuous energy minimization for multi-target tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):58–72, 2013.

[50] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *IEEE International Conference on Image Pprocessing (ICIP)*, pages 3464–3468, 2016.

[51] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *IEEE International Conference on Image Processing*, pages 3645–3649, 2017.

[52] S. Sun, N. Akhtar, H. Song, A. Mian, and M. Shah. Deep affinity network for multiple object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):104–119, 2019.

[53] B. Shuai, A. Berneshawi, X. Li, D. Modolo, and J. Tighe. Siammot: siamese multi-object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12372–12382, 2021.

[54] Z. Cai, M. Saberian, and N. Vasconcelos. Learning complexity-aware cascades for deep pedestrian detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–16, 2019.

[55] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[56] R. Wang, F. Bunyak, G. Seetharaman, and K. Palaniappan. Static and moving object detection using flux tensor with split gaussian models. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 420–424, 2014.

[57] C.R. del-Blanco, F. Jaureguizar, and N. Garcia. An efficient multiple object detection and tracking framework for automatic counting and video surveillance applications. *IEEE Transactions on Consumer Electronics*, 58(3):857–862, 2012.

[58] K. Yamaguchi, A.C. Berg, L.E. Ortiz, and T.L. Berg. Who are you with and where are you going? *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1345–1352, 2011.

[59] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2137–2155, 2016.

[60] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M. Yang, and S. Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *arXiv:1511.04136*, 2015.

[61] J. Ferryman and A. Shahrokni. PETS2009: Dataset and challenge. *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 1–6, 2009.

[62] P. Zhu, L. Wen, X. Bian, H. Ling, and Q. Hu. Vision meets drones: a challenge. *arXiv:1804.07437v2*, 2018.

[63] S. Oh et al. A large-scale benchmark dataset for event recognition in surveillance video. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3153–3160, 2011.

[64] V. Ulman et al. An objective comparison of cell-tracking algorithms. *Nature methods*, 14:1141—1152, 2017.

[65] R. Pelapur, K. Palaniappan, and G. Seetharaman. Robust orientation and appearance adaptation for wide-area large format video object tracking. *IEEE International Conference on Advanced Video and Signal-Based Surveillance*, pages 337–342, 2012.

[66] K. Palaniappan, F. Bunyak, P. Kumar, I. Ersoy, S. Jaeger, K. Ganguli, A. Haridas, J. Fraser, R. M. Rao, and G. Seetharaman. Efficient feature extraction and likelihood fusion for vehicle tracking in low frame rate airborne video. *IEEE International Conference on Information Fusion*, pages 1–8, 2010.

[67] A. Bhattacharyya. On a measure of divergence between two multinomial populations. *Sankhyā: the Indian Journal of Statistics (1933-1960)*, 7(4):401–406, 1946.

[68] Matej Kristan et al. The visual object tracking VOT2016 challenge results. In *European Conference on Computer Vision (ECCV)*, volume LNCS 9914, 2016.

[69] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[70] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2961–2969, 2017.

[71] F. Bunyak, K. Palaniappan, S.K. Nath, and G. Seetharaman. Flux tensor constrained geodesic active contours with sensor fusion for persistent object tracking. *Journal of Multimedia*, 2(4):20, 2007.

[72] F. Bunyak, K. Palaniappan, S. K. Nath, and G. Seetharaman. Geodesic active contour based fusion of visible and infrared video for persistent object tracking. In *IEEE Workshop on Applications of Computer Vision (WACV)*, pages 35–35, 2007.

[73] N. Al-Shakarji, F. Bunyak, H. AliAkbarpour, G. Seetharaman, and K. Palaniappan. In *Multi-cue vehicle detection for semantic video compression in georegistered aerial videos*, pages 56–65, 2019.

[74] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied mathematics*, 5(1):32–38, 1957.

[75] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[76] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005.

[77] G. Koch, R. Zemel, R. Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2, page 0, 2015.

[78] Y. Rubner, . Tomasi, and L. J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of Computer Vision*, 40(2):99–121, 2000.

[79] Pengfei Zhu, Longyin Wen, Xiao Bian, Ling Haibin, and Qinghua Hu. Vision meets drones: A challenge. *arXiv preprint arXiv:1804.07437*, 2018.

[80] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, and Q. Tian. The unmanned aerial vehicle benchmark: Object detection and tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 370–386, 2018.

[81] Alex L. Chan. A description on the second dataset of the U.S. Army: research laboratory force protection surveillance system. 2009.

[82] S. Anjum and D. Gurari. CTMC: cell tracking with mitosis detection dataset challenge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 982–983, 2020.

[83] E. Asante, D. Hummel, S. Gurung, Y. Kassim, N. Al-Shakarji, K. Palaniappan, V. Sittaramane, and A. Chandrasekhar. Defective neuronal positioning correlates with aberrant motor circuit function in zebrafish. *Frontiers in Neural Circuits*, 15, 2021.

[84] R. Bao, N. Al-Shakarji, F. Bunyak, and K. Palaniappan. Dmnet: Dual-stream marker guided deep network for dense cell segmentation and lineage tracking. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3354–3363, 2021.

[85] N. Al-Shakarji, K. Gao, F. Bunyak, H. Aliakbarpour, E. Blasch, P. Narayaran, G. Seetharaman, and K. Palaniappan. Impact of georegistration accuracy on wide area motion imagery object detection and tracking. In *IEEE International Conference on Information Fusion (FUSION)*, pages 1–8, 2021.

[86] H. Fan, D. Du, L. Wen, P. Zhu, Q. Hu, H. Ling, M. Shah, J. Pan, A. Schumann, B. Dong, D. Stadler, D. Xu, F. Bunyak, G. Seetharaman, G. Liu, V. Haritha, P. S. Hrishikesh, J. Han, K. Palaniappan, K. Zhu, L. W. Sommer, L. Zhang, L. Shine, M. Yao, N. Al-Shakarji, S. Li, T. Sun, W. Sai, W. Yu, X. Wu, X. Hong, X. Wei, X. Zhao, Y. Zhao, Y. Gong, Y. Yao, Y. He, Z. Zhao, Z. Xie, Z. Yang, Z. Xu, Z. Luo, and Z. Duan. Visdrone-mot2020: The vision meets drone multiple object tracking challenge results. In *European Conference on Computer Vision (ECCV)*, pages 713–727. Springer International Publishing, 2020.

[87] N. Al-Shakarji, E. Ufuktepe, F. Bunyak, H. Aliakbarpour, G. Seetharaman, and K. Palaniappan. Semi-automatic system for rapid annotation of moving objects in surveillance videos using deep detection and multi-object tracking

techniques. In *IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–6, 2020.

[88] E. Ufuktepe, V. Ramtekkar, K. Gao, N. Al-Shakarji, J. Fraser, H. AliAkbarpour, G. Seetharaman, and K. Palaniappan. pytag: Python-based interactive training data generation for visual tracking algorithms. volume 11398, 2020.

[89] L. Wen, P. Zhu, D. Du, X. Bian, H. Ling, Q. Hu, J. Zheng, T. Peng, X. Wang, Y. Zhang, L. Bo, H. Shi, R. Zhu, A. Jadhav, B. Dong, B. Lall, C. Liu, C. Zhang, D. Wang, F. Ni, F. Bunyak, G. Wang, G. Liu, G. Seetharaman, G. Li, H. Ardo, H. Zhang, H. Yu, H. Lu, J.-N. Hwang, J. Mu, J. Hu, K. Palaniappan, L. Chen, L. Ding, M. Lauer, M. Nilsson, N. M. Al-Shakarji, P. Mukherjee, Q. Huang, R. Laganiere, S. Chen, S. Pan, V. Kaushik, W. Shi, W. Tian, W. Li, X. Chen, X. Zhang, Y. Zhang, Y. Zhao, Y. Wang, Y. Song, Y. Yao, Z. Chen, Z. Xu, Z. Xiao, and Z. Tong. Visdrone-mot2019: The vision meets drone multiple object tracking challenge results. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 189–198, 2019.

[90] N. Al-Shakarji, F. Bunyak, H. Aliakbarpour, G. Seetharaman, and K. Palaniappan. Performance evaluation of semantic video compression using multi-cue object detection. In *IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–8, 2019.

[91] A. Bouix, N. Al-Shakarji, K. Gao, F. Bunyak, A. Chazot, A. Hafiane, and K. Palaniappan. Robust target tracking using adaptive color feature and likelihood fusion. In *Geospatial Informatics, Motion Imagery, and Network Analytics VIII*, volume 10645, page 106450L, Apr 2018.

[92] P. Zhu, L. Wen, D. Du, X. Bian, H. Ling, Q. Hu, H. Wu, Q. Nie, H. Cheng, C. Liu, X. Liu, W. Ma, L. Wang, A. Schumann, D. Wang, D. Ortego, E. Luna, E. Michail, E. Bochinski, F. Ni, F. Bunyak, G. Zhang, G. Seetharaman, G. Li, H. Yu, I. Kompatsiaris, J. Zhao, J. Gao, J. M. Martinez, J. C. S. Miguel, K. Palaniappan, K. Avgerinakis, L. Sommer, M. Lauer, M. Liu, N. M. Al-Shakarji, O. Acatay, P. Giannakeris, Q. Zhao, Q. Ma, Q. Huang, S. Vrochidis, T. Sikora, T. Senst, W. Song, W. Tian, W. Zhang, Y. Zhao, Y. Bai, Y. Wu, Y. Wang, Y. Li, Z. Pi, and Z. Ma. Visdrone-vdt2018: The vision meets drone video detection and tracking challenge results. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.

[93] S. Lyu, M.-C. Chang, D. Du, W. Li, Y. Wei, M. D. Coco, P. Carcagnì, A. Schumann, B. Munjal, D.-Q. Dang, D.-H. Choi, E. Bochinski, F. Galasso, F. Bunyak, G. Seetharaman, J.-W. Baek, J. T. Lee, K. Palaniappan, K.-T. Lim, K. Moon, K.-J. Kim, L. Sommer, M. Brandlmaier, M.-S. Kang, M. Jeon, N. M. Al-Shakarji, O. Acatay, P.-K. Kim, S. Amin, T. Sikora, T. Dinh, T. Senst, V.-G. Che, Y.-C. Lim, Y. m. Song, and Y.-S. Chung. Ua-detrac 2018: Report of avss2018 amp; iwt4s challenge on advanced traffic monitoring. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2018.

[94] N. Al-Shakarji, F. Bunyak, G. Seetharaman, and K. Palaniappan. Multi-object tracking cascade with multi-step data association and occlusion handling. pages 1–6. IEEE, 2018.

[95] N. Al-Shakarji, F. Bunyak, G. Seetharaman, and K. Palaniappan. Robust multi-

object tracking for wide area motion imagery. pages 1–5, 2018.

[96] N. M. Al-Shakarji, F. Bunyak, G. Seetharaman, and K. Palaniappan. A hybrid local and global multi-object tracking with semantic spatial and appearance modules. volume 10645, page 106450B, 2018.

[97] Y. M. Kassim, N. Al-Shakarji, E. Asante, A. Chandrasekhar, and K. Palaniappan. Dissecting branchiomotor neuron circuits in zebrafish - toward high-throughput automated analysis of jaw movements. pages 943–947, 2018.

[98] N. Al-Shakarji, Y. Kassim, and K. Palaniappan. Unsupervised learning method for plant and leaf segmentation. pages 1–4, Apr 2017.

[99] N. Al-Shakarji, F. Bunyak, G. Seetharaman, and K. Palaniappan. Vehicle tracking in wide area motion imagery using kc-loft multi-feature discriminative modeling. In *IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–6, 2017.

[100] N. Al-Shakarji, F. Bunyak, G. Seetharaman, and K. Palaniappan. Robust multi-object tracking with semantic color correlation. pages 1–7, 2017.

[101] S. Lyui, M. Chang, D. Du, L. Wen, H. Qi, Y. Li, Y. Wei, L. K., T. Hu, M. DelCoco, P. Carcagnì, D. Anisimov, E. Bochinski, F. Galasso, F. Bunyak, G. Han, H. Ye, H. Wang, K. Palaniappan, K. Ozcan, L. W. L. Wang, M. Lauer, N. Watcharapinchai, N. Song, N. Al-Shakarji, S. Wang, S. Amin, S. Rujikietgumjorn, T. Khanova, T. Sikora, T. Kutschbach, V. Eiselein, W. Tian, X. Xue, X. Yu, Y. Lu, Y. Zheng, Y. Huang, and Y. Zhang. Ua-detrac 2017: Report of avss2017 amp; iwt4s challenge on advanced traffic monitoring. pages 1–7, 2017.

[102] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin, T. Vojir, G. Häger, A. Lukezic, G. Fernandez, A. Gupta, A. Petrosino, A. Memarmoghadam, A. Garcia-Martin, A. S. Montero, A. Vedaldi, A. Robinson, A. J. Ma, A. Varfolomieiev, A. Alatan, A. Erdem, B. Ghanem, B. Liu, B. Han, B. Martinez, C.-M. Chang, C. Xu, C. Sun, D. Kim, D. Chen, D. Du, D. Mishra, D.-Y. Yeung, E. Gundogdu, E. Erdem, F. Khan, F. Porikli, F. Zhao, F. Bunyak, F. Battistone, G. Zhu, G. Roffo, G. R. K. S. Subrahmanyam, G. Bastos, G. Seetharaman, H. Medeiros, H. Li, H. Qi, H. Bischof, H. Possegger, H. Lu, H. Lee, H. Nam, H. J. Chang, I. Drummond, J. Valmadre, J. c. Jeong, J. i. Cho, J.-Y. Lee, J. Zhu, J. Feng, J. Gao, J. Y. Choi, J. Xiao, J.-W. Kim, J. Jeong, J. F. Henriques, J. Lang, J. Choi, J. M. Martinez, J. Xing, J. Gao, K. Palaniappan, K. Lebeda, K. Gao, K. Mikolajczyk, L. Qin, L. Wang, L. Wen, L. Bertinetto, M. K. Rapuru, M. Poostchi, M. Maresca, M. Danelljan, M. Mueller, M. Zhang, M. Arens, M. Valstar, M. Tang, M. Baek, M. H. Khan, N. Wang, N. Fan, N. Al-Shakarji, O. Miksik, O. Akin, P. Moallem, P. Senna, P. H. S. Torr, P. C. Yuen, Q. Huang, R. Martin-Nieto, R. Pelapur, R. Bowden, R. Laganière, R. Stolkin, R. Walsh, S. B. Krah, S. Li, S. Zhang, S. Yao, S. Hadfield, S. Melzi, S. Lyu, S. Li, S. Becker, S. Golodetz, S. Kakanuru, S. Choi, T. Hu, T. Mauthner, T. Zhang, T. Pridmore, V. Santopietro, W. Hu, W. Li, W. Hübner, X. Lan, X. Wang, X. Li, Y. Li, Y. Demiris, Y. Wang, Y. Qi, Z. Yuan, Z. Cai, Z. Xu, Z. He, and Z. Chi. The visual object tracking vot2016 challenge results. In *European Conference on Computer Vision Workshop (ECCVW)*, pages 777–823, 2016.

[103] M. Felsberg, M. Kristan, J. Matas, A. Leonardis, R. Pflugfelder, G. Häger, A. Berg, A. Eldesokey, J. Ahlberg, L. Cehovin, T. Vojir, A. Lukezic, G. Fernan-

dez, A. Petrosino, A. Garcia-Martin, A. S. Montero, A. Varfolomieiev, A. Erdem, B. Han, C.-M. Chang, D. Du, E. Erdem, F. S. Khan, F. Porikli, F. Zhao, F. Bunyak, F. Battistone, G. Zhu, G. Seetharaman, H. Li, H. Qi, H. Bischof, H. Possegger, H. Nam, J. Valmadre, J. Zhu, J. Feng, J. Lang, J. M. Martinez, K. Palaniappan, K. Lebeda, K. Gao, K. Mikolajczyk, L. Wen, L. Bertinetto, M. Poostchi, M. Maresca, M. Danelljan, M. Arens, M. Tang, M. Baek, N. Fan, N. Al-Shakarji, O. Miksik, O. Akin, P. H. S. Torr, Q. Huang, R. Martin-Nieto, R. Pelapur, R. Bowden, R. Laganière, S. B. Krah, S. Li, S. Yao, S. Hadfield, S. Lyu, S. Becker, S. Golodetz, T. Hu, T. Mauthner, V. Santopietro, W. Li, W. Hübner, X. Li, Y. Li, Z. Xu, and Z. He. The thermal infrared visual object tracking vot-tir2016 challenge results. In *European Conference on Computer Vision Workshop (ECCVW)*, pages 824–849, 2016.

[104] N. Al-Shakarji, F. Bunyak, and K. Palaniappan. Cs-loft: Color and scale adaptive tracking using bhattacharyya distance and max pooling consistency. 2016.

[105] Y. Wu, J. Lim, and M. Yang. Object tracking benchmark. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015.

[106] A.W.M. Smeulders, D.M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactopm on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.

[107] F. Porikli. Integral histogram: a fast way to extract histograms in cartesian spaces. *IEEE Confrence on Computer Vision and Pattern Recognition*, 1:829–836, 2005.

[108] R.T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *IEEE transaction on Pattern Analysis and Machine Intelligence*, 27(10):1631–1643, 2005.

[109] S. Hare, A. Saffari, P. Torr, et al. Struck: structured output tracking with kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2096–2109, 2016.

[110] K. Zhang, L. Zhang, and M. Yang. Real-time compressive tracking. *European Conference on Computer Vision*, 7574:864–877, 2012.

[111] T. Ba Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1177–1184, 2011.

[112] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *European Conference on Computer Vision*, 2350:661–675, 2002.

[113] K. Nummiaro, E. Koller-Meier, and L. Van Gool. An adaptive color-based particle filter. *Image and Vision Computing*, 21(1):99–110, 2003.

[114] Z. Zivkovic and B. Krose. An EM-like algorithm for color-histogram-based object tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[115] M. Centir, P. Fragneto, D. Denaro, B. Rossi, and C. Marchisio. A combined color-correlation visual model for object tracking using particle filters. *International Symposium on Image and Signal Processing and Analysis*, 2013.

[116] M. Danelljan, Shahbaz K.F., M. Felsberg, and J. Van de Weijer. Adaptive color attributes for real-time visual tracking. *IEEE Confrence on Computer Vision and Pattern Recognition*, 2014.

[117] J.F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. *European conference on Computer Vision*, LNCS 7575:702–715, 2012.

[118] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus. Learning color names for real-world applications. *IEEE Trans. on Image Processing*, 18(7):1512–1523, 2009.

[119] B. Berlin and P. Kay. *Basic color terms: Their universality and evolution.* Univ of California Press, 1991.

[120] A. Bhattacharyya. On a measure of divergence between two multinomial populations. *Sankhyā: The Indian Journal of Statistics (1933-1960)*, 7(4):401–406, 1946.

[121] M. Kristan, J. Matas, et al. The visual object tracking VOT2015 challenge results. *IEEE International Conference on Computer Vision Workshops*, pages 1–23, 2015.

[122] VOT-Challenge Group. http://www.votchallenge.net.

[123] Martin. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4310–4318, 2015.

[124] F. Solera, S. Calderara, and R. Cucchiara. Learning to divide and conquer for online multi-target tracking. In *IEEE International Conference on Computer Vision*, pages 4373–4381, 2015.

[125] B. Wu and R. Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. *International Journal of Computer Vision*, 75(2):247–266, 2007.

[126] H. Kieritz, S. Becker, W. Hübner, and M. Arens. Online multi-person tracking using integral channel features. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 122–130, 2016.

[127] C. Kuo and R. Nevatia. How does person identity recognition help multi-person tracking? In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1217–1224, 2011.

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

[129] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah. Part-based multiple-person tracking with partial occlusion handling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1815–1821, 2012.

[130] J. Prokaj and Gé. Medioni. Using 3D scene structure to improve tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1337–1344, 2011.

[131] V. Reilly, H. Idrees, and M. Shah. Detection and tracking of large number of targets in wide area surveillance. *European Conference on Computer Vision*, LNCS 6313:186–199, 2010.

[132] X. Shi, P. Li, H. Ling, W. Hu, and E. Blasch. Using maximum consistency context for multiple target association in wide area traffic scenes. In *IEEE International Conference on Speech and Signal Processing (ICASSP)*, pages 2188–2192, 2013.

[133] Nguyen.T. L. Anh, F. Bremond, and J. Trojanova. Multi-object tracking of pedestrian driven by context. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2016.

[134] L Marcenaro, M Ferrari, L Marchesotti, and Carlo S Regazzoni. Multiple object tracking under heavy occlusions by using kalman filters based on shape matching. In *Proceedings of the IEEE International Conference on Image Processing*, volume 3, pages III–III, 2002.

[135] Sanjivani Shantaiya, Kesari Verma, and Kamal Mehta. Multiple object tracking using kalman filter and optical flow. *European Journal of Advances in Engineering and Technology*, 2(2):34–39, 2015.

[136] Dan Mikami, Kazuhiro Otsuka, and Junji Yamato. Memory-based particle filter for face pose tracking robust under complex dynamics. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 999–1006, 2009.

[137] M. Naphade et al. The 2018 NVIDIA AI city challenge. In *IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, pages 53–60, 2017.

[138] P. Zhu et al. VisDrone-VDT2018: The vision meets drone video detection and tracking challenge results. In *European Conference on Computer Vision (ECCV)*, volume LNCS 11133, pages 496—-518, 2019.

[139] S. Lyu et al. UA-DETRAC 2017: Report of AVSS2017 & IWT4S challenge on advanced traffic monitoring. In *IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–7, 2017.

[140] D. Lam, R. Kuzma, K. McGee, S. Dooley, M. Laielli, M. Klaric, Y. Bulatov, and B. McCord. xView: Objects in context in overhead imagery. *arXiv:1802.07856*, 2018.

[141] M.E. Farmer, X. Lu, H. Chen, and A.K. Jain. Robust motion-based image segmentation using fusion. *IEEE Int. Conf. on Image Processing*, 5:3375–3378, 2004.

[142] T. Gautama and M.A. Van Hulle. A phase-based approach to the estimation of the optical flow field using spatial filtering. *IEEE Trans. on Neural Networks*, 13(5):1127–1136, 2002.

[143] A. Basharat et al. Real-time multi-target tracking at 210 megapixels/second in wide area motion imagery. *IEEE Workshop on Applications of Computer Vision (WACV)*, pages 839–846, 2014.

[144] R.O. Chavez-Garcia and O. Aycard. Multiple sensor fusion and classification for moving object detection and tracking. *IEEE Trans. on Intelligent Transportation Systems*, 17(2):525–534, 2016.

[145] M. Siam, H. Mahgoub, M. Zahran, S. Yogamani, M. Jagersand, and A. El-Sallab. MODNET: Moving object detection network with motion and appearance for autonomous driving. *Int. Conf. Intelligent Transportation Systems*, 2017.

[146] B. Heo, K. Yun, and J.Y. Choi. Appearance and motion based deep learning architecture for moving object detection in moving camera. In *IEEE Int. Conf. on Image Processing (ICIP)*, pages 1827–1831, 2017.

[147] M.J. Shafiee, B. Chywl, F. Li, and A. Wong. Fast YOLO: A fast you only look once system for real-time embedded object detection in video. *arXiv:1709.05943*, 2017.

[148] H. AliAkbarpour, K. Palaniappan, and G. Seetharaman. Parallax-tolerant aerial image georegistration and efficient camera pose refinement—without piecewise homographies. *IEEE Trans. on Geoscience and Remote Sensing*, 55(8):4618–4637, 2017.

[149] J. Redmon and A. Farhadi. YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[150] K. Palaniappan, I. Ersoy, and S.K. Nath. Moving object segmentation using the flux tensor for biological video microscopy. In *Pacific-Rim Conference on Multimedia (PCM)*, pages 483–493, 2007.

[151] S. Nath and K. Palaniappan. Adaptive robust structure tensors for orientation estimation and image segmentation. In *LNCS-3804: Proc. ISVC'05*, pages 445–453, 2005.

[152] H.H. Nagel and A. Gehrke. Spatiotemporally adaptive estimation and segmentation of OF-Fields. In *European Conference on Computer Vision (ECCV)*, volume LNCS 1407, pages 86–102, 1998.

[153] J. Van De Weijer, T. Gevers, and A.W.M. Smeulders. Robust photometric invariant features from the color tensor. *IEEE Trans. on Image Processing*, 15(1):118–127, 2006.

[154] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 38(1):142–158, 2016.

[155] J. Deng, W. Dong, R. Socher, L. Li, Kai L., and Li F. ImageNet: a large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[156] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[157] H. Possegger, T. Mauthner, P. M Roth, and H. Bischof. Occlusion geodesics for online multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1306–1313, 2014.

[158] W. Rahmaniar, W. Wang, and H. Chen. Real-time detection and recognition of multiple moving objects for aerial surveillance. *Electronics*, 8(12):1373, 2019.

[159] F. Yang, S. Sakti, Y. Wu, and S. Nakamura. A framework for knowing who is doing what in aerial surveillance videos. *IEEE Access*, 7:93315–93325, 2019.

[160] E. Barmpounakis and N. Geroliminis. On the new era of urban traffic monitoring with massive drone data: The pNEUMA large-scale field experiment. *Transportation Research Part C: Emerging Technologies*, 111:50–71, 2020.

[161] L. Wang, F. Chen, and H. Yin. Detecting and tracking vehicles in traffic by unmanned aerial vehicles. *Automation in Construction*, 72:294–308, 2016.

[162] H.L. Yang, D. Lunga, and J. Yuan. Toward country scale building detection with convolutional neural network using aerial images. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 870–873, 2017.

[163] L. Ivanovsky, V. Khryashchev, V. Pavlov, and A. Ostrovskaya. Building detection on aerial images using u-net neural networks. In *IEEE Conference of Open Innovations Association (FRUCT)*, pages 116–122, 2019.

[164] J. V. Stafford. Implementing precision agriculture in the 21st century. *Journal of Agricultural Engineering Research*, 76(3):267–275, 2000.

[165] N. Zhang, M. Wang, and N. Wang. Precision agriculture—a worldwide overview. *Computers and Electronics in Agriculture*, 36(2-3):113–132, 2002.

[166] R. Aktar, D. E. Kharismawati, K. Palaniappan, H. Aliakbarpour, F. Bunyak, A. E. Stapleton, and T. Kazic. Robust mosaicking of maize fields from aerial imagery. *Applications in Plant Sciences*, 8(8):e11387, 2020.

[167] A. Al-Kaff, M.J. Gómez-Silva, F.M. Moreno, A. de la Escalera, and J.M. Armingol. An appearance-based tracking algorithm for aerial search and rescue purposes. *Sensors*, 19(3):652, 2019.

[168] C.D. Rodin, L.N. de Lima, de Alcantara A.F., D. Haddad, T.A. Johansen, and R. Storvold. Object classification in thermal images using convolutional neural networks for search and rescue missions with unmanned aerial systems. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2018.

[169] H.S. Munawar, J. Z., H. Li, D. Mo, and L. Chang. Mining multispectral aerial images for automatic detection of strategic bridge locations for disaster relief missions. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 189–200, 2019.

[170] H. Ling, Y. Wu, E. Blasch, et al. Evaluation of visual tracking in extremely low frame rate wide area motion imagery. In *International Conference on Information Fusion*, 2011.

[171] E. Blasch, G. Seetharaman, S. Suddarth, K. Palaniappan, et al. Summary of methods in wide-area motion imagery (WAMI). In *Proceeding SPIE 9089*, 2014.

[172] K. Palaniappan, R. Rao, and G. Seetharaman. Wide-area persistent airborne video: Architecture and challenges. In B. Banhu et al., editors, *Distributed Video Sensor Networks: Research Challenges and Future Directions*, chapter 24, pages 349–371. Springer, 2011.

[173] C. Leong, T. Rovito, O. Mendoza-Schrock1, C. Menart, J. Bowser, L. Moore1, S. Scarborough, M. Minardi, and D. Hascher. Unified coincident optical and radar for recognition (UNICORN) 2008 Dataset, 2019. https://github.com/AFRL-RY/data-unicorn-2008.

[174] C. Cohenour, F. van Graas, R. Price, and T. Rovito. Camera models for the wright patterson air force base (WPAFB) 2009 wide-area motion imagery (WAMI) data set. *IEEE Aerospace and Electronic Systems Magazine*, 30(6):4–15, 2015.

[175] C. Cohenour, R. Price, T. Rovito, and F. van Graas. Corrected pose data for the wright patterson air force base 2009 wide area motion imagery data set. *J. Applied Remote Sensing*, 9(1):096048, 2015. https://www.sdms.afrl.af.mil/index.php?collection=wpafb2009.

[176] R. Ilin and S. Clouse. Extraction and classification of moving targets in multi-sensory MAMI-1 data collection. In *National Aerospace and Electronics Conference (NAECON)*, pages 387–391, 2015.

[177] https://www.sdms.afrl.af.mil/index.php?collection=mami2013.

[178] K. Palaniappan, M. Poostchi, H. Aliakbarpour, R. Viguier, J. Fraser, F. Bunyak, A. Basharat, S. Suddarth, E. Blasch, R. Rao, and G. Seetharaman. Moving object detection for vehicle tracking in wide area motion imagery using 4D filtering. In *IEEE International Conference. on Pattern Recognition (ICPR)*, pages 2830–2835, 2016.

[179] M. Poostchi, H. Aliakbarpour, R. Viguier, F. Bunyak, K. Palaniappan, and G. Seetharaman. Semantic depth map fusion for moving vehicle detection in aerial video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 32–40, 2016.

[180] S. Razakarivony and F. Jurie. Vehicle detection in aerial imagery: A small target detection benchmark. *Journal of Visual Communication and Image Representation*, 34:187–203, 2016.

[181] L. Wen, D. Du, Z. Cai, et al. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *Computer Vision and Image Understanding*, 193:102907, 2020.

[182] R. Stiefelhagen, K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, and P. Soundararajan. The CLEAR 2006 evaluation. In *International evaluation workshop on classification of events, activities and relationships*, pages 1–44, 2006.

[183] J. Luiten, A. Osep, P. Dendorfer, P. Torr, A. Geiger, L. Leal-Taixé, and B. Leibe. HOTA: A higher order metric for evaluating multi-object tracking. *Int. J. Comp. Vision*, 129(2):548–578, 2021.

[184] MOT tool kit. https://motchallenge.net/devkit/.

[185] H. Aliakbarpour, K. Palaniappan, and G. Seetharaman. Robust camera pose refinement and rapid SfM for multiview aerial imagery—without RANSAC. *IEEE Geoscience and Remote Sensing Letters*, 12(11):2203–2207, 2015.

[186] H. Aliakbarpour, K. Palaniappan, and G. Seetharaman. Stabilization of airborne video using sensor exterior orientation with analytical homography modeling. In *Machine Vision and Navigation*, pages 579–595. Springer, 2020.

[187] Caglayan Dicle, Octavia I Camps, and Mario Sznaier. The way they move: Tracking multiple targets with similar appearance. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2304–2311, 2013.

[188] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[189] W. Tian and M. Lauer. Joint tracking with event grouping and temporal constraints. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–5, 2017.

[190] L. Wen, P. Zhu, et al. VisDrone-MOT2019: the vision meets drone multiple object tracking challenge results. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[191] L. Wen, P. Zhu, D. Du, et al. VisDrone-MOT2019: the vision meets drone multiple object tracking challenge results. In *International Conference on Computer Vision (ICCV)*, 2019.

[192] D. Sun, X. Yang, M. Liu, and J. Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8934–8943, 2018.

[193] R. Ananthakrishnan and A. Ehrlicher. The forces behind cell movement. *Int. J. Biological Sciences*, 3(5):303, 2007.

[194] R. Evans, I. Patzak, L. Svensson, K. De Filippo, K. Jones, A. McDowall, and N. Hogg. Integrins in immunity. *J. Cell Science*, 122(2):215–225, 2009.

[195] D. Montell. Morphogenetic cell movements: Diversity from modular mechanical properties. *Science*, 322:1502–1505, 2008.

[196] F. Bunyak, K. Palaniappan, S. K. Nath, T. Baskin, and G. Dong. Quantitative cell motility for in vitro wound healing using level set-based active contour tracking. In *IEEE Int. Symp. on Biomedical Imaging (ISBI)*, pages 1040–1043, 2006.

[197] J. Condeelis and J. W. Pollard. Macrophages: obligate partners for tumor cell migration, invasion, and metastasis. *Cell*, 124(2):263–266, 2006.

[198] C. Zimmer, B. Zhang, A. Dufour, A. Thébaud, S. Berlemont, V. Meas-Yedid, and J. Marin. On the digital trail of mobile cells. *IEEE Signal Processing Magazine*, 23(3):54–62, 2006.

[199] K. Palaniappan, F. Bunyak, S. Nath, and J. Goffeney. Parallel processing strategies for cell motility and shape analysis. In *High-throughput Image Reconstruction and Analysis*, pages 39–87, 2009.

[200] I. Ersoy, F. Bunyak, J. M. Higgins, and K. Palaniappan. Coupled edge profile active contours for red blood cell flow analysis. In *IEEE Int. Symp. on Biomedical Imaging (ISBI)*, pages 748–751, 2012.

[201] E. Meijering. Cell segmentation: 50 years down the road [life sciences]. *IEEE Signal Processing Magazine*, 29(5):140–145, 2012.

[202] T. Scherr, A. Bartschat, M. Reischl, J. Stegmaier, and R. Mikut. *Best Practices in Deep Learning-based Segmentation of Microscopy Images.* KIT Scientific Publishing, 2018.

[203] F. A G. Pena, P. Fernandez, P. T. Tarr, T. I. Ren, E. M. Meyerowitz, and A. Cunha. J-regularization improves imbalanced multiclass segmentation. In *IEEE Int. Symp. on Biomedical Imaging (ISBI)*, pages 1–5, 2020.

[204] X. Li, Y. Wang, Q. Tang, Z. Fan, and J. Yu. Dual u-net for the segmentation of overlapping glioma nuclei. *IEEE Access*, 7:84040–84052, 2019.

[205] J. Li, Z. Hu, and S. Yang. Accurate nuclear segmentation with center vector encoding. In *International Conference on Information Processing in Medical Imaging*, pages 394–404. Springer, 2019.

[206] Cell Tracking Challenge. http://celltrackingchallenge.net.

[207] K. Parvati, P. Rao, and M. Mariya Das. Image segmentation using gray-scale morphology and marker-controlled watershed transformation. *Discrete Dynamics in Nature and Society*, 2008, 2008.

[208] J. Wang, K. Sun, et al. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[209] P. Matula, M. Maška, D.V. Sorokin, P. Matula, C. Ortiz-de Solórzano, and M. Kozubek. Cell tracking accuracy measurement based on comparison of acyclic oriented graphs. *Public Library of Science (PloS one)*, 10(12):e0144959, 2015.

[210] KIT-Sch-GE. http://celltrackingchallenge.net/participants/KIT-Sch-GE.

[211] PURD-US. http://celltrackingchallenge.net/participants/PURD-US.

[212] CALT-US. http://celltrackingchallenge.net/participants/CALT-US.

[213] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25:1097–1105, 2012.

[214] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[215] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé. MOT20: a benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020.

[216] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the CLEAR MOT metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008.

[217] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision*, volume LNCS9914, pages 17–35, 2016.

[218] Y. Li, C. Huang, and R. Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2953–2960, 2009.

[219] Anton Andriyenko and Konrad Schindler. Multi-target tracking by continuous energy minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1265–1272, 2011.

[220] Longyin Wen, Wenbo Li, Junjie Yan, Zhen Lei, Dong Yi, and Stan Z Li. Multiple target tracking based on undirected hierarchical relation hypergraph. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1282–1289, 2014.

[221] Guang Han, Xiaoyi Yu, and Liu Liu. Robust multi-object tracking based on higher-order graph and min-cost flow network. In *IEEE International Conference on Systems and Informatics*, pages 484–490, 2017.

[222] Andreas Ess, Bastian Leibe, and Luc Van Gool. Depth and appearance for mobile scene analysis. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[223] Tino Kutschbach, Erik Bochinski, Volker Eiselein, and Thomas Sikora. Sequential sensor fusion combining probability hypothesis density and kernelized correlation filters for multi-object tracking in video data. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–5, 2017.

[224] Y.C. Lim and M. Kang. Multi-pedestrian detection and tracking using unified multi-channel features. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–5, 2017.

[225] Y. Song, . Yoon, K. Yoon, and M. Jeon. Online and teal-time tracking with the GM-PHD filter using group management and relative motion analysis. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2018.

[226] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2009.

[227] MOT tool kit. https://motchallenge.net.

VITA

Noor Al-Shakarji was born in Baghdad/ Iraq as the first granddaughter to a big family. She graduated from the Computer Science Department at the University of Technology-Iraq with the top rank of over 200 students in her Bachelor's study in 2005. She attended the graduated school, the Computer Science Department at the University of Technology-Iraq to complete her master's degree after getting the award from The Ministry of Higher Education/Iraq for her top rank in Bachelor's study. She was also in the top rank for her Master's study over 7 graduated students in 2008. For that, she was hired at the Computer Science Department at the University of Technology-Iraq in 2008 as a lecturer.

In 2009, She got married to Ahammd who worked with her at the same University, and she gave birth to two kids Yaqeen and Roya.

In 2013, she got a scholarship to complete her Ph.D. in Computer Science in The United States which was rewarded by the Higher Committee for Education Development in Iraq/ Prime minister's office. She attended the Electrical Engineering and Computer Science Department at The University of Missouri-Columbia to complete her Ph.D. degree. She joins the Computational Imaging and Visualization Analysis Laboratory (CIVA) supervised by Dr. Kannappan Palaniappan in 2015. She works as a research assistant in the CIVA lab on different computer vision problems and she worked on a collaborative project with the Biological Science Department – at the University of Missouri Columbia on a Zebrafish jaw tracking movement project.

She won a number of computer vision challenges in the field of moving object tracking such as the top 3 ranks for the ISBI 2021 Cell Tracking Challenge (CTC-6), The top rank in the graduate student category at the 24th Conference on Neural and Information Processing Systems (NeurIPS2020) SpaceNet 7 challenge, and the top 10 ranks among 300 international participant teams at the same previous conference for the SpaceNet 7 challenge. In 2018, she was awarded the 1907 Women in Engineering Student Award at the University of Missouri-Columbia. She received her Ph.D. degree in Computer Science from the University of Missouri-Columbia in May 2022.