Tracking in Aerial Hyperspectral Videos using Deep Kernelized Correlation Filters

Burak Uzkent*, Aneesh Rangnekar*, and Matthew J. Hoffman**

*Chester F. Carlson Center for Imaging Science, Rochester Institute of Technology
**School of Mathematical Sciences, Rochester Institute of Technology

{bxu2522, apr2635, mjhsma}@rit.edu

Abstract—Hyperspectral imaging holds enormous potential to improve the state-of-the-art in aerial vehicle tracking with low spatial and temporal resolutions. Recently, adaptive multi-modal hyperspectral sensors have attracted growing interest due to their ability to record extended data quickly from aerial platforms. In this study, we apply popular concepts from traditional object tracking, namely (1) Kernelized Correlation Filters (KCF) and (2) Deep Convolutional Neural Network (CNN) features to aerial tracking in hyperspectral domain. We propose the Deep Hyperspectral Kernelized Correlation Filter based tracker (DeepHKCF) to efficiently track aerial vehicles using an adaptive multi-modal hyperspectral sensor. We address low temporal resolution by designing a single KCF-in-multiple Regions-of-Interest (ROIs) approach to cover a reasonably large area. To increase the speed of deep convolutional features extraction from multiple ROIs, we design an effective ROI mapping strategy. The proposed tracker also provides flexibility to couple with the more advanced correlation filter trackers. The DeepHKCF tracker performs exceptionally well with deep features set up in a synthetic hyperspectral video generated by the Digital Imaging and Remote Sensing Image Generation (DIRSIG) software. Additionally, we generate a large, synthetic, single-channel dataset using DIRSIG to perform vehicle classification in the Wide Area Motion Imagery (WAMI) platform. This way, the high-fidelity of the DIRSIG software is proved and a large scale aerial vehicle classification dataset is released to support studies on vehicle detection and tracking in the WAMI platform.

Index Terms—vehicle tracking, hyperspectral sensing, deep features.

I. INTRODUCTION

Aerial object tracking is a popular topic due to its large range of applications in security, traffic surveillance, autonomous driving, and UAV monitoring. Tracking from aerial platforms can be performed with a number of data modalities including, but not limited to, grayscale [1], thermal [2], color [3] and most recently, hyperspectral imagery [4], [5], [6]. Each modality has been exploited for a unique application and comes with its own set of advantages and disadvantages. For the scope of this paper, we will focus on two specific types of sensor modalities: (1) Wide Area Motion Imagery (WAMI), and (2) Adaptive Hyperspectral Imagery.

The WAMI platform can scan up to a 5 km × 5 km area at 2 frames per second (fps), with vehicles occupying roughly 100 - 200 pixels. One can perform persistent tracking utilizing this large field of view image. The low spatial resolution of WAMI has good performance in vehicle tracking under certain circumstances but is not super helpful when it comes to handling background clutter, occlusions, and low contrast objects. To counter this, multiple appearance-based features like textures, color histograms, and histogram of gradients, and motion cues are used in combination, leading to multiple heat maps for the search area. This is not feasible in real-time tracking as the methods can become computationally expensive and hence there exists a need to balance between designing complex models and facilitating real-time implementations. The datasets recorded by WAMI are: (1) WPAFB 2009 [7] and (2) CLIF [8], where each one includes a single video with less than 100 frames. This is detrimental towards training standalone machine learning and deep learning based architectures since the amount of available data is quite low for training purposes. Since data collection from an aerial platform is lengthy and costly, it is possible to utilize deep learning architectures as feature encoders rather than an end-to-end tracking system. However, WAMI is not a practical platform since it provides single-channel imagery and the deep learning architectures are trained on ImageNet [9] which has RGB images.

The unique challenges posed by aerial platforms can be better addressed by smarter multi-modal data acquisition. In this direction, the Rochester Institute of Technology Multi-object Spectrometer (RITMOS) concept is utilized by a number of trackers [10], [11], [5], [6] as an example that can collect a small, targeted amount of hyperspectral data. The RITMOS captures data in two different modalities: (1) a full frame single channel image, and (2) limited hyperspectral data from the desired pixel locations. It can acquire a full-frame single channel image in about 0.1 sec and scan a row of pixels hyperspectrally in 1 ms. Such an adaptive and multi-modal data concept provides more freedom to address aerial tracking challenges. Driven by this freedom and specifications, we design a discriminative tracker to operate on this platform as shown in Fig. 1. We refer the readers to [12], [5] for more information on the workings of RITMOS.

Synthetic Imagery concept

The Digital Imaging and Remote Sensing (DIRSIG) software has been used before to generate spectral scenarios for varied applications that use conventional computer vision techniques and deep learning based models [13], [6], [14], [15]. Since flying spectral sensors on an aerial platform is still an ongoing
area of development due to the high costs involved, we evaluate our tracker on synthetic scenarios generated using DIRSIG by [16], [17], [5]. In particular, we focus on two scenarios: (1) with trees and (2) without trees. We track 43 vehicles in both scenarios as shown in Fig. 2. In addition, we generate a synthetic single-channel aerial dataset for training a CNN and use it to perform vehicle classification on the real WAMI platform, similar to [14] (Sect. [V]).

Motivations
The rich sensory information from hyperspectral imagery has been utilized by generative trackers [5], [6]. The discriminative and deep learning driven trackers on the other hand have recently improved the traditional object tracking dramatically. The main challenges behind the application of discriminative and deep learning trackers in aerial hyperspectral images are:

- Well-established discriminative algorithms such as Efficient Convolutional Operators (ECO) [3], Kernelized Correlation filters (KCF) [18], Struck [19], and Tracking-learning-detection (TLD) [20] are mostly associated with close-angle color image single/multi object tracking at high video frame rates and thus, consider a small region of interest (ROI).

- The cost of collecting data from aerial platforms makes it hard to find/collect large samples of aerial data, leaving the community to split a single video into training and validation sets [21], [22]. This leads to very optimistic
results during off-line tracking due to minimal variance in the training dataset, which results poor performance during online tracking.

II. RELATED WORK

Tracking-by-detection algorithms exploited low-level features such as Histogram of Oriented Gradients and Color-naming features [23, 24, 25] to perform discriminative tracking until the emergence of deep CNN architectures in the computer vision field. The first correlation filter tracker - the Minimum Output Sum of Squared Error (MOSSE) filter [26], used a grayscale channel to learn the classifier vector. Following the MOSSE tracker, a correlation filter accommodating multi-channel features was proposed to boost tracking [27]. Later, the Scale Adaptive with Multiple Features (SAMF) tracker [24] was proposed to concatenate multi-channel HoG and color-naming features. Finally, a kernelized version of correlation filter (KCF) using multi-channel features was proposed to further improve tracking without drastically increasing the computational complexity [29, 18].

The first studies utilizing CNN architectures in object tracking focused on employing the features learned in architectures such as AlexNet [29], VGGNet [30] trained on the ImageNet dataset [9, 31] extracted low-level features from VGGNet to learn a more discriminative correlation filter. Specifically, they encoded objects with the activations of the first several convolutional layers from VGGNet. This setting provided them with a $64 \times 64 \times 96$ dimensional low-level feature set that can be interpreted as a more advanced version of HoG features. They reported slight improvement in the Visual Object Tracking Challenge 2015 (VOT2015) object tracking challenge with deep CNN features over the HoG features. Going deeper is a major key to achieving the state-of-the-art in most computer vision challenges, however, the nature of deep CNN architectures prohibits us from applying high-level features in tracking-by-detection algorithms. This is mainly due to increasing translation invariance in deeper layers resulting from spatial pooling operations.

The object tracking community later migrated to training architectures to perform object tracking in an end-to-end framework [32, 33, 34]. In this direction, Siamese Networks have gained the reputation of the most efficient and effective architecture in tracking. Two branches consisting of the same architecture layers are used in a typical Siamese Network. The bottom branch is provided the ground truth of an object of interest in a ROI, whereas the top branch is assigned the task of estimating the position of the object given the new ROI. Late fusion of the branches is performed and the new position is regressed. The Siamese Networks have surpassed all the other deep tracking-by-detection algorithms in the VOT2015 challenge. Due to the scarcity of annotated datasets for aerial tracking, it is difficult to develop an end-to-end deep learning tracker for aerial platforms.

There is scarcity of annotated datasets for aerial tracking in which deep learning or traditional trackers can be trained and evaluated. UAV123 [35], recently released by Meuller et al., has a ground sampling distance (GSD) that is significantly lower than the high-altitude aerial platforms - thus resulting in objects occupying more than 500 - 1000 pixels. The dataset has sequences at 30 fps, drastically higher than standard WAMI and spectral sequences, which are generally in the 1.42 fps - 2 fps range. Flying RITMOS on an aerial platform is still an ongoing area of development and due to lack of any other real dataset in this area, we use synthetically generated Rochester Institute of Technology Multi-object Spectrometer (RITMOS)-like data to evaluate the performance of our proposed tracker. This way, we prevent probable overfitting that would have been caused due to training and testing on the same dataset by using deep learning models as feature encoders in our tracker.

Contributions

This study addresses the unique challenges posed by the application of discriminative trackers to aerial platforms. A novel method that employs a discriminative tracker is proposed to tackle low temporal (around 1.42 fps) and spatial resolutions (0.3 m). Primarily, we design a method to enlarge the area considered by the tracker to handle the low temporal resolution. Given the rich hyperspectral imagery, we utilize pre-trained deep convolutional networks as feature encoders to boost tracking performance. To accommodate deep features in a near real-time tracking system, we design a region-of-interest (ROI) mapping strategy that only forward passes the large ROI and projects the individual ROIs to the large ROI feature maps (Fig. 1). Finally, the proposed tracker is evaluated on a synthetic hyperspectral video generated by the Digital Imaging and Remote Sensing (DIRSIG) software [36]. To prove the high-fidelity of this video, a large single-channel aerial dataset is synthesized using DIRSIG and a deep learning framework is trained on it to classify images from the real dataset (WAMI). We refer the readers to following link to access our synthetic vehicle classification dataset (https://buzkent86.github.io/datasets/).

To the best of our knowledge, this is the first time an adaptive hyperspectral sensor-inspired discriminative tracker (DeepHKCF) has been proposed to perform robust single target tracking in spectral aerial imagery that can be generalized to the WAMI platform.

III. PROPOSED TRACKER

As discussed in the previous section, the tracking platform has low frame rate making the global camera motion removal step necessary to perform consistent tracking. In this direction, we register the input frame to the canonical frame where the tracking is initialized using standard computer vision techniques. First, keypoints in the images are extracted using the Scale Invariant Feature Transform (SIFT) [37] and described with gradient orientation histograms. In the next step, the homography matrix between two images is estimated with the Random Sample Consensus (RANSAC) [38] algorithm. Finally, the input image is warped to the canonical image using the accumulated homography matrix over time.

The core of the proposed tracker is built upon the work of Henriques et al. [28, 18] with Kernelized Correlation Filters (KCF). The KCF has emerged as a high accuracy tracker
that can operate at hundreds of frame rate under specific conditions. Its computational efficiency is derived from the correlation filter framework that represents training examples using a circulant matrix. The fact that a circulant matrix can be diagonalized by Discrete Fourier Transform (DFT) is the key to reducing the complexity of any tracking method based on the correlation filter. The off-diagonal elements become zero whereas the diagonal elements represent the eigenvalues of the circulant matrix. The KCF applies a kernel to transform the feature channels to a more discriminative domain.

Essentially, the KCF solves the problem in the form of ridge regression:

\[
E(w) = \frac{1}{2}||y - \sum_{c=1}^{C}(w_c * x_c)||^2 + \frac{\lambda}{2} \sum_{c=1}^{C}||w_c||^2
\]  

where \(y\) represents the desired continuous response, \(w\) represents the correlation filter and \(x_c\) represents template for the given channel. The parameter \(C\) enables one to integrate features in multiple channel space: an earlier version based on this formulation employed grayscale feature \((C = 1)\) to learn the solution vector \(w\). Later, multi-channel features such as Color, HoG and a concatenation of them showed improved accuracy [18], [27], [39], [40], [41]. To reduce the complexity of the closed-form solution for Eqn. [1] an elementwise multiplication in the frequency domain was proposed for \(\hat{w}\) [26]:

\[
\hat{w} = \frac{\hat{x}^* \odot \hat{y}}{\hat{x}^* \odot \hat{x} + \lambda},
\]

where \(\odot\) and \(*\) denote the parameter in Fourier domain and conjugate of a complex number whereas \(\odot\) and \(\lambda\) are the element-wise multiplication, and a regularization term to prevent divisions by zero.

The solution to the kernelized version of ridge regression is given by [42] as follows:

\[
\alpha = (K + \lambda I)^{-1} y
\]

where \(K\) is the kernel matrix and \(\alpha\) is the vector of coefficients \(\alpha_i\), that represent the solution in the kernel-transformed dual space. The diagonalized Fourier domain dual form solution (non-linear version) is then expressed as

\[
\hat{\alpha} = \hat{y}(\hat{k}^{xx} + \lambda)^{-1},
\]

where \(k^{xx}\) is the first row of the kernel matrix \(K\) and is the kernel’s autocorrelation.

For multiple channel cases, we obtain \(\hat{k}^{xx}\), which represents the first row of the kernel matrix \(K\) in the frequency domain, also known as gram matrix. It can be formulated as:

\[
k^{xx} = \exp(-\frac{1}{\alpha^2}(||x||^2 + ||x'||^2 - 2\sum_{c=1}^{C}(\hat{x}_c^* \odot \hat{x}_c'))).
\]

where \(x\) concatenates the individual vectors for \(C\) channels: \(x = [x_1, \ldots, x_C]\). In training step, the arbitrary vector \(x'\) is replaced by \(x\), and in test step, it is replaced by \(z\).

To detect the object of interest, we typically wish to evaluate the regression function \(f(z)\) on several locations in the image, i.e. several candidate patches, which can be modeled by cyclic shifts.

\[
f(z) = w^T z = \sum_{i=1}^{n} \alpha_i \kappa(z, x_i)
\]

where \(\kappa\) is the kernel correlation of \(x\) and \(z\).

Eqn. [7] then translates into the following equation in time domain:

\[
r(z) = F^{-1}(\hat{k}^{xz} \odot \hat{\alpha})
\]

where \(r\) denotes the correlation response at all cyclic shifts of the first row of the kernel matrix.

The temporal information can be further integrated into the tracker by updating the filter and target template at every frame as follows:

\[
\hat{\alpha}_t = (1 - \beta)\hat{\alpha}_{t-1} + \beta \hat{x}_t, \\
\hat{x}_t = (1 - \beta)\hat{x}_{t-1} + \beta x_t
\]

where \(\beta\) is the learning rate. This correlation filter framework only estimates the translation of the object whereas the scale of the object can be updated by running a correlation filter on different size ROIs with same centroids [24]. By correlating a filter with different ROIs, we can get multiple response maps and choose the one with highest confidence to estimate the new scale of the target. In this study, we do not estimate the scale of the target as the scenarios are captured from a fixed altitude platform.

### A. Single KCF-Multiple ROIs Approach

Discriminative trackers like KCF learn to function in an online manner by collecting positive and negative samples and then detecting the target of interest in a ROI to update the classifier. The standard form KCF requires small ROIs as the appearance-based features deteriorate with larger background context. Unfortunately, these features are hard to collect from aerial imaging platforms due to their low spatial resolution. Moreover, there are two other limitations: (1) Increasing the context size leads to background dominated features resulting in confusion between different objects and (2) The platform we consider has lower temporal resolution (1.4 fps) leading to large displacement of objects in successive frames. Adding the platform motion into this picture makes the application of vanilla-form KCF in aerial platforms extremely difficult. To handle these challenges, we propose a single KCF-in-multiple ROIs approach (Fig. [3]). Our approach applies the same KCF to different ROIs overlapping each other to minimize the likelihood of target loss. It is essential to have reasonable overlap between the ROIs (Sect. [V-G]) as we filter each ROI with a Hanning window to avoid distortion at boundaries due to FFT operation. This approach can be formulated by modifying Eqn. [8] as:

\[
r(z_{ij}) = F^{-1}(\hat{k}^{xz_{ij}} \odot \hat{\alpha})
\]
where \( i \) and \( j \) represent the indexes for different ROIs. A simple way to estimate the new position of the target in this framework would be using the peak-to-side-lobe ratio (PSR) values in ROIs and finding the position of the pixel with maximum confidence in all ROIs as:

\[
r(\mathbf{z}_{\text{final}}) = \arg\max_{i,j \in \sqrt{m}} (\text{PSR}(\mathbf{z}_{ij}))
\]

where \( m \) represents the number of ROIs in full ROI. The PSR, on the other hand, denotes the margin between the peak value in the response map and the mean of the sidelobe corresponding to the area excluding the 11x11 pixels around the peak. The result is normalized by the standard deviation of the sidelobe as follows.

\[
\text{PSR}(\mathbf{z}_{ij}) = \frac{\max(\text{PSR}(\mathbf{z}_{ij})) - \mu_{\text{sidelobe}}}{\sigma_{\text{sidelobe}}}
\]

This position estimation approach can be softened by considering all the ROIs with PSR values larger than a predetermined threshold, \( T \). In this case, the Eqn. 12 can be reorganized as follows.

\[
r(\mathbf{z}_{\text{final}}) = \sum_{i} \sum_{j} \beta_{ij} \ast r(\mathbf{z}_{ij}),
\]

\[
\beta_{ij} = \begin{cases} 
0, & \text{if } \text{PSR}(\mathbf{z}_{ij}) > T \\
\text{PSR}(\mathbf{z}_{ij}), & \text{otherwise}.
\end{cases}
\]

By softening our decision, we perform low-pass filtering and avoid jumps to other objects that has a high PSR value in only one ROI.

As mentioned earlier, the single KCF-multiple ROIs approach can better handle the low temporal resolution than the traditional KCF. On the other hand, it increases the complexity linearly from \( O(n \log(n)) \) to \( O(m \times n \log(n)) \), where \( m \) represents the number of ROIs in the full ROI. The low temporal frame rate of the scenario helps us accommodate this approach in the DeepHKCF tracker. It is possible to further increase the frame rate by running the KCF on the multiple ROIs in parallel as the ROI operations are independent.

B. Traditional Low-level Features

In this study, we follow the KCF tracker and concatenate multiple features as in the SAMF \cite{24} tracker. More specifically, we concatenate the Felzenszwalb’s HoG (fHoG) \cite{25} feature channels and pure hyperspectral channels and apply the Gaussian kernel operation to learn a more discriminative model as follows:

\[
k_{xz} = \exp(-\frac{1}{\alpha^2}(||x||^2 + ||z||^2 - \beta_{ij}^2))
\]

\[
2F^{-1}(\sum_{c_{fHoG}=1} \hat{z}_{c_{fHoG}} \odot \xi_{c_{fHoG}} + \sum_{c_{HSI}=1} \hat{z}_{c_{HSI}} \odot \xi_{c_{HSI}})
\]

where \( c_{fHoG} \) and \( c_{HSI} \) represent the hyperspectral and fHoG feature channels. The number of hyperspectral and fHoG feature channels in this study are 61 and 31 respectively. Additionally, in the results section (Sect. \text{V}), we experiment with fHoG features and hyperspectral features alone to observe how well they perform individually.

C. Deep Convolutional Features

In this study, we follow an approach similar to \cite{31} to learn a discriminative model for the KCF. Low spatial resolution scenario enables us to pursue a slightly higher level of abstraction of objects. In particular, we apply the activations of the fifth convolutional layer learned in VGGNet \cite{80} trained over ImageNet \cite{9}. Additionally, we experiment with different levels of object abstractions in the experiments section (Sect. \text{V}).

DIRSIG imagery provides us with a full-frame grayscale image as well as a narrow field of view hyperspectral image at 1.42 fps. Unlike other aerial platforms, it provides hyperspectral data in the visible wavelength range, enabling the use of deep CNN architectures trained on ImageNet consisting of RGB images. One can pick the central red, green and blue channels and forward-pass them through the layers of interest. Another approach could be computing the average of red, green and blue channels in their respective range to come up with the representative red, green and blue channel images to feed the CNN. Our experiments favor the first approach as the latter approach introduces undesired noise due to the averaging operation.

Fast Convolutional Features with ROI Mapping

The single KCF-multiple ROIs approach treats each ROI independently to compute the filter response. This requires forward-passing individual ROIs through the CNN architecture. Such an inefficient approach leads to a slower tracker. To increase the run-time performance and perform near real-time tracking at the platform frame-rate, we use the ROI mapping strategy commonly used in convolutional object detectors such as Fast R-CNN \cite{43}, Faster R-CNN \cite{44}, and R-FCN \cite{45}. This way, we only forward-pass the full ROI and project the individual ROIs to the feature maps extracted from the full ROI as shown in Fig. \text{4}.
IV. Vehicle Classification in WAMI Platform by Using a Synthetic Dataset

As discussed in Section II detecting cars with high accuracy is a major problem of tracking algorithms utilizing the WAMI platform. This is due to two major reasons: (1) the lack of a large dataset captured from the WAMI platform and (2) the lack of color channels prevents smooth transfer learning from the networks trained on the ImageNet.

In this study, we build a synthetic single-channel vehicle classification dataset using DIRSIG and fine-tune a CNN to perform vehicle classification on the real platform (WAMI). To build this dataset, we generate full-frame hyperspectral images captured from the Mega-Scene I scene available in DIRSIG with different settings. The simulation setting is designed as a function of time, and hence the brightness in the scene varies as a function of sunlight which can then lead to a more general dataset. In particular, nine simulations from different months in a year are generated to find representative samples of varying conditions. We keep the other parameters similar to the simulation used to generate the RITMOS-like scenario. Overall, nine different simulations are generated from the same scene with the same vehicular traffic and platform motion to the tracking video.

A. Temporal Data Augmentation

It is possible to generate just one frame per simulation. However, to increase the number of simulations, we add more temporal-variance and change the initial platform location. Changing the platform location in a large number of simulations can be a tedious task, and to avoid that, we perform temporal data augmentation by generating low frame rate videos on a moving platform. More specifically, the frame rate for each simulation is set to 0.2 fps, resulting in 20 images per simulation. This way, we can capture cars from different angles with different backgrounds.

B. Hyperspectral Data Augmentation

The data augmentation is highly important in our case as we mimic the WAMI platform in a dataset consisting of fully synthetic images. In particular, it is difficult to approximate the spectral sensitivity curve of a real platform synthetically. The same car samples from different wavelengths are augmented to better approximate the WAMI platform internal mechanics. We stick with 61 channels in the visible (400 nm) to near infrared (1000 nm) wavelength range. In a single-band image setting with 0.2 fps, we produce about 180 images leading to small spectral variance in the dataset. By using all 61 channels, we generate over 1000 images over the 9 simulations, considering time and spectral depth. This approach has the potential downside of generating to a dataset dominated by highly similar images. To address this, we sample 6 bands from 6 uniform distributions covering the 61 channels as shown in Fig. 5. This increases the spectral variance while ensuring a reasonably large gap between the augmented images at different wavelengths.
C. Positive and Negative Samples Collection

The procedure described above produces 27613 vehicle chips (64 × 64 px) and the vehicles are located in the central position of the positive chips. Similar to the WAMI platform, a vehicle is represented by 20 × 10 pixels on average in the generated scenarios. Adding context in positive samples seems to improve the learned weights in a CNN [22]. To collect negative samples, we perform hard-negative mining by considering areas surrounding the positive samples. A negative sample is randomly captured from an area whose center is T = 10 - 30 pixels away from the center of the positive sample.

Our final dataset consists of 55226 chips captured from different positions of Mega-Scene I at different times. To validate the performance on the WAMI platform, we annotate 600 positive and negative chips from the CLIFF06 and CLIFF07 videos captured from the WAMI platform. Some of the positive samples from the training and validation dataset can be visualized in Fig. 6. Finally, we train a well-known CNN architecture to perform vehicle classification on the WAMI platform.

D. Training the models

The architecture used in this study is the ZFNet, an optimized version of AlexNet. We adopt two different training strategies: (1) training from scratch and (2) fine-tuning the weights learned on the ImageNet using the synthetic aerial vehicle detection dataset. In the latter approach, the learning rate is set to 0.0001 other than the classification layer. The classification layer is assigned a learning rate of 0.0005. On the other hand, in the former method, we tune the learning rate to 0.1. The ZFNet from scratch is trained for 4200 iterations with the batch size of 64 whereas the one pre-trained on ImageNet is trained for 400 iterations with the same batch size. In this two experiments, the networks are validated on the 600 samples from WAMI (CLIFF06 and CLIFF07). To integrate further context information into the learned weights, we introduce dilated convolutions with hole size 2 and 1 in the 1st and 2nd convolutional layers [46]. Finally, we follow a two-stage training strategy that uses the 200 WAMI samples to further update the weights from the fine-tuned ZFNet. The model is validated on the remaining 400 WAMI samples. This, as expected, boosts the classification accuracy on the WAMI platform. Training is performed on the NVIDIA Tesla K80 GPU in Caffe framework [47].

As seen in Table I, over 93% accuracy is achieved by only using our synthetic dataset to train the ZFNet. This

### TABLE I: Performance of the trained neural networks on the WAMI vehicle classification task. In the “*” case, 200 WAMI samples are used to further fine-tune and validate the network on the 400 WAMI samples. [22] splits the WPAFB2009 video (WAMI) into training and validation.

| Method          | ZFNet (from Scratch) | ZFNet (ImageNet) | ZFNet (*) | AlexNet (WAMI [22]) |
|-----------------|----------------------|------------------|-----------|---------------------|
| Accuracy (%)    | 93.20                | 92.230           | 97.0      | 97.1                |

- Method: ZFNet (from Scratch), ZFNet (ImageNet), ZFNet (*), AlexNet (WAMI [22])
- Accuracy (%): 93.20, 92.230, 97.0, 97.1
proves the high fidelity of the hyperspectral scenario used to evaluate the deep hyperspectral kernelized correlation filter tracker. To further improve the accuracy, a small amount of WAMI samples are used to train a more WAMI domain-specific model. This improves the accuracy up to 97% reaching the state-of-the-art in vehicle classification in WAMI platform. Finally, with the availability of this dataset, the need to collect a large amount of training samples from the WAMI platform is removed. To support the aerial vehicle detection related studies, we plan on releasing the full images with ground truth locations of the vehicles. This will give more freedom to train detection-domain architectures such as Faster-RCNN [44], R-FCNN [45], YOLO9000 [48], and SSD [49].

V. Tracking Experiments

DIRSIG is a very useful program for generating remote sensing images with high fidelity. This is proved in the previous section where we generated a large synthetic dataset representative of the WAMI platform and trained a convolutional network to classify real images from the WAMI platform. A hyperspectral tracking video representative of the RITMOS sensor was generated in previous studies [4], [6], [13], [10]. The hyperspectral tracking scenario has two different videos: (1) without trees and (2) with dense trees (25% full occlusion by trees). Both videos have 1.42 fps and 157 frames with same vehicular traffic, and platform position. For our study, we used both videos to evaluate the performance of the proposed DeepHKCF tracker, its variants, and other hyperspectral state-of-the-art trackers.

A. Hyperparameter Tuning

This section discusses the hyperparameters that need to be tuned to perform optimal tracking considering both runtime performance and accuracy. The KCF has a number of hyperparameters including padding size, desired Gaussian response width, learning rate and Gaussian kernel bandwidth. Our approach has three main hyperparameters: (1) full ROI size, (2) overlap between ROIs and (3) PSR threshold to remove the contribution of noisy response maps. We set the size of a single ROI to $(48 \times 48)$ pixels since each vehicle occupies about $20 \times 10$ pixels and hence reasonable content is captured. This removes the need to have a padding size hyperparameter in the DeepHKCF. The other KCF hyperparameters are tuned to similar values as the original KCF paper. The overlap between ROIs is set to 64% in each dimension (Sect. V-C) whereas the full ROI size and PSR threshold are set to 96 × 96 pixels (Sect. V-H) and 7 respectively.

B. Tracking Performance Metrics

For analyzing the performance of our tracker and its variants, we use two metrics: (1) Central Location Error and (2) Precision, which are defined as follows:

**Central Location Error**

The central location error (CLE) for a dataset can be calculated in three effective steps: (1) The central location error is defined as the average Euclidean distance between the predicted center location of the target and the ground truth of a frame. (2) The average center location error over all the frames of one sequence is used to then summarize the overall performance value for that sequence. (3) Lastly, the average central location error of a dataset is calculated by averaging the central location error across all the sequences in the dataset. Ideally, it is preferred to have a low central location error.

**Precision**

Precision can be defined as thresholding the Euclidean distance between the prediction and ground truth centroid. In the paper, the final Precision scores are obtained by: (1) dividing the number of successful frames to the total number of frames in a sequence to get the Precision score at the respective threshold. (2) Performing the same operation on all the sequences and averaging to compute the final Precision score on a dataset. Pr 20 px and Pr 50 px represent the precision at 20 and 50 pixels Euclidean distance thresholds. The threshold is slid between between 0 to 50 pixels by 1 pixel interval to draw the Precision figures. Ideally, it is preferred to have a high Precision value.

C. Results on the No-trees Scenario

After tuning the hyperparameters of DeepHKCF, we test it on the 43 vehicles in the no-trees scenario. We compare the DeepHKCF to a number of variants of the proposed single KCF multiple ROIs approach as seen in Table I. Furthermore, we perform experiments on the original KCF algorithm (single KCF-single ROI approach) with the same ROI size $(48 \times 48)$ px. Additionally, we compare the proposed tracker to Efficient Convolutional Operator tracker (ECO) [1] that is ranked first in the VOT16 tracking benchmark. For fair comparison, we increase the learning rate of the ECO tracker to match our scenarios and use the same features. Similar to DeepHKCF and HKCF, we use the activations of the 5th convolutional layer of the VGGNet and fHoG features. To determine the ROI area, ECO considers the padding size of 3.5, larger than the optimal padding size (2.0) of the KCF. We keep this hyper-parameter same as including further background deteriorates the features. For $20 \times 10$ pixel vehicle, the ECO and vanilla-form KCF trackers have a search area of $80 \times 45$, and $60 \times 30$ pixels whereas it is $96 \times 96$ for the proposed DeepHKCF. Fig. 15 and Table I show the performances of the proposed DeepHKCF, its variants and the baseline KCF and ECO trackers.

The DeepHKCF performs exceptionally well in the no-trees scenario, achieving 70% precision at the 20 px threshold and outperforming all the baseline methods by a large margin. Meanwhile, the proposed HKCF with fHoG features performs substantially worse than the one with deep features. However, it outruns the original KCF with fHoG features by a large margin at 50 px precision as shown in Table I proving the contribution of the proposed single KCF multiple ROIs approach in low frame rate tracking. Concatenating hyperspectral features with fHoG slightly degrades the precision whereas hyperspectral feature channels alone performs worse.
than the former methods. This indicates that the NIR channels do not contribute to tracking in the KCF framework. The ECO tracker, on the other hand, delivers 10 - 15% lower accuracy than the DeepHKCF trackers at 20 px precision and about 20% worse in terms of the precision at 50 px and central location error. All in all, the DeepHKCF tracker with ROI mapping (FastDeepHKCF) achieves optimal results considering its reasonably high tracking accuracy and highest operation rate among the DeepHKCF trackers.

### D. Results on the Dense Trees Scenario

In addition to conducting experiments on the no-trees scenario, we run the DeepHKCF tracker and its variants on the same scenario with dense trees. This is an extremely challenging scene dominated by large trees and their shadows as shown in Fig. 2. On average, a vehicle is fully occluded in 1 out of 4 frames. Severe occlusions combined with low frame rate make this a more challenging scene. The DeepHKCF trackers outperform the other baseline methods other than ECO tracker by a large margin as in the no-trees scenario (Fig. 7D). At 20 px precision, the DeepHKCF tracker achieves about 30% accuracy whereas others perform 10-20% worse. On the other hand, among the DeepHKCF trackers, the FastDeepHKCF, delivers similar precision at 50 px and higher frame rate. Similar to the no trees scenario, the combination of hyperspectral feature channels with fHoG degrades the performance with respect to the fHoG-only features. We believe that this could be due to more frequent switching to non-vehicle objects with similar hyperspectral features to the target of interest through occlusions. By using fHoG-only features, it is less likely to switch to an object that does not appear like a vehicle. The dramatic drop in precision rates between the no-trees and dense trees scenarios is easily seen in Fig. 2. This is likely due to three major reasons: (1) high frequency of severe occlusions, (2) low video frame rate, and (3) relatively smaller search area considered by our single KCF-multiple ROIs approach. The combination of the first two reasons leads to dramatically large displacement of objects in between the frames where they are visible. This results in the targets being located out of the search area of the DeepHKCF tracker. There are two solutions to address the challenge of tracking through severe occlusions. The first and less practical solution is increasing the full ROI size in each dimension. This way, we increase the likelihood of keeping the target in our search area traveling through severe occlusions. However, this will also reduce the run-time performance. A more practical solution could be delivered by leveraging a Bayes Filter. For instance, [6], [9] uses a Bayes Filter in a Multi-dimensional Assignment algorithm to update the measurements in light of the later measurements in the same scenario. This way, we can low-pass the unlikely jumps that occurs during severe occlusions. On the other hand, we believe that, increasing the search area in a practical manner might be the key to achieving state-of-the-art performance in scenarios dominated by trees (see Sect. V-H).

### E. Experiments on Temporally Down-sampled Video

In the previous sections, we evaluated the proposed trackers and the baseline methods on the 1.42 fps videos with dense trees and without trees. The ECO tracker performs only
We believe that this might be due to slowly moving or stopped vehicles. To further test their performance with respect to overlap ratio, we down-sample the vehicles. The ECO tracker misses more targets due to the overlap ratio in terms of precision and central location error showing margin in terms of precision and central location error showing robustness to extreme target displacement in successive frames. Finally, the FastDeepHKCF delivers the optimal results considering the trade-off between tracking accuracy and run-time performance.

G. Effect of Overlap Ratio

We experiment on the DeepHKCF tracker with ROI mapping (FastDeepHKCF) as a function of the overlap ratio between the adjacent ROIs. As mentioned before, it is necessary to have overlap between the adjacent ROIs as the hanning window is applied to the features before the FFT operation. The hanning window filters the noise at the boundaries resulting from FFT operation. Increasing overlap ratio, at the same time, leads to increased complexity ($O(m * n\log(n))$) due to a larger number of ROIs ($m$) in the full ROI (96 × 96 px). Fig. 8(a) shows the precision rates of FastDeepHKCF tracker with different overlap ratios between the adjacent ROIs.

The 64% and 75% overlap ratios lead to drastically better results than the lower ones (Fig. 8a). Considering the accuracy/speed trade-off, 64% overlap ($m = 16$) is used as the optimal setting.

TABLE V: Comparison of the DeepHKCF tracker to HLT. The 5D and 2D refer to the number of past frames considered in MDA [50]. The experiments are carried out on a CPU with 8GB RAM and 2.9GHz i5 processor. The best result in each category has been highlighted for better understanding.

As seen in Table VI, the use of a Bayes Filter and the multi-dimensional assignment algorithm (MDA) is crucial in a scenario largely dominated by occlusions. We can see the effect of reducing the length of the time window in MDA as the HLT’s performance drops drastically by reducing the width from 5-D to 2-D, especially for the scenario with trees. The proposed DeepHKCF trackers outperform HLT in the no-trees scenario by about 30% in terms of central location error, thus establishing its dominance in a scenario without occlusion.

In the previous section the proposed tracker is compared to the state-of-the-art discriminative trackers. In this section, we compare the proposed DeepHKCF tracker to the generative hyperspectral trackers [5], [6]. These trackers are designed for the DIRSIG scenarios and extensively use the multi-dimensional assignment algorithm (MDA) [50]. HFT [5] relies on off-line trained road and car classifiers to optimize the search space. The hyperspectral histograms are then computed in a sliding window to assign similarity scores. The obtained heat map is then thresholded and post-processed to find the blobs, which are assigned to the target using MDA. The HLT [6], on the other hand, learns a generative target model using hyperspectral likelihood maps rather than using off-line trained classifiers. The blobs are extracted from the final thresholded map and track statistics are updated from the past N frames using the MDA. Here, we compare DeepHKCF only with HLT in Table V since HFT relies on car and asphalt classifiers trained on the samples from the same scene.
H. Effect of ROI Size

The overlap between the adjacent ROIs is an essential part of the DeepHKCF tracker as it ensures consideration of the every single point in the full ROI by the correlation filter. Another key parameter in this direction is the full ROI size since we have a low temporal resolution and occlusion-dominated scene. We enlarge the ROI size of the optimal DeepHKCF tracker and observe the performance in the scene with trees. The results are shown in Fig. 8a. In this experiment, the run-time performance of the tracker is ignored as the goal is to measure the contribution of full ROI size.

As shown in Fig. 8b, the larger ROI size with the same overlap ratio does not necessarily lead to better performance while quadratically increasing the speed. This could be due to growing confusion as the larger ROIs contain a higher number of similar objects. Additionally, these results demonstrate the obvious need to couple the tracking-by-detection algorithms to a Multi-dimensional Assignment algorithm in a Bayes Filter framework in occlusion-dominated scenes [6, 5, 4].

VI. Conclusion

Adaptive multi-modal sensors are becoming increasingly important in the aerial tracking domain due to the unique challenges posed by this platforms. In this study, we propose a tracking-by-detection algorithm driven tracker inspired by a multi-modal sensor and deep features. This approach replaces the traditional template-matching based hyperspectral trackers with a new state-of-the-art tracker becoming increasingly popular in traditional visual object tracking. More specifically, we delivered a new framework to handle low temporal resolution in aerial platforms in KCF tracker, called single KCF-multiple ROIs approach. To further boost the tracking accuracy, we replaced the traditional features with deep CNN features. Finally, an ROI mapping approach was proposed to speed up extracting features in a single KCF-multiple ROIs approach. The proposed DeepHKCF tracker was evaluated on synthetic scenarios generated by DIRSIG software. In the scenario with no-trees, the DeepHKCF tracker performs exceptionally well with 80% precision at 50 px, outperforming other trackers. In the same scenario but dominated by occlusions, it is outperformed by trackers employing a multi-dimensional assignment algorithm and Bayes Filter. To prove the high-fidelity of the DIRSIG generated scenarios, we build a synthetic, aerial vehicle classification dataset to perform classification on the real-platform (WAMI). Our dataset, consisting of 55226 samples, was used to train CNNs to perform binary classification. We achieve about 93.2% on the WAMI samples by only training on synthetic dataset. This dataset can be highly beneficial in aerial detection and tracking due to limited amount of publicly available data in those domains.

In future work, we plan on supporting the DeepHKCF tracker by integrating a multi-dimensional assignment algorithm and Bayes Filter to better handle severe occlusions.

ACKNOWLEDGEMENTS

This work has been supported by the Dynamic Data Driven Applications Systems Program, Air Force Office of Scientific Research under Grant FA9550-11-1-0348.

REFERENCES

[1] R. Pelapur, S. Candemir, F. Bunyak, M. Poostchi, G. Seetharaman, and K. Pulaniappan, “Persistent target tracking using likelihood fusion in wide-area and full motion video sequences,” in Information Fusion (FUSION), 2012 15th International Conference on. IEEE, 2012, pp. 2420–2427.
[2] J. Portmann, S. Lynen, M. Chli, and R. Siegwart, “People detection and tracking from aerial thermal views,” in Robotics and Automation (ICRA), 2014 IEEE International Conference on. IEEE, 2014, pp. 1794–1800.
[3] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, “Eco: Efficient convolution operators for tracking,” arXiv preprint arXiv:1611.09224, 2016.
[4] B. Uzkent, M. J. Hoffman, and A. Vodacek, “Efficient integration of spectral features for vehicle tracking utilizing an adaptive sensor,” in IS&T/SPIE Electronic Imaging, 2015, pp. 940.707–940.707.
