Efficient Feature Extraction from Wide Area Motion Imagery by MapReduce in Hadoop

Erkang Cheng[a], Liya Ma[a], Adam Blaisse[a], Erik Blasch[b], Carolyn Sheaff[b], Genshe Chen[c], Jie Wu[a] and Haibin Ling[a]

[a]Department of Computer Information Sciences, Temple University, Philadelphia, PA, 19122
[b]Air Force Research Laboratory Rome, NY, 13441
[c]Intelligent Fusion Technology Germantown, MD 20876

ABSTRACT

Wide-Area Motion Imagery (WAMI) feature extraction is important for applications such as target tracking, traffic management and accident discovery. With the increasing amount of WAMI collections and feature extraction from the data, a scalable framework is needed to handle the large amount of information. Cloud computing is one of the approaches recently applied in large scale or big data. In this paper, MapReduce in Hadoop is investigated for large scale feature extraction tasks for WAMI. Specifically, a large dataset of WAMI images is divided into several splits. Each split has a small subset of WAMI images. The feature extractions of WAMI images in each split are distributed to slave nodes in the Hadoop system. Feature extraction of each image is performed individually in the assigned slave node. Finally, the feature extraction results are sent to the Hadoop File System (HDFS) to aggregate the feature information over the collected imagery. Experiments of feature extraction with and without MapReduce are conducted to illustrate the effectiveness of our proposed Cloud-Enabled WAMI Exploitation (CAWE) approach.

Keywords: WAMI, Feature extraction, Hadoop, MapReduce

1. INTRODUCTION

An increasing amount of large size video data requires a scalable and efficient computer vision framework to perform image exploitation in real time. A single computer solution is limited in processing power and storage. Current options with the high-performance computer (HPC) hardware are emerging, such as a multi-core central processing unit (CPU) and a graphics processing unit (GPU) card. A typical HPC example is to develop a distributed application using the message passing interface (MPI). Although a multi-core system is powerful, it is complicated to develop Wide-Area Motion Imagery (WAMI) applications using traditional parallel/distributed programming techniques.

MapReduce[1] was original introduced by Google as a programming model and an associated implementation for processing and generating large data sets. Recently, MapReduce technology is popular in both the academic and the industry community to handle large scale data. One of the main reasons is that there is an open source implementation of Hadoop. Hadoop, written in Java, is a top-level Apache project which offers the similar features as Google File System (GFS). With tremendous contributions from open source community, Hadoop is a chosen infrastructure of many companies such as Amazon, Facebook and Ebay and is able to store and process data on the order of the petabytes. An interested reader can refer a detailed list of companies using Hadoop.[2]

MapReduce in Hadoop is a distributed computing paradigm which provides a programming model for analyzing large data sets. MapReduce is based on the observation that large data processing tasks have the same basic structure: a computation is applied over a large number of distributed records to generate partial results, and then combines them to get the final outputs. More specially, there are two important concepts in MapReduce: *map* and *reduce*, where a *map* function processes a $< key, value >$ pair to generate a set of intermediate $< key, value >$ pairs; and a *reduce* function merges together all intermediate values associated to a given key to yield the final result. In a Hadoop framework, the master distributes data into slave nodes by using map function. Then slave nodes process their own data associated with a key value. The intermediate results of each node is then reduced by master to get the final results.

E-mail: {ekcheng, liya, tuc47904, jiewu, hbling}@temple.edu, {erik.blasch.1, carolyn.sheaff}@us.af.mil, gchen@intfusiontech.com

MapReduce is heavily used in large scale machine learning fields. The computer vision community also applies MapReduce in different applications. Liu et al.[3] proposed a face tracking algorithm that uses multiple cues and a particle filtering algorithm. The map functions were utilized in parallel over the particle predictions and the reduce functions computed the updated parameters. Li et al.[4] developed a landmark classification system which used a dataset of 6.5 million images taken from Flickr, where the Support Vector Machine (SVM) training is performed on a Hadoop cluster with 60 nodes. White et al.[2] provide a summary of current computer vision tasks involved with MapReduce.

There are several open source libraries for computer vision applications which make use of Hadoop. Picarus[5] is a computer vision and machine learning web service and library for large-scale visual analysis. Hadoopy[6] is the python based Hadoop framework. HIPI[7] is another open source library for computer vision tasks which offers the image based MapReduce. By converting images into Strings and then decoding each image in every map task, HIPI provides a high level framework for image representation in Hadoop.

Past reported results of WAMI techniques have resulted from the distribution of the Columbus Large Image Format (CLIF) data set.[11] Since the landmark publication by Porter,[12] feature extraction has been an important component of WAMI exploitation for tracking,[13,14,17,18] architecture design,[15] and registration.[16] Feature extraction has also supported methods for learning,[19] situation awareness,[20] and semantic uncertainty analysis.[21] One important aspect of WAMI tracking is feature extraction as a method of context-driven methods[22] using spatial,[23] temporal[24] context. Context supports patterns of life analysis,[26] traffic scenes,[27] and small target detection[25] in target tracking. Recently, feature extraction is included in activity analysis,[25] tracking,[29] and high throughput[30] applications. With many developments in WAMI technology, it is important to develop methods that are efficient, scalable, and timely using developments in software and hardware processing.

In this paper, the objective is to investigate MapReduce in Hadoop for large scale feature extraction tasks for WAMI. The experiments are conducted on the Columbus Large Image Format (CLIF) dataset. Our motivation is to divide a large dataset into several splits by the MapReduce framework in Hadoop, where each split contains a subset of images. Each split is then delivered to its corresponding slave node in the Hadoop system. Split WAMI images are processed by an assigned slave node. In our study, feature extraction of a WAMI image is performed in the slave nodes. Specifically, each split is a list of $< key, value >$ pairs, where key denotes the image name and $value$ is the actual data of the specified image. The input of map function in MapReduce is a pair $< key, value >$. The map function computes the feature extraction of the specified image. Experiments of feature extraction with and without MapReduce are conducted to illustrate the effectiveness of our proposed Cloud-Enabled WAMI Exploitation (CAWE) approach.

The rest of the paper is organized as follows. Introduction of Hadoop and MapReduce is listed in Sec. 2. In Sec. 3, feature extraction of WAMI using Hadoop is illustrated. Experimental results are presented in Sec. 4. Finally, Sec. 5 concludes the paper.

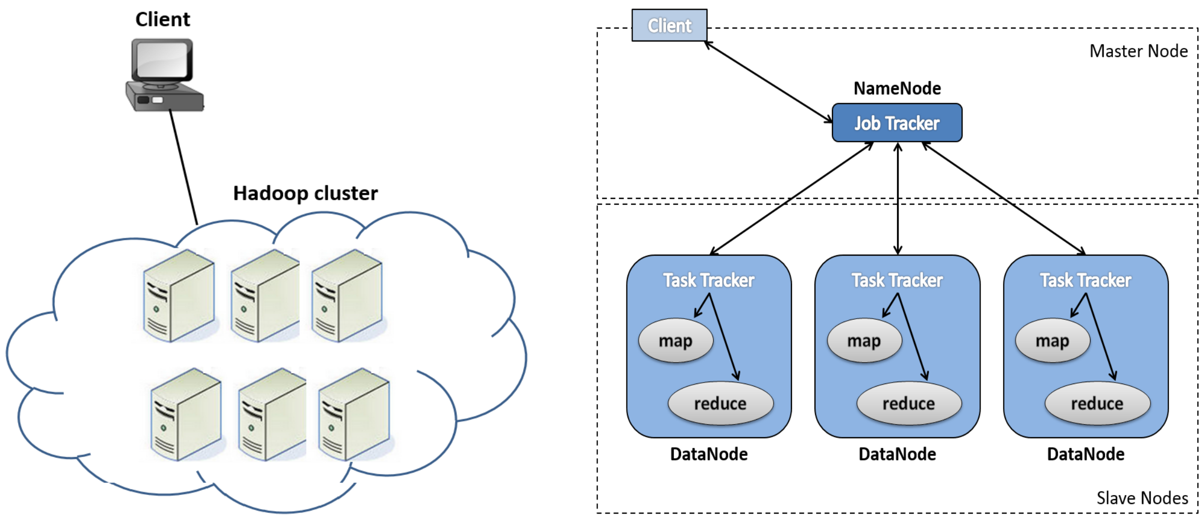
2. METHODS

In this paper, our goal is to investigate feature extraction on WAMI data using a Hadoop framework. Using a Hadoop framework, it is easy to scale the conventional feature extraction algorithm for a large scale dataset. Introduction of Hadoop and MapReduce is given in this section.

2.1 Hadoop

In general, Hadoop is an open source framework for processing large scale of data by distributing computations in a Hadoop cluster. The Hadoop system is also named Hadoop File System (HDFS). A typical Hadoop system is shown in Fig. 1 (a). A Hadoop cluster is a cloud network which has many parallel machines to store and process large data sets. A Client machine can communicate with the Hadoop cluster. Therefore, users have the ability to submit applications to a cluster from individual clients. Also, the computation results can be returned back to users through the HDFS.

A fully configured Hadoop cluster has several components.[8] These components include: NameNode, DataNode, Secondary NameNode, JobTracker and TaskTracker. The components of Hadoop is illustrated in Fig. 1 (b). The NameNode is the master of HDFS that directs the slave DataNode daemons to perform the low-level input/output tasks. Each slave machine in the cluster hosts a DataNode daemon to perform the assigned work by the JobTacker. Secondary NameNode is an assistant daemon of NameNode. JobTracker determines the execution plans of the work loaded from the client. The JobTracker will decide which files to process, assign tasks to each slave node and also monitor these assigned tasks. Then, each TaskTracker is responsible for the work assigned to itself and reports to the JobTracker.



(a) (b)

Figure 1. (a) A Hadoop cluster has many parallel machines that stores and processes large data sets. Client computers send jobs into this computer cloud and obtain results. (b) The components of Hadoop system.

2.2 MapReduce

MapReduce in Hadoop is a distributed computing technology which provides a programming model for processing large data sets. There are two core concepts in a MapReduce procedure: map and reduce. Each phase has $< key, value >$ pairs as an input and output.

- Map: specifies a function that processes a $< key, value >$ pair to generate a set of intermediate $< key, value >$ pairs.
- Reduce: merges together all intermediate $< key, value >$ pairs associated with a given key to compute the final result.

More specifically, in the map phase, users can specify a map function that processes the input, a list of $< key, value >$ pairs and generates a set of intermediate $< key, value >$ pairs. And in map phase, a reduce function that merges all intermediate $< key, value >$ pairs associated with the same key. These intermediate values are then used to compute the final results.

Fig. 2 gives the data flow of MapReduce. In a Hadoop framework, the master distributes data into slave nodes by using a user-specific map function. The input is divided into fixed-size splits by HDFS. Each split is constructed to a $< key, value >$ pair by map function. Then, each slave node processes its assigned $< key, value >$ pairs and obtains intermediate outputs which are also $< key, value >$ pairs. These $< key, value >$ pairs are then merged by the reduce function to get the final results. Therefore, the master performs the task distribution and jobs monitoring functions. The slave node only executes its own task assigned by the master.

3. WAMI FEATURE EXTRACTION USING HADOOP

3.1 SIFT Feature Extraction

Feature extraction in Wide-Area Motion Imagery (WAMI) is an important task required for applications such as target tracking, traffic management and accident discovery. In this paper, the Scale Invariant Feature Transforms (SIFT) feature extractor is investigated for WAMI data. SIFT[9] has been proven to be one of the most robust local invariant feature descriptors. The computation of SIFT contains four stages: scale space extrema detection; keypoint localization; orientation assignment; and keypoint descriptor.

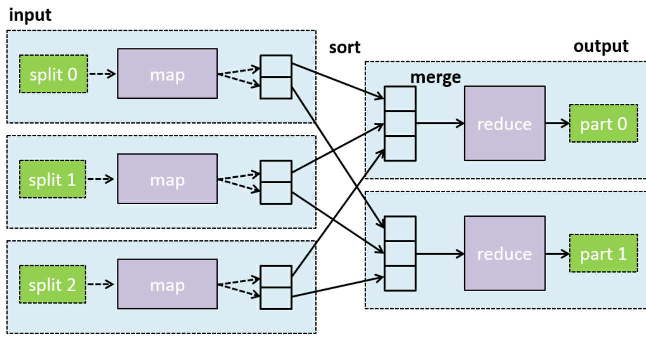
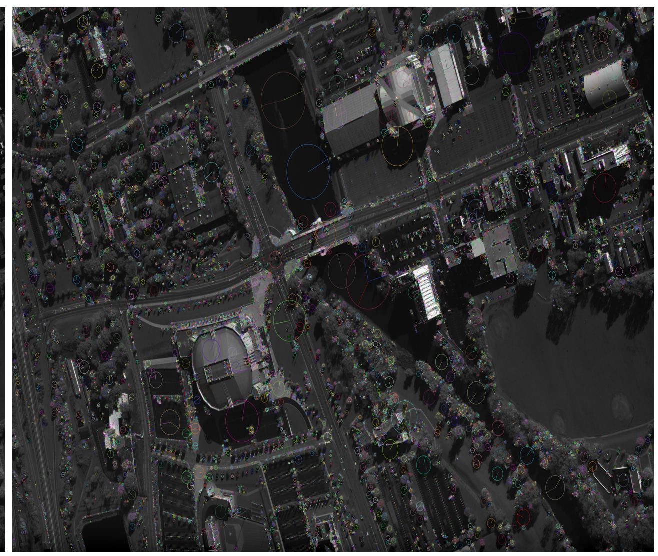


Figure 2. MapReduce data flow with multiple reduce tasks.[8]

(a) Original WAMI image



(b) SIFT keypoints (in color)

Figure 3. Feature extraction of a WAMI image.

Specially, in this paper, keypoints localization is evaluated. The first step is to identify keypoints in scale space by checking for locations that are extrema of a difference-of-Gaussian function (DOG). Let the scale space of an image be $L(x, y, \delta)$, obtained by convolving the image $I(x, y)$ and the Gaussian kernel $G(x, y, \delta)$. The pixel is located at the x, y position, and δ is parameter of the Gaussian kernel. Differencing of two nearby scaled images separated by a constant factor k is denoted as:

$$D(x, y, \delta) = L(x, y, k\delta) - L(x, y, \delta). \tag{1}$$

Local extrema of a DOG function are extracted as keypoints. Each point is checked to see whether it is larger or smaller than all neighbors, using neighboring pixels as defined by a selected image location and scale operator. Fig. 3 gives of examples of keypoints extraction of SIFT.

3.2 MapReduce for WAMI

Usually, a WAMI data set contains numerous images. Also, one WAMI image is contains a large number of pixels. It is time expensive to process the large-scale data on one machine.

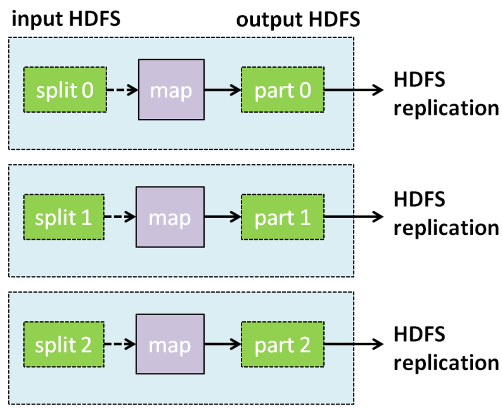


Figure 4. MapReduce data flow without reduce tasks.[8]

Our motivation is using the MapReduce techniques for feature extraction is to speed up the analysis. There are two feasible solutions by using MapReduce in Hadoop. The first one is to follow the standard MapReduce data flow as given in Fig. 2. In this approach, the dataset is divided into several groups, where each group contains a subset of WAMI images. With the consideration of large size of WAMI image, each image is further divided into several parts (splits). Thus, a split is a $< key, value >$ pair for map function, where key is the image file name and the $value$ is the image data of current split. Eventually, by dividing an image I into $m \times n$ splits, it generates $(m \times n) < key, value >$ pairs. These pairs have the same key to indicate they belong to the same image. Image data of corresponding split and related coordinate information are recorded in $value$. Hadoop distributes the splits into slave nodes in the cluster. Each slave node performs the assigned task which is the SIFT feature extraction in our work. Specially, the map function processes the input $< key, value >$ pair to execute the SIFT feature extraction. The immediate output of the map function is also a $< key, value^* >$ pair in our application. $value^*$ contains information of SIFT keypoints of specified image part defined by $< key, value >$ pair. The reduce function merges the immediate output of map functions. Therefore, $< key, value^* >$ pairs with the same key are combined together to form a whole image. The output image is eventually the SIFT keypoints of an image with file name key. The limitation is that it requires additional computations to divide one image into several parts. Moreover, it usually generates large number of splits in a Hadoop system which in turn needs more communication between the NameNode and DataNode.

Another straightforward solution is to divide the dataset into several groups (splits), where each group has a subset of images. Therefore, each image denoted by a $< key, value >$ pair is an input for map function in MapReduce. Typically, a key is the image file name and the $value$ represents data of the image. In such configuration, each $< key, value >$ pair corresponds to one WAMI image. Similarly, Hadoop distributes the splits into slave nodes in the cluster. Each slave node performs the assigned task which is a SIFT feature extraction in our work. Specially, the map function processes the input $< key, value >$ pair to execute the SIFT feature extraction of an image associated with the input pair. The output of the map function is also a $< key, value^* >$ pair, where $value^*$ is the image recorded keypoints extracted by SIFT. Consequently, the SIFT keypoint results are written back to HDFS. A user can then query the SIFT feature extraction results through the client machine. It is a simplified MapReduce framework because there is no reduce function in this approach. Fig. 4 provides the illustration of MapReduce data flow without the reduce step. This solution is intuitively suitable for most computer vision applications. We adapt this framework for our SIFT feature extraction task. The entire process is summarized in Algorithm. 1.

4. EXPERIMENTS

4.1 Data Set

Our experiments are conducted on the CLIF dataset. Two datasets are evaluated in the experiments. The first one has 1000 WAMI images and there are 2000 WAMI images in the second dataset. These WAMI images are a size of 1004×668 pixels.

Algorithm 1 MapReduce for feature extraction of WAMI

1: Input: A dataset of WAMI images;
2: Output: Feature extraction of input images;
3: Build Hadoop system;
4: Upload image data into HDFS;
5: HDFS splits images into a set of $< key, value >$ pairs, where each image corresponds to one pair.
6: HDFS distributes $< key, value >$ pairs and assigns a list of $< key, value >$ pairs to slave machines (DataNode);
7: Each slave machine processes a list of $< key, value >$ pairs by a map function. The map function processes an input $< key, value >$ pair to perform feature extraction of an image defined by the input $< key, value >$ pair. The map function generates a $< key, value^* >$ pair output. key of the output pair is the same of the input pair. $value^*$ of the output pair records the feature extraction result.
8: Each output pair $< key, value^* >$ of a map function is saved back to HDFS as a feature extraction result.

4.2 Build Hadoop System

We follow the description[10] to build the Hadoop system on Ubuntu Linux with multiple nodes. The version of Ubuntu is 12.04 and Hadoop version is Hadoop 1.2.0. The Java version used in the experiment is Sun 1.7. Hadoop requires Secure Shell (SSH) to access and manage its nodes. The communication between master nodes and slave nodes are based on SSH. We therefore need to configure SSH access to Hadoop for the client. An Rivest-Shamir-Adleman (RSA) key pair is created for the communication.

The Hadoop system configuration has 16 Dell PowerEdge R210 II servers. Each server has an Intel dual core Celeron G530 processor at 2.4GHz. Each server has 4GB of Read Access Memory (RAM) and a 500GB harddrive. The servers are then connected using Cisco Small Business 300 Series Managed Switches.

4.3 Evaluation of Feature Extraction

We compare the efficiency of feature extraction using the CLIF dataset. Two configurations are applied in the experiments. The first one is to process all the images on a single machine. The other is to execute feature extraction on the Hadoop system with MapReduce. The average computation time is used for evaluation.

Table. 1 and Table. 2 list the comparison results of the two protocols. In the first dataset, the average run time is reduced 27.44% by using MapReduce framework in a Hadoop system as compared to a not using MapReduce. The run time decreases by 13.69% in the second dataset with 2000 WAMI images in it. The results show that a Hadoop system with MapReduce is efficient to process the large datasets.

Table 1. Run time (in seconds) of feature extraction on dataset 1.

	Single machine	MapReduce
1st run	375.4900	358.1010
2nd run	374.1250	325.1100
3rd run	373.3420	324.1190
4th run	370.9130	318.8710
5th run	368.1400	322.7410
Average	372.4020	329.7884

5. CONCLUSION

In this paper, an efficient approach for Wide Area Motion Imagery (WAMI) feature extraction on a Hadoop system is proposed. Comparison with and without MapReduce on the CLIF dataset demonstrates the efficiency of our proposed framework. In future, other feature extraction algorithms will be studied on Hadoop. Moreover, machine learning algorithms could be also developed within a Hadoop system for WAMI data. Building our multiple methods, the Cloud-Enabled WAMI Exploitation (CAWE) system can increase the timeliness of WAMI tracking and target exploitation for real-time situation awareness.

Table 2. Run time (in seconds) of feature extraction on dataset 2

	Single machine	MapReduce
1st run	775.4380	687.3260
2nd run	796.6300	688.6130
3rd run	797.3640	693.6470
4th run	782.0860	689.8910
5th run	778.6830	684.5060
Average	798.0402	688.7966

Acknowledgement

This research was supported in part by the Air Force Research Laboratory, Rome, NY under contract number FA8750-14-C-0043. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force.

REFERENCES

[1] Dean, J. and Ghemawat, S., Mapreduce: simplified data processing on large clusters. In *Communications of the ACM*, 51(1), 107–113 (2008).

[2] White, B., Tom, Y., Jimmy, L., Davis L., Web-scale computer vision using mapreduce for multimedia data mining. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, (2010).

[3] Liu, K., and Li, S., and Tang, Liang., Wang, Lei., Liu, Wei., Fast face tracking using parallel particle filter algorithm. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, (2009).

[4] Li, Y., Crandall, D., Huttenlocher, D., Landmark classification in large-scale image collections. In *Proc. of International Conference on Computer Vision*, (2009).

[5] http://www.picar.us/en/latest/

[6] http://www.hadoopy.com/en/latest/

[7] Sweeney, C., Liu, L., Arietta, S., Lawrence, J., HIPI: A Hadoop Image Processing Interface for Image-based MapReduce Tasks. In *Chris. University of Virginia*, (2011).

[8] Lam C., Hadoop in action. In *Manning Publications Co.*, (2010).

[9] Lowe, D., Object recognition from local scale-invariant features. In *Proc. of International Conference on Computer Vision*, (1999).

[10] http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-multi-node-cluster/

[11] Mendoza-Schrock, O., Patrick, J. A., Blasch, E., Video Image Registration Evaluation for a Layered Sensing Environment. In *Proc. IEEE Nat. Aerospace Electronics Conf. (NAECON)*, (2009).

[12] Porter, R., Fraser, A. M., Hush, D., Wide-Area Motion Imagery: Narrowing the Semantic Gap. In *IEEE Signal Processing Magazine*, Vol.27, No. 5, pp 56-65, (2010).

[13] Palaniappan, K., Bunyak, F., Kumar, P., Ersoy, I., Jeager, S., Ganguli, K., Haridas, A., Fraser, J., Rao, R. M., Seetharaman, G., Efficient feature extraction and likelihood fusion for vehicle tracking in low frame rate airborne video. In *Intl. Conf. on Information Fusion*, (2010).

[14] Ling, H., Wu, Y., Blasch, E., Chen, G., Bai, L., Evaluation of visual tracking in extremely low frame rate wide area motion imagery. In *Int. Conf. on Info Fusion*, (2011).

[15] Palaniappan, K., Rao, R. M., Seetharaman, G., Wide-area persistent airborne video: Architecture and challenges. In *Distributed Video Sensor Networks*, Springer, pp 349-371, (2011).

[16] Wu, Y., Chen, G., Blasch, E., Ling, H., Feature Based Background Registration in Wide Area Motion Imagery. In *Proc. SPIE*, Vol. 8402, (2012).

[17] Reilly, V., Idrees, H., Shah, M., Detection and tracking of large number of targets in wide area surveillance In *ECCV*, pp. 186–199, (2010).

[18] Prokaj, J., Medioni, J., Using 3d scene structure to improve tracking. In *CVPR*, pp. 1337–1344, (2011).

[19] Liang, P., Teodoro, G., Ling, H., Blasch, E., Chen, G., Bai, L., Multiple Kernel Learning for Vehicle Detection in Wide Area Motion Imagery. In *Int. Conf. on Info Fusion*, (2012).

[20] Blasch, E., Seetharaman, G. Palaniappan, K., Ling, H., Chen, G., Wide-Area Motion Imagery (WAMI) Exploitation Tools for Enhanced Situation Awareness. In *IEEE Applied Imagery Pattern Recognition Workshop*, (2012).

[21] Blasch, E., Costa, P. C. G., Laskey, K. B., Ling, H., Chen, G., The URREF Ontology for Semantic Wide Area Motion Imagery Exploitation. In *Proc. of the Seventh International Conference on Semantic Technologies for Intelligence, Defense, and Security (STIDS)*, (2012).

[22] Shi, X., Ling, H., Blasch, E., Hu, W., Context-Driven Moving Vehicle Detection in Wide Area Motion Imagery. In *Intl Conf. on Pattern Recognition (ICPR)*, (2012).

[23] Liang, P., Shen, D., Blasch, E., Pham, K., Wang, Z., Chen, G., Ling, H., Spatial Context for Moving Vehicle Detection in Wide Area Motion Imagery with Multiple Kernel Learning. In *Proc. SPIE*, Vol. 8751, (2013).

[24] Liang, P., Ling, H., Blasch, E., Seetharaman, G., Shen, D., Chen, G., Vehicle Detection in Wide Area Aerial Surveillance using Temporal Context. In *Intl Conf. on Info Fusion*, (2013).

[25] Mathew, A., Asari, V. K., Tracking small targets in wide area motion imagery data. In *Proc. SPIE*, Vol. 8663, (2013).

[26] Gao, J., Ling, H., Blasch, E., Pham, K., Wang, Z., Chen, G., Pattern of life from WAMI objects tracking based on visual context-aware tracking and infusion network models. In *Proc. SPIE*, Vol. 8745, (2013).

[27] Shi, X., Li, P., Hu, W., Blasch, E., Ling, H., Using Maximum Consistency Context for Multiple Target Association in Wide Area Traffic Scenes. In *Int'l Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, (2013).

[28] Levchuck, G., Detecting coordinated activities from persistent surveillance. In *SPIE Newsroom*, 5 June (2013), DOI: 10.1117/2.1201305.004886).

[29] Gao, J., Ling, H., Blasch, E., Pham, K., Wang, Z., Chen, G., Context-aware tracking with wide-area motion imagery. In *SPIE Newsroom*, (2013).

[30] Basharat, A., Turek, M., Xu, Y., Atkins, C., Stoup, D., Fieldhouse, K., Tunison, P., Hoogs, A., Real-time multi-target tracking at 210 megapixels/second in wide area motion imagery. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, (2014).