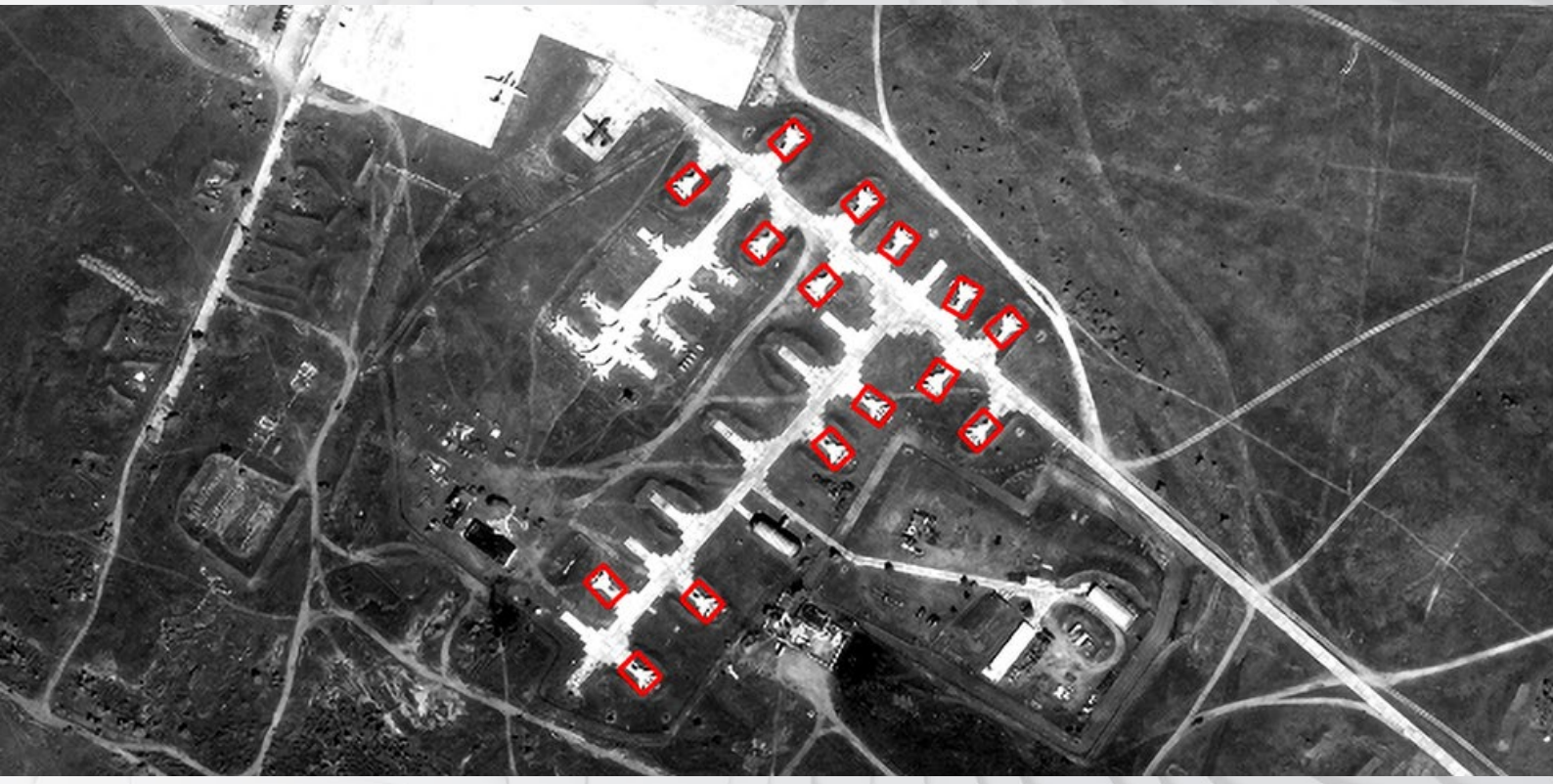
BAE SYSTEMS



Imagery courtesy of Maxar Technologies.

DeepObject™

Technical Brief

GXP®
Geospatial eXploitation Products™

Geospatial solutions
to ensure a safer world.

Table of Contents

1. Executive summary

The Geospatial Intelligence (GEOINT) community is facing a big data challenge; data is being collected faster than analysts can process it. To detect geospatial objects automatically from geospatial imagery and 3-D point clouds, DeepObject™ uses a branch of artificial intelligence (AI) known as deep learning. DeepObject helps Image Analysts (IAs) be more efficient and productive by performing the initial object detection, which means the IAs only have to verify objects from the initial detections. The DeepObject workflow improves productivity of IAs by an order of magnitude based on this case study.

DeepObject has four innovations:

1. Simplicity Learning
   - Improves recall defined as (true positive) / (true positive + false negative) by simplifying the learning objective and reducing variations in pixels with three normalizations: scale normalization, rotation normalization, and color normalization.
2. Double Convolutional Neural Networks (CNNs)
   - Achieves both high detection speed and high positional accuracy when objects are sparse in geospatial imagery.
3. Rotation Pattern Match (RPM)
   - Improves precision defined as (true positive) / (true positive + false positive) by using CNN as an object descriptor, generating rotation patterns from only positive training samples, and matching against these rotation patterns.
   - Provides a partial solution to the "almost unlimited negative training samples" challenge and offers a more robust solution to the "perturbation attacks" in deep learning (Zhang, 2017b).
4. Sub-pixel Relative Positional Accuracy
   - DeepObject achieves sub-pixel positional accuracy by dedicating a separate model to estimating the precise locations of objects.

DeepObject uses five metrics for benchmarks:

1. Precision
2. Recall
3. Speed
4. Positional accuracy
5. Number of training samples

There are two unique challenges that pose difficulties in object detection geospatial imagery: the number of positive training samples and positional accuracy. For example, it is difficult for an IA sifting through thousands of training samples and trying to locate specific fighter jets. However, with DeepObject, these two requirements are met based on our four innovations. The final output objects are in ground space coordinate systems with three accuracy metrics: confidence level, horizontal positional accuracy, and vertical positional accuracy. These three metrics are essential for applications such as precision targeting, site monitoring, and change detection.

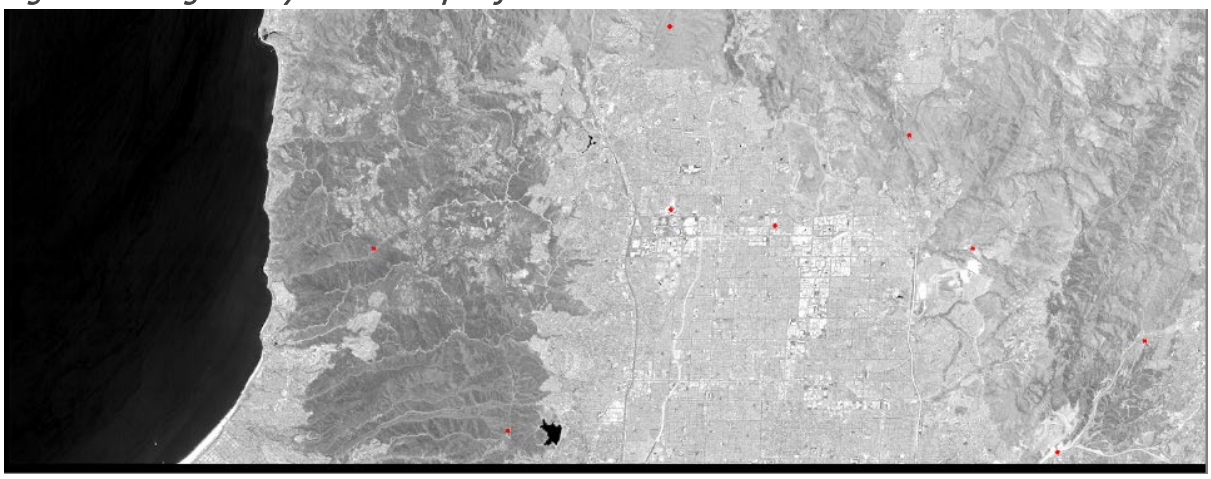
DeepObject can be used to transform digital photogrammetry to 'Intelligent Photogrammetry'. One of the most time consuming tasks in photogrammetry is Digital Elevation Model (DEM) generation. Our case study indicates that DeepObject DEM can achieve root mean square error (RMSE) of 0.95 meters from stereo satellite images.

2. Image Analysts (IAs) vs. DeepObject

DeepObject helps IAs to be much more efficient and productive by eliminating fatigue and time constraints. IAs take contextual information and reasoning and make more accurate detections than DeepObject. In this case study, we used three IAs who have a combined 50 years of image analysis experience; Analyst1, Analyst2, and Analyst3. The IAs were asked to find small airplanes in a timely manner from a huge WorldView image (courtesy of Maxar Technologies). Unfortunately, like the rest of the GEOINT community, the IAs succumbed to time constraints and fatigue, proving DeepObject to outperform each IA.

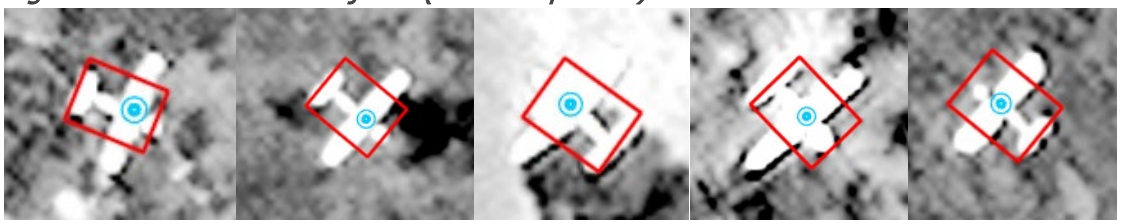
The first image studied was a WorldView image that covers 876 km^2, has 3,340 million pixels with an image Ground Sample Distance (GSD) of 0.51 meters (Figure 1). There are several small airplanes which, even for the IAs, are difficult to detect. Since there is no airport in the image, the IAs had to search the entire image very carefully. Using the SOCET GXP® Visual Identification search tool, Analyst1 found five small airplanes in 10 hours and 40 minutes at a search zoom level of 200%. Analyst2 used a search zoom level of 65% and found four small airplanes in 2 hours. Analyst3 used a search zoom level of 100% and found 4 small airplanes in 8 hours. DeepObject detected 9 small airplanes in 43 minutes using a desktop computer of Intel® Xeon® CPU E5-2650 v4 @ 2.20 GHz with two NVIDIA TITAN V GPUs (Table 1).

Figure 1. Image Analysts vs. DeepObject.



Analyst1 spent 10 hours and 40 minutes to find five small airplanes (blue circles in Figure 2). DeepObject took 43 minutes to detect nine small airplanes (red dots in Figure 1). Analyst1 used the following method: Visual identification using the automatic vertical roaming tool in SOCET GXP software; zoom level 200%; roaming speed in urban areas – 240 to 480 pixels per second; roaming speed in non-urban areas – 360 to 600 pixels per second. Imagery courtesy of Maxar.

Figure 2. Five detected objects (small airplanes)



Red bounding boxes are from DeepObject and blue dots + circles are from Analyst1. Detected objects from Analyst1 have similar positional accuracy (about 1 pixel). The positional accuracy of DeepObject is measured as the distances between the center of an object vs. the center of the corresponding bounding box. Imagery courtesy of Maxar.

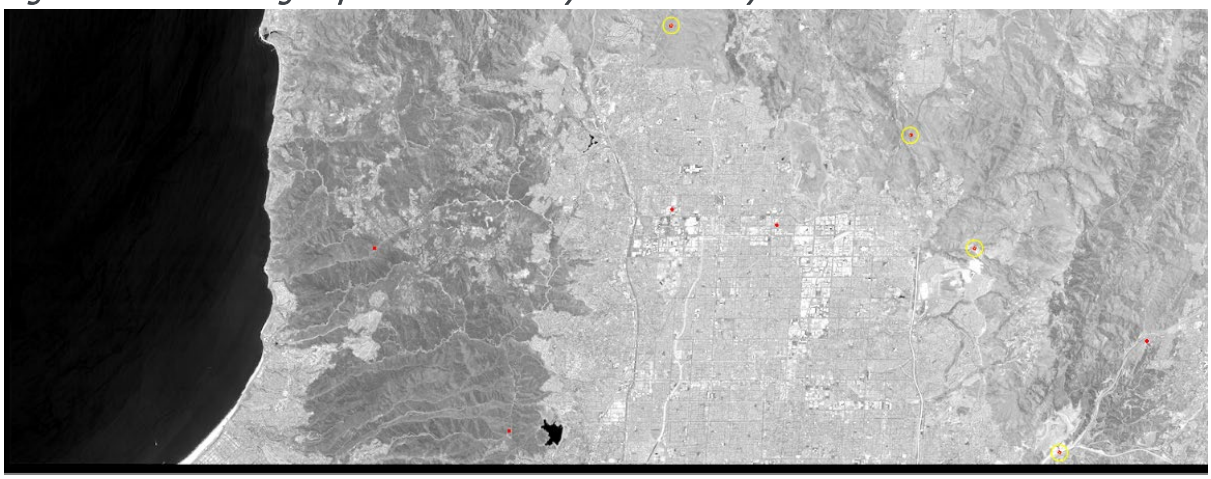
The positional accuracy of an IA is his visual ability to place a point on the perceived center of an object, which is not well-defined as shown in the first three small planes (Figure 2). In other words, the centers of objects may be arbitrary for an IA. DeepObject uses its fourth model to estimate the precise center positions of an object, which is similar to least squares adjustment in photogrammetry.

Figure 3. Four missing airplanes from Analyst1.

IAs could always do better than DeepObject if there were no fatigue or time constraints. IAs can take contextual information and reasoning to make more accurate detections than DeepObject. Analyst1 needed to take a break every 90 minutes due to fatigue while DeepObject ran constantly. Imagery courtesy of Maxar.

Analyst1 missed the four airplanes because the background of the first missing airplane (left) is indistinguishable unless appropriate dynamic range adjustment (DRA) is locally applied to this small region. During the search, SOCET GXP Multiport™ applied one set of DRA to all pixels in the Multiport instead of just a small localized region; the third and fourth missing airplanes were so small that it is very difficult to identify them when the zoom level is less than 500%. If Analyst1 had used a 500% zoom level, he would have to spend more than 42 hours instead of 10 hours and 40 minutes. Analyst2 and Analyst3 both identified the second missing plane, however, they both missed two other small planes. DeepObject outperformed Analyst1 in this case due to time constraints. It would have taken only a few minutes to verify the nine detections from DeepObject if Analyst1 ran DeepObject prior to the search. This workflow makes an IA an order of magnitude with a productivity gain of hours to minutes.

Figure 4. Five missing airplanes from Analyst2 and Analyst3.



Analyst2 used a zoom level between 60%-70% (vs. Analyst1's 200%) and found four airplanes (yellow circles) but missed five small airplanes in 2 hours. Analyst3 found the same four airplanes in 8 hours. Analyst2 used much less time than Analyst1, however, Analyst2 missed more detections due to time constraints. Analyst2 missed five airplanes because the zoom level is about 65% and the five missing airplanes are too small to identify at 65% zoom level. Analyst2 would have detected all airplanes if he had used a zoom level of > 300%, and had spent 40 hours instead of 2 hours. Imagery courtesy of Maxar.

Figure 5. Airplane missed by Analyst1, but detected by DeepObject, Analyst2, and Analyst3.

The yellow circle is where Analyst2 placed the center, which is several pixels off vs. the center from DeepObject, which is within 1 pixel positional accuracy. Imagery courtesy of Maxar.

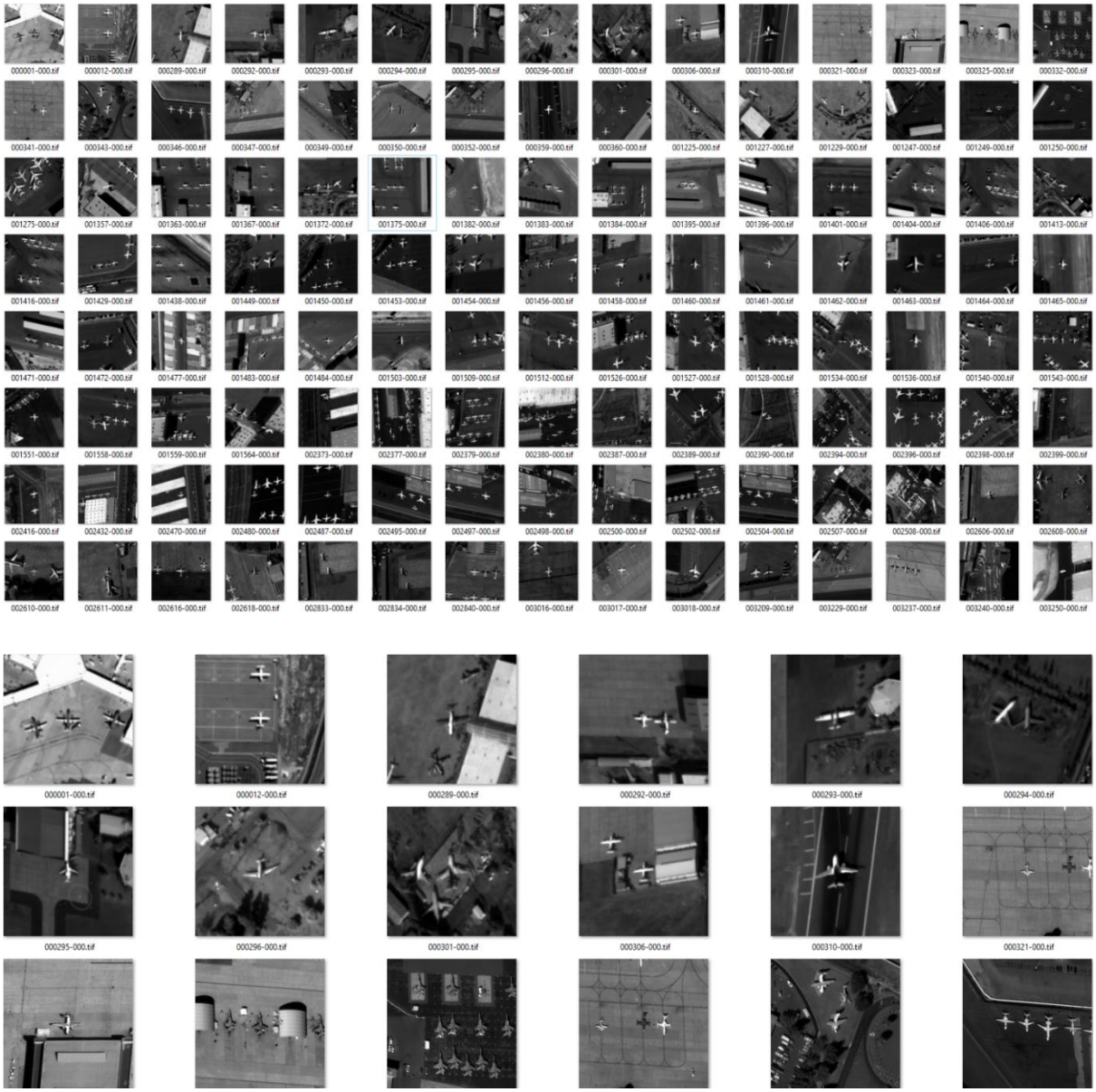
Table 1. Image Analysts vs. DeepObject on small airplane detection metrics.

	Precision	Recall	Time (minute)	Positional accuracy (pixel)
Analyst1	100.0%	55.5%	640	1
Analyst2	100.0%	44.4%	120	1
Analyst3	100.0%	44.4%	480	1
DeepObject	100.0%	100.0%	20	1

IAs can use DeepObject to perform the primitive and time-consuming initial detections, and then they can verify the objects from those detections. This workflow can significantly improve productivity of IAs. IAs can achieve 1 pixel positional accuracy while DeepObject can also achieve 1 pixel positional accuracy. If IAs had unlimited time, they could detect all objects. DeepObject is much faster than IAs. For recall, which is defined as true positive/(true positive + false negative), the higher the better.

Out of the 832 positive training samples and 3,517 negative training samples used to train DeepObject, none of the positive training samples were collected from our detection image (Figure 6).

Figure 6. Positive training samples.



There are many different types of airplanes, some in the air and some that are grounded, and the way they appear depends on the angles and lighting conditions (shadows). All positive training samples were collected from grounded airplanes at airports, while the detection image in this Image Analysts vs. DeepObject case study shows small airplanes that are flying. Flying airplanes have very different background images than airports. Imagery courtesy of Maxar.

DeepObject can achieve higher accuracy by more training using false positives and false negatives as training samples. The more we use DeepObject, the higher accuracy DeepObject can achieve when we use false positives and false negatives as training samples for DeepObject.

DeepObject can help IAs to be much more productive when it is used to perform the primitive and time-consuming initial object detections. Given the initial detections from DeepObject to the three IAs, they would only have to spend a few minutes to

verify the detections instead of hours to find these same objects. The productivity gain is hours vs. minutes.

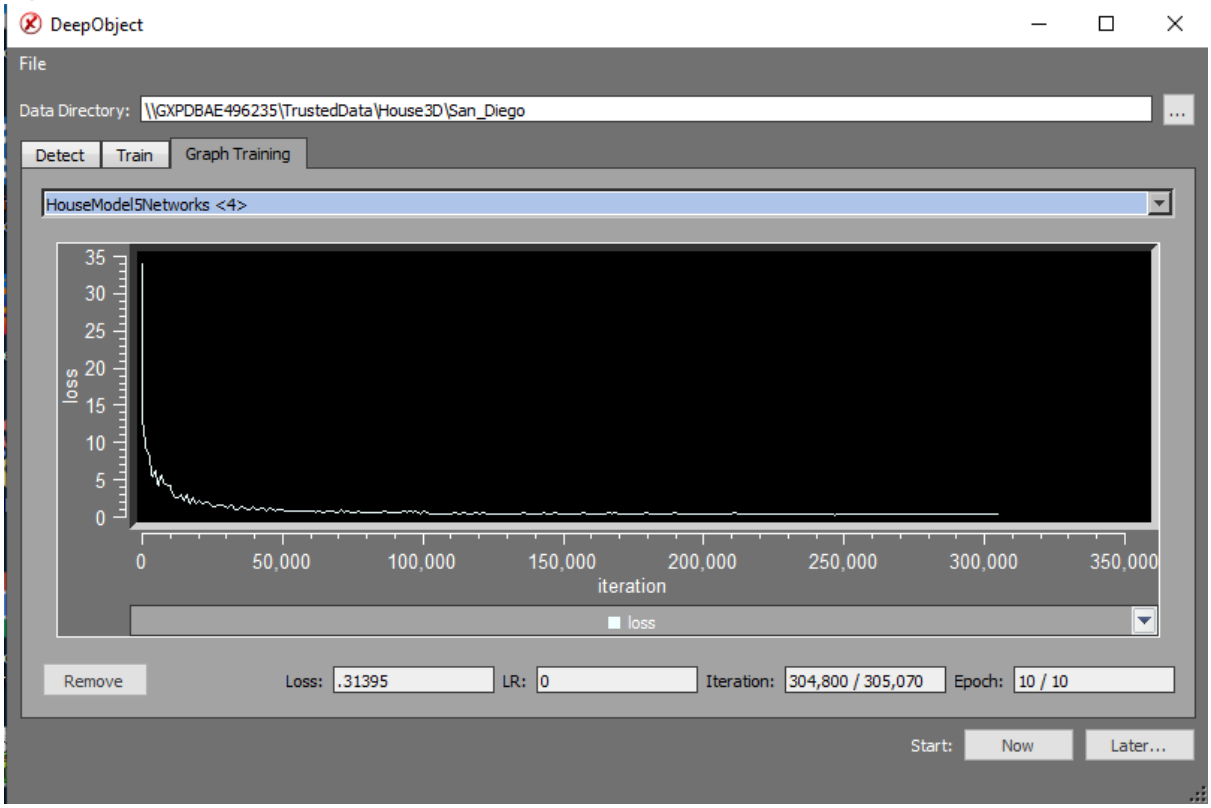
3. Achieving sub-pixel relative positional accuracy

DeepObject uses a separate model to estimate the precise locations of objects. The fourth model of DeepObject can achieve average positional accuracy of 1/3 of a pixel (Figure 7). After 305,070 iterations of training, the average loss is 0.31 pixel. The average loss is a Euclidean distance loss computed as:

$$loss = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(x^i - x_o^i)^2 + (y^i - y_o^i)^2} \quad (1)$$

Where, (x_o, y_o) are the true sample and line coordinates, and (x, y) are the estimated sample and line coordinates.

Figure 7. Sub-pixel positional accuracy.

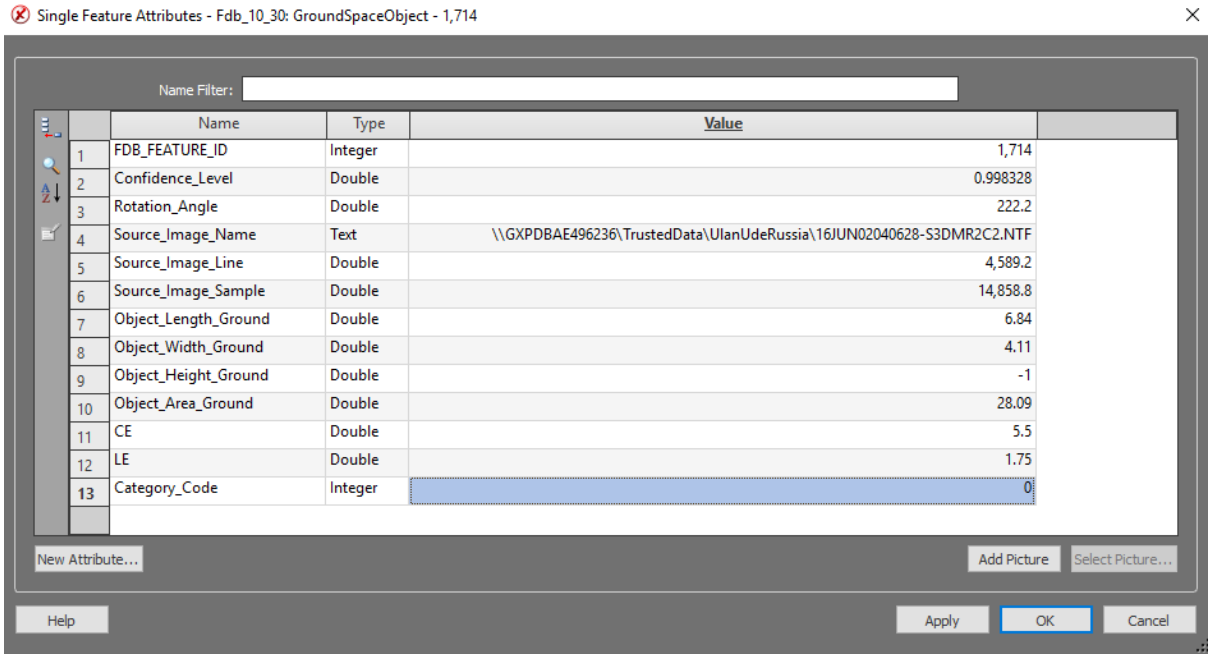


The average loss is 0.31 pixel after 305,070 iterations of training.

The final output objects are in ground space coordinate systems with three accuracy metrics: horizontal absolute positional accuracy (CE), vertical absolute positional accuracy (LE), and confidence level (Figure 8). For applications such as precision targeting, we need object coordinates (X, Y, Z) in a ground space coordinate system.

We also need the confidence level, which measures the reliability of the automatically detected object. DeepObject uses photogrammetric, rigorous error propagation algorithms to estimate the horizontal absolute positional accuracy (CE = 5.5 meters) and vertical absolute positional accuracy (LE = 1.75 meters) (Figure 8).

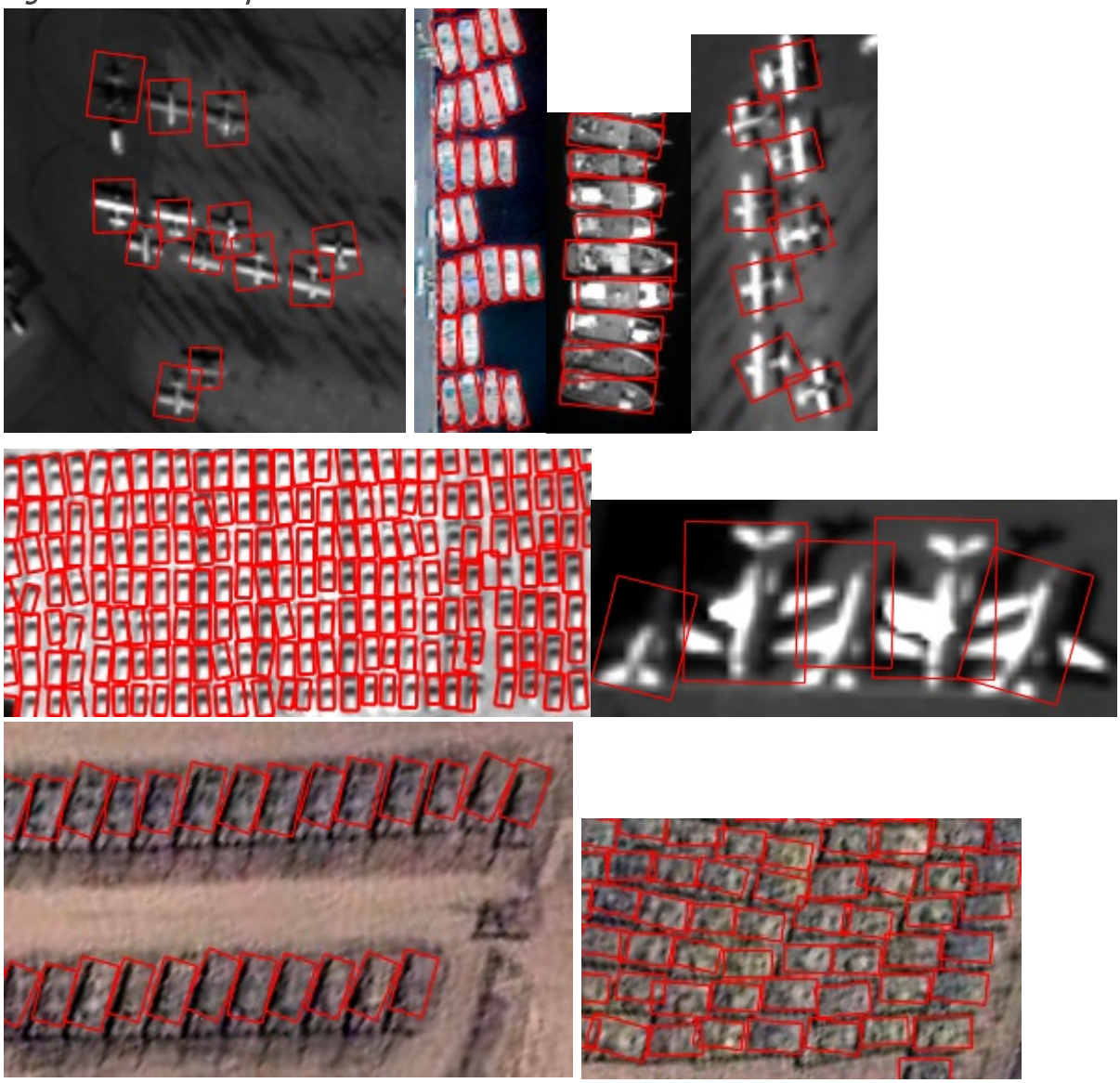
Figure 8. Three accuracy metrics: Confidence level, CE, and LE.



Confidence level is the probability of a positive object detection measured at 99.78%. CE and LE are absolute positional errors measured at 90% probability in XY and Z respectively in meters.

With sub-pixel relative positional accuracy, DeepObject can detect objects in highly clustered patterns, however, objects such as airplanes may overlap (Figure 9). For satellite images, the gap between small objects such as tanks and cars may be less than 1 pixel. Achieving less than 1 pixel relative positional accuracy enables DeepObject to separate those objects.

Figure 9. Clustered patterns.



Small planes overlap up to 30%. Gaps between adjacent tanks are less than 1 pixel. Gaps between parked cars from satellite imagery (GSD = 0.35 meters) are less than 1 pixel. Small boats are chained together. Red bounding boxes are from DeepObject output feature database. Imagery courtesy of Maxar.

4. See the *almost* invisible

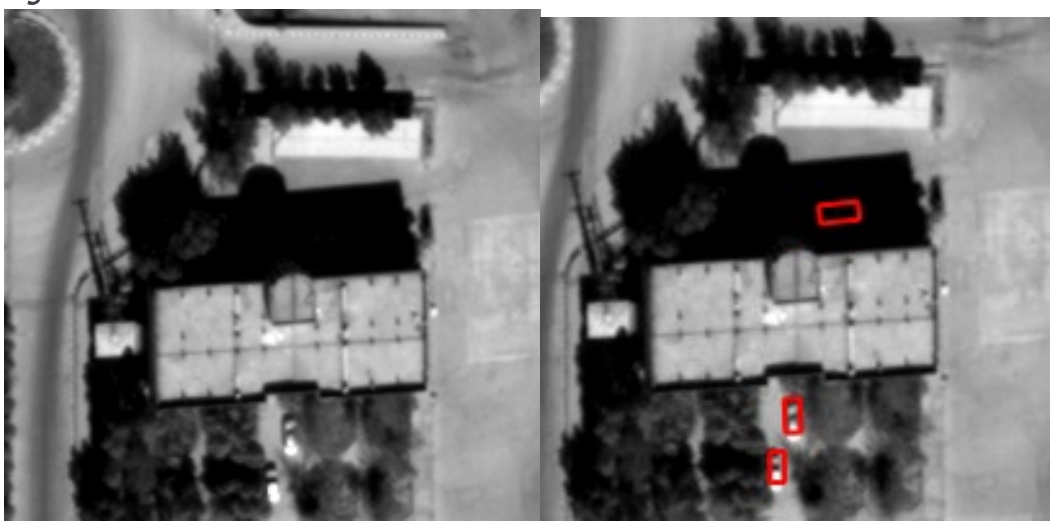
DeepObject can detect objects that are ***almost*** invisible to human vision. Objects in shadows may not be visible to the human eye unless we apply a local DRA with a zoom level > 400% (Figures 10, 11, 12, 13, and 14). Alternatively, we may adjust image brightness and contrast to bring light to objects that are in shadows at the cost of blurring objects that are not in shadows. DeepObject uses a window sliding algorithm to detect objects. Inside each window of pixels, DeepObject applies the three normalizations (Simplicity Learning) to make objects *visible to DeepObject* by supressing background pixels and enhancing object pixels.

Viewed at 100% zoom level with the DRA on, the car in the shadow is not visible to the human eye as shown (Figure 11). Captured 200% zoom level with DRA on, the car in the shadow is still not visible to the human eye (Figure 12). At 400% zoom level, the car in the shadow becomes barely visible (Figure 13).

Figure 10. Almost invisible object detected by DeepObject.

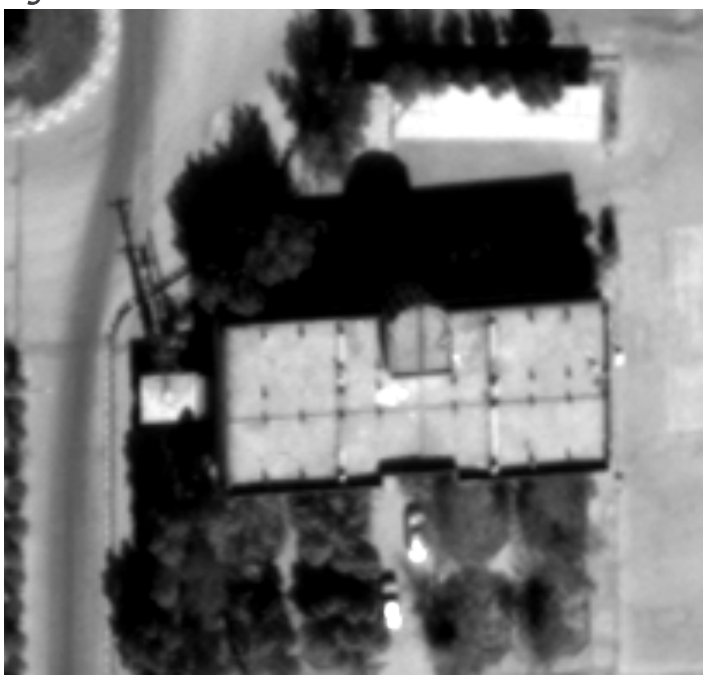
There are seven visible cars in this WorldView-3 image with GSD of 0.34 meters. There is one almost invisible car in the building shadow that is also detected by DeepObject. The above image is displayed at 100% zoom level. Imagery courtesy of Maxar.

Figure 11. Invisible car in the shadow.



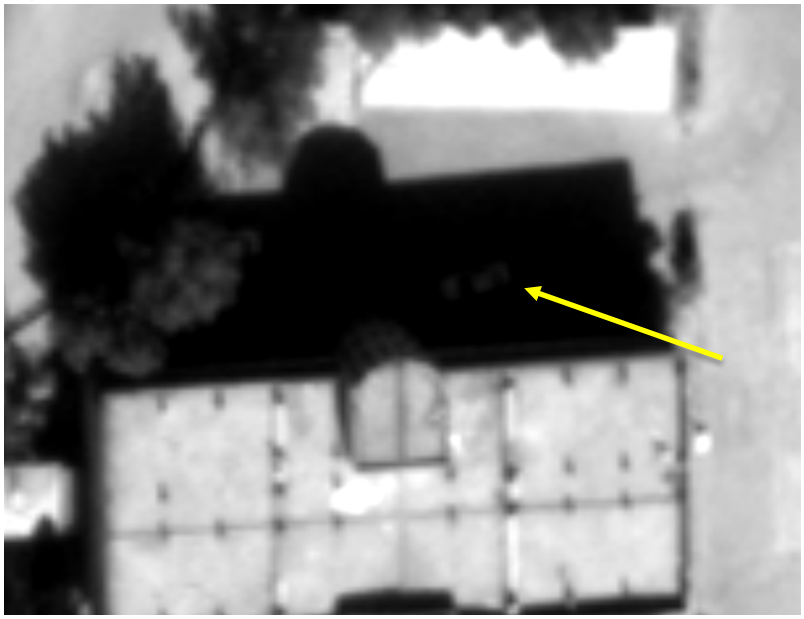
Both left and right images are captured at 100% zoom level. The car in the shadow in the left image is invisible to the human eye. The right image shows the detection from DeepObject. Imagery courtesy of Maxar.

Figure 12. Invisible car at 200% zoom.



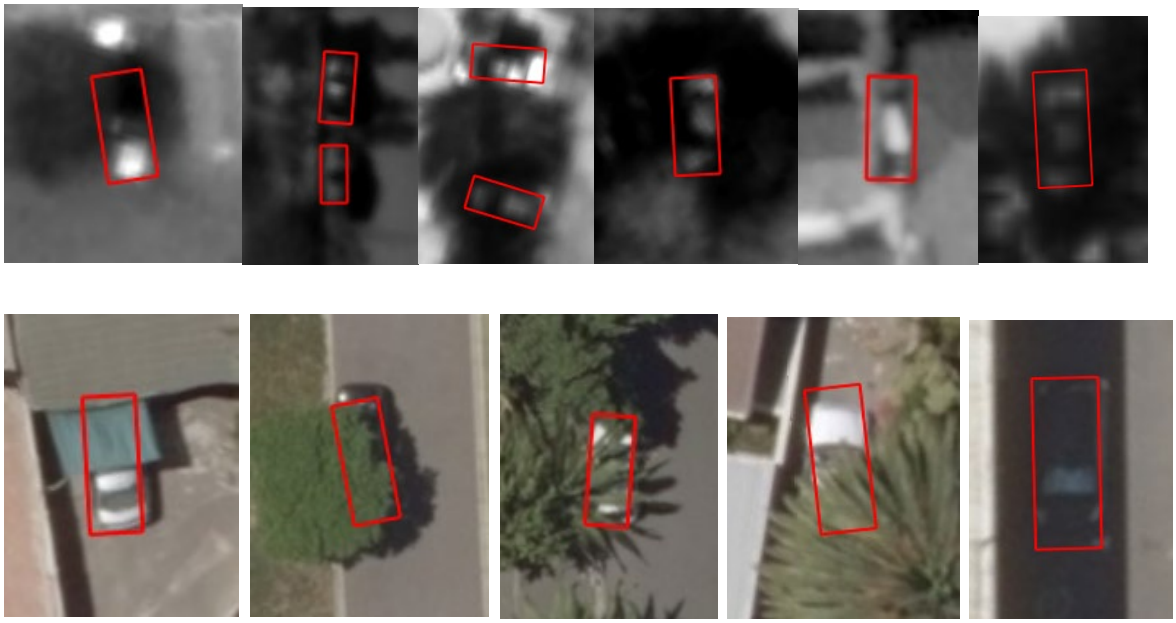
At 200% zoom level, the car in the shadow is still invisible to an IA. Imagery courtesy of Maxar.

Figure 13. Barely visible car at 400% zoom.



At 400% zoom level, the car in the shadow is barely visible to an IA. Imagery courtesy of Maxar.

Figure 14. Easy to miss vehicles for human vision.



Small objects in satellite images are easy to miss by human vision as shown in the upper row, but not for DeepObject. DeepObject can detect partially visible vehicles from aerial images. The pickup truck on the right (second row) is barely recognizable to the human eye. Imagery courtesy of Maxar.

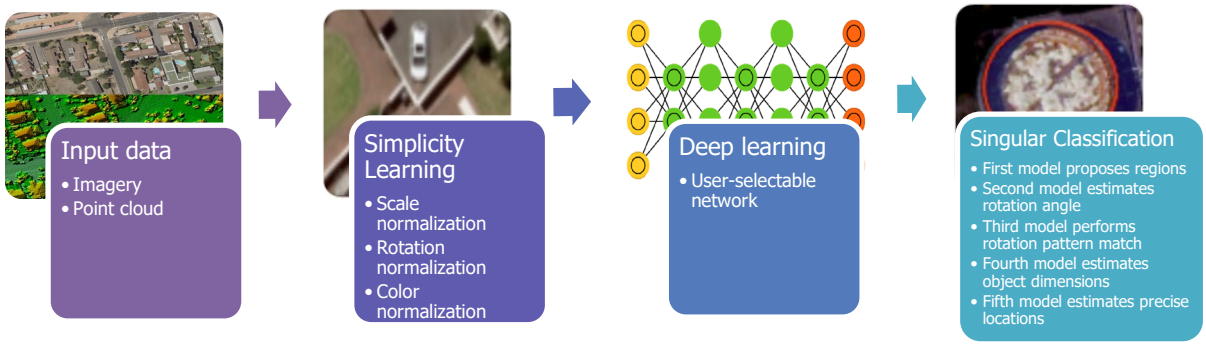
5. Simplicity Learning and Singular Classification

One of the challenges, not just in deep learning but also in many machine learning systems, is the provision of sufficient training samples. Object detection from images is a very complex problem. Let us take the detection of cars from images, as an example. We can assume that there are ten car manufacturers, each producing ten models in ten colors, i.e. 10 x 10 x 10 = 1,000 different cars. This is a manageable

number, but we are working with an image of a car. The image may look very different at different scales, under different lighting conditions, and from different perspectives. We assume that each of these three factors has twenty variations, i.e. 20 x 20 x 20 x 1,000 = 8 million different cases, a much increased challenge. On top of that, we want to detect 1,000 different types of objects such as dogs or cats at the same time, which gives us 8 million x 1,000 = 8 billion different cases; an extremely complex problem.

There are special challenges to detecting objects in geospatial imagery such as, the images are often huge, there may be thousands of objects of interest in the imagery (or very few), the orientations of the objects are unpredictable, satellite image quality (signal to noise ratio) is usually poor, and the objects may contain very few (<100) pixels. Because of these challenges, getting enough positive training samples is difficult, i.e. trying to detect thousands of specific fighter jets. Most geospatial applications require very high positional accuracy of detected objects, which DeepObject has addressed (Figure 15).

Figure 15. DeepObject architecture.



- *Inputs are images or 3-D point clouds.*
- *Simplicity Learning simplifies the learning task by reducing pixel variations using scale normalization, color normalization, and rotation normalization, enhancing the learning object by depressing background 3-D points and enhancing object pixels.*
- *Taking advantage of the most recent/advanced deep CNNs with adaptation for geospatial images and 3-D point clouds, the middle processing deep learning module is user-selectable. DeepObject uses several mega networks such as GoogleNet, VGGNet, and ResNet with adaptation to geospatial imagery. DeepObject has this open architecture such that users can plug in their own networks without any source code change.*
- *Singular Classification uses up to six models to detect objects. The first model, of maximum translation invariance property, filters out regions of no objects, with high speed and extremely high recall. The second model, with minimum translation invariance property, estimates rotation angles of objects. The third model performs RPM to improve precision and reduce false positives. The fourth model achieves sub-pixel positional accuracy by estimating precise locations of objects. The fifth and sixth models estimate object sizes (length and width).*

Simplicity Learning satisfactorily reduces the number of samples required to train the CNN. The conventional approach is data augmentation, i.e. adding large numbers of augmented training samples to cover expected variations in scale, color, and rotation. Data augmentation increases the amount of training samples, which is useful to deal with overfitting. However, many training samples from data augmentation may be degenerate, i.e. they do not contribute much. Scale augmentation and color augmentation do not work like the human brain. Instead, we have worked with scale normalization rather than scale augmentation, color normalization rather than color augmentation, and rotation normalization rather than rotation augmentation. Together, these innovations vastly reduce the number of training samples that the system requires to ensure success.

While this case study focuses on rotation normalization, all three normalization approaches to reducing the number of samples required are important, and are an integral part of Simplicity Learning. Quality training samples are equally important, for example, the 8 million permutations of cars enumerated above. If we reduce to a single scale variation, however, the 8 million are reduced to 400,000. If we reduce rotation variation to 2 rather than 20, then there are only 40,000 permutations. Computer vision and image processing have the algorithms to deal with scale and color variation respectively: we use the advanced photogrammetric algorithms with which we have been working for decades to help with scale normalization, and have developed a set of proprietary algorithms to address color normalization. Color normalization is a key part of *see the almost invisible* in DeepObject.

We found that rotation for some objects in geospatial images is not invariant, i.e. vehicles. Based on this rotation-variant property, we have developed algorithms for more accurate object detection in geospatial images. The most important discovery between the object in the image chip and the object in the training sample is the relationship between rotation angle and "average objectness score," which has a maximum value around zero degrees and the probability computed from the softmax layer (Equation 2) (Figure 16). The agreement between the two graphs is clear, but the objectness score is more reliable for binary classification in our case because the non-vehicle training samples are open-ended. While we are able to acquire enough positive training samples, it is very difficult to find enough negative training samples (non-vehicle training samples). The objectness score is the value $\boldsymbol{e}$ (Equation 2).

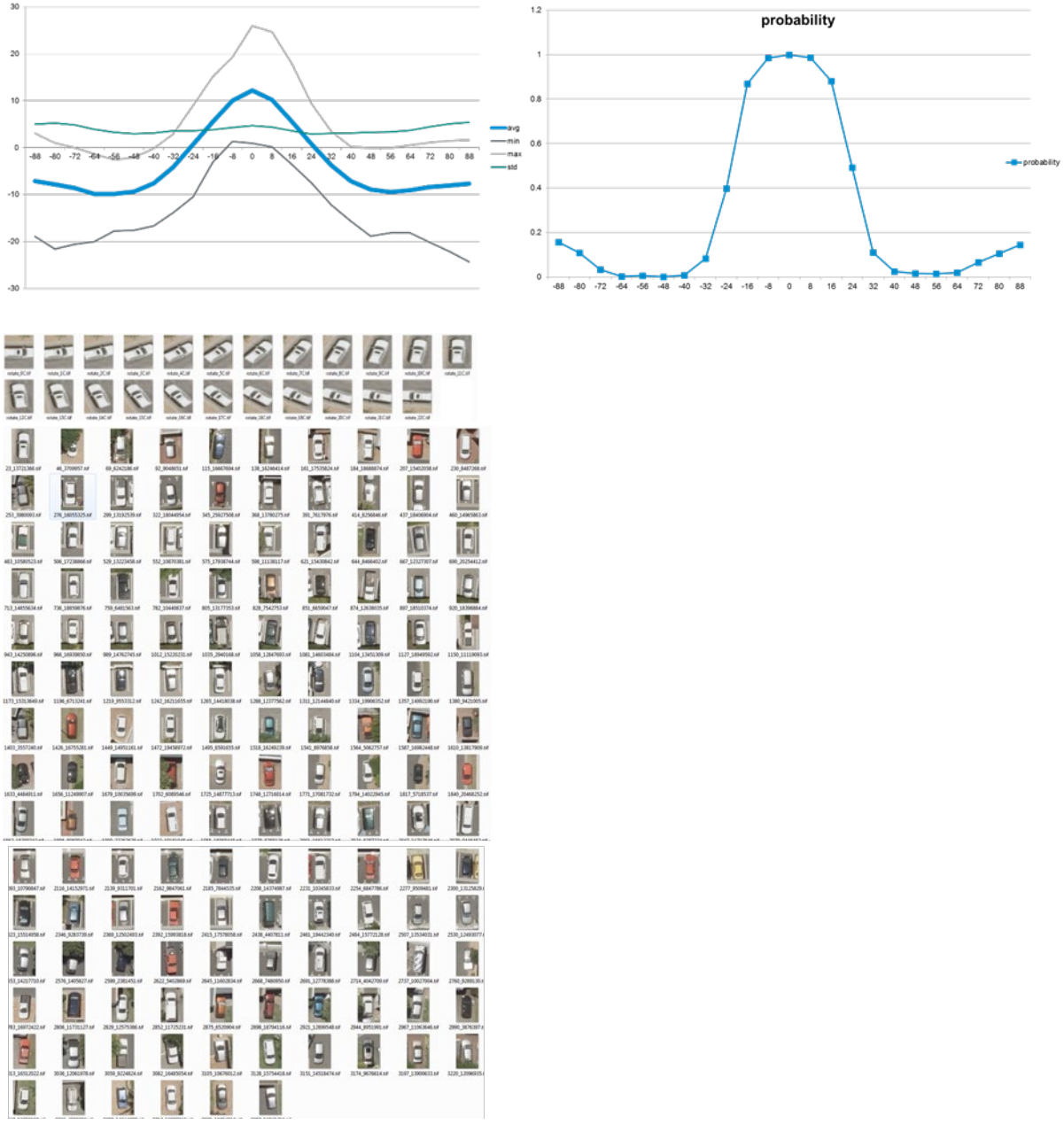
We have also worked on *Singular* Classification, starting from the softmax equation in deep learning:

$$soft\max : \frac{e^{z_i}}{\sum_{j=1}^{c} e^{z_j}} \tag{2}$$

The provision of negative training samples is problematical because it is open-ended. Again, deep learning does not work in exactly the same way as the human brain. We have developed and patented twelve new algorithms, which reduce false positives by up to ten times (US Patent # 10346720).

We carried out an experiment matching our system to a human. Again, Analyst1 from the previous experiment achieved an accuracy of 99.9%, doubtless invoking his experience of parking lots, whereas DeepObject achieved 99.7%. For example, Analyst1 identified three partially occluded cars (in addition to many non-occluded cars) using contextual information and reasoning, whereas DeepObject was equally successful (Figure 17).

Figure 16. Rotation-variant object detection.



It has been proven that convolution is translation-invariant in convolution networks, but we are aware of no study about rotation and found that rotation for at least some objects in geospatial images is not invariant. This property shown in the top-left graph for vehicles: the bold blue "average" line indicates that the average objectness score of 146 cars reaches a maximum when there is zero rotation between the chip and training samples. Based on the above property, we have developed twelve new algorithms to have more accurate detection of objects in geospatial images. The top-right graph indicates the probability computed from the softmax layer. The two graphs show very similar behavior, but the objectness score is more reliable for binary classification in our case because the non-vehicle training samples are open-ended. The large panel to the left shows 146 vehicles, which are training samples that are oriented in the same direction with operator errors up to ~5°. In addition, the small panel to the left shows an image chip where the training samples are compared and rotated from -88° to 88° in 8° increments. All 146 vehicles were recognized as vehicles. Imagery courtesy of Professor Dunn, University of Vermont.

Figure 17. Competition between DeepObject and Analyst1.

Both the Analyst1 and DeepObject identified all three cars partly shrouded by trees (left-hand side). The car in the right image has length of 3.99 meters according to human. The same car has a longer length according to DeepObject. DeepObject uses its fifth model to estimate precise length. The average positional accuracy of DeepObject is 0.68 pixel. Imagery courtesy of Professor Dunn, University of Vermont.

The key issue in the use of deep learning is the provision of quality training samples, which are the new currency in deep learning. In a traditional scenario, using the classes of algorithms that preceded deep neural networks, a software supplier would discover that automatic object detection failed to recognize a car, but an imagery analyst found it. The procedure followed would be to generate a bug report and deploy developers to fix it. With deep learning, the missing car could be automatically collected as a positive training sample and be added to the training sample database. Once the training sample database is updated, the deep learning network is re-trained. Thus, in the scenario sketched here, this potential cost savings can be very significant because there are no software code changes involved.

Not all training samples are created equal; mistakes are more likely to have a greater gradient and a stronger influence on the decision boundary that separates object from "non-object." The quality of training samples is as important as the quantity. As we have noted, many training samples from data augmentation may be degenerate, and mistakes increase quality. Thus, geospatial software that is deep learning based collects user knowledge and experience via the logged training samples along with every mistake that can then turn into an enhancement. DeepObject uses a feature database management system to manage quality training samples. A "Future Intelligent Geospatial Intelligence System" would be an intelligent system that could become smarter by learning from its mistakes. This intelligent system could detect and monitor defense-relevant objects at near-human accuracy and super human speed, which may be the "game changer" in the GEOINT domain allowing a significant reduction in software engineering and enhancement costs.

DeepObject is not only more effective, but also cheaper. The cornerstones of our approach are CNNs combined with Simplicity Learning and Singular Classification. Widespread adoption will substantially benefit thousands of IAs. We can free them up from tedious manual object detection so that they can do more creative work with their brains.

6. Double CNNs

DeepObject uses Double CNNs (Zhang & Hammoud, 2019) to detect objects. Translation invariance is a double-edged sword in CNN. It allows detection of a particular object if we only care about its existence, not its position in an image. CNN with maximum translation invariance has high-speed, but low positional accuracy. CNN with minimum translation invariance has high positional accuracy but low-speed. DeepObject uses Double CNNs to achieve both high-speed and high positional accuracy. The first CNN (first model) is trained with maximum translation invariance property and is very fast with extremely high recall to minimize missing detections. The output from the first model are region proposals for the second CNN, which has high positional accuracy, but low-speed. The second CNN is trained with a minimum translation invariance property. Double CNNs can be faster than single CNN when objects are sparse in overhead geospatial images (Table 2).

Table 2. Pros and cons between Faster R-CNN (Ren et al., 2016) and Double CNNs for object detection from overhead imagery.

	Faster R-CNN	Double CNNs
Detection time	Feature extraction * (1 + RPN / feature extraction)	Feature extraction * (1 + 1.778 * object pixels / total pixels)
Precision		Higher due to double-check using two different CNNs and RPM
Recall	Same	Same
Positional accuracy		Higher due to the second CNN with minimum translation invariance
Number of positive training samples		Fewer due to Simplicity Learning
Training time	Training one big model	Train five smaller models
Bounding box + orientation	Bounding box	Bounding box + orientation

RPN is the additional region proposal networks.

Taking full advantage that objects are very sparse and translation invariance plays an important role on detection speed, our Double CNNs method has the following advantages (Table 2):

1. Can be faster than Faster R-CNN when objects are very sparse, which may be the case in the GEOINT domain.
2. High positional accuracy because the second CNN has minimum translation invariance.
3. Double-check to reduce false positive error rate. The first CNN with maximum translation invariance generates region proposals, as well as, performs double-check since the two CNNs are trained differently and they do not share weights at all.

DeepObject needs to train up to six models instead of just one model as the case for Faster R-CNN. However, the six models for DeepObject are much smaller than the one model for Faster R-CNN. Training time and detection time are directly proportional to model size. When the model size of DeepObject is 1/6 of the model size for Faster R-CNN, the training time for both can be similar. Because of our Simplicity Learning, training time for DeepObject on a desktop computer with two high-end NVIDIA GPUs is around 20 hours on the average.

7. Case studies

There are seven case studies in this section:

1. The first case study documents the accuracy improvement with our **RPM**, which reduces the false positive error rate and increases precision from 89.9% to 99.7%.
2. The second case study shows speed-up (2.89 times), as well as, improved positional accuracy (1 pixel) with our ***Double CNNs***.
3. The third case study uses 3-D point clouds as input to detect 3-D objects with the combination of CNN and 3-D model fitting.
4. The fourth case study deals with small objects in satellite imagery.
5. The fifth case study uses Synthetic Aperture Radar (SAR) images as input to detect ships.
6. The sixth case study uses DeepObject to automatically generate DEM from Digital Surface Model (DSM) as part of our intelligent photogrammetry.
7. The seventh case study automates oil volume estimation from satellite images.

7.1 Airplane detection from satellite imagery

The purpose of this case study is to evaluate the accuracy, both precision and recall, of DeepObject and to verify the false positive error rate reduction of "rotation pattern match" (Zhang, 2017) algorithm in DeepObject as we found in the third

model. We used WorldView images (courtesy of Maxar) from four airports (HKG, LAX, JFK, and LHR) for this case study. For each airport, we have multiple images from different dates with different look angles and GSDs. Training samples are extracted from images that are different from detection images. In other words, no training samples are collected from detection images. We have verified that the RPM reduces false positive error rate and increases precision from 89.9% to 99.7%. In this case study, the average precision is 99.7% and the average recall is 100.0%. Without RPM, the average precision is 89.9%. Our Simplicity Learning contributes to the 100.0% recall.

This case study indicates that DeepObject provides a partial solution to one of the most challenging problems in deep learning, which is the almost unlimited negative training sample requirement. DeepObject matches rotation patterns of only positive training samples. In the GEOINT space, we may be able to collect enough positive training samples, but the negative training samples are anything else and almost unlimited. Many organizations in the deep learning space are trying to collect a huge number of training samples to deal with this challenge, but our case study indicates that DeepObject only needs a small number of training samples.

DeepObject uses its third model to perform RPM (US Patent #10347720). Machine learning, and deep learning in particular, is sensitive to variations in unseen images as documented in "Slight Street Sign Modifications Can Completely Fool Machine Learning Algorithms" (Ackerman, 2017). Our RPM is performed outside of deep learning and acts as a non-machine learning component of DeepObject. The output from deep learning is verified against known rotation patterns. That is the reason why DeepObject can achieve higher precision than not using RPM.

Table 3. Precision and recall of jumbo jet detection.

Airport	Precision (%) with RPM	Recall (%) with RPM	Precision (%) without RPM	Recall (%) without RPM
LAX	100.0	100.0	88.1	100.0
JFK	100.0	100.0	96.3	100.0
LHR	100.0	100.0	94.3	100.0
HKG	100.0	100.0	97.8	100.0

For each of the four airports, we ran DeepObject with and without RPM. RPM is designed to improve precision by reducing false positive error rate. The 100.0% recall is contributed by our Simplicity Learning algorithm.

Table 4. Precision and recall of large and medium airplane detection.

Airport	Precision (%) with RPM	Recall (%) with RPM	Precision (%) without RPM	Recall (%) without RPM
LAX	98.5	100.0	82.1	100.0
JFK	99.2	100.0	77.1	100.0
LHR	100.0	100.0	89.9	100.0
HKG	100.0	100.0	93.6	100.0

For each of the four airports, we ran DeepObject with and without RPM. RPM is designed to improve precision by reducing false positive error rate.

7.2 Fighter jets detection from satellite imagery

In this case study, we detect Su 21 fighter jets. We used ten WorldView images over Russia (three from Kursk, three from Komsomolsk, two from Borisovsky Khotilovo, and two from Petrozavodsk) with GSDs ranging from of 0.345 to 0.554 for collecting training samples (courtesy of Maxar). Our test image is from Domma, Russia with GSD of 0.427 meters. We did not collect any training samples from the test image.

We ran ten iterations of training. Each iteration consists of the following steps:

1. Add false negatives (or missing detections) as positive training samples into the training feature database.
   - For the first iteration, collect a small number of positive samples manually from one of the six training images.
   - For the rest of the iterations, collect false negatives from the previous iteration as positive training samples.
2. Add false positives as negative training samples into the training feature database.
   - For the first iteration, collect a small number of negative training samples from one of the training images.
   - For the rest of the iterations, collect false positives in the output feature database from the previous iteration as negative training samples.
3. Augment training samples from the training feature database with maximum translation invariance for the first model.
4. Augment training samples from training database with minimum translation invariance for the rest of the four models.
5. Train the first model from step 3.
6. Train rest of the four models from step 4.

7. Run detection on eight training images.

After ten iterations of training, we collected a total number of 432 positive training samples (Su fighter jets) (Figure 18). The same jets may be collected more than once when they are in multiple images. We collected 3,869 negative training samples or non-Su 21 fighter jets. Using false positives as negative training samples ensures that they are quality training samples. Similar to a human learning from their mistakes, DeepObject learns from its mistakes by using only false positives and false negatives as training samples (Zhang, 2017a).

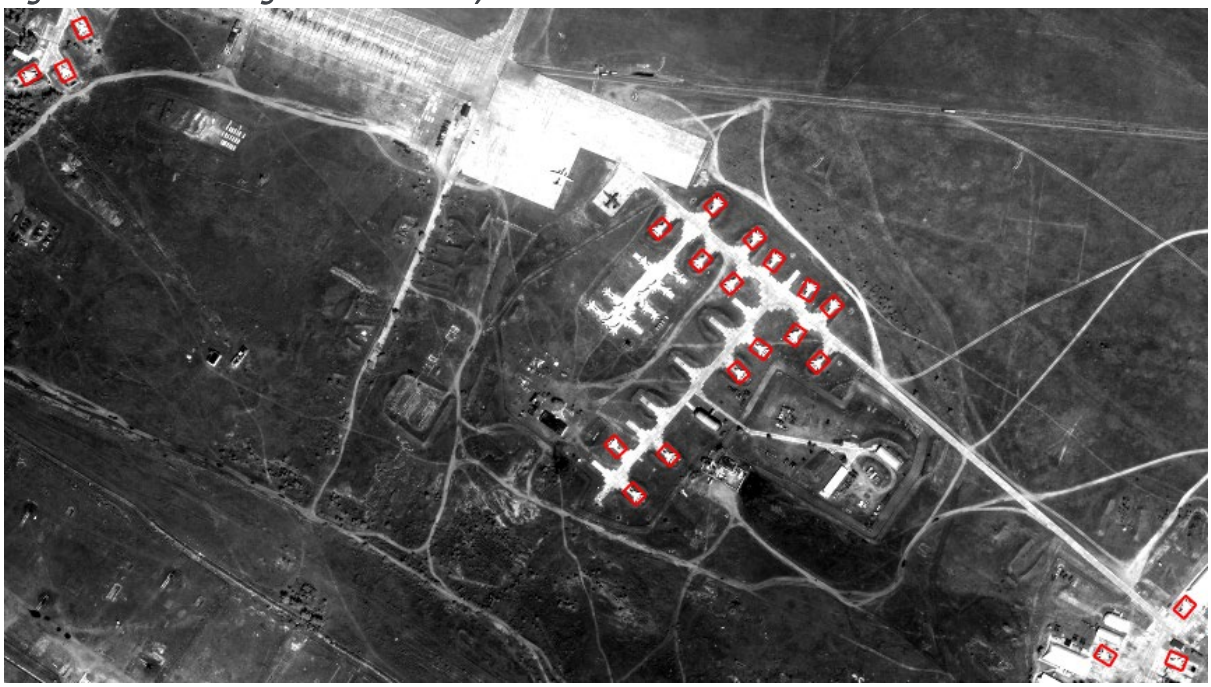
Figure 18. Positive training samples.

A subset of 432 positive training samples (Su 21 fighter jets) from ten images and four locations. The pixel values on Su 21 fighter jets vary significantly from different sun angles and different look angles. The first digit of an image name such as 1 in the 1-15-0.tif is the training feature database index or training image index. The second digit is the feature ID, and the third digit such as 0 is the category ID. Even in the same image, the Su 21 fighter jet has very different colors (pixel values) such as 1-15-0.tif, which has low pixel values vs. 1-3-0.tif, which has high pixel values. The object is always in the center with one orientation. In other words, the positive training samples are rotated such that the rotation angle of each image chip is always close to zero. Imagery courtesy of Maxar.

Our final test image for speed comparison is from Domma, Russia with GSD of 0.427 meter and an image size of 20100 x 37504 or 753,830,400 pixels (Figure 19). It is typical that objects are sparse as in our case; there are only 21 Su fighter jets in an area of 137 km^2. We ran two tests, one using five models jointly (Double CNNs), and another using only Model1 (single CNN). Our Double CNNs completed in 3 minutes and 42 seconds vs. 20 minutes with our single CNN. That is a speed up of 4.4 times. It should be noted that the 4.4 times speed up is in comparison with our single CNN,

not Faster R-CNN. We did not implement Faster R-CNN and have no speed comparison metrics. We used a desktop computer of Intel i7-7820X CPU, 2 NVIDIA GeForce RTX 2080 Ti GPUs, and 32 GB RAM for this case study.

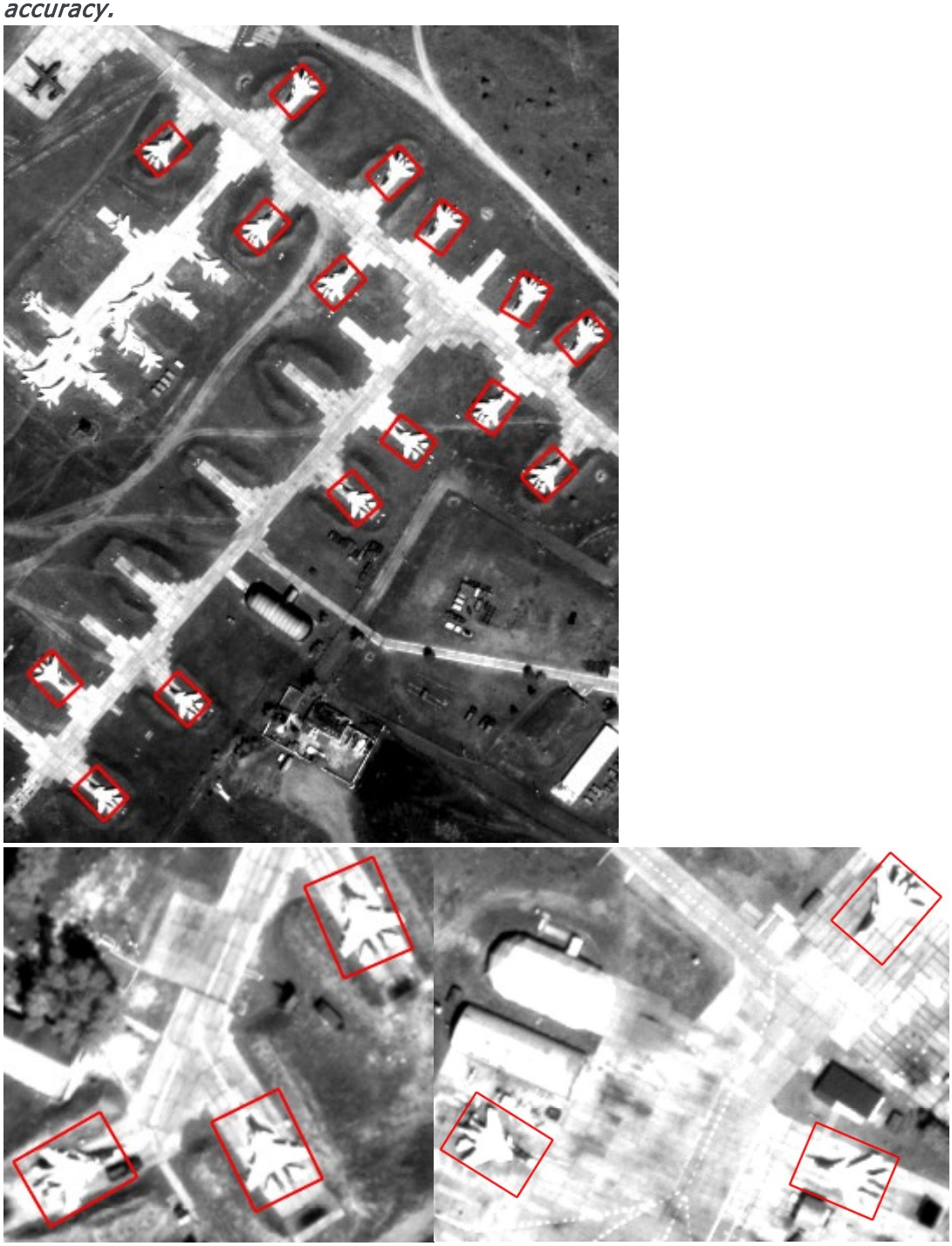
Figure 19. Test image from Domma, Russia with GSD of 0.427 meter.



A total of 21 Su fighter jets were detected (red bounding boxes) in an area of 137 km^2. The detection speed is 3.4 million pixels per second with our Double CNNs and 0.63 million pixels per second with our single CNN. Our Double CNNs achieved a speed up of 4.4 times. It should be noted that the 4.4 times speed up is in comparison with our single CNN, not Faster R-CNN. Image courtesy of Maxar.

Average positional accuracy is about 1.09 pixels or 0.5 meters. Positional accuracy is the distance between the center of a bounding box and its object center. There are two other types of airplanes, which are not detected as Fu 21 fighter jets. The orientations of a bounding box is also the orientation of a Su fighter jet (Figure 20). The average orientation accuracy is about 6 degrees, which is measured by the difference between the bounding box orientation angle and its object orientation angle.

Figure 20. Average 1.09 pixels positional accuracy and average 6 degree orientation accuracy.



The orientation of a bounding box represents the orientation of an object. Both recall and precision are 100.0%. The upper image displays different types of fighter jets that are not detected. Imagery courtesy of Maxar.

Table 5. D2CNN Case Study Metrics.

Metrics Names	Metrics Values
Precision	100.0%
Recall	100.0%
Average positional accuracy	1.09 pixels
Detection speed	3.4 million pixels per second
Positive training samples	432
Average orientation accuracy	6 degree
Hardware	2 NVIDIA DeForce RTX 2080 Ti GPUs

In many automatic object detection cases, it is difficult to collect enough positive training samples, while it is easy to collect enough negative/background training samples. This is because negative/background training samples are collected from false positives during our iterative training. Therefore, the number of negative/background training samples is not one of the main objectives.

In our simple case study, both precision and recall are 100.0%. It should be noted that other complex objects such as camouflaged military vehicles, the recall, and precision are likely much lower than 100.0%. Our average positional accuracy is about 1 pixels. Our detection speed with Double CNNs is 3.4 million pixels per second. We used 432 positive training samples.

7.3 3-D object detection from 3-D point clouds

In the GEOINT space, positional accuracy is as important as precision and recall in many use cases. For example, for 3-D mapping application positional accuracy requires XYZ coordinates of each vertex of an object at pixel level accuracy. Unfortunately, convolutional networks in deep learning are invariant to translation. In other words, the positional accuracy from deep learning object detection is inherently poor.

Combining DeepObject 3-D Model Fitting and dedicating a separate model to estimate precise position, our 3-D DeepObject has the best of both worlds. DeepObject can detect object position (a bounding box) with close to human level accuracy, while 3-D Model Fitting can achieve pixel level positional accuracy. The output (bounding boxes) from DeepObject are the input for 3-D Model Fitting. A bounding box from DeepObject can significantly reduce the search space for 3-D Model Fitting. Our latest test indicates that 3-D DeepObject can achieve much higher positional accuracy than DeepObject or 3-D Model Fitting alone can achieve.

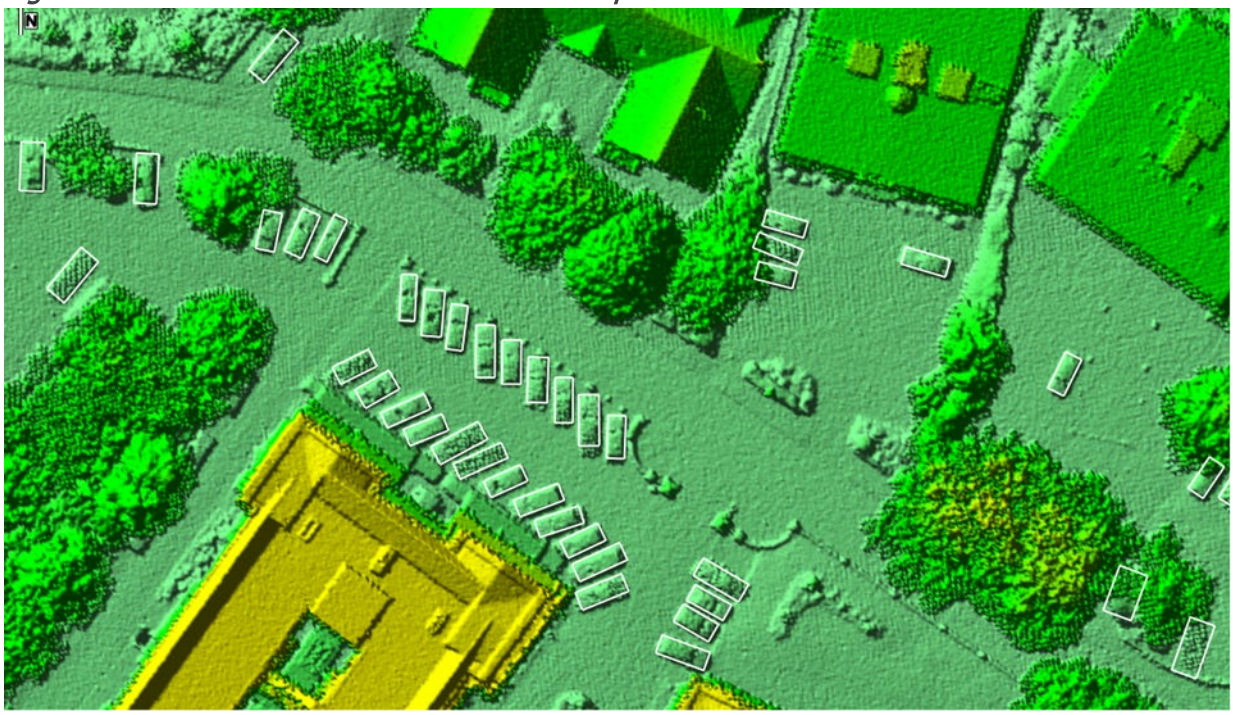
To extract precise XYZ coordinates of each corner of a 3-D object, we use 3-D point clouds as input instead of images. 3-D point clouds can be from LiDAR or from

photogrammetric stereo image matching. 3-D point clouds allow us to perform 3-D Model Fitting after DeepObject bounding box based object detection. Based on our experiments, 3-D point clouds have much less variations than images, which simplify DeepObject training requirements. On the other hand, 3-D point clouds do not provide the same amount of information to identify/distinguish objects than images (Figure 21).

Figure 21. Detection of houses in a sparse LiDAR point cloud.

This material is based on data services provided by the OpenTopography Facility with support from the National Science Foundation under NSF Award Numbers 1226353 and 1225810. The density is 1.4 points per square meter. We have detected more than 44,000 houses in this dataset, of which this figure is an excerpt, with an accuracy around 99%. Note that Automatic Feature Extraction (AFE) cannot extract houses with LiDAR at 1.4 points per square meter, i.e. a point spacing of 0.8 m. It should be noted that the major challenge for house detection is overhanging trees. Due to dry weather in San Diego, there are not many overhanging trees in our dataset. This contributes to high accuracy as well.

Figure 22. Detection of cars in a dense LiDAR point cloud.



The cars have a positional accuracy of less than 1 post spacing, i.e. less than 10 cm in this case. Cars are too small to be detected by our hand-crafted AFE. Further details reported by Zhang (2017). LiDAR data of part of Bournemouth, UK, courtesy of Ordnance Survey.

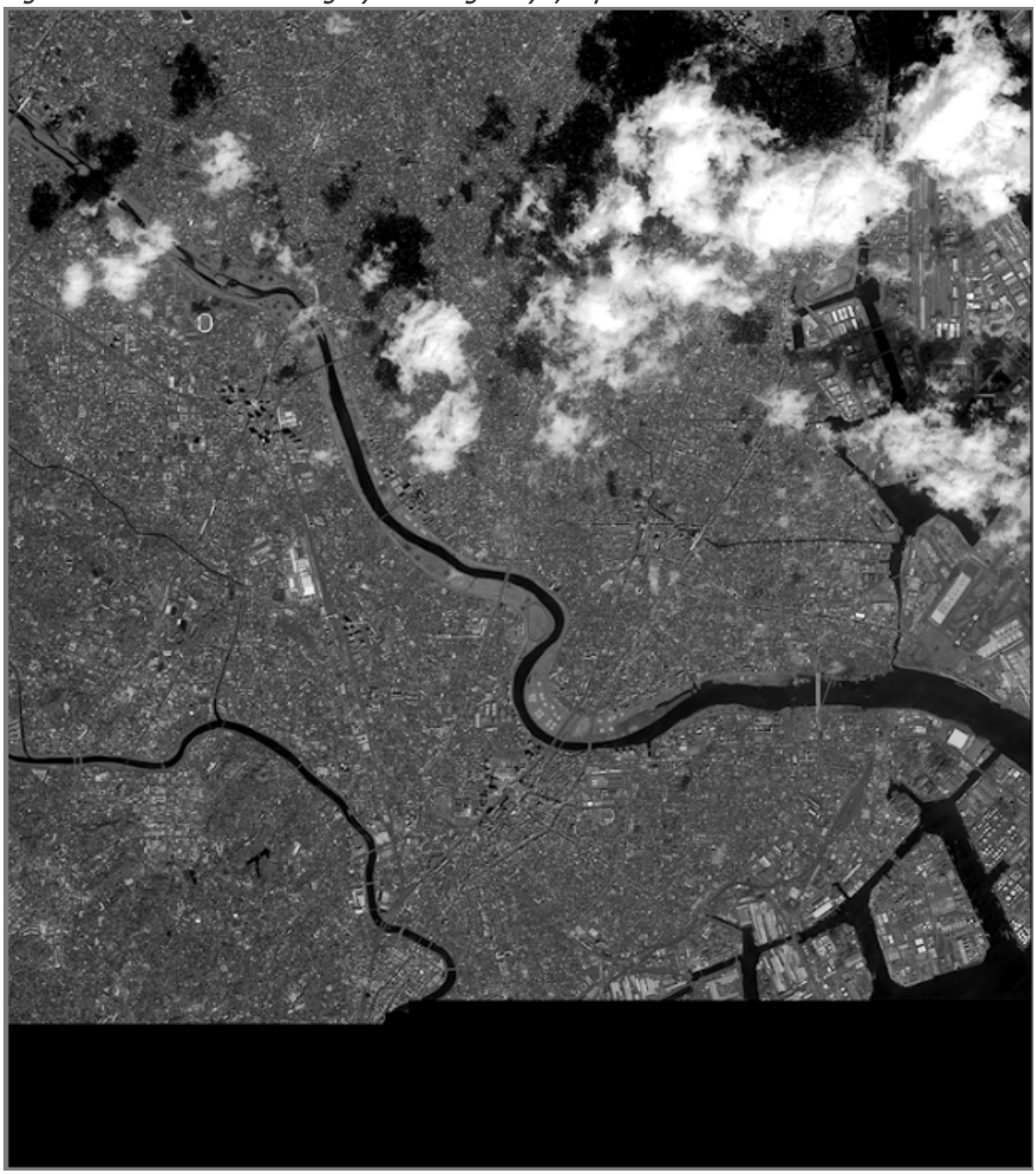
3-D DeepObject has achieved 3-D mapping accuracy based on our latest test (Figure 22). The input is a LiDAR point cloud with a spatial resolution of approximately 100 points/square meter. The output are cars and small trucks. The precision is 96.5%, the recall is 99.0%, and the positional accuracy defined as XYZ coordinates of each corner is less than 1 pixel or 0.10 meters. We used around 1,000 positive training samples and around 2,000 negative training samples. The training took less than 15 hours on a NVIDIA M6000 GPU. 3-D DeepObject has extracted around 6,000 objects in 66 minutes. Among the 66 minutes, 20 minutes were running DeepObject and 46 minutes were running 3-D Model Fitting. While DeepObject is AI, 3-D Model Fitting is hand-crafted AFE.

7.4 Small object detection from satellite imagery

This case study is to detect small objects (vehicles) from satellite imagery. We started with a WorldView-4 image (courtesy of Maxar) with GSD of 0.3 meters, covering the city of Tokyo, Japan (Figure 23). This image has 44514 x 50260 (2.3 billion) pixels. Because of this dataset's large size, we could not estimate for precision and recall metrics. Instead, we used a similar, but smaller WorldView-3 image covering Bandar Abbas, Iran to determine precision and recall (Table 6). DeepObject detected 7,876 vehicles in the Bandar Abbas image with 157 false positives and 236 false negatives (Table 6). Some of the vehicles are very difficult

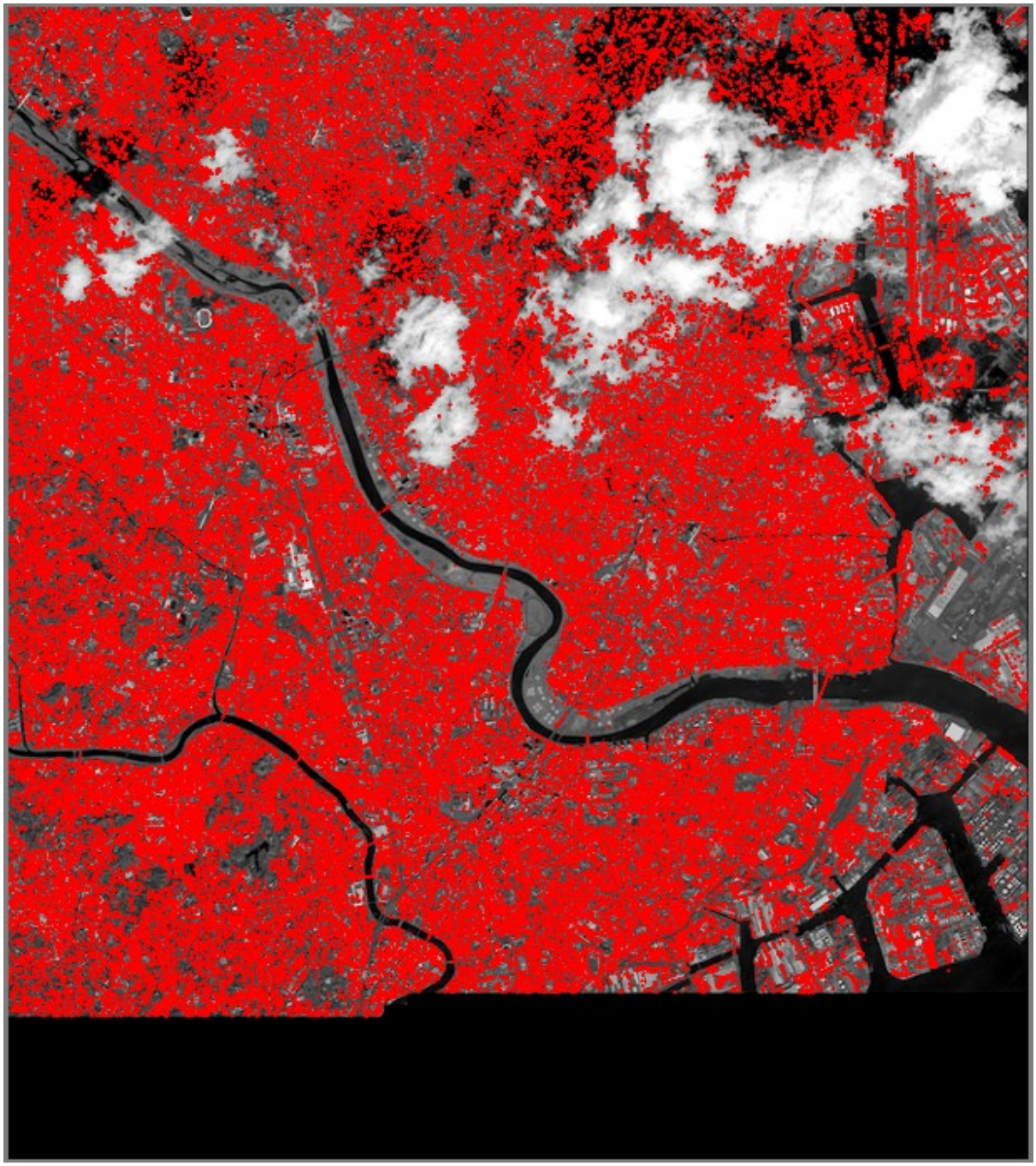
even for humans to detect as discussed in Section 4. DeepObject also detected 216,009 vehicles in 5 hours and 49 minutes, or 10 vehicles per second using a Intel i7-5930X CPU with 4 NVIDIA TITAN X GPUs (Table 6).

Figure 23. WorldView-4 imagery covering Tokyo, Japan.



The total area is 201 square km with GSD of 0.3 meters. Imagery courtesy of Maxar.

Figure 24. Vehicle detection output.



DeepObject detected 216,009 vehicles in 349 minutes or 10 vehicles per second using an Intel i7-5930X CPU with 4 NVIDIA TITAN X GPUs. The average positional accuracy is 0.77 pixels, which is critical to separate vehicles since the gap between adjacent parked vehicles is less than 1 pixel (Figure 25). Imagery courtesy of Maxar.

Figure 25. Parked vehicles.

Gap between adjacent vehicles is less than 1 pixel. This picture is captured at 300% zoom level. Imagery courtesy of Maxar.

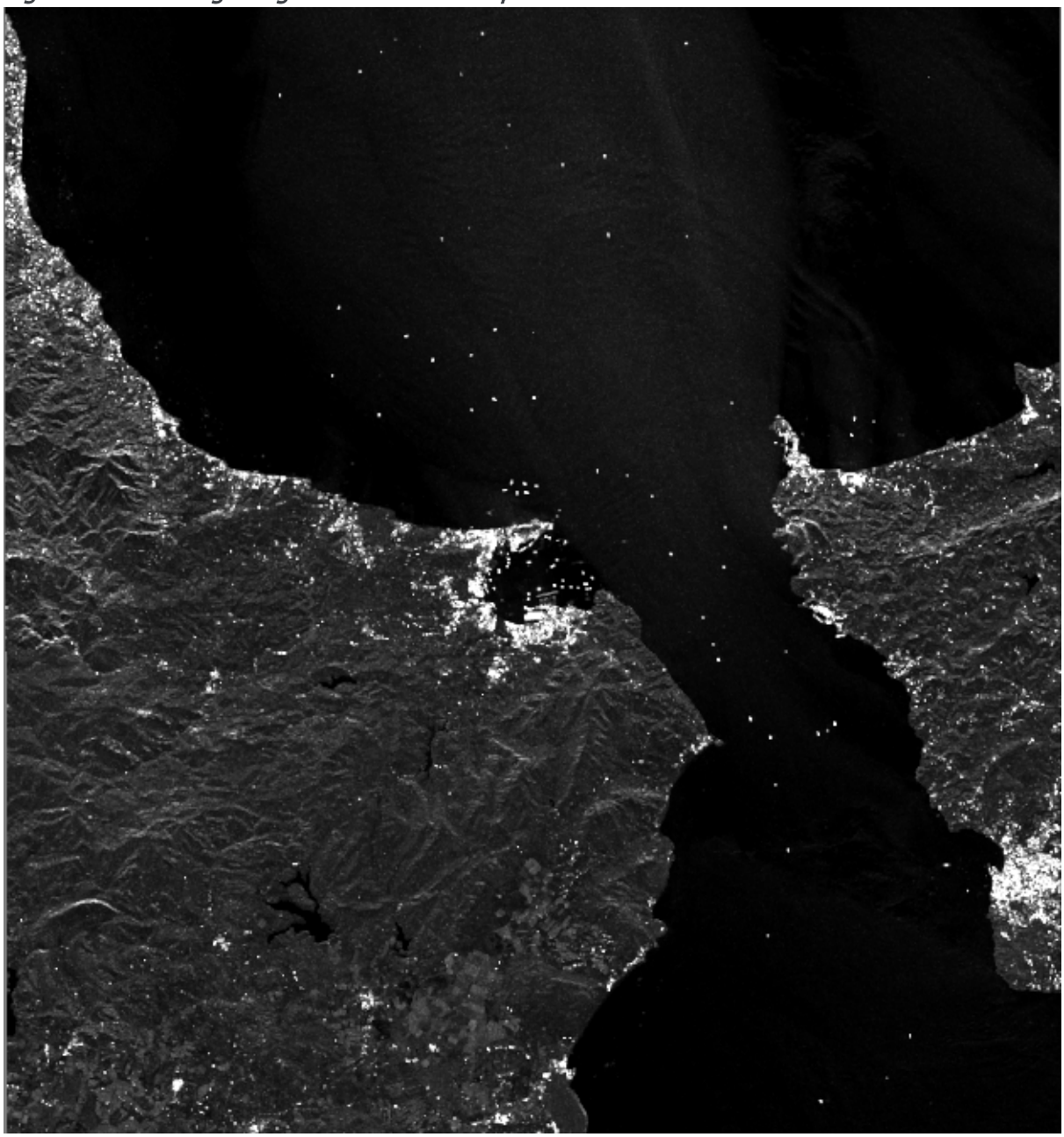
Table 6. Metrics of small objects in satellite imagery detection.

Metrics Names	Metrics Values
Precision	98.0% (from WorldView-3 image)
Recall	97.0% (from WorldView-3 image)
Average positional accuracy	0.8 pixels (from WorldView-3 image)
Number of objects	216,009 (from WorldView-4 image)
Speed	349 minutes (from WorldView-4 image)
Positive training samples	2,475
Average orientation accuracy	6 degree (from WorldView-3 image)
Hardware	4 NVIDIA TITAN X

7.5 Ship detection from SAR imagery

The objective of this case study is to detect ships from SAR images. Each bright spot on the sea is a ship, which is very different from its surrounding water in RADARSAT-2 SAR imagery (courtesy of MDA) (Figure 26). This makes the object detection an easy case, however, there are also bright spots on land. The challenge is to distinguish bright spots on land from on the sea.

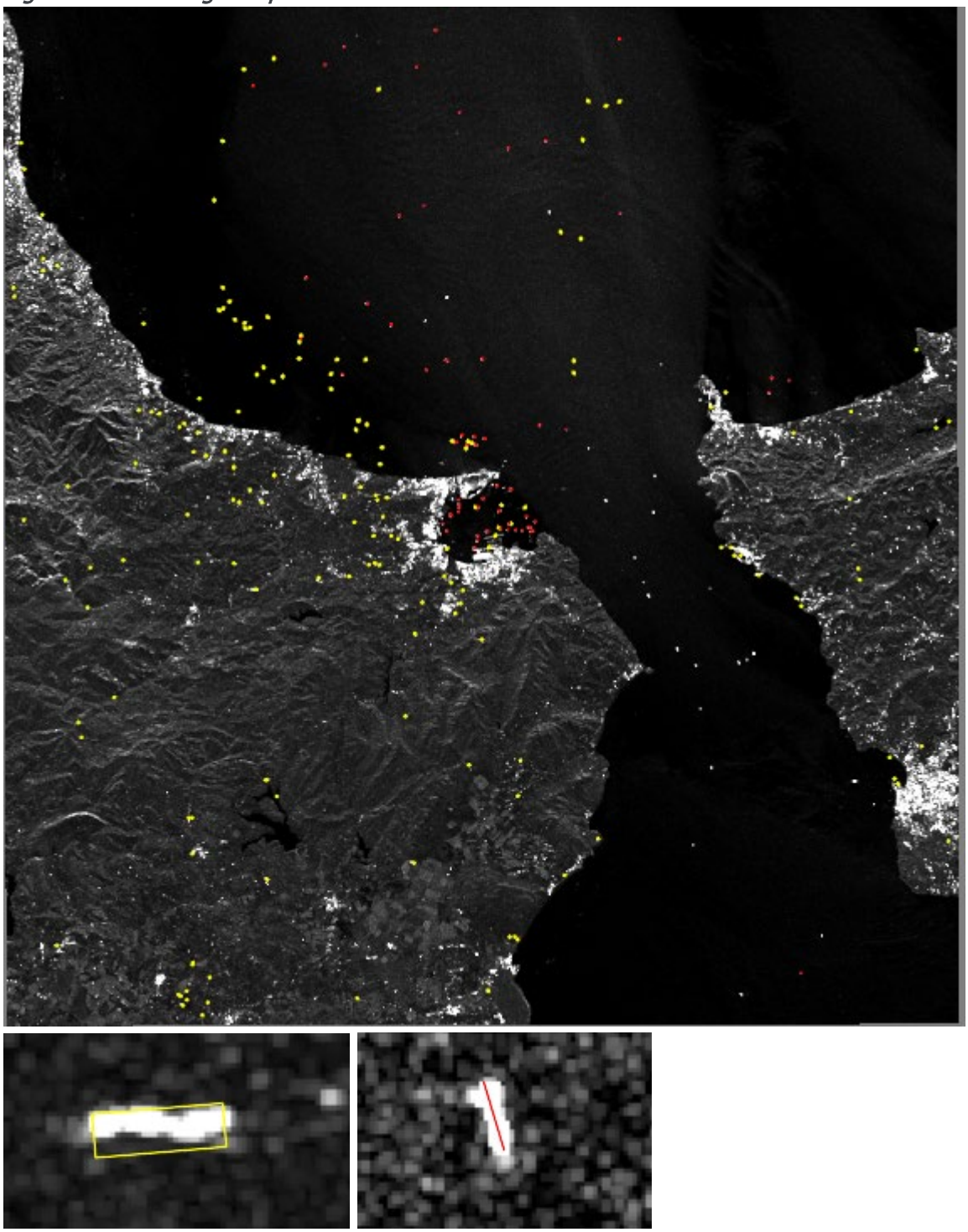
Figure 26. Training image with over 60 ships.



The GSD is 12.5 meters. There are many similar bright spots on land. RADARSAT-2 SAR imagery courtesy of MDA.

We have collected 58 positive training samples and 172 negative training samples (Figure 27).

Figure 27. Training samples.

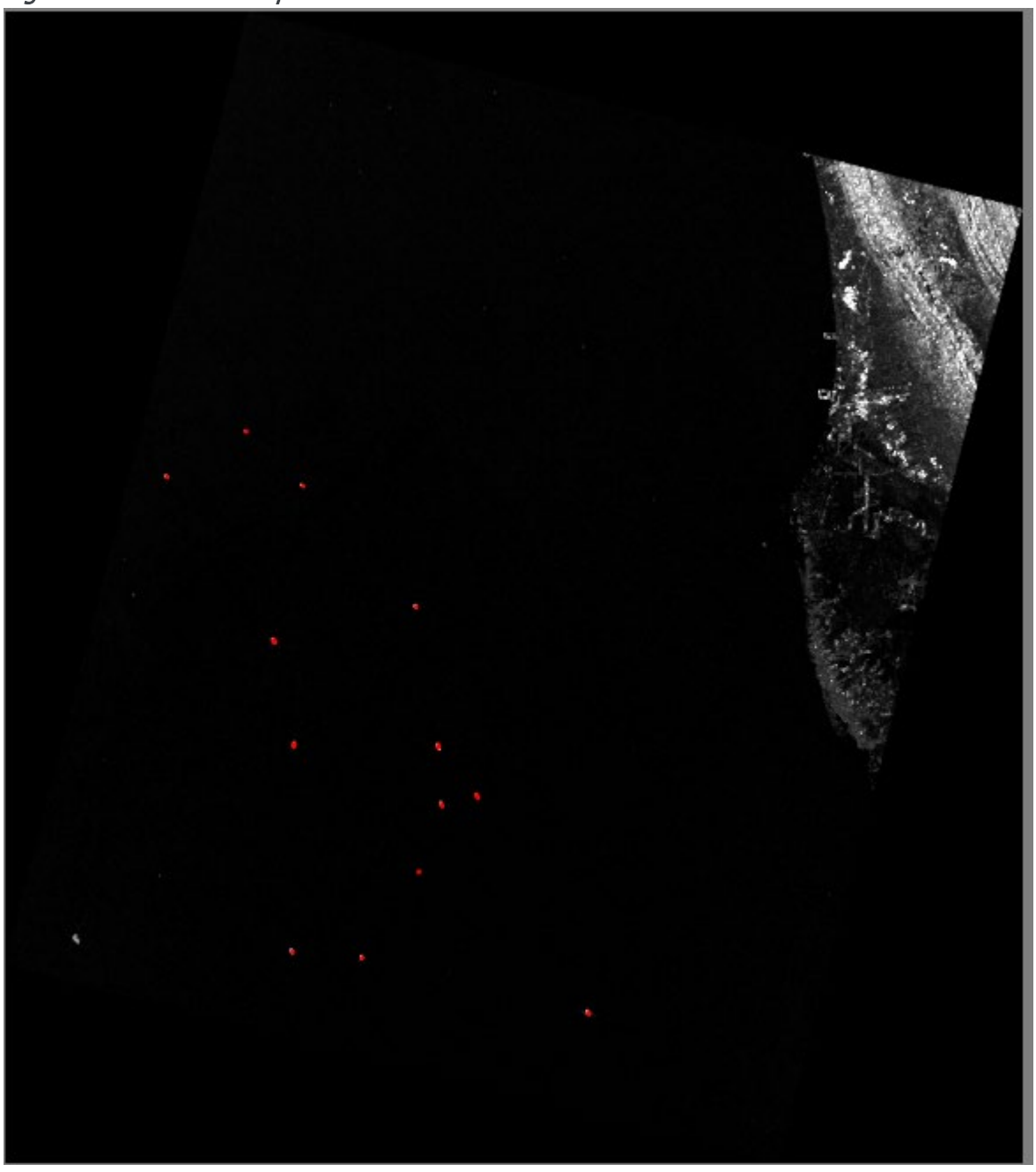


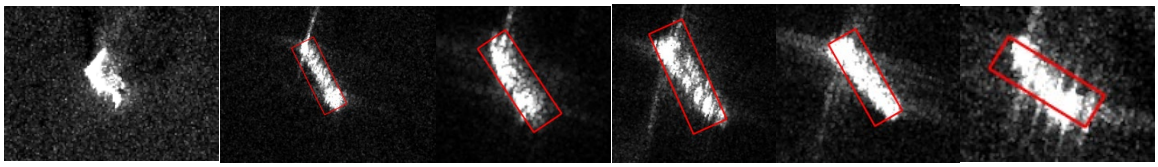
58 positive training samples (red lines) and 172 negative training samples (yellow bounding boxes). We did not use all ships as positive training samples. On the second row, the left image is a negative training sample. On the second row, the right image is a positive training sample. Humans identify the left object as not a ship because it is on land. DeepObject identifies the left object as not a ship

using background pixel values. The background pixel values of the right object are much smaller than that of the left image. RADARSAT-2 SAR imagery courtesy of MDA.

The detection/test image has a GSD of 2.5 meters, which is much smaller than the training image GSD 12.5 meters (Figure 28). DeepObject has detected thirteen ships as shown in red bounding boxes. There is one questionable object at the lower-left corner. As shown on the second row, the object in the left image does not fit into the general shape of a ship. All training samples have a shape similar to the right image in the second row, which is one of the thirteen detected ships. There are a total of 933 million pixels. DeepObject completed the detection in 1 minute or about 15.5 million pixels per second using a 2 NVIDIA RTX 2018 Ti GPUs.

Figure 28. Detected ships.





Thirteen ships were detected. There are no false positives (even on land). There is one questionable object on the lower-left corner that DeepObject did not detect. As shown on the second row, the zoomed in object on the left image shows that it's shape does not fit into a typical shape of a ship like you see in the zoomed in right images. If we count this object as a false negative, the recall is 92.8%. Otherwise, the recall is 100.0%. RADARSAT-2 SAR imagery courtesy of MDA.

Table 7. Metrics of SAR image object detection.

Metrics Names	Metrics Values
Precision	100.0%
Recall	92.8% to 100.0%
Speed	15.5 million pixel per second
Positive training samples	58
Negative training samples	172
Average positional accuracy at GSD 12.5 meters	0.8 pixel
Average rotation angle accuracy	5 degree
Hardware	2 NVIDIA RTX 2080 Ti

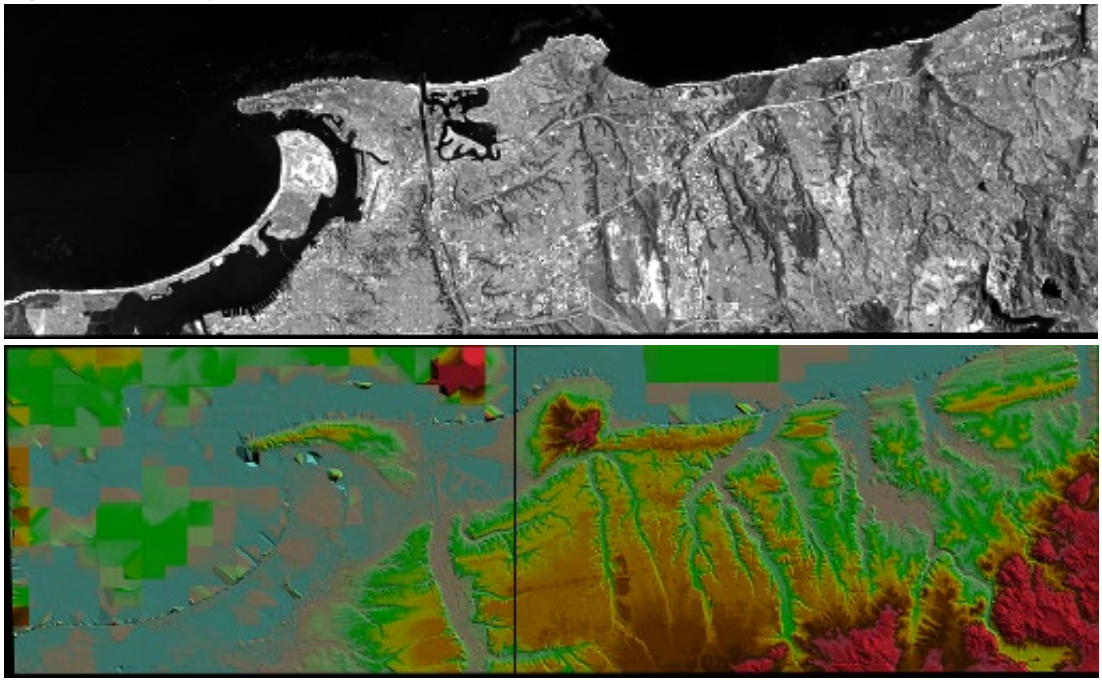
7.6 Intelligent photogrammetry

We used deep learning CNN to modernize the five major automation tasks in digital photogrammetry: tie point extraction and matching, DSM generation, DEM generation, AFE, and image segmentation. In the last four years, we have developed DeepObject, which uses deep learning to detect objects from imagery and 3-D point clouds. One of the innovations in DeepObject is that the positional accuracy is sub-pixel. Sub-pixel positional accuracy allows detected objects such as corners of manufactured structures to be used as tie points for triangulation. For DEM generation, we need to take out manufactured structures, vehicles, vegetation, trees, and forestry from DSM. If we can detect houses, trees, and buildings in the DSM then we can remove them from DSM. Our SOCET GXP AFE works accurately when the input is LiDAR 3-D point clouds. DeepObject is a natural extension to AFE and can detect 2-D objects from imagery with close to human-level accuracy. CNN has shown great promise to classify each pixel into different categories and to segment pixels into homogenous regions in the last few years. Applying deep learning to photogrammetry transforms digital photogrammetry to intelligent photogrammetry.

DEM production is one of the most time consuming tasks in photogrammetry (Zhang, 2020). We have trained five DeepObject models: LargeBuildingModel, BuildingModel, HouseModel, TreeModel, and GroundPointModel from DSM extracted using SOCET GXP Next Generation Automatic Terrain Extraction (NGATE) and

Automatic Spatial Modeler (ASM). Applying these five models sequentially, and in this order to a DSM of 1,141,346,234 posts, 1 meter post spacing, covering San Diego, California, USA (Figure 29), we have achieved RMSE of 0.95 meter (Table 8).

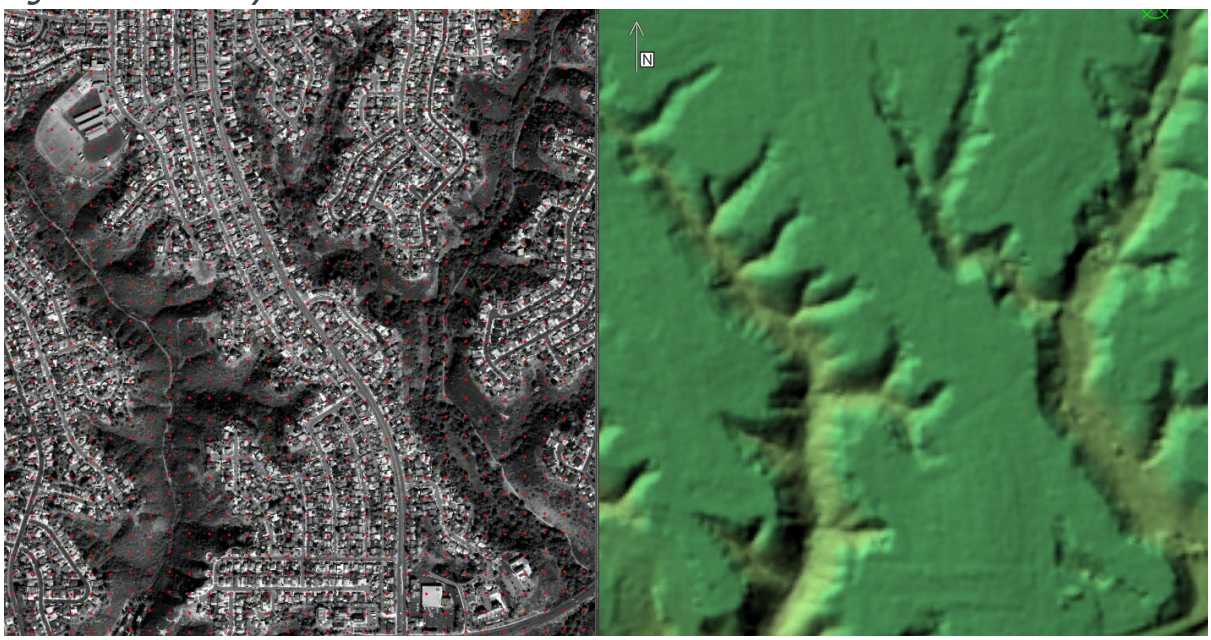
Figure 29. Study area and DSM.



Terrain Shaded Relief (TSR) image of DSM extracted using SOCET GXP ASM with stereo satellite images (WorldView 50cm images, courtesy of Maxar). The upper image is the left image of the stereo pairs. The lower image is the TSR of DSM with 1 meter post spacing. The total area is over 1000 km^2 and the land area is over 750km^2. This is a difficult area for DEM production because the terrain is not flat and there are lots of houses, trees, and buildings. There are also a few densely forested areas in this terrain. Imagery courtesy of Maxar.

To evaluate the accuracy of automatically generated DEM, we compute the elevation differences of 1,007 manually edited ground posts against the elevation from the automatically generated DEM (Figure 30). The root mean square error of the DEM is 0.95 meters. The 1,007 manually edited ground posts has a post spacing of 50 meters. At every 50 meter location, we manually edited a post to the ground. Some posts are on the tops of houses, trees, or buildings. Using 3-D stereo view, we manually edited those posts to the ground based on nearest ground posts and human intelligence. This is a very time consuming process, and that is why automatic DEM generation is the top priority of our intelligent photogrammetry.

Figure 29. Accuracy of DEM.



The left image has 1,007 manually edited ground posts (red dots) with a spacing of 50 meters. The right image is the TSR of DEM. The RMSE computed from the elevation differences of these 1,007 ground posts vs. the DEM is 0.95 meters. Imagery courtesy of Maxar.

To validate our DeepObject based DEM generation vs. other DEM generation algorithms, we used the same DSM, the same 1,007 ground posts, and generated an Accuracy Comparison Table (Table 8). We evaluated a handcrafted, third-party DEM generation software with an RMSE of 2.55 meters. SOCET GXP NGATE and ASM can automatically generate DEM from DSM; their DEM accuracy is 1.86 meters. DeepObject based DEM generation is much more accurate than the third-party software, as well as SOCET GXP NGATE and ASM (Table 8).

Table 8. Accuracy Comparison of automatic DEM generation from stereo satellite images.

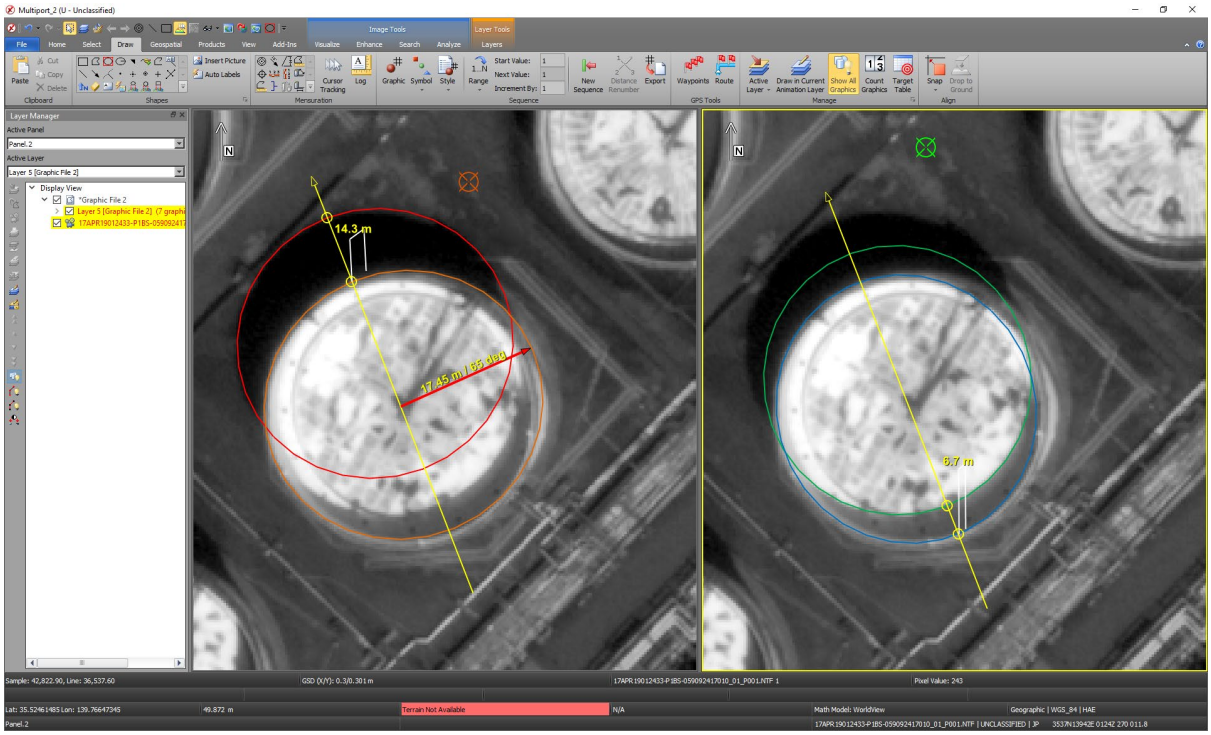
	Third-party DEM	NGATE/ASM DEM	DeepObject DEM
Root Mean Square Error (meter)	2.55	1.86	0.95

DeepObject DEM can significantly reduce the DEM production cost from satellite stereo images, aerial stereo images, Unmanned Aerial Vehicle (UAV) stereo images, sparse LiDAR 3-D point clouds, and dense LiDAR 3-D point clouds. DEM generation from stereo satellite images is more difficult than the other sources.

7.7 Automating oil volume estimation

In this case study, we demonstrate our method to automate the process of measuring storage levels of floating-lid Petroleum, Oil, and Lubricants (POL) tanks. We can manually estimate the floating-lid level using shadows and sun azimuth angle with SOCET GXP (Figure 30). Can we automate the entire manual process with DeepObject and machine learning (shadow segmentation)?

Figure 30. Computing oil volume based on shadow.



In this example, the tank has a radius of 17.45m, is 14.3m tall, and the lid is 6.7m deep. So, the capacity of the tank is $\pi r^2 h$ ~ 13670 cubic meters ~ 2171 barrels. The empty volume due to the lowered lid is ~ 6409 cubic meters ~ 1017 barrels. The tank is currently holding ~ 7270 cubic meters ~ 1154 barrels. Imagery courtesy of Maxar.

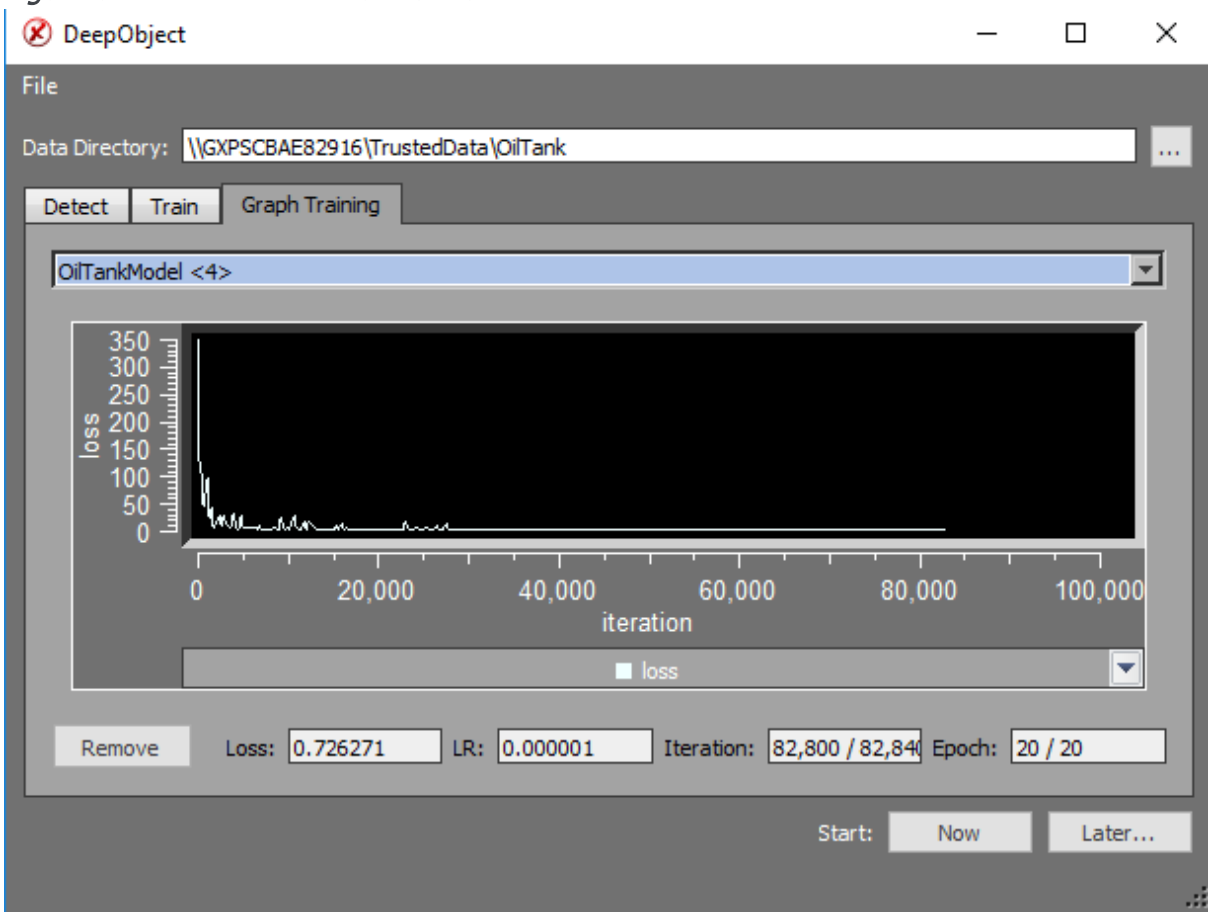
The first step is to detect POL tanks with floating-lid. DeepObject can detect POL tanks with floating-lid from satellite images (Figure 31). The four red circles are the output from DeepObject detection.

Figure 31. Detected POL tanks with floating-lid.

In the right image, there is only one POL tank with floating-lid. The other six POL tanks are not floating-lid. DeepObject can separate POL tanks with or without floating-lid through training samples. Imagery courtesy of Maxar.

Accuracy of the estimated tank inventories is of critical importance to its applications. DeepObject uses its fourth model OilTankModel<4> to precisely estimate the center of tanks to sub-pixel accuracy (Figure 32). The average image line and sample coordinates of the POL tank centers is 0.73 pixels.

Figure 32. Precise centers of POL tanks.



After 82,800 iterations of training, the average loss is 0.73 pixels. In other words, the centers of POL tanks have an average accuracy of 0.73 pixels. This is important to achieve high accuracy of estimated tank inventories.

With the precise locations and sizes/shapes of POL tanks, the next step is to determine the lid level using shadows and sun azimuth angles. We use image segmentation to classify each pixel within each circle into shadow and non-shadow. For binary image segmentation (classifying image pixels into shadow vs. non-shadow), we can achieve 96% precision and 97% recall. This work is in the Research & Development (R&D) phase. With more training, the accuracy can become higher.

With the precise locations and sizes/shapes of POL tanks, the shadow pixel counts within each POL tank and sun azimuth angles from image metadata, SOCET GXP has the photogrammetry capability to estimate the POL inventories.

8. Conclusions

DeepObject has been developed to address the big data challenge that the GEOINT community is facing. DeepObject can help IAs to be much more efficient and productive. IAs can use DeepObject to perform the primitive and time-consuming initial object detection and after that, they only need to verify objects from those detections. This workflow can significantly improve productivity of IAs.

There are two unique characteristics in object detection from geospatial imagery. One is the number of positive training samples. For example, it is hard to find thousands of training samples for specific fighter jets. The other one is positional accuracy, which is critical for geospatial applications. DeepObject can meet both requirements by achieving sub-pixel positional accuracy with only several hundred training samples.

Our case studies indicate that deep learning, specifically DeepObject, can perform object detection with high precision, recall, and speed. These case studies are based on very limited training samples. The biggest advantage of DeepObject is its continual learning capability. DeepObject uses mistakes (false positive and false negative) as "quality training samples" to continuously improve its accuracy. DeepObject uses a database management system to store and manage training samples. DeepObject can be used to transform digital photogrammetry to intelligent photogrammetry. One of the most time consuming tasks in photogrammetry is the DEM generation. Our case study indicates that DeepObject DEM can achieve RMSE of 0.95 meters from stereo satellite images. The more DeepObject is used, the more "quality training samples" can be obtained to further improve the accuracy of this technology, allowing the learning capabilities of DeepObject to be unlimited.

9. Bibliography

Ackerman, E., 2017. Slight Street Sign Modifications Can Completely Fool Machine Learning Algorithms [https://spectrum.ieee.org/cars-that-think/transportation/sensors/slight-street-sign-modifications-can-fool-machine-learning-algorithms]

Goodfellow, I., Y. Bengio and A. Courville, 2016. Deep Learning, The MIT Press, Cambridge, MA and London, UK, 775 pp.

He, K., X. Zhang, S. Ren and J. Sun, 2015. Deep residual learning for image recognition, 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), 26 June–1 July, Las Vegas, Nevada, 770-778. [http://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf]

LeCun, Y. and M'A Ranzato, 2013. Deep learning tutorial, ICML, Atlanta, GA, 16 June [http://www.cs.nyu.edu/~yann/talks/lecun-ranzato-icml2013.pdf]

Singh, A., 2017. Deep learning will radically change the ways we interact with technology, Harvard Business Review, digital article, 30 January. [https://hbr.org/2017/01/deep-learning-will-radically-change-the-ways-we-interact-with-technology]

Ren, S., K. He, R. Grishick, and J. Sun. 2016. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks: arXiv:1506.01497.

Zhang, B., 2016. Big geospatial data + deep learning + high performance computing = geospatial intelligence, presentation S6260, 2016 GPU Technology Conference, San Jose, CA, 4-7 April. [http://on-demand.gputechconf.com/gtc/2016/presentation/s6260-bingcai-zhang-big-geospatial-data-deep-learning-high-performance-computing.pdf]

Zhang, B., 2017a. 3D DeepObject for precision 3D mapping, featured presentation S7149, 2017 GPU Technology Conference, San Jose, CA, 8-11 May. [https://on-demand.gputechconf.com/gtc/2017/presentation/s7149-bingcai_zhang-deepobject-for-precision-printing.pdf]

Zhang, B. 2017b. Rotation variant object detection in deep learning, Nov. 8. US Patent #10347720 [https://patents.google.com/patent/US10346720B2/en]

Zhang, B. and R. Hammoud, 2019. Fast and accurate target detection in overhead imagery using double convolution neural networks, invited presentation, SPIE Defence + Commercial Sensing, Baltimore Convention Center, Baltimore, Maryland, 14-18 April. [https://www.spiedigitallibrary.org/conference-proceedings-of-spie/10988/109880O/Fast-and-accurate-target-detection-in-overhead-imagery-using-double/10.1117/12.2519795.full?SSO=1]

Zhang, B. 2020. Intelligent photogrammetry for terrain production. BAE Systems technical report approved for public release.

BAE Systems, Inc. is the U.S. subsidiary of BAE Systems plc, an international defense, security and aerospace company which delivers a full range of products and services for air, land, and naval forces, as well as advanced electronics, security, information technology solutions, and customer support services.

Enabling development of the most advanced geospatial intelligence, BAE Systems GXP™ software enables rapid discovery, exploitation, and dissemination of mission-critical geospatial data. From key military, security, and incident response operations, to a variety of commercial development and research initiatives, GXP provides a comprehensive suite of solutions to inform effective decision-making and ensure a safer world.