Smart Autopilot Drone System for Surface Surveillance and Anomaly Detection via Customizable Deep Neural Network

Conference Paper · January 2020

DOI: 10.2523/IPTC-20111-MS

4 authors, including:

Kui Liu

35 PUBLICATIONS **1,146** CITATIONS

SEE PROFILE

David Castineira

Hess

42 PUBLICATIONS **454** CITATIONS

SEE PROFILE

Please fill in the name of the event you are preparing this manuscript for.	2020 International Petroleum Technology Conference
Please fill in your 5-digit IPTC manuscript number.	20IPTC-20111-MS
Please fill in your manuscript title.	Smart Autopilot Drone System for Surface Surveillance and Anomaly Detection via Customizable Deep Neural Network

Please fill in your author name(s) and company affiliation.

Given Name	Surname	Company
Xiang	Zhai	Quantum Reservoir Impact
Kui	Liu	Quantum Reservoir Impact
William	Nash	Quantum Reservoir Impact
David	Castineira	Quantum Reservoir Impact

Abstract

Copter-based unmanned aerial vehicle (drone) systems are being utilized for surveillance, inspection and security purposes for well sites, gathering centers, pipelines, refineries, and other surface facilities. However, most of the practices largely rely on humans, including drone operation, data transfer, image analysis, etc. In this paper, we present a comprehensive, cloud-enabled, human-free autopilot drone system and its application in field surveillance and anomaly detection powered by customizable deep neural network and computer vision models.

The proposed system consists of customized quadcopter drones equipped with high-definition cameras, thermal imaging and gas sensing devices, autopiloted by cloud-connected onboard computers. A series of advanced algorithms are developed and deployed onboard and over the Cloud for processing and diagnosing the image/thermal/gas sensing data collected by the drones in real-time or near real-time, including accurate 2D geospatial aerial mapping, anomaly detection and classification for events like oil leak, gas leak, facility failure, human activities, etc. Object detection deep learning models are customized and parallelized for low-profile multi-core single board computers.

In our case study, a pre-configured drone flew along the same path twice at a 6-month gap. A robust, iterative image registering algorithm is developed to precisely align and overlay images taken at different days at the same or similar GPS locations, even with significant changes to the environment due to season shift, human activities, camera angles or height variations. Local changes are filtered and selected based on their sizes and magnitudes in the residual images by subtracting pairs of perfectly overlaid scenes. Pre-trained Residual Convolutional Neural network (He et al. 2015) is rapidly re-trained to further classify the type of changes using the techniques of transfer learning and data augmentation. An ROC of 99% was achieved in the multi-task binary classification, wherein the detected changes are divided into positive anomalies (such as oil/gas leak, facility failures, unauthored human activities) and negative

(natural/insignificant) signals. Comparing against a support vector machine baseline with a ROC=92%, the ResNet model demonstrates significant, more promising detection accuracy at a faster training time.

This innovative integrated platform is presented that combines physical drone, onboard imaging/sensing devices, cloud connectivity, onboard and back-end control system, deep learning and computer vision architecture for situational awareness of oil & gas fields and the mining industry. It achieves full automation of mass surveillance, data acquisition and storage, diagnostics and asset situational understanding. The system architecture, especially the onboard and cloud computation engines, can be readily transferred and applied to other common drone platforms.

Introduction

The surface facilities and surface environment of oil & gas assets, including wellheads, pumps, liquid handling and storage facilities, pipelines, working conditions, human activities, and so on, require periodic surveillance, assessment, and maintenance. Historically, the assessment was done largely via manual human visual inspection. The development of Internet-of-Thing (IoT) and cloud infrastructure enables smart sensors that have been widely utilized nowadays to perform passive real-time or near real-time in-situ or remote monitor and sensing to the surface conditions. However, despite the availability of many specialized sensors and cameras, there is still in lack of general methods that can be used perform multi-task surveillance and inspection work, especially for the tasks that require human experience and judgment. Currently, a phenomenal broadening usage of civilian unmanned aerial vehicles (UAV, or drone hereafter) and industrial drones are applied to various industries, including agriculture, law reinforcement, environment, construction, etc. (Wang et al. 2014, Pajares 2015, Kochetkova 2018, Gómez & Green 2017). Dependent on the detailed applications, fixed-wing drones can perform top-view long-range large-area survey, and the rotary-wing drones (copter-based drones) can be used to conduct detailed multi-angle inspection with almost zero spatial requirements for taking-off and landing. Compared to fixed cameras/sensors, which gives direct high resolution measurement but with limited coverage, and satellite remote sensing service that has wide coverage but lacks details or abundancy in sensing methods, one of the greatest advantages of drone is its flexibility, i.e., drones can be easily expanded and reused for different scenarios.

Until today, most of the drone-based surveillance and sensing is largely performed by human operators. Human operators (pilots) are required to be present on-site to conduct the flight, collect and transport the data, and manually examine the data. For example, to conduct pipeline surveillance with a drone, most of the time a human is still required to manually scan the video footage while the drone is flying or after the flight. The drones are primarily being used as a “flying camera” or “flying data acquisition device,” instead of a smart autonomous robot system. The reasons for this, besides some government regulatory requirements such as line-of-sight flight, include the difficulties of processing the data collected by the drone. Such practices do not utilize the drone in the most efficient way, especially for those routine surveillance workloads, which require repeatability at economical costs. Moreover, one of the major points of using a drone is that drones can provide easy flexible remote sensing. When the surface conditions or accessibility to the terrain are challenging, such as in remote areas, mountain and hills, desert, extreme temperatures, or other harsh environments, it eliminates many of the advantages of utilizing drones by operating the drone manually.

All these situations call for an automated, economic, wide-coverage, reusable and expandable surveillance system. Thus, we propose to solve this problem through one or multiple drone systems equipped with autopilot functionalities, multiple sensors, and connected to the Cloud. We used rotary-wing copter-based drones, but most of the discussion applies to other drone designs. Not only the flight of the drone is autopiloted, but also the data acquisition and data analytics are automated. The key challenge is the automation of data analysis, including algorithms, system integration, and deployment.

One of the major tasks that this paper is focusing on is routine anomaly detection to the surface environment of oil & gas fields and mining fields, such as unauthorized human activities, safety, damaged facilities, misplaced tools, materials & supplies, work activities, oil & gas leaks, etc. Similar tasks that are not explicitly shown in this paper but can potentially benefit from the same workflow include environmental monitoring, flare detection, pipeline tracking/surveillance, etc. Although most of those anomalies have very distinct visual evidence, historically the detection was performed manually because it requires human judgment with their experience and sometimes even intuition. These cognitive tasks were believed to be very difficult to solve using machines. In the recent decade, however, major breakthroughs are achieved in the computer vision and artificial intelligence (AI) community so that many of the individual cognitive tasks such as image recognition and machine translation can now be solved with human-level accuracy with deep learning (Deng et al. 2009, Simonyan & Zisserman 2014, Szegedy et al. 2017, He et al. 2015, Razavian et al. 2014, LeCun, Bengio & Hinton 2015, Ma et al. 2018). We will show that with carefully designed software systems, the visual anomaly detection tasks, ranging from data acquisition to final report, can be largely automated with minimal human intervention, using AI-powered algorithms.

Another major bottleneck of the automation of the workflow is the limitation of computational power. Many of the advanced data analysis requires high performance computing capability, like graphics processing unit (GPU), especially on the onboard device of drones. Civilian or light industrial drones usually cannot carry heavy powerful computing devices because most of the power is consumed providing thrust to balance its weight. Recent developments in compact computing technology has partially made it possible to conduct some power-demanding computation on some relatively lightweight small devices. Still, it is critical to smartly design and distribute the computation tasks across onboard devices (i.e., edge computing) and the Cloud. Integration of hardware design and cloud infrastructure design, sometimes including front-end user interface, and communication network (WIFI, LTE), are necessary to deliver a fast, scalable and reliable automated service.

This paper is organized as follows. First, the paper introduces the infrastructure configuration of the autopilot drone system, including the physical drone itself, as well as various onboard and cloud components and their integration. Then the automation of data pipeline with the system is discussed. The automated cognitive anomaly detection workflow is shown in detail, including change detection and change identification, using the latest computer vision and AI technologies, and how the algorithms can be deployed to the Cloud and the onboard device in the production environment. We demonstrate our methodology and workflow with a real case study for a remote field in Latin America. We then discuss our effort of deploying a deep learning-based object detection algorithm on the onboard device for true real-time edge computing and cognitive analysis. In the final discussion, we show that it is possible to conduct more profound data analysis by combining sensors of different kinds.

Methods

The proposed smart autopilot drone system is supposed to perform surface surveillance and anomaly detection. The key components of the proposed cloud and high-performance computation system consist of autopilot drone system (Figure 1), data acquisition and data transportation automation, cognitive data analysis pipeline and its cloud and edge deployment.

The pipeline of cognitive data analysis, shown in Figure 2, contains automatic change detection and change identification. For change detection, the advanced computer vision algorithm is applied to detect the changes on the ground. Any possible changes such as the human, vehicles, new constructed buildings and stocks could be detected. Iterative registration is applied to align the images taken months apart. For change identification, customizable deep neural networks are utilized to further classify the categories of the objects of interest from the previously detected candidates. Data augmentation and neural network transfer learning (Pan et al. 2010, Yosinski et al. 2014) is applied to deal with the problem of training data insufficiency.

Details of the above key modules will be further described in the following sections.

Autopilot Drone System.

The high-level architecture of the entire drone system is shown in Figure 1. The drone system contains a physical drone for conducting the flight and data acquisition tasks, cloud infrastructures for data storage and data process, a web-based graphic user interface that users can access on a PC or mobile device to interact with the physical drone via the Cloud, as well as examine the data analysis results.

The physical drone system contains a real copter-based drone, a single-board computer and sensors, and cameras. The copter-based drone, with an industrial-standard onboard flight controller, can be used to conduct manual flight via remote control or can be controlled by other devices via serial communication. It is possible to use some developer versions of commercial drones, such as DJI Matrics series, or use custom-built copter drones made with main-stream expandable flight controllers such as PixHawk.

The single-board computer (SBC) is mounted on the drone. This onboard device is the most critical component of the physical drone. It serves as a central brain of the system and the medium to connect the drone to the Cloud. In our experiment, we found it is necessary to have a decently powerful computer that can run a fully functional Linux operation system for this role, instead of using combinations of microprocessors. In particular, the onboard SBC being utilized here runs with a 4-core ARM Cortex 1.2GHz CPU, 1GB LPDDR2 900MHz RAM, 2.4GHz 802.11n wireless adapter and a variety of expandable interfaces like general-purpose IO pins (GPIO), USB, serial UART ports, I2C, SPI ports, camera serial interface, etc. The onboard SBC is programmed to send high-level flight commands to the drone via serial communication, such as way-points (i.e., flight to destinated coordinates), hot-points (i.e., flight around a coordinate at certain radius) and other missions. In the meantime, the SBC reads, records and analyzes drone flight status such as battery level, GPS coordinates, speed, acceleration, direction, compass reading, inclination, barometer, altitude, and other information while the drone is flying or standing-by. Most data acquisition sensors, such as the high definition (HD) camera, the thermal camera, and the gas sensors are directly connected and communicated with the SBC instead of the drone itself. This allows the SBC to control the data acquisition in a more flexible way, such as adjusting the data

sampling rate based on the flight mission. The SBC is also connected to the cloud infrastructure via 3G/4G LTE wireless, or Wi-Fi if presented. In addition to the built-in failsafe configurations of the drone flight controller (such as return the drone to home or land the drone in critical conditions), the SBC is also programmed to be able to override the cloud command and terminate the ongoing mission to return to the "home" position, directly land, or switch to a new mission, such as critical low battery life, on reception of a high-priority command through the Cloud.

The GPS-based autopilot functionality is implemented on the SBC. In a basic way-point mission (i.e., fly to a GPS coordinate in straight line), a PID-based control program deployed on the SBC automatically adjusts the thrusts of the propellers of the drones based on its target GPS coordinates, current GPS coordinates, and velocity, to minimize the time and overshoot for the drone to reach the destination. Similarly, hot-point missions (fly around an object at a certain radius), hover, take-off, and landing are implemented and deployed to the SBC.

The optical camera mounted on the UAV is based on a 16mm wide-angle lens. Inevitably, image distortion must be calibrated in preprocessing. The resolution of the airborne frame is 6000×4000 and the field of view is 60 degrees. A thermal infrared camera is mounted beside the optical camera on the same baseline. The thermal camera has a much lower resolution as 80x60, yet still deliver a spatial resolution about 40 centimeters when the drone flies at 15 meters above the ground.

The cloud infrastructure contains primarily two major components: storage and computation. We have experimented with Amazon AWS and Google Cloud and validated that both cloud services provide sufficient capabilities to handle the drone data flow. The cloud storage contains a file system that is used to save all the data collected by the drone and the results processed by the drone, as well as a non-relational database for metadata. The computation component contains several microservices that conduct different image process tasks, such as image registration (Szeliski 2006), change detection and change identification via deep neural network (see below for more details) and are triggered automatically once new data is uploaded to the cloud storage. Two-way communication between the SBC and the Cloud is maintained all the time as long as the SBC has an internet connection.

A PC/mobile-friendly web-based graphical user interface (Figure 3) is built for user to communicate with the Cloud and the drone. Users can plan flight missions and launch the drone from any remote location with internet access. Using Google Maps' API, users can easily plan, view, modify, or monitor missions laid on top of Google Maps, using either map or satellite view as the drone is in flight. Users can also view the data collected by the drone and results generated by the cloud computation engines.

Although the drone is cloud-enabled via the SBC, it can still be controlled by the manual remote control directly. In fact, in our practice, the drone is configured so that the remote control always has a higher priority in controlling the drone over the SBC. This safety best practice is especially helpful when experimenting new functionalities. In real field tests, this adds a "human-expert vetting" layer to the autopilot drone operation.

Data Acquisition and Data Transportation Automation.

The drone is cloud-enabled through the SBC and the SBC is programmed to maintain two-way communication with the Cloud when it has internet access.

The onboard SBC constantly collects drone status data and sensor data and saves it on its local file system. A separate background node service is actively pushing the data to the Cloud and deletes the local

files once it confirms that the data has been safely uploaded to the Cloud. There are two major benefits of this architectures:

- The data acquisition is not interrupted when the drone is not connected to the internet, temporarily loses the connection, or is in a slow internet environment.
- It is possible to send the drone to an internet-free zone and return with all the data saved on board, and then the drone can automatically upload the data once it is in an internet available environment. In this case, although the data uploading is not in real-time during a flight mission, it still significantly accelerates and standardizes the data acquisition pipeline. There is no need to manually connect the drone to a computer or remove and insert an SD card to a card reader.

While the onboard SBC is constantly attempting to upload data to the Cloud, it is also configured to constantly listen to the cloud database where the users might upload a new flight mission, or send a flight commands such as "start mission" or "terminate mission." For example, a complete flight mission contains several way-point and hot-point tasks that are pushed to the Cloud by the user via the web app, and the drone automatically downloads that mission, and then a mission execution order is issued and is received by the SBC. If the SBC detects that the drone is in idle mode in the destinated area and has sufficient battery life for the mission, the SBC will then start to send the individual flight tasks to the drone to conduct the flight mission step by step. The latency between the remote order to the actual execution of the flight are largely dependent on the internet speed and stability, which can range from 1 second to 10 seconds but are independent on the physical distance between the remote users and the drone. Therefore, it is completely possible to let the drone sit in the field with minimal attention and the users can send flight commands remotely. However, although it is technically feasible to send flight gesture orders in real-time via the web→Cloud→SBC→drone pipeline to mimic the manual operation with physical remote control, such as take-off, flying forward, rotate, land, hover, etc., we have found that this remote control is not very efficient due to the internet latency. Instead, we always pre-define the entire flight and data acquisition mission and upload it to the Cloud through the web app and have the drone automatically download and sync with the Cloud.

Automated Cognitive Data Analysis.

As stated in the Introduction section, the primary goal is to utilize drones for routinely surface surveillance and report anomalies. Most anomalies that happened on the surface of an oil and gas field are associated with some visible changes. Traditionally, this was largely done by manually examine and compare images. We will show that such cognitive tasks can be automated with very high accuracy. Figure 2 shows the high-level 2-step workflows; thus, first detecting changes occurring among different days, then further classifying the types of the changes. We will describe this practical workflow in detail.

Change Detection.

Change detection deals with the automatic detection of significant changes occurring in a scene by the elaboration of single or multiple frame sequences captured from single or multiple viewpoints by fixed or moving image devices. In our case, to perform change detection, the drone is precisely programmed to take a series of images at a set of given GPS coordinates with GPS-enabled autopilot functionality. At a different time, the drone could conduct the same missions. As can be seen in Figure 4, the Base frameset and the Day frameset are two different sets of airborne images obtained at the same GPS coordinates but

from different flights of different days or even seasons. Change is proposed to be detected in Day set compared to Base set. The workflow starts with a process called image registration, which computes the homography transformation between two frames in close coordinates but on different flights and thus these two frames are aligned. This step minimizes the impact of the inevitable misalignment of the frames on different days due to the intrinsic uncertainty of the GPS, as well as slight variations of view angles. In our case, Pairwise Speeded Up Robust Features (SURF) (Figure 5) (Bay et al. 2008) extraction is performed in each image. In this step, point correspondences among images are extracted. Some points of interest are selected at distinctive locations in the images such as corners, blobs, and T-junctions. The neighborhood pixels of every interest point can be represented by a feature vector. Next the feature vectors can be matched between image pairs. SURF extraction can be achieved by relying on integral images for image convolution and by building on the strength of the leading existing detectors and descriptors. For instance, a Hessian matrix-based measure can be used for the detector and a distribution-based descriptor. After feature matching, bundle adjustment (Triggs et al. 1999) is performed to minimize the reprojection error between the image locations of observed and predicted image points. It is expressed as the sum of squares of a large number of nonlinear, real-values functions. Thus, the minimization is achieved using nonlinear least-squares algorithms. It amounts to an optimization problem on the 2D or 3D structure and viewing parameters (i.e., camera pose and possibly intrinsic calibration and radial distortion), to obtain a reconstruction which is optimal under certain assumptions regarding the noise pertaining to the observed image features.

One of the major challenges of change detection tasks with aerial images arises when the two images to be compared were taken at different times or flights, even if they were taken at similar locations. Thus, the natural environmental change or hardware fluctuation might result in unwanted changes, such as landscape difference due to season shift, day weather conduction, a different CMOS noise level of the camera at different days, etc. (see Figure 6 for an example). The shift/change of the surface due to natural conditions makes it very difficult to align the image because the features contain many noises and are thus more difficult to match. Traditional alignment methods mostly may fail. In our proposed Iterative Homography Estimation (IHE) algorithm, an initial homography matrix is calculated. If the estimated homography matrix is valid (determinant of the homography infinitely approach to 1 and it is an affine transformation), the homography transformation is finalized. If not, outliers of the incorrect matched features shall be removed according to the estimated Homography matrix H', homography estimation is thus improved (Figure 7).

After and only after all the valid homography matrices are extracted from IHE, the valid homography matrices are applied to extract the change detection in the following steps. The next step after image alignment is to compute the difference between the two frames by simply subtracting the two frames. Ideally, all the pixels would be filtered except the ones that are real changes. However, different weather conditions could result in global brightness differences or large-scale brightness variations between the frames in different flights, which could result in a roughly uniform or low-spatial frequency gray background in the subtracted image, which could be filtered out using a high-pass filter. After the subtraction of the images and the distribution of background brightness, it is possible to find the pixels with significant brightness differences and standing out from the background. Contours of the blobs can thus be extracted and bounding boxes as well. Figure 8 shows a few examples of accurate image alignment

and the detected changes and their bounding boxes.

The highlighted and detected bounding boxes of changes are then fed into the change identification workflow.

Change Identification.

The mission of change detection is to detect unnatural changes (anomalies) of any type or shape. However, change detection does not provide information about the properties of the change other than size and location. Change identification, in addition to change detection, is therefore necessary to further classify the type of change. With the bounding boxes of the detected changes, we further implemented this change identification to complete the full cognitive data processing pipeline.

Deep neural network (DNN), often referred to as deep learning (DL), has been the driving engine for many breakthroughs in artificial intelligence and computer vision in the recent decade. Image classification is one of the major branches of DL, in particular, deep convolutional neural network (ConvNet). Modern ConvNets have achieved regular human performance in the task of image classification on some standard benchmarking datasets. However, DL-based image classification models are usually very deep and large, containing tens to hundreds of hidden layers and tens to hundreds of millions of parameters, and thus require large amount of training data. For example, the ImageNet dataset (Deng et al. 2009) contains more than 14 million images; popular image classification models like VGG19 (Simonyan & Zisserman 2014), Inception model (Szegedy et al. 2017), ResNet (He et al. 2015) require millions of images to calibrate their hundreds-of-million parameters.

In our case, 2,207 image patches scaled to the same 224x224 size are obtained from the change detection tasks based on our airborne image samples for training and testing. We have manually labeled all the 2,207 image samples into 12 major categories: Human (19 samples), Vehicle (26), Tools & Materials (5), Unidentifiable (37, anomalies with unclear types), Animals (58), Fabrics (189), Building (180), Plants (1,429), Stone (46), Shades (16), Water (20), and None (182, no real change other than weather). Furthermore, the categories are grouped into two major classes: positive (87 samples, true anomalies, including human, vehicle, tools & materials and unidentifiable), and negative (2,120 samples, false anomalies, all other categories). All the 2,207 samples are randomly stratified split into a training dataset and a test dataset on a 0.8:0.2 ratio.

Like most anomaly detection tasks, the data are highly imbalanced, meaning that most data points are normal and thus negative, and only a small fraction of data are truly anomalies. The mission of anomaly detection is to identify (almost) all anomaly while keeping the false alarm rate low. However, with this highly imbalanced data, the machine learning/deep learning model is unlikely to achieve that mission. For example, even the model just blindly predicts all cases as negative, it would still achieve an accuracy of 2120/2207=96%, while completely failing in the anomaly detection mission. To overcome this challenge, we augmented the training data by creating synthetic positive samples to bring the positive-to-negative ratio close to 1:4 in the training data. The synthetic samples are generated by rotating and flipping true samples, considering the azimuthal invariance properties of aerial images (see Figure **10** for an example).

Even with the data augmentation, the training data is still only around 4,000, significantly insufficient to train any large-scale deep learning model. Training a big DL model with such a small dataset would result in severe overfitting issues, i.e., the model would learn from the training data very well but fail to generalize on the test data. Thanks to the structure of the DL model, it is still possible to train a high-

quality DL model with small datasets, using the technique of transfer learning. Inside ConvNet models, the shallow layers behave as feature extraction layers using convolutional and max-pooling operations, and the deeper layers are trying to combine the features and making prediction. We selected ResNet-50, an excellent very deep convolutional neural network as our model. The ResNet-50 model is firstly pre-trained with the ImageNet dataset, and thus the model's shallower layers already learn how to extract various features from general images. We removed the last two layers of the standard ResNet-50 model and appended with several fully connected layers and a final SoftMax layer for 12-category classification tasks. In the actual training phase, we freeze the parameters of shallow layers and only train the last several layers. Because the number of parameters that need to be calibrated is now much smaller (on the order of tens of thousands instead of 50 million), we have successfully trained the model with only ~4,000 training images. The shallow pre-trained layers are merely used as a feature extractor for high-quality discriminative features, and the last few layers are the true classifier. An illustration of the model setup and transfer learning configuration can be found in Figure **9**.

Cloud Computation and Edge Computing.

Most algorithms mentioned above are deployed as serverless and stateless microservices on AWS Cloud for service scalability consideration. The computation services are initiated automatically once new data is pushed to the cloud storage. However, current cloud infrastructure service providers all impose some limitations in terms of available RAM and time out for those microservices, which are not favorable for computationally heavy services, such as most computer vision algorithms and prediction tasks with large scale deep learning model. It is, therefore, necessary to break large workflows into small isolated computations and use a permanent server to orchestrate all the microservices or use one microservice to trigger the next one. It is worth to note that, although deep learning-based tasks like change identification can greatly benefit from CUDA-based GPU especially in the training phase, in the serving phase, the forward prediction time even with only desktop-level CPU is completely acceptable. Therefore, those deep learning-based tasks are well suited for those serverless cloud microservices.

Although the application described in this paper does not yet particularly require high scalability, in general, the design and deployment of the data processing services should take scalability into consideration. This requires some careful design of the cloud service configurations. Moreover, the bandwidth of the internet speed in the fields is often very limited; it is therefore not feasible to have a central cloud service to keep streaming and processing the data acquired by all the drones that are in operation. Therefore, real-time data processing over edge devices is considered a good option in this application.

Edge computing, in contrast to cloud computing, utilizes edge devices. They are usually less expensive, lightweight, low power computation devices installed in or close to the data acquisition device. They aim at processing the data in real-time when the data is being sampled. Edge computing helps to decrease the complexity of large-scale highly-available cloud services, promotes true real-time data analytics, and reduces the requirements of high bandwidth internet. For example, instead of sending high definition image/video to the Cloud from the drone, it is possible to first process the image/video on the drone and only send the processed results to the Cloud. On the other side, the edge devices usually have much less computation power in comparison with the central Cloud, especially in the case of drone application, hence some computational expensive operations still must be performed on the Cloud.

In our application, because change detection and identification require comparison with previous recorded data stored on the Cloud, it is natural to deploy these algorithms on the Cloud. As an effort to reduce internet speed and cloud requirement, we have experimented with deploying some deep neural network based object detection algorithm directly on to the onboard single board computer. See the object detection section for more details.

Result

All the images are taken by the drone from a nadir viewpoint, in which the camera looks down on top of the targets of interest. The resolution of the airborne frame is 6000×4000. The drone image dataset contains a based image subset (dry season) and a test image subset (rain season). Both image subsets are based on the same 16mm wide-angle lens. The two subsets were taken from the same flight course close to the Andes Mountain area, but 6 months apart. This real-world dataset is realistic, natural, and challenging for video surveillance domain in terms of its resolution, diversity in landscapes, and human activity/event categories than existing airborne based datasets. Performing change detection and identification is very challenging, an object of interest such as a human only occupies 25-40 pixels. Figure 8 shows more examples of image alignment and detected bounding boxes of changes.

The dataset for change identification training is formed by sampling patches that we are interested in (positive anomaly: human, tools & materials, vehicle and unidentifiable) and patches that we have no interested object (negative signals: other categories of the 12) at the center. Note that the patch is the input to the models, and the models need not only the pixels occupied by the objects but also context information to help classify an object. Thus, the patches are not tightly surrounding the object. We use slightly larger ROI with size 224x224, because intuitively including some local context could help identify the changes. The choice is tested and proven in the experiment.

From the result of change detection, we select 2,207 image patches as the validated detected changes. The 2,207 patches are labeled into positive (87 patches) and negative (2,120) categories. In the 80% training data, the positive samples are augmented by rotation and flipping (see Figure **10**) to bring the positive-to-negative ratio from 1:25 to about 1:4 in the training data. All negative samples are also lightly augmented to double its size. In the end, 3,852 images are used for training and 922 images are used for test. The data are split at caution so that there is no information leak between training and test dataset, i.e., no image in the test dataset was synthetically generated from images in the training dataset.

Using the technique of transfer learning, the training to the 12-class ResNet model converges very quickly within 10 epochs in 10 min on an Nvidia GTX 1080 GPU workstation. As for baseline comparison, we also tuned and trained a support vector machine classifier (SVM or SVC) on the same training data and evaluate on the same test data. The results are further aggregated to positive and negative, i.e., binary. We found that the best results are achieved via training on multi-class data and then manually aggregate into binary prediction, instead of directly training a binary classification model. The reason is that 12-class labels contain much more information than simple positive vs. negative, and this helps the neural network be more discriminative and more generalizable.

Figure 13 shows the confusion matrices of the ResNet model and the SVM model for the 12-category classification tasks. It shows that both models deliver decent prediction capability on the test data. The ResNet model has an average F1-score=0.82, however, it clearly outperforms the SVM model with an

average F1-score=0.67. For the binary classification task, the detailed precision/recall/F1-score is reported in Table 1. The ResNet model outperforms the SVM baseline method by large margins in every aspect, especially for positive cases. The SVM model does not seem to benefit from the data augmentation practices and still have relatively poor performance for the actual anomaly cases, not as effective as ResNet, which is expected because ConvNet models are known to perform better with a bigger dataset.

Both the ResNet and SVM models output probabilities of their predictions. Therefore it is possible to choose different thresholds to decide if the prediction is positive or negative. This can be very beneficial for the anomaly detection tasks because in some application one might prefer a higher recall rate (sensitivity to anomaly) over precision (1-false alarm rate). By adjusting the threshold, we obtained the ROC curves and the precision-recall curves of both models in Figure 11 and Figure 12. The ResNet model achieves an almost perfect 0.99 area under the curve, compared to the SVM with 0.92. The precision and recall curves show that, with ResNet, there is large freedom of selecting proper threshold and still have great performance in both precision and recall. For example, with threshold = 0.2, the recall is 94%, meaning that for all anomalies, the model recognized 94% of them; and the precision is 78%, suggesting that only 22% of reported anomalies are false alarm. With SVM, the corresponding recall and precision are only 78% and 38%, respectively, indicating that every three out of five alarms issued by the SVM model are false alarms. In our application, a human is still behind the scene to validate all the warnings issued by the algorithm, and we can afford a higher false alarm rate. Therefore, we select a threshold=0.1, and the corresponding recall is 97%, i.e., 97% of the anomalies are successfully reported in the test set, and the false alarm rate is about 30% (precision=70%).

Edge Computing for Object Detection

In this section, we show an experiment of edge computing, in which we perform real-time object detection on the onboard SBC while the drone is conducting flight missions.

Object detection, different from image classification used in the change identification task mentioned above, is a computer vision tasked to identify different objects and their bounding boxes given a single image. That is, the object detection task takes a single image frame as the input and outputs a list of bounding boxes, the names of the objects cropped by those bounding boxes, and the associated prediction confidence. In contract, in the change identification task, we have already obtained the bounding boxes of the region of interest in the frames from the change detection task, and the algorithm only needs to classify the entire cropped image. In recent years, the deep learning community has significantly improved the speed (frame rate per second) and accuracy of the object detection algorithms with deep learning.

There are two major challenges to deploy the algorithms on the onboard computer to perform true-real-time object detection tasks. The first one arises from the application scenario. Unlike the applications of the self-driving car, mobile phone photograph, medical image process, and other scenarios, there are several special properties of object detection tasks in the drone surveillance application that impose both challenges and opportunities:

- **Objects of interest can only be viewed from the top, not from the side.** Hence, there is much less variation of the same type of object except for the azimuthal angle. The objects in drone images are azimuthal invariant, meaning that by rotating an image the object does not change. This is different from many object detection tasks. For example, in the self-driving car application, it is

unlikely for the model to witness an upside-down car or pedestrian. Another fact from this top-view-only property is that there will be much less chance of object overlapping in the drone images, which will make the tasks easier compared to many other object detection tasks.

- **Lack of resolution.** Most objects of interest in drone images are relatively small, ranging from 1 to 10% of the field of view. If there are too many layers of convolutional and max-pooling operations, the objects of interest might get compressed too aggressively.
- **Missing prediction opportunity.** Many modern deep learning-based object detection algorithms are fully convolutional, meaning that those algorithms can take almost arbitrary sized input. In their prediction layers, equivalently the algorithms are making one prediction to every 32x32 blocks. For example, if the input image size is 608x608, equivalently the algorithms are treating the input as a 19x19 coarse grid and giving one prediction per coarse grid box per anchor (the type of bounding box). If the objects of interest are too small, and there are more than one objects within one coarse grid box, the object detection algorithms will ignore some of the objects and only pick one object.
- **Object size inconsistency.** Similar to most other object detection problems, objects of the same type can have different size due to the camera distance. That is, the same object might occupy 40x40 pixel region in one frame and occupy only 15x15 pixel region in another frame.

To address many of the issues using the easiest and most controllable way, we flew the drone at consistent height across different missions. The height is determined based on the sizes of the objects of interests as well as the field of view so that it is unlikely that two objects of interest will occupy the same coarser grid. To increase variety of the data but still keep data consistency, we augmented the data by randomly rotating the images while keeping the aspect ratio and image size, as well as combined several public aerial image datasets (such as VEDAI by Sébastien et al. 2015 and UCAS-AOD by Zhu et al. 2015) rescaled to have consistent resolution and size as our own dataset. To eliminate the impact of distortion around the edge of the drone images, we only used the central 80% crops of all images to train the model.

We chose YOLO v3 (Redmon et al. 2018), a powerful state-of-the-art object detection algorithm to test the feasibility of object detection with drone image. Although Yolo v3's accuracy is lower than some of its competitors like RCNN-based methods in the standard benchmarking dataset, YOLO v3 is extremely fast due to its single-stage nature, meaning that it treats the object detection jobs as a single regression problem in contrast to two-stage algorithms like RCNN, which first generates region-of-interest and then predicts the objects within the region of interest. YOLO v3 achieves an incredible 20 frames per second (FPS) prediction speed for 608x608 image on a single Nvidia Titan X GPU, almost the same as the 24 FPS of standard movies.

We have successfully trained a YOLO v3 model on the combined dataset for vehicle identification tasks due to its abundance among all datasets. Figure 14 shows the detection examples of the trained YOLO v3 model on two test samples from the VEDAI dataset.

The second major challenge is the lack of sufficient computation power on the onboard device. The state-of-the-art deep learning-based algorithms demand large RAM, high computation speed and GPU supports. YOLO v3, despite its simplicity and relatively low requirement of computation, still needs 140.69 billion number of FLOP when predicting on a single 608x608 frame. Such a big model is not possible to be loaded into the onboard SBC we are using. As a compromise, we chose YOLO v3-tiny to

deploy to the 4-core ARM CPU SBC. YOLO v3-tiny is a minimal version of YOLO v3 requiring only 5.56 billion number of FLOPs per prediction, 4% of the standard YOLO v3 model, with a moderate sacrifice in model accuracy, which is 33.1mAP on the COCO dataset (Lin et al. 2014), 60% of its full YOLO v3 counterpart. However, it still takes more than 9 seconds to process a single frame on the drone onboard SBC. The reason is that the SBC does not have a CUDA-based GPU and so all the computation happens on one thread of the ARM-CPU, instead of parallelly running on the thousands of CUDA cores on Nvidia GPU workstations. In regular flight mode, the drone is traveling about 10 meters per second, and the 1/9 frame per second or 0.11 FPS suggests that the two adjacent frames that can be processed on the onboard SBC would be at least 90 meters separated. It is only acceptable when the drone flies at above 100 meters relative altitude to the ground so that the field of view is around 100 meters. However, the objects of interest would be too small in this case.

To solve this problem, we have combined an open-source project called NNPACK with Darknet, a C/C++ based deep learning framework developed by the same author of Yolo series. NNPACK (Dukhan 2016, also the forked version https://github.com/zxzhaixiang/darknet-nnpack) is an acceleration solution for neural network computations by providing high-performance implementations of convolutional operations on multi-core CPUs. With the help of NNPACK, we managed to deploy a full C/C++ compiled YOLO v3-tiny model to the 4-core ARM CPU SBC and achieved a 0.91 FPS frame rate, in comparison with 0.11 FPS without parallelizing the model over multi-core CPU, an almost 9X acceleration. This ~1FPS is fully acceptable in real drone deployment environment with the inter-frame distance shift only 1s*10m/s=10 meters.

Summary and Discussion

This paper focuses on an essential copter-based system for Surface Surveillance and Anomaly Detection. We proposed a smart autopilot drone system consists of an autopilot drone system, automated data acquisition, and transportation module, cognitive analysis pipeline and a cloud computation and edge computing platform. This system is one of the first attempts at detection and identification of O&G field surface anomalies from image sequences using deep neural networks and cloud computation techniques on a drone platform. Experiment results demonstrate that our proposed algorithms achieve very promising accuracy and robustness and perform better than traditional image classification methods.

In this particular study, we were more interested in unauthorized human activities in the field asset. Therefore, the objects of interest are human, vehicles, moved tools and materials, and other visible evidence. With more data provided, our work suggests that it is very possible to detect and identify many more different types of anomalies, such as surface facilities, oil leaks, truck activities, etc. So far, all the automated data analysis discussed are based on visible image. Given that the drone is equipped with an aligned thermal imaging camera and special gas sensors, it is also possible to conduct the data analysis by aggregating different data sources. For example, when combining thermal images with visible images, a completely new dimension of data is suddenly available, allowing for more in-depth analysis, such as wellhead leaks, facility failure/malfunction, and even pipeline leaks. Figure 15 shows two examples of drone images of an oil field overlaid with thermal mapping. Not only are the visible features shown but also the surface temperature of the facilities is available. This suggests that it is possible to build classifiers

to directly predict the health status of facilities when combined with standard visible band image classifiers.

Acknowledgment

The authors thank Robotic Air System (RAS) for conducting the flight and providing most of the data for training and testing the models. The authors also thank for the open-source libraries https://github.com/zxzhaixiang/darknet-nnpack for accelerating the edge computing algorithms.

References

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L. SURF: Speeded Up Robust Features, Computer Vision and Image Understanding, vol. 110, no. 3, 346-359, 2008.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li. Fei-fei. ImageNet: A Large-Scale Hierarchical Image Database. IEEE Conference on Computer Vision and Pattern Recognition, 2009.

Dukhan M. NNPACK: an acceleration package for neural network computations, 2016, https://github.com/Maratyszcza/NNPACK

Gómez, G., Green, D. R. Small unmanned airborne systems to support oil and gas pipeline monitoring and mapping. Arab J Geosci 10:202, 2017.

He, K., Zhang X., Ren S. Sun J. Deep Residual Learning for Image Recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015

Kochetkova, L. I. Pipeline monitoring with unmanned aerial vehicles. Journal of Physics: Conference Series 1015: 042021, 2018

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollar, P., Zitnick, C. L., Microsoft COCO: Common Objects in Context, CoRR, 2014

LeCun, Y., Bengio, Y., Hinton, G. Deep learning. Nature 436(521), 2015.

Ma, Z., Karimi Vajargah, A., Lee, H., Kansao, R., Darabi, H., & Castineira, D. Applications of Machine Learning and Data Mining in SpeedWise® Drilling Analytics: A Case Study. Society of Petroleum Engineers. doi:10.2118/193224-MS, 2018

Pajares, G. Overview and current status of remote sensing applications based on unmanned aerial vehicles (UAVs). Pho Eng Rem Sens 81(4):281-329

Pan, S. J., Yang, Q. A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 22(10), 1345–1359, 2010.

Razavian, A. S., Azizpour, H., Sullivan, J., Carlsson, S. CNN features off-the-shelf: An astounding baseline for recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 512–519, 2014.

Redmon, J., Farhadi, A. Yolov3: An Incremental Improvement. arXiv preprint arXiv:1804.02767, 2018.

Simonyan, K., Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015, 2014

Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. Proceedings of the AAAI Conference on Artificial Intelligence, 2017.

Szeliski, R. Image Alignment and Stitching: A Tutorial. Foundations and Trends in Computer Graphics and Computer Vision. vol.2, no.1, 1-104, 2006.

Sébastien, R., Frédéric, J. Vehicle Detection in Aerial Imagery: A small target detection benchmark. Journal of Visual Communication and Image Representation, 2015.

Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A. Bundle Adjustment — A Modern Synthesis. ICCV: Proceedings of the International Workshop on Vision Algorithms, 1999.

Wang, C., Li, S., Shen, Y., Song, Y. Evaluation of Feature Detectors and Descriptors for Motion Detection from Aerial Videos, Proceedings of the International Conference on Pattern Recognition, 2014.

Yosinski, J., Clune, J., Bengio, Y., Hod, L. How transferable are features in deep neural networks? Advances in Neural Information Processing Systems 27 (NIPS). 2014

Zhu, H., Chen, X., Dai, W., Fu, K., Ye, Q., Jiao, J., Orientation robust object detection in aerial images using deep convolutional neural network. Proceedings of the International Conference on Image Processing, 3735–3739, 2015.

Figures

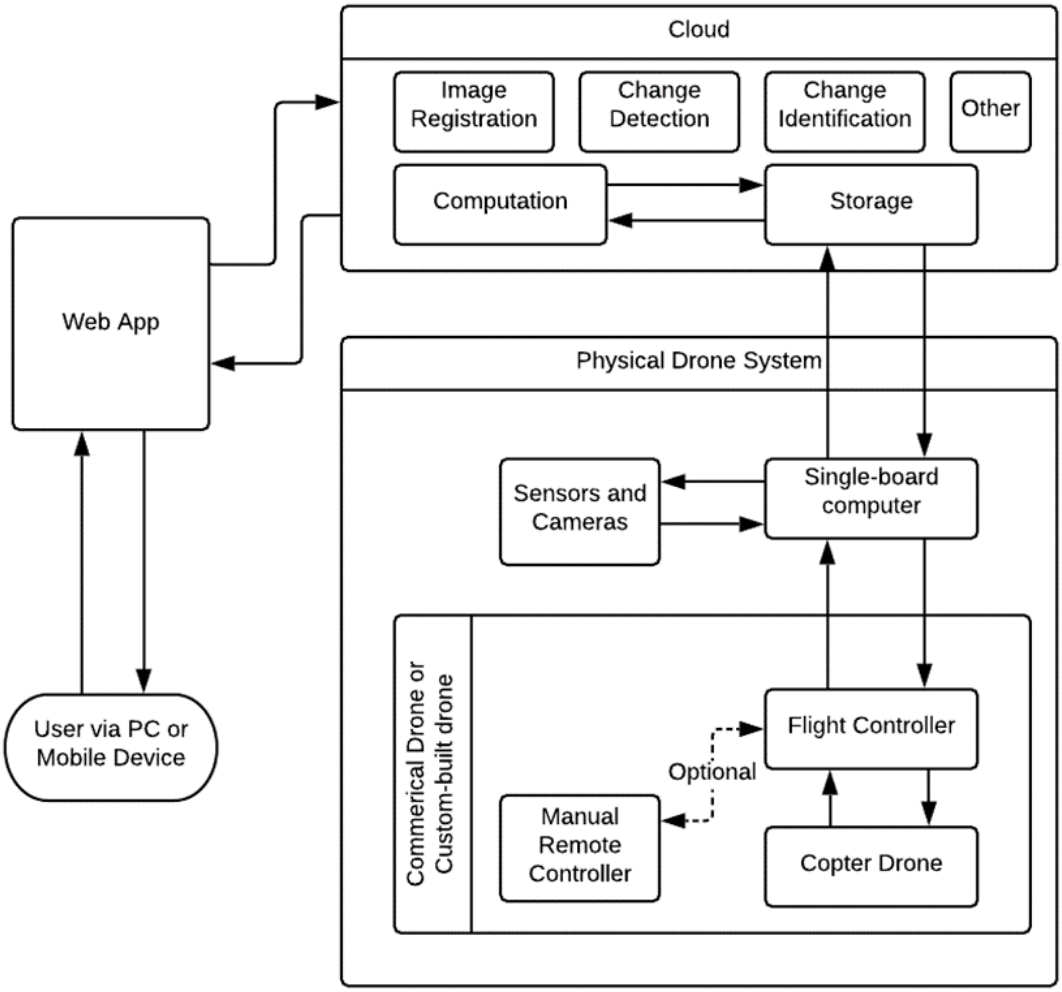

Figure 1—Autopilot drone system.

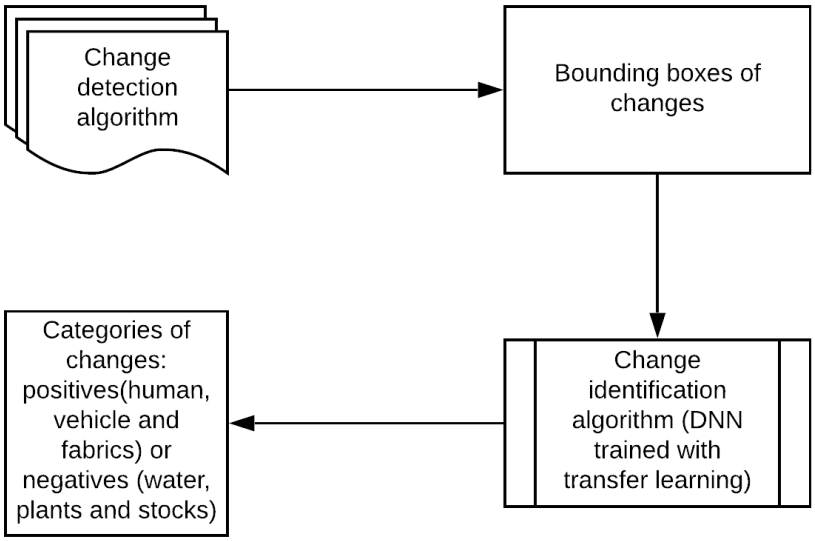

Figure 2—High-level flowchart of change detection and identification.

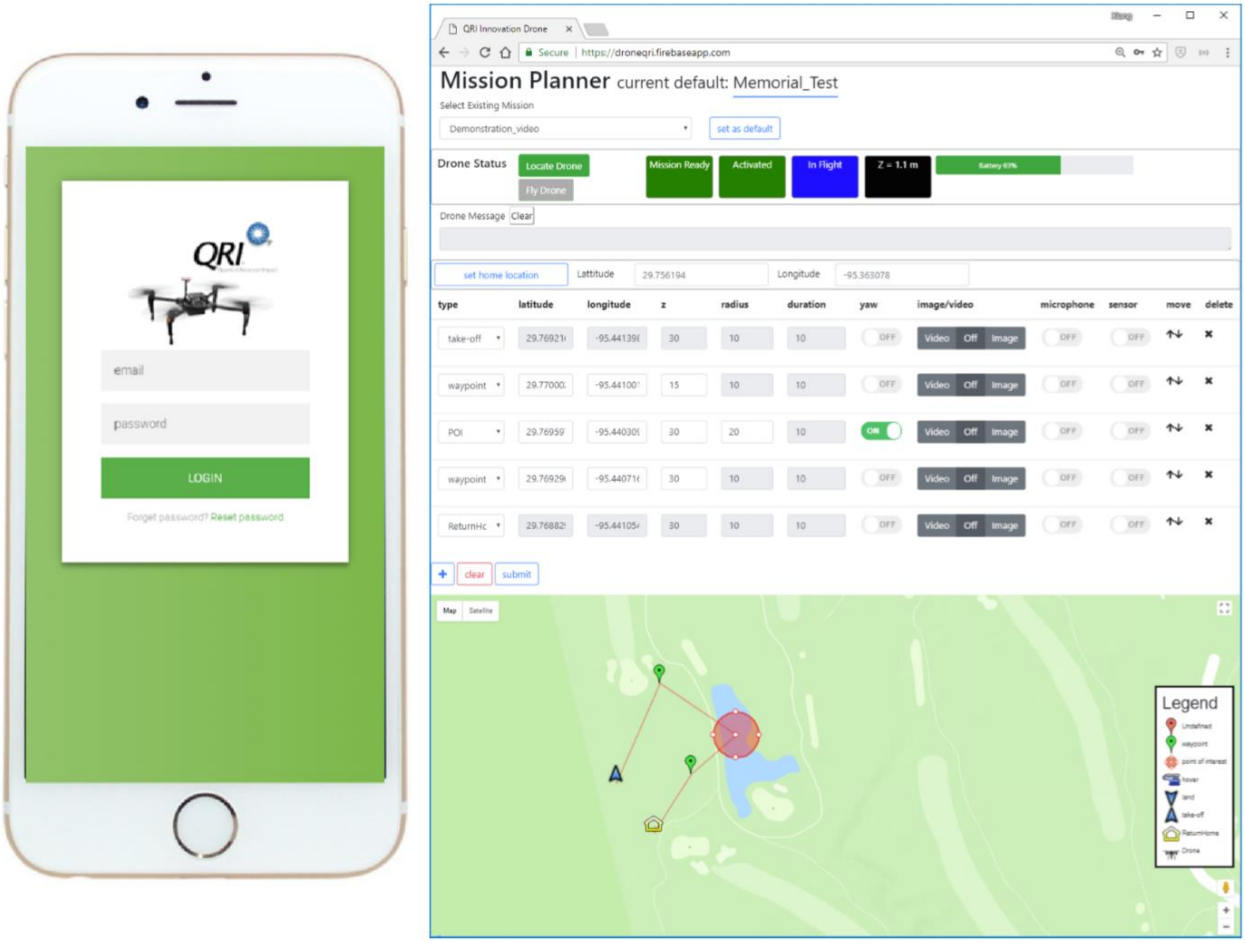

Figure 3—Web-based GUI/Mission Planner.

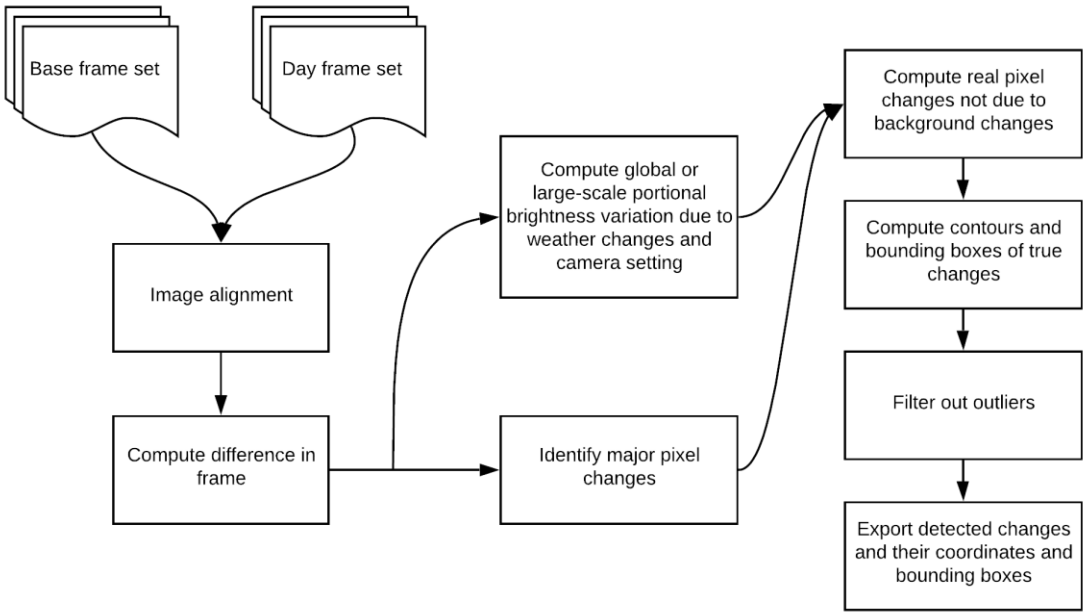

Figure 4—Workflow of change detection.

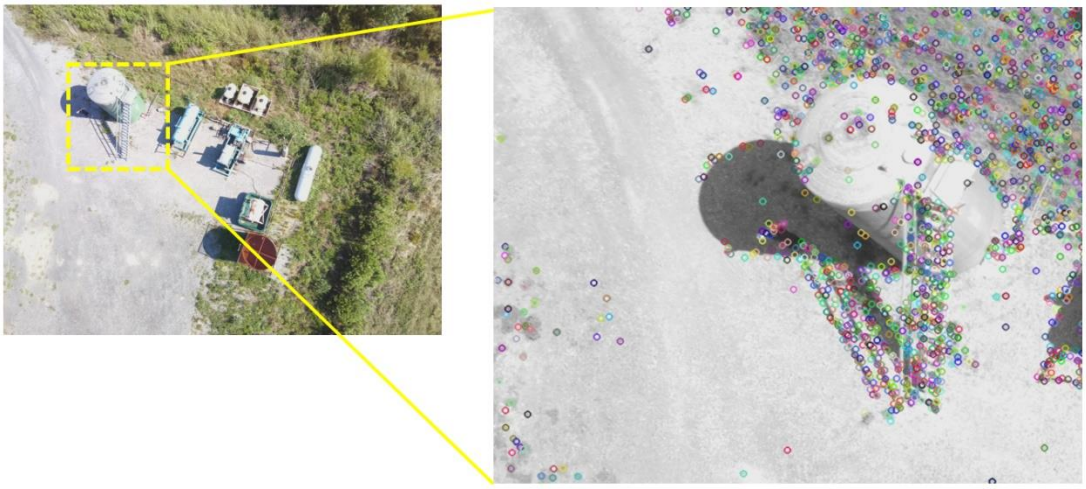

Figure 5—Feature detection (SURF).

Figure 6—Frames taken at the similar GPS coordinate in dry (left) and rain (right) season.

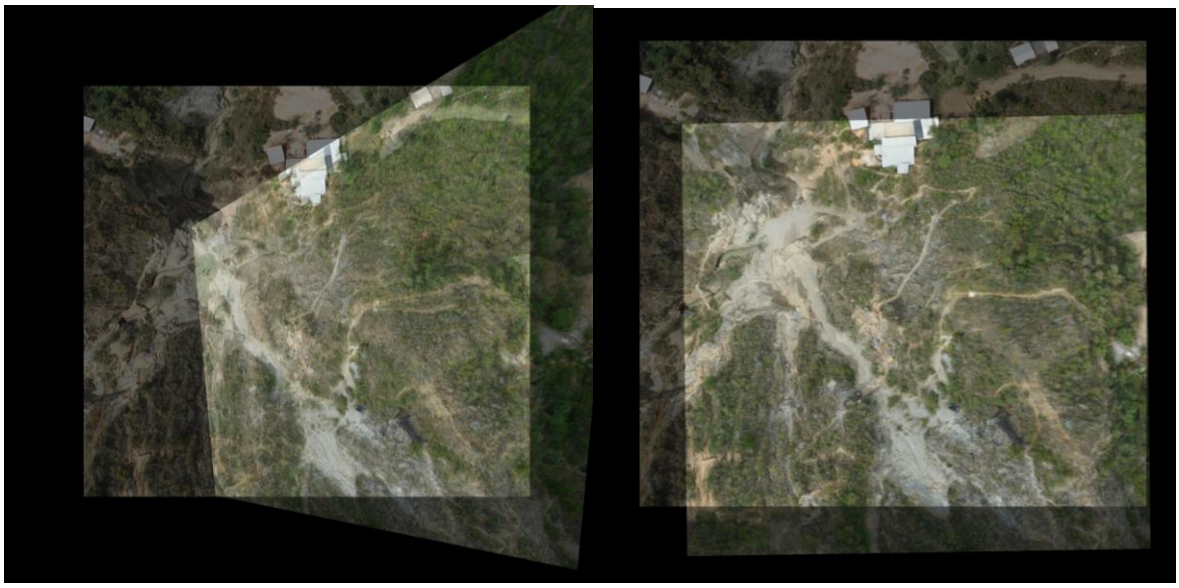

Figure 7—Image alignment by the traditional way (left) and iterative homography estimation (right).

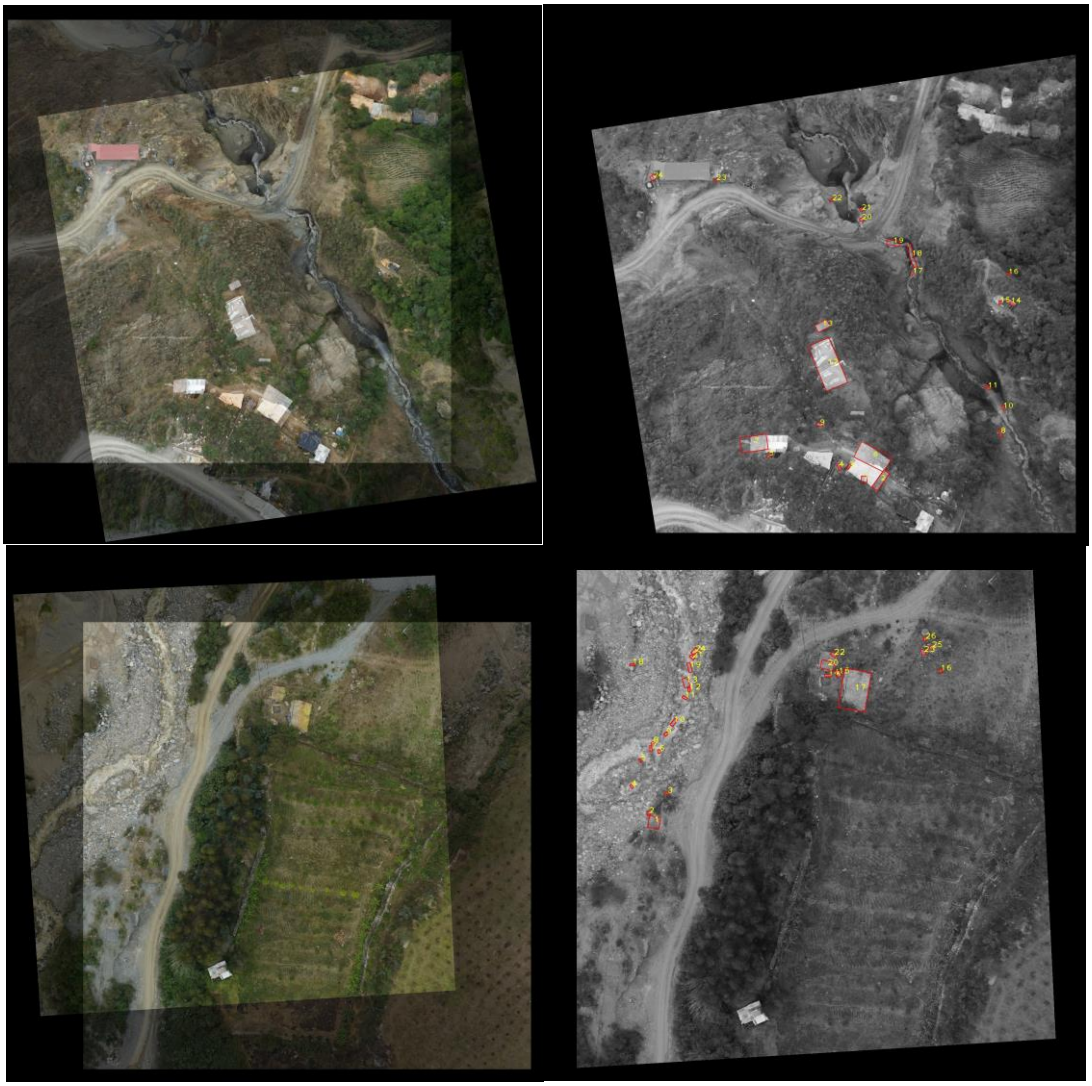

Figure 8—More examples of alignment (left) and change detection (right).

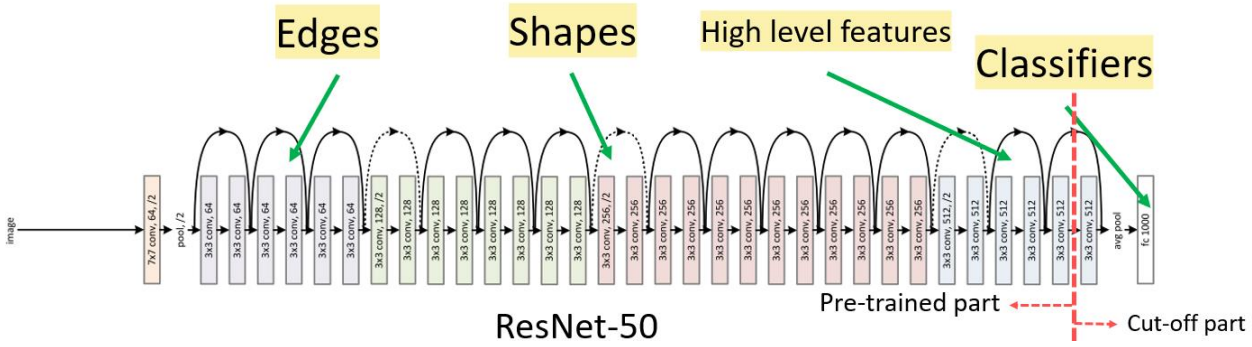

Figure 9**—Structure of ResNet and transfer learning.**

Figure 10**— An example of data augmentation to samples.**

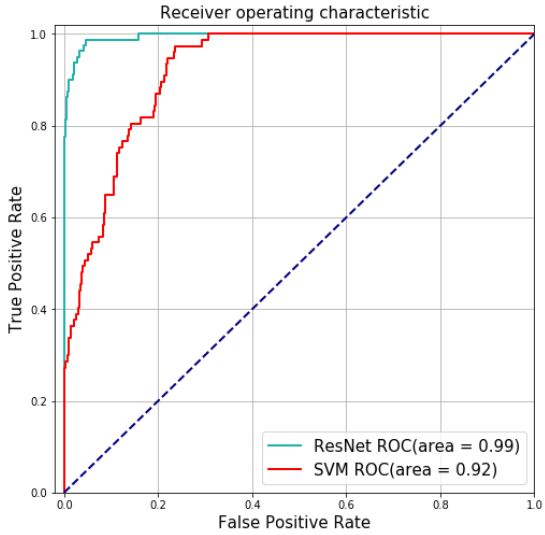

Figure 11—ROC Curve of the ResNet model and the support vector machine model on the test dataset.

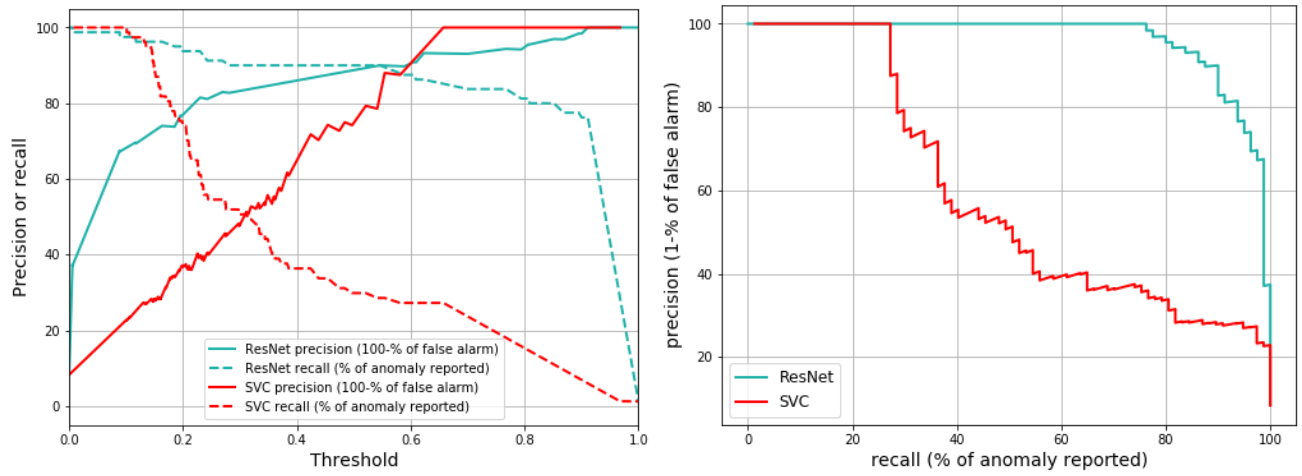

Figure 12—Precision and recall of ResNet model and support vector classifier on the test dataset. The precision of a model is the number of true positive divided by the total number of positive predictions, i.e., precision is the true alarm rate or 100% less the false alarm rate. Recall of a model is the true positive rate, i.e., out of all positive cases, the percentage of cases predicted to be positive by the model.

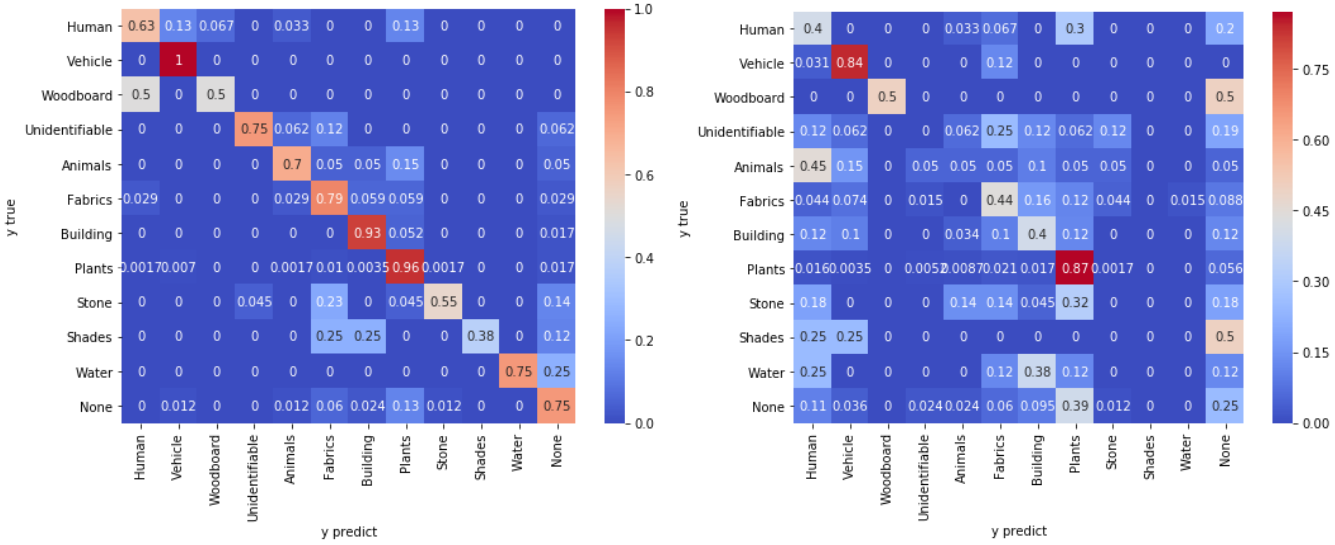

Figure 13—Confusion matrix of the 12-class ResNet (left) and the SVM model (right). Both confusion matrices are evaluated on the test dataset and normalized to the size of each category.

	Precision		Recall		F1-score		Support
	SVM	ResNet	SVM	ResNet	SVM	ResNet	
Positive	0.32	0.92	0.50	0.67	0.39	0.78	80
Negative	0.68	0.95	0.67	0.99	0.68	0.97	842
Avg/total	0.65	0.95	0.66	0.95	0.65	0.94	922

Table 1— Precision, recall, and F1-score on the test dataset from SVM and ResNet models. The default threshold=0.5 was used for all values.

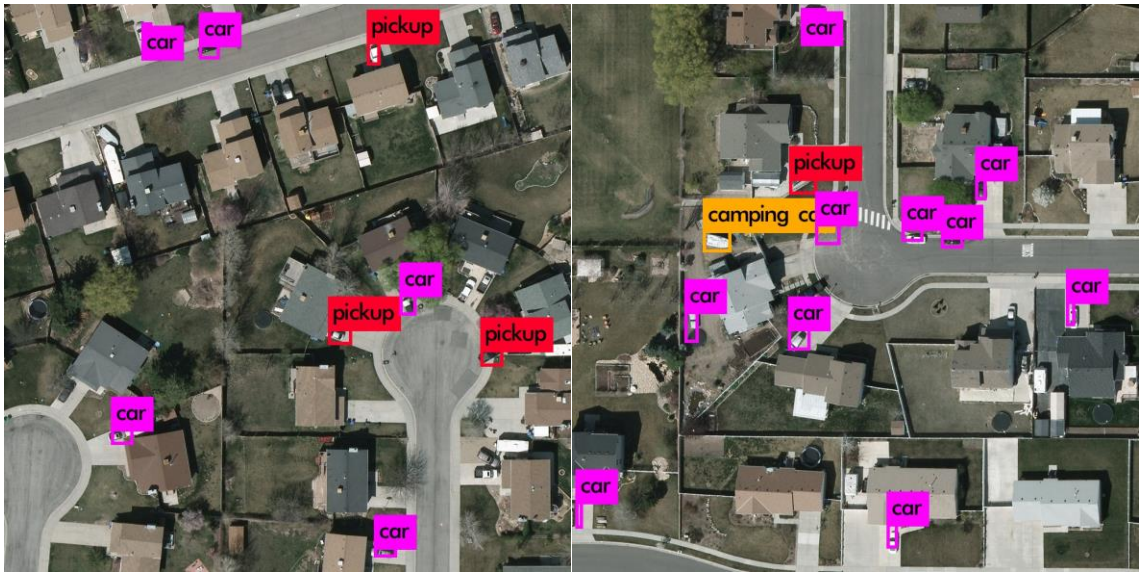

Figure 14—Object detection with the YOLO v3 model.

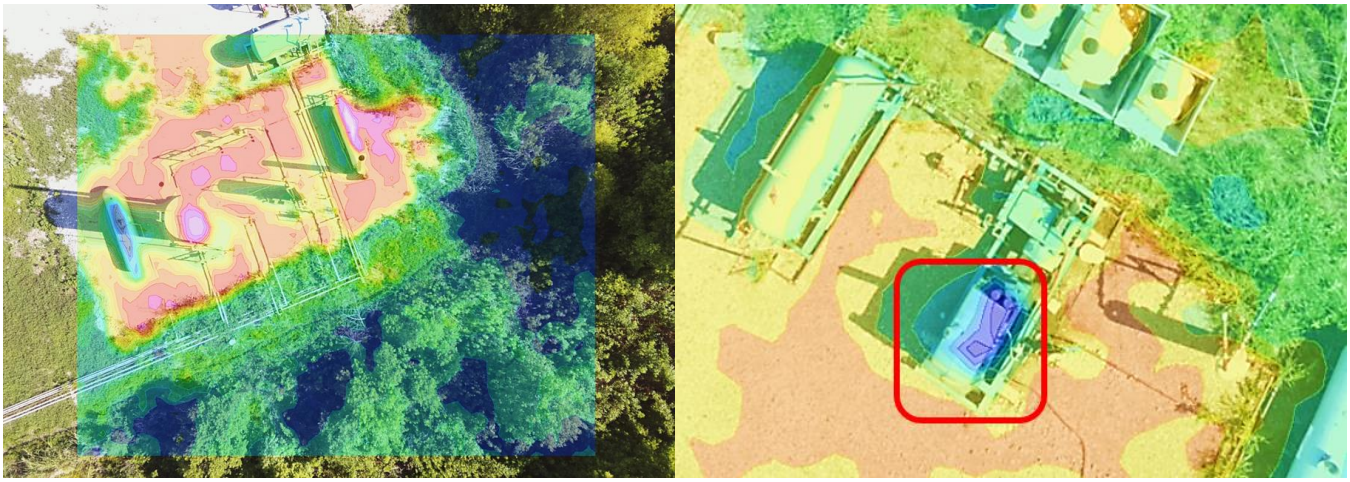

Figure 15—Examples of oil field drone images overlaid with the thermal image. The right panel shows a malfunctioning facility with significantly lower temperature than it should be.