Early Wildfire Detection, Geolocation, and 3D Reconstruction Using Aerial Visible and Infrared Images

Shun Li

A Thesis

in

The Department

of

Mechanical, Industrial and Aerospace Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Mechanical Engineering) at

Concordia University

Montréal, Québec, Canada

May 2023

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Shun Li**

Entitled: **Early Wildfire Detection, Geolocation, and 3D Reconstruction Using Aerial Visible and Infrared Images**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Mechanical Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Chun-Yi Su

_____ External Examiner
Dr. Wei-Ping Zhu

_____ Examiner
Dr. Chun-Yi Su

_____ (Co-) Supervisor
Dr. Youmin Zhang

_____ (Co-) Supervisor
Dr. Jun Yan

Approved by _____
Sivakumar Narayanswamy, Chair
Department or Graduate Program Director

_____ 2023

_____
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Early Wildfire Detection, Geolocation, and 3D Reconstruction Using Aerial Visible and Infrared Images

Shun Li

In this work, a novel unmanned aerial vehicle (UAV) system to detect and monitor early wildfire is proposed and verified with experimental flight tests on a DJI M300 quadrotor UAV. Several strategies and algorithms are employed to fuse the on-board sensor information from the visible camera, infrared camera, inertial measurement unit (IMU), and global navigation satellite system (GNSS). Developed strategies achieve the following functions: wildfire smoke and flame detection, wildfire spot geolocation, visible-infrared image registration, and wildfire local environment 3D reconstruction. For wildfire flame and smoke detection and segmentation, ResNet and attention-gate U-net are trained and deployed, which provide the semantic information for other modules in the proposed system. Simultaneous localization and mapping (SLAM) and multi-view stereo (MVS) algorithms are used to recover the camera poses and estimate the geolocation of the wildfire spot. With the estimated geolocation information, a mode-based visible-infrared image registration algorithm is developed to decrease the false alarm by fusing the visible and infrared images. After that, structure from motion (SfM) and MVS are utilized to recover the 3D scene of the local environment around the wildfire. Three independent outdoor experiments are conducted to verify the proposed detection, geolocation, registration, and local environment reconstruction algorithms. DJI M300 UAV together with a H20T camera are used as the platform in those flight testing experiments. The proposed system is proven to have promising applications in wildfire management.

Acknowledgments

As this is being completed, I would like to thank Concordia University for giving me this opportunity to complete my graduation and draw a successful conclusion to my master's study.

Then, I would like to thank my supervisors, Prof. Youmin Zhang and Prof. Jun Yan . Their rigorous academic style, highly academic enthusiasm, and the work style of excellence give me great inspiration and encouragement in both academic and life. Whether it's funding or learning opportunities, they always try their best to provide for me. Through these valuable learning opportunities, I have gradually grown up. The wildfire detection and management project has been working for over a year. It is because of the praise given to me by the two teachers when I succeeded and the encouragement and guidance given to me when I failed that this project was able to reach today.

Our laboratory group gives me a place like home, and everyone in the laboratory has tried their best to help me in life and study. And let me achieve what I want! First, I would like to thank Linhan Qiao for giving me a lot of help in deep learning. After that, I would like to thank Qiaomeng Qiao for his help in the field of the multi-view geometry. Finally, I would like to thank the current students, Yufei Fu, Jin Li, and Xiaobo Wu as well as the graduated students, Zhewen Xing, Yintao Zhang, and Tianqi Yang for their hard work in helping me to prepare and realize the outdoor experiments of this work. Thank you for your great help!

Contents

List of Figures

List of Tables

Chapter 1

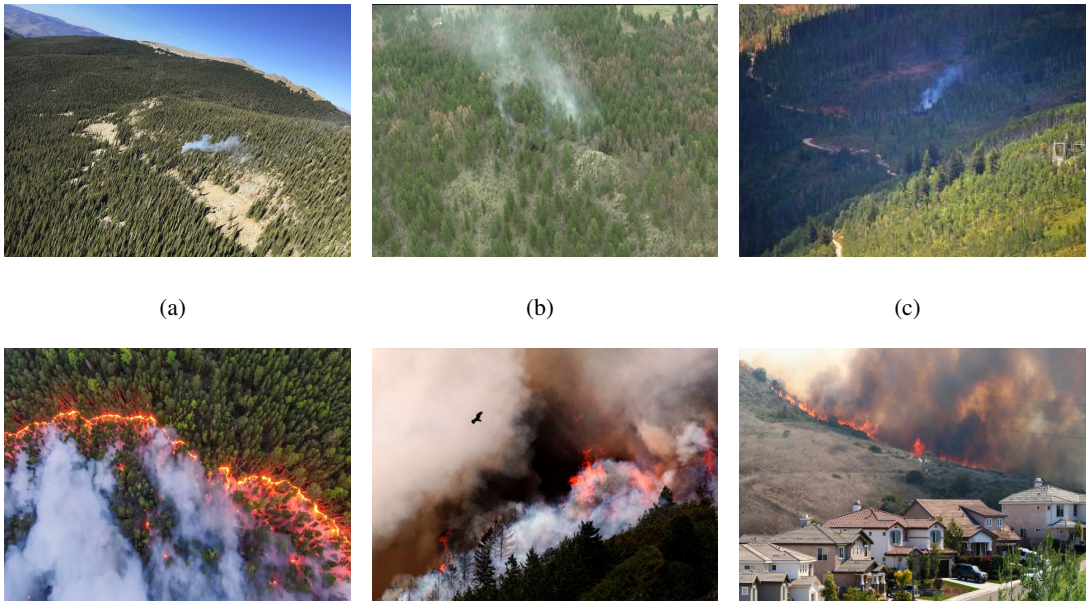
Introduction

1.1 Motivation

Wildfires are causing tremendous resources loss every year, in particular in forests worldwide in recent years due to climate change and global warning. Based on the statistics from the California Department of Forestry and Fire Protection (CALFIRE), there are an average of 3,311 fires and 21,343 areas of forest burnt weekly in the state of California in 2022 [1]. Moreover, wildfire threatens the habitats of animals and the safety of people who live near the forest [2]. In the report of U.S. Fire Administration [3], 16,600 people were injured, and 52 firefighters sacrificed in the tragedy of wildfire in 2022. However, significant injuries and damage can be minimized or even eliminated if the early wildfire can be suppressed before it is out of control. Early wildfire detection and management are essential in wildfire protection and governance [4, 5].

According to [6], the propagation and evolution of wildfire can be temporally divided into 6 phases. Wildfires before the "flash-over" phase can be considered at an early stage. For wildfire management, different metrics are concerned at different stages: For early wildfire, the geolocation of the fire spot [7], the temperature and intensity of the fire, and local environment information [8] are expected to be acquired. Geolocation is the guide to managing the fire suppression mission. While the local vegetation and terrain information can be used to predict the propagation of the fire. However, the fire front and the area information of the roaring wildfire are usually more essential to manage the control strategies than geolocation or the local information.

Early wildfire usually has barely visible flame and imperceptible smoke. Compared with roaring wildfires, satellites or watchtowers usually have insufficient resolution to detect the detailed area for the early wildfire. Wildfire is usually ignited at random locations in a wide range due to unexpected and various igniting sources, such as lightning, severe heat, and human activities. Hence, a wide-scope but the concrete, flexible, time-saving, and cost-friendly approach is needed to detect early wildfires. Unmanned aerial vehicles (UAVs) or drones have the considerable spatial working ability, flexible work time and modes, and relatively low cost [4, 9, 10]. Moreover, the spatial flexibilities enhance UAVs' ability to perceive wildfire at both long and short ranges. The flexibility of mounting various payloads allows drones to capture different physical information about wildfires as shown in Fig. 1.1.



(a) (b) (c)

(d) (e) (f)

Figure 1.1: (a)-(c): Early wildfire spot with thin smoke and no visible flame, which is hard to be detected with large-scale methods (e.g., satellites); (d)-(f): Roaring wildfire with large-scale smoke and apparent flame, which can be identified even with human eyes.

To the best of the author's knowledge, although the visible [11–13], infrared [14, 15], even multi-spectral cameras [16–18] are widely used for wildfire detection purposes, most of these works focus on wildfire monitoring and detection of the captured images. Moreover, visible and infrared images lack efficient fusion methods. Few real-world experiments have been conducted to verify

the system's performance under the real-time situation for the effects of these proposed methods. For wildfire spot geolocation and 3D scene reconstruction, only a few works have been carried out, and mostly use the fixed-baseline stereo camera [19, 20]. However, the fixed baseline limited the distance (depth) estimation range of the wildfire spot.

Therefore, this work aims to design and develop a novel early wildfire spot perception framework by integrating aerial visible-infrared information fusion. The proposed framework focused on not only the image-based wildfire detection, but also the geolocation and 3D reconstruction to provide more information around the wildfire spots for the firefighters.

1.2 System overview

The input of the proposed framework is the real-time data of on-board sensors, including visible camera, infrared camera, inertial measurement unit (IMU), and global navigation satellite system (GNSS). While the output is detection results, geolocation, and reconstructed 3D scene around the wildfire spot.

The brief architecture of the proposed framework is shown in Fig. 1.2.

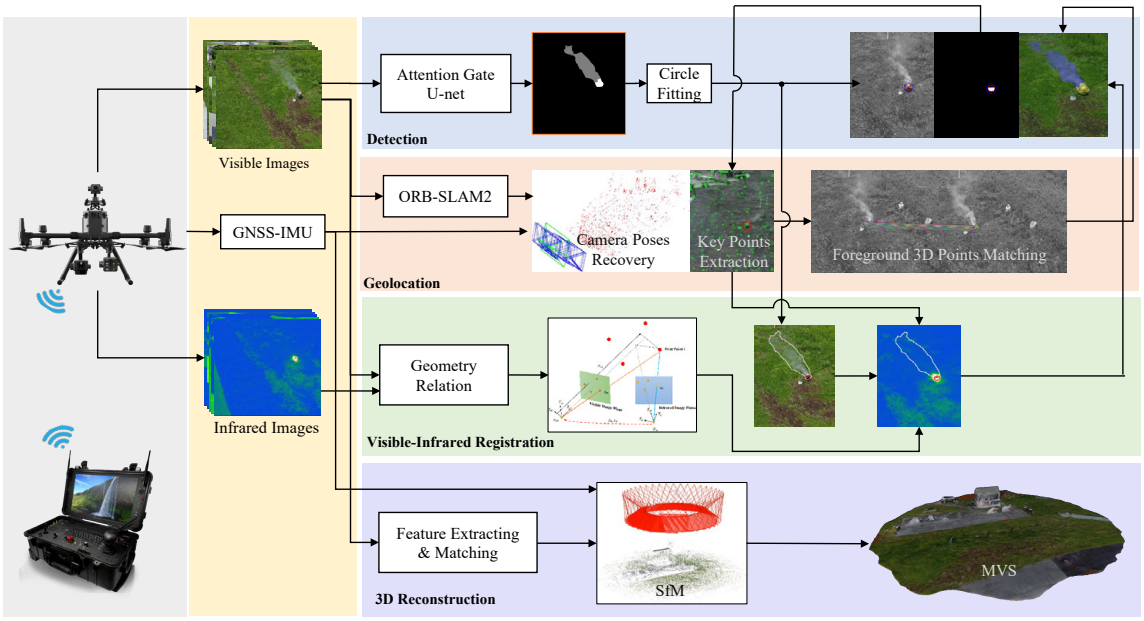


Figure 1.2: The architecture of the proposed framework : Aerial visible and infrared images, data of IMU and GNSS are used in four main functional modules: Detection, Geolocation, Registration, and Reconstruction.

To monitor potential wildfires, a three-stage strategy is designed. The three stages are *detection*, *geolocation*, and *reconstruction* . The "Detection & Semantic Segmentation" module is used in the *detection* stage; While the "Distance Estimation & Geolocation" and "Visual-Infrared Registration" module are designed for the *geolocation* stage; finally, the "3D Reconstruction" module is for the *reconstruction* stage. The following three sections introduce the logic and functions of the three above-mentioned modules, according to their stages separately.

1.2.1 Detection stage

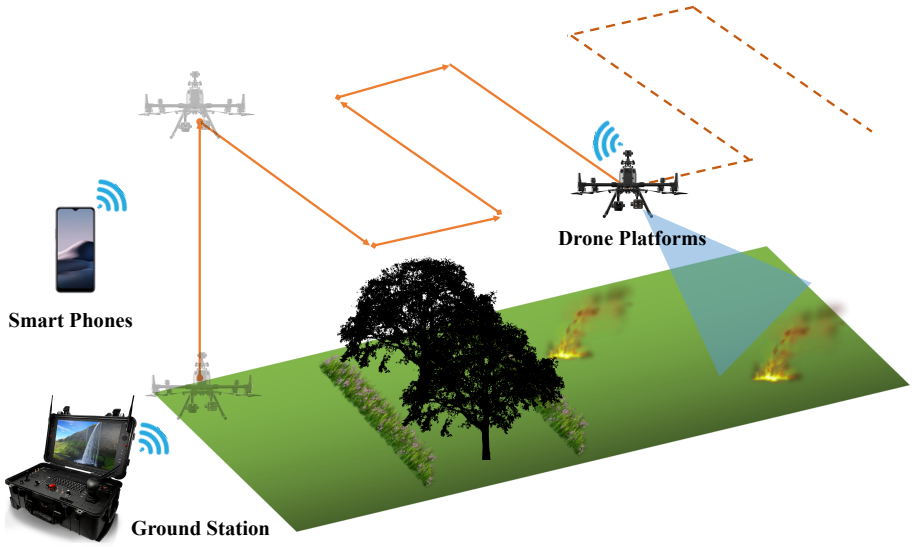


Figure 1.3: Detection stage: UAV flies along a predefined zigzag trajectory to cover the given zone.

As shown in Fig.1.3, the early wildfire monitoring mission starts with the detection stage. During this stage, the UAV will take off and fly along a predefined zigzag trajectory, ensuring that each part of the given zone can be covered. The zigzag path planning considers the UAV's power, the zone shape, and the sensor's effectual working distance. On the on-board computer, aerial images captured from a visible zoom camera are processed by the ResNet-18-based image classification algorithm. The visible image is classified into three possible classes: normal, smoke, and fire; simultaneously, color space is used to locate the suspect heat spot on captured infrared images. A confirmation request will be triggered if the classification algorithm or the infrared image-based algorithm detects the potential wildfire. Otherwise, the UAV will keep flying along the predefined

path until the mission finishes or the alarming power triggers returning requests.

As shown in Fig.1.4, visible and infrared images captured at the same time-stamp will be fed into the two types of detectors to determine if further confirmation is needed.

- If they both have negative results, no event will be triggered;
- If both two detectors give positive results or only infrared image-based detector has a positive result, the confirmation will be triggered and the gimbal will be rotated;
- If only the visible image-based detector predicts positive, a lower threshold will be used to segment the infrared image. At this moment, if the infrared image-based detector gives positive result, then the confirmation request will be triggered. Otherwise, this frame will be marked as negative.

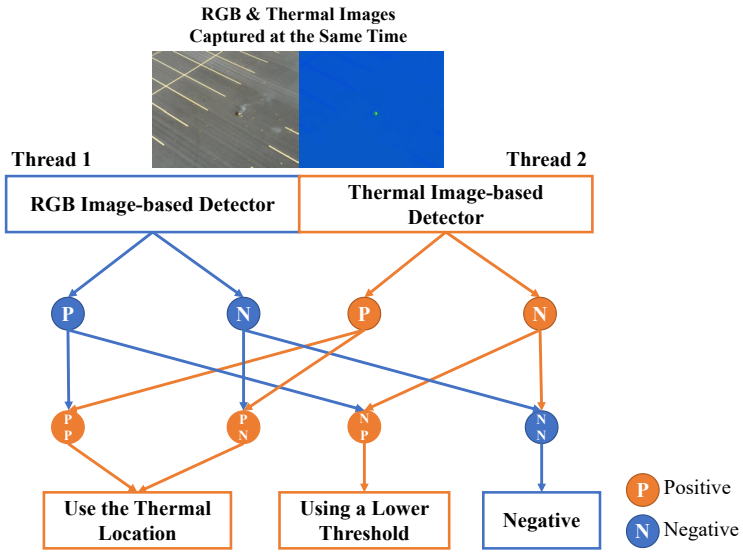


Figure 1.4: Confirmation request generating process.

Once the confirmation is triggered, the UAV will interrupt the patrolling mission and hover. Then a gimbal controlling algorithm will rotate the camera facing the potential wildfire point according to the segmentation on the visible images. The visible images are now sent to the ground station computer, on which the attention gate U-net-based algorithm runs online to segment the wildfire flame and smoke. Meanwhile, the visible zoom camera on H20T will also be set to "5×" in order to capture the potential wildfire point in detail, as shown in Fig. 1.5:

(a) zoom camera at "2×" (b) zoom camera at "5×"

Figure 1.5: DJI H20T zoom camera.

1.2.2 Geolocation stage

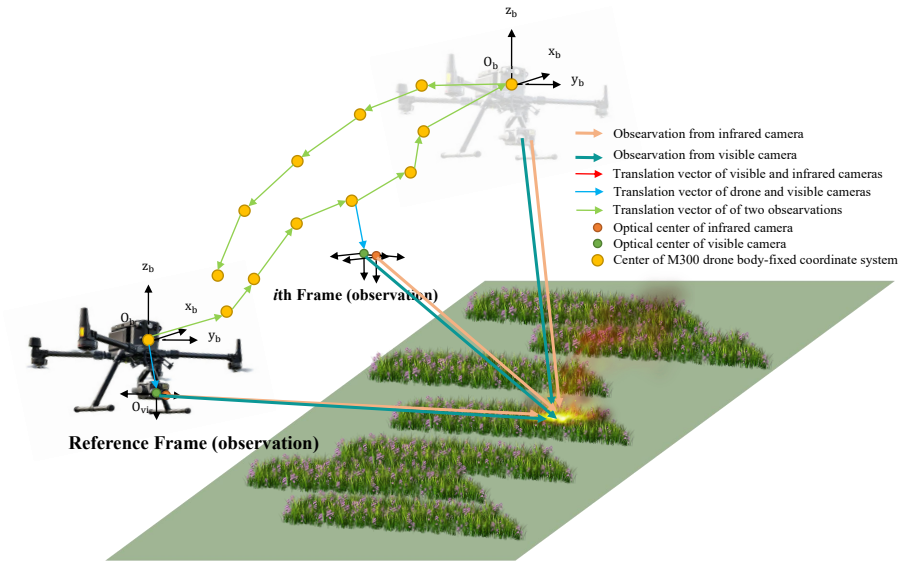


Figure 1.6: Geolocation stage: UAV flies laterally to estimate the depth of the wildfire spot.

After the confirmation generated by the attention gate U-net, UAV flies laterally from left to right to capture the visible images to finish the triangulation of the wildfire spot. The left-to-right movement is to generate enough parallax and improve the reliability of the depth estimation. As shown in Fig. 1.6, the visible images will be captured at each waypoint. The depth estimated is then used with the gimbal angle to calculate the geolocation of the wildfire spot.

The camera poses are recovered in real-time by monocular SLAM and IMU-GNSS information through sensor fusion. Once the camera poses with correct scale is recovered, the 3D points around

the wildfire spot can be triangulated by epipolar geometry. Using the semantic information, only the 3D points predicted as "fire" will be reconstructed.

After the triangulation, the depth of the wildfire spot can be estimated. With the estimated depth and the camera poses, the observed wildfire spot can be geolocated via geometry relationship.

The depth is also an input of the "Visual-Infrared Registration" module. Based on the projection model of the visible and infrared cameras, this module estimates the extrinsic and intrinsic parameters by optimization algorithm. The biggest advantage of this module is the calibration-free property. With this module, the prediction from the deep learning-based algorithms can be validated by the direct heat information from infrared camera. This strategy reduces the false fire alarm rate of the pure visible-based detection.

1.2.3 Reconstruction stage

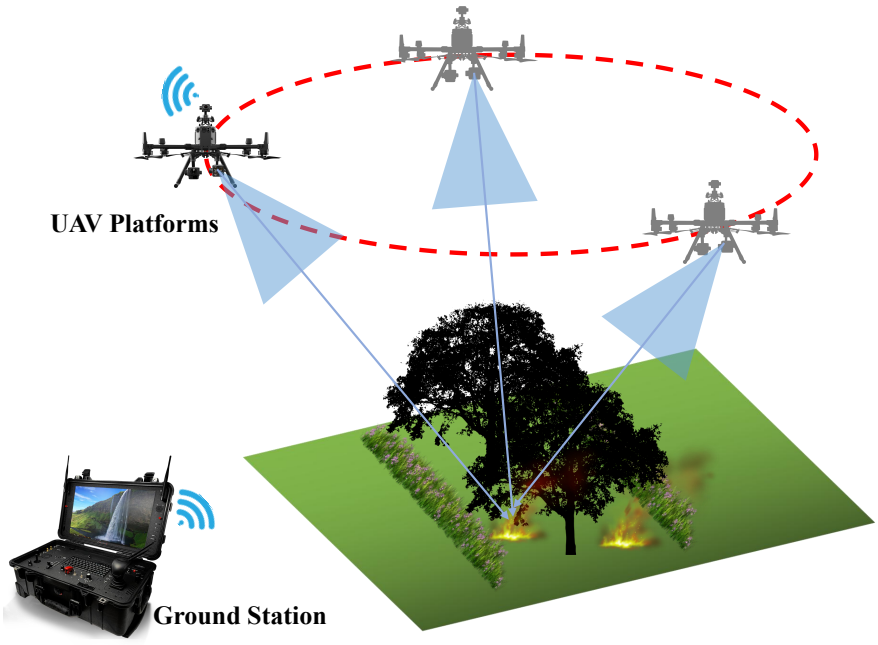


Figure 1.7: Environment 3D reconstruction stage: The UAV flies around the wildfire spot to capture more details of the local environment.

After the wildfire spot is geolocated, a circular trajectory is planned. As shown in Fig. 1.7, UAV flies around the estimated wildfire spot. During this stage, high-resolution images are now captured by H20T camera to reconstruct the local environment of the wildfire point. After finishing the whole

trajectory, all the captured images are sent to the ground station. The structure from motion (SfM) algorithm is first used to reconstruct the sparse scene: The image features and descriptors are extracted and calculated in parallel supported by the ground station GPU. Then the GNSS information is used to match the features between the images sequence. With the matched features, incremental global bundle adjustments is performed to reconstruct the camera trajectory and sparse 3D points of the scene.

The output from the SfM will be densified to the dense depth map. Then initial mesh is reconstructed, followed by mesh refinement. Finally, the terrain and vegetation information will be textured to the refined mesh from the captured images via multi-view geometry (MVS) algorithm. Comparing with the general-purpose wild environment reconstruction algorithms, the reconstruction here is designed for the wildfire management with the higher real-time property.

1.3 Thesis organization

This thesis is organized into seven Chapters.

- The motivation for this work and the proposed framework architecture are introduced in Chapter1.
- The second Chapter reviews the previous works, including the UAV-based wildfire management, aerial image processing, depth estimation algorithms, and multi-view geometry in the relevant fields.
- In Chapter3, the first *detection* stage is demonstrated: a two-step wildfire detection strategy is designed, which utilizes the computational power of UAV on-board computer and ground station computer.
- Chapter4demonstrates the wildfire spot geolocation algorithm in the *geolocation* stage. Both altitude and vision-based methods are introduced and compared. The model-based visible-infrared images registration algorithm is also introduced in this Chapter.
- Chapter5is for the *reconstruction* stage. The SfM-MVS algorithm is utilized to reconstruct the 3D local environment around the wildfire spot.

- In Chapter6, the outdoor experiment platform with DJI M300 UAV and H20T camera is introduced first. Then three outdoor experiments for the three main stages are designed and conducted to validate the proposed algorithm framework. The experiment results are also illustrated.

- Finally, in Chapter7, concluding remarks are presented, and recommended future works are described.

Chapter 2

Related Works

Compared with watchtowers, satellites, and other pre-placed sensors, the main advantage of UAV-based platforms for wildfire detection missions is the flexibility of the UAVs as well as multiple types of payload sensors. Based on that various physical information of the wildfire, the wildfire properties can be well-monitored.

According to the three stages of the proposed framework, the previous work in the early wildfire detection, geolocation, and 3D reconstruction are reviewed in this Chapter. It is notable that the critical process of the wildfire spot geolocation is to estimate the distance from the camera to the wildfire point, which is also known as the depth. Thus monocular camera-based depth estimation works are reviewed in this Chapter.

2.1 Aerial images based wild fire detection

2.1.1 Visible images

Due to their low price and intuitive results, RGB cameras are extensively applied in wildfire detection missions. Methods and algorithms in computer vision can be directly implemented in wildfire image processing. Both traditional computer vision and deep learning-based methods are wildly applied to the wildfire detection mission.

RGB, HSV, and HSI color space, Otsu's segmentation method [21] and optical flow [22] [23] are implemented to detect, segment, and track the wildfire flame. Fuzzy logic and extended Kalman

filters are combined in [24] to segment the wildfire smoke. In [25], the authors proposed 7 segmentation rules based on the YCbCr and RGB color space to segment the wildfire pixel.

Compared with threshold and rule-based methods, combining the human-defined features (color, texture, motion features) with traditional machine learning algorithms (random forest, support vector machine, etc.) usually achieve higher accuracy. Flame texture and color features in YCbCr space are extracted as wildfire features in [26], where an extreme learning machine is trained to classify through these extracted features. Multiple features and learning-based methods cross-experiments have been conducted and compared in [27].

The deep learning-based image processing methods represented by convolutional neural networks (CNN) can extract more types of features. These deep learning-based methods can be categorized into three main applications: Classification, semantic and instance segmentation, and object detection [28]. Several classification algorithms based on CNN, including VGG16, ResNet50 [29], and MobileNet, etc. have been implemented with DJI M100 and Odroid-XU4 on-board computer [30]. Based on Faster-RCNN, a spatial texture analysis algorithm for wildfire flame detection is proposed [31]. In [32], DJI M210 is used as the UAV platform on which deep learning algorithms are implemented for those mentioned above three main tasks. Many general and popular object detection and segmentation frameworks have been applied in the wildfire detection task: YOLOv3 for fire detection is trained and implemented on the DJI drones, and Manifold on-board computer in [33] and [34]. A 360-degree sensor and region-based method are proposed to reduce the false alarm generated by the DeepLabV3+ neural network framework in [35]. Using the RGB image patches and CNN, the wildfire flame is classified and roughly located in [36].

Smoke and flame during the wildfire, with no rigid structure or salient features, can be considered as particular targets in the field of object detection tasks. For general object detection tasks, many popular frameworks are proposed and verified: In [37], ratio-and-scale-aware YOLO (RSA-YOLO) is proposed to detect small pedestrians in the outdoor environment. In paper [38], faster region-based convolutional neural network (Faster R-CNN) is applied for object re-detection by tightly coupling with a multi-stream network, which solves the drift problem of tracking. However, the performance of those proposed general frameworks significantly relies on the computational platform and size of the training dataset.

On the other hand, U-net has a relatively lightweight structure and acceptable segmentation performance training with smaller datasets [39]. In [40], attention gate (AG) enhanced U-net and squeeze net modules [41] are applied for fire or flame detection. Although such a plan only worked on the flame classification, it showed that it is possible to squeeze the size of the U-net and combine it with an attention mechanism to decrease its false-positive detection.

Thus, U-net combined with attention gate are fine-tuned for wildfire segmentation in this framework.

2.1.2 Infrared images

Infrared cameras directly generate thermal images by capturing information that reflects the abnormal heat distribution of the wild environment. Each pixel in thermal images stands for the temperature of that corresponding point.

A DJI M600 drone-based wildfire detection, localization, and suppression system is proposed in [42], in which thermal images is used to segment wildfire pixels. Besides the value of each pixel in the thermal images, motion features of the wildfire can also be used for detecting: Optical flow and Otsu's segmentation methods are employed to segment and track the fire using thermal images in [43]. However, only considering the brightness of the infrared to detect the wildfire may lead to false alarms. Thus, in [44], both brightness motion and flicker features are taken into consideration to address this problem. Another methods to reduce the false alarms is the sensor fusion: According to [45], combining visible and infrared images can provide more accurate and reliable data for fire detection than using either one alone. Moreover, visible images can provide detailed texture information and color features of the scene, while infrared images can provide thermal radiation data and temperature differences between objects; the infrared-based detection methods can also take the reference from the smoke detection results from the visible images. Visible images can help to identify the type and location of fire sources, while infrared images can help to estimate the intensity and spread of a fire [46]. The combination can reduce the false alarms for the wildfire detection [47–49].

2.1.3 Visible-infrared images registration

Image registration aims to transform different images captured by the same or different sensors to an accordant coordinate system [50–52]. Among image registration methodologies, feature-based methods are the most common category. Feature-based methods first extract features from the salient structure. Then extracted features are matched, and the transformation of two images is calculated. Several works [53–56] are proposed to improve the feature extraction and matching process, while some works aim to optimize the transformation description mathematically [57, 58]. In [59], mutual information is used to register the images for medical purpose.

From the perspective of the visible and infrared images fusion, there are plenty of methods have been proposed: [60] proposes a fast and effective image fusion method based on guided filtering. The method decomposes each source image into a base layer and a detail layer using a Gaussian filter, which can achieve state-of-the-art performance for fusion of multispectral, multifocus, multimodal, and multiexposure images. Deep-learning-based methods are also used in the fusion application. In [61], an effective image fusion method using a deep learning framework is proposed to generate a single image which contains all the features from infrared and visible images.

However, feature extracting and matching are critical processes for feature-based registration. As far as we know, infrared images captured in a wild environment have barely salient or stable features compared with an artificial environment. Thus, based on camera geometry and projection relationship, we proposed a registration without feature extraction and matching.

2.2 Monocular camera-based depth estimation

Monocular cameras are considered to be bearing-only sensors, which makes it challenging to estimate the depth with a single image. Many methodologies are proposed to address this problem: traditional geometry-based methods (structure from motion (SfM) with multi-view stereo (MVS), simultaneous localization mapping (SLAM), etc.), and deep learning-based methods.

The descriptor-based feature matching is widely used in the SLAM and SfM systems. The role of feature detecting and matching is to connect the images so that the images can provide different observations of the same scene and these connections can also constrain the poses of the different

images.

2.2.1 Feature detection algorithm

Feature points can be categorized into blob detection and corner detection. Blob is the image zone with relatively high brightness in the gray-scale image. While corners refer to the corners of objects or the parts where lines intersect in the image. The blob detection methods mainly include the second-order Laplacian-Gaussian edge extraction algorithm (LOG), scale-invariant feature transformation algorithm (SIFT) [62,63], speeded up robust features (SURF) [63] and so on. Corner detection mainly includes Harris corner feature extraction, features from accelerated segment test (FAST) [64], oriented FAST and Rotated Brief (ORB) [65] and so on.

Table 2.1: Comparison between different feature detection algorithms.

Feature Detection Algorithm	Advantages	Disadvantages
SIFT [62,63]	It has great advantages in dealing with severe scale and rotation situations and is very robust to brightness and noise	The calculation dimension is large, and the operation speed is very slow
SURF [63]	Comparing with SIFT, the calculation process of SURF is simpler, and the time consumption is shorter. It has rotation and scale invariance, good robustness	Calculation is relatively slow
FAST [64]	Fast calculation speed	Susceptible to noise, no rotation invariance
ORB [65]	Fast calculation speed and stable rotation invariance	No scale invariance

2.2.2 Geometry-based sparse methods

With multiple images, the scene's structure (even the dense map) and the camera poses can be optimized and reconstructed simultaneously. SfM-MVS framework [66] are wildly adopted in the field of geoscience: SfM are employed to access landslide dynamics [67] and map landslide displacements [68], estimate depth and surface of snow [69, 70], monitor agricultural landscape [71, 72]. Compared with SfM, multiple types of sensors (LiDAR, inertial measurement unit (IMU),

global navigation satellite system (GNSS), stereo camera, RGB-D camera, monocular camera, etc.) can be integrated to accomplish the 3D structure and camera poses online. In [13], SLAM is used to navigate the UAV while monitoring the wildfire in the simulation environment. However, SfM and monocular SLAM usually needs further scale correction because of the up-to-scale reconstruction. SLAM systems based on multi-sensors in [73–76] are used to map the forest. In [77], several popular SLAM frameworks are tested and compared in the forest environment.

2.2.3 Geometry-based dense methods

Dense depth image can be critical in the field of MVS. Dense depth images can be estimated by multiple geometry-based methods. In [78], the plane-sweeping algorithm is first proposed; then in [79], this algorithm is used in the 3D reconstruction. The main idea of plane-sweeping is to split the depth range into several planes and observe the points projected onto the image plane. The color consistency of the projected points can be used to determine if the points are on the surface of the objects. However, this algorithm is difficult to deal with the occlusion problem. When the scene's texture is weak and the illumination changes greatly, noise points will be reconstructed.

PatchMatch algorithm is first proposed in [80] and utilized in the field of MVS in [81]. The PatchMatch algorithm is a random initialization method that propagates and searches randomly from left to right and top to bottom of the image and iterates continuously to find the most suitable patch to achieve the purpose of image restoration. In [81], the PatchMatch algorithm finds an approximate nearest neighbor based on a plane. Then the depth and normal vectors of the neighboring pixels are transformed to the current pixel to obtain a spatial plane, and the spatial plane is projected onto the image to compare the consistency of the two. If the latter is more consistent, the stored value of the current pixel is replaced by the stored value of the adjacent pixel. So the propagation and random search are performed. This method yields better sub-pixels, but it is not effective when facing regions with missing textures

2.2.4 Deep learning-based dense methods

Different from geometry-based methods, deep learning-based depth estimation aims to acquire depth with a single image or multiple images: Considering the difficulty of traditional MVS in dealing with weak textured areas, thin structures, and reflective and projected surfaces, deep learning-based MVS algorithms are developed. In [82,83], convolutional neural networks (CNN) are introduced to be used in depth estimation. After that, MVSnet [84] is proposed, which is an end-to-end reconstruction model. As shown in Fig.2.1, the whole model can be separated into six parts: 1) feature extraction, 2) homograph transformation, 3) cost volume regularization, 4) possibility calculation, 5) initial depth map calculation, 6) depth map refinement.

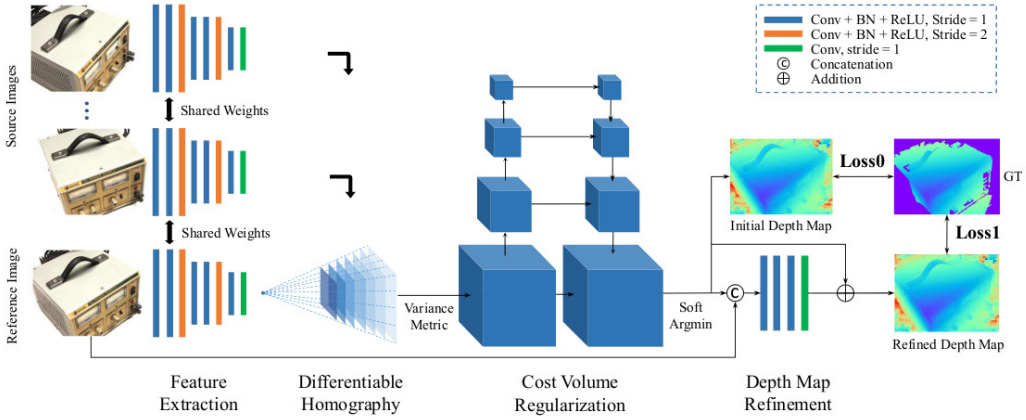


Figure 2.1: The network design of MVSNet. Input images will go through the 2D feature extraction network and the differentiable homograph warping to generate the cost volume. The final depth map output is regressed from the regularized probability volume and refined with the reference image.

In recent years, with the development of deep learning-based computer vision, various improvements have been proposed: depth estimation using pyramid-based depth residuals [85, 86]; semi-supervised adversarial framework training with a small number of images [87]; attention-based methodologies [88, 89]. For geoscience field, [90, 91] employ CNN on long-range monocular depth estimation applications.

2.3 3D reconstruction

The SfM-MVS process is the typical method to reconstruct the 3D scene. SfM is usually used to recover the camera poses and generate the sparse landmarks. Then MVS systems take the SfM's output as the input to reconstruct the mesh of the 3D scene. Some popular SfM-MVS frameworks are demonstrated in Table 2.2.

Table 2.2: Some SfM-MVS frameworks and their functions.

Frameworks	Point Cloud	Dense Cloud	Surface Mesh	Textured
Bundler [92]	✓	×	×	×
CMVS [93]	×	✓	×	×
Colmap [94,95]	✓	✓	✓	×
Meshlab [96]	×	×	✓	✓
MVE [97]	✓	✓	✓	×
MVS-texturing [98]	×	×	×	✓
OpenMVG [99]	✓	✓	×	×
OpenMVS [100]	×	×	✓	✓
Theia [101]	✓	×	×	×
VisualSfM [102]	✓	✓	×	×

2.3.1 Structure from motion (SfM)

SfM determines the scene's spatial and geometric relationship through the camera's movement. After the feature points are extracted and matched, the SfM reconstruction algorithm is used to recover the parameters of the camera, the relative position relationship between the cameras and the sparse point cloud of the scene. Finally, the bundled adjustment (BA) optimizes the camera poses, camera parameters, and scene structure.

The process of the SfM can be summarized as follows: feature extraction, feature matching, the estimate of the initial poses (rotation matrix R and translation vector t), triangulation, and BA. SfM-based reconstruction uses the camera motion trajectory to estimate the camera parameters. The feature points generated by the feature matching are relatively noisy, so the geometric constraints are often used to delete the wrongly matched feature point pairs. SfM can be categorized into two

main types, incremental method and global method.

Incremental reconstruction

Incremental reconstruction is to restore the camera's rotation matrix R and translation matrix t from two images, then add the image, use perspective-n-point (PnP) to calculate the R and t of the third image, and reconstruct more scenes through triangulation. The interference of noise will cause drift, so BA is required. Every time a new image is added after triangulation, it needs to be optimized.

However, incremental methods are known to have a fatal drift hazard due to the accumulation of errors and intractable loop closures of camera trajectories. Meanwhile, the quality of incremental SfM reconstruction strongly depends on the selection of initial image pairs and the order in which subsequent images are added. Incremental reconstruction of large-scale scenes will cause scene drift due to error accumulation, and it takes a lot of time to bundle and adjust repeatedly.

Global reconstruction

Most of the global SfM processing can be divided into the following steps. First, the global rotation of each view is calculated in the scene, and then the translation and structure of each view are also calculated on this basis. The benefit of separating the whole process into two steps is that relative two-view rotation can be estimated fairly accurately even for small baselines, which is not true for relative translation. These methods consider the entire polar graph, whose nodes represent views and where edges link views with sufficiently consistent matching points. All cycles of the graph generate multi-view constraints, that is, local relative motions in successive cycle nodes should be connected identically at the end of the cycle. Enforcing these constraints reduces the risk of drift in the incremental method. At the same time, since it only needs to perform a global BA on the data, its improvement in time efficiency is also very obvious. However, the disadvantage of global SfM is also obvious. Since it processes all data simultaneously, it is particularly sensitive to noise. If this processing method is used, the data must be carefully "filtered" to remove the influence of wrong epipolar information.

Comparison with Visual SLAM (v-SLAM) and SfM

Both the v-SLAM system and the SfM system can estimate the cameras poses and generate the spares point cloud. However, there are several differences between visual SLAM (v-SLAM) system and SfM system:

- The cameras in the v-SLAM system are a known model for which the camera intrinsic parameter is calibrated and fixed. But for the SfM system, the cameras can be different models with different intrinsic parameters, which can be treated as an unknown parameter participating the optimization.
- The input images of the v-SLAM system are usually time-related, but for SfM system, the input of the images can be random sequences.
- v-SLAM system is designed for online mapping and localization while the SfM system is proposed for offline map reconstruction.
- Under the usual situation, the reconstruction error of the v-SLAM system are much bigger than the SfM system.

Due to the features introduced above, v-SLAM systems are used for online H20T camera pose recovery and wildfire spot depth estimation. And SfM is used for the offline wildfire spot local environment 3D reconstruction.

2.3.2 Multi-view stereo (MVS)

The traditional pipeline of the MVS algorithm can be summarized as three categories:

- Images and camera poses → Point cloud reconstruction → Bounding box initialization → Volumetric fusion → Final refinement [93, 103]
- Images and camera poses → Depth range initialization → Depth map reconstruction → Bounding box initialization → Volumetric fusion → Final refinement [104–108]
- Images and camera poses → Silhouette extraction → Final refinement [109–112]

The first two categories can be considered as "depth→mesh" methods, in which the depth information of the objects should be first recovered and then fuse the depth information of every image to form the scene mesh.

Compared with silhouette-based reconstruction, depth-based methods usually achieve higher precision. These algorithms have relatively loose requirements of the hardware and have a larger range of the scene. Both geometry-based and deep-learning-based methods to estimate the dense depth map are introduced in Section 2.2. The intuitive logic behind the silhouette-based reconstruction is that the different views of the object can be treated as silhouette projection. Then using these views to "cut" the space, the final common part of the view will be the reconstructed object. The advantage of this method is its simplicity. However, the shortcomings are also apparent: this method relies heavily on the image resolution and the object's contour. Moreover, because of the "cutting method" to reconstruct mesh, the generated surface usually needs to be smoothed later.

Chapter 3

Early Wildfire Detection

As mentioned in Chapter 1, at the early wildfire detection stage, a two-phase strategy is designed: Running on the on-board computer, color space-based heat source segmentation, U-net-based smoke segmentation and ResNet-based classification algorithms determine together if there are early wildfire spots in the monitored area. If the wildfire is detected, the attention-gate U-net-based wildfire flame and smoke algorithm running on the ground station is used to provide the semantic information for the other modules.

ResNet-18 and U-net are more sensitive to smoke than flame. Thus the infrared image is used to verify if there is suspect heat, because those two models may generate false positive alarms.

3.1 ResNet-18-based wildfire classification

ResNet-18 is used for the on-board early forest fire classification, through which the RGB images captured by the H20T camera are classified into “normal”, “potential fire” and “potential smoke”. The priority of the 3 classes is “fire” $>$ “smoke” $>$ “normal”. Thus, wildfire spot with both flame and smoke is classified to “fire”. The classification model is relatively light weight which can process the camera video stream in time while the UAV is flying.

The ResNet [29] series use identity mapping to address the vanishing gradient and exploding gradient problem generated by the layers becoming deep. Fig. 3.1 shows the ResNet-18 architecture: green block shows the basic component in the ResNet; layers among the residual block learn

residual value instead of the original value, which makes the network more robust while training. Although ResNet has many variations, they can be separated into 3 parts: input, output, and convolutional module.

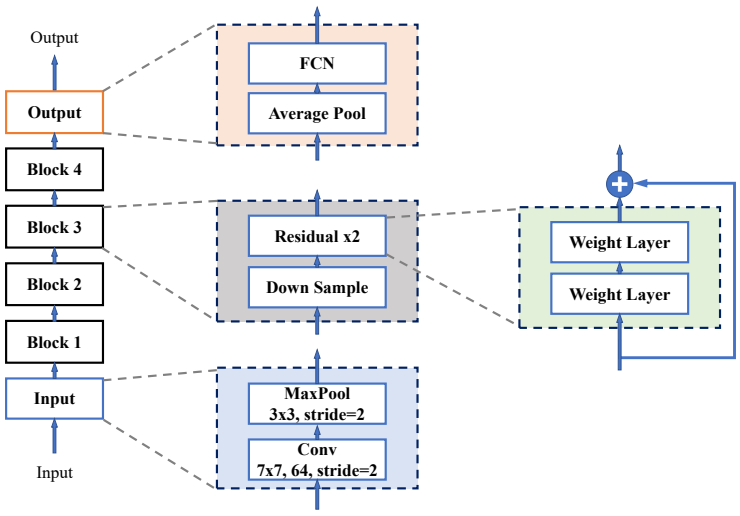


Figure 3.1: Brief architecture of ResNet-18 based wildfire classifier.

The input module consists of a 7×7 convolutional layer and a 3×3 max-pooling layer; the middle convolutional layer consists of 4 similar residual blocks. The features map channel accumulates from 64 to 512; finally, after an average pooling and fully connected layer, feature channel changes from 512 to 3, according to three possible classes of an image.

The model is implemented by Pytorch [113]. The training and validating process are introduced in Chapter6.

3.2 U-net based wildfire smoke segmentation

U-net [39] is widely used in image segmentation. In the proposed framework, U-net is designed to run on the on-board computer for a complement of the ResNet-18. Hence the early wildfire usually has very light flame, the U-net here is designed to segment smoke and background only to reduce the complexity and model size to fit in the on-board computational environment. The flame segmentation is then processed by attention-gate U-net running on ground station computer.

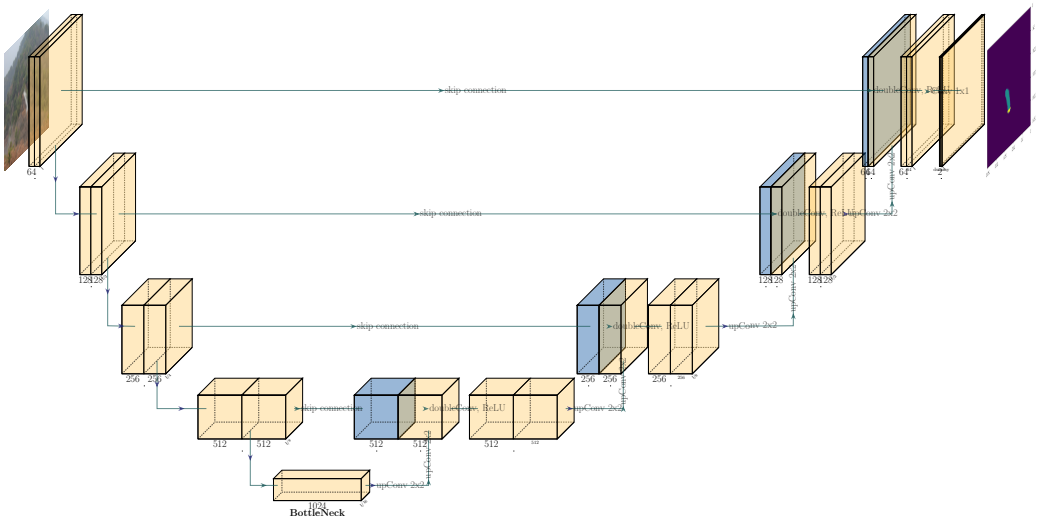


Figure 3.2: U-net structure for forest fire segmentation.

The network architecture is as shown in above Fig. 3.2, which can be described as consisting of a contraction path (left) and an expansion path (right). The contraction path is similar to the traditional convolutional network. It consists of an un-padded convolution with a convolution kernel size of 3×3. After each convolution, the features pass through a ReLU function and a maximum pool with a size of 2×2 and a stride of 2. After max-pooling, the number of channels is doubled.

The expansion path is up-convolution with the size of 2×2. The output channels of the upper convolution are half of the original and then connected in series with the corresponding feature map (after cropping) to obtain channels of the same size as the original and then pass through two sizes of 3×3 convolution and ReLU function. The corresponding cropping feature map is necessary because there will be a loss of boundary pixels in the process of our convolution. In the last layer, the desired target type is obtained through convolution with a convolution kernel size of 1×1. In this network, there are 23 convolutional layers. The network adopts the common Encoder-Decoder structure, and adds to the original structure the operation of directly intercepting information from the encoder and placing it in the decoder. This operation can effectively retain the edge detail information in the original image and prevent excessive edges.

Compared with traditional fully convolutional networks, U-net can achieve relative acceptable precision from less training data based on the application of skip connection.

3.3 Attention gate U-net based wildfire flame and smoke segmentation

One critical mission of the attention-gate U-net is to provide the location of the fire on the visible image for the other modules in the framework. Running on the ground computer, the model is responsible for segment the fire, smoke and background.

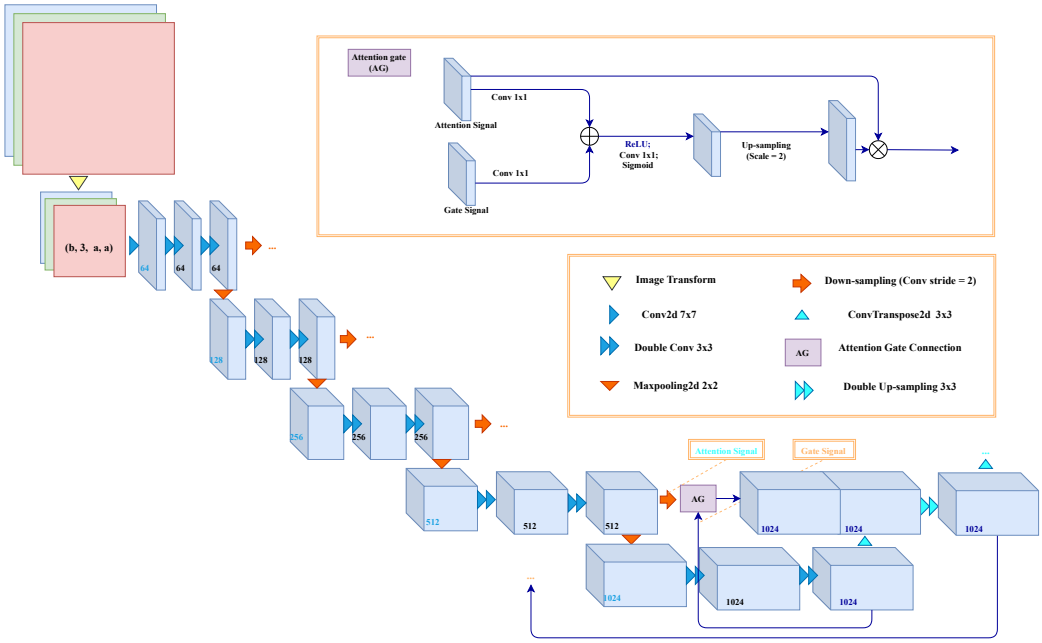


Figure 3.3: Architecture of Encoder part and bottle-neck part of attention gate U-net and the attention gate for wildfire segmentation.

An attention gate enhanced U-net based on ResNet-34 is firstly designed to generate semantic information. A trick is that the suspected flame area is labelled in the training dataset. So that the model could predict the suspected flame area even though there are no apparent flames.

Fig.3.3is a brief architecture statement of the proposed U-net. The main difference between this model and the original U-net is that the attention gates (AGs) replaced the skip connections of the original U-net. The skip connections just concatenated the Encoder outputs with the upsampling of Decoder parts. Because the gate signals play the role of "query", the foreground of attention signals is "paid more attention" when the attention signals from the Encoder part flow in. Therefore, such a model could decrease false-positive detection and increase detection accuracy.

The attention gate could be formulated as Eq. (1)

$$\begin{aligned} \boldsymbol{q}^l_{attention} &= \boldsymbol{\phi}^T(\boldsymbol{\sigma}_1(\boldsymbol{W}^T_{\boldsymbol{x}}\boldsymbol{x}^l_i + \boldsymbol{W}^T_{\boldsymbol{g}}\boldsymbol{g}_i + \boldsymbol{b}_{\boldsymbol{g}})) + \boldsymbol{b}_\phi \\ \boldsymbol{att}^l_i &= \boldsymbol{\sigma}_2(\boldsymbol{q}^l_{attention}(\boldsymbol{x}^l_i, \boldsymbol{g}_i; \bigtriangledown)) \end{aligned} \tag{1}$$

where $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$ separately remain the activation of ReLU and sigmoid function; $\bigtriangledown$ consists of a set of AG parameters: the linear transformations $\boldsymbol{W}_{\boldsymbol{x}} \in \mathbb{R}^{F_l \times F_{int}}$, $\boldsymbol{W}_{\boldsymbol{g}} \in \mathbb{R}^{F_g \times F_{int}}$, $\boldsymbol{\phi} \in \mathbb{R}^{F_{int} \times 1}$, and the bias terms $\boldsymbol{b}_{\boldsymbol{g}} \in \mathbb{R}^{F_{int}}$, $\boldsymbol{b}_{\boldsymbol{\phi}} \in \mathbb{R}$; $\boldsymbol{att}^l_i$ is acquired by linearly mapping concatenated features $\boldsymbol{x}$ and $\boldsymbol{g}$ to a $\mathbb{R}^{F_{int}}$ dimensional space.

Except for the AGs, the entire U-net structure is based on ResNet-34. Between every two down-sampling blocks, there is a residual connection to keep the feature information of the last block. The main advantage of such ResNet-based U-net is that a smaller training dataset could still ensure the prediction performance because features are extracted during each block and fed into the next block as residual. Transfer learning is applied to save training time.

Focal loss of cross-entropy is used as a loss function for this segmentation network model [114]. It is defined as:

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t) \tag{2}$$

where $\gamma \geq 0$ is a tunable *focusing* parameter; α_t represents α-balanced variant; p_t is defined as:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise.} \end{cases} \tag{3}$$

In the above $p \in [0, 1]$ is the model's estimated probability for the class with label $y = 1$.

The proposed model is built and fine-tuned with Fastai [115]. A variant of the 1 cycle policy [116], which uses a warm-up and annealing for the learning rate while doing the opposite with the momentum parameter [115], is applied to train the model.

3.4 Color space and sliding window-based thermal image segmentation

3.4.1 HSV color space

The infrared images captured by the H20T represent the heat radiation at each pixel. Inspired by the segmentation method proposed in [21], the HSV color space and threshold-based methods are employed to segment the potential heat source. After that, morphological operations are used to suppress the noise. Finally, the sliding window finds out the highest potential heat from each patch.

Considering the difficulty separating colors and brightness in RGB space, HSV color space uses hue, saturation, and value to represent a color. Only the hue channel is used for describing the color, and the other two channels describe purity and brightness.

In the proposed system, the H20T "north pole isothermal" mode generates the thermal images, as shown in Fig. 3.4. Under this mode, with the temperature increasing, the color of this point will change from blue to red; the brightness of this point will also be higher. It is noteworthy that thermal images generated directly by H20T are still in the format of RGB.

(a) (b)

Figure 3.4: Thermal and RGB images captured by H20T: (a) Thermal images generated by the H20T "north pole isothermal" mode. (b) Original RGB image (red circle: kettle).

Using HSV color space makes it possible to find a threshold for three channels, especially for the color channel. Conversion between RGB and HSV color space can be defined by Eq. (4)-(7).

$$
\begin{aligned}
R' = R/255 \quad & C_{max} = max(R', G', B') \\
G' = G/255 \quad & C_{min} = min(R', G', B') \\
B' = B/255 \quad & \Delta = C_{max} - C_{min}
\end{aligned} \tag{4}
$$

$$
H = \begin{cases}
0 & \Delta = 0 \\
60 \times (\frac{G'-B'}{\Delta} + 0) & C_{max} = R' \\
60 \times (\frac{B'-R'}{\Delta} + 2) & C_{max} = G' \\
60 \times (\frac{R'-G'}{\Delta} + 4) & C_{max} = B'
\end{cases} \tag{5}
$$

$$
S = \begin{cases}
0 & C_{max} = 0 \\
\frac{\Delta}{C_{max}} & C_{max} \neq 0
\end{cases} \tag{6}
$$

$$
V = C_{max} \tag{7}
$$

Then a threshold is defined based on the experiment to segment the heat point and background:

$$
P_{heat} = \begin{cases}
1 & T_l < H(x, y) < T_u \\
0 & \text{otherwise}
\end{cases} \tag{8}
$$

In Eq. (8), P is the rule to segment the heat point; $H(x, y)$ is the H value at (x, y); T_l and T_u are the lower and upper boundaries, respectively.

After the threshold segmentation, there will be discrete pixels around the heat point. The morphological operation can be used as the geometry filter to reduce the noise. Most of the noise can be suppressed with the opening operation (erosion first, then dilation).

3.4.2 Sliding window

It could be seen in Fig.3.5, the sliding window (in green) traversals every patch of the thermal image and compare the heat pixels inside. For the benefit of the real-time property, the size of sliding window in our proposed system is chosen from three fixed values.

After the sliding window traversal, there are usually multiple windows indicating that the heat points. To find out the most reliable path, non-maximum suppression (NMS) algorithm is employed:

The heat pixels amount in the sliding window is used as the score. Consider there are 6 rectangular boxes, named from A to F, sorted according to the category score of the candidate boxes, assuming that the score of belonging to the wildfire spot is sorted as $A < B < C < D < E < F$.

- Start with the maximum score rectangular box F, and judge whether the intersection over union (IOU) values of $A-E$ and F are greater than a certain threshold.
- Assuming that the overlapping of B, D and F exceeds the set threshold, then remove B and D; and mark the first rectangular box F, which is retained.
- From the remaining rectangular boxes A, C, and E, select E with the highest score, and then judge the overlapping between E and A, C. If the degree of overlap is greater than the given threshold, then throw it away; and mark E as the reserved rectangle.

This process is repeated until all the preserved rectangles are found.

For example, patch 1 is chosen as the location of the heat point in the following Fig. 3.5:

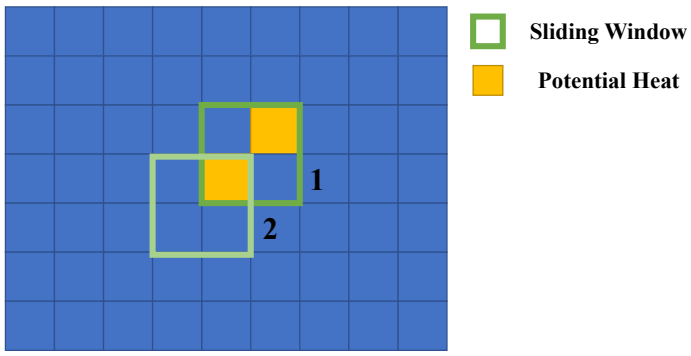


Figure 3.5: Sliding window generate the patches with heat point pixels.

Chapter 4

Early Wildfire Spot Geolocation

This Chapter demonstrates two early wildfire spot geolocation methods: UAV altitude-based geolocation and vision-based geolocation. To estimate the distance to the wildfire spot (depth), both methods need the wildfire spot location on the visible images. Compared with bare U-net running on the on-board computer, attention-gate U-net can segment wildfire flame and smoke with higher accuracy to provide the semantic information for the whole system. On the other hand, this more complex model needs a more powerful computational resource.

The key to geolocating the wildfire spot is to estimate the distance to the wildfire spot, which is also known as the depth of the wildfire spot in the camera-fixed coordinates system in the field of computer vision. The following altitude-based and vision-based wildfire spot geolocation methods are mainly designed to estimate this depth.

Once the depth is estimated, the geolocation of the wildfire spot can be calculated with the UAV's navigation information.

4.1 Altitude-based geolocation

4.1.1 Gimbal control

As shown in Fig.4.2, gimbal angles α and β are needed to estimate the distance to the wildfire. A simple way to simplify the geometry calculation is to put the heat point in the center of the infrared image. Then the gimbal angle can be directly used in the geometry calculation without considering

the angle between the observation and optical center. The input of the gimbal controller is the heat location on the infrared images and the current gimbal angles.

After the confirmation request is trigged, the gimbal controller will rotate and zoom the H20T camera to focus on the potential early forest fire point with the location in the thermal image. Inspired by [117], a PD controller is designed for the rotating task, which is shown in Fig. 4.1: X_{error} and Y_{error} are defined as the pixel difference between the desired position and the current position of the fire point. Then the controller is designed and tuned to generate the control command for the DJI gimbal controller.

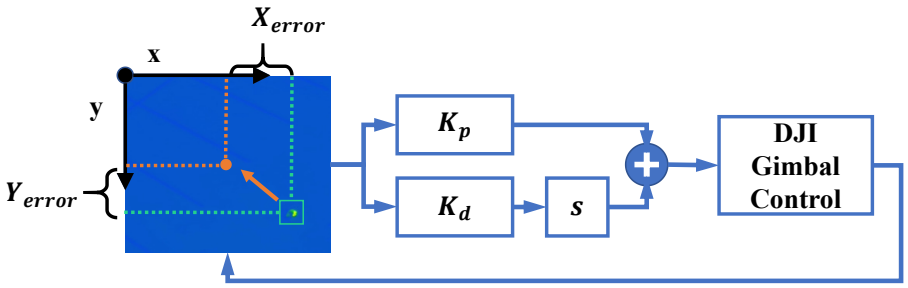


Figure 4.1: Gimbal control by tuning K_p and K_d of a PD controller.

The reason to use PD controller for the gimbal is the unknown parameter of the infrared camera. A well-tuned PD controller can work without the mathematical model of the controlling target.

4.1.2 Geolocation

Assumption 1: The UAV detects a relatively small area compared to the whole mission zone. So the terrain changes very slightly around the detected area. The earth's radius can be approximated to a constant in this area.

Assumption 2: The altitude of UAV can be estimated with limited error, which can be eliminated compared with the detected area.

Assumption 3: The offset between the UAV body-fixed coordinate system to the H20T camera-fixed coordinate system can be ignored comparing to the distance to the wildfire. Thus, the global positioning system (GPS) information of the UAV can be directly applied to the H20T camera.

The geolocation of the early wildfire point can be estimated by the gimbal angle as shown in

Fig.4.2.

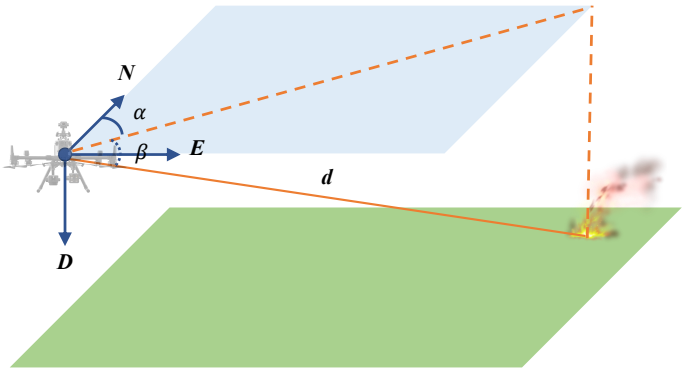


Figure 4.2: Geolocation of the early forest fire point.

In the navigation coordinate system (NED), the relative location can be defined as:

$$\begin{cases} l_N = d\cos\beta\cos\alpha \\ l_E = d\cos\beta\sin\alpha \\ l_D = d\sin\beta \end{cases} \tag{9}$$

Notice that the l_D can be estimated by the UAV navigation system, the distance to the wildfire spot d can be expressed as:

$$d = \frac{l_D}{\sin\beta} \tag{10}$$

where the α and β are the yaw and pitch angle of the gimbal, which can be also accessed by the DJI OSDK. Once the local position is calculated, the global position of the fire point can be easily expressed with the UAV's GPS information as:

$$\begin{cases} lon = lon_{M300} + \dfrac{l_N}{R_e} \\ lat = lat_{M300} + \dfrac{l_E}{R_e\cos(lon)} \\ alt = alt_{M300} - l_D \end{cases} \tag{11}$$

where the lon, lat, alt stand for longitude, latitude, and altitude of GPS location, R_e is the average radius of the earth.

The disadvantage of the altitude-based method is apparent: The precision of this method is strongly dependent on the three assumptions mentioned earlier. If the terrain or the altitude estimation is affected by random noise, the distance estimation may not satisfy the localization requirements. However, this estimation method is very simple and convenient, which is more appropriate for use on the on-board calculation system.

4.2 Vision-based wildfire spot geolocation

The multi-view stereo technique is a more reliable and robust approach to estimating the depth. The first step is to recover the camera poses when observing the wildfire spot. Then with the semantic information provided by the attention-gate U-net, 3D points around the wildfire spot can be first matched with image features and descriptors. Then with the triangulation method, the depth of the triangulation can be calculated.

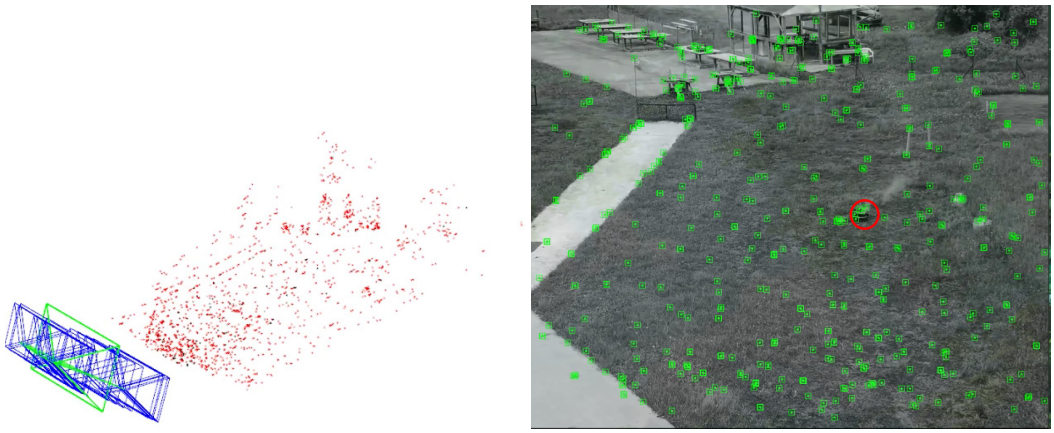
4.2.1 Camera trajectory recovery

One of the reasons the stereo camera may encounter a problem under the wildfire depth estimation can be explained as follows: For the concern of safety, the UAVs used for wildfire monitoring and management usually work at a relatively long range to the fire points. That is, the depth to be estimated is much larger than that in a normal usage scenario of a stereo camera. Such a large depth value versus a relatively small baseline leads to deficient parallax. Under this situation, noise contributes to most of the depth estimation.

The monocular reconstruction method can address this problem. The baseline between two measurements can be adjusted according to enough parallax requirements. However, there are still two main problems:

- Depth estimation through triangulation requires accurate trajectory of the camera, which can not be satisfied by the position and altitude provided by the UAV platform directly.
- Monocular camera trajectory recovery needs initialization.

- Because the monocular camera is considered as the bearing-only sensor, the scale of the reconstruction is unknown.



(a) (b)

Figure 4.3: (a) Camera trajectory recovered by ORB-SLAM2. Red points: landmarks; blue and green frames: current frame and historical keyframes. Under the outdoor experiment situation, most of the landmarks are located in the same plane. (b) ORB features extracted for trajectory recovery. Green rectangles: ORB features; the red circle is the synthetic wildfire point.

To address those problems, ORB-SLAM2 [118] is first employed to recovery the camera poses without scale; Then the ground truth scale is corrected by the fused position (GPS and IMU) provided by the UAV platform.

Monocular camera poses recovery initialization

At the beginning of the monocular camera poses recovery, both 3D points and camera poses are unknown. Thus the homograph matrix H or the essential matrix E is needed to estimate the initial motion.

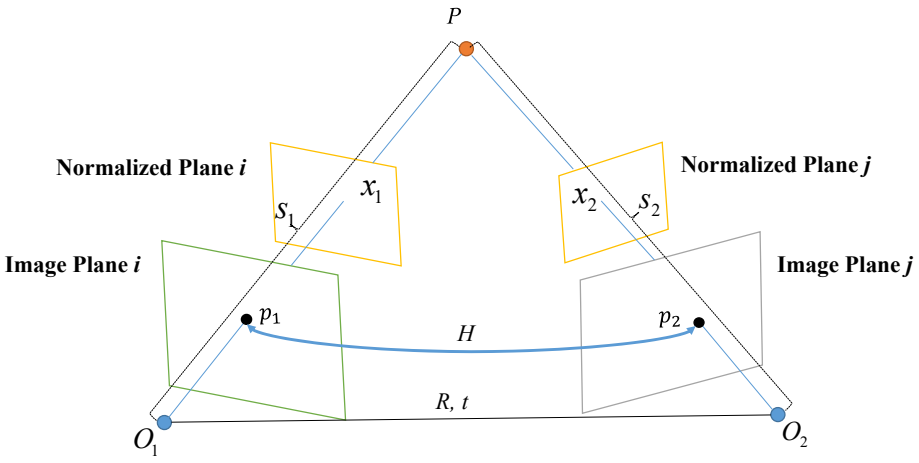


Figure 4.4: Triangulation scenario between 2 images.

Homograph matrix

The homograph matrix describes the 3D points projected into the image plane are in the same 3D plane. It also can be used to describe the translation between the two images. It should be noticed that the 3D points around the wildfire are usually on an approximate plane. Under this situation, the homograph matrix is more accurate than the essential matrix E to estimate the movement.

As shown in Fig.4.4, a 3D point P projecting on two image planes, the projective points on the images plane are p_1 and p_2 respectively. According to the definition of the homograph matrix, there is a translation relationship between the two points:

$$p_2 \simeq H p_1 \tag{12}$$

where the $\simeq$ is known as up-to-scale equal, which is also one property of the homograph matrix. Expand the above equation:

$$\begin{pmatrix} u_2 \\ v_2 \\ 1 \end{pmatrix} \simeq \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \tag{13}$$

According to the third row, remove this non-zero constant:

$$\begin{aligned} lu_2 &= \frac{h_1u_1 + h_2v_1 + h_3}{h_7u_1 + h_8v_1 + h_9} \\ v_2 &= \frac{h_4u_1 + h_5v_1 + h_6}{h_7u_1 + h_8v_1 + h_9}. \end{aligned} \tag{14}$$

rearrange it:

$$\begin{aligned} lh_1u_1 + h_2v_1 + h_3 - h_7u_1u_2 - h_8v_1u_2 = u_2 \\ h_4u_1 + h_5v_1 + h_6 - h_7u_1v_2 - h_8v_1v_2 = v_2 \end{aligned} \tag{15}$$

A pair of matching points can construct 2 constraints. Thus solving 8 unknown parameters needs 4 pairs of matches at least. Stack all 8 equations in a row, and a linear equation is formed:

$$\begin{pmatrix} u_1^1 & v_1^1 & 1 & 0 & 0 & 0 & -u_1^1u_2^1 & -v_1^1u_2^1 \\ 0 & 0 & 0 & u_1^1 & v_1^1 & 1 & -u_1^1v_2^1 & -v_1^1v_2^1 \\ u_1^2 & v_1^2 & 1 & 0 & 0 & 0 & -u_1^2u_2^2 & -v_1^2u_2^2 \\ 0 & 0 & 0 & u_1^2 & v_1^2 & 1 & -u_1^2v_2^2 & -v_1^2v_2^2 \\ u_1^3 & v_1^3 & 1 & 0 & 0 & 0 & -u_1^3u_2^3 & -v_1^3u_2^3 \\ 0 & 0 & 0 & u_1^3 & v_1^3 & 1 & -u_1^3v_2^3 & -v_1^3v_2^3 \\ u_1^4 & v_1^4 & 1 & 0 & 0 & 0 & -u_1^4u_2^4 & -v_1^4u_2^4 \\ 0 & 0 & 0 & u_1^4 & v_1^4 & 1 & -u_1^4v_2^4 & -v_1^4v_2^4 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \end{pmatrix} = \begin{pmatrix} u_2^1 \\ v_2^1 \\ u_2^2 \\ v_2^2 \\ u_2^3 \\ v_2^3 \\ u_2^4 \\ v_2^4 \end{pmatrix} \tag{16}$$

The approach to solve the H is also known as direct linear transformation (DLT). Then the rotation matrix and translation matrix can be calculated by decomposing H. The numerical and analytical methods can do the decomposition by OpenCV [119].

Essential matrix

According to the epipolar constraints, the essential matrix $E = t \wedge R$. It is a matrix of 3×3 with 9 unknown variables. The essential matrix satisfies the following properties:

- The essential matrix is defined by the epipolar constraint. Since the epipolar constraint is the constraint of an equal-to-zero equation, the constraint is still satisfied after multiplying E by any non-zero constant.

- The singular value of the essential matrix E must be in the form of $[\sigma, \sigma, 0]^T$.
- Due to the equivalence of scales, E has 5 degrees of freedom.

To estimate the essential matrix from the feature matches, the eight-point-algorithm is used. Consider a pair of matched points, their normalized coordinates are $x_1 = [u_1, v_1, 1]^T$, $x_2 = [u_2, v_2, 1]^T$. Then x_1 and x_2 satisfy:

$$(u_2, v_2, 1) \begin{pmatrix} e_1 & e_2 & e_3 \\ e_4 & e_5 & e_6 \\ e_7 & e_8 & e_9 \end{pmatrix} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} = 0 \tag{17}$$

Rewrite the matrix E in the form of vector,

$$e = [e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9]^T, \tag{18}$$

Then the epipolar constant can be rewritten in the linear form:

$$[u_2u_1, u_2v_1, u_2, v_2u_1, v_2v_1, v_2, u_1, v_1, 1] \cdot e = 0 \tag{19}$$

Stack all the points into one equation and obtain a linear equation system:

$$\begin{pmatrix} u_2^1u_1^1 & u_2^1v_1^1 & u_2^1 & v_2^1u_1^1 & v_2^1v_1^1 & v_2^1 & u_1^1 & v_1^1 & 1 \\ u_2^2u_1^2 & u_2^2v_1^2 & u_2^2 & v_2^2u_1^2 & v_2^2v_1^2 & v_2^2 & u_1^2 & v_1^2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ u_2^8u_1^8 & u_2^8v_1^8 & u_2^8 & v_2^8u_1^8 & v_2^8v_1^8 & v_2^8 & u_1^8 & v_1^8 & 1 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \end{pmatrix} = 0 \tag{20}$$

Once the essential matrix E is estimated, the next step is to recover the camera's movement.

This process is obtained by singular value decomposition (SVD). Let the SVD decomposition of E be:

$$E = U\Sigma V^T \tag{21}$$

The rotation matrix R and translation vector t can be obtained by the following equations.

$$\begin{aligned} t_1^\wedge &= UR_Z\left(\frac{\pi}{2}\right)\Sigma U^T, \quad R_1 = UR_Z^T\left(\frac{\pi}{2}\right)V^T \\ t_2^\wedge &= UR_Z\left(-\frac{\pi}{2}\right)\Sigma U^T, \quad R_2 = UR_Z^T\left(-\frac{\pi}{2}\right)V^T. \end{aligned} \tag{22}$$

Notice that there are 4 possible pairs of R and t. Then the positive depth condition is used to validate R and t. Only the R and t pair that makes both depths in the 2 camera-fixed coordinate system positive is chosen.

To avoid the effects of degradation and noise, homograph matrix H and essential matrix E are usually estimated simultaneously. Then the initial motion of the camera is recovered by the one with a smaller reprojection error.

Monocular camera poses tracking

Once the monocular camera SLAM is initialized, the existing 3D points can track new frame poses. This process solves the 2D-3D problem, which is known as PnP problem. It describes estimating the camera pose when the n 3D points and their projection are known. Unlike the previous 2D-2D problem, it only needs 3 points at least to estimate the camera pose. There are many ways to solve PnP problem. This work uses non-linear optimization to construct and iteratively solve a least-square problem, usually known as the bundle adjustment (BA) approach.

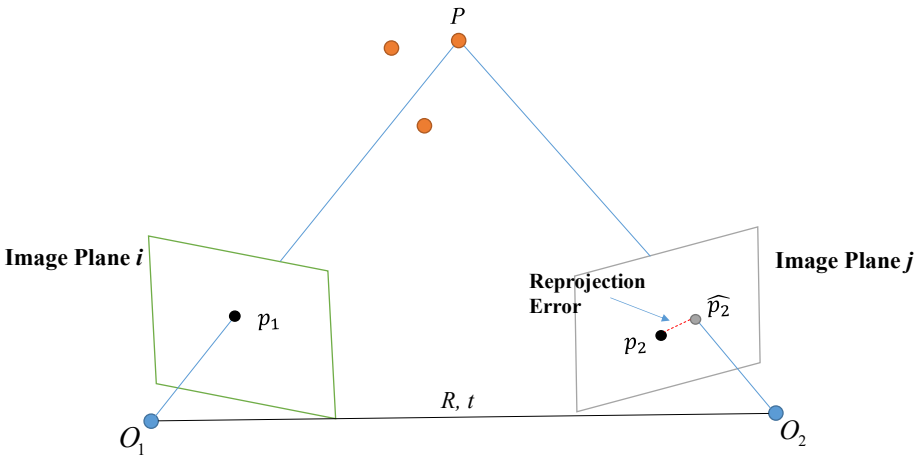


Figure 4.5: The reprojection error.

Consider that there are n 3D points P in the space, and their projection position on the image plane is p. The objective is to calculate the camera pose R and t by optimization. Suppose the coordinates of a point is $P_i = [X_i, Y_i, Z_i]^T$ and their projected pixel coordinates are $p_i = [u_i, v_i]^T$. According to the pin-hole projection model, the relationship between the 2D pixel position and the 3D position is:

$$s_i p_i = KTP_i \tag{23}$$

or in the matrix form:

$$s_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = KT \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{24}$$

where K is the camera intrinsic matrix, and T is the camera pose in the form of Lie linear algebra:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, T = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \tag{25}$$

This equation includes a conversion from homogeneous coordinates to non-homogeneous coordinates implicitly. Due to the camera poses and noise of the observation points, there is always a

residual in the equation. Therefore, summing up the residuals can construct a least-square problem and then minimize it to find the most possible camera pose and 3D points' coordinates:

$$P_i^*, T^* = \arg\min \frac{1}{2} \sum_{i=1}^{n} ||p_i - \frac{1}{s_i} K T P_i|| \tag{26}$$

This residual term is the error of the projected position and the observed position, which is called the reprojection error. This residual has the dimension of 2.

After bundle adjustment (BA) optimization, the camera poses and 3D points are both refined.

Scale correction

Visible camera intrinsic parameters are estimated with the camera calibration method proposed in [120]. Although ORB-SLAM2 estimates camera pose for each frame, only those with a proper parallax of the potential fire point are chosen to participate in fire spot depth estimation. That is for suppressing the noise generated by the insufficient baseline.

The gimbal angle is set fixed during each depth estimation process, thus the translation vector of the body-fixed coordinate system is equal to that of the camera-fixed coordinate system. Then the translation provided by the UAV the platform can be used to correct the scale estimate by ORB-SLAM2 as Eq. (27).

$$\begin{cases} s_i = & \sqrt{||t_b^i - t_b^{ref}||^2} \\ t_c^{*i} = & s_i \dfrac{t_c^i - t_c^{ref}}{\sqrt{||t_c^i - t_c^{ref}||^2}} \end{cases} \tag{27}$$

where s_i is the real-world scale of the ith frame; t is the translation vector, superscript ref, i and subscript c, b denote the frame index and coordinate systems respectively. And t_c^{*i} is the scale of the ith frame in the camera-fixed coordinate system.

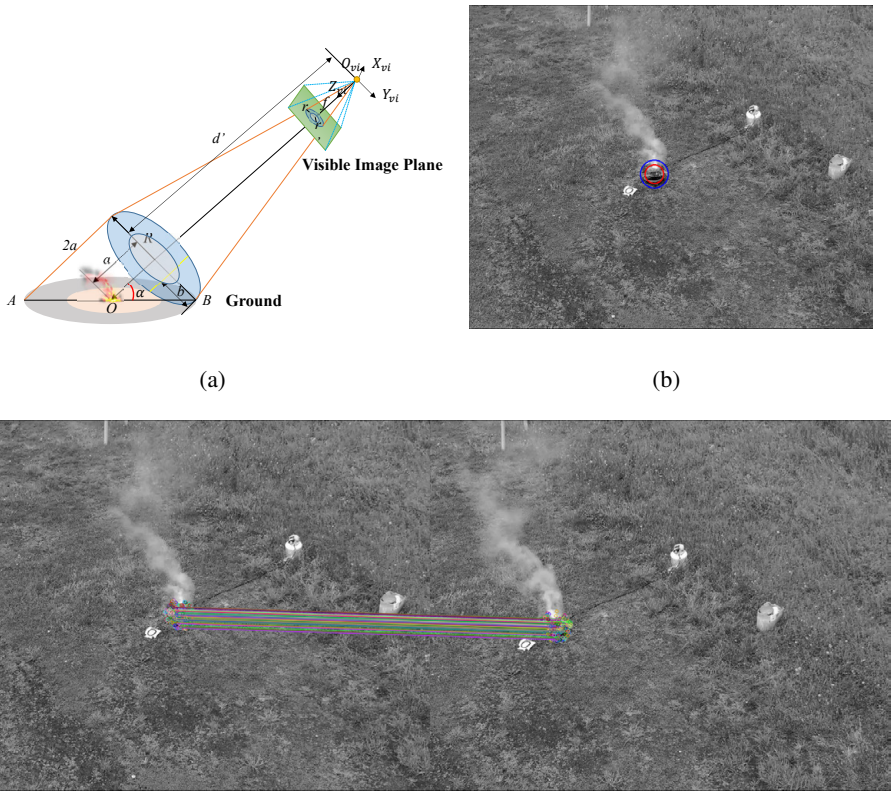
4.2.2 Depth estimation and geolocation

Feature matching

To make the estimation more robust, instead of directly estimating the depth of the centre of the fire spot, the depth value around a suspect fire point is used to represent the depth of the fire point. First, the contour of the potential fire point is calculated, and a circle containing the contour with the minimum area is then fitted. As shown in Fig. 4.6(a), the strategy for choosing a region of interest (ROI) is designed as follows:

- The center of ROI is the same as the minimum-area circle.
- The circle patch area is 1.5 times the minimum-area circle within a given threshold. The threshold is adjusted to limit the ROI area.

Then, ORB features [65] are extracted evenly in that patch. Features matching algorithm and histogram filtering are implemented to find corresponding reliable points between the reference frame and each additional frame. Finally, these matched points are triangulated into 3D points with the camera intrinsic parameters and frame poses, which is estimated and corrected in Section 4.2.1.



(a) (b)

(c)

Figure 4.6: Depth estimation feature extracting and matching: (a) Projection relationship of the proposed depth estimation strategy. Red circle in the middle: minimum-area circle containing fire point; blue circle: the circle area where the matched points are found. (b) ROI based on the U-net segmentation: red circle: minimum-area circle containing fire points; blue circle: designed patches. (c) ORB features extracted and matching results: left image is reference frame; right image is the 34th frame.

Triangulation

The poses of a sequence of images and the matching relationship of the features are both known. Triangulation solves the problem: "to recover the 3D points from the 2D points on the images".

As shown in Fig.4.4, the 3D point P is projected on the 2 images. The feature points on the image plane i and j are p_1 and p_2 respectively. To simplify the derivation, the normalized plane is introduced. The z coordinate of the normalized plane in its camera-fixed coordinates system always equals 1. The points x_1 on the normalized plane share the same projection location with the 3D point P. So the projection of P can be substituted by the x_1 on the normalized plane. x_2 on the normalized plane j has the same properties. s_1 and s_2 are the depth to be calculated by triangulation.

According to the projection relationship, x_1 on normalized plane satisfies the following constraints:

$$s_1 \underbrace{\begin{bmatrix} x^1 \\ y^1 \\ 1 \end{bmatrix}_C}_{x_1} = \underbrace{\begin{bmatrix} R & t \end{bmatrix}}_{T_1} \underbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{P^W} \tag{28}$$

where the $T_1 \in \mathbb{R}^{3\times4}$ is the translation matrix between the world-fixed and camera-fixed coordinate systems. $P^W \in \mathbb{R}^{4\times1}$ is the homogeneous coordinate of 3D point P in the world-fixed coordinate system. Now let v_1 substitute the right side and use the row vector to represent T_1:

$$v = \underbrace{\begin{bmatrix} R & t \end{bmatrix}_{3\times4}}_{T_1} \underbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{P^W} = \begin{bmatrix} t_0 P^W \\ t_1 P^W \\ t_2 P^W \end{bmatrix} \tag{29}$$

Notice that the vector x_1 and v_1 are parallel now. The cross-product of the 2 vectors should equal zero.

$$\begin{cases} yt_2 P^W - t_1 P^W = 0 \\ -xt_2^P W + t_0 PW = 0 \end{cases} \tag{30}$$

Similarly, projection on the images j shares the same relationship. The equations of the left and right images can be compacted to the following $Ax = 0$ problem:

$$\begin{bmatrix} yt_2 - t_1 \\ t_0 - xp_2 \\ y't_2' - t_1' \\ t_0' - x'p_2' \end{bmatrix} P^W = 0 \tag{31}$$

SVD decomposition can solve this equation. In real-world implementation, C++ library Eigen

[121] and Sophus [122] are used for the formula.

Depth filtering

As shown in Fig. 4.6(a), A and B are the 3D points with the maximum and minimum depth values in the estimated zone, respectively. Valid depth range can be expressed as Eq. (32):

$$\begin{cases} d_A = d'(1 + \dfrac{2r}{f \tan \alpha}) \\ d_O = d'(1 + \dfrac{r}{f \tan \alpha}) \\ d_B = d' \end{cases} \tag{32}$$

The points along the bright yellow line in the blue circle plane are shown in Fig. 4.6(a) have the same depth value. It can be implied that the amount of points with the same depth value is proportional to the yellow line length because the feature points in the patch area are extracted evenly.

To cull the outliers among the estimations, both valid depth range and three-sigma limits are implemented. The process to estimate and filter the depth can be summarised as Algo. 1.

Algorithm 1 Depth estimation outliers culling.

Require: Visible camera focal length f; gimbal pitch angle α; depth estimation values set $d = \{d_1, d_2, \cdots, d_n\}$ generated from triangulation.

Ensure: The depth estimation values after culling.

1: Find the overall median as initial d_O.

2: Calculate the upper boundary d_{tu} and lower boundary d_{tl} of the valid depth values according to Eq. (32).

3: Calculate the standard deviation σ and calculate the 3σ boundaries $d_{\sigma u}$ and $d_{\sigma l}$.

4: Update the upper and lower boundary: $d_u = \max(d_{tu}, d_{\sigma u}), d_l = \min(d_{tl}, d_{\sigma l})$. Then cull the estimation outside of the boundaries.

5: Determine if the depth is converged. If converged, stop and return; otherwise go to step 1.

6: **return** estimated depth

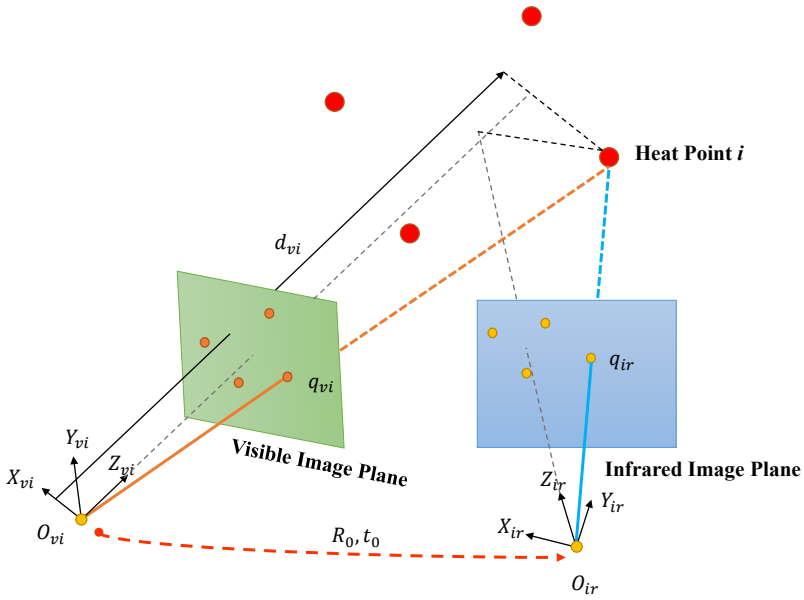


Figure 4.7: Geometry relation of visible and infrared cameras when observing the same heat points.

Depth-based wildfire geolocation

Once the depth is estimated, the geolocation of the wildfire spot can be derived from the geometry relationship with the gimbal angle.

As shown in Fig.4.2, the depth is now estimated. With the gimbal the angle of the H20T camera, we can directly use the depth to generate the GPS location with Eq. (9) and (11).

4.3 Model-based infrared-visible images registration

As shown in Fig.4.7, the transformation between visible and infrared cameras can be expressed by rotation R and translation t, known as camera extrinsic parameters. And the estimated depth in visible images is denoted as d_{vi}. A heat point i, whose position in camera coordinates are p_{vi} and p_{ir}, is projected into the two cameras. And the pixel values in image coordinates are denoted as q_{vi} and q_{ir}. This projection process can be expressed by Eq. (33):

$$p = dK^{-1}q, p = \begin{bmatrix} x \\ y \\ d \end{bmatrix}, q = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{33}$$

where K represents the camera intrinsic matrix (visible or infrared camera), and d is the depth of the 3D point i.

The same 3D point i projected in the infrared camera coordinate system and visible camera coordinate system satisfies the following equation:

$$p_{ir} = Rp_{vi} + t \tag{34}$$

Substitute Eq. (33) into Eq. (34):

$$q_{ir} = \frac{d_{vi}}{d_{ir}} K_{ir} R K_{vi}^{-1} q_{vi} + \frac{1}{d_{ir}} K_{ir} t \tag{35}$$

As shown in Fig.1.6, under the normal situation, the distance between d_{ir} and d_{vi} is much less than that between the drone and the fire points. Thus, $\frac{d_{vi}}{d_{ir}} \approx 1$. Then Eq. (35) can be simplified to:

$$q_{ir} = K_{ir} R K_{vi}^{-1} q_{vi} + \frac{1}{d} K_{ir} t \tag{36}$$

where d is the depth of the early forest fire or heat point. However, it is unfeasible to manually calibrate the extrinsic parameters (R and t) and infrared camera intrinsic parameter (K_{ir}) for the DJI H20T camera.

R, t K_{vi} and K_{ir} are the camera-based constants. To estimate the unknown registration parameters, Eq. (36) can be rewritten to:

$$q_{ir} = R' q_{vi} + \frac{1}{d} t', R' \in \mathbb{R}^{3\times3}, t' \in \mathbb{R}^{3\times1} \tag{37}$$

where $R' = K_{ir} R K_{vi}^{-1}$ and $t' = K_{ir} t$ denotes the unknown parameters of the two cameras, which could be estimated by the least squares method. The following shows estimation process:

Change Eq. (37) in the form of matrix:

$$\begin{bmatrix} u_{ir} \\ v_{ir} \\ 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} u_{vi} \\ v_{vi} \\ 1 \end{bmatrix} + \frac{1}{d} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \tag{38}$$

One pair of matched points provides two constraints. Thus, to estimate the R' and t', at least six pairs of matched points are needed. When the number of measurements is more than six at different depths, a linear least-squares problem can be constructed. For the convenience of programming, parameters to be estimated can be written into Eq. (39).

$$x = \{R_{11}, R_{12}, R_{13}, \cdots, R_{32}, R_{33}, t_1, t_2, t_3\} \in \mathbb{R}^{12} \tag{39}$$

If there are n observations, the cost function is:

$$f(x) = \sum_{j=1}^{n} ||q_{ir}^{j} - (R' q_{vi}^{j} + \frac{1}{d} t')||^2, j = \{1, 2, \cdots, n\} \tag{40}$$

The whole problem can be expressed as:

$$x^* = arg \min_{x} f(x) \tag{41}$$

After the transformation is estimated, the corresponding wildfire points can be easily located in infrared images according to Eq. (37).

Chapter 5

Early Wildfire Spot Local Environment 3D Reconstruction

In this work, 3D reconstruction takes the reference to the most popular multi-view geometry based reconstruction method: First structure from motion (SfM) is used to recover the camera poses and sparse 3D points. Then with the multi-view stereo (MVS) system to densify the sparse points cloud, reconstruct the mesh and texture mesh.

Feature extraction and feature matching are important for SfM because they enable accurate and robust reconstruction of 3-D scenes from multiple 2-D images. Feature extraction helps to reduce the dimensionality of the image data and to find correspondences between images. Feature matching helps to establish geometric constraints between images and to estimate camera poses and scene structure.

SfM is used to recover the camera poses rather than the SLAM system because of the precision. Compared with the SLAM system, SfM is used after the image-capturing process, thus, the global bundle adjustment (GBA) can be conducted. Because of the real-time properties of the SLAM system, the scale of its bundle adjustment problem is usually limited. Thus, methods like “marginalization” or “co-visibility graph” that comprise precision and real-time properties are used. However, SfM systems are offline, which can usually conduct GBA multiple times. So the SfM-based camera and sparse 3D points recovery are used.

5.1 Feature extracting and matching

5.1.1 Scale space

The basic idea of scale space is: the Gaussian kernel is the only kernel that can generate multi-scale space. In the input image model, the parameters of the scale are continuously transformed by the Gaussian blur function, and finally a multi-scale space sequence is obtained. The spatial function $L(x, y, \sigma)$ of a certain scale in the image is obtained by convolution of the Gaussian function $G(x, y, \sigma)$ with variable parameters and the original input image $I(x, y)$. This image sequence is known as Gaussian pyramid.

$$\begin{aligned} G\left(x_i, y_i, \sigma\right) &= \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(x - x_i)^2 + (y - y_i)^2}{2\sigma^2}\right) \\ L(x, y, \sigma) &= G(x, y, \sigma) * I(x, y) \end{aligned} \tag{42}$$

where σ is known as a scale space factor. Smaller σ represents a less blurred image.

An image pyramid is a set of results obtained from the same image at different resolutions. The generation process consists of two steps:

- Gaussian smoothing (Gaussian blur) is performed on the image.
- Downsampling is performed on the image, and after dimensionality reduction sampling, a series of images with continuously reduced sizes are obtained.

The traditional SIFT algorithm extracts feature points by establishing the difference of Gaussian (DoG) function method. First, the Gaussian difference image is obtained from the difference of neighboring images at the same scale layer in a group of different scale parameters. Then, the obtained difference image is convolved with the original image $I(x, y)$ to obtain the DoG function of Eq. (43).

$$\begin{aligned} D(x, y, \sigma) &= [G(x, y, k\sigma) - G(x, y, \sigma)] * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{43}$$

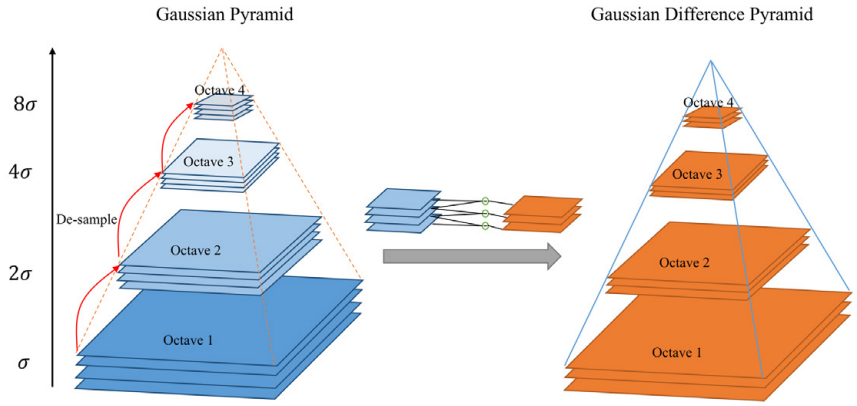


Figure 5.1: Gaussian pyramid: The original image are first convolved by Gaussian kernel, then de-sampled to be the pyramid. The Gaussian difference pyramid is generated by the difference of the Gaussian pyramid.

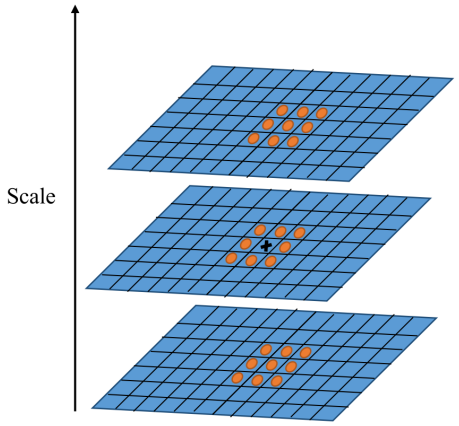


Figure 5.2: Feature point (the black cross) is detected, which considers the neighbor and scale information.

5.1.2 Feature point detection

Feature points are composed of local extremal points in the DoG space. To find the extreme value points of the DoG function, each pixel point is compared with all its neighbors to see if it is larger or smaller than its neighbors in the image and scale domains.

As shown in Fig. 5.2, the middle detection point is compared with its 8 neighboring points at the same scale and 9×2 points corresponding to the upper and lower neighboring scales for a total of 26 points to ensure that the extreme value points are detected in both scale space and two-dimensional image space. Some extreme value points are located at the edge position of the image, because the edge points of the image are difficult to locate and also easily disturbed by noise, we regard these points as unstable extreme value points and need to be removed. The principal curvature values are

larger in the direction of the edge gradient, while they are smaller along the edge direction. The principal curvature of the DoG function $D(x)$ of the candidate feature points is proportional to the eigenvalues of the 2×2 Hessian matrix H, as in Eq. (44). D_{xx} denotes the image x-direction derived twice for a certain scale in the DoG pyramid.

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \tag{44}$$

Let $\alpha = D_{xx}$ be the maximum eigenvalue of H, $\beta = D_{yy}$ be the minimum eigenvalue of H. Then we have:

$$\begin{aligned} \frac{Tr(H)^2}{Det(H)} &= \frac{(\alpha+\beta)^2}{\alpha\beta} \\ det(H) &= \alpha\beta \\ Tr(H) &= \alpha+\beta \end{aligned} \tag{45}$$

where $Tr(H)$ is the trace of matrix H, $Det(H)$ is the determinant of matrix H. When two eigenvalues are equal their values are minimum, so in order to detect whether the principal curvature is under some threshold Tr, it is only necessary to detect the relationship between the magnitude of the ratio and the threshold T and filter the unstable edge response points.

Therefore, feature point extraction can be summarized as the following steps:

- Construct a Gaussian scale space to generate Gaussian blurred images of different scales.
- Performing downscaled adoption to obtain a series of images with decreasing size.
- DoG space extreme value detection to remove some edge response points.

5.1.3 Descriptor calculation and feature matching

After the above steps, the feature points that exist at different scales have been found. In order to achieve the rotation invariance, it is necessary to assign values to the directions of the feature points. The gradient of the pixels in the neighborhood of the feature point is used to determine its orientation parameter, and then the gradient histogram of the image is used to find the stable orientation of the local structure of the key point.

For the feature point that has been detected, we can obtain the corresponding σ, which determines the Gaussian image at that scale. To determine the orientation of the feature point, a gradient histogram statistic is used to count the contribution of the image pixel points in a certain area to the generation of the feature point orientation.

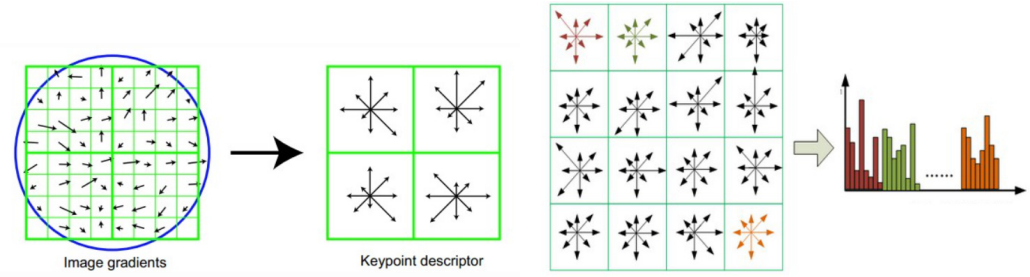
With the above steps, for each feature point, there are three types of information: location, scale, and orientation. The next step is to create a descriptor for each feature point so that it does not change with various changes, such as illumination changes, viewpoint changes, etc. And the descriptor should have a high degree of uniqueness in order to improve the probability of correct matching. And the descriptor should have high uniqueness in order to increase the probability of correct matching of feature points.

As shown in Fig. 5.3(a), the center of the left image is the position of the current feature point. Each cell represents a pixel in the scale space where the neighborhood of the feature point is located. The direction of the arrow represents the gradient direction of the pixel, the length represents the gradient magnitude, and then use the Gaussian window to weight the operation. Finally, the gradient histogram of 8 directions is drawn on each $3\sigma \times 3\sigma$ block (the side length of each square area in the image is 4), and the cumulative value of each gradient direction is calculated to form a seed point, as shown in the right figure. Each feature point consists of 4 seed points, and each seed point has vector information in 8 directions. This joint neighborhood directional information enhances the noise resistance of the algorithm and also provides more rational fault tolerance for feature matching containing localization errors.

As shown in Fig. 5.3(b) [63], in the actual computation, in order to enhance the robustness of matching, it is suggested to use 4×4 for each key point to describe a total of 16 seed points, so that one key point can generate a 128-dimensional SIFT feature vector.

To match the features in different images, the similarity is calculated to find the best matched points, which is evaluated by Euclidean distance as in Eq. (46).

$$\begin{aligned} R_i &= (r_{i1}, r_{i2}, \cdots, r_{i128}) \\ S_i &= (s_{i1}, s_{i2}, \cdots, s_{i128}) \\ d(R_i, S_i) &= \sqrt{\sum_{j=1}^{128} (r_{ij} - s_{ij})^2} \end{aligned} \tag{46}$$



(a) (b)

Figure 5.3: The process to generate the SIFT descriptor: (a) The orientation of each $3\sigma \times 3\sigma$ grid, which has 8 direction information. (b) Real-world implementation using 4×4 grids to generate the descriptor.

where R_i is the descriptor of ith feature in the original image, and S_i is in the target image. The similarity d is the Euclidean distance between R_i and S_i.

The final chosen feature should satisfy Eq. (47).

$$\frac{d(R_i, S_j)}{d(R_i, S_p)} < \mu \tag{47}$$

where S_j and S_p are the minimum and second minimum distance in the target image. μ is a given threshold.

5.2 Global bundle adjustment (GBA)

The mathematical principle of BA is introduced in Section4.2. However, in this Section of SfM, there are three main differences from the previous BA in SLAM system:

- Camera intrinsic matrix K is optimized as an estimated parameter.
- The BA here is always for the global scope.
- The GNSS information is used to constrain the GBA as prior position.

The IMU and GNSS information is fused with EKF to provide the UAV's attitude and position information. Compared with the pure vision-based SfM, the fused position can provide the actual scale of the camera trajectory and the 3D scene.

As shown in Fig.5.4, several images are captured around the scene. There are several landmarks observed by those frames. As described in Section 4.2, reprojection error is generated from the estimated landmarks projected onto the frame, which is a 2-dimensional vector.

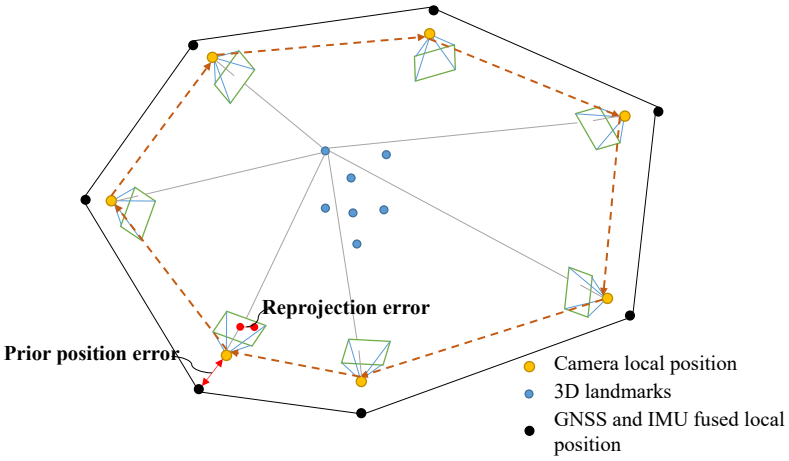


Figure 5.4: Several images are captured from a circular trajectory. The black points are the local position in NED fused by the IMU and GNSS. While in the optimization process, the vision-based SfM can not recover the correct scale of the 3D scene. However, with the constraints from the prior position, the translation vector can be constrained.

The reason for using the fused local position rather than the global position is the noise of the GNSS system. The global position is used to geolocate the wildfire spot after finishing the whole scale correction. Considering the frame i, there are j observations of the landmarks to form the cost function. The two types of residual in the cost function are:

- Reprojection residual error e^v:

$$e_{i,j}^{v} = p_{i,j} - K_i(R_i P_j + t_i) \tag{48}$$

- Prior position residual error e^p:

$$e_i^p = z_i - t_i \tag{49}$$

where $p_{i,j}$ is the observation for landmark j on image i. P_i is the coordinates value of landmark j in the world-fixed system. The rotation and translation are explicitly represented by R_i and t_i for the ith frame. z_i is the fused prior position of the i frame.

Accumulate all of the observations, the final cost function is:

$$
\begin{aligned}
f(R,t,P,K) &= \sum_{i=1}^{m}\sum_{j=1}^{n} ||e_{i,j}^{v}||^2 + \omega \sum_{i=1}^{m} ||e_i^p||^2 \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} ||p_{i,j} - K_i(R_i P_j + t_i)||^2 + \omega \sum_{i=1}^{m} ||z_i - t_i||^2
\end{aligned}
\tag{50}
$$

where ω is a tunable constant to control the ratio between the two types of the residuals.

Eq. (50) indicates a nonlinear least squares problem. The parameters to be optimized are the camera poses R, t, 3D points position P, and camera intrinsic matrix K. In this work, ceres [123] is used to define and solve the problem. The great advantage of the ceres library is the automatic derivation. There is no need to define the derivative of Eq. (50) manually.

5.3 Multi-view stereo (MVS) reconstruction

In this work, the MVS algorithm is used as the "post-processing" of the SfM. After the poses and a sparse point cloud is recovered by the SfM system. The output can be used in the following MVS system to generate the surface information of the local environment scene.

The open-source OpenMVS [100] system is used here. OpenMVS is a relatively classic MVS open source library, which integrates the entire technical solution of 3D reconstruction (camera model, multi-view stereo geometry, dense reconstruction, surface reconstruction, point cloud fusion, texture mapping).

The process of the OpenMVS-based reconstruction involves several steps:

- Scene Reconstruction: This step uses a global formulation of the depth-map estimation problem that minimizes a cost function based on photo-consistency and smoothness constraints. The cost function is optimized using a PatchMatch-based algorithm that iteratively updates the depth values by random sampling and propagation. The output is a set of depth-maps, one for each input image, that represent the distance from the camera to the scene surface for each pixel.

- Densify Point Cloud: This step converts the depth-maps into a point-cloud by projecting each

pixel with a valid depth value into 3D space using the camera parameters. The point-cloud is then filtered to remove outliers and redundant points using a visibility test that checks if a point is occluded by other points from multiple viewpoints. The output is a dense and accurate point-cloud that covers most of the scene surface.

- Reconstruct Mesh: This step uses an implicit surface representation based on signed distance functions (SDF) to approximate the geometry of the point-cloud. The SDF values are computed by fitting radial basis functions (RBF) to the point-cloud using an adaptive octree structure. The mesh surface is then extracted from the SDF using an efficient marching cubes algorithm that preserves sharp features and topology. The output is a watertight mesh surface that approximates the point-cloud well.

- Refine Mesh: This step improves the quality of the mesh surface by applying several operations: noise removal, hole filling, sharp feature preservation, and non-rigid deformation. Noise removal and hole filling are done by applying Laplacian smoothing and Poisson reconstruction respectively to the mesh surface. Sharp feature preservation is done by detecting and preserving edges and corners using curvature analysis and normal voting. Non-rigid deformation is done by minimizing an energy function that balances data fidelity (alignment with images) and regularization (smoothness and rigidity) terms using an as-rigid-as-possible (ARAP) framework. The output is a refined mesh surface that captures fine details and aligns well with images.

- Texture Mesh: This step computes a texture atlas for coloring the mesh surface by projecting and blending images on it. The texture atlas consists of a set of charts (patches) that map regions of the mesh surface to regions of texture images. The charts are generated by partitioning the mesh surface into patches using a greedy algorithm that minimizes distortion and boundary length. The patches are then packed into texture images using an optimization algorithm that maximizes resolution and minimizes empty space. The colors for each patch are computed by blending images on it using a visibility-consistent seam leveling algorithm that reduces color inconsistencies along patch boundaries.

Chapter 6

Experiment Design and Results

In this Chapter, the experiments platform with M300 UAV and DJI H20T camera is firstly introduced. The hardware and software environment are both demonstrated. Then three experiments are designed to validate the proposed framework's detection, geolocation and reconstruction modules. These outdoor experiments are performed in a flight club[1] airport in Montreal, Canada.

The first experiment is for the detection stage: UAV flies along a zigzag path to cover a given zone and detect the suspected wildfire spot. The second experiment consists of indoor and outdoor experiments: for the indoor experiment, infrared-visible registration parameters are estimated; in the outdoor experiment, UAV will fly laterally to estimate the geolocation of the wildfire spot. In the meantime, registration for the wildfire spot will be performed. In the third experiment, UAV will fly around the wildfire spot to capture the higher resolution images for the SfM-MVS system. Finally, the 3D local environment is reconstructed on the ground computer. The trajectories and experiments results are demonstrated in the following three sections.

6.1 Experiment platform

6.1.1 Hardware

ZenMuse H20T camera is used as the on-board sensors. It embeds with a wide camera, a zoom camera, an infrared camera, and a laser rangefinder in one payload, which significantly simplifies

[1] https://www.wimac.ca/

Figure 6.1: ZenMuse H20T camera and DJI M300 UAV: (a) ZenMuse H20T camera. (b) DJI M300 UAV with multiple payloads.

the payload mounting and improves the system reliability. DJI M300 UAV is used as the carrier platform. The UAV platform hardware is shown Fig.6.1.

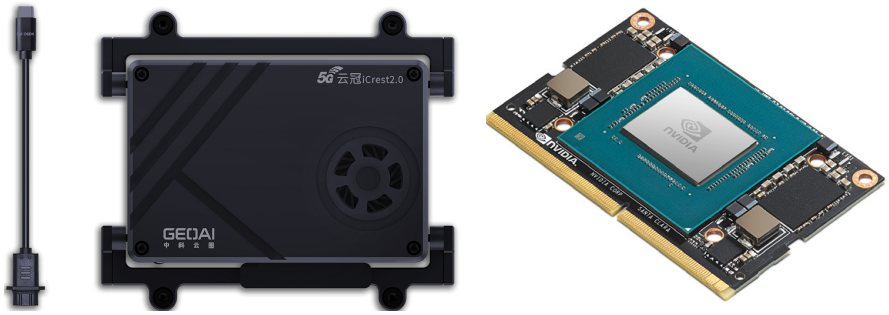
The on-board computer is the computational unit for real-time high-level mission control, the hardware and software device for real-time execution of the image processing and motion control algorithms for a mission. It equips with GPU(s) to accelerate deep learning-based image processing inference under limited power. Thus, the iCrest2.0 on-board computer provided by GEOAI and NVIDIA is chosen for this work. This on-board computer is based on the NVIDIA Jetson Xavier NX, which can bring up to 21 TOPs accelerated computing power to run modern neural networks in parallel and process data from multiple high-resolution sensors. With the development of GEOAI, iCrest2.0 can communicate with DJI M300 and H20T camera online.

The ground station computer should be able to monitor DJI M300 mission states and accelerate a more complex image segmentation algorithm. For the outdoor experiment, we used Lenovo IdeaPad 15 with Intel i5-6300HQ CPU and NVIDIA GeForce GTX 950M GPU.

The communication between the on-board computer and the ground station is through Wi-Fi. Wireless cards are equipped both on the two computers, and they both connected to a same local area network (LAN) generated by a router.

6.1.2 Software

Software is developed based on the robot operating system (ROS) [124] and DJI on-board software development kit (DJI-OSDK) [125], which provides communication between our software



(a) (b)

Figure 6.2: The iCrest2.0 onboard computer and NVIDIA Jetson Xavier NX: (a) iCrest2.0. (b) Jetson Xavier NX.

applications and hardware abstraction. As shown in Fig. 6.3, sub-modules are designed to work on the on-board computer and the ground station separately.

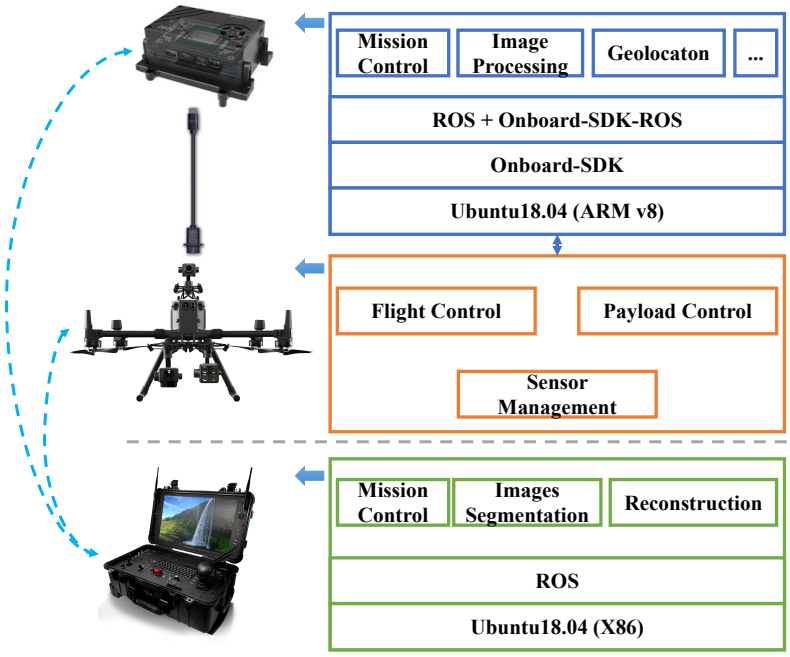


Figure 6.3: Configuration and functions of the proposed system framework.

With the application programming interfaces (APIs) provided by the DJI-OSDK, applications running on iCrest2.0 can access the real-time data of sensors and send control commands to the flight control and payload control modules (such as gimbal driver, etc.) via a wired connection. A task manager, image segmentation applications, and 3D reconstruction applications run simultaneously

on the ground station computer. The former application is designed to monitor and control the whole detection mission; the latter is an image segmentation application as a "completion" of the on-board computer.

The ROS supports the LAN connection naturally. Different nodes running on the different machines which are under a same LAN can communicate with each other if they are both registered to a common master node. In this work, the master node is running in the on-board computer. The reason here is the reliability: because of the limited connection ability of router. If the Wi-Fi connection is affected by the distance, master node running on the on-board computer can make sure at least the mission control and part of the image processing applications can run normally.

For the benefit of the real-time application, TensorRT, developed by NVIDIA, is used for optimizing and accelerating the deep learning-based algorithms inference [126]. According to the Pytroch and TensorRT implementation project website [127], ResNet-18 can achieve around 7 times faster after using the TensorRT on NVIDIA Xavier NX.

The proposed early wildfire detection system does not heavily rely on the hardware platform, which mainly benefits from the hardware abstraction of the ROS. Theoretically, the proposed algorithm and software can be embedded in any UAV platform, on-board sensors and computer with similar functions as we use.

In summary, the proposed early wildfire detection system takes the hardware and sensors limitation, algorithms false alarm, and computation power into consideration, achieving considerable robustness.

6.2 Early wildfire spot detection experiments

6.2.1 Flight trajectory

In the outdoor experiment, a synthetic fire point, which generates heat and smoke through burning leaves and wood is placed in front of the M300 UAV. After taking off, the system performs according to our proposed strategy. After the early forest fire is detected and confirmed, the system gives alarm. Finally, M300 returns home and finishes the mission. Fig. 6.4 shows the trajectory of

the M300 during the experiment. A video recording of this experiment can be found on YouTube.[1]

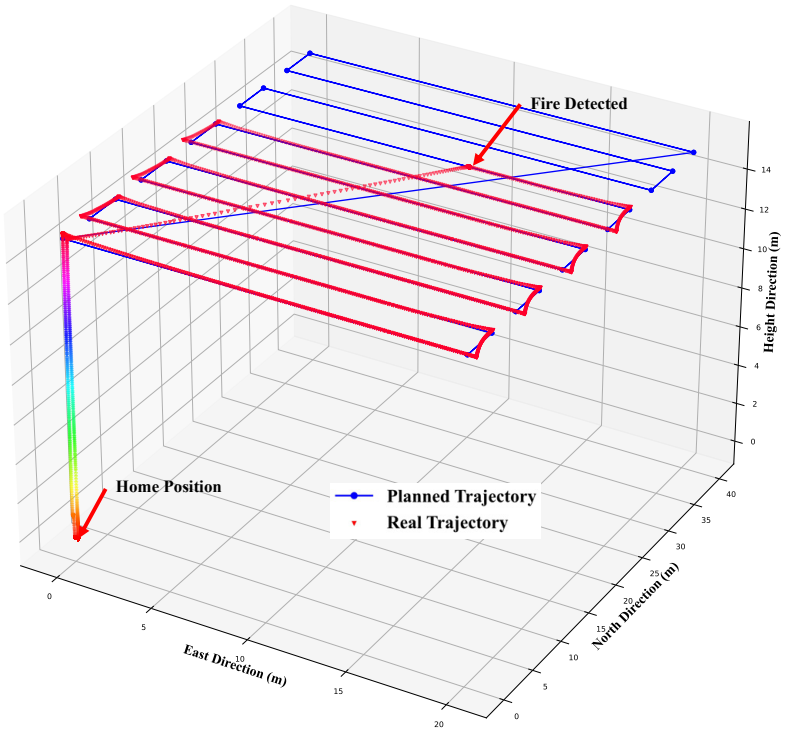


Figure 6.4: M300 planned trajectory and real flight test trajectory

It should be noted that the laser rangefinder can only be manipulated manually due to the insufficient APIs of DJI-OSDK. But except for that, all the other processes can be done automatically by the proposed system.

6.2.2 Gimbal control

Through ground testing, the control frequency of the DJI gimbal controller, as shown in Fig. 4.1 is $1 \sim 2$Hz. The main reason is that the gimbal control API in DJI OSDK is based on the ROS "service" communication method, which has a long delay time. Thus the proposed PD controller has a frequency of 1Hz.

In the controller software implementation, once the errors are less than the given thresholds, or it takes too long to control the gimbal, the controller will stop. The following Fig. 6.5 shows the control output and the pixel error along the yaw and pitch directions; After nine times of control,

[1] https://www.youtube.com/watch?v=dQG73LW8jxQ

the pixel errors along x and y directions both converge into the given boundary (20 pixels in the experiment).

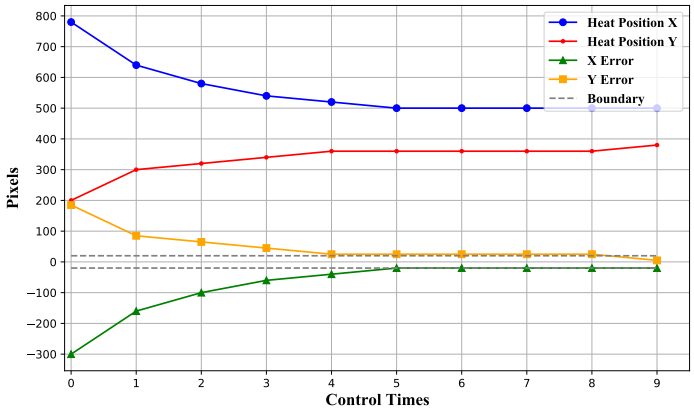


Figure 6.5: Heat point position and pixel errors: after errors converge into the given boundary, control commanding stops.

6.2.3 Infrared images segmentation and localization

RGB color space uses all three channels to express color features, which means using this model may be more challenging to segment heat points through threshold-based methods than HSV color space, which only uses the "H" channel for color. Under the "north pole isothermal model", images from H20T infrared camera change from blue to red with the temperature increases. All the three channels in RGB color space will change during the temperature changes, but only the H channel for HSV color space. For example, when the temperature is relatively low, the red channel can not be used to locate the heat point, as shown in Fig. 6.6:

(a) RGB camera (b) Infrared camera

(c) R (d) G (e) B

(f) H (g) S (h) V

Figure 6.6: HSV and RGB color space channels. The red circles indicate heat point. In this example, "R" channel fails to reflect the heat position.

Therefore, the H channel is used for the following processing, as shown in Fig. 6.7:

(a) H channel threshold segmentation (b) Morphology opening (c) Sliding window localization result (sliding window size: 40×40)

Figure 6.7: Heat point segmentation and localization process.

The location in the infrared image is used to rotate the gimbal.

6.2.4 ResNet-18-based classification

The training and validation datasets are collected from the internet as well as synthetic experiments pictures.

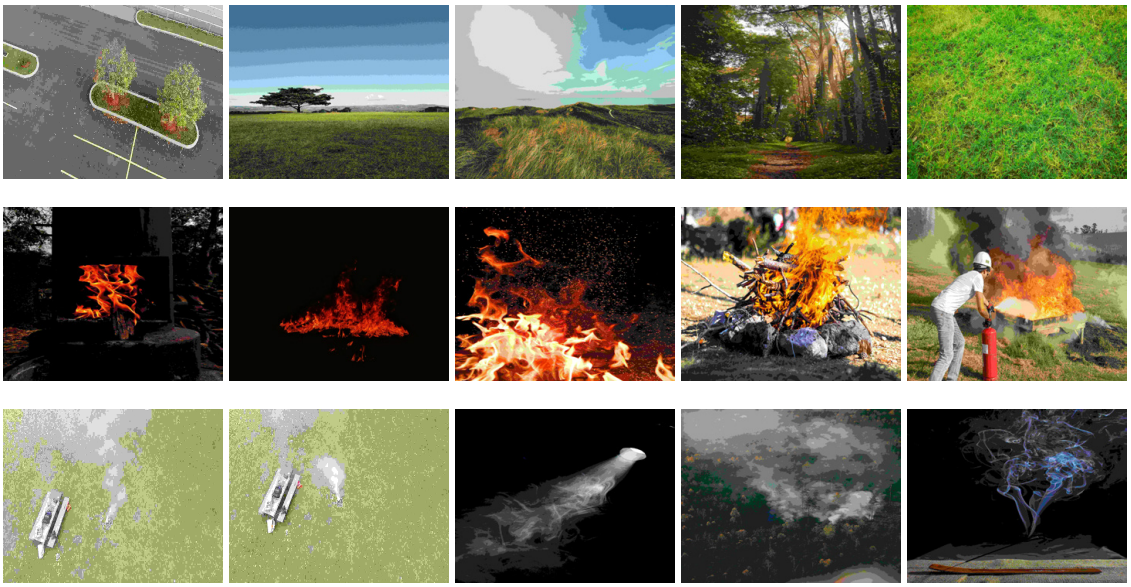


Figure 6.8: Examples of three different class datasets for training ResNet-18-based forest fire classifier. From top to bottom, the three rows of the examples belong to the "normal", "fire", and "smoke" classes respectively.

In the training process, the learning rate is set to 1×10^{-5}. The dataset with 500 images is resized to 255×255. 400 images are used for training. Both batch normalization and data augmentation

includes random flipping and rotating and are implemented. Other 100 images of a validation set are randomly picked from the entire 500 images dataset. After 20 epochs, the average validation accuracy of "fire", "smoke" and "normal" achieves over 82%. The following Fig. 6.9 shows training loss and accuracy during 50 epochs.

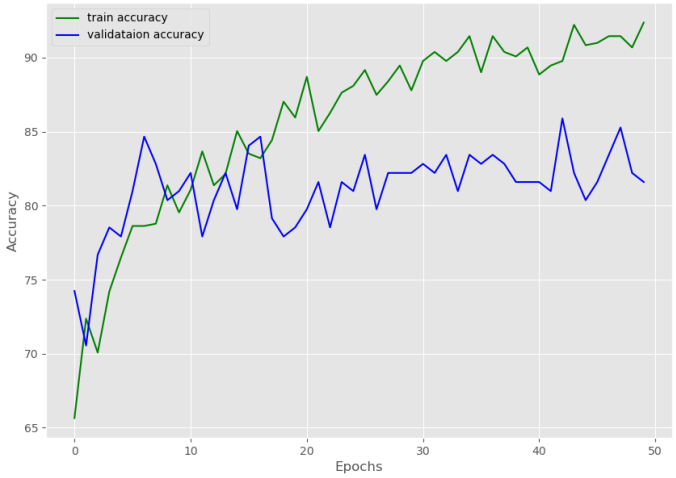


Figure 6.9: ResNet-18 training and validation accuracy during 50 epochs.

In the outdoor experiment, parameters trained from 20 epochs are used. Some example classification results are shown in Fig. 6.10.

Figure 6.10: ResNet-18 classification result examples.

U-net-based smoke segmentation

Similar to the ResNet-18, the training validation set for U-net is also collected from the internet and our synthetic data, which the DJI Phantom 4 Pro captures. The mask region is generated by Labelme [128] manually. Some example images and masks of the training and validation set are shown in Fig. 6.11.

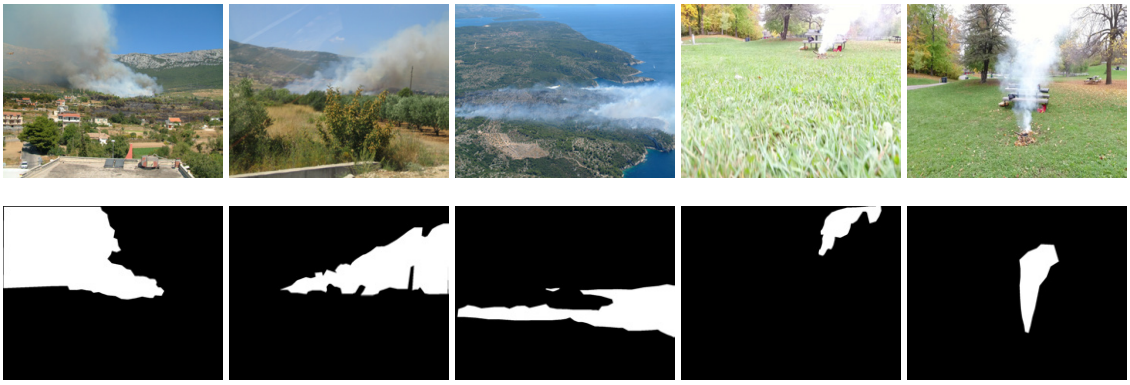


Figure 6.11: Examples of three different class datasets for training U-net-based forest fire smoke segmentation. The first row is the training image examples, and second row is the training mask labeled manually.

The U-net is trained on the ground station computer. During the training process, learning rate and batch size are set to 1×10^{-6} and 8 separately. In the pre-processing procedure, random image flipping, rotating and random brightness contrast adjustments are implemented before batch normalization and training. After 90 epochs, the accuracy achieves over 75% and changes slightly with any epoch increasing as Fig. 6.12 shows:

Figure 6.12: U-net training loss and accuracy during 120 epochs.

In the outdoor experiment, the frame rate of the online segmentation is set to 5Hz. Some segmentation result video frames are shown in Fig.6.13: The first row shows the original images with the mask (in translucent red); the second row shows the segmented smoke in white.

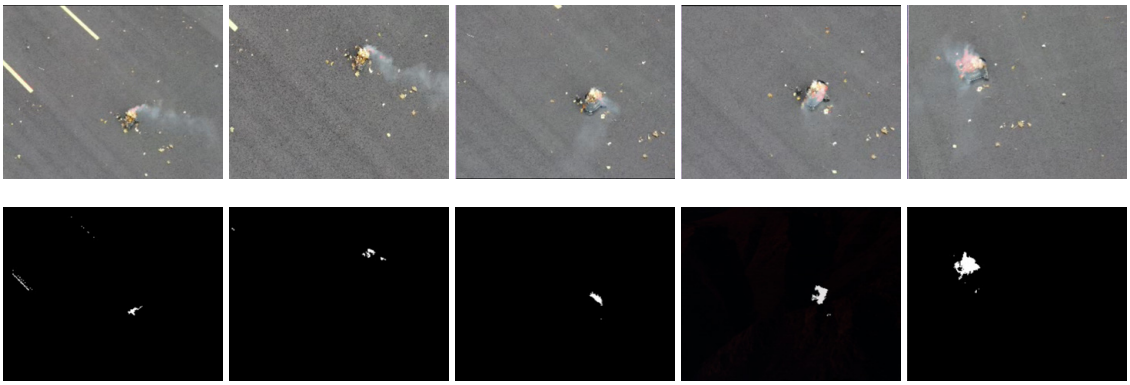


Figure 6.13: U-net smoke segmentation result examples. First row is the original image examples, the second row is the corresponding prediction generated by the U-net.

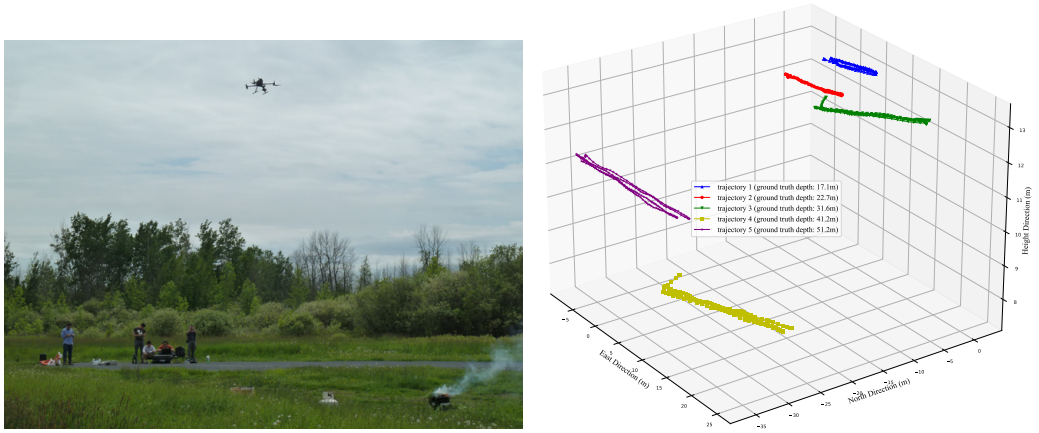
Once the smoke is segmented successfully, wildfire alarm will be sent to different terminals. And the next stage will be triggered.

6.3 Depth estimation experiment

6.3.1 Flight trajectory

Five independent outdoor experiments are implemented to validate the performance of the proposed framework. The scenario of each experiment is shown in Fig. 6.14. A synthetic wildfire point with flame and smoke is placed on the grass-covering ground to simulate a wildfire. The M300 UAV flies along the predefined trajectories in these experiments, in which the depth of wildfire is 17.1m, 22.4m, 31.6m, 41.2m, and 51.2m, respectively. A laser range finder measures the ground truth depth at each reference frame. At the same time, the H20T camera takes infrared and visible images at a frequency of 20Hz. The recorded UAV's real trajectories are shown in Fig. 4.3. The video to demonstrate this experiment can be found on YouTube.[1]

[1] https://www.youtube.com/watch?v=Sk7nWwYyFZI



(a) (b)

Figure 6.14: (a) A synthetic early wildfire spot is placed in front of the M300 UAV, while M300 flies horizontally to estimate the distance to the wildfire spot. (b) Five independent experiments with different trajectories are conducted.

6.3.2 Attention gate U-net segmentation

Hardware: Single GTX 1660 (6GB) GPU is used for training the attention gate U-net for wildfire segmentation.

Dataset: The dataset used in this work consists of two parts:

- Mixed dataset: consisting of 619 images mixed by the wildfire images from Kaggle [129] and experiment captured image. These images are used in the first training steps;
- Experiment dataset: based on the concept of transfer learning, 119 captured images by different distances in the designed experiment are used at the second training step.

All these images are labelled through the MATLAB tool (labelling) into three classes: background, smoke, and flame.

Evaluation: Pixel-level accuracy, ROC-AUC-score, macro F1-score, and average precision are selected as the evaluation metrics. Table6.1shows quantitatively the performance comparison between the original U-net and our model, attention gate U-net. And the items are explained as follows:

- Models: models "OU" and "AU" stands for original U-net and attention dataset (619 images), "2" means to train the model with an experiment dataset (119 images);
- lr: the statistics of appropriate learning rates found for Encoder part of the model and unfrozen entire model respectively;
- Epochs: statistics of the appropriate training epochs found through observing the changing of validation loss;
- Pixel-ACC: the statistics of pixel-level accuracy;
- ROC-AUC: the statistics of the scores of the area under the curve (AUC) of receiver operating characteristic (ROC);
- AP: the statistics of the scores of average precision, which summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold;
- Macro F1-score: the statistics of scores of the harmonic mean of the precision and recall, in which "Macro" means to calculate metrics for each label and find their unweighted mean.

Table 6.1: Quantitative performance comparison of original U-net with our work.

Models	lr	Epochs (Mixed dataset+ Experiment dataset)	Pixel-ACC (%)	ROC-AUC (%)	AP (%)	Macro F1-score
OU1	$4.36e^{-5}, 8.31e^{-6}$	$16 + 17$	96.748	91.874	76.799	62.562
AU1	$6.30e^{-5}, 8.31e^{-6}$	$17 + 6$	96.703	**92.343**	75.131	62.162
OU2	$3.63e^{-5}, 2.08e^{-5}$	$11 + 6$	99.419	84.192	53.734	57.139
AU2	$3.63e^{-5}, 9.99e^{-6}$	$10 + 4$	99.464	**89.341**	55.463	57.571

The "Pixel-ACC" shows that the attention gate U-net slightly improves the detection accuracy and the "ROC-AUC" shows that the model decreased the false positive obviously.

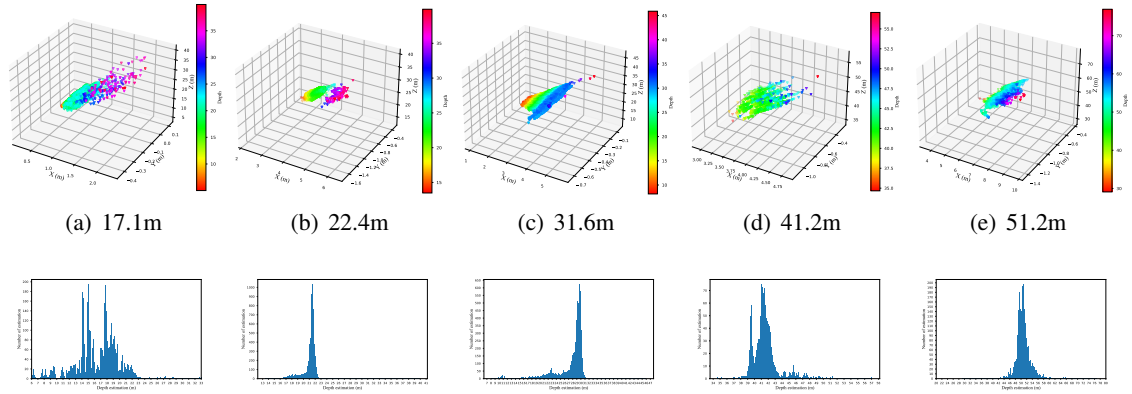
Fig.6.15shows some examples of the segmentation results. The segmentation performance is acceptable even as there are red clothes disturbances, which is shown in Fig. 6.15(e) and (j).

(a) o 17.1m (b) o 22.7m (c) o 31.6m (d) o 41.2m (e) o 51.2m

(f) p 17.1m (g) p 22.7m (h) p 31.6m (i) p 41.2m (j) p 51.2m

Figure 6.15: Examples of segmentation results of different depth. o: the images split from the original videos captured by DJI M300; p: attention gate U-net prediction results.

As stated in Section4.2, the depth of synthetic wildfire point observed in reference frames of 5 independent observations is estimated: The triangulated 3D points in the visible camera-fixed coordinate system and the depth distribution are shown in Fig.6.16 respectively. Ground truth (GT) depth, amount of frames, number of triangulated 3D points, median depth, estimated the depth and the errors are shown in Table 6.2.



(a) 17.1m (b) 22.4m (c) 31.6m (d) 41.2m (e) 51.2m

(f) 17.1m (g) 22.4m (h) 31.6m (i) 41.2m (j) 51.2m

Figure 6.16: First row: triangulated 3d points under visible camera-fixed coordinates system. Second row: depth distribution histogram.

In the second (GT: 22.4m) and third (GT: 31.6m) experiments, relatively looser criteria are used to match extracted ORB features, resulting in more matched features and triangulated 3D points.

Table 6.2: Depth estimation results of individual experiments.

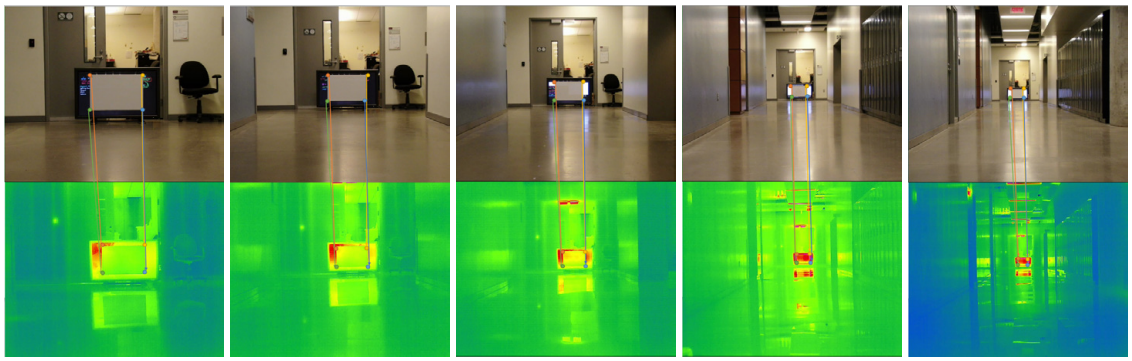
GT(m)	Frames	3D points	Median(m)	Depth(m)	Error(%)
17.1	424	6880	15.06	17.02	0.44
22.4	296	12616	21.42	21.46	5.78
31.6	647	12374	29.07	29.67	8.76
41.2	602	2424	40.93	41.26	0.15
51.2	733	5337	50.59	50.85	0.68

Simultaneously, more mismatched features are generated, increasing the estimation errors.

6.4 Visible-infrared images registration experiments

Parameters Estimation

To estimate the model parameters of the DJI H20T camera with the least square method proposed in Section 4.3, an experiment is designed as follows: A stable heating object (which in this work is a television with a whiteboard) is placed in front of the DJI H20T camera. The whiteboard makes the corners more distinct for manual matching. Then the camera is moved from $6m$ to $30m$. During this process, we capture the visible images, infrared images, and distances measured by the embedded laser rangefinder in H20T. There are totally 28 groups of measurements. After that, four pairs of corresponding matches in the visible and infrared images are found by hand. Each pair in each group of measurements stands for a single observation. Thus there are a total of 112 observations. Some examples of captured images and matched pairs are shown in Fig. 6.17.



(a) 6.9m (b) 9.7m (c) 13.0m (d) 22.5m (e) 28.4m

Figure 6.17: Examples of captured visible and infrared images at different distances. The first and second rows show the visible and infrared images, respectively. The corners of the whiteboard are matched manually.

Then all of the 112 observations are fed into a parameter estimator program developed with general graph optimization (g^{2}o) [130] library. We also implement the chi-square method to cull outliers. The output of the program is the estimated R' and t'.

To evaluate the estimation results, the following registration error (pixels) is defined:

$$e = \frac{1}{4}\sum_{i=1}^{4}\sqrt{{e_x^i}^2 + {e_y^i}^2} \tag{51}$$

Fig.6.18shows that the registration error decreases versus the distance. This is because: as the distance gets larger, the value of $\frac{d_v i}{d_i r}$ gets closer to 1. In other words, Eq. (37) describes the transformation of the visible-infrared system more accurately with the depth getting larger.

The registration error is less than 10 pixels when the depth is larger than 15 meters, and less than 6 pixels for depth over 30 meters.

Outdoor Visible-Infrared Image Registration

With the proposed model described by Eq. (37) and estimated registration parameters (R', t') in Section6.4, the contours of wildfire points predicted by proposed attention gate U-net can be transformed into infrared images, which is shown in Fig. 6.19:

[1] https://github.com/ConcordiaNAVlab/registration_vi_ir_images

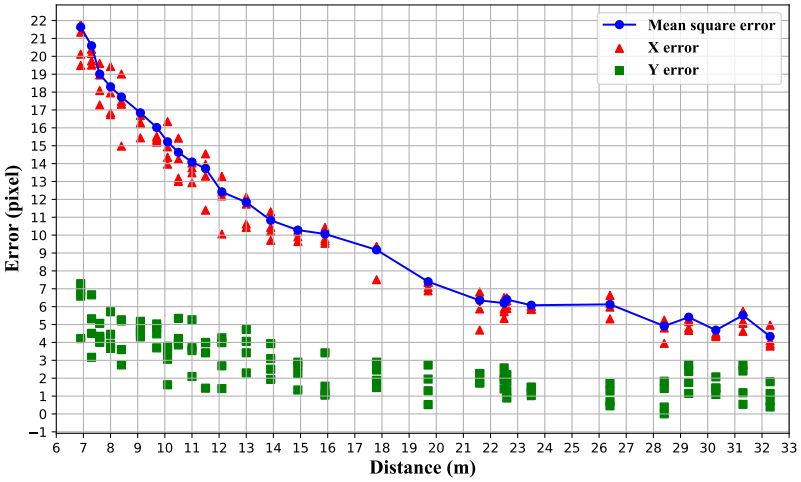
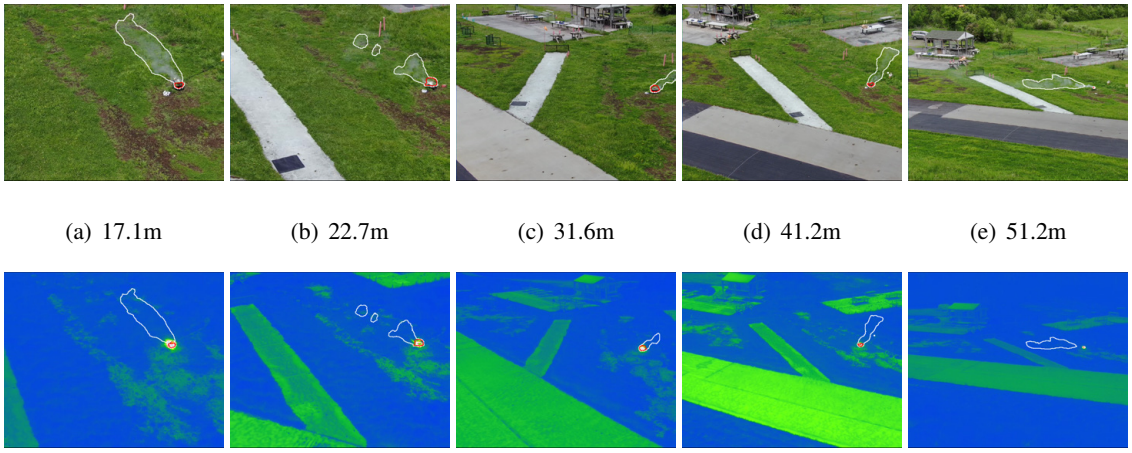


Figure 6.18: Mean square error, 4 x-direction errors and 4 y-direction errors. They both decrease with the depth getting larger, and x-direction errors contribute most of the mean square error.



(a) 17.1m (b) 22.7m (c) 31.6m (d) 41.2m (e) 51.2m

(f) 17.1m (g) 22.7m (h) 31.6m (i) 41.2m (j) 51.2m

Figure 6.19: Smoke and fire point contours are transformed from visible images into infrared images with the proposed model. White contours: smoke; red contours: wildfire point. The field of view (FOV) of two types of cameras are different.

With the confirmation of infrared images, false positive detection generated by attention gate U-net can be further reduced.

6.5 3D reconstruction experiments

6.5.1 Flight trajectory

In this outdoor experiment, a synthetic fire spot, which generates heat and smoke through burning leaves and wood, is placed near a cabin in the flight club airport. Then, the UAV flies around the wildfire spot along a circular trajectory, as shown in Fig. 6.20. During the flight, 40 high-resolution images of the scene are captured. This experimental scenario can be found on YouTube.[1]

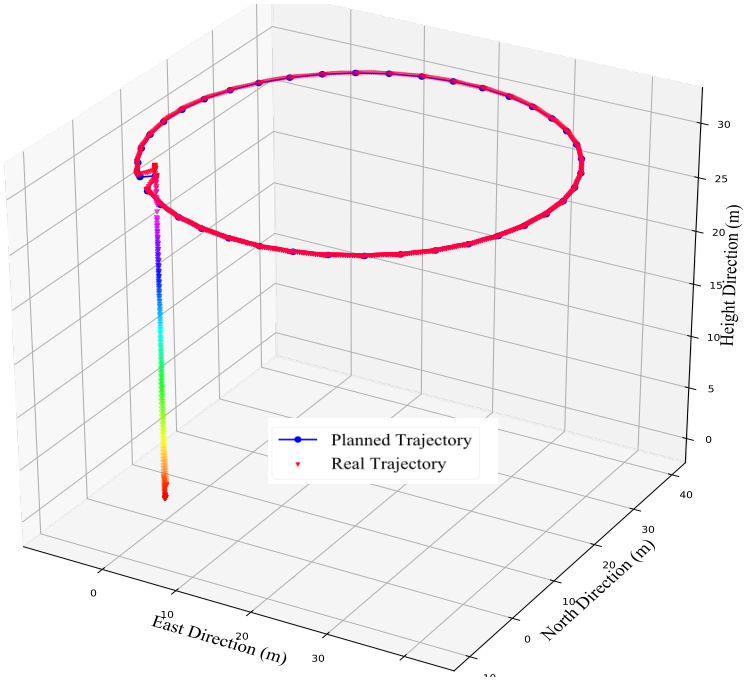


Figure 6.20: The circular trajectory takes the estimated wildfire spot as the center. When flying, the M300 UAV keeps controlling the H20T camera towards the wildfire spot.

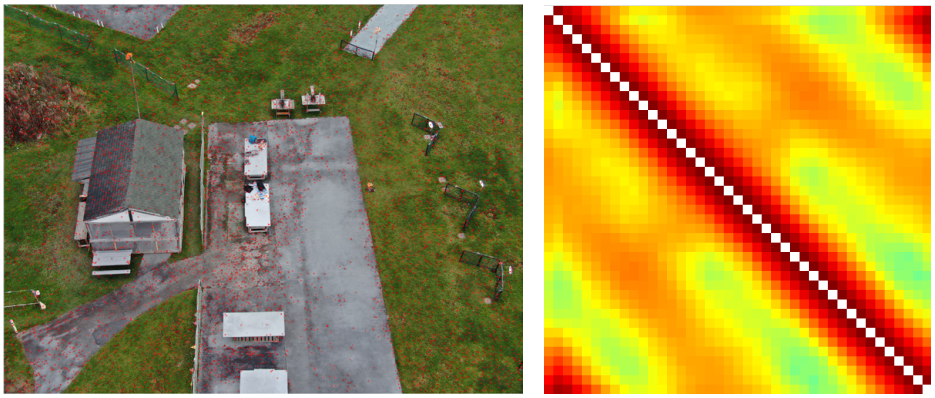
6.5.2 Reconstruction process

After the image set is captured after the flight, the images are sent to the ground station to perform SfM and MVS reconstruction.

The SIFT features are extracted from each image and then matched via sequence and prior positions based methods. Comparing with the brute force method, the GNSS prior position constraints can dramatically decrease the mismatches.

[1] https://www.youtube.com/watch?v=eU_TpNbQI8M

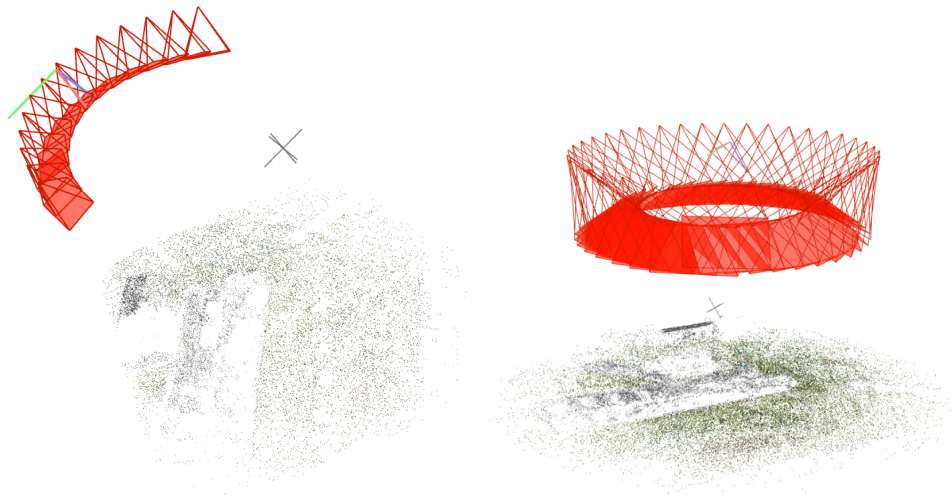
As shown in Fig. 6.21(b), the column and row represent the image. While the color represents the amount of matched features. For example, the element in the ith row and jth column represents the matching number between image ith and image jth. Due to the circular shape of the trajectory, the first several images of the sequence have relatively higher amount of matching with the last several images.



(a) (b)

Figure 6.21: GPU-based feature extraction and matching results: (a) SIFT features extracted from the high-resolution image (size: 5148×3888). (b) Feature matching matrix, the row and column of the matrix represent the image ID; Deeper color stands higher amount of matching.

After that, the best image is chosen, and incremental global BA is performed. Finally, the sparse point cloud and camera poses are recovered.



(a) (b)

Figure 6.22: SfM process and its results: (a) The reference frame is chosen and incremental global BA is performed. (b) The camera poses and sparse landmarks are recovered.

Some information on the SfM process is shown in Table 6.3.

Table 6.3: Information of the SfM process.

Images	Observations	3D points	Mean observations	Mean reprojection error
40	229525	47213	5738.125	1.2581

The output of the SfM system can be used as the initial input for the MVS system. As shown in Fig.6.23, the dense point cloud is first calculated. Then, the initial mesh is created, followed by mesh refinement. In the last step, the mesh is textured by the input images.

The time consumptions of each steps in SfM-MVS are shown in Table **??**. The total time consumption of this process is 449.27s, which can satisfy the real-time requirements.

Table 6.4: Time consumption of each steps in SfM-MVS (second).

Total	Feature extraction	Sparse reconstruction	Point cloud densify	Mesh reconstruction	Mesh refine	Texturing
449.27	18.21	43.68	20.75	21.38	319.59	25.66

Compared with the Google Earth 3D map[1], the advantages of the current UAV-based 3D reconstruction is with the live-update feature. The general 3D reconstruction frameworks are usually

designed for the large scale environment reconstruction, which needs powerful computational resources to spend much longer time to compute. Due to the flexibility of the UAVs, the local environment around the wildfire spot can be captured and reconstructed in real-time. Only the images related to the wildfire are sent to be reconstructed which reduces the computational load.

[1] https://earth.google.com/web/@45.45536928,-73.91600284,25.22622155a,74.58212502d,35y,47.57768908h,70.27467085t,0r

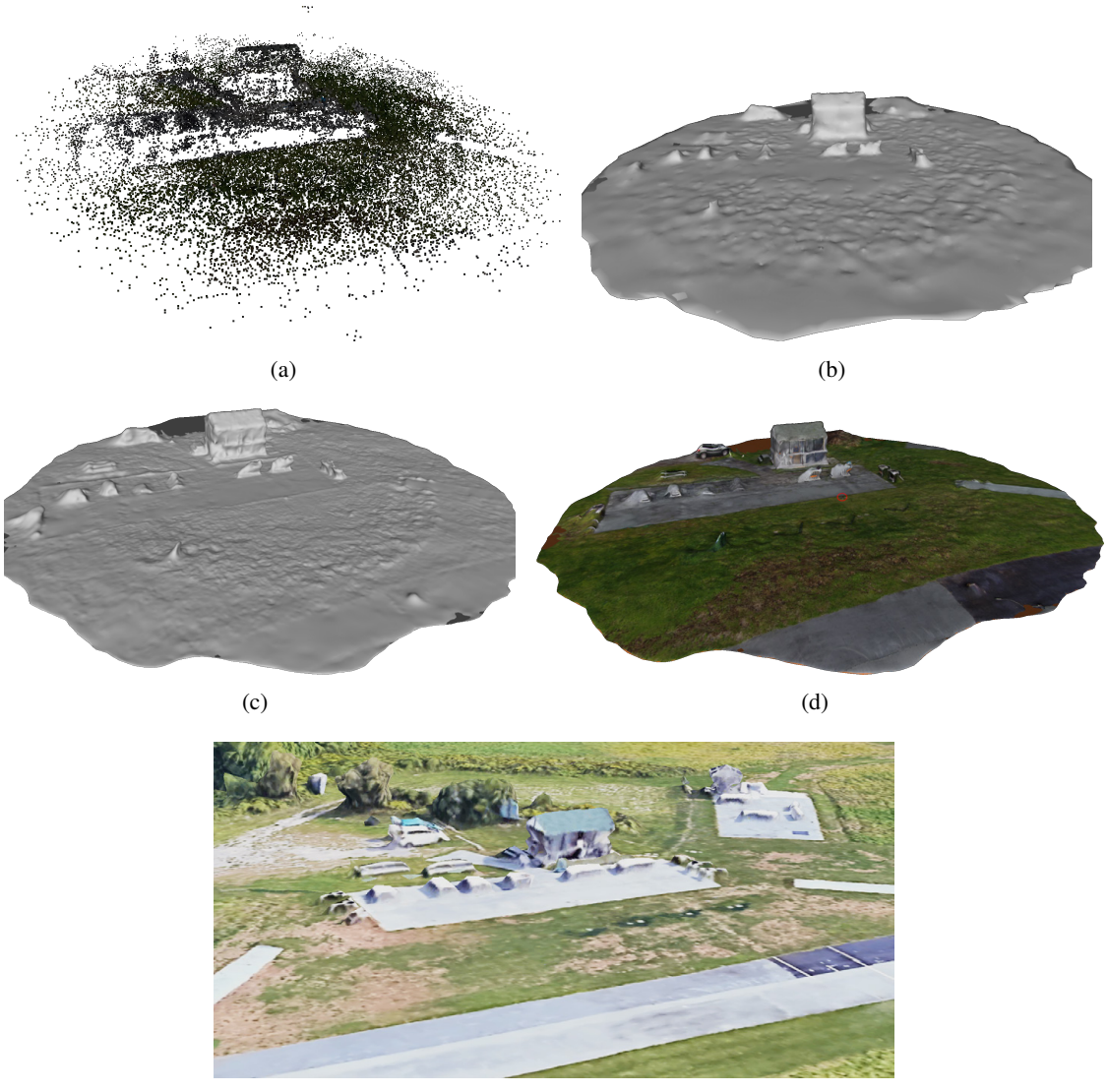


Figure 6.23: MVS process and its results: (a) The reference frame is chosen and incremental global BA is performed. (b) The initial mesh is generated from the dense point cloud. While some of the details of the scene are lost. (c) The initial mesh is refined, and details of the scene are now recovered. (d) The refined mesh is textured with the input images. (e) The same scene was recovered by Google Earth.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

Detecting early wildfires from UAV images faces many challenges, such as small fire size, complex background, image degradation, and limited processing power. To address these problems, this paper presents a novel early wildfire detection, localization, and reconstruction framework, which utilizes the following algorithms:

- A ResNet-based algorithm is used to classify UAV images into fire or non-fire categories based on their visual features. This algorithm is trained and fine-tuned on a large dataset of aerial images containing various types of fires and backgrounds.
- A threshold-based algorithm using the HSV color-space is applied to segment the potential heat spots from the infrared images. With the sliding window algorithm, the heat spot can be localized. Then, the visible and infrared images are aligned and fused to reduce the false alarm rate of the wildfire detection.
- A U-net-based algorithm is used to segment wildfire regions for distinguishing smoke and flame for early fire detection from UAV images based on their pixel-level labels.
- An attention-gate U-net-based algorithm is used to estimate the depth map of wildfire regions from monocular UAV images based on their semantic information. This algorithm leverages an attention mechanism to focus on relevant features for depth/distance estimation.

- A monocular simultaneous localization and mapping (SLAM) system is used to estimate the pose and trajectory of the H20T camera. This system also integrates global navigation satellite system (GNSS) data to improve its accuracy and recover the correct scale.
- A geometry model is proposed to register wildfire spots in visible and infrared images with their estimated depth based on their relative positions and orientations. This model does not require any calibration or parameters of the infrared camera.
- A GNSS-aided SfM-MVS algorithm is used to reconstruct the local environment around the early wildfire spot based on multiple UAV images taken from different viewpoints. This algorithm uses GNSS data to improve its scalability and efficiency .

This paper proposes a framework for wildfire monitoring and detection using Unmanned Aerial Vehicles (UAVs). The framework consists of three modules: wildfire "detection", "geolocation", and local environment "reconstruction". The wildfire detection module uses both deep learning-based and traditional computer vision-based algorithms to detect wildfire flame and smoke with visible and infrared image fusion. The use of various sensors and fusion algorithms improves the robustness and reduces the false alarm rate of the wildfire detection mission. The wildfire geolocation module estimates the depth and location of the wildfire spot using gimbal control. The vision-based depth/distance estimation algorithm does not depend on the terrain information, which enhances the stability of the estimation. The wildfire reconstruction module employs a GNSS-aided SfM-MVS algorithm to reconstruct the local environment information (terrain, vegetation, etc.) from the UAV images. The proposed framework is implemented on a DJI M300 UAV equipped with a DJI H20T camera that can capture both visible and infrared images. Several independent outdoor experiments using synthetic fire points are conducted to evaluate the performance of each module in the framework. Experimental results show that the proposed framework can achieve high accuracy and reliability in detecting, localizing, and reconstructing early wildfires from UAV images.

In summary, the proposed framework provides the reliable information of existence, geolocation and local environment for the early wildfire, which can be used as fundamental guidance for further fire fighting missions.

7.2 Future work

From the perspective of algorithms and real-world applications, there are several improvements can be considered in the future:

- Optimizing sensor fusion and image processing algorithms: By using more advanced sensor fusion techniques such as Kalman filter or particle filter to fuse GNSS and IMU data, the accuracy and efficiency of the proposed system can be improved. By using more efficient feature extraction methods such as ORB or SURF, the image processing algorithms will also be improved. The computational cost will be reduced and the robustness of our system under different conditions will be increased by these methods.

- Handling more complex scenarios: More complex scenarios will be handled by this system. These scenarios include multiple wildfires that may interfere with each other's detection or geolocation; dynamic environments that may change due to wind or human activities; occlusions that may block part of the fire or scene from view. In terms of scalability, adaptability, and reliability, new challenges will be posed for this system by these scenarios. To cope with these challenges and enhance the performance of our system, new strategies and algorithms will be explored further.

- Developing a real-time communication and visualization module: A real-time communication module will be developed that can transmit sensor data and fused images between UAVs or between UAVs and ground stations using wireless networks. A visualization module will also be developed that can display sensor data, fused images, geolocation information, 3D reconstruction results on a user interface using maps or augmented virtual reality techniques. By these modules, users will be enabled to monitor the wildfire situation in real time and interact with the system in an easy and intuitive way.

- Moreover, multiple UAV cooperation application can improve the automation of UAV-based wildfire management. For example, the sensor's information can be shared and merged by multiple UAVs to monitor the same zone at faster speed; the wildfire spots' geolocations can be sent to the fighting UAVs for early wildfire suppression.

Bibliography

[1] CALFIRE, "Year to date wildfire statistics," 2022. [Online]. Available: https://www.fire.ca.gov/stats-events/

[2] A. M. Martens, U. Silins, H. C. Proctor, C. H. Williams, M. J. Wagner, M. B. Emelko, and M. Stone, "Long-term impact of severe wildfire and post-wildfire salvage logging on macroinvertebrate assemblage structure in Alberta's Rocky Mountains," *International Journal of Wildland Fire*, vol. 28, no. 10, pp. 738–749, 2019.

[3] USFA, "Firefighter fatalities in the United States," 2022. [Online]. Available: https://www.apps.usfa.fema.gov/firefighter-fatalities/

[4] O. M. Bushnaq, A. Chaaban, and T. Y. Al-Naffouri, "The role of UAV-IoT networks in future wildfire detection," *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 16 984–16 999, 2021.

[5] J. Li, Y. Xiong, J. She, and M. Wu, "A path planning method for sweep coverage with multiple UAVs," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8967–8978, 2020.

[6] C.-Y. Lee, C.-T. Lin, C.-T. Hong, and M.-T. Su, "Smoke detection using spatial and temporal analyses," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 6, pp. 1–11, 2012.

[7] A. Mohapatra and T. Trinh, "Early wildfire detection technologies in practice—a review," *Sustainability*, vol. 14, no. 19, p. 12270, 2022.

[8] A. Bouguettaya, H. Zarzour, A. M. Taberkit, and A. Kechida, "A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms," *Signal Processing*, vol. 190, p. 108309, 2022.

[9] M. A. Akhloufi, A. Couturier, and N. A. Castro, "Unmanned aerial vehicles for wildland fires: Sensing, perception, cooperation and assistance," *Drones*, vol. 5, no. 1, p. 15, 2021.

[10] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin, "Fault-tolerant cooperative navigation of networked UAV swarms for forest fire monitoring," *Aerospace Science and Technology*, vol. 123, p. 107494, 2022.

[11] S. Sudhakar, V. Vijayakumar, C. S. Kumar, V. Priya, L. Ravi, and V. Subramaniyaswamy, "Unmanned aerial vehicle (UAV) based forest fire detection and monitoring for reducing false alarms in forest-fires," *Computer Communications*, vol. 149, pp. 1–16, 2020.

[12] X. Li, H. Gao, M. Zhang, S. Zhang, Z. Gao, J. Liu, S. Sun, T. Hu, and L. Sun, "Prediction of forest fire spread rate using UAV images and an LSTM model considering the interaction between fire and wind," *Remote Sensing*, vol. 13, no. 21, p. 4325, 2021.

[13] S. S. Esfahlani, "Mixed reality and remote sensing application of unmanned aerial vehicle in fire and smoke detection," *Journal of Industrial Information Integration*, vol. 15, pp. 42–49, 2019.

[14] F. De Vivo, M. Battipede, and E. Johnson, "Infra-red line camera data-driven edge detector in UAV forest fire monitoring," *Aerospace Science and Technology*, vol. 111, p. 106574, 2021.

[15] M. Sarkar, X. Yan, B. A. Erol, I. Raptis, and A. Homaifar, "A novel search and survey technique for unmanned aerial systems in detecting and estimating the area for wildfires," *Robotics and Autonomous Systems*, vol. 145, p. 103848, 2021.

[16] F. Carvajal-Ramírez, J. R. Marques da Silva, F. Agüera-Vega, P. Martínez-Carricondo, J. Serrano, and F. J. Moral, "Evaluation of fire severity indices based on pre-and post-fire multispectral imagery sensed from UAV," *Remote Sensing*, vol. 11, no. 9, p. 993, 2019.

[17] J.-i. Shin, W.-w. Seo, T. Kim, J. Park, and C.-s. Woo, "Using UAV multispectral images for classification of forest burn severity−A case study of the 2019 Gangneung forest fire," *Forests*, vol. 10, no. 11, p. 1025, 2019.

[18] J. M. Fernández-Guisuraga, L. Calvo, and S. Suárez-Seoane, "Monitoring post-fire neighborhood competition effects on pine saplings under different environmental conditions by means of UAV multispectral data and structure-from-motion photogrammetry," *Journal of Environmental Management*, vol. 305, p. 114373, 2022.

[19] K. Lu, R. Xu, J. Li, Y. Lv, H. Lin, and Y. Liu, "A vision-based detection and spatial localization scheme for forest fire inspection from UAV," *Forests*, vol. 13, no. 3, p. 383, 2022.

[20] V. Ciullo, L. Rossi, and A. Pieri, "Experimental fire measurement with UAV multimodal stereovision," *Remote Sensing*, vol. 12, no. 21, p. 3546, 2020.

[21] C. Yuan, Z. Liu, and Y. Zhang, "UAV-based forest fire detection and tracking using image processing techniques," in *IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*, Denver, USA, 2015.

[22] C. Yuan, Z. Liu, and Y. Zhang, "Aerial images-based forest fire detection for firefighting using optical remote sensing techniques and unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, vol. 88, no. 2, pp. 635–654, 2017.

[23] C. Yuan, Z. Liu, and Y. Zhang, "Vision-based forest fire detection in aerial images for firefighting using UAVs," in *IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*, Denver, USA, 2016.

[24] C. Yuan, Z. Liu, and Y. Zhang, "Learning-based smoke detection for unmanned aerial vehicles applied to forest fire surveillance," *Journal of Intelligent & Robotic Systems*, vol. 93, no. 1-2, pp. 337–349, 2019.

[25] J. Vasconcelos Reinolds de Sousa and P. Vieira Gamboa, "Aerial forest fire detection and monitoring using a small UAV," *KnE Engineering*, vol. 5, no. 6, p. 242–256, 2020.

[26] C. E. Prema, S. Vinsley, and S. Suresh, "Efficient flame detection based on static and dynamic texture analysis in forest fire detection," *Fire Technology*, vol. 54, no. 1, pp. 255–288, 2018.

[27] F. A. Hossain, Y. M. Zhang, and M. A. Tonima, "Forest fire flame and smoke detection from UAV-captured images using fire-specific color features and multi-color space local binary pattern," *Journal of Unmanned Vehicle Systems*, vol. 8, no. 4, pp. 285–309, 2020.

[28] L. P. Osco, J. M. Junior *et al.*, "A review on deep learning in UAV remote sensing," *International Journal of Applied Earth Observation and Geoinformation*, vol. 102, p. 102456, 2021.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[30] C. Kyrkou and T. Theocharides, "Deep-learning-based aerial image classification for emergency response applications using unmanned aerial vehicles," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Long Beach, CA, 2019.

[31] P. Barmpoutis, K. Dimitropoulos, K. Kaza, and N. Grammalidis, "Fire detection from images using Faster R-CNN and multidimensional texture analysis," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, 2019.

[32] D. Kinaneva, G. Hristov, J. Raychev, and P. Zahariev, "Early forest fire detection using drones and artificial intelligence," in *IEEE International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, 2019.

[33] Z. Jiao, Y. Zhang, L. Mu, J. Xin, S. Jiao, H. Liu, and D. Liu, "A YOLOv3-based learning strategy for real-time UAV-based forest fire detection," in *IEEE Chinese Control And Decision Conference (CCDC)*, Hefei, China, 2020.

[34] Z. Jiao, Y. Zhang, J. Xin, L. Mu, Y. Yi, H. Liu, and D. Liu, "A deep learning based forest fire detection approach using UAV and YOLOv3," in *IEEE International Conference on Industrial Artificial Intelligence (IAI)*, Shenyang, China, 2019.

[35] P. Barmpoutis, T. Stathaki, K. Dimitropoulos, and N. Grammalidis, "Early fire detection based on aerial 360-degree sensors, deep convolution neural networks and exploitation of fire dynamic textures," *Remote Sensing*, vol. 12, no. 19, p. 3177, 2020.

[36] Q. Zhang, J. Xu, L. Xu, and H. Guo, "Deep convolutional neural networks for forest fire detection," in *International Forum on Management, Education and Information Technology Application (IFMEITA)*, Guangzhou, China, 2016.

[37] W.-Y. Hsu and W.-Y. Lin, "Ratio-and-scale-aware YOLO for pedestrian detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 934–947, 2020.

[38] Y. Liu, L. Zhang, Z. Chen, Y. Yan, and H. Wang, "Multi-stream siamese and faster region-based neural network for real-time object tracking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 11, pp. 7279–7292, 2020.

[39] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-assisted Intervention*, pp. 234–241. Springer, 2015.

[40] J. Zhang, H. Zhu, P. Wang, and X. Ling, "ATT squeeze U-net: A lightweight network for forest fire detection and recognition," *IEEE Access*, vol. 9, pp. 10 858–10 870, 2021.

[41] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz *et al.*, "Attention U-net: Learning where to look for the pancreas," *arXiv preprint arXiv:1804.03999*, 2018.

[42] A. Moffatt, N. Turcios *et al.*, "Collaboration between multiple UAVs for fire detection and suppression," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, Athens, Greece, 2021.

[43] C. Yuan, Z. Liu, and Y. Zhang, "Fire detection using infrared images for UAV-based forest fire surveillance," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, Miami, USA, 2017.

[44] B. U. Töreyin, R. G. Cinbiş, Y. Dedeoğlu, and A. E. Çetin, "Fire detection in infrared video using wavelet analysis," *Optical Engineering*, vol. 46, no. 6, pp. 067 204–067 204, 2007.

[45] M. Tlig, M. Bouchouicha, M. Sayadi, and E. Moreau, "Infrared-visible images' fusion techniques for forest fire monitoring," in *2022 6th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, pp. 1–6. IEEE, 2022.

[46] S. M. Nemalidinne and D. Gupta, "Nonsubsampled contourlet domain visible and infrared image fusion framework for fire detection using pulse coupled neural network and spatial fuzzy clustering," *Fire Safety Journal*, vol. 101, pp. 84–101, 2018.

[47] S. Verstockt, A. Vanoosthuyse, S. Van Hoecke, P. Lambert, and R. Van de Walle, "Multi-sensor fire detection by fusing visual and non-visual flame features," in *2010 4th International Conference on Image and Signal Processing (ICISP)*, pp. 333–341. Trois-Rivières, Canada: Springer, 2010.

[48] J. R. Martinez-de Dios, B. C. Arrue, A. Ollero, L. Merino, and F. Gómez-Rodríguez, "Computer vision techniques for forest fire perception," *Image and Vision Computing*, vol. 26, no. 4, pp. 550–562, 2008.

[49] I. Bosch, A. Serrano, and L. Vergara, "Multisensor network system for wildfire detection using infrared image processing," *The Scientific World Journal*, vol. 2013, 2013.

[50] B. Zitova and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977–1000, 2003.

[51] Z. Xiong and Y. Zhang, "A critical review of image registration methods," *International Journal of Image and Data Fusion*, vol. 1, no. 2, pp. 137–158, 2010.

[52] J. Ma, Y. Ma, and C. Li, "Infrared and visible image fusion methods and applications: a survey," *Information Fusion*, vol. 45, pp. 153–178, 2019.

[53] X. Liu, Y. Ai, B. Tian, and D. Cao, "Robust and fast registration of infrared and visible images for electro-optical pod," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1335–1344, 2018.

[54] Q. Jiang, Y. Liu, Y. Yan, J. Deng, J. Fang, Z. Li, and X. Jiang, "A contour angle orientation for power equipment infrared and visible image registration," *IEEE Transactions on Power Delivery*, vol. 36, no. 4, pp. 2559–2569, 2020.

[55] V. Sanchez, G. Prince, J. P. Clarkson, and N. M. Rajpoot, "Registration of thermal and visible light images of diseased plants using silhouette extraction in the wavelet domain," *Pattern Recognition*, vol. 48, no. 7, pp. 2119–2128, 2015.

[56] J. Han, E. J. Pauwels, and P. De Zeeuw, "Visible and infrared image registration in man-made environments employing hybrid visual features," *Pattern Recognition Letters*, vol. 34, no. 1, pp. 42–51, 2013.

[57] C. Min, Y. Gu, Y. Li, and F. Yang, "Non-rigid infrared and visible image registration by enhanced affine transformation," *Pattern Recognition*, vol. 106, p. 107377, 2020.

[58] J. Ma, J. Zhao, Y. Ma, and J. Tian, "Non-rigid visible and infrared face registration via regularized gaussian fields criterion," *Pattern Recognition*, vol. 48, no. 3, pp. 772–784, 2015.

[59] J. P. Pluim, J. A. Maintz, and M. A. Viergever, "Mutual-information-based registration of medical images: a survey," *IEEE Transactions on Medical Imaging*, vol. 22, no. 8, pp. 986–1004, 2003.

[60] S. Li, X. Kang, and J. Hu, "Image fusion with guided filtering," *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2864–2875, 2013.

[61] H. Li, X.-J. Wu, and J. Kittler, "Infrared and visible image fusion using a deep learning framework," in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 2705–2710. IEEE, 2018.

[62] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the 7th IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157, 1999.

[63] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[64] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *2006 9th European Conference on Computer Vision (ECCV)*, pp. 430–443. Springer, 2006.

[65] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, pp. 2564–2571. IEEE, 2011.

[66] M. W. Smith, J. L. Carrivick, and D. J. Quincey, "Structure from motion photogrammetry in physical geography," *Progress in Physical Geography*, vol. 40, no. 2, pp. 247–275, 2016.

[67] D. Turner, A. Lucieer, and S. M. De Jong, "Time series analysis of landslide dynamics using an unmanned aerial vehicle (UAV)," *Remote Sensing*, vol. 7, no. 2, pp. 1736–1757, 2015.

[68] A. Lucieer, S. M. d. Jong, and D. Turner, "Mapping landslide displacements using structure from motion (SfM) and image correlation of multi-temporal UAV photography," *Progress in Physical Geography*, vol. 38, no. 1, pp. 97–116, 2014.

[69] P. Harder, M. Schirmer, J. Pomeroy, and W. Helgason, "Accuracy of snow depth estimation in mountain and prairie environments by an unmanned aerial vehicle," *The Cryosphere*, vol. 10, no. 6, pp. 2559–2571, 2016.

[70] Y. Bühler, M. S. Adams, A. Stoffel, and R. Boesch, "Photogrammetric reconstruction of homogenous snow surfaces in alpine terrain applying near-infrared UAS imagery," *International Journal of Remote Sensing*, vol. 38, no. 8-10, pp. 3135–3158, 2017.

[71] B. U. Meinen and D. T. Robinson, "Mapping erosion and deposition in an agricultural landscape: Optimization of UAV image acquisition schemes for SfM-MVS," *Remote Sensing of Environment*, vol. 239, p. 111666, 2020.

[72] N. Pineux, J. Lisein, G. Swerts, C. Bielders, P. Lejeune, G. Colinet, and A. Degré, "Can dem time series produced by UAV be used to quantify diffuse erosion in an agricultural watershed?" *Geomorphology*, vol. 280, pp. 122–136, 2017.

[73] C. Qian, H. Liu, J. Tang, Y. Chen, H. Kaartinen, A. Kukko, L. Zhu, X. Liang, L. Chen, and J. Hyyppä, "An integrated GNSS/INS/LiDAR-SLAM positioning method for highly accurate forest stem mapping," *Remote Sensing*, vol. 9, no. 1, p. 3, 2016.

[74] M. Pierzchała, P. Giguère, and R. Astrup, "Mapping forests using an unmanned ground vehicle with 3D LiDAR and graph-SLAM," *Computers and Electronics in Agriculture*, vol. 145, pp. 217–225, 2018.

[75] F. Nie, W. Zhang, Y. Wang, Y. Shi, and Q. Huang, "A forest 3-D Lidar SLAM system for rubber-tapping robot based on trunk center atlas," *IEEE/ASME Transactions on Mechatronics*, 2021.

[76] C. Gollob, T. Ritter, and A. Nothdurft, "Forest inventory with long range and high-speed personal laser scanning (PLS) and simultaneous localization and mapping (SLAM) technology," *Remote Sensing*, vol. 12, no. 9, p. 1509, 2020.

[77] J. Garforth and B. Webb, "Visual appearance analysis of forest scenes for monocular SLAM," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 1794–1800. IEEE, 2019.

[78] R. Yang and M. Pollefeys, "Multi-resolution real-time stereo on commodity graphics hardware," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I–I. IEEE, 2003.

[79] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys, "Real-time plane-sweeping stereo with multiple sweeping directions," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE, 2007.

[80] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009.

[81] M. Bleyer, C. Rhemann, and C. Rother, "Patchmatch stereo-stereo matching with slanted support windows." in *Bmvc*, vol. 11, pp. 1–11, 2011.

[82] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 2024–2039, 2015.

[83] Y. Cao, Z. Wu, and C. Shen, "Estimating depth from monocular images as classification using deep fully convolutional residual networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3174–3182, 2017.

[84] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "MVSNet: Depth inference for unstructured multi-view stereo," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 767–783, 2018.

[85] M. Song, S. Lim, and W. Kim, "Monocular depth estimation using laplacian pyramid-based depth residuals," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 11, pp. 4381–4393, 2021.

[86] J. Liu, X. Zhang, Z. Li, and T. Mao, "Multi-scale residual pyramid attention network for monocular depth estimation," in *International Conference on Pattern Recognition (ICPR)*, pp. 5137–5144. IEEE, 2021.

[87] R. Ji, K. Li, Y. Wang, X. Sun, F. Guo, X. Guo, Y. Wu, F. Huang, and J. Luo, "Semi-supervised adversarial monocular depth estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 10, pp. 2410–2422, 2019.

[88] X. Xu, Z. Chen, and F. Yin, "Multi-scale spatial attention-guided monocular depth estimation with semantic enhancement," *IEEE Transactions on Image Processing*, vol. 30, pp. 8811–8822, 2021.

[89] S. Aich, J. M. U. Vianney, M. A. Islam, and M. K. B. Liu, "Bidirectional attention network for monocular depth estimation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11 746–11 752. IEEE, 2021.

[90] V.-C. Miclea and S. Nedevschi, "Monocular depth estimation with improved long-range accuracy for UAV environment perception," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–15, 2021.

[91] D. Mo, C. Fan, Y. Shi, Y. Zhang, and R. Lu, "Soft-aligned gradient-chaining network for height estimation from single aerial images," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 3, pp. 538–542, 2020.

[92] N. Snavely, "Bundler: Structure from motion (SfM) for unordered image collections," 2008. [Online]. Available: http://phototour.cs.washington.edu/bundler/

[93] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, 2009.

[94] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4104–4113, 2016.

[95] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[96] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "Meshlab: An open-source mesh processing tool," in *Eurographics Italian Chapter Conference*, vol. 2008, pp. 129–136. Salerno, Italy, 2008.

[97] S. Fuhrmann, F. Langguth, and M. Goesele, "MVE - A multi-view reconstruction environment," in *Eurographics Workshop on Graphics and Cultural Heritage*. The Eurographics Association, 2014.

[98] M. Waechter, N. Moehrle, and M. Goesele, "Let there be color! - Large-scale texturing of 3D reconstructions," in *Proceedings of the European Conference on Computer Vision*. Springer, 2014.

[99] P. Moulon, P. Monasse, R. Perrot, and R. Marlet, "OpenMVG: Open multiple view geometry," in *International Workshop on Reproducible Research in Pattern Recognition*, pp. 60–74. Springer, 2016.

[100] D. Cernea, "OpenMVS: Multi-view stereo reconstruction library," 2020. [Online]. Available: https://cdcseacave.github.io/openMVS

[101] C. Sweeney, T. Hollerer, and M. Turk, "Theia: A fast and scalable structure-from-motion library," in *Proceedings of the 23rd ACM International Conference on Multimedia*, pp. 693–696, 2015.

[102] C. Wu, "VisualSFM: A visual structure from motion system," 2011. [Online]. Available: http://www.cs.washington.edu/homes/ccwu/vsfm

[103] M. Lhuillier and L. Quan, "A quasi-dense approach to surface reconstruction from uncalibrated images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 418–433, 2005.

[104] E. Tola, C. Strecha, and P. Fua, "Efficient large-scale multi-view stereo for ultra high-resolution image sets," *Machine Vision and Applications*, vol. 23, pp. 903–920, 2012.

[105] Y. Yao, S. Li, S. Zhu, H. Deng, T. Fang, and L. Quan, "Relative camera refinement for accurate dense reconstruction," in *2017 International Conference on 3D Vision (3DV)*, pp. 185–194. IEEE, 2017.

[106] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys, "Pixelwise view selection for unstructured multi-view stereo," in *2016 14th European Conference on Computer Vision (ECCV)*, pp. 501–518. Springer, 2016.

[107] S. Galliani, K. Lasinger, and K. Schindler, "Massively parallel multiview stereopsis by surface normal diffusion," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 873–881, 2015.

[108] N. D. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla, "Using multiple hypotheses to improve depth-maps for multi-view stereo," in *2008 10th European Conference on Computer Vision (ECCV)*, pp. 766–779. Springer, 2008.

[109] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang, "SurfaceNet: An end-to-end 3d neural network for multiview stereopsis," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2307–2315, 2017.

[110] A. Kar, C. Häne, and J. Malik, "Learning a multi-view stereo machine," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[111] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *International Journal of Computer Vision*, vol. 38, pp. 199–218, 2000.

[112] S. M. Seitz and C. R. Dyer, "Photorealistic scene reconstruction by voxel coloring," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1067–1073. IEEE, 1997.

[113] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2017.

[114] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988, 2017.

[115] J. Howard and S. Gugger, "Fastai: a layered API for deep learning," *Information*, vol. 11, no. 2, p. 108, 2020.

[116] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay," *arXiv preprint arXiv:1803.09820*, 2018.

[117] D. Regner and J. Salazar, "Object tracking control using a gimbal mechanism," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 43, pp. 189–196, 2021.

[118] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[119] G. Bradski, "The OpenCV library," *Dr. Dobb's Journal of Software Tools*, 2000.

[120] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[121] G. Guennebaud and B. Jacob, "Eigen v3," 2010. [Online]. Available: http://eigen.tuxfamily.org

[122] H. Strasdat, "Sophus," 2023. [Online]. Available: https://strasdat.github.io/Sophus/latest/

[123] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," 2022. [Online]. Available: https://github.com/ceres-solver/ceres-solver

[124] ROS, "ROS: Robot operating system," 2021. [Online]. Available: https://www.ros.org/

[125] DJI, "Onboard SDK: Expand the capabilities of your aerial platform with onboard compute," 2021. [Online]. Available: https://developer.dji.com/onboard-sdk/

[126] N. Ayoub and P. Schneider-Kamp, "Real-time on-board deep learning fault detection for autonomous UAV inspections," *Electronics*, vol. 10, no. 9, p. 1091, 2021.

[127] Nvidia-Ai-Iot, "torch2trt: An easy to use pytorch to tensorrt converter," 2021. [Online]. Available: https://github.com/NVIDIA-AI-IOT/torch2trt

[128] K. Wada, "labelme: Image polygonal annotation with Python," 2018. [Online]. Available: https://github.com/wkentaro/labelme

[129] K. Kutlu, "Create smoke and fire detection algorithm," 2020. [Online]. Available: https://www.kaggle.com/datasets/kutaykutlu/forest-fire

[130] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g^{2}o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3607–3613. IEEE, 2011.