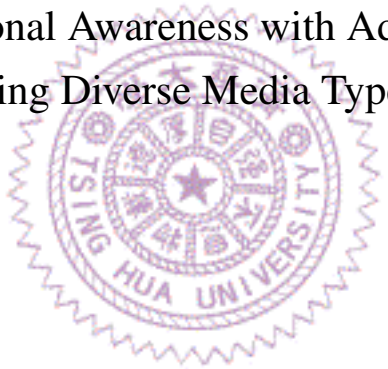
國立清華大學電機資訊學院資訊工程研究所
碩士論文
Department of Computer Science
College of Electrical Engineering and Computer Science
National Tsing Hua University
Master Thesis

增進高樓火災的狀況認知：利用空拍機上的多模態感測器與分類器

Enhancing Situational Awareness with Adaptive Firefighting Drones: Leveraging Diverse Media Types and Classifiers



范孜亦
Tzu-Yi Fan

指導教授：徐正炘 博士
Advisor: Cheng-Hsin Hsu, Ph.D.

中華民國 111 年 6 月
June, 2022

國立清華大學
資訊工程研究所

碩士論文

增進高樓火災的狀況認知：利用空拍機上的多模態感測器與分類器

范孜亦 撰

1116

國立清華大學碩士學位論文
口試委員會審定書

增進高樓火災的狀況認知：利用空拍機上的多模態感測器與分類器

Enhancing Situational Awareness with Adaptive Firefighting Drones: Leveraging Diverse Media Types and Classifiers

本論文係范孜亦君 (109062551) 在國立清華大學資訊工程研究所完成之碩士學位論文，於民國 111 年 6 月 10 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

_____

_____

_____

_____

_____

所 主 任 _____

Acknowledgments

I'm glad to thank . . .

致謝

感謝…

中文摘要

高樓火災會威脅到現代都市的生活安全品質，若火災發生在較高樓層，消防人員無法確切掌握火勢和受災人的位置，故我們提出透過無人機裝載多模態感測器，自動探索高樓環境，提供消防人員所需的資訊，在過去已經有一些研究關於無人機群的工作和路線安排，但他們並沒有考慮到抵達一個偵測點之後，應該如何做資料搜集才能達到最好的準確率，我們的論文專注於補足這一塊的缺失，當無人機抵達一個待偵測點時，提供給他一個偵測資訊串，告訴它該在哪些位子進行拍攝，在每一個位子又該用哪個感測器和選擇哪一個分類器來分析感測器搜集的資料。為了更加具體討論我們的方法，在我們的論文中主要偵測的事件為窗戶開關和每扇窗戶後面有多少受災人，其他消防人員所需資訊仍可運用相同的方法提供偵測資訊串給無人機。總結來說，在這篇論文當中，我們搜集了第一個用多模態感測器拍攝而來的窗戶資料庫，並將我們的選擇問題公式化，提出兩個演算法來解決我們的問題，最後我們實作了一個事件驅動模擬器搭配虛擬城市環境評估我們演算法的表現，也製作了使用者介面展示，並控制真實的無人機展示我們系統的運作流程。從實驗數據中可知我們的方法優於沒有產生偵測資訊串的方法，在偵測得準確度上提升了 50%，在眾多窗戶的測試中，成功達到預設的偵測時間限制和偵測準確率的比例比沒有產生偵測資訊串的方法高 100%，同時也降低了 6.78 倍的能源消耗。從真實的無人機展示證實了我們系統的可行性，以及我們具體如何將狀況認知資訊提供給消防員。

Abstract

High-rise fires are among the largest threats to safety in modern cities, and autonomous drones with multi-modal sensors can be employed to enhance situational awareness in such unfortunate disasters. In this paper, we study the fine-grained measurement selection problem for drones being dispatched to perform situation monitoring tasks in high-rise fires. Our problem considers multiple sensor/media types, classifier designs, and measurement locations, which were overlooked in prior waypoint scheduling studies. For concrete discussion, we adopt window openness and human detection as the target situations, while other situations can be readily supported by our solution as well. More specifically, we: (i) collect a very first multi-modal window dataset, (ii) mathematically formulate the fine-grained measurement selection problem and solve it using two algorithms, and (iii) create an event-driven simulator and real testbed to evaluate our algorithms individually. The evaluation results from the event-driven simulator demonstrate that our proposed algorithms achieve higher classification accuracy (up to 50% improvement), deliver more feasible solutions (up to 100% improvement), and reduce energy consumption (up to 6.78 times reduction), compared to the current practices. The demonstration of our real testbed shows that the practice of our work, and how may it provide the situational awareness in the real high-rise firefighting.

Contents

List of Figures

List of Tables


TSING HUA UNIVERSITY

Chapter 1

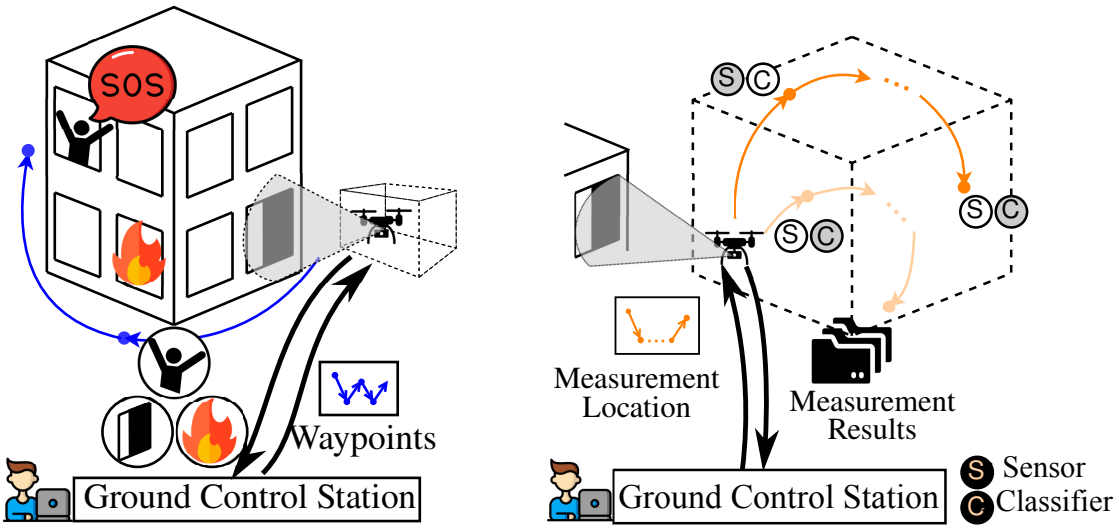
Introduction

United Nations reported that 55% of people live in urban areas as of in 2018, which is projected to reach 68% by 2050 [43]. Such a high rate of urbanization makes housing the urban populations extremely challenging. Taller high-rise buildings have been constantly built to vertically accommodate more people for living, working, socializing, and entertaining. These high-rise buildings, however, are more *vulnerable* to hazards, such as fires, earthquakes, terrorism, and disease outbreaks. As the growth of multimedia technologies and hardware devices, more and more innovative multimedia systems are developed to improve the safety in these urban living environments.

In this thesis, we consider well-recognized urban safety concerns: *high-rise fires*, which often result in severe casualties. For example, Grenfell Tower Fire in London led to 72 fatalities [113] in 2017. The challenges of firefighting in high-rise buildings are quite different from firefighting in traditional buildings for many reasons [28]. For instance, if fires break out at floors that are not reachable by fire ladders, firefighters must enter the high-rise buildings with oxygen masks/tanks, which put them in great danger. Such danger is mainly caused by the *situational unawareness* of firefighters when they are in high-rise buildings. The most critical situations in high-rise fires include: (i) the locations and severity levels of fires, (ii) regions affected by smoke, (iii) locations and vital signs of inhabitants, and (iv) window openness. These situations are not independent, e.g., sudden increases of fresh air due to a window loss could lead to rapid spreading of fires, known as wind-driven high-rise fires, which in turn result in unsurvivable environments [50].

One way to improve *situational awareness* in high-rise fires is to deploy sensors and surveillance systems throughout high-rise buildings. Doing so, however, is time-consuming, expensive, and error-prone, while sensors (such as magnetic switches) often work for homogeneous situations (such as window openness). Adding to that, sensors (like smoke detectors) are typically connected through proprietary networks, which may not interoperate with the communication networks used by firefighters. Therefore, it is

very likely that firefighters are provided partial, if any, sensor readings, and have to search through fire scenes in a brute force manner, putting them in danger of fatal casualties.



(a) (b)

Figure 1.1: Adaptive firefighting drones: (a) coarse-grained waypoint scheduling and (b) fine-grained measurement selection.

In contrast, we aim to enhance the situational awareness in high-rise fires by leveraging fine-grained measurement selection from a suite of *heterogeneous classifiers* and *multi-modal sensors* mounted on *drones*, which fly around the exterior of buildings. As different sensors, classifiers, and measurement locations affect the detection results [54], it is challenging to find the best combinations of them. Fig. 1.1 reveals the considered usage scenario, in which a ground control station instructs one or multiple drones to monitor the situations in high-rise fires. Fig. 1.1(a) shows the coarse-grained waypoint scheduling problem addressed in prior studies [177, 158, 93], where the ground control station schedules waypoints for drones to perform different location-dependent monitoring tasks. To the best of our knowledge, these existing studies largely ignored the detailed properties of heterogeneous sensors, media types, and classifiers. Instead, they assumed that each monitoring task can be completed within a given time duration while achieving a known accuracy level. These assumptions, unfortunately, deviate from the reality. For example, each reading of an ultrasound sensor is a single distance value, and a window openness classifier may require multiple such readings at slightly different locations in front of a given window. Adding to that, high-rise fire scenes are diverse and dynamic, rendering the time and accuracy of different combinations of locations, sensors, and classifiers hard to predict.

To fill in the gap, we zoom into the fine-grained planning within a bounding box of each monitoring task, as shown in Fig. 1.1(b). We consider multiple sensor/media types,

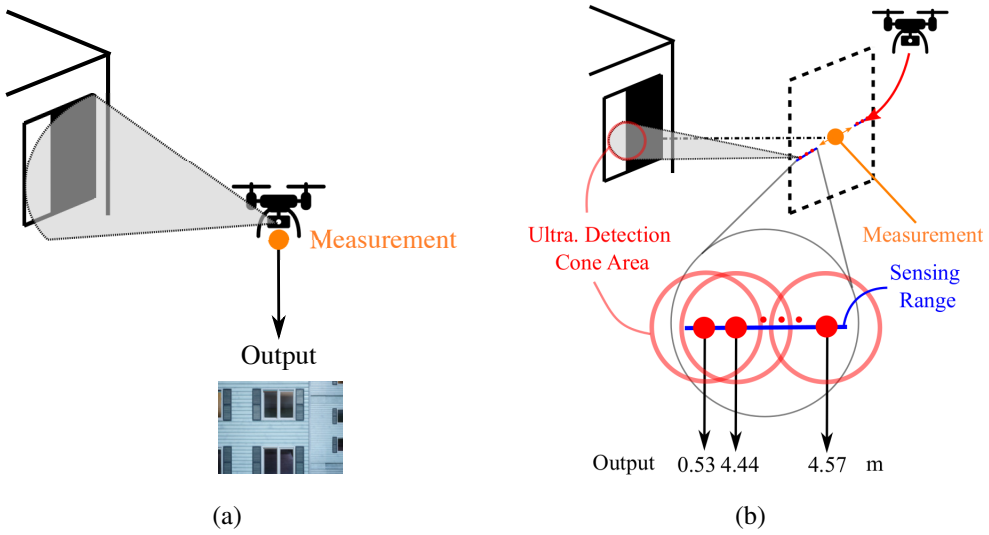


Figure 1.2: Representative one-shot and accumulated sensors: (a) RGB camera and (b) ultrasound sensor.

classifier designs, and measurement locations to accomplish every monitoring task. More specifically, our job is to select the best sequence of *measurements*, which are the combinations of locations, sensors, and classifiers. Our problem has a general setup, e.g., some sensors/classifiers are *one-shot*, where a single measurement gives rich enough data for classifications (Fig. 1.2(a)); while others are *accumulated*, where all prior measurements are jointly analyzed for classifications (Fig. 1.2(b)). RGB cameras and ultrasound sensors are representative sensors that are one-shot and accumulated, respectively. Upon analyzed by machine-learning classifiers, semantically-meaningful measurement results are sent back to the ground control station, and presented to firefighting officials. While our system and algorithms can be applied to enhance the awareness of various high-rise fire related situations, we adopt window openness as a representative situation for concrete discussion.

Handling multi-modal sensor data and computing the best measurement sequence for each monitoring task are not easy tasks for several reasons. First, we need to generate multi-modal sensor dataset by ourselves as most of existing datasets only come with RGB images. Second, different sensors/classifiers achieve diverse accuracy levels while consuming different resource amounts. Hence, finding a good trade-off between accuracy and timeliness is critical to resource-constrained drones. Third, fusing results from different sensor/classifiers may lead to even better trade-off mentioned above. Last, high-rise fire scenes are highly dynamic, and thus the measurement sequence needs to be adaptive. For example, fires may cause thick smoke, which renders RGB camera less accurate and calls for other sensors/classifiers.

1.1 Contributions

Existing situation awareness work focus on coarse-grained waypoint scheduling [177, 158, 93], which do not consider the arrangement of multi-modal sensors, classifiers, and detection locations when a drone arrives at a waypoint. In this thesis, we solve the measurement selection problem by several steps. We create both real and virtual dataset with multi-modal sensors. Then, we come up with our measurement selection algorithms to generate the measurement sequence. Finally, we evaluate our system both in an event-driven simulator and a real testbed. Our contributions can be concretized as below:

- **A very first multi-modal window dataset.** We collect our dataset both in real (WinSet [54]) and virtual environments. Window images are collect at diverse distances and angles. All of them are well-labeled. We also design and implement various classifiers for these multi-modal sensor data.
- **Measurement selection algorithms.** We mathematically formulate the measurement selection problem considering the trade-off between accuracy and time cost of diverse classifiers. We then propose efficient algorithms to compute the measurement sequence, adaptively selecting the best combinations of sensors and classifiers.
- **An event-driven simulator and real testbed.** We build an event-driven simulator connected with a photo-realistic simulator to evaluate the performance of our algorithm. We also build a real testbed to demonstrate the whole process of our system.

Some contributions above are accepted by two conference. The real window dataset, WinSet, is presented as a note paper in ACM BuildSys in 2021. Our measurement selection algorithms and the event-driven simulator [53] are presented in the special session of ACM MMSys is 2022.

1.2 Organizations

We briefly introduce our work and challenges in Chapter 1. Then, we talk about the general idea of smart city, high-rise firefighting, drone-based applications, waypoint scheduling, classifiers and regressors, and sensor fusions in Chapter 2. In Chapter 3, we analyze the pros and cons of the related work. Then, we concretize our system design in Chapter 4. Based on the components we need in our system, we introduce our real and virtual dataset in Chapter 5 and Chapter 6, classifiers we designed and implemented for the multi-modal sensor data in Chapter 7, and the measurement selection algorithms in Chapter 8. We then show the performance of our work in Chapter 9 and the real testbed demonstration

in Chapter 10. Finally, our conclusion and future works can be viewed in Chapter 11.

Chapter 2

Background

2.1 Smart City

We need many different systems and equipments to support humans' life in a city, such as transportation taking people to different places, water systems to bring people warm showers, and gas pipelines for people cooking a delicious breakfast. As the technology grows, the sufficient computation resources and large-scaled internet services can apply more services for people in the city. The concept of smart city is collecting data from Internet of Things and applying services that make people's life more convenient in modern cities [160]. The applications of smart city can across diverse aspects. For example, smart transportation that can change the traffic signal frequency according to the traffic flow, surveillance systems that can track suspicious people automatically to improve security, and smart medical treatments by monitoring patients' sensor data to protect inhabitants' health.

2.2 High-Rise Firefighting

The definitions of high-rise buildings are various among cities. The most common definition is defined by National Fire Protection Association (NFPA). They define a building with height greater than 75 feet (about 9 to 15 floors) as a high-rise buildings. However, to be practical, for any fire department, if they could not use their equipments to reach the highest point of the buildings, the buildings should be considered as high-rise buildings [104].

As this nature of high-rise buildings, firefighters need to go inside the building to perform rescue operations. The first action when the firefighters arrive is to control the lobby [42], which means to access all related devices and materials that could help firefighting, such as the alarm panel showing the real-time progression of fire, and the depos-

itory box. There are three things inside the depository box. The first one are preplans, which illustrate the design of the architecture, indicating where the stairwells, elevators are, and recording the information of standpipe systems, ventilation system, elevator recall procedures and emergency numbers. The second one is the keys of the buildings and the elevators. All of them should be labeled clearly, so that firefighters could access the room or equipments rapidly. The third one are the fire phones, which can communicate with the phones in the elevators. The phones are important to the communication between the attack firefighting team inside the buildings and the incident commanders, especially when the radio systems fail. Having the lobby control, firefighters could get into the situations and arrange the operations. Rescuing savable lives is the highest priority in the operation, so firefighters would try to allocate the victims and carry out the rescue, which is not an easy task in high-rise buildings [56, 145]. Then, after making sure the safety of the victims, firefighters would start to control and extinguish the fires.

The above description just shows the standard priority during a fire operation. Most of the work can be planned in advance. However, the past can not help the future all the time. The incident commanders would adapt their strategies according to the real-time fire report. According to this, we could know the importance of the situational awareness, which could apply sufficient information to the incident commanders to make the suitable plan.

2.3 Drone-Based Applications

A drone, which is also named as an unmanned aerial vehicle, can fly autonomously without any pilot inside. People can use a pad to control the drone remotely. Drones are first designed for the military purpose. They act an important role in the modern wars, which can recon the target area, launch missiles, or decoy enemy. In these two decades, drones become more popular to the public. As the development of emerging technology, people invent diverse types of drones [174], which are different from the number of rotors, the weight, the power source, and the degree of self-control ability. People could also install diverse payloads on drones with various functionalities, such as multi-modal sensors for monitoring, or medicine sending, based on their requirements.

Drones are used in widespread applications, such as delivering system, traffic or disaster monitoring, agriculture, and pollution detection. Mogili [112] survey paper about using drones for precision agriculture. For contries which economy relies on agriculture, they often have large area of cropland (with an average area of 234 acres in 2013 [99]). This could cost a lot of human power to manage such a large area. Therefore, people implement drones for crop monitoring, and precision agriculture [119]. Using the sen-

sors installed on the drones, farmers could estimate the height of crop [17] and even the analyze soil and field [131]. Farmers could also combine the drones with spraying systems, which can precisely control the amount of insecticide or water [70], and spray them to cover the whole cropland easily. Drones could also support the management of disasters [138]. Based on different timing, drone can have different reaction for disaster. In day life, drone could patrol around interested area to see if there is any clue that may cause disasters, such as flood early detection. Unlike some sudden disasters, e.g. an explosion by chemical materials, flood develops slowly, which can be observed. If we could have early detection of flood, people could be evacuated completely and decrease the damage. Early detection of flood could be detected by the satellites or helicopters, but both of them cost a lot of money and can not cover large enough area. Srikudkao [159] proposed to use small drones installed with multi-modal sensors (photographic, ultrasound and Global Positioning System (GPS) sensors) to detect water flow. Different from flood, earthquakes often happen without any sign. However, drones could still contribute to the rescue operations. Different damage types of buildings are important factor that if people could survive under the collapse. Using drones to collect aerial images could map the affected area and help the incident commanders to prioritize their work [148]. Moreover, after any disasters, drones could use their monitoring system to provide damage assessment, which is a essential information for the government in recovery movements. Drones could also be used in firefighting, either in forest fires or high-rise fires. We will discuss about it in the related work.

To expand more applications for drones, people still need to overcome a bunch of challenges for the drone applications [123]. For example, the flying time of the drone is short. People need to consider the trade-off between power and the drone weight, and find the best drone for their applications. They could also design some rapid charging way to overcome this problem. Better computation capabilities are also important for applications that require immediate response. Cloud computing is one of the solution, but it generates another problem, communication latency. To name just a few, there are still lots of challenges for drone applications need to conquer. They worth to call research community to make efforts in them.

2.4 Sensor Fusion

Sensor fusion is to combine sensor data or the data generated by the original sensor data, and achieve better performance instead of only using one input [46]. Based on the definition, we could know that sensor fusion is not limited to use multi-modal sensors for one application. Using a series of data from the same types of sensors could also be a kind of

sensor fusion.

With only one sensor input has several disadvantages. The first one is reliability. As we only rely on one sensor, if it breaks out, we may lose the functionalities of our applications. Besides, there are limitations for every kind of sensors, such as the field of view (FoV), detection range, and the precision of the detection. These features then cause the uncertainty of the detection from the sensors. All of these drawbacks could be solved by sensor fusion, and implemented in diverse applications of sensor fusion, such as human activity recognition [32], medical assistance [108], and automated driving [191, 180].

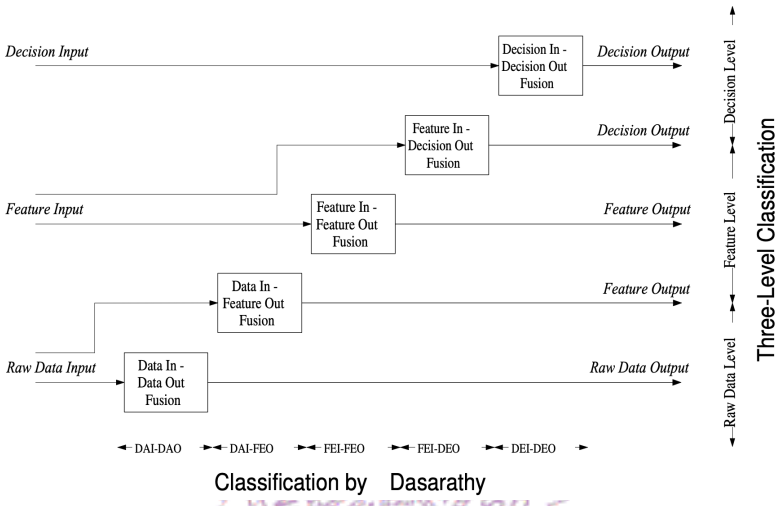


Figure 2.1: Sensor fusion category based on the input/output [46].

Sensor fusions could be categorized by their input/output [41]. Fig. 2.1 shows the details of the categories of sensor fusion. There are many types of the input/output depending on the processing degree. Following the order from the low processing degree to the high processing degree, the input/output could be the raw sensor data, features that generated from the raw sensor data (e.g. corners, edges, and textures), and the decision. There are various fusion methods and architectures for fusing different types of input/output.

2.5 Classifiers and Regressors for Multi-Modal Analytics

Classifiers and regressors are predictive models that learns from the training data to build a predictive model. People could use the predictive model to map the input data to an output prediction. The main difference between classifiers and the regressors are their output format. The prediction from the classifiers are discrete, while the one from the regressors are continuous. Take the house selling as an example. We have the details of a house, such as how large it is and where it is located. A regressor could predict the

selling price of the house, which is a real number that can be an integer or float, As for the classifiers, they may separate prices into several range, such as 0–1000, 1000–2000, and more than 2000 dollars. The output of the classifiers would be the price belongs to one of the defined range, which has only three possibilities in our example. From this example, we could know that the functionalities of classifiers and regressors are almost the same. People could choose from them based on their output requirements.

There are many work using classifiers and regressors to analyze sensor fusion data. Medina [97] collected the movement and orientation of human activity at home by installing sensors and inertial miniature board on basic commodities to build smart objects, such as a cup, a toothbrush, and a spoon. Then, they used fuzzy logic to generate sensor fusion features from spatial and temporal sensor data, and generated a activity recognition models to classify what the man was doing. Kumar [83] measured the speed, absolute linear acceleration, change in altitude, the roll and pitch of the vehicles, and trained classifiers that can predict if the car collides something, rolls over, or falls off. They evaluated the performance of three models with different structures, which were the Gaussian mixture model, decision tree, and the naive bayes. As for the regressors, Hobson [117] built a regressor to predict the level of occupancy by combining the data of lighting, CO_2, plug loads, and the counts of Wi-Fi enabled devices in an academic building. Mustapha [117] collected data from the wear devices and use Support Vector Machine (SVM) or Convolotional Neural Network (CNN) for each sensor data first, and then use a regressor to combine all data together to predict the speed and load of pedestrians.

Chapter 3

Related Work

3.1 Heterogeneous Sensors on Drones

Using heterogeneous sensors to classify situations at high-rise fire scenes is challenging, and the same challenge was faced by researchers building autonomous cars. For example, Hu et al. [68] surveyed the latest work on multi-sensor fusion, where multi-modal sensors were employed for classification applications like obstacle detection and collision avoidance. The research community has also investigated the possibility of mounting multi-modal sensors on drones. For fire detection, Lewicki et al. [89] developed a system using multiple thermal cameras. Their system realized low-power joint autonomous driving and multi-organ target detection. Dang-Ngoc et al. [40] proposed a fire detection system based on RGB cameras, which recognized the color and motion features from images. In addition, Wolfgang et al. [82] proposed a drone-based fire detection system using an aspirating smoke sensor, two gas sensors (H_2 and C_xH_x), and a microwave radiometer. Among them, microwave radiometer was used to detect hidden fire sources. Their design was demonstrated to effectively reduce the possibility of fire re-ignition. Wang et al. [178] proposed a fiber-lasers-based gas sensor to detect fires using the concentration of C_2H_2, CO and CO_2. *In our project, we also plan to mount multi-modal sensors on drones for the classifiers*.

3.2 Firefighting Drones

Firefighters have always been considered as high-risk occupations. In order to ensure the safety of firefighters, different drones have been used in the field of firefighting. However, due to maneuverability and other reasons, it is often difficult for drones to directly reach fire sources. Ando et al. [16] proposed a new type of fire extinguishing drone. It comes with a 2-meter long robot arm, which can ensure stable flight when the water is discharged

through the carried hose. By doing so, the drone can inject water into the fire sources. Ogawa et al. [125] proposed a fire extinguishing mechanism by injecting gases (CO_2, N_2, etc.). In particular, they installed a gas-filled rubber balloon on a four-axis drone and then controlled the gas release remotely. Alshbatat et al. [13] proposed to have drones throw fire extinguishing balls. Their drones can control the release of the fire extinguishing balls. Upon being released, the fire extinguish ball will be activated once reaching any fire sources. *These projects on firefighting drones are also orthogonal to our project, which aims to build situation classifiers.*

3.3 Coarse-Grained Waypoint Scheduling for Drones

As the technology advances in sensing, mobility, and computing, drones have been employed for enhancing the security and safety of our living environments. In normal times, drones can act as patrollers surveilling buildings [134]. During emergency, drones can localize fire sources, throw extinguishing balls [51], aid in relief efforts [12], and even assess the environments for hazards afterwards. Many of above-mentioned tasks can not be done by a single drone. Therefore, how to coordinate drones to complete these tasks is an important research problem. There are research papers on scheduling waypoints for multiple agents in different settings to have drones monitor a large area for an extensive time duration. Some of these papers cast the waypoint scheduling problem into combinatorial optimization problems. Valavanis et al. [172] formulated such problems and proved them to be NP-hard. Another work [105] proposed heuristic algorithms built upon Mixed Integer Linear Programming (MILP), Markov Decision Process (MDP), and game theory for the same problems. A few other papers took the monitoring quality and the safety of drones into considerations. For example, Wallar et al. [177] designed a planner for drones to continuously monitor risky regions on a 2D map. We note that the situations at interested locations may change, and the accuracy levels of detected situations drop over time. To keep classification results reliable, one needs to carefully choose the visiting frequency of each location. For instance, Smith et al. [158] proposed a dynamic approach to patrol areas with drones, and Sea et al. [149] balanced the workloads of drones visiting multiple regions. There are several driving use cases for these settings. For example, Beard et al. [22] and Kim et al. [80] controlled drones to visit target areas in the presence of dynamic threats and hostile environments. Lin et al. [91] made drones to rendezvous at unspecified locations, and Leary et al. [85] used drones to capture geo-dispersed targets in no-fly zones. *While drone coordination is a key component of our testbed, the current proposal concentrates on the development of classifiers for high-rise fire situations.*

3.4 Window Dataset

Windows are the openings of high-rise buildings, and are crucial to situation dynamics in high-rise firefighting. Among all window-related situations, *window openness* is the most critical one to detect. To the best of our knowledge, there is only one dataset focusing on openness as a window state. Safavi et al. [144] collected a sound dataset with four window states: open, close, open-to-close, and close-to-open. Unfortunately, such a sound dataset is not useful to our project, in which image-based sensors are also used. While not for window openness, RGB images of windows have been collected for applications, like window localization [94, 121]. Examples of such datasets include Tylecek et al. [171], Teboul [165], Gadde et al. [57], Ceylan et al. [29], Daftry et al. [39], and TSG-20 [151]. Besides RGB images, datasets from other sensors were also considered for dataset collection. For example, Wang et al. [179] and Malihi et al. [101] employed LiDAR to gather building structure datasets, while Sirmacek et al. [156] adopted thermal cameras to capture a building opening dataset. The above mentioned window-related datasets only considered a single sensor modality, and thus is different from what we propose to do. There were also attempts on composing multi-modal datasets for window localization. For example, Jarzabek et al. [74] and Lin et al. [90] concurrently collected RGB and thermal images of buildings for detecting air leaks to save Heating, Ventilation and Air Conditioning (HVAC) energy. Different from our proposed project, their datasets only considered two modalities, and are not publicly available. *Since there exists no multi-modal window dataset, we have to collect a very first dataset ourselves.*

Chapter 4

Drone-Based High-Rise Firefighing

As illustrated in Fig. 1.1, a drone-based high-rise firefighting system enhances the situational awareness by instructing drones to fly through waypoints for detecting situations. Our work focuses on the fine-grained planning at each waypoint. For completeness, we give a high-level component diagram in Fig. 4.1, and present an overview on both the coarse-grained waypoint scheduling and fine-grained measurement selection operations in the following.

4.1 Coarse-Grained Waypoint Scheduling

Our system takes inputs form users, i.e., firefighting officials, for monitoring tasks based on fire reports, earlier classification results, and their domain knowledge. Sample tasks could be detecting the presence of fire, trapped humans, and opened windows. The *task generator* at the ground control station assigns tasks for drones by specifying the locations, visiting frequencies, and so on. Then the waypoint scheduler associates waypoints (3D coordinates) with individual drones, which then fulfill all tasks periodically. Here, each waypoint specifies a single coordinate for performing a task, such as detecting window openness. The *precise*, or fine-grained, locations for activating sensors, such as RGB cameras and ultrasound sensors, are not determined as they may need to be dynamically adapted based on the earlier classification results.

4.2 Fine-Grained Measurement Selection

The ground control station sends waypoint sequences to drones. Drones capture and analyze sensor data at each waypoint on the sequences. Whenever a drone reaches a waypoint, the *measurement selection algorithm* computes a measurement sequence within a given bounding box to optimize the classification results fused from previous measure-

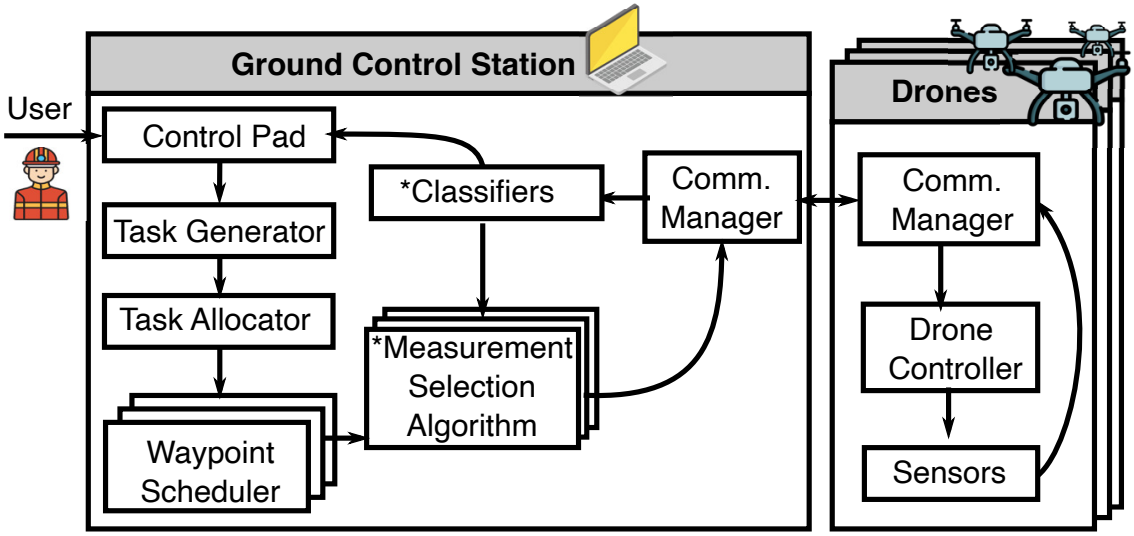


Figure 4.1: Drone-based high-rise firefighting system; asterisks indicate considered components.

(a) (b)

Figure 4.2: Our simulator: (a) high-rise buildings and (b) a flying drone.

ments at that waypoint. The computation of the best measurement sequence considers accuracy level, resource consumption, available time, etc. A measurement leads to an one-time execution of a particular classifier on collected sensor data at a specific location. The three key components of our system are *sensors*, *classifiers*, and *measurement selection algorithms*, which are developed in the next two sections. Last, we mention that classifiers and measurement selection algorithms can be offloaded to the ground control station if: (i) drones have stringent resource constraints or (ii) algorithms have high complexity levels. The relevant data transfers are done via the *communication managers* at the ground control station and drones.

Chapter 5

Real Dataset Collection

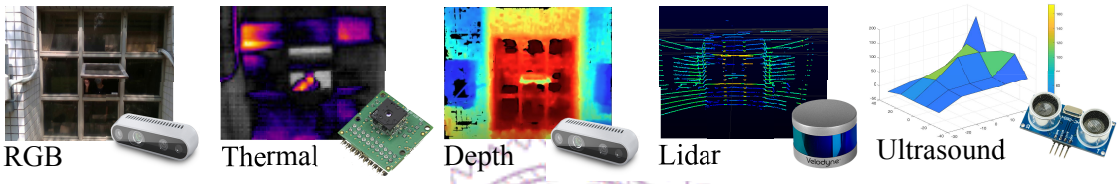


Figure 5.1: Different sensor modalities and their captured images and representations.

We collect the first multi-modal window state dataset named *WinSet* [54] for smart cities. We employ the following sensor modalities: RGB, thermal, depth, LiDAR, and ultrasound, as illustrated in Fig. 5.1. WinSet consists of two parts: datasets A and B. Dataset A collects the multi-modal window images with distinct states from different angles and at diverse distances. Possible usages of this dataset include finding the best sensor combination for detecting each window state, and designing concrete state classifiers.A classifier for human behind, for example, could be a part of BMS [23, 24], applying data to help victim rescue in firefighting, or used by the police department to locate active shooters or prevent suicide cases, as mentioned earlier. Dataset B collects the multi-modal images of various types of windows with different openness states at a given angle (0°) and distances (1, 2, and 3 m). One possible usage of this dataset is designing state classifiers that work across different window types. *To the best of our knowledge, WinSet is the very first multi-modal window dataset.*

While we believe WinSet can be leveraged in many usage scenarios, we demonstrate three sample ones in this thesis.

- **Multi-modal image segmentation.** We train a Neural-Network (NN) based classifier to segment the captured window images. This classifier is multi-modal because it takes both RGB and thermal images as inputs.
- **Distinguishability of different sensors.** We quantify the distinguishability of each

sensor modality on window states. We report the sample results from window openness.

- **Open window detection.** We propose two classifiers for window openness. We then evaluate their performance using our dataset.

Our dataset is publicly available online [54]. We intend to make this dataset public and open-source; future contributions from the research community are highly welcome.

5.1 Collection Procedure

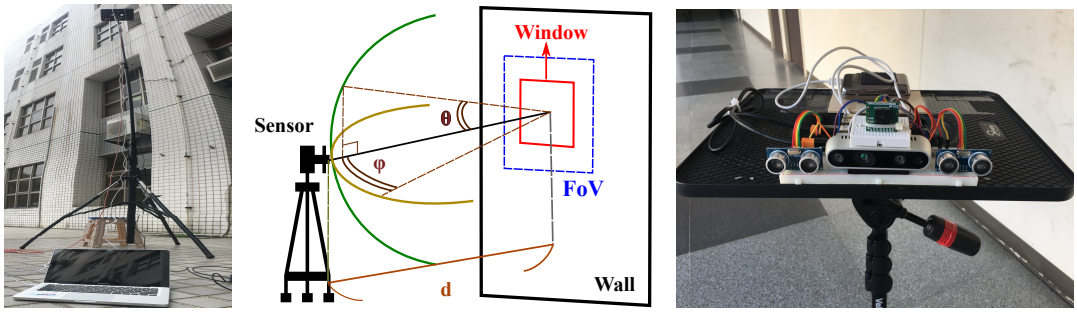
We present our datasets A and B in this section.

Table 5.1: Specifications of Adopted Sensors

A	B	Modality	Make/Model	Technology	Sampling Rate	Field of View	Raw Format
✓	✓	RGB	Intel Realsense D435	Visible Light	680×480 @15 fps	69°×42°	.bag
✓		Thermal	FLIR Lepton 2.0	Long Wavelength Infrared	80×60 @8.6 fps	50°×40°	.bin, .png
	✓	Thermal	FLIR Lepton 3.5	Long Wavelength Infrared	160×120 @8.7 fps	57°×45°	.bin, .png
✓	✓	Depth	Intel Realsense D435	Active Stereo Infrared	680×480 @15 fps	86°×57°	.bag
✓		LiDAR	Velodyne Puck VLP-16	Laser	1875×16 @10 fps	360°×30°	.pcap
	✓	Ultrasound	OSEPP HC-SR04	Sonar	1×1 @10 fps	60°×60°	.txt

5.1.1 Dataset A

Hardware. We choose four off-the-shelf sensors to collect dataset A, as reported in Table 5.1. The RGB camera captures visible lights into images. The thermal camera uses a microbolometer to detect the infrared (emitted by objects), translates it into relative temperature, and creates thermal images. The depth camera collects two parallel images, and then calculates the depth of each pixel. Last, LiDAR emits laser beams, assesses the duration for the beams to bounce back, calculates the distances to objects, and generates a 3D-map for the surrounding environment. Because we plan to collect dataset A from various angles, we purchased a 7-meter tripod with a pan-tilt head, as illustrated in Fig. 5.2(a).



(a) (b) (c)

Figure 5.2: Data collection setup: (a) 7-meter tripod, (b) tripod setup for dataset A, and (c) sensor setup for dataset B.

To capture multi-modal data, we attach all sensors on a platform, which is then mounted on the tripod.

Software. We use software tools provided by the sensor manufactures to capture image frames. For example, we install Intel Realsense Viewer from their official GitHub [73] on our laptop to record RGB and depth image frames. The tool compresses each pair of RGB and depth image frames into a .bag file. For the thermal sensor, we set it up on a Raspberry Pi 3, compile the code of Lepton module from Groupgets [62] to get thermal image frames. We also save the original thermal data using a Python script into .bin files. For LiDAR, we utilize VeloView [81] to capture the detected 3D points, which are saved into a .pcap file, frame by frame.

Steps. We collect data from four exterior windows on our campus. We vary the following parameters: (i) distance $d \in \{3, 6, 12\}$ meter, (ii) polar angle $\theta \in \{0°, 30°\}$, and (iii) azimuthal angle $\varphi \in \{0°, 30°, 60°\}$, as shown in Fig. 5.2(b). We consider different window states: (i) openness, (ii) human, and (iii) lighting. Each measurement lasts for 10 seconds. Because the tripod is heavy, and the ground is uneven, it took us a lot of time to set up the tripod. We spent about 8 hours collecting all multi-modal data from a window, letting alone processing and labeling the collected dataset.

5.1.2 Dataset B

Hardware. We also choose four sensors for dataset B, as reported in Table 5.1. Compared to dataset A, we use a thermal camera with a higher resolution of 160 x 120. We also add two ultrasound sensors. Both exterior and interior windows are considered in dataset B. We set up all sensors on a platform, which is revealed in Fig. 5.2(c). We set two ultrasound sensors at the two sides of the Intel Realsense, inspired by Bai et al. [19]. This is to detect the two casements that are common among several window types. All sensors are connected to a Raspberry Pi 4 for better mobility.

Software. For RGB and depth sensors, we use Intel Realsense rs-capture instead of Intel Realsense Viewer to conserve energy of Raspberry Pi 4. Different from other sensors, the ultrasound sensors only gets readings rather than images. We program the ultrasound sensors to detect ten times every second, and save the values into a .txt file.

Steps. In dataset B, we consider diverse window types, such as sliding, awing, barred, and screened windows. We point the sensors to the center of each window (both θ and φ are 0°), and collect the dataset at three different distances $d \in \{1, 2, 3\}$ m. Each measurement lasts for 10 seconds. We focus on openness state. For each measurement, we capture image frames from the RGB, depth and ultrasound sensors first, and then capture the thermal image frames. This two-step approach is attributed to the RAM limitation of the Raspberry Pi 4. We only save 20 image frames for each measurement, because we observe extensive temporal redundancy. We have collected 12 windows, which can be classified into 6 different types. This is not an easy task because most buildings have homogeneous window types. Furthermore, campus buildings are crowded with students, and we have to capture the dataset during off hours.

5.1.3 Semantic Labeling

We annotate each image pixel with multiple labels, including glass, window frame, wall, floor, ceiling, human, open window, and background. Among them, background indicates unlabeled pixels. In addition, each pixel may be annotated with two or more labels. In particular, we choose Labelme [176] as our labeling tool to annotate our images. Because RGB images have the highest resolution, we decide to label RGB images only. The labels can be readily propagated to other sensor data. Moreover, for each RGB video, we only label 20 equal-distanced images frames due to the temporal redundancy. We recruit six students to label WinSet. The labelled images are visually inspected by an expert, and about 32% of the labelled images were returned to the students for re-labeling. In total, more than 236 human-hours were spent to label WinSet.

5.2 Dataset Format

We number all windows into a01–a04 and b01–b12 for datasets A and B, as summarized in Table 5.2.

Data format conversion. To save data conversion time for engineers and researchers, we provide alternate data formats other than the raw format from the multi-modal sensors. Table 5.3 summarizes the file formats of datasets A and B. We note that the raw formats are in italic fonts in this table. In addition to merge per-frame data files (e.g., images)

Table 5.2: Type of Each Window

Window Type	**Sliding**	**Sliding + Screen**	**Sliding + Curtain**	**Sliding + Barred**	**Awing**	**Casement**
Window Index	a01, a02, a03, b01, b02, b03, b04	b05	b06	b07, b08, b09	a04, b10, b11	b12

Table 5.3: Data File Format

	RGB	**Thermal**	**Depth**	**LiDAR**	**Ultrasound**	**Semantic Label**
A	png, mp4*	*bin*, *png*, mp4*	*bag**, mp4*	*pcap**	N/A	*json*, img, txt, label_img, viz
B	png, txt	*bin*, *png*	bin, png, txt	N/A	*txt*	*json*, img, txt, label_img, viz

- *Italic fonts* indicate raw data formats from sensors.
- * indicates an aggregated file saved for each measurement.

into aggregated data files (e.g., videos), we also process the RGB, thermal, and depth sensor data as follows. For RGB and depth sensors, we use the converter from the Intel Realsense SDK to extract image frames from these two sensors. For dataset A, we encode the .png image frames into a 10-second .mp4 video using H.264 codec. The resulting video files can be used to detect moving humans in each measured video. For dataset B, since we focus on window openness window, we leave the data in lossless .png format. For the thermal sensor, we also use H.264 codec to encode the .png image frames into .mp4 videos in dataset A. We keep raw .png files in dataset B. For the semantic labels, we extract several file formats from the original .json files: *img* files are the RGB images to be labeled, *txt* lists the names of labeled objects, and *label_img* is a .png file with only one channel, where the pixel value indicates the label. In addition, we also merge *img* and *label_img* into a multi-layer *viz* image. The aforementioned alternate data file formats could speed up the adoption of WinSet.

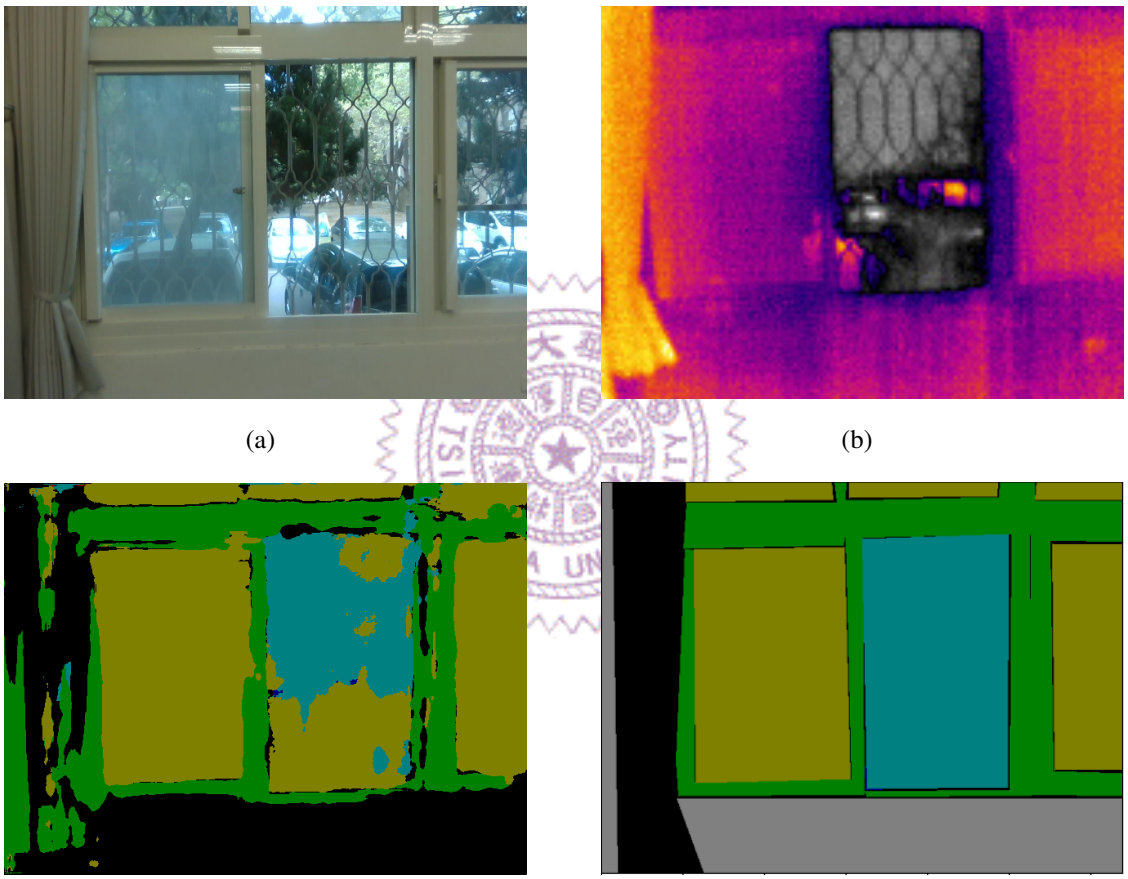
Naming convention. We name the dataset files in a self-explanatory way by systematically encode the settings in the file/directory name. Take `a01_03_00_00_cw_np_l0` as an example. `a01` indicates the window number, while the next three numbers represent the distance (3 m), polar angle (0°), and azimuthal angle (0°). These are followed by close window `cw` (versus `ow`), no human `np` (versus `wp`), and lighting off `l0` (versus `l1`). The file directories are organized into the following hierarchical structure:

- Dataset (A or B)
- Sensor modality (such as `rgb`)
- Data format (such as `png`)

- Window number (such as `a01`)
- Distance and angle (such as `a01_03_00_00`)
- Window state (such as `a01_03_00_00_cw_np_l0`)

The inner-most folders may contain multiple data files. WinSet contains 634 measured files/folders for each sensor, which occupy about 152.3 GB space in total.

5.3 Sample Usage Scenarios



(a) (b)

(c) (d)

Figure 5.3: Sample results from MFNet testing set: (a) RGB image, (b) thermal image, (c) segmentation results, and (d) ground truth.

5.3.1 Multi-modal Image Segmentation

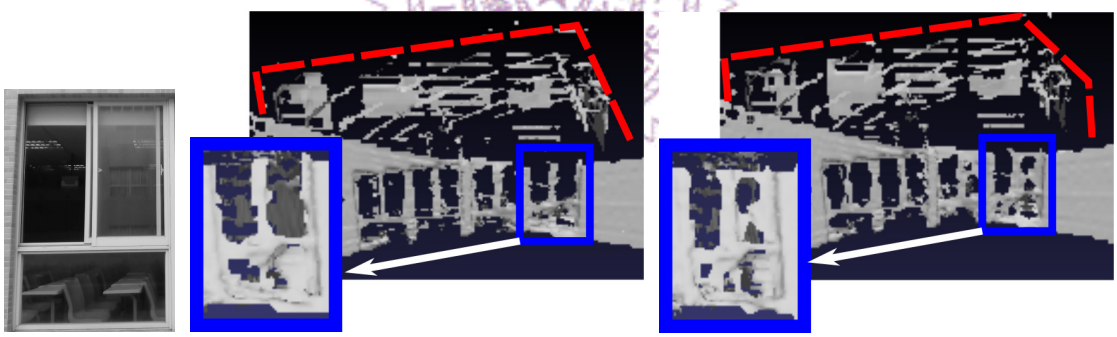
Most existing segmentation studies solely consider RGB modality. However, RGB images may be less reliable in some situations, e.g., at night or under extremely brightness. One way to cope with this issue is to leverage additional sensor modalities. For example, MFNet [64] presents a multi-modal neural network to segment images concurrently using

RGB and thermal images. There are two encoders in MFNet. One is for RGB images, and the other is for thermal ones. They separately extract the features from the two different sensor modalities, and combine them together with a decoder.

To show one possible usage of our dataset B, we retrain the MFNet with 719 pairs of RGB and thermal images. We use the remaining 320 pairs for testing. More specially, we upsample our thermal images from 160×120 to 640×480, and then stitch them with the corresponding RGB images. The resulting 4-channel images and semantic labels are sent into the MFNet for training. We set the batch size to 8, and train the model for 100 epochs.

We observe a testing accuracy of 70% on the image segmentation results. Fig. 5.3 demonstrates sample results from one of a fairly challenging window. The figure reveals that while MFNet successfully segments glass, window frames, and open window, it fails to segment walls. This, however, is not surprising because walls are often featureless. *This usage scenario demonstrates that WinSet can be used to build multi-modal neural networks for computer vision applications.*

5.3.2 Distinguishability of Different Sensors



(a) (b) (c)

Figure 5.4: measured results from the LiDAR sensors with open/close windows are hard to distinguish: (a) RGB, (b) LiDAR with a close window, and (c) with an open window.

To understand which sensors have better distinguishability among different window states, we analyze the image frames from dataset A. In the following discussion, we take window openness as an example. Other window states can be analyzed with the same approach.

Observations on raw sensor data. When we change the window states, difference can be observed in the data from most sensors except the LiDAR. Fig. 5.4 shows the 3D meshes built from the point clouds collected from the LiDAR sensor. The figure shows that LiDAR can detect the classroom behind the window in both states. In other words,

(a)

(b)

Figure 5.5: Sample dataset of a whole image and a crop image: (a) whole image and (b) crop image.

there is no distinctive difference between open and close windows. Hence, we exclude the LiDAR sensor from the following discussion.

Distinguishability metrics. For any two images, X and Y, we define four metrics to quantify their distinguishability. The first two metrics are based on the sample histograms of X and Y.

- *Histogram Correlation (HC)* indicates how close the two (grayscale) histograms match.
- *Mutual Information (MI)* is defined as $H(X)+H(Y)-H(X,Y)$, where $H(X)$ and $H(Y)$ are the entropies of the histograms and $H(X,Y)$ is their joint entropy.

For HC and MI, higher values mean lower distinguishability. The next two metrics are based on the analysis on pixel-wise difference $Z = X - Y$. Our intuition is: when Z has higher *uniformity*, X and Y have lower distinguishability.

- *Image Energy (ENG)*, a.k.a. the angular second moment, quantifies how often each pair of values appear as adjacent pixels. It is computed by the sum of the squared elements in the Gray-Level Co-occurrence Matrix (GLCM) [20] to characterize the image texture.
- *Number of Gaussian till Homogeneity (NG)* recursively counts the number of the Gaussian filter we need to apply until the sum of the square difference between the input and output images converges. Here, we define convergence as the sum of the square difference drops below 10^{-5}.

Higher ENG values mean lower distinguishability, while lower NG values mean lower distinguishability.

Comparison procedure. While our long-distance dataset contains 10-sec videos, the above mentioned metrics work on individual images. To compute statistically meaningful results, for each window, distance, and azimuthal angle, we randomly select 100 pairs

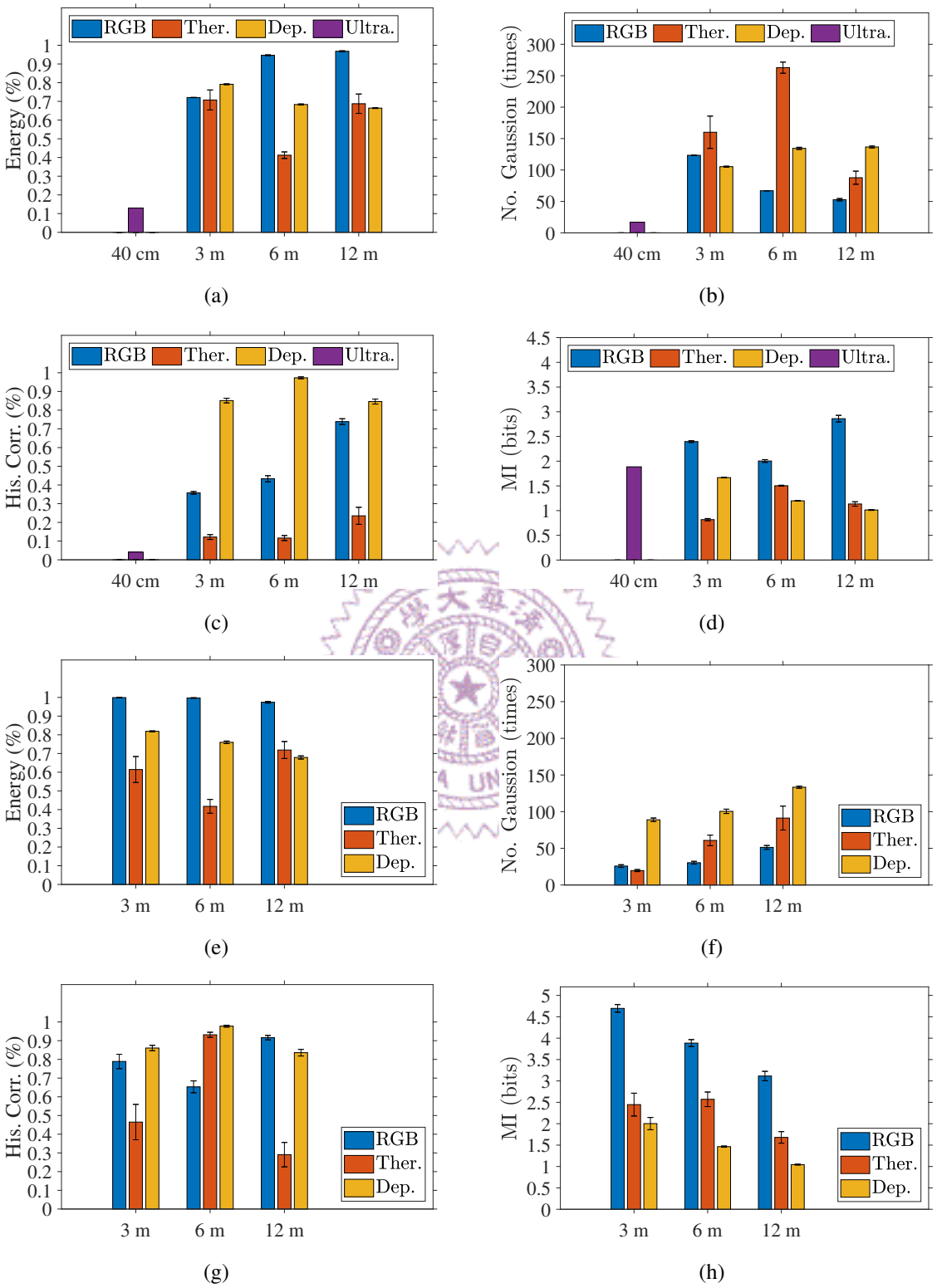


Figure 5.6: Distinguishability using different sensors on whole windows at different distance. Different states : (a)–(d) and Same states : (e)–(h). With metrics ENG : (a), (e); NG : (b), (f); HC : (c), (g); and MI : (d), (h).

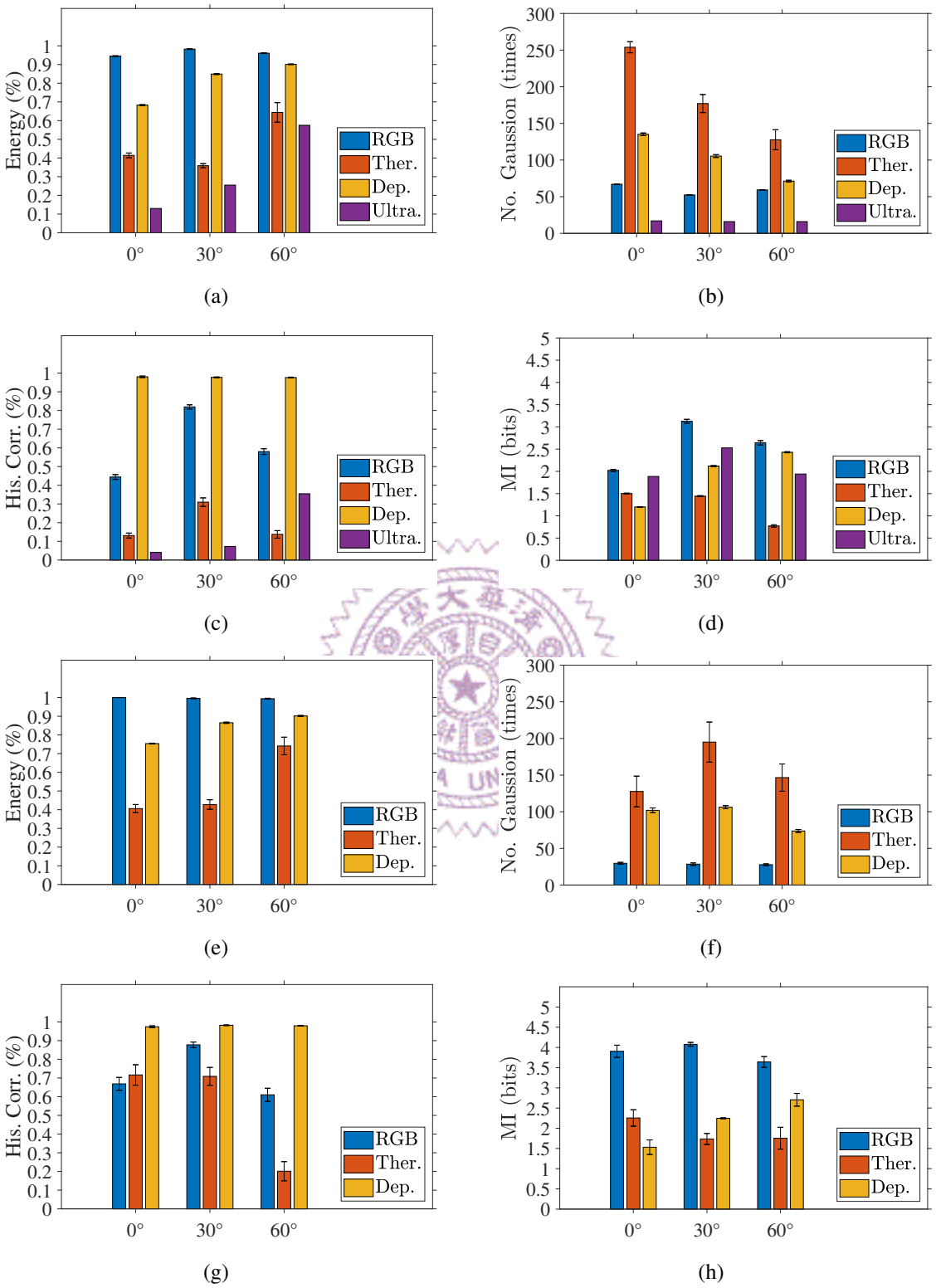


Figure 5.7: Distinguishability using different sensors on whole windows at different angles. Different states : (a)–(d) and Same states : (e)–(h). With metrics ENG : (a), (e); NG : (b), (f); HC : (c), (g); and MI : (d), (h).

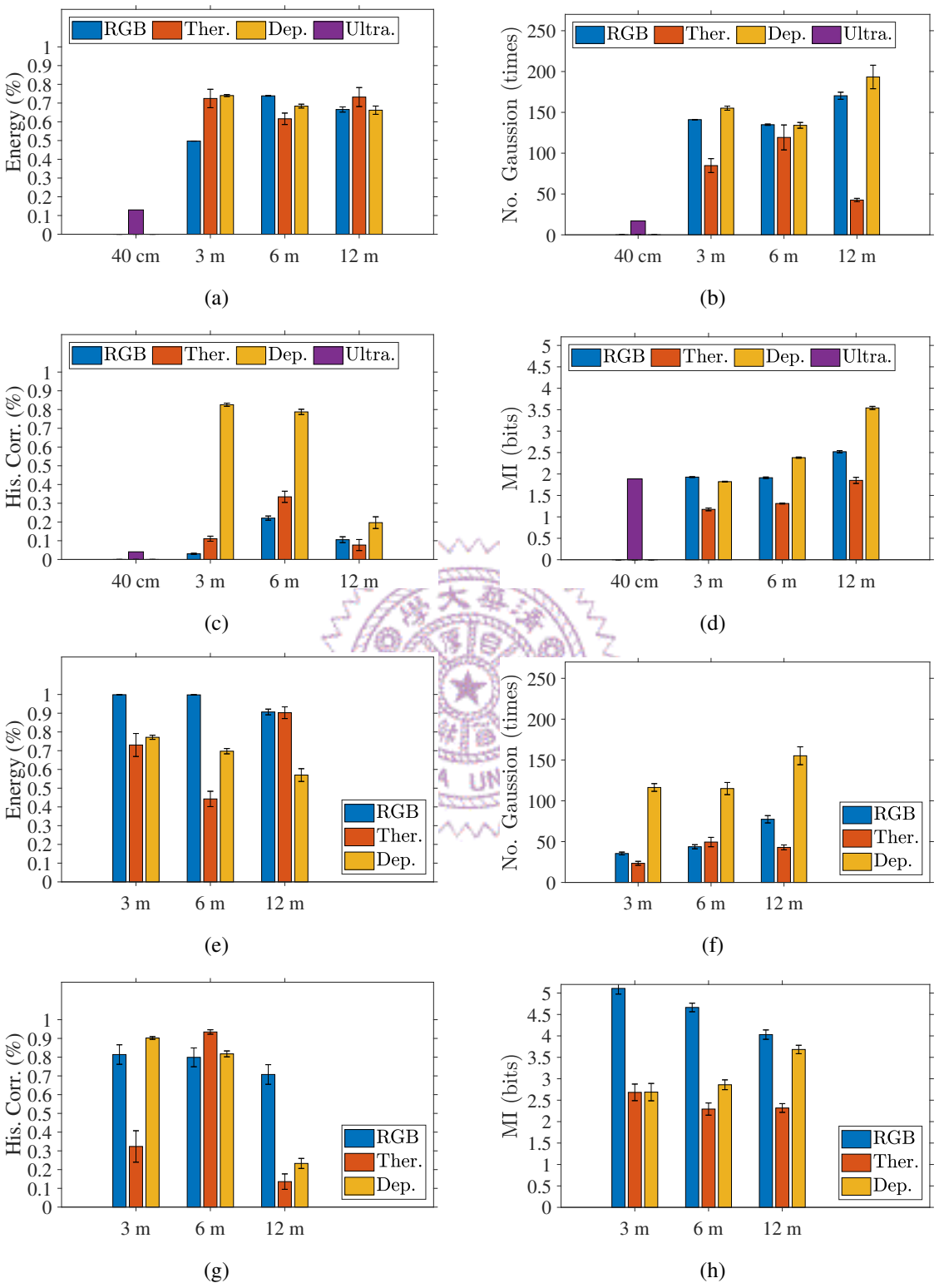


Figure 5.8: Distinguishability using different sensors on cropped windows at different distance. Different states : (a)–(d) and Same states : (e)–(h). With metrics ENG : (a), (e); NG : (b), (f); HC : (c), (g); and MI : (d), (h).

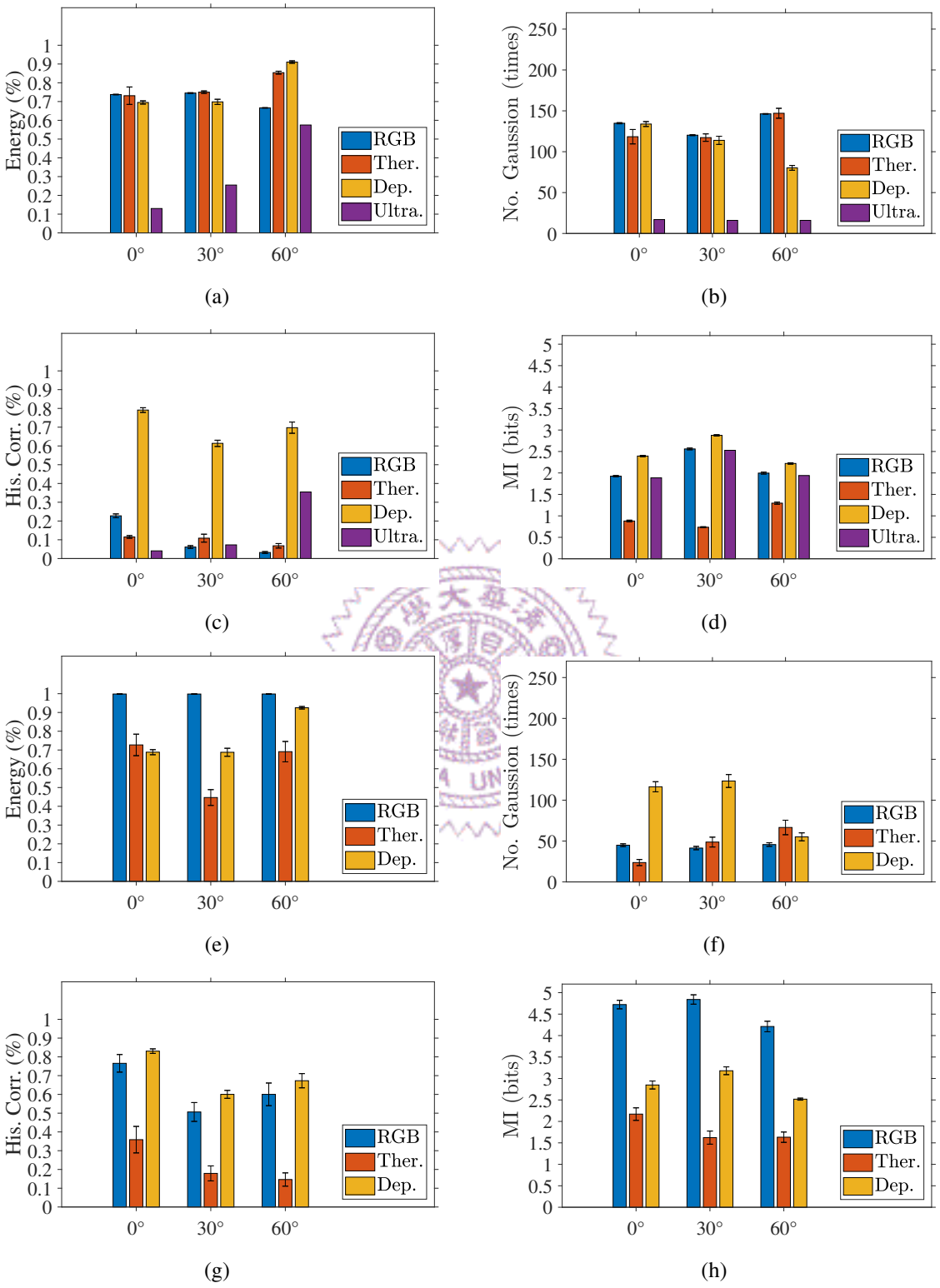


Figure 5.9: Distinguishability using different sensors on cropped windows at different distance. Different states : (a)–(d) and Same states : (e)–(h). With metrics ENG : (a), (e); NG : (b), (f); HC : (c), (g); and MI : (d), (h).

of images X and Y with different open-window states. We then compute their distinguishability. *In this case, a good sensor selection should show as high distinguishability as possible.* In addition, we also randomly select 100 pairs of images X and Y with the same open-window states (both open or close), and compute their distinguishability. *In this case, a good sensor selection should show as low distinguishability as possible.* We notice that for the 100 sample pairs, we report the average results with 95% confidence intervals in the figures. Last, we run the same experiments with *whole* and *cropped* images, where the whole images include both windows and surrounding walls and the cropped images only include the windows. Fig. 5.5 are examples of a whole image and a cropped image. Such cropping can be geometrically derived if the building blueprints and drone position/orientation are known.

Results. We use bar charts to represent our analysis. We choose one of the buildings in our dataset and make analysis. For whole images, we have Fig. 5.6 to show the distinguishability at different distance and Fig. 5.7 shows the distinguishability at different angles across our four sensors. For cropped images, we use Fig. 5.8 and Fig. 5.9 to show the results at different distances and angles. We talk about some general findings first, then share what we find about the open-window state detecting ability of each sensor in whole and cropped images, and discuss our analysis to other buildings and the best metrics for our chosen sensors. To prove that our analysis can generalize to other buildings, we also analyze the data from other buildings. We choose data at distance d = 3 m and azimuthal angle φ = 0 $^\circ$ from every buildings as our sample result and we plot them at Fig. 5.10.

We make the following observations:

- *ENG has bad performance on analyzing the distinguishability of our sensors.* The values of every sensor from the different state (Fig. 5.6(a)/Fig. 5.7(a)) are almost the same as the values from the same state (Fig. 5.6(e)/Fig. 5.7(e)). The discrimination of ENG is too low to use.
- *Depth sensor doesn't work for any situation.* Depth sensor gets almost the same values in the different/same state from our four metrics, so it shows depth poor ability at discriminating the open-window state.

Here are the findings of *whole images*:

- *RGB works the best at 3 m, thermal works the best at 6 m, and no one has a stable distinguishability at 12 m.* Although RGB doesn't get the lowest value at 3 m in Fig. 5.6(c) and Fig. 5.6(d), we find its difference between the value from the different state and the same state is the largest when comparing to Fig. 5.6(g) and Fig. 5.6(h). At 6 m, the differences of the thermal sensor in the different state (Fig. 5.6(b)/Fig. 5.6(c)) and the same state (Fig. 5.6(f)/Fig. 5.6(g)) are the largest.

When getting to 12 m, values from all sensors are almost equal at the same state and the different state.

- *Thermal works better than RGB when the sensor is at the large angles to the window.* From Fig. 5.7, we can see thermal sensor gets the largest difference from the different state and the same state. However, its ability to distinguish window is getting poor when the angle is getting larger from Fig. 5.7(b). When the angle is 60°, the values from the different state (Fig. 5.7(c)/Fig. 5.7(d)) become almost equal with the same state (Fig. 5.7(g)/Fig. 5.7(h)).

Here are the findings of *cropped images*:

- *RGB has the best performance at every distance.* From Fig. 5.8, all values from the RGB meet our expectation, and the margins are also the largest ones. It also works at 12 m, which differs from the whole images. Thermal works well at 3 m and 6 m, but its results worse than RGB and cannot work at 12 m. We believe is because the resolution of the sensor is too low to tell the difference of figures that are far away.
- *RGB works the best at different azimuthal angles.* From Fig. 5.9, we can see that RGB is the best to detect object in different angles. Its values change the most in the two experiment (same/different state), which is unlike to the whole images. We believe the reason is that cropped images remove the artifact from parts except for windows and make the metrics can focus on the window parts. Thermal works well at 0 $degree$ and 30 $degree$, but performance poor ar 60 $degree$. The data from Fig. 5.9 proves our findings.

Based on our findings to the difference of the different/same state, we can use RGB as our detecting sensor if having the blueprint of a building, and using thermal to detect open-window states when we are not familiar with the building and can not locally set up the position of windows.

- *Our findings are general.* Fig. 5.10 shows the sample results from different buildings in our dataset. Most of the result are the same as what we find above. It proves that our result can be applied to various kinds of windows.
- *MI is the most reliable to analyze the distinguishability of open-window state for RGB, and NG and HC are the better metrics for thermal.* We normalize the results from our four metrics into values within 0–1, and show the difference from the different/same state in Fig. 5.11. The metric that get the largest values in Fig. 5.11 is the best one for the sensor. We will use it as the criterion of our window state detector.

5.3.3 Open Window Detection

We design two open window detection algorithms using dataset B. Fig. 5.12 shows the high-level architecture of our open window detection algorithms. The inputs are the sen-

sor data, and the outputs are bounding boxes labeled with window states, i.e. open or close. The first step is *localization*, which either: (i) leverages existing image-based window detection algorithms [92, 94, 74, 179, 101] or (ii) combines the GPS coordinates with the building blueprints to localize the windows in sensor data. We assume the bounding boxes are given and crop the windows from the sensor data. For the second step, we propose two pipelines to determine if each window is open, which are detailed below.

Thermal Window Classification: TWC. Thermal sensors estimate temperature by measuring the infrared emitted from objects. When a window is close, we expect to observe consistent temperature from the window. In contrast, when the window is open, the temperature of the window (glass) would be quite different from that of the opening. Based on this, we develop our Thermal Window Classification (TWC) pipeline in Fig. 5.13. The first step is normalization. We normalize the raw thermal data in $[0, 65535]$ to $[0, 255]$. Next, we put the pixel values into a histogram with an empirically chosen bin size of 20. We then find all local minima of the histogram excluding the boundary bins. If there exists only one local minimum, we declare the window is open, since there are two groups of thermal readings.

Ultrasound Window Classification: UWC. Depth sensors employ infrared to detect the distance, which can go through transparent glass; in contrast, ultrasound sensors employ sound waves to detect glass, but its coverage cones are nontrivial to model. For example, Huang et al. [71] proposed to calibrate the coverage cone of their ultrasound sensor to the depth image. They then compute the average distance in the coverage cone on the depth image with the distance from the ultrasound sensor to determine if glass exists. We follow their procedure to compute a *theoretical coverage* each distance to an open window. We then move the center point of the ultrasound sensor around to derive an *actual coverage*. Unfortunately, we have found that the two coverages are quite different, i.e., we cannot compute the coverage cone simply using the FoV of our ultrasound sensor. Fig. 5.14 shows a sample result at 50 cm.

Hence, we ditch the depth sensor and develop an Ultrasound Window Classification (UWC) pipeline, which compares the distance returned by the ultrasound sensor m_u with the distance given in the window localization step d_u (e.g., using the GPS coordinates and building blueprint). The pipeline first checks the ratio between the window dimension and the distance to the building. It asks the robot, drone, or human to move closer if the distance is too long. This is followed by a disparity check: iff $m_u > d_u + \alpha$, we determine a window is close; here, α is a cushion parameter, which is empirically set to 20 cm if not otherwise specified. We note that because the actual coverage may run into window frame or wall easily, we propose to repeat the ultrasound measurement K times with different center points. We either control robots/drones or ask human to point

the ultrasound sensor at the chosen center points for individual measurements. Then, as long as one center point returns an opening window state, we declare the window is open. Fig. 5.14 shows a sample image with five center points, which are used in the rest of the paper. Fig. 5.16 summarizes the UWC pipeline.

Baseline algorithms. We have also implemented two existing algorithms serving as our baseline algorithms. Zheng et al. [190] analyzes RGB images to identify the lowest pixel intensity to divide pixels into two portions, corresponding to window and opening. However, they did not give details on how they identify the lowest pixel intensity. Therefore, we reuse our approach of finding local minima in TWC to complete this work. Huang et al. [71] considers depth and two ultrasound sensors to determine if there exists glass. We have found that two ultrasound sensors could interfere with each other. Hence, we only activate one ultrasound sensor throughout the experiments.

Results. Fig. 5.17 shows the overall performance of our two pipelines and two baseline algorithms in terms of accuracy, precision, recall, and F1-scores. We observe that the best performing pipeline is either our proposed TWC or UWC, except for the recall. In particular, our TWC leads to 16% lower recall than Zheng; while both Huang and UWC result in even lower recall values. The inferior recall performance of our TWC and UWC pipelines may be attributed to the limited resolution of thermal sensors (160×120) and ultrasound sensors (one reading per coverage cone). In contrast, for precision, Huang and UWC performs the best. In fact, they all get 100% precision, because if the ultrasound sensor detects glass, it is almost always correct.

Fig. 5.18 reports the performance at different distances. We make the following four key observations on this figure. First, the ultrasound based algorithms (Huang and UWC) perform worse at longer distance. For example, at 0.3 m, they both perform perfectly in accuracy (100% is achieved). However, our proposed UWC constantly outperforms Huang once the distance goes between 0.5 m. In terms of recall, Huang and UWC drops to close to 0 at 1 m. Second, the RGB/thermal based algorithms (Zheng and TWC) perform better at longer distance. Furthermore, our proposed TWC always outperforms Zheng in both accuracy and F1-score, while the performance gap increases as the distance is increased. For example, the F1-score of TWC is only 60% at 1 m, which is 9% lower than that of Zheng. However, the F1-score of TWC increases to 88.7% at 3 m, which is 22.7% larger than that of Zheng. **Based on this figure, we recommend the UWC pipeline for fine-grained detection ($\leq$ 40 cm), and the TWC pipeline for coarse-grained detection ($>$ 40 cm).**

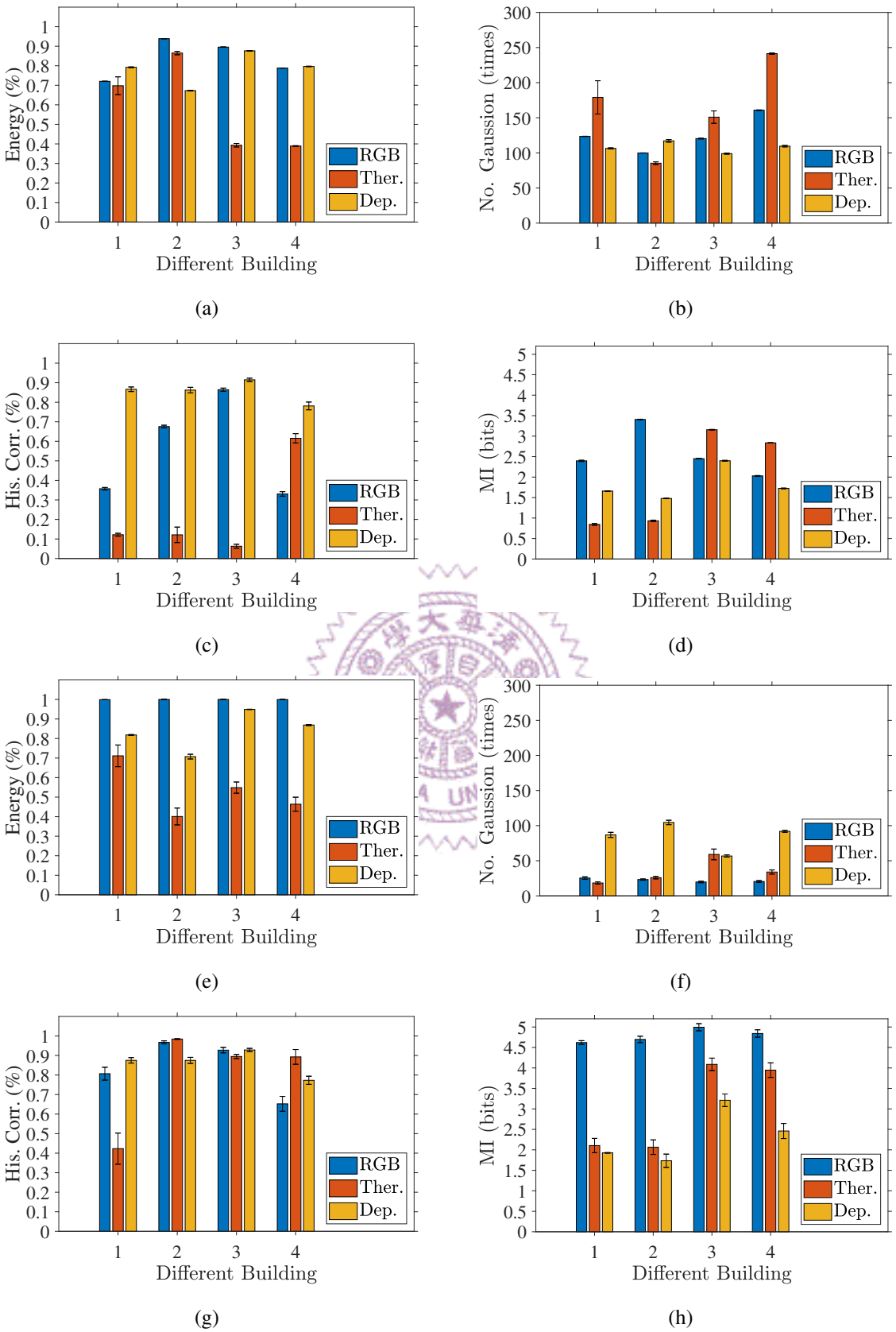


Figure 5.10: Distinguishability using different sensors on whole windows at different buildings. Different states : (a)–(d) and Same states : (e)–(h). With metrics ENG : (a), (e); NG : (b), (f); HC : (c), (g); and MI : (d), (h).

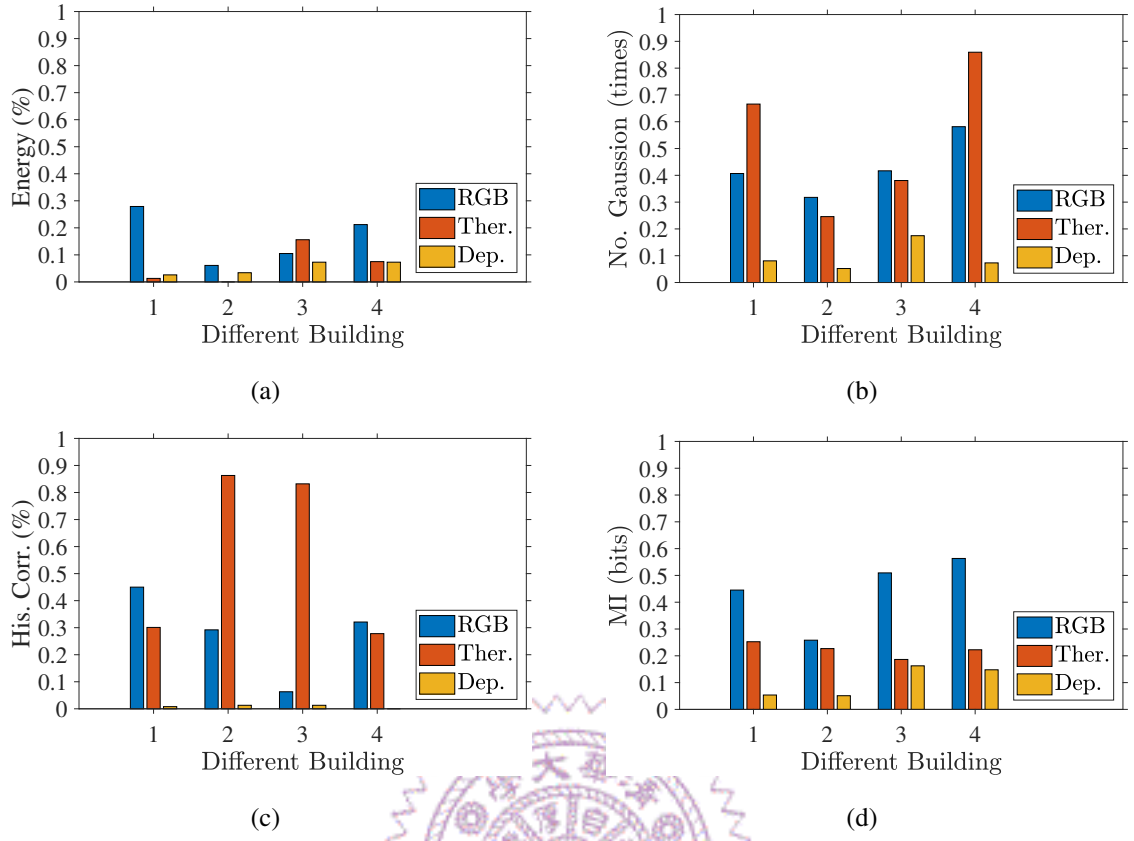


Figure 5.11: Distinguishability of metrics using different sensors on whole windows at different buildings. : (a) ENG, (b) NG, (c) HC, (d) MI.

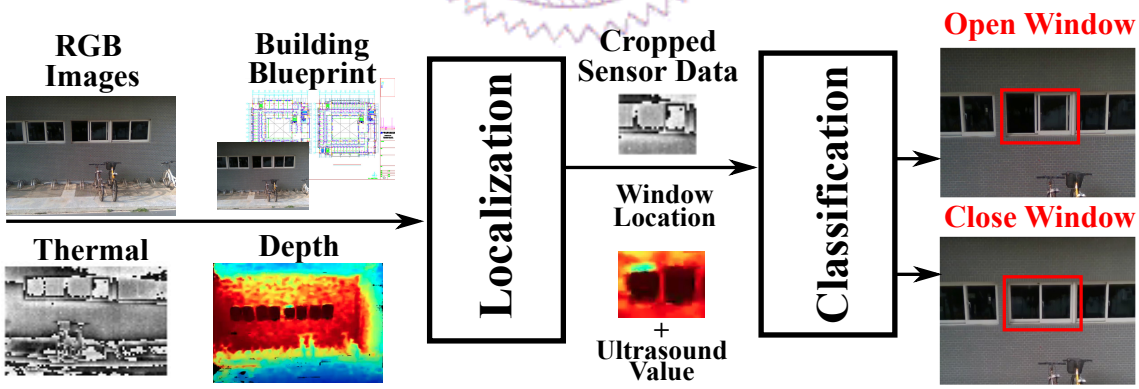


Figure 5.12: Open window detection in two steps.

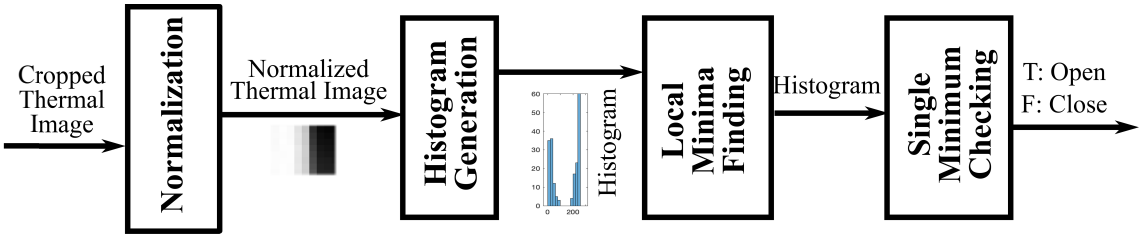


Figure 5.13: Our proposed TWC pipeline.

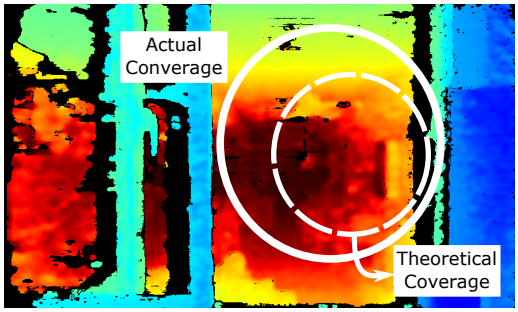


Figure 5.14: The nontrivial difference between theoretical and actual coverages of our ultrasound sensor at 50 cm.

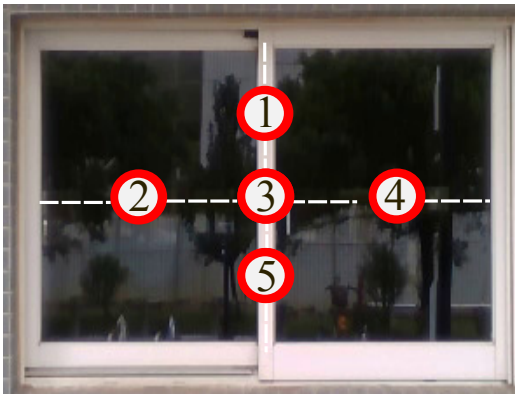


Figure 5.15: Sample center points used by the UWC pipeline.

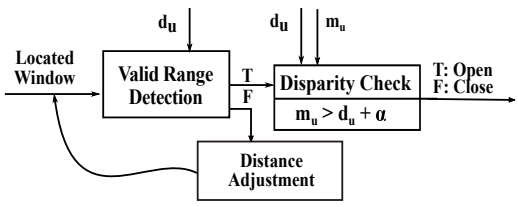


Figure 5.16: Our proposed UWC pipeline.

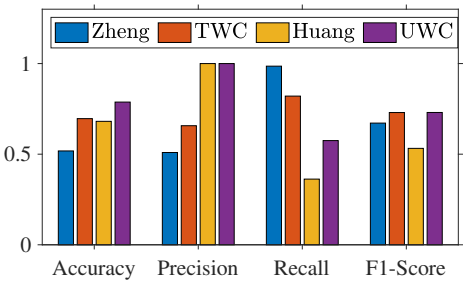


Figure 5.17: Overall performance comparisons across different pipelines and baseline algorithms.

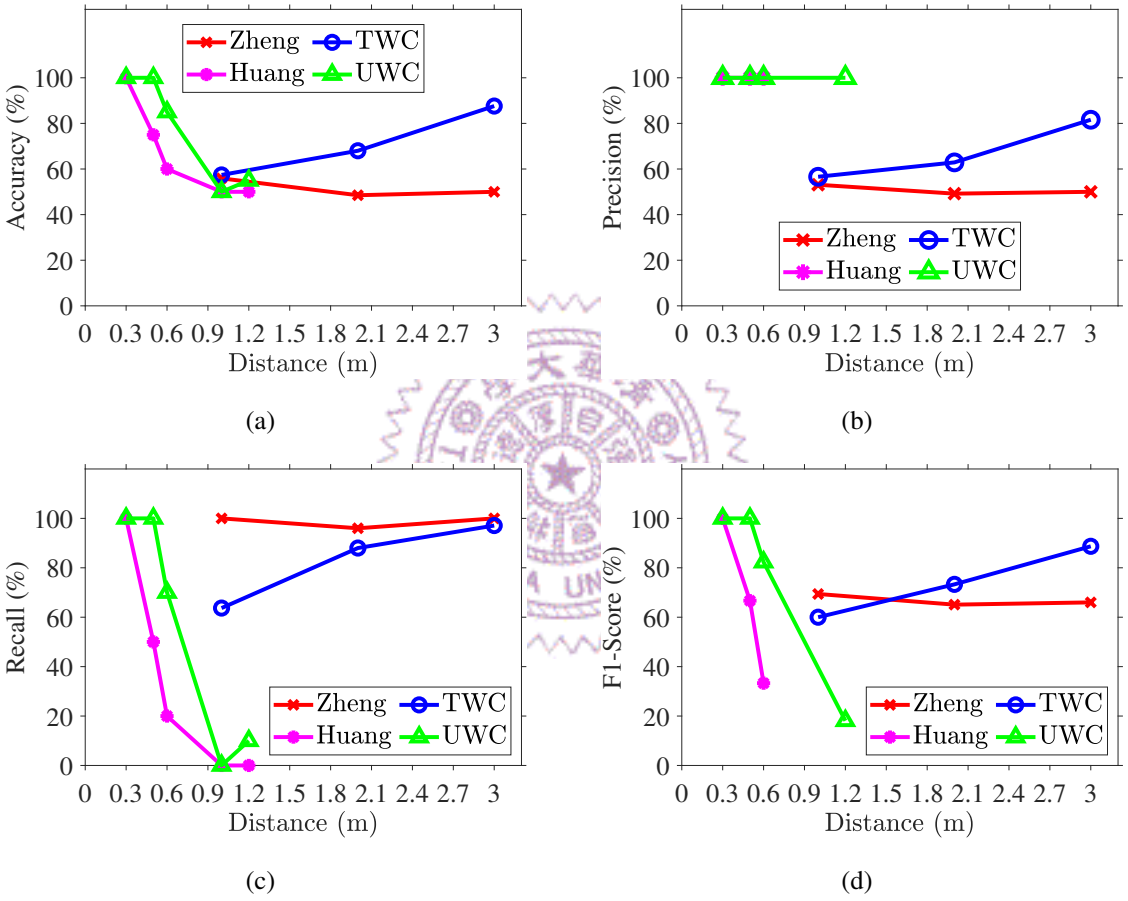
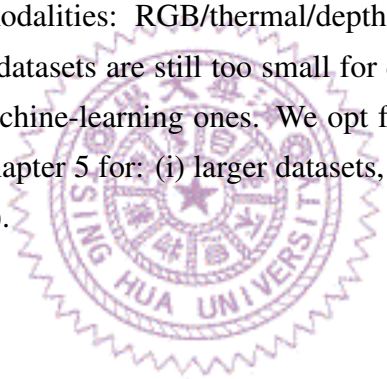


Figure 5.18: The performance of our two pipelines and two baseline algorithms at different distances: (a) accuracy, (b) precision, (c) recall, and (d) F1-score.

Chapter 6

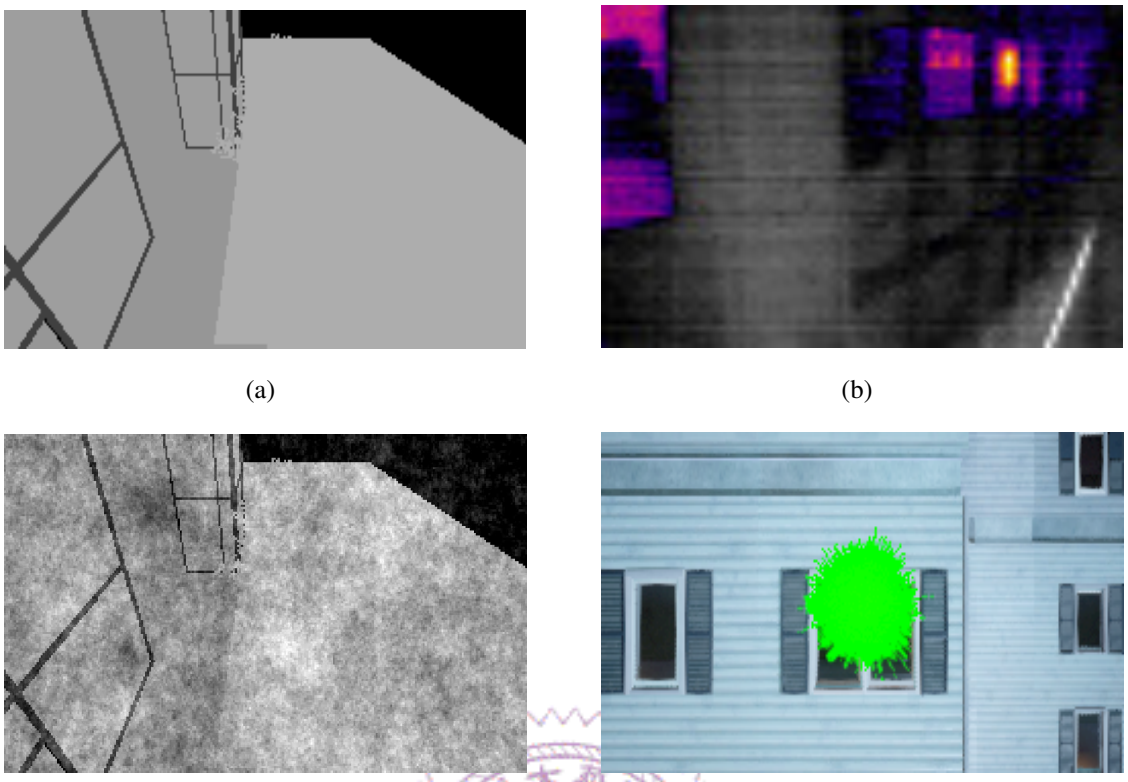
Synthesized Dataset Collection

Even though we created the first multi-modal window dataset in Chapter 5 with multiple (partially labeled) window states, such as openness, human behind, and light on/off, and employed five sensor modalities: RGB/thermal/depth cameras, LiDAR, and ultrasound sensors, the resulting datasets are still too small for developing window openness classifiers, especially the machine-learning ones. We opt for a photo-realistic simulator rather than a real setup as Chapter 5 for: (i) larger datasets, (ii) lower cost, and (iii) more diversity (like window types).



6.1 AirSim

We have decided to build a detailed simulator, implemented considered (virtual) sensors in it, and collect different sensor data under interested situations. After carefully surveying all open-source options, we have chosen AirSim [153] for its rich features. AirSim is an active project for drone simulations, which is essentially a plugin of Unreal Engine [48]. Unreal Engine offers a virtual, yet realistic, environment for drones to fly. Built on Unreal Engine, AirSim offers aerodynamic simulations and high-quality images (and other sensor data) through a suite of APIs (Application Programming Interfaces). It creates a virtual environment, in which multiple APIs, including drone control and sensor activation ones, are provided to applications. AirSim and Unreal communities have accumulated many diverse virtual environments that can be adopted. Leveraging on public 3D models [133], we have built several high-rise buildings in different styles and heights to collect a realistic dataset. The virtual environment adopts lightmap to simulate the reflection of window glasses. Fig. 4.2 illustrates sample views of our high-rise community.



(a) (b)

(c) (d)

Figure 6.1: Sample thermal images from: (a) an AirSim camera, (b) a FLIR camera, and (c) our enhanced AirSim camera; (d) detection area of our AirSim ultrasound sensor.

6.2 Sensor Implementations

We consider the vision and non-vision virtual sensors in AirSim, including RGB/thermal/depth cameras, LiDAR, and distance sensors. Depth cameras and LiDAR, however, do not work well for classifying window states [54]. In AirSim, a RGB camera can be configured following real cameras' specifications, such as Field-of-View (FoV). If not otherwise specified, we apply the specifications of Intel RealSense D435. A thermal camera in AirSim only considers infrared emitted from the objects for the sake of simplicity. More specifically, it gathers the temperature and emissivity of object materials, such as humans, trees, and elephants, to derive colors of individual segments to form thermal images. Fig. 6.1(a) shows such a sample thermal image in which each object has a uniform color (temperature). This is quite different from images from a real thermal camera, shown in Fig. 6.1(b). To address this issue, we have augmented the virtual thermal camera to introduce Gaussian and uniform noises to emulate the Johnson, flicker, and fixed-pattern noises [25]. Fig. 6.1(c) shows an improved thermal image captured by the augmented thermal camera, which is more realistic. The distance sensor in AirSim emits a ray to measure the distance of an object in front of it. This over-simplified design devi-

ates from real distance sensors, such as an ultrasound sensor, which analyzes the echoed sound waves bounced back from a *cone area* rather than a *single point*. Therefore, we have built a virtual ultrasound sensor in AirSim by introducing multiple rays covering a circular detection area. To make the detection area more realistic, we add a Gaussian noise to its radius, as illustrated in Fig. 6.1(d). The detection area is a function of the distance and detection angle. If not otherwise specified, we set the detection angle to be 15°, following the specifications of HC-SR04. Our ultrasound sensor returns the smallest distance among all considered rays.

Among these virtual sensors, we adopt the RGB camera as a representative one-shot sensor and the ultrasound sensor as a representative accumulative sensor in the rest of this paper for brevity. Other virtual sensors can be classified into one-shot or accumulative ones, and readily incorporated into the proposed solution.

6.3 Collection Procedure

Situation Generation. The situations we are interested in in this paper are open window and human number detection classifiers. Therefore, we make a script to control each windows of high-rise buildings in our virtual high-rise community. We can open and close all windows by pressing a button. For human number detection, we choose the human models created by Renderpeople [137] as our detection targets. Their human models are various with gender, dressing, and shape, which can test the abilities of our classifiers. We also make a script to generate a defined number of humans in a fixed area behind each windows. The generated human model and their locations are set randomly so that we can collect a human dataset with dissimilar images.

We target a 10-floor high-rise building for data collection. We select 20 windows (2 per floor), and define a bounding box with a width $x = 7.1$ m, a height $y = 5.4$ m, and a depth $z = 6.0$ m for each window. Outside of this bounding box, the RGB camera or ultrasound sensor cannot cover a good portion of the target window. For the RGB camera, we equally cut the bounding box into 10^3 candidate locations. For the ultrasound sensor, we place all candidate locations along the centered horizontal axis of each (horizontal sliding) window. We vary the number of locations per half-window among $\{1, 2, 3, 4\}$ and the distance to windows among $\{0.5, 1, 1.5, 2\}$ m. This leads to 16 location sets clustering around the center of the two halves of the window, where any two adjacent ones are apart by about 4 cm. To be more realistic, we also add some random noise to the locations of drones. The situations we collect includes open/close windows, human number from 0 to 6 and different light conditions (on/off). For each combination of the situations, we collect data ten times for each sensors and locations. Note that we use both

RGB and ultrasound sensor only when there is no people behind windows.

Segmentation Dataset. Unlike the real dataset, which need to label each image artificially, we collect a segmentation dataset to help our classifiers focus on the window part only. We change the materials of the window glasses that can be recognized by the segmentation function in AirSim. Then, we use the function to color our target window frames and window glasses, and make other objects into black. Finally, we use the same procedure as RGB sensors to collect the segmentation dataset.

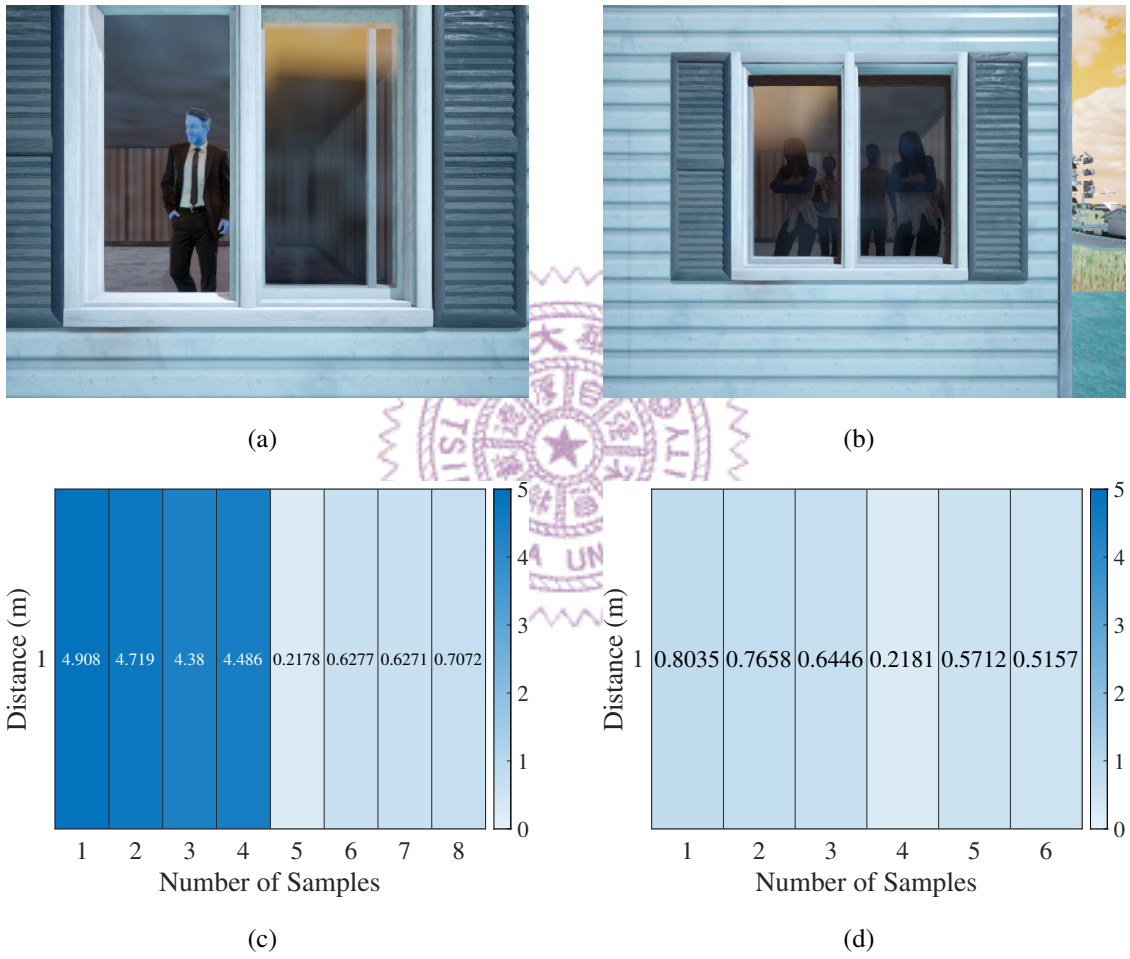


Figure 6.2: Sample RGB data from our realistic dataset: (a) `HR01_windowframe27_6_30_30_ow_p1_l0_00`, and (b) `HR01_windowframe15_12_30_30_cw_p5_l0_00`; sample ultraosund data from our realistic dataset with different candidate locations along the centered horizontal axis of a window: (c) `HR01_windowframe15_0.5_0_0_ow_p0_l0_00` with 4 candidate locations; (d) `HR01_windowframe15_0.5_0_0_cw_p0_l0_00` with 3 candidate locations.

6.4 Dataset Format

We use the similar naming convention as our real dataset. Take `HR01_windowframe15_1_0_0_cw_p1_l0_00` as an example. `HR01` and `windowframe15` indicate which building the target window is from, and the index of the target window in the building. The following three numbers represent the coordinate in the defined bounding box for RGB data, while for ultrasound data, `1` represents the distance (1 m) to windows and `0_0` shows that the ultrasound sensors collect data around the center of windows. These location messages are followed by close window `cw` (versus `ow`), one human behind `p1` (versus `p0`, `p2`, `p3`, `p4`, `p5`, `p6`), and lighting off `l0` (versus `l1`). The last element `00` specifies the serial number of the ten-times detection. Fig. 6.2 demonstrates sample results of our realistic dataset.

The files directory is organized into the following hierarchical structure:

- Sensor modality (such as `rgb` and `ultrasound`)
- Number of candidate locations along the centered horizontal axis of a window (we only have `numCanLoc1` for `rgb`, and `numCanLoc1` to `numCanLoc4` for `ultrasound`)
- Sensor data named with our defined naming convention

We do not separate our collected data into different window and situations in the realistic dataset for the simplification of data accessing. As we have more windows and states than the real dataset, our realistic dataset occupy about 2 TB space in total.

Chapter 7

Classifier Designs and Implementations

XXX high-level illustration to show the idea of classifiersXXX

We develop and implement several classifiers for open window and human number detection. In order to adapt to diverse environments in high-rise firefighting, we select and design our classifiers with different computation resources, analyzing time, and the accuracy performance.

7.1 Classifiers for Window Openness

Our classifiers are composed of two components: the *data processor* and *analyzer*. The data processors gather and clean the sensor data to meet the input requirements of the analyzers. Then, the analyzers determine the window states. The data processors of single-shot classifiers take one measurement of rich-media data, such as images. They crop the subject window out from the collected data[1], and apply image filters (such as color transform) required by the analyzers. The data processors of accumulative classifiers, on the other hand, combine all prior measurements at the current waypoint into rich-media data (images) using interpolation/extrapolation. By doing so, the same set of analyzers can also be leveraged.

Some classifiers may require sensor data from a *site survey*, which could be done at different granularities. For example, ultrasound sensors only collect distances, and thus may live with surveying a single window of the whole building. In contrast, RGB cameras are sensitive to the difference among windows (e.g., at different floors), and thus require surveying individual windows. Higher-value buildings, such as city halls, schools, and hospitals are wroth for finer-grained site surveys for higher classification accuracy. In addition to sensor modalities, some analyzers may also impose additional site survey

[1]We assume the window location/dimension are given by floor-plans (dictated for public buildings [145]) or vision-based localization approaches [120].

requirements. We propose the following sample analyzers to determine window openness:

- **Histogram [190].** The histogram classifier analyzes the pixel value distribution of the collected sensor data, and does not require any window data from the planning phase. We observe that the pixel values are quite similar when the window is close, while the pixel values between the glass area and the opening are very different when the window is open. Based on this feature, we generate a histogram for the collected sensor data first. If we only find a local minimum in the histogram, then the histogram classifier judge it as an open window.
- **Sum of Absolute Differences (SAD) [189].** SAD is method that people calculate the similarity between two images by pixel value difference. Even though the collected data may not be images from our selected sensors, we still can compare the value difference between them. We need a pair of window sensor data (an open one and a close one) from the planning phase. These two sensor are our ground truth. We compare the sensor data collected by drone with our ground truth value by value. If the error to the open window ground truth is smaller, we can take the collected data with open window situation. However, SAD requires the comparing data to match one by one. If the number of values are different, or the values are pointing to different places, SAD may not react the real error between two data.
- **Oriented Fast and Rotated Brief (ORB) [143].** The principle of the ORB classifier is also an image similarity comparison. We collect a pair of window sensor data (an open one and a close one) from the planning phase as our ground truth, and compare the sensor data collected by drone with our ground truth. The difference from the SAD is that ORB extracts features between two comparing data, like edges, or corners, then match the features between two comparing data, and calculate the match rate as their similarity. If the collected data has higher similarity to the open window sensor data, we judge it as an open window. ORB can fix the problem we just mentioned above, but it can not work on sparse data as ORB may not find any features on it.
- **Support Vector Machine (SVM) [124].** SVM is a type of supervised learning model. It could find a best hyperplane to separate the input labeled training data into two groups, which meet the requirements of our open window classification. We first label our window images into openness and closeness, and extract the features of the target windows, such as the shapes [96], texture [161], and the color distribution [31]. Then, we train the SVM model to find a hyperplane to classify the images.
- **Random Forest (RF) [129].** There are many decision trees in the RF. Each tree take different features to make the decisions. Then, we select the decision agreed

by most of the trees. We generate the same features as SVM from our sensor data in RF, and train it to classify window states.

We have implemented and tested four sample classifiers for each of the representative sensor modalities: the RGB camera and ultrasound sensor. We avoid the combination of the RGB camera and SAD analyzer, because SAD is very sensitive to minor spatial displacement. We also skip the combination of the ultrasound sensor and ORB analyzer, because images interpolated/extrapolated from ultrasound measurements are smooth, and thus featureless.

7.2 Classifiers for Human Detection

The topic of human detection has already been researched for many years. There is no need to develop a unique one by ourselves. Most of the human detection classifiers include three steps [47]: (i) region of interest (ROI) selection, predicting areas on the images that might be humans, (ii) classification, classifying if the ROI is human, and (iii) tracking, finding the moving trace of the detected humans in a series of images. For each step, there are numerous strategies, we selected several existing work with diverse computation resources, analyzing time, and accuracy performance as our human number detection classifiers.

XXX the human classifiers we selected XXX

7.3 Classifier Certainty and Accuracy

The performance of classifiers in our system can be quantified in two different metrics: *certainty* and *accuracy*. Here, certainty λ captures how confident when a classifier determines the window state. Take ORB analyzer as an example, its certainty could be the fraction of matched features. Certainty, by nature, is short-term: each measurement gives a certainty. In contrast, accuracy a describes the long-term performance of a classifier: what the fraction of correctly classified states is. The long-term accuracy has been shown to be a function of short-term certainty [15, 169]. We therefore model accuracy as a function of certainty, i.e., $a = R(\lambda)$, which can be built in various ways. In this thesis, we adopt a regression model $R(\lambda)$ for each location, sensor, and classifier, using a large dataset collected from our simulator. With these models, we get to map classifier certainty to its accuracy at run-time. Last, we notice that the definition of accuracy can be generalized to the results from multiple classifiers in a measurement sequence, which is detailed in the next section.

Chapter 8

Measurement Selection Problems

We need to consider which sensors, classifiers and measurement locations for drones when the drones arrive waypoints. The drones could conduct multiple measurements, which we name them a measurement sequence, at a waypoint to get as high detection accuracy as possible under a time budget. In this chapter, we discuss the measurement selection problem, and propose algorithms to solve the problem.

8.1 Notations

A measurement sequence consists of a series of measurement $m_i = (p_i, s_i, c_i)$, specifying the location p_i, sensor s_i, and classifier c_i. While the selections of sensors and classifiers are discrete by nature, the selections of locations can be either continuous or discrete. We opt to discretize the locations for two reasons. First, selections of continuous locations lead to higher site survey and certainty/accuracy modeling overhead. Second, some sensor modalities only work at very few locations. For instance, an ultrasound sensor emits sound wave within a cone, which should completely fall on the window glass (without hitting nearby wall). Considering the properties of sensors and classifiers, we define a set of *measurement candidates* $\mathbf{M} = \{m_1, m_2, \ldots, m_M\}$. Given m_i, we can compute its expected accuracy $a_i = \Theta(m_i)$, sensing time $t_i^s = \Omega(s_i)$, and classification time $t_i^c = \Phi(s_i, c_i)$, where $\Theta(\cdot)$, $\Omega(\cdot)$, and $\Phi(\cdot)$ are empirically derived functions. We can estimate the data transfer time t_i^t by the bandwidth and data size.

The measurement selection problem computes a measurement sequence $\mathbf{L} = (m_{l(1)}, m_{l(2)}, \ldots, m_{l(L)})$, where $l(i) \in [1, M]$, so that the accuracy of the final result *fused* from the outputs of all L measurements can reach a user specified target $\hat{A}$ within a time limit $\hat{T}$. Since high-rise fire scenes are dynamic, our system *recomputes* a measurement sequence after completing *every* measurement. Doing so allows us to: (i) check if the achieved fused accuracy exceeds $\hat{A}$, and exits this waypoint earlier, and (ii) revise the

remaining measurement sequence to adapt to the fire-scene dynamics. To facilitate the *recurring* re-computations, for each execution of our measurement selection algorithms, we use $\mathbf{L}'$ to denote the completed measurement sequence and $\mathbf{M}'$ to denote the set of unselected measurement candidates.

XXX a block diagram XXX

8.2 Fusing the Measurement Results

Suppose we get measurement results from L binary classifiers in $\mathbf{L}$, whose accuracy levels are $a_{l(1)}, a_{l(2)}, \ldots, a_{l(L)}$. We propose two policies to fuse the measurement results from L for the binary classifiers and multi-class classifiers individually.

For the results from binary classifiers, we use '1' and '0' to represent the output of the classifiers. We define all measurement results from $\mathbf{L}$ as $\mathbf{R}(\mathbf{L}) = \{\mathbf{L}_1, \mathbf{L}_0\}$, where $\mathbf{L}_1 \subseteq \mathbf{L}$ indicates the measurements with result '1', and $\mathbf{L}_0 = \mathbf{L} \setminus \mathbf{L}_1$ represents the measurements with result '0'. We use $\hat{R}(\mathbf{R}(\mathbf{L}))$ to represent the fused result from all measurement results, and following are the two proposed policies to derive it:

Majority vote. This policy adopts the majority results, i.e., $\hat{R}(\mathbf{R}(\mathbf{L})) = \arg\max_{i=\{0,1\}} |\mathbf{L}_i|$. This policy is simple as it does not take the individual measurement accuracy into consideration. However, L must be odd to avoid tie breaking.

Probability-based. We compute the final classification result using all measurement accuracy. More specifically, we define $P(1|\mathbf{R}(\mathbf{L}))$ and $P(0|\mathbf{R}(\mathbf{L}))$ to be the probabilities of results '1' and '0' being true, given $\mathbf{L}$. We then go with the one with higher probability, i.e., $\hat{R}(\mathbf{R}(\mathbf{L})) = \arg\max_{i\in\{0,1\}} P(i|\mathbf{R}(\mathbf{L}))$. We concretize these two probabilities in the following theorem.

Theorem 1. *The probability that '1' being true equals to:*

$$P(1|\mathbf{R}(\mathbf{L})) = \frac{\prod_{m_i\in\mathbf{L}_1} a_i \prod_{m_j\in\mathbf{L}_0}(1-a_j)}{\prod_{m_i\in\mathbf{L}_1} a_i \prod_{m_j\in\mathbf{L}_0}(1-a_j) + \prod_{m_i\in\mathbf{L}_1}(1-a_i)\prod_{m_j\in\mathbf{L}_0} a_j}$$

and the probability of '0' being true equals to $P(0|\mathbf{R}(\mathbf{L})) = 1 - P(1|\mathbf{R}(\mathbf{L}))$.

Proof. We define $P(\mathbf{R}(\mathbf{L}))$ as the probability that we get results $\mathbf{R}(\mathbf{L})$ which may happen in two conditions. One condition is when the measurement results indexed by $\mathbf{L}_1$ are correct (and the others are wrong), which occurs with probability $P(c1) = \prod_{m_i\in\mathbf{L}_1} a_i \prod_{m_j\in\mathbf{L}_0}(1-a_j)$. Another condition is when measurements with indexes $\mathbf{L}_1$ are incorrect, with probability $P(c2) = \prod_{m_i\in\mathbf{L}_1}(1-a_i)\prod_{m_j\in\mathbf{L}_0} a_j$. We have $P(\mathbf{R}(\mathbf{L})) = P(c1) + P(c2)$. As '1' being true is equivalent to the first condition happens given results $\mathbf{R}(\mathbf{L})$, we have $P(1|\mathbf{R}(\mathbf{L})) = P(c1)/P(\mathbf{R}(\mathbf{L}))$. □

For the results from the multi-class classifiers, XXX **Majority vote.** XXX **Probability-based.**XXX

8.3 Formulation

Our problem computes a complete measurement sequence $\mathbf{L}$ before the time limit $\hat{T}$ so that the *expected accuracy* $A(\mathbf{L})$ can reach $\hat{A}$. Here, $\mathbf{L}$ is composed of the prior measurement sequence $\mathbf{L}'$ and the following measurement sequence $\hat{\mathbf{L}}$. It is not hard to see that a Traveling Salesman Problem (TSP) can be reduced to our problem in polynomial time. In a special case of our problem, we set an uniform measurement accuracy > 0.5 and $\hat{T} = \infty$. Based on the condorcet's jury theorem [84], more measurements with accuracy larger than 0.5 could make the final accuracy close to 1, which is called the power of public. Then, under this setup, the measurement selection problem becomes to use the least time to reach all measurement candidates to get the largest accuracy, which is the same as the concept of TSP problem. Hence, our problem is NP-hard.

The precise definition of $A(\mathbf{L})$ depends on the adopted fusing policy and the type of the classifiers (binary or multi-class). The following are the two definitions of $A(\mathbf{L})$ for the binary classifiers. With the majority vote policy, we define $A_M(\mathbf{L})$ by summing up the probability of all measurement results, where more than half of them are correct. We assume that the accuracy from each classifier is independent, and write:

$$A_M(\mathbf{L}) = \sum_{k=\lceil L/2 \rceil}^{L} \sum_{\substack{\mathbf{L}_1 \subseteq \mathbf{L}, |\mathbf{L}_1|=k, \\ \mathbf{L}_0 = \mathbf{L} \setminus \mathbf{L}_1}} \Big(\prod_{m_i \in \mathbf{L}_1} a_i \prod_{m_j \in \mathbf{L}_0} (1 - a_j) \Big). \tag{8.1}$$

In this equation, a_i represents: (i) the expected accuracy from long-term average models $\Theta(m_i)$ if the measurement belongs to $\hat{\mathbf{L}}$, and (ii) the mapped accuracy from short-term certainty models $R(\lambda_i)$ if the measurement belongs to $\mathbf{L}'$.

Suppose the drone gets results $\mathbf{R}(\mathbf{L}') = \{\mathbf{L}'_1, \mathbf{L}'_0\}$ from its prior measurements. We define $A_P(\mathbf{L})$ as expected accuracy gotten by $\mathbf{L} = (\mathbf{L}', \hat{\mathbf{L}})$ following the probability-based policy, and set $P(1|R(\emptyset)) = 0.5$ if $\mathbf{L}' = \emptyset$. The following theorem derives the fused accuracy.

Theorem 2. *We can compute* $A_P(\mathbf{L})$ *by:*

$$A_P(\mathbf{L}) = \sum_{\substack{\hat{\mathbf{L}}_1 \subseteq \hat{\mathbf{L}}, \\ \hat{\mathbf{L}}_0 = \hat{\mathbf{L}} \setminus \hat{\mathbf{L}}_1}} \max \Bigg(P(1|\mathbf{R}(\mathbf{L}')) \prod_{m_i \in \hat{\mathbf{L}}_1} a_i \prod_{m_j \in \hat{\mathbf{L}}_0} (1 - a_j), \\ (1 - P(1|\mathbf{R}(\mathbf{L}'))) \prod_{m_i \in \hat{\mathbf{L}}_1} (1 - a_i) \prod_{m_j \in \hat{\mathbf{L}}_0} a_j \Bigg). \tag{8.2}$$

Proof. Given $\hat{\mathbf{L}}$, we have $2^{|\hat{\mathbf{L}}|}$ result combinations by enumerating all subsets $\hat{\mathbf{L}}_1 \subseteq \hat{\mathbf{L}}$ and letting $\mathbf{R}(\hat{\mathbf{L}}) = \{\hat{\mathbf{L}}_1, \hat{\mathbf{L}}_0\}$ with $\hat{\mathbf{L}}_0 = \hat{\mathbf{L}} \setminus \hat{\mathbf{L}}_1$ and $\mathbf{R}(\mathbf{L}) = \{\mathbf{L}'_1 \cup \hat{\mathbf{L}}_1, \mathbf{L}'_0 \cup \hat{\mathbf{L}}_0\}$. For each

result combination $\mathbf{R}(\mathbf{L})$, we can compute the probability of '1' being true by:

$$P(1|\mathbf{R}(\mathbf{L})) = \frac{P(1|\mathbf{R}(\mathbf{L}'))\prod_{m_i\in\hat{\mathbf{L}}_1} a_i \prod_{m_j\in\hat{\mathbf{L}}_0}(1-a_i)}{P(1|\mathbf{R}(\mathbf{L}'))\prod_{m_i\in\hat{\mathbf{L}}_1} a_i \prod_{m_j\in\hat{\mathbf{L}}_0}(1-a_i) + (1-P(1|\mathbf{R}(\mathbf{L}')))\prod_{m_i\in\hat{\mathbf{L}}_1}(1-a_i)\prod_{m_j\in\hat{\mathbf{L}}_0} a_j},$$

following Theorem 1. Results $\mathbf{R}(\mathbf{L})$ occur in two possible conditions: all measurements with result '1' are correct or wrong. The occurrence probability of these two conditions $P(\mathbf{R}(\mathbf{L}))$ equals to the denominator in the above equation for $P(1|\mathbf{R}(\mathbf{L}))$. Next, we derive the expected accuracy by summing up the products of all possible $\mathbf{R}(\mathbf{L})$'s occurrence probability and its fused accuracy, i.e.,

$\sum_{\substack{\hat{\mathbf{L}_1}\subseteq\hat{\mathbf{L}},\ \hat{\mathbf{L}_0}=\hat{\mathbf{L}}\setminus\hat{\mathbf{L}}_1 \\ \mathbf{R}(\mathbf{L})=\{\mathbf{L}'_1\cup\hat{\mathbf{L}}_1,\mathbf{L}'_0\cup\hat{\mathbf{L}}_0\}}} P(\mathbf{R}(\mathbf{L}))\times\max(P(1|\mathbf{R}(\mathbf{L})),1-P(1|\mathbf{R}(\mathbf{L})))$, which can be simplified to Eq. (8.2). Here we get the maximum from the probabilities of '1' and '0', as the probability-based policy believes in the higher-probability one. □

As for the multi-class classifiers, we also define two definitions for the expected accuracy $A(\mathbf{L})$:

Majority vote.XXX

Probability-based.XXX

Next, we use $T(\mathbf{L})$ to indicate the total time of finishing all measurements in $\mathbf{L}$, which equals to:

$$T(\mathbf{L}) = \sum_{i=1}^{L}(t^s_{l(i)} + t^c_{l(i)} + \frac{|p_{l(i)} - p_{l(i-1)}|}{V} + t^a_{l(i)} + t^t_{l(i)}), \tag{8.3}$$

where $t^s_{l(i)}$, $t^c_{l(i)}$, and $t^t_{l(i)}$ are the sensing, classification, and data transfer times of $m_{l(i)}$, respectively. The third term in the summation is the moving time. Here, V represents the flying speed and $p_{l(0)}$ is the drone's initial location at the current waypoint. The fourth term $t^a_{l(i)}$ is the measurement selection algorithm's running time.

With the above derived functions, we write our measurement selection problem as:

$$\max_{\mathbf{L}}\ U(A(\mathbf{L}),T(\mathbf{L})) = \sqrt{(1-e^{-\alpha A(\mathbf{L})})\times e^{-\beta\frac{T(\mathbf{L})}{\hat{T}}}} \tag{8.4a}$$

$$\text{s.t.}\quad A(\mathbf{L})\geq\hat{A};\quad T(\mathbf{L})\leq\hat{T}. \tag{8.4b}$$

Here, the objective function $U(A(\mathbf{L}),T(\mathbf{L}))$ in Eq. (8.4a) aims to find a good trade-off between the accuracy and time, controlled by weights α and β, which are system parameters. The utility value increases when: (i) the accuracy increases or (ii) the time decreases. We normalize $T(\mathbf{L})$ by $\hat{T}$ to keep it between 0 to 1. Notice that, our proposed algorithms work with utility functions other than the one in Eq. (8.4a), as no mathematical properties are assumed. For example, a weighted sum of $A(\mathbf{L})$ and $T(\mathbf{L})$ could be adopted as the utility function. The constraints in Eq. (8.4b) guarantee that $A(\mathbf{L})$ is higher than the target accuracy $\hat{A}$, and $T(\mathbf{L})$ is less than the time limit $\hat{T}$. We emphasize that our formulation does not specify the length of $\mathbf{L}$, i.e., L is also an output of our problem. Last, we note

that by adopting either $A_M(\mathbf{L})$ or $A_P(\mathbf{L})$, the same formulation works for either majority vote or probability-based fusion.

8.4 Proposed Algorithms

Algorithm 1 Our Optimized Algorithm (OPT)

function EXHAUSTIVE SEARCH($\mathbf{M}'$, $\hat{\mathbf{L}}$, $\mathbf{L}'$)
  for every $m_i \in \mathbf{M}'$ **do**
    $\hat{\mathbf{L}_{tmp}} \leftarrow \hat{\mathbf{L}} \cup \{m_i\}$ ▷ Add m_i to $\hat{\mathbf{L}}$
    $\mathbf{L}_{tmp} \leftarrow \mathbf{L}' \cup \hat{\mathbf{L}_{tmp}}$ ▷ Concatenate the visited $\mathbf{L}'$ and $\hat{\mathbf{L}_{tmp}}$
    if $T(\mathbf{L}_{tmp}) < \hat{T}$ **then** ▷ Check if $\mathbf{L}_{tmp}$ can make the deadline
      $Utility_{tmp} \leftarrow U(A(\mathbf{L}_{tmp}), T(\mathbf{L}_{tmp}))$
      $M'_{tmp} \leftarrow \mathbf{M}' \setminus \{m_i\}$
      $Utility_{new}, \hat{\mathbf{L}}_{new} \leftarrow ExhaustiveSearch(\mathbf{M}'_{tmp}, \hat{\mathbf{L}_{tmp}}, \mathbf{L}')$ ▷ Go over every measurement sequence
      if $Utility_{new} > Utility_{tmp}$ **then** ▷ Check if adding other measurements can get larger utility
        $\hat{\mathbf{L}} \leftarrow \hat{\mathbf{L}}_{new}$
        $Utility \leftarrow Utility_{new}$
      else
        $\hat{\mathbf{L}} \leftarrow \hat{\mathbf{L}}_{tmp}$
        $Utility \leftarrow Utility_{tmp}$
      end if
    end if
  end for
  return $Utility, \hat{\mathbf{L}}$
end function

The measurement selection problem may be solved optimally using exhaustive search. We refer to the optimal algorithm as OPT, which serves as a benchmark. Algo. 1 demonstrates the pseudo code of our exhaustive search algorithm. We consider every element in the set of unvisited measurement candidates $\mathbf{M}'$. Every time we add one element to $\hat{\mathbf{L}_{tmp}}$, and check if the time consumption of the $\mathbf{L}_{tmp}$ will still smaller than the time budget $\hat{T}$. If yes, we calculate the fusion accuracy and utility of the $\mathbf{L}_{tmp}$. Note that your could choose $A_M(\cdot)$ $A_P(\cdot)$ to get the fusion accuracy based on your requirements. Then, We recursive the function and try to go through every combination of m_i in $\mathbf{M}'$. Finally, we return the $\hat{\mathbf{L}}$ with the largest value from the utility function. However, if the candidate measurement

set M' is large, we need to take exponential time to check every possibility.

Algorithm 2 Our Heuristic Algorithms (HEU_M, HEU_P)

Require: $\mathbf{M}'$, $\mathbf{L}'$, $A(\mathbf{L}') < A'$, $T(\mathbf{L}') < \hat{T}$
  Initialize: $Utility \leftarrow U(A(\mathbf{L}'), T(\mathbf{L}'))$, $\hat{\mathbf{L}} \leftarrow \emptyset$
2: **for** every $m_i \in \mathbf{M}'$ **do**
    $\hat{\mathbf{L}_{tmp}} \leftarrow \hat{\mathbf{L}} \cup \{m_i\}$ ▷ Add m_i to $\hat{\mathbf{L}}$
4: $\mathbf{L}_{tmp} \leftarrow \mathbf{L}' \cup \hat{\mathbf{L}_{tmp}}$ ▷ Concatenate the visited $\mathbf{L}'$ and $\hat{\mathbf{L}_{tmp}}$
    if $T(\mathbf{L}_{tmp}) < \hat{T}$ **then** ▷ Check if $\mathbf{L}_{tmp}$ can make the deadline
6: $Utility_{tmp} \leftarrow U(A(\mathbf{L}_{tmp}), T(\mathbf{L}_{tmp}))$
    if $Utility_{tmp} > Utility$ **then** ▷ Find the m_i generating the largest utility
8: $Utility \leftarrow Utility_{tmp}$
    $\hat{\mathbf{L}} \leftarrow \hat{\mathbf{L}_{tmp}}$
10: **end if**
    end if
12: **end for**

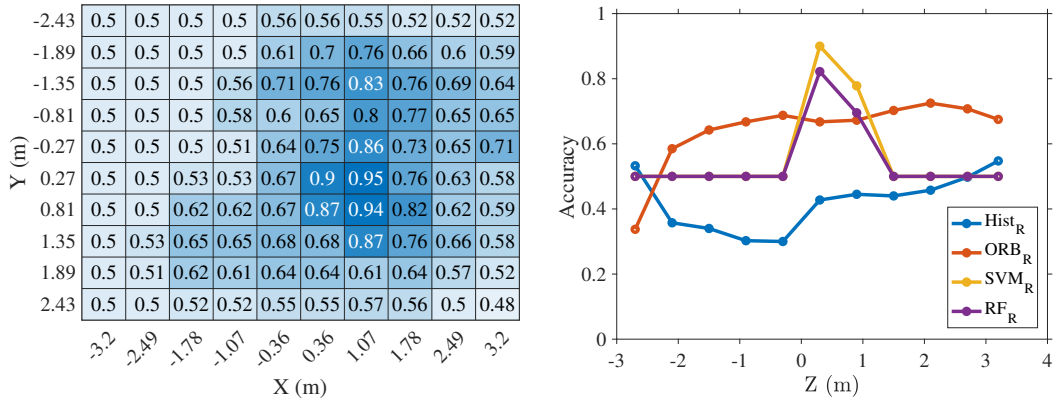
For real-time measurement selections, we propose an efficient heuristic algorithm that greedily adds the measurement that leads to the highest utility function value increase. We repeat the process until: (i) the time runs out, (ii) the target accuracy is achieved, or (iii) the utility function value can not be further improved. We call this algorithm as *HEU*, which comes with two variants: HEU_M and HEU_P depending on which usages of the two accuracy fusion policies. Algo. 2 explains the details of our algorithms. We first check if the existing $\mathbf{L}'$ is already over the time limit $\hat{T}$or reaches the target accuracy $\hat{A}$. If no, we then use our algorithms to find and add a better measurement into the original measurement sequence. We check every measurement m_i in $\mathbf{M}'$ iteratively. If adding a m_i may not exceed $\hat{T}$, we then calculate the utility and check if the utility is larger than the current best utility. If yes, we change the best utility to the new one and update the $\hat{\mathbf{L}_{tmp}}$. After the iteration, we could find a measurement sequence with the greatest utility for the next step. Even though it may not find the best measurement sequence because of considering one new measurement only, HEU_M and HEU_P could still provide satisfied utility with fast response time.

Chapter 9

Performance Evaluations

9.1 Implementations

In our high-rise community in AirSim, we chose the same 10-floor high-rise building used for dataset collection in Chapter 6 to conduct our evaluations. This building consists of 56 horizontal sliding windows. We separate them into two groups: (i) 20 windows for site surveying (2 per floor), which are collected in Chapter 6 and (ii) 36 windows for evaluations (half of them are open)[1]. We use the dataset as the site surveying to construct look-up tables for expected accuracy $\Theta(\cdot)$ for each candidate location we set in the bounding box using different sensors/classifiers, as well as sensing time $\Omega(\cdot)$ and classification time $\Phi(\cdot)$. We also model the certainty to accuracy mapping $R(\cdot)$ as linear functions. These tables/models are used as inputs by our algorithms and simulator.



(a) (b)

Figure 9.1: Sample accuracy: (a) SVM an RGB ($Z = 0.03$ m) and (b) classifiers on RGB at various Z's (centered).

Our proposed classifiers achieve good accuracy after being trained with the 20 site-

[1]Real site surveys and measurements can be done if cost is not a concern.

surveyed windows. We show sample accuracy results in Fig. 9.1. Fig. 9.1(a) reveals that the SVM classifier works well with an RGB camera, especially at the center of individual windows. Fig. 9.1(b) reports the performance of all four classifiers with an RGB camera at different depth (while X and Y are centered). As we train our classifiers at the central point of the bounding box, the accuracy gets better when close a location to center. An exception is the histogram classifier, which requires no training (site surveying). Because none of the classifiers/sensors perform well at all locations, our measurement selection algorithm that fuses multiple measurements is crucial to achieve high accuracy.

We have built an event-driven simulator using C++. For each target window, we first invoke the measurement selection algorithm to get $\mathbf{L}$. We then fly the drone to the location of the next measurement on $\mathbf{L}$. Next, our simulator calls AirSim to gather the virtual sensor (e.g., a photo-realistic RGB camera) data and then executes the classifiers. Upon getting the (single) measurement result, we check if the time runs out. If not, we invoke the measurement selection algorithm again for an updated $\mathbf{L}$. Note that, the measurement selection algorithms and classifiers can be hosted on the drone or at the ground control station depending on the capabilities of hardware devices. If not otherwise specified, we place these algorithms on the ground control station. Our simulator saves detailed log files, which are analyzed offline to quantify the performance of different measurements selection algorithms under the same scenarios.

We have implemented our proposed HEU_M and HEU_P algorithms. We have also implemented the benchmark OPT algorithm by enumerating all possible measurements. We are not aware of any prior work considering the fine-grained measurement selection algorithm. Hence, we also implemented the following three algorithms to mimic the current practices: (i) CHC_R (Central Histogram Classifier) performs a measurement with the RGB camera and histogram classifier at the center of the bounding box [190], (ii) CHC_U selects 4 locations at 1 m from the window for the ultrasound sensor [71], and (iii) RAN (Random) selects random measurements until the time $\hat{T}$ is up or the target accuracy $\hat{A}$ is met.

9.2 Setup

We run the same evaluation setup with our and other measurement selection algorithms to compare their performance. We set the drone speed $V = 3.5\ m/s$ and network bandwidth to be 24 Mbps. We set the drone rotor power consumption to be 3800 W [44], WiFi energy consumption to be 0.13 mJ/bit [162]. An RGB camera takes 0.36 W (and 1 s) to take an image, and an ultrasound sensor takes 0.01 W (and 0.49 s) to collect a distance value. We vary the following parameters. *Target accuracy* $\hat{A} \in \{0.7, 0.8, 0.9\}$, where 0.8 is the

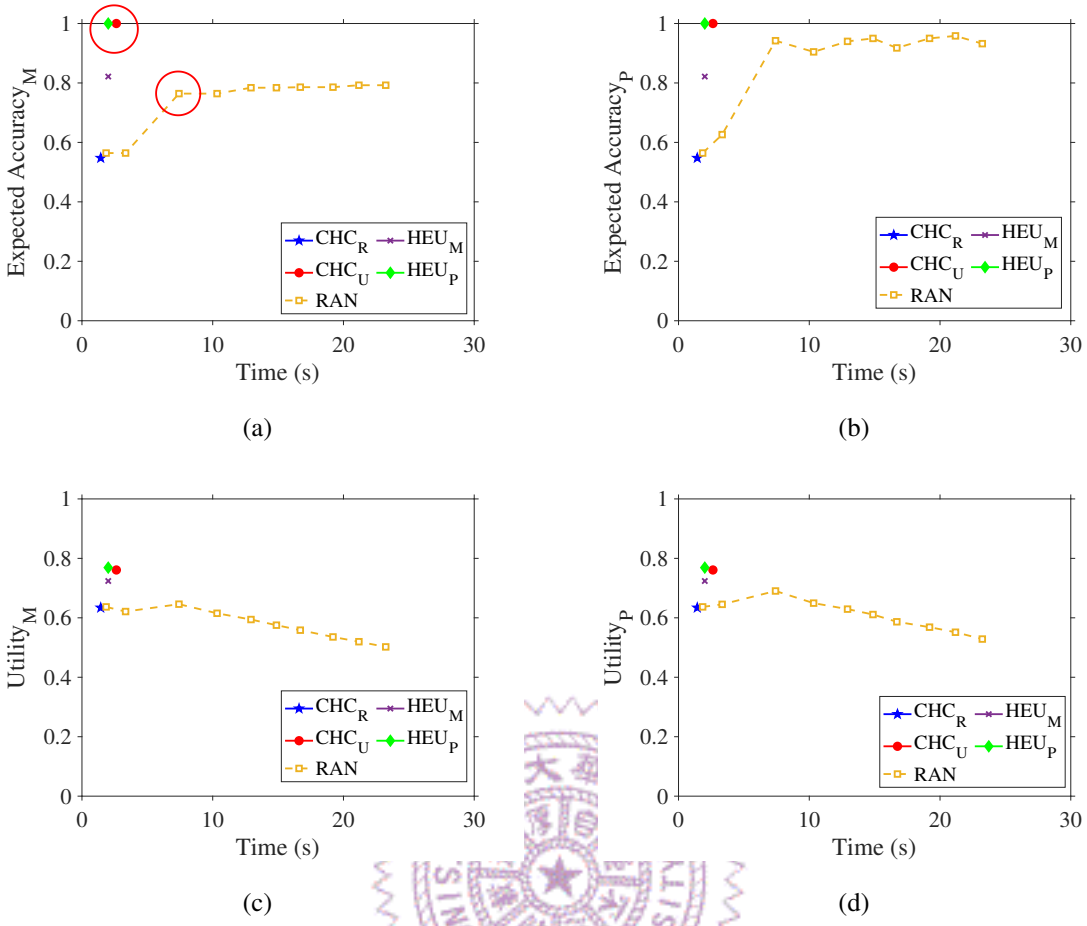


Figure 9.2: Results from a sample window: (a)–(b) expected accuracy and (c)–(d) utility function. We circle the ultrasound measurements in (a).

default parameter. *Time limit* $\hat{T} \in \{5, 10, 30, 60\}$ *seconds* for each window, where 30 is the default parameter. To avoid unnecessary measurements, we remove candidates with expected accuracy < 0.5. We define three *candidate sampling policies* to filter out some measurement candidates for better accuracy and lower overhead. The policies are: (i) E_8 (Equally-distance), which keeps $8 \times 8 \times 8$ centered locations, (ii) E_4, which skips every other location of E_8 along all three axes, and (iii) E_2, which does the same to E_4. We refer to no sampling as E_{10}, which is the default parameter. Note that candidate sampling is not applied on the ultrasound sensor, as it has much fewer candidates.

We measure the following performance metrics:

- *Overall accuracy* $O_M(\mathbf{L})$ and $O_P(\mathbf{L})$ represent the accuracy compared to the ground truth using the majority vote and probability-based fusion, respectively.
- *Expected accuracy* $A_M(\mathbf{L})$ and $A_P(\mathbf{L})$ represent the expected accuracy computed by Eqs. (8.1) and (8.2).
- *Utility function* $U_M(\cdot)$ and $U_P(\cdot)$ represent the utility function values computed by Eq. (8.4a) using the majority vote and probability-based fusing policies.

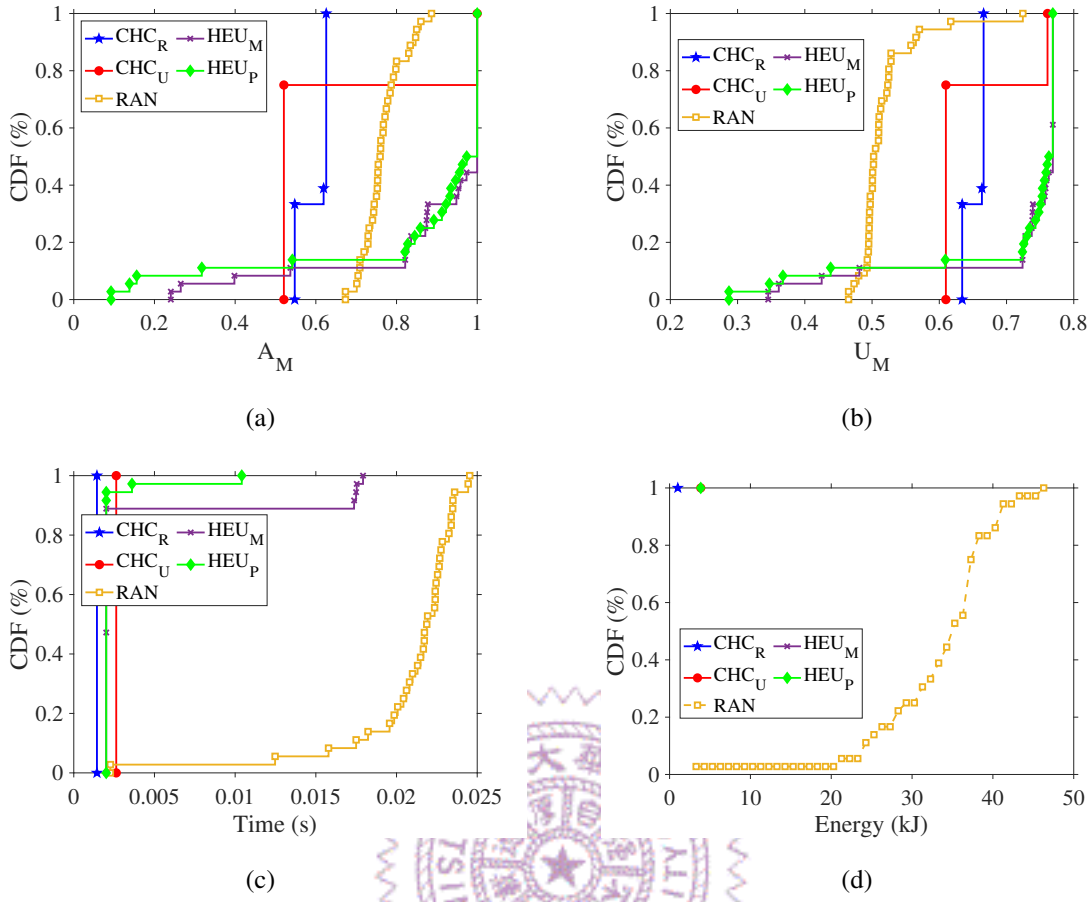


Figure 9.3: CDFs from 36 windows: (a) expected accuracy, (b) utility, (c) measurement time, and (d) energy consumption.

- *Total measurement time* $T(\mathbf{L})$ computed by Eq. (8.3). We also report the *running time* of the algorithm.
- *Number of measurements* is the length of $\mathbf{L}$.
- *Feasible ratio of measurements* F_M and F_P are the fraction of computed measurement selection that satisfy the accuracy $\hat{A}$ and time $\hat{T}$ constraints in Eq. (8.4b); the subscript indicates the fusion policy.
- *Energy* consumption of the drone when moving, sensing, and transferring data.

9.3 Results

Our proposed algorithms outperform the baselines. We first plot the accuracy and utility function values over time from a sample window in Fig. 9.2. We make several observations from this figure. First, all considered algorithms complete all measurements within $\hat{T}$. Second, our proposed HEU_M and HEU_P significantly outperform the baselines in terms of expected accuracy (Figs. 9.2(a) and 9.2(b)), which could be as high as 40%.

Third, our HEU_P finishes much earlier with a single measurement while achieving very high accuracy. The good performance of HEU_M and HEU_P could be attributed to the higher utility values in Figs. 9.2(c) and 9.2(d), compared to other algorithms. Note that the utility values drop along the time axis, because the penalty from longer measurement time. This is however not a concern as the achieved accuracy increases over time. Last, as similar observations are made with metrics defined with majority vote and probability-based fusion, we only report the majority vote variant in the following figures for brevity.

Table 9.1: Overall Results with Default Parameters

Algorithm	**CHC_R**	**CHC_U**	**RAN**	**HEU_M**	**HEU_P**
O_M (%)	50	75	72.22	88.89	88.89
O_P (%)	50	75	66.67	94	100
F_M (%)	0	25	19.44	88.89	86.11
F_P (%)	0	25	66.67	94.44	100
Mean (std.) L	1 (0)	1 (0)	9.36 (1.79)	1.22 (0.64)	1.67 (0.85)

Next, we plot the final results across all windows as Cumulative Distribution Functions (CDFs) in Fig. 9.3. Fig. 9.3(a) reveals that baselines result in over 60% of windows suffering from expected accuracy lower than 52% (CHC_U), 62% (CHC_R), and 78% (RAN); while our HEU_M and HEU_P result in 90+% expected accuracy on 60+% windows. Our good performance can be explained by the higher utility values reported in Fig. 9.3(b). Fig. 9.3(c) presents the total measurement time. Our HEU_M and HEU_P are both faster than RAN. *This means that our algorithms achieve higher accuracy within a shorter time.* Shorter measurement times also lead to lower energy consumption as reported in Fig. 9.3(d). Our algorithms consume one-seventh energy compared to RAN.

Table 9.1 reports the overall results compared to the ground truth. It is clear that our HEU_M and HEU_P outperform the baselines in terms of overall accuracy (by at least 13.89%) and feasible ratio (by at least 27.77%). Such a good performance is achieved without high overhead, on average only 1.67 and 1.22 measurements are needed for each window. *In summary, our proposed HEU_M and HEU_P achieve much higher overall and expected accuracy (up to 100% and 50% improvement), realize much higher feasible ratio (up to 100% improvement), and consume much less energy (only one-seventh), compared to the baselines.* Note that because O_M and O_P (and F_M and F_P) lead to similar observations, we report sample results in O_P (and F_P) in the rest of this thesis.

Implications of diverse parameters. We first plot the performance of different algorithms under diverse target accuracy in Fig. 9.4, where errorbars represent the 95% confidence levels across windows. Fig. 9.4(a) shows that our HEU_M and HEU_P deliver good overall accuracy, which meets the increasing target accuracy. This can be observed

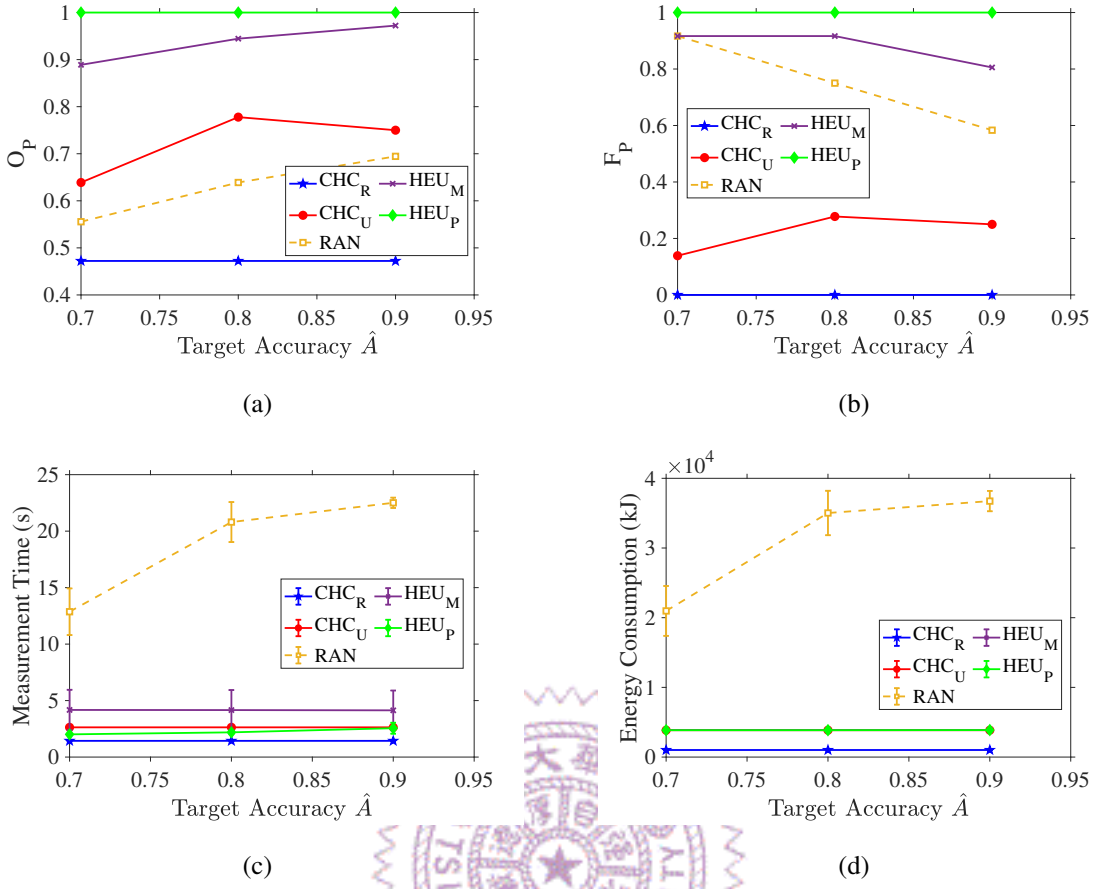
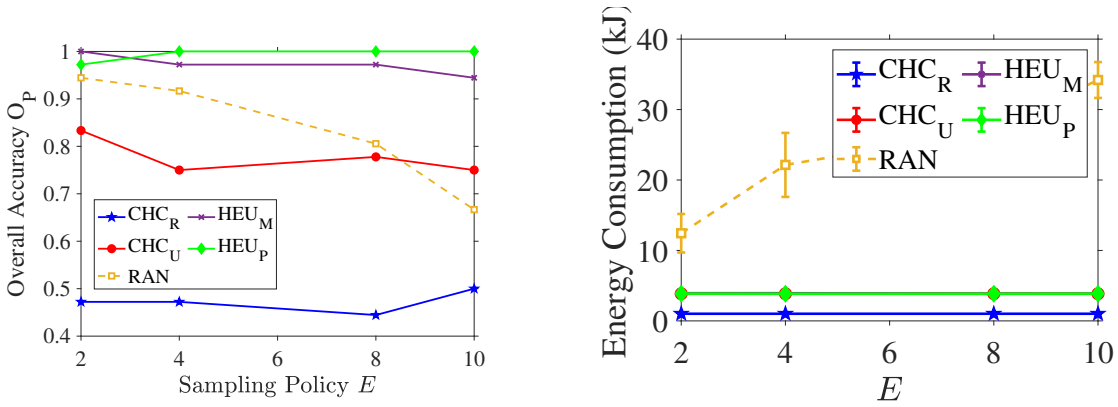


Figure 9.4: Implications of diverse target accuracy: (a) overall accuracy, (b) feasible ratio, (c) measurement time, and (d) energy consumption.

by the feasible ratios of 80% (HEU_M) and 100% (HEU_P) at $\hat{A} = 0.90$ in Fig. 9.4(b), which are at least 25% higher than that of RAN. Fig. 9.4(c) reveals that the measurement times of HEU_M and HEU_P do not increase even when $\hat{A} = 0.90$, showing that our algorithms scale well with the target accuracy. This can be observed by the constant energy consumption in Fig. 9.4(d). Our algorithms also perform well under different time limits between 5 and 60 s. The figures are omitted due to page limit. Fig. 9.5 gives the implications of different sampling policies, where smaller E values mean fewer measurement candidates and thus lower complexity. We report overall accuracy and energy consumption. We observe that our proposed HEU_M and HEU_P still achieve almost perfect overall accuracy (Fig. 9.5(a)) when E is reduced. In addition, their energy consumption levels remain almost constant when E is increased. Last, we compare our proposed algorithms against the benchmark OPT using a sample window and highly-sampled measurement candidates (with 7 candidates in total) under default settings. We cannot consider larger problem size because of the exponential complexity of OPT. Compared to the expected accuracy of OPT (0.68 and 0.83 in A_M and A_P), our HEU_M and HEU_P achieve (almost) optimal accuracy, i.e.,



(a) (b)

Figure 9.5: Implications of diverse sampling policies: (a) overall accuracy and (b) energy consumption.

0.69/0.83 and 0.63/0.74, respectively. *In summary, our proposed algorithms achieve good performance under diverse target accuracy and sampling policies.*

Chapter 10

Real Drone Implementations

We build a real high-rise building model, and control tello drones to be coordinated by our system. The detection results are transferred back to our laptop, which acts as the ground control station, and demonstrate to the users.

10.1 Environment Setup

City Model. We plan to use bookshelves pasted with window images as the high-rise building models.

Hardware. As we plan to run an indoor simulation, we choose Tello [34] as our firefighting drone. It is a quadcopter drone (shown in Fig. **??**), weights about 80 gram, and can fly about 13 minutes for one charge. As the drone is quite small, we decide not to install multi-modal sensors on it for safe flight.

We select XXX laptop as our ground control station.

We use Wifi to connect with Tello. Tello could transfer collected data from 100 m away at most.

Software. XXX talk about how we design and what we want to show on the dashboard, the image processing part in the virtual environmentXXX

10.2 System Design

XXX talk about the connection between above mentioned components, and the working processXXX

10.3 Evaluations.

Setup. XXX talk about how we set the number, like waypoints, time budget, accuracyXXX **Results.** XXX Show some demo results, and show statistics drone take to detect one waypoints, final accuracyXXX

Chapter 11

Conclusion

We studied the problem of developing autonomous firefighting drones to improve situational awareness in high-rise fires. Different from prior arts that focus on the coarse-grained waypoint scheduling problem, we solved the fine-grained measurement selection problem at each waypoint. Our problem is to adaptively select the best combinations of the sensor modality, classifier, and locations to achieve a target accuracy within a given time limit. We formulated the problem into an optimization problem and proposed two algorithms, HEU_M and HEU_P to solve it. We also implemented an event-driven simulator leveraging on the photo-realistic AirSim to evaluate our algorithms. Our extensive evaluation results revealed the merits of our proposed algorithms: they achieve high expected/overall accuracy, high feasible ratios, while consuming shorter measurement time and lower energy consumption, compared to baseline algorithms. Furthermore, our algorithms work well under diverse target accuracy, time limits, and sampling policies.

Our work can be extended in several dimensions:

- The various site survey strategies and more complex network conditions can be further compared for pros/cons.
- The performance of our classifiers in diverse environments, such as different window types and lighting conditions can be tested.
- More comprehensive measurement selection algorithms can be developed for performance guarantees at the cost of longer running time.
- The considered fine-grained measurement selection can be integrated with the coarse-grained waypoint schedule algorithms for a coherent, complete system of autonomous firefighting drones.

Bibliography

[1] 360° video viewing dataset in head-mounted virtual reality, 2017. http://nmsl.cs.nthu.edu.tw/dropbox/360dataset.zip.

[2] After mixed year, mobile ar to drive $108 billion vr/ar market by 2021, 2017. https://goo.gl/P9N0z0.

[3] Facebook, 2017. https://www.facebook.com/.

[4] Facebook Oculus Rift, 2017. https://www.oculus.com.

[5] HTC Vive, 2017. https://www.htcvive.com.

[6] Oculus Video, 2017. ://www.oculus.com/experiences/rift/926562347437041/.

[7] Samsung Gear VR, 2017. http://www.samsung.com/global/galaxy/gear-vr.

[8] YouTube, 2017. https://www.youtube.com/.

[9] H. Ahmadi, S. Tootaghaj, S. Mowlaei, M. Hashemi, and S. Shirmohammadi. Gset somi: a game-specific eye tracking dataset for somi. In *Proc. of ACM International Conference on Multimedia Systems (MMSys'16)*, pages 42:1–42:6, Klagenfurt, Austria, May 2016.

[10] H. Ali, C. Seifert, N. Jindal, L. Paletta, and G. Paar. Window detection in facades. In *Proc. of IEEE International Conference on Image Analysis and Processing (ICIAP'07)*, pages 837–842, Modena, Italy, September 2007.

[11] O. Alon, S. Rabinovich, C. Fyodorov, and J. R. Cauchard. *Drones in Firefighting: A User-Centered Design Perspective*. Association for Computing Machinery, New York, NY, USA, 2021.

[12] A. Alshbatat. Fire extinguishing system for high-rise buildings and rugged mountainous terrains utilizing quadrotor unmanned aerial vehicle. *MECS Press International Journal of Image, Graphics and Signal Processing*, 11(1):23, January 2018.

[13] A. I. N. Alshbatat. Fire extinguishing system for high-rise buildings and rugged mountainous terrains utilizing quadrotor unmanned aerial vehicle. *International Journal of Image, Graphics and Signal Processing*, 11(1):23, 2018.

[14] A. I. N. Alshbatat. Fire extinguishing system for high-rise buildings and rugged mountainous terrains utilizing quadrotor unmanned aerial vehicle. *International Journal of Image, Graphics and Signal Processing*, 11(1):23, 2018.

[15] A. Amin, F. Al-Obeidat, B. Shah, A. Adnan, J. Loo, and S. Anwar. Customer churn prediction in telecommunication industry using data certainty. *Elsevier Journal of Business Research*, 94:290–301, January 2019.

[16] H. Ando, Y. Ambe, A. Ishii, M. Konyo, K. Tadakuma, S. Maruyama, and S. Tadokoro. Aerial Hose Type Robot by Water Jet for Fire Fighting. *IEEE Robotics and Automation Letters*, 3(2):1128–1135, Apr. 2018.

[17] D. Anthony, S. Elbaum, A. Lorenz, and C. Detweiler. On crop height estimation with uavs. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4805–4812. IEEE, 2014.

[18] J. G. Apostolopoulos, P. A. Chou, B. Culbertson, T. Kalker, M. D. Trott, and S. Wee. The road to immersive communication. *Proceedings of the IEEE*, 100(4):974–990, 2012.

[19] J. Bai, S. Lian, Z. Liu, K. Wang, and D. Liu. Smart guiding glasses for visually impaired people in indoor environment. *IEEE Transactions on Consumer Electronics*, 63(3):258–266, August 2017.

[20] A. Baraldi and F. Panniggiani. An investigation of the textural characteristics associated with gray level cooccurrence matrix statistical parameters. *IEEE transactions on geoscience and remote sensing*, 33(2):293–304, March 1995.

[21] E. Bazan, P. Dokladal, and E. Dokladalova. Quantitative Analysis of Similarity Measures of Distributions. In *Proc. of British Machine Vision Conference (BMVC'19)*, Cardiff, United Kingdom, September 2019.

[22] R. Beard, T. McLain, M. Goodrich, and E. Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *IEEE transactions on robotics and automation*, 18(6):911–922, December 2002.

[23] K. Benson, G. Bouloukakis, C. Grant, V. Issarny, S. Mehrotra, I. Moscholios, and N. Venkatasubramanian. Firedex: A prioritized iot data exchange middleware for

emergency response. In *Proc. of ACM International Middleware Conference (Middleware'18)*, pages 279–292, Rennes, France, December 2018.

[24] K. Benson, G. Wang, N. Venkatasubramanian, and Y. Kim. Ride: A resilient iot data exchange middleware leveraging sdn and edge cloud resources. In *Proc. of IEEE/ACM International Conference on Internet-of-Things Design and Implementation (IoTDI'18)*, pages 72–83, Orlando, FL, April 2018.

[25] E. Bondi, D. Dey, A. Kapoor, J. Piavis, S. Shah, F. Fang, B. Dilkina, R. Hannaford, A. Iyer, L. Joppa, et al. Airsim-w: A simulation environment for wildlife conservation with uavs. In *Proc. of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, pages 1–12, 2018.

[26] A. Borji, M. Cheng, H. Jiang, and J. Li. Salient object detection: A survey. *arXiv preprint arXiv:1411.5878*, 2014.

[27] W. Budiharto, A. A. Gunawan, J. S. Suroso, A. Chowanda, A. Patrik, and G. Utama. Fast object detection for quadcopter drone using deep learning. In *2018 3rd International Conference on Computer and Communication Systems (ICCCS)*, pages 192–195. IEEE, 2018.

[28] Central Programme Office, National Fire Chiefs Council, UK. Hazard - fires in tall buildings, 2020.

[29] D. Ceylan, N. Mitra, Y. Zheng, and M. Pauly. Coupled structure-from-motion and 3D symmetry detection for urban facades. *ACM Transactions on Graphics (TOG'14)*, 33(1):1–15, January 2014.

[30] T. Chang, G. Bouloukakis, C. Hsieh, C. Hsu, and N. Venkatasubramanian. Smartparcels: Cross-layer iot planning for smart communities. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 195–207, 2021.

[31] O. Chapelle, P. Haffner, and V. N. Vapnik. Support vector machines for histogram-based image classification. *IEEE transactions on Neural Networks*, 10(5):1055–1064, 1999.

[32] C. Chen, R. Jafari, and N. Kehtarnavaz. A survey of depth and inertial sensor fusion for human action recognition. *Multimedia Tools and Applications*, 76(3):4405–4425, 2017.

[33] F. Chollet. Keras. `https://github.com/fchollet/keras`, 2015.

[34] C. Computers. Specifications for dji-tello, 2022. https://www.comx-computers.co.za/DJI-TELLO-specifications-183956.htm.

[35] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski. Viewport-adaptive navigable 360-degree video delivery. In *Proc. of IEEE International conference on communications (ICC'17)*, page Accepted to Appear, Paris, France, May 2017.

[36] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara. A deep multi-level network for saliency prediction. In *Proc. of International Conference on Pattern Recognition (ICPR'16)*, pages 3488–3493, Cancun, Mexico, December 2016.

[37] U. G. B. Council. Leadership in energy and environmental design, 2008. http://www.usgbc.org/leed.

[38] L. D'Acunto, J. Berg, E. Thomas, and O. Niamut. Using mpeg dash srd for zoomable and navigable video. In *Proc. of ACM International Conference on Multimedia Systems (MMSys'16)*, pages 34:1–34:4, Klagenfurt, Austria, May 2016.

[39] S. Daftry, C. Hoppe, and H. Bischof. Building with drones: Accurate 3D facade reconstruction using mavs. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA'15)*, pages 3487–3494, Seattle, WA, May 2015.

[40] H. Dang-Ngoc and H. Nguyen-Trung. Aerial Forest Fire Surveillance - Evaluation of Forest Fire Detection Model using Aerial Videos. In *2019 International Conference on Advanced Technologies for Communications (ATC)*, pages 142–148, Oct. 2019.

[41] B. V. Dasarathy. Sensor fusion potential exploitation-innovative architectures and illustrative applications. *Proceedings of the IEEE*, 85(1):24–38, 1997.

[42] C. DelBello. High-rise and mid-rise firefighting: Lobby control basics, 2020. https://www.firerescue1.com/high-rise/articles/high-rise-and-mid-rise-firefighting-lobby-control-basics-kDUyBVO8Z

[43] Department of Economic and Social Affairs, United Nations. 68says un, 2018.

[44] DJI. Dji agras mg-1p sprayer drone. https://talosdrones.com/product/dji-agras-mg1-p-sprayer-drone/.

[45] Elecfreaks. Ultrasonic Ranging Module HC-SR04. https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf,.

[46] W. Elmenreich. An introduction to sensor fusion. *Vienna University of Technology, Austria*, 502:1–28, 2002.

[47] M. Enzweiler and D. M. Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2179–2195, 2008.

[48] Epic Games. Unreal engine, 2019.

[49] R. Escombe, E. Ticona, V. Chavez-Perez, M. Espinoza, and D. Moore. Improving natural ventilation in hospital waiting and consulting rooms to reduce nosocomial tuberculosis transmission risk in a low resource setting. *Springer BMC infectious diseases*, 19(1):1–7, January 2019.

[50] M. et al. *Fire fighting tactics under wind driven conditions: laboratory experiments*. Fire Protection Research Foundation, 2009.

[51] C. Ezequiel, M. Cua, N. Libatique, G. Tangonan, R. Alampay, R. Labuguen, C. Favila, J. Honrado, V. Canos, C. Devaney, et al. Uav aerial imaging applications for post-disaster assessment, environmental management and infrastructure development. In *Proc. of IEEE International Conference on Unmanned Aircraft Systems (ICUAS'14)*, pages 274–283, Orlando, FL, May 2014.

[52] C. Fan, J. Lee, W. Lo, C. Huang, K. Chen, and C. Hsu. Fixation prediction for 360 video streaming to head-mounted displays. In *Underreviewed at ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'17)*, Taiper, Taiwan, June 2017.

[53] T.-Y. Fan, F. Liu, J.-W. Fang, N. Venkatasubramanian, and C.-H. Hsu. Enhancing situational awareness with adaptive firefighting drones: Leveraging diverse media types and classifiers. In *Proc. of the 13th ACM Multimedia Systems Conference*, Athlone, Ireland, 2022.

[54] T.-Y. Fan, T.-C. Tsai, C.-H. Hsu, F. Liu, and N. Venkatasubramanian. Winset: the first multi-modal window dataset for heterogeneous window states. In *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 192–195, 2021.

[55] J. Feuvre and C. Concolato. Tiled-based adaptive streaming using mpeg-dash. In *Proc. of ACM International Conference on Multimedia Systems (MMSys'16)*, pages 41:1–41:3, Klagenfurt, Austria, May 2016.

[56] M. Fishlock. A new set of tactical options to assist incident commanders and firefighters at high-rise incidents., 2013. http://www.highrisefire.co.uk/tactic.html.

[57] R. Gadde, R. Marlet, and N. Paragios. Learning grammars for architecture-specific facade parsing. *Springer International Journal of Computer Vision*, 117(3):290–316, March 2016.

[58] T. Giitsidis, E. G. Karakasis, A. Gasteratos, and G. C. Sirakoulis. Human and fire detection from high altitude uav images. In *2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 309–315. IEEE, 2015.

[59] F. Gong, C. Li, W. Gong, X. Li, X. Yuan, Y. Ma, and T. Song. A real-time fire detection method from video with multifeature fusion. In *Comput. Intell. Neurosci.*, 2019.

[60] R. Gonzalez and R. Woods. Filtering in the frequency domain. In *Digital Image Processing*, chapter 4, pages 270–271. Pearson, November 2018.

[61] P. C. Gray, A. B. Fleishman, D. J. Klein, M. W. McKown, V. S. Bézy, K. J. Lohmann, and D. W. Johnston. A convolutional neural network for detecting sea turtles in drone imagery. *Methods in Ecology and Evolution*, 10(3):345–355, 2019.

[62] Groupgets. LeptonModule, 2019. https://github.com/groupgets/LeptonModule.

[63] A. Guillen-Perez, R. Sanchez-Iborra, M.-D. Cano, J. C. Sanchez-Aarnoutse, and J. Garcia-Haro. Wifi networks on drones. In *2016 ITU Kaleidoscope: ICTs for a Sustainable World (ITU WT)*, pages 1–8. IEEE, 2016.

[64] Q. Ha, K. Watanabe, T. Karasawa, Y. Ushiku, and T. Harada. Mfnet: Towards real-time semantic segmentation for autonomous vehicles with multi-spectral scenes. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'17)*, pages 5108–5115, Vancouver, Canada, September 2017.

[65] J.-E. Haugeard, S. Philipp-Foliguet, F. Precioso, and J. Lebrun. Extraction of windows in facade using kernel on graph of contours. volume 5575, 06 2009.

[66] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1397–1409, June 2013.

[67] B. W. Hobson, D. Lowcay, H. B. Gunay, A. Ashouri, and G. R. Newsham. Opportunistic occupancy-count estimation using sensor fusion: A case study. *Building and environment*, 159:106154, 2019.

[68] J.-W. Hu, B.-Y. Zheng, and W. et al. A survey on multi-sensor fusion based obstacle detection for intelligent ground vehicles in off-road environments. *Frontiers of Information Technology & Electronic Engineering*, 21:675–692, 2020.

[69] C.-Y. Huang, C.-H. Hsu, Y.-C. Chang, and K.-T. Chen. GamingAnywhere: An open cloud gaming system. In *Proc. of ACM International Conference on Multimedia Systems (MMSys'13)*, pages 36–47, Oslo, Norway, February 2013.

[70] Y. Huang, W. C. Hoffmann, Y. Lan, W. Wu, and B. K. Fritz. Development of a spray system for an unmanned aerial vehicle platform. *Applied Engineering in Agriculture*, 25(6):803–809, 2009.

[71] Z. Huang, K. Wang, K. Yang, R. Cheng, and J. Bai. Glass detection and recognition based on the fusion of ultrasonic sensor and RGB-D sensor for the visually impaired. In *Proc. of SPIE Target and Background Signatures IV*, pages 118–125, Berlin, Germany, October 2018.

[72] Y. Imamura, S. Okamoto, and J. H. Lee. Human tracking by a multi-rotor drone using hog features and linear svm on images captured by a monocular camera. In *Proc. of the International MultiConference of Engineers and Computer Scientists (IMECS'16)*, volume 1, pages 8–13, Hong Kong, China, March 2016.

[73] Intel. Intel Realsense SDK, 2021. `https://github.com/IntelRealSense/librealsense`.

[74] M. Jarzabek, D. Lin, and H. Maas. Supervised detection of facade openings in 3D point clouds with thermal attributes. *MDPI Remote Sensing*, 12(3):543, February 2020.

[75] Jaunt: Cinematic virtual reality, 2017. `https://www.jauntvr.com/`.

[76] N. Jayapandian. Cloud enabled smart firefighting drone using internet of things. In *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 1079–1083. IEEE, 2019.

[77] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu. A survey of deep learning-based object detection. *IEEE Access*, 7:128837–128868, September 2019.

[78] I. Kang, R. Cimurs, J. H. Lee, and I. H. Suh. Fusion drive: End-to-end multi modal sensor fusion for guided low-cost autonomous vehicle. In *2020 17th International Conference on Ubiquitous Robots (UR)*, pages 421–428. IEEE, 2020.

[79] Y. Kavak, E. Erdem, and A. Erdem. A comparative study for feature integration strategies in dynamic saliency estimation. *Signal Processing: Image Communication*, 51:13–25, November 2017.

[80] Y. Kim, D.-W. Gu, and I. Postlethwaite. Real-time optimal mission scheduling and flight path selection. *IEEE Transactions on Automatic control*, 52(6):1119–1123, June 2007.

[81] Kitware and Velodyne. VeloView, 2014. `https://www.paraview.org/VeloView`.

[82] W. Krull, R. Tobera, and et al. Early forest fire detection and verification using optical smoke, gas and microwave sensors. *Procedia Engineering*, 45:584–594, 2012.

[83] N. Kumar, D. Acharya, and D. Lohani. An iot-based vehicle accident detection and classification system using sensor fusion. *IEEE Internet of Things Journal*, 8(2):869–880, 2020.

[84] K. K. Ladha. The condorcet jury theorem, free speech, and correlated votes. *American Journal of Political Science*, pages 617–634, 1992.

[85] S. Leary, M. Deittert, and J. Bookless. Constrained uav mission planning: A comparison of approaches. In *Proc. of IEEE international conference on computer vision workshops (ICCV'11 Workshops)*, pages 2002–2009, Barcelona, Spain, November 2011.

[86] S. C. Lee and R. Nevatia. Extraction and integration of window in a 3d building model from ground view images. volume 2, pages II–113, 01 2004.

[87] S. C. Lee and R. Nevatia. Extraction and integration of window in a 3d building model from ground view images. volume 2, pages II–113, 01 2004.

[88] F. S. Leira, T. A. Johansen, and T. I. Fossen. Automatic detection, classification and tracking of objects in the ocean surface from uavs using a thermal camera. In *2015 IEEE aerospace conference*, pages 1–10. IEEE, 2015.

[89] T. Lewicki and K. Liu. Aerial sensing system for wildfire detection: Demo abstract. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*,

SenSys '20, pages 595–596, New York, NY, USA, Nov. 2020. Association for Computing Machinery.

[90] D. Lin, Z. Dong, X. Zhang, and H. Maas. Unsupervised window extraction from photogrammetric point clouds with thermal attributes. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 4:10–14, June 2019.

[91] J. Lin, A. Morse, and B. Anderson. The multi-agent rendezvous problem. In *Proc. of IEEE international conference on decision and control*, pages 1508–1513, Maui, HI, December 2003.

[92] F. Lippoldt. Window detection in facades for aerial texture files of 3D CityGML models. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR'19)*, pages 11–19, Long Beach, CA, June 2019.

[93] F. Liu, T.-Y. Fan, C. Grant, C.-H. Hsu, and N. Venkatasubramanian. DragonFly: Drone-Assisted High-Rise Monitoring for Fire Safety. In *Proc. of IEEE International Symposium on Reliable Distributed Systems (SRDS'21)*, Virtual, September 2021.

[94] H. Liu, Y. Xu, J. Zhang, J. Zhu, Y. Li, and S. C. H. Hoi. DeepFacade: A deep learning approach to facade parsing with symmetric loss. *IEEE Transactions on Multimedia*, 22(12):3153–3165, December 2020.

[95] H. Liu, H. Zheng, F. Li, and H. Cai. A hybrid model for predicting window opening state in buildings based on non-intrusive monitoring. *SAGE Indoor and Built Environment*, page 1420326X20940362, July 2020.

[96] Y. Liu, Y. Yin, and S. Zhang. Hand gesture recognition based on hu moments in interaction of virtual reality. In *IEEE 2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics*, volume 1, pages 145–148, 2012.

[97] M. A. Lopez Medina, M. Espinilla, C. Paggeti, and J. Medina Quero. Activity recognition for iot devices using fuzzy spatio-temporal features as environmental sensor fusion. *Sensors*, 19(16):3512, 2019.

[98] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI'7)*, pages 674–679, Vancouver, BC, Canada, August 1981.

[99] J. M. MacDonald, P. Korb, and R. A. Hoppe. Farm size and the organization of us crop farming. Technical report, 2013.

[100] D. Madrzykowski, S. Kerber, S. Kumar, and P. Panindre. Wind, fire and high-rises: firefighters and engineers conduct research to combat a lethal threat. *Mechanical Engineering Magazine*, 132(7):22–27, 2010.

[101] S. Malihi, M. Valadan Zoej, M. Hahn, and M. Mokhtarzade. Window detection from UAS-derived photogrammetric point cloud employing density-based filtering and perceptual organization. *MDPI Remote Sensing*, 10(8):1320, August 2018.

[102] A. Mavlankar and B. Girod. Pre-fetching based on video analysis for interactive region-of-interest streaming of soccer sequences. In *Proc. of IEEE International Conference on Image Processing (ICIP'09)*, pages 3061–3064, Cairo, Egypt, November 2009.

[103] A. Mavlankar and B. Girod. Video streaming with interactive pan/tilt/zoom. In *Signals and Communication Technology*, pages 431–455. 2010.

[104] D. M. McGrail. *Firefighting operations in high-rise and standpipe-equipped buildings*. PennWell Books, 2007.

[105] T. McLain and R. Beard. Coordination variables, coordination functions, and cooperative timing missions. *AIAA Journal of Guidance, Control, and Dynamics*, 28(1):150–161, November 2005.

[106] S. Mehrotra, A. Kobsa, N. Venkatasubramanian, and S. Rajagopalan. Tippers: A privacy cognizant iot environment. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pages 1–6, 2016.

[107] H. Mei, X. Yang, Y. Wang, Y. Liu, S. He, Q. Zhang, X. Wei, and R. Lau. Don't hit me! glass detection in real-world scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'20)*, pages 3687–3696, June 2020.

[108] J. J. A. Mendes Jr, M. E. M. Vieira, M. B. Pires, and S. L. Stevan Jr. Sensor fusion and smart sensor in sports and biomedical applications. *Sensors*, 16(10):1569, 2016.

[109] G. Menzies and J. Wherrett. Windows in the workplace: examining issues of environmental sustainability and occupant comfort in the selection of multi-glazed windows. *Elsevier Energy and buildings*, 37(6):623–630, June 2005.

[110] L. Merino, F. Caballero, J. Martinez-de Dios, and A. Ollero. Cooperative fire detection using unmanned aerial vehicles. In *Proceedings of the 2005 IEEE international conference on robotics and automation*, pages 1884–1889. IEEE, 2005.

[111] L. Merino, F. Caballero, J. R. Martínez-de Dios, J. Ferruz, and A. Ollero. A cooperative perception system for multiple uavs: Application to automatic detection of forest fires. *Journal of Field Robotics*, 23(3-4):165–184, 2006.

[112] U. R. Mogili and B. Deepak. Review on application of drone systems in precision agriculture. *Procedia computer science*, 133:502–509, 2018.

[113] M. Moore Bick. Grenfell tower inquiry: Phase 1 report overview - report of the public inquiry into the fire at grenfell tower on 14 june 2017, 2019. `https://www.grenfelltowerinquiry.org.uk/phase-1-report`.

[114] V. Moustaka, A. Vakali, and L. Anthopoulos. A systematic review for smart city data analytics. *ACM Computing Surveys (CSUR)*, 51(5):1–41, December 2018.

[115] K. Muhammad, J. Ahmad, and S. W. Baik. Early fire detection using convolutional neural networks during surveillance for effective disaster management. *Elsevier Neurocomputing*, 288:30 – 42, 2018.

[116] K. Muhammad, J. Ahmad, I. Mehmood, S. Rho, and S. W. Baik. Convolutional neural networks based fire detection in surveillance videos. *IEEE Access*, 6:18174–18183, 2018.

[117] S. Mustapha, A. Kassir, K. Hassoun, Z. Dawy, and H. Abi-Rached. Estimation of crowd flow and load on pedestrian bridges using machine learning with sensor fusion. *Automation in Construction*, 112:103092, 2020.

[118] National Institute for Occupational Safety and Health (NIOSH),. Three Fire Fighters Die in a 10-Story High-Rise Apartment Building - New York, August 1999. `https://tinyurl.com/y63p84j7`,.

[119] A. S. Natu and S. Kulkarni. Adoption and utilization of drones for advanced precision farming: A review. *International journal on recent and innovation trends in computing and communication*, 4(5):563–565, 2016.

[120] M. Neuhausen, C. Koch, and M. Konig. Image-based window detection: an overview. 2016.

[121] M. Neuhausen, A. Martin, M. Obel, P. Mark, and M. Konig. A cascaded classifier approach to window detection in facade images. In *Proc. of IAARC International*

Symposium on Automation and Robotics in Construction (ISARC'17), pages 690–697, Taipei, Taiwan, July 2017.

[122] M. Neuhausen, A. Martin, M. Obel, P. Mark, and M. König. A cascaded classifier approach to window detection in facade images. 06 2017.

[123] H. P. D. Nguyen and D. D. Nguyen. Drone application in smart cities: The general overview of security vulnerabilities and countermeasures for data communication. *Development and Future of Internet of Drones (IoD): Insights, Trends and Road Ahead*, pages 185–210, 2021.

[124] W. S. Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

[125] S. Ogawa, S. Kudo, M. Koide, H. Torikai, and Y. Iwatani. Development and control of an aerial extinguisher with an inert gas capsule. In *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, pages 1320–1325, Dec. 2014.

[126] One Source Heating and Cooling. How does opening your windows impact your HVAC system?, 2016. https://www.onesourceair.com/blog/opening-windows-impact-hvac-system.

[127] OpenTrack: head tracking software, 2017. `https://github.com/opentrack/opentrack`.

[128] N. Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, January 1979.

[129] M. Pal. Random forest classifier for remote sensing classification. *International journal of remote sensing*, 26(1):217–222, 2005.

[130] L. Parcalabescu, N. Trost, and A. Frank. What is multimodality? *arXiv preprint arXiv:2103.06304*, 2021.

[131] J. Primicerio, S. F. Di Gennaro, E. Fiorillo, L. Genesio, E. Lugato, A. Matese, and F. P. Vaccari. A flexible unmanned aerial vehicle for precision agriculture. *Precision Agriculture*, 13(4):517–523, 2012.

[132] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan. Optimizing 360 video delivery over cellular networks. In *Proc. of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, pages 1–6. ACM, 2016.

[133] QuadArt. Modular house, 2017. https://www.unrealengine.com/marketplace/en-US/product/modular-houses.

[134] C. Reardon and J. Fink. Air-ground robot team surveillance of complex 3D environments. In *Proc. of IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR'16)*, pages 320–327, Lausanne, Switzerland, October 2016.

[135] C. Reardon and J. Fink. Air-ground robot team surveillance of complex 3d environments. In *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 320–327, 2016.

[136] M. Recky and F. Leberl. Windows detection using k-means in cie-lab color space. In *2010 20th International Conference on Pattern Recognition*, pages 356–359, Aug 2010.

[137] Renderpeople. Scanned 3d people pack, 2019. https://www.unrealengine.com/marketplace/en-US/product/9c3fab270dfe468a9a920da0c10fa2ad.

[138] A. Restas et al. Drone applications for supporting disaster management. *World Journal of Engineering and Technology*, 3(03):316, 2015.

[139] A. Restas et al. Drone applications for supporting disaster management. *Scientific Research Publishing World Journal of Engineering and Technology*, 3(03):316, 2015.

[140] M. Riegler, M. Larson, C. Spampinato, P. Halvorsen, M. Lux, J. Markussen, K. Pogorelov, C. Griwodz, and H. Stensland. Right inflight?: A dataset for exploring the automatic prediction of movies suitable for a watching situation. In *Proc. of ACM International Conference on Multimedia Systems (MMSys'16)*, pages 45:1–45:6, Klagenfurt, Austria, May 2016.

[141] H. Riemenschneider, U. Krispel, W. Thaller, M. Donoser, S. Havemann, D. Fellner, and H. Bischof. Irregular lattices for complex shape grammar facade parsing. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1640–1647. IEEE, 2012.

[142] J. J. Roldan-Gomez, E. Gonzalez-Gironda, and A. Barrientos. A survey on robotic technologies for forest firefighting: Applying drone swarms to improve firefighters' efficiency and safety. *MDPI Applied Sciences*, 11(1):363, 2021.

[143] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[144] S. Safavi, T. Iqbal, W. Wang, P. Coleman, and M. Plumbley. Open-window: A sound event data set for window state detection and recognition. In *Proc. of International Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE'20)*, Tokyo, Japan, August 2020.

[145] O. Safety, H. Administration, et al. Fire service features of buildings and fire protection systems. *Occupational Safety and Health Administration US Department of Labor*, 2015.

[146] A. K. Saha, J. Saha, R. Ray, S. Sircar, S. Dutta, S. P. Chattopadhyay, and H. N. Saha. Iot-based drone for improvement of crop quality in agricultural field. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 612–615. IEEE, 2018.

[147] M. Saqib, S. D. Khan, N. Sharma, P. Scully-Power, P. Butcher, A. Colefax, and M. Blumenstein. Real-time drone surveillance and population estimation of marine animals from aerial imagery. In *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6. IEEE, 2018.

[148] C. Schweier and M. Markus. Classification of collapsed buildings for fast damage and loss assessment. *Bulletin of earthquake engineering*, 4(2):177–192, 2006.

[149] V. Sea, A. Sugiyama, and T. Sugawara. Frequency-based multi-agent patrolling model and its area partitioning solution method for balanced workload. In *Proc. of Springer International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 530–545, June 2018.

[150] C. Seifert. Tsg-60:tourist sights graz 60.

[151] C. Seifert. Tsg-20: Tourist sights graz 20, 2004.

[152] U. Shafi, R. Mumtaz, N. Iqbal, S. M. H. Zaidi, S. A. R. Zaidi, I. Hussain, and Z. Mahmood. A multi-modal approach for crop health mapping using low altitude remote sensing, internet of things (iot) and machine learning. *IEEE Access*, 8:112708–112724, 2020.

[153] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, pages 621–635. Springer, November 2018.

[154] E. Shin, R. Yus, S. Mehrotra, and N. Venkatasubramanian. Exploring fairness in participatory thermal comfort control in smart buildings. In *Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environments*, pages 1–10, 2017.

[155] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[156] B. Sirmacek, L. Hoegner, and U. Stilla. Detection of windows and doors from thermal images by grouping geometrical features. In *Proc. of IEEE Joint Urban Remote Sensing Event (JURSE'11)*, pages 133–136, Munich, Germany, April 2011.

[157] P. Skorput, S. Mandzuka, and H. Vojvodic. The use of unmanned aerial vehicles for forest fire monitoring. In *Proc. of IEEE International Symposium ELMAR*, pages 93–96, Zadar, Croatia, September 2016.

[158] S. Smith and D. Rus. Multi-robot monitoring in dynamic environments with guaranteed currency of observations. In *Proc. of IEEE conference on decision and control (CDC'10)*, pages 514–521, Atlanta, GA, December 2010.

[159] B. Srikudkao, T. Khundate, C. So-In, P. Horkaew, C. Phaudphut, and K. Rujirakul. Flood warning and management schemes with drone emulator using ultrasonic and image processing. In *Recent Advances in Information and Communication Technology 2015*, pages 107–116. Springer, 2015.

[160] K. Su, J. Li, and H. Fu. Smart city and the applications. In *2011 international conference on electronics, communications and control (ICECC)*, pages 1028–1031. IEEE, 2011.

[161] R. Suganya and S. Rajaram. Feature extraction and classification of ultrasound liver images using haralick texture-primitive features: Application of svm classifier. In *IEEE 2013 international conference on recent trends in information technology (ICRTIT)*, pages 596–602, 2013.

[162] L. Sun, R. K. Sheshadri, W. Zheng, and D. Koutsonikolas. Modeling wifi active power/energy consumption in smartphones. In *2014 IEEE 34th international conference on distributed computing systems*, pages 41–51, 2014.

[163] S.-M. Tang, C.-H. Hsu, Z. Tian, and X. Su. An aerodynamic, computer vision, and network simulator for networked drone applications. In *Proc. of the 27th ACM Annual International Conference on Mobile Computing and Networking (MobiCom '21)*, page 831–833, 2021.

[164] Y. Tang, C. Hou, S. Luo, J. Lin, Z. Yang, and W. Huang. Effects of operation height and tree shape on droplet deposition in citrus trees using an unmanned aerial vehicle. *Computers and Electronics in Agriculture*, 148:1–7, 2018.

[165] O. Teboul. Ecole centrale paris facades database, 2008.

[166] O. Teboul, I. Kokkinos, L. Simon, P. Koutsourakis, and N. Paragios. Parsing facades with shape grammars and reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1744–1756, 2012.

[167] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios. Segmentation of building facades using procedural shape priors. pages 3105–3112, 11 2010.

[168] The OpenCV Library, 2000. `http://opencv.org`.

[169] C. Thiel. *Multiple classifier fusion incorporating certainty factors*. PhD thesis, Verlag nicht ermittelbar, 2004.

[170] B. Thomas. A survey of visual, mixed, and augmented reality gaming. *ACM Computers in Entertainment (CIE)*, 10(1):3:1–3:33, 2012.

[171] R. Tylecek and R. Sara. Spatial pattern templates for recognition of objects with regular structure. In *Proc. of Springer Pattern Recognition*, pages 364–374, Berlin, Heidelberg, 2013.

[172] K. Valavanis and G. Vachtsevanos. In *Handbook of unmanned aerial vehicles*. Springer, 2015.

[173] D. Ventura, M. Bruno, G. J. Lasinio, A. Belluscio, and G. Ardizzone. A low-cost drone based application for identifying and mapping of coastal fish nursery grounds. *Estuarine, Coastal and Shelf Science*, 171:85–98, 2016.

[174] B. Vergouw, H. Nagel, G. Bondt, and B. Custers. Drone technology: Types, payloads, applications, frequency spectrum issues and future developments. In *The future of drone use*, pages 21–45. Springer, 2016.

[175] T. Vigier, J. Rousseau, M. Silva, and P. Callet. A new hd and uhd video eye tracking dataset. In *Proc. of ACM International Conference on Multimedia Systems (MMSys'16)*, pages 48:1–48:6, Klagenfurt, Austria, May 2016.

[176] K. Wada. labelme: Image Polygonal Annotation with Python, 2016. `https://github.com/wkentaro/labelme`.

[177] A. Wallar, E. Plaku, and D. Sofge. Reactive motion planning for unmanned aerial surveillance of risk-sensitive areas. *IEEE Transactions on Automation Science and Engineering*, 12(3):969–980, July 2015.

[178] J. Wang and H. Wang. Tunable fiber laser based photoacoustic gas sensor for early fire detection. *Infrared Physics & Technology*, 65:1–4, 2014.

[179] R. Wang, J. Bach, and F. Ferrie. Window detection from mobile lidar data. In *Proc. of IEEE Workshop on Applications of Computer Vision (WACV'11)*, pages 58–65, Kona, Hawaii, January 2011.

[180] Z. Wang, Y. Wu, and Q. Niu. Multi-sensor fusion in automated driving: A survey. *Ieee Access*, 8:2847–2868, 2019.

[181] T. Wen, G. Chen, Y. Zhang, Y. Xiao, B. Wang, and B. Hu. Research on fire detection method of high-rise residential buildings based on cloud edge fusion computing. In *2021 IEEE International Conference on Emergency Science and Information Technology (ICESIT)*, pages 414–418, 2021.

[182] wohaiyo. Streetscenewindowdetectiondataset, 2019, 2019.

[183] P. Woznowski, X. Fafoutis, T. Song, S. Hannuna, M. Camplani, L. Tao, A. Paiement, E. Mellios, M. Haghighi, N. Zhu, et al. A multi-modal sensor infrastructure for healthcare in a residential environment. In *2015 IEEE International Conference on Communication Workshop (ICCW)*, pages 271–277. IEEE, 2015.

[184] M. Yu, H. Lakshman, and B. Girod. A framework to evaluate omnidirectional video coding schemes. In *Proc. of IEEE International Symposium on Mixed and Augmented Reality (ISMAR'15)*, pages 31–36, Fukuoka, Japan, September 2015.

[185] C. Yuan, Z. Liu, and Y. Zhang. Uav-based forest fire detection and tracking using image processing techniques. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 639–643. IEEE, 2015.

[186] C. Yuan, Z. Liu, and Y. Zhang. Fire detection using infrared images for uav-based forest fire surveillance. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 567–572. IEEE, 2017.

[187] C. Yuan, Y. Zhang, and Z. Liu. A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques. *NRC research Press Canadian journal of forest research*, 45(7):783–792, 2015.

[188] R. Yus, G. Bouloukakis, S. Mehrotra, and N. Venkatasubramanian. Abstracting interactions with iot devices towards a semantic vision of smart spaces. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 91–100, 2019.

[189] C. Zhao, W. S. Lee, and D. He. Immature green citrus detection based on colour feature and sum of absolute transformed difference (satd) using colour images in the citrus grove. *Elsevier Computers and Electronics in Agriculture*, 124:243–253, 2016.

[190] H. Zheng, F. Li, H. Cai, and K. Zhang. Non-intrusive measurement method for the window opening behavior. *Elsevier Energy and Buildings*, 197:171–176, August 2019.

[191] A. Ziebinski, R. Cupek, H. Erdogan, and S. Waechter. A survey of adas technologies for the future perspective of sensor fusion. In *International Conference on Computational Collective Intelligence*, pages 135–146. Springer, 2016.

[192] S. Zolanvari, S. Ruano, A. Rana, A. Cummins, R. da Silva, M. Rahbar, and A. Smolic. Dublincity: Annotated lidar point cloud and its applications. *arXiv preprint arXiv:1909.03613*, 2019.