Internet of Artificially Intelligent Things: A Case Study on Security and privacy through Face Detection and Recognition with drones.

By

Kosmas Sourlis

to

The School of Computing

for the BSc in Computer Science

Thesis Supervisor
Players Andreas

Module Professor
Tasos Stylianou

University of Derby
Mediterranean College
2022

Acknowledgments

I would first like to thank my professors Dr. Andreas Plageras and Mr. Stylianou Tasos, for the guidance that they have provided me throughout this amazing journey. It has been extremely interesting and profitable. Finally, I would like to thank my family and the university for all the support they provided.

Abstract | In recent years, drones have seen rapid growth in numerous fields such as agriculture, law enforcement, medical care military, etc. This increasing number of drones has raised privacy and security concerns that need to be managed. Computer vision is an emerging field of artificial intelligence and combined with very high resolutions cameras of drones it can assist in numerous cases in law enforcement. This study is focused primarily on existing drone applications. It concludes with a proposition idea related to drones in law enforcement and more specifically identifying criminals in real-time with help of artificial intelligence facial recognition algorithms.

KEYWORDS | Drones, Safety, Artificial Intelligence, OpenCV, Computer vision

Table of Contents

1. Introduction **7**

1.1 Current Study Overview *7*

1.2 Scientific Contribution *7*

1.3 Outline *8*

2. Related Work **8**

2.1 Security and privacy in drones *8*

2.2 Drones in agriculture *9*

2.3 Drones in healthcare *10*

2.4 Drones in law-enforcement *11*

3. Computer Vision **12**

3.1 Computer Vision and Image Processing *13*

3.2 Useful Computer Vision Packages *14*

3.3 OpenCV *14*

4. Proposed Idea **20**

4.1 Why haven't we seen existing many applications for law enforcement implemented yet? *20*

4.2 Idea *21*

4.3 Technologies *22*

4.4 Application architecture *22*

4.5 Database schema *23*

4.6 Database relations *25*

4.7 Database requests *26*

4.8 Input system *26*

4.9 Graphical user interface *29*

4.10 Data preprocessing *30*

4.11 Training the model *31*

4.12 Recognizing the face in real-time *32*

5. Experimental Results (Testing/Evaluation) **33**

6. Conclusion **34**

7. Future Work and Directions **35**

Figure Number	Page
(1)	15
(2)	16
(3)	17
(4)	18
(5)	19
(6)	20
(7)	23
(8)	27
(9)	28
(10)	29
(11)	29
(12)	29
(13)	30
(14)	31
(15)	31
(16)	32
(17)	33

1. Introduction

In recent years we have seen numerous innovative applications of drones in various fields. The military, law enforcement, agriculture, and health care are some key areas where drones are active. Drones have very useful sensors and cameras that help them accomplish various complicated tasks. they can be used for surveillance and transportation. For example, an interesting and useful area of activity in health care is blood transportation, also in agriculture crop management is assisted by numerous drone applications. Computer vision is a fascinating field of artificial intelligence. Its subject is the extraction of useful information by processing videos and images. It applies very sophisticatedly artificial intelligence algorithms to solve very complicated problems. Face recognition and object recognition are two areas where computer vision is related. Drones can capture very useful images and record high-resolution videos with a bird's eye view. The latter combined with computer vision can assist in the solution of very complex problems like detecting and identifying all kinds of objects from a large distance. This paper proposes an implementation related to face recognition and drone programming. It is related to the field of law enforcement. The main goal of the implementation is to identify the faces of potential criminals in real-time during drone patrolling. I strongly believe that this is an extremely useful application that can potentially assist many law enforcement units across the globe as it aims to increase the efficiency of police patrolling and criminal detection overall. Since this implementation is quite complex it has many challenges that need to be overcome. One of the main challenges is making the face recognition algorithm as light as possible. If the algorithm is heavy, then it will be almost impossible to run it in real-time. Another challenge is to make the application as secure as possible while maintaining the speed of the application. Recent studies have shown that drones are prone to cyber-attacks, therefore because drones often carry sensitive information, security is very important. The main objective of this study is to provide an innovative implementation of drones in law enforcement, that will hopefully be extended to match the standards of each law enforcement unit.

1.1 Current Study Overview

This study is mostly focused on drones and computer vision and how these two can be combined to produce very impressive results. The field in which drones have a presence is very wide, from the military to law enforcement so far, we have seen many interesting applications that have strongly assisted in the solution of complex problems. Computer vision and UAVs have unlimited potential and I strongly believe that the best innovations with these two are yet to come.

1.2 Scientific Contribution

Face recognition is not a new area of study it is around for quite a long time. Nevertheless, real-time face recognition is something that has not been implemented widely in society yet. Especially law enforcement units don't use extensively face recognition in most countries. The implementation presented in this paper aims to provide a flexible and easy-to-modify software that will identify criminals in real-time. It utilizes the incredible technology of computer vision and the extremely useful bird-eye view that drones provide.

1.3 Outline

My study is divided into two sections theoretical and practical. My development methodology for the practical part will be waterfall. In the theoretical part the areas that I will focus my research are:

- Existing drone applications.
- Computer vision
- OpenCV
- Drone security threats.

The practical part will be divided into 5 sections

- Requirement analysis
- Design
- Development
- Testing
- Deployment

In the requirements analysis phase, I will decide the technologies that need to be used for my implementation. Furthermore, in the design phase, I will analyze the application architecture and provide a schema that will be followed throughout the development process. The development process will be split into different phases that will be decided in the design phase. Moreover, I plan to spend a month testing the application and about a week deploying it.

The application will aim to enhance security on IOD (internet of drones) and produce an innovative way to reinforce social security.

2. Related Work

2.1 Security and privacy in drones

The internet of things is a very sophisticated network of devices that are capable to connect and provide very useful services (Lin et al., 2018). These devices, can either gather useful information via sensors, interact with the world through various actuators or do both. Throughout the years drones evolved significantly and became an essential part of the fascinating internet of things and have provided a wide range of services through various applications. It is no secret that drones will control the low aerial space in years to come. With the increasing number of drones in the low aerial space traffic management has become challenging, also due to very sensitive data transfer via drones' security concerns related to traffic conjunction have been raised that need to be accessed (Labib, Brust, Danoy and Bouvry, 2021; Lee, Gyu La and Kim, 2018) A great study for managing drone traffic on various missions can be seen at Veerappan and Chennattu, 2022.

UAV applications for civil and military purposes have increased significantly. This is because drones are very cost-efficient to maintain and can provide very useful information with a bird-eye-view and deliver packages in hard-to-reach locations. Some fields in which drones

are used are courier services, search and rescue operations, security surveillance, etc. (Shafique, Mehmood and Elhadef, 2021). An interesting study shows how drones can be used for accident prevention, by detecting cracks of concrete in buildings that are prone to clattering down (Ali Alheeti et al., 2022). Artificial intelligence and more specifically machine learning have a very solid contribution to the development of more complex and utilitarian applications. Therefore, this broad range of applications has caused security threats that need to be managed. Drones handle very sensitive data in the form of audio-video or images through communication channels such as WIFI which is not the most secure protocol for data transfer. The biggest security concerns related to data leaks are spoofing, false data injection, jamming, etc (Shafique, Mehmood and Elhadef, 2021). So far, we have seen some very interesting research on Dos cyber-attack detection techniques more can be reviewed at (Ouiazzane, Addou and Barramou, 2021).

Apart from security concerns related to data leaks and traffic conjunction, there are more security concerns that are physical security related. Drones are not always designed with physical safety in mind and that can lead to unpleasant events such as physical accidents for example (Drones falling and injuring civilians). Additionally, a big issue is the possibility of violation of personal space, drones can reach places and record video or take photos of people without their consent. Based on data that was collected by the Canadian Public safety such incidents have caused a lot of trouble and had led to blackmailing and other unpleasant events. Therefore, privacy concerns are also a thing to worry about (Yaacoub, Noura, Salman and Chehab, 2020; Lee, Gyu La and Kim, 2018).

The issues mentioned above are something that must be taken seriously by everyone related to the field of drones, measures should always be taken to prevent unpleasant situations from happening. The most common ways to counterattack all these threats and concerns are data encryption mainly during data transform, multi-factor authentication protocols, anti-malware software, and strict legislations regarding flight protocols for UAVs from governments (Labib, Brust, Danoy and Bouvry, 2021; Lin et al., 2018; Shafique, Mehmood and Elhadef, 2021; Yaacoub, Noura, Salman and Chehab, 2020; Michailidis and Vouyioukas, 2022; Oruc, 2022; Tanveer et al, 2022).

2.2 Drones in agriculture

Agriculture is a sector in which UAVs have been used extensively. One area that which drones are utilized in agriculture is in monitoring and assessing crops with remote sensing. Furthermore, drones are also used for the precision distribution of agricultural chemicals and biological control agents. Additionally, they can be used to monitor the health of animals and retrieve samples remotely.

Remote sensing via the camera is an old invention and its first application dates way back to the mid-1800s. In World War 2 soldiers captures infrared photos from hot aired balloons and airplanes and utilized them to detect camouflaged military equipment and facilities. Nowadays, infrared photos that are captured from drones are used to assess the agricultural development process and ensure its integrity. Therefore, by measuring the reflectance between visible and near-infrared light, vegetation health and growth potential can be assessed and that can be very useful to farmers.

In agriculture, the conditions under which the data are collected are essential. Light reflects differently depending on the type of surface. Therefore, several different types of drones are

used with different kinds of sensors for different purposes. The two main types of sensors that are used in agriculture drones are passive and active sensors. Active sensors can emit energy and detect the reflection of that emitted energy, on the other hand, passive sensors can only measure the emitted or reflected energy from a scene. Passive sensors are more cost-efficient than active sensors, but they do not work well when ambient conditions have a strong presence in the scene. Meanwhile, active sensors are heavier and less cost-efficient, but they are capable of producing reliable data on variated ambient conditions.

Some examples of existing applications are:

- **Seedling emergence assessment.** There is a need for constant observation of crops during the first stages of crop development. This is because if something goes wrong in the early stages of the development process due to environmental factors, farmers can replant and potentially save a damaged set of crops. The latter is achieved through emergent mapping and measurement of seedlings. Drones can collect very high-resolution images and provide information on whether germination is unsuccessful in a specific area.

- **Crop damage assessment.** During the crop development process, it is very likely to have damage. Damages can be caused by unwanted weather conditions, insects, etc. Drones can calculate the area and the size of the damage and provide very useful information.

- **Water management.** One of the biggest challenges in agriculture is water management. One common way that farmers use to manage water supply is soil moisture sensors. This is not the most efficient way to manage this issue because results can be extracted from these sensors only when severe damage to crops has already taken place. Drones can have a very important role in gathering useful information that can be used to manage faster and more efficient water distribution.

- **Livestock applications**. Recently agriculture has seen increased usage of livestock drone applications. This is because drones are very cos cost-efficient at the same time can provide high-quality video. Livestock monitoring can provide services such: as observing animals' behaviors or infrastructure that is responsible to keep animals in one place like fences and gates. Also, thermal sensors in drones are used to detect if animals have a fever and therefore aid in preventing diseases via early diagnosis (Merwe et al., 2020).

2.3 Drones in healthcare

Another sector that which drones are used is health care. Below ways that drones are in health care are analyzed.

- **Medical transportation.** In many cases across the globe medical transportation is a problem due to geographical issues. The latter can be overcome because drones can move at an incredibly high speed of 40-60 miles per hour and cover a great distance in a very short time. Therefore, drones have been used for all sorts of medical transportations Mexico has many regions that are tough to reach, recently a company named Aidronix in Mexico began the development of an innovative application that its goal is to distribute medical supplies

such as medications, vaccines antibiotics, etc. to rural regions of Mexico (Wulfovich, Rivas and Matabuena, 2018). Moreover, blood is very important and often lifesaving, but many times there is a shortage, especially in African countries. The most common way to deliver blood in most African countries is via an ambulance or car. Because Africa does not have a good road network blood deliverance is often delayed. In 2016, San Francisco Bay-based Zipline began a drone delivery operation in Rwanda, that increased the speed and deliverance of blood. More specifically, between 2016 and 2019 over 4000 missions have been completed, that delivered 7000 units of blood, and 1/3 of these missions were lifesaving situations (Ling, G. and Draghic, N.,2019).

- **Emerging cardiac care.** In the USA, cardiac arrests that happen outside the hospital are one of the most common mortality reasons. This type of drone application is emerging and currently under development. Drones can likely be automatically deployed and transfer AED (automated external defibrillators) with the help of GIS systems. The latter can save valuable time, aid significantly in a potentially crucial situation, and save a human life. Various studies showed that AED transformation was able to reach faster its destination than traditional ways (Zègre-Hemsey et al., 2018).

2.4 Drones in law-enforcement

In many different cases, drones have played a significant role in providing not only a basic form of security but also been involved in many, different law enforcement scenarios. In detail, and by their first appearance in 1936 by Lt. Cmdr. Fahrney, who was developing an Aircraft Project, managed to introduce the term of the drone in his report. Soon drones have been introduced as military devices that were described as Unmanned Aerial Vehicles, specifically in 1946 in the United States. Most importantly, the name “drone” was taken from the male bee word from the Old English term and the systems that divide them were named Unmanned Aerial System (UAS), Small Unmanned Aerial System (sUAS), Micro Aerial Vehicle (MAV), Unmanned Aerial Vehicle (UAV) and finally the Remotely Operated Aircraft (ROA), who all manage to make the technological world take over the public and private service of our community (Jones, 2019). During the last years, drones, as told above, have played a very important role in military operations and especially in the Iraq and Afghanistan wars. Moreover, drones have played a significant role in public maintenance such as surveying roads, equipment, and pipelines, specifically in the Prudhoe Bay area of Alaska. As we may also have seen, drones have also managed to be included within the transportation sector and even the delivery of products, with many large-scaled companies being a part of that new norm. With drones to be a huge part of the present and especially the upcoming future, farmers have also maintained to keep these changes for themselves and worked with drones to count and check their cattle and monitor their fence conditions throughout their lives and during their work (Jones, 2019). Although drones are an important asset in the everyday lives of people during their work, they are also taking over the marketplace and sometimes even replacing smartphones as gifts. Last year, some people wished to have a smartphone on their holidays, but instead these days there are even children that wish to have a drone just for themselves, with the latter showing how important they have been lately. In terms of law enforcement again, currently, 910 public safety agencies own drones and the law enforcement purchases have managed to be increased by two-thirds of the last.

However, despite how important and helpful drones can be and will be soon, even more, governments had to decide on what should be better for the safety of citizens and police officers to be included in the basic work and functionalities. Specifically, the military, in terms of war, was testing the required tools to make the trade. These tools of the trade were including both the tactics and the equipment of the drones themselves. Specifically, history has shown that in different circumstances the military and law enforcement, in general, has made a transition from military application to tactical equipment (Jones, 2019).

As history also suggests and shows, the use of drones in the military and in every military mission that is available, will be also available for law enforcement. This means that their efficiency, effectiveness, and economical value are now tested for them to be used in more dangerous or important cases that may require saving human lives. In addition, the use of drones in the private and public sectors has increased largely over the last decade and this shows us how important they are also in terms of aircraft. Specifically, in 2012, the FAA (Federal Aviation Administration) Modernizations and Reform Act was using drones to safely guide other aircraft into their domestic airspace and thus guide them to their desired goal (Jones, 2019). In this way, they were tasked to develop their regulations and make them relevant for the use of UAS in public spaces. Regarding the FAA, they require the free use of drones in the law enforcement guidelines that also persist of the Fourth amendment considerations that use force and arming the drones themselves and thus making them lethal. This law has been enforced by many states and the legislation was about the use of drones themselves only by law enforcement. However, many states also provide inconsistent information for the legislation, which also makes it difficult to find a middle line between the requirements for the drone policies and the actual purpose of the legislation for use in law enforcement.

Many research methods have indicated that due to legislation and regulatory analysis, which eventually leads to uncertainty, many law enforcement agencies have not yet developed their acceptable policies for the right use of drones, and maybe not even drones themselves within their premises. This shows that they were all designed to showcase the procedures for the armed drones and also display a proactive approach to the armed drone policy development for law enforcement (Jones, 2019). As more and more legislations have come to the surface and made themselves visible, since the application of the Fourth Amendment, in most cases law enforcement has to obtain a warrant to search a property or an individual and assume their respected working scenarios except in specific cases, where these cases will later be examined by the court. However, most of the states have managed to enact legislation that governed the use of drones for law enforcement, and all of that happened by the end of 2017. Although everything was functioning well, these laws have been extremely restrictive for drones and caused certain confusion that only can be addressed when an incident occurs, and the actual results are decided in a court of law. Specifically, any unlawful action of drones that assumes that surveillance of an individual on their owned property or residence, could cause a violation of their Fourth Amendment and lead to the right to privacy concerns on the new technology (Jones, 2019; Whelan, J., Almehmadi and El-Khatib, 2022).

3. Computer Vision

Computer Vision is simply explained to be a form of modeling and replicating vision that applies to human abilities and is also used in computer software and hardware. Specifically, it is a

discipline that includes a variety of ways to rebuild, understand and interrupt a 3D scene from the 2D images that represent the scene and adjust its properties and the structure of the scene (Pulli, Baksheev, Kornyakov and Eruhimov, 2012). There is a variety of hierarchical values for computer vision and mostly these values are divided into three categories that can be based upon certain things and circumstances. In detail, low-level vision includes the basic processing of images with the sole purpose of extracting the features for the final results. Furthermore, the intermediate-level vision can include some basic object recognition and the final interpretation of a 3D scene. Finally, for the whole category's purpose in the computer vision hierarchy, the high-level vision, as a final step, includes a basic and conceptual description representing an activity, behavior, and intention of a scene.

3.1 Computer Vision and Image Processing

Specific processing studies about images are considered to be the main aspect of image processing. Specifically, the input and output of the image processing situation are the actual images that are about to be processed. Computer vision is explicitly constructing meaningful descriptions of certain objects from their image that are physical and the final output of the computer vision process is the actual description or the interpretation of the structures in a 3D scene (Tutorialspoint, 2022).

In terms of the final applications and how computer vision finds them inside specific fields, first the robotics field specializes in specific sectors. In detail, with robotics, Computer Vision manages to accomplish localization and determine whether a robot is located automatically and the navigation of the technology. Furthermore, it can avoid specific obstacles and also in most situations assembly, by peg-in-hole, welding, or painting. Also, it achieves manipulation, for example with PUMA robot manipulator and finally the Human-Robot Interaction (HRI) which achieves more intelligent robots that are also able to interact and serve people.

For medicine, the classification and the final detection of Computer Vision can be achieved with a lesion or cell classification and in generic cases even tumor detection. Moreover, 2D and 3D segmentation with also 3D reconstruction of human organs with the help of famous features and capabilities, such as MRI or ultrasound. Finally, for medicine, technology has also evolved in the surgery part of the section, which also includes vision-guided robotics surgery (Tutorialspoint, 2022).

Security has also played a significant role within the technology sector of Computer Vision, which also requires biometrics, such as fingerprint scanners, iris detectors, and face recognition systems, and finally surveillance-detecting activities and behaviors.

For transportation purposes, autonomous vehicles and safety issues often dissolved with driver vigilance monitoring for safer distances and driving security (Tutorialspoint, 2022).

For the final category, industrial automation applications can lead to industrial inspections with defect detection techniques and assembly just like the robotics category. Also, the barcode and package label reading all include the industrial automation application. Finally, object sorting and OCR for document understanding techniques can also include the final results for this category (Tutorialspoint, 2022).

3.2 Useful Computer Vision Packages

Regarding some specific packages for the Computer Vision technique, certain features also help provide the best behavior for the packages that can be installed. In specific, for computer vision functioning with Python, there is a very popular library that is called OpenCV (Open Source Computer Vision). This is a library that makes use of specific programming functions that fully aim to have real-time computer vision. The library is written in C++ and it is the primary interface for the language overall. For this specific scenario, the most available and used feature is the code method below:

```
pip install opencv_python-X.X-cp36-cp36m-winX.whl
```

The value X represents the actual Python version of the machine that it is installed into and of course the win32 or 64-bit system that it is running on the specific computer. For anaconda environments specific, the following code can be used to install the OpenCV environment:

```
conda install -c conda-forge OpenCV
```

For the specific reading, writing, and displaying of each image separately, most CV applications can easily get images and input and of course produce the actual desired images as output. For instance, the specific function can be used in terms of how crucial it is to follow functions for the sole purpose of making the images function properly and are divided into three different sections. Firstly, the read() function, can read the image and it is used solely for this specific purpose. OpenCV imread() can read and support different kinds of images and their formats, such as PNG, JPEG, JPG, TIFF, and many more. Secondly, the imshow() function, can function with the sole purpose of showing the image to the window, and the window then itself will automatically fit the image itself to the desired image size, finally allowing the program to support various image types, such as PNG, JPG, JPEG, TIFF and many more, just like the first step. Finally, for the third step, the inwrite() function, manages to write the image, which allows the OpenCV imwrite() function to support various image types, just like the three other categories mentioned before, which include, PNG, JPG, JPEG, TIFF and many others (Tutorialspoint, 2022).

3.3 OpenCV

As previously mentioned for the OpenCV functions and their specificity when it comes to imread, imwrite, and imshow functions, there are specific examples that can be described and analyzed for the final purpose of the essay. Specifically, the following example can show in Python code, how an image is being read in one format and shown in a window, and finally written in another format with the same type of the image. At first, we can import the general OpenCV package by importing the cv2 package itself inside the code:

```
import cv2
```

After doing the basic installation of the package, we can now read a particular and specific image with the help of the above-mentioned imread() function which is clearly shown below in a self-declared image variable:

```
image = cv2.imread('image_flower.jpg')
```

With the sole purpose of finally showing the image to be displayed, we use the imshow() function with the name of the window to show the image with the name **image_flower** and finally provide us with the result from the below-taken code:

```
cv2.imshow('image_flower',image)
cv2.destroyAllwindows()
```

Figure 1: The final result of a taken image (Tutorialspoint, 2022)

With the below-taken code we can clearly observe that the same image can be altered to another type by providing another name for it with the imwrite() function:

```
cv2.imwrite('image_flower.png',image)
```

The actual output of the above code will soon provide us with the true value, as this represents an actual PNG icon, and the final wish of deleting the actually created window with the flower is achieved with the end method of destroyallWindows().

In other words, the color space conversion for the OpenCV language includes a variety of other different technologies and paradigms that are often included within such categories. For the OpenCV color space conversion, the images are not stored within using the conventional RGB color. However, they are stored within the reverse order which often is the BGR order. This includes and sums how the default color code for reading each image is BGR and also the cvtColor() function for the conversion is converting the image from the first color to another Tutorialspoint, 2022.

For example, if we'd like to consider converting an image from BGR to grayscale, we would use a certain and specific method, by first importing the library and then applying the imread() function to final show it with imshow() just as the code below shows.

```
import cv2
```

```
image = cv2.imread('image_flower.jpg')
```

```
cv2.imshow('BGR_Penguins',image)
```

Figure 2: Simple image of penguins (TutorialPoint, 2022)

To provide the best method to show the above image in grayscale we just need to confirm the previous steps and also use imshow() again to apply the new image as grayscale, just like it is shown below.

```
image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow('gray_penguins',image)
```

Figure 3: The same image in grey color (Tutorialspoint, 2022)

OpenCV can also be displayed in edge detection which achieves further details and advice can be given also for the final result. Edge detection is allowing humans to see through a rough sketch and can easily recognize some object types and their poses which they have. The latter is achieved because the edge can play a significant role for humans and in the applications of computer vision overall. For image and edge detection OpenCV provides a mostly simple and useful function which is called Canny() and the sole purpose of it is to detect the edges of the image. For example, after importing the main Computer Vision library and the NumPy as np for the main further library methods, we can read the following wishing image to be edited, and then the read function should be used properly as a declared variable. Additionally, with the Canny() function, we can detect the edges of the image we would like to edit and after doing that, with the help of the imwrite() function we can detect the edges. Finally, and as shown below, we use imshow() to show the edges of the image, and imshow() provides us with the final result.

```
import cv2
import NumPy as np
```

```
image = cv2.imread('Penguins.jpg')
```

```
cv2.imwrite('edges_Penguins.jpg',cv2.Canny(image,200,300))
```

```
cv2.imshow('edges', cv2.imread("edges_Penguins.jpg'))
```

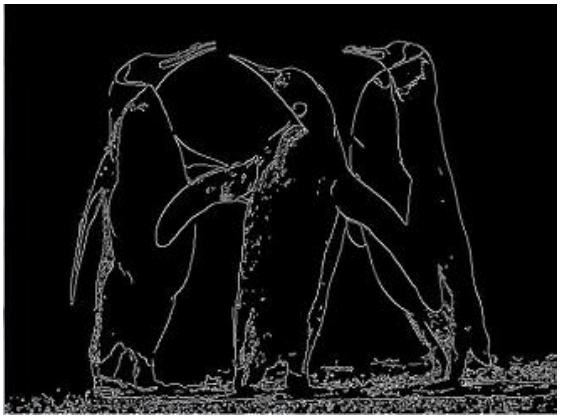


Figure 4: The same image edge detection (Tutorialspoint, 2022)

Except for the main and basic methods of image detection shown above, two other techniques require face and eyes detection. Firstly, face detection, represents a fascinating application of computer vision that also makes a more realistic and futuristic display. With the OpenCV built-in functionality, the library can perform the face detection method and with the Haar classifier for face detection, it can finally be done (Tutorialspoint, 2022).

At first, with the Haar cascade classifier data, we can find the actual dataset in the OpenCV package and import them directly into the code. For the actual code and the final face detection system to be working properly, we can define a variable with a base name of "face_detection" and provide the code with the necessary path to read the XML data directly. Then, with the imread() function the image is read and finally grayscaled with the color functions. Finally, the image takes detection testing for faces and draws a rectangle to provide the user with the detected final face from the image.

```
import cv2
import NumPy as np
```

```
face_detection=
cv2.CascadeClassifier('D:/ProgramData/cascadeclassifier/
haarcascade_frontalface_default.xml')
```

```
img = cv2.imread('AB.jpg')
```

```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

```
faces = face_detection.detectMultiScale(gray, 1.3, 5)
```

```
for (x,y,w,h) in faces:
   img = cv2.rectangle(img,(x,y),(x+w, y+h),(255,0,0),3)
cv2.imwrite('Face_AB.jpg',img)
```

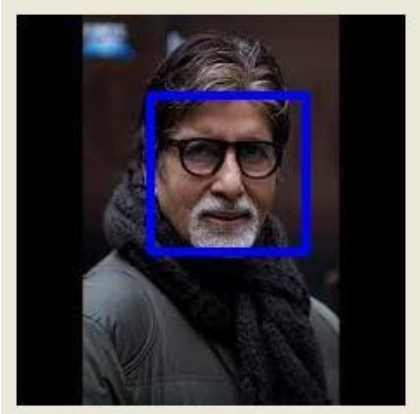


Figure 5: Face detection with OpenCV CascadeClassifier function (TutorialPoint, 2022)

Finally, except for the Face Detection application on computers with the help of OpenCV, the system provides us with another well-structured and organized application for eye detection. Eye detection is another application for Computer Vision, which can make the appliance of images more realistic and futuristic in the end. OpenCV has a functionality that can detect eyes from an image and the Haar cascade is used also in this part (Tutorialspoint, 2022). Specifically, to define and finally detect the eyes of a person within an image, we should be able to make the first configurations within the code and import the basic libraries for cv2 and NumPy as np for the code, just like the code snippet below.

```
import cv2
import NumPy as np
```

After that, using the Haar cascade classifier for the detection of the face we declare a basic variable just like the other previous methods and import the actual classifier within our code to finally be able to read the image with the well-known imread() function.

```
eye_cascade = cv2.CascadeClassifier('D:/ProgramData/cascadeclassifier/haarcascade_eye.xml')
```

```
img = cv2.imread('AB_Eye.jpg')
```

As the first configurations for the final detection of the face directly from the image have been taken, we should now be able to grayscale the image itself and then detect the multiscale for finally performing the actual face detection. In this way, we can import the detectMultiScale method and grayscale the image itself as shown below.

```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

```
eyes = eye_cascade.detectMultiScale(gray, 1.03, 5)
```

With the drawing of the grayscale method and the final cuts for the result on face detection, we can now provide the algorithm with the loop that draws the rectangle for each eye and can also detect the face directly from the eyes of the individual shown within the image (Tutorialspoint, 2022).

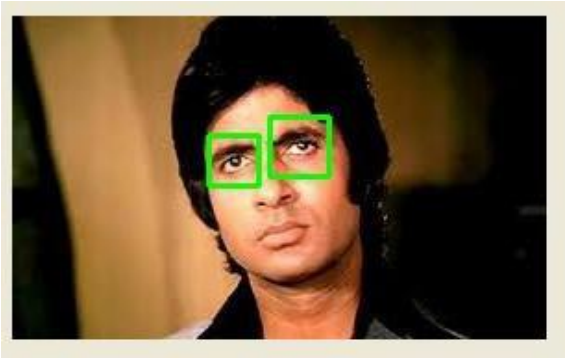


Figure 6: Eye detection with OpenCV CascadeClassifier function (TutorialPoint, 2022)

4. Proposed Idea

Drone applications have a vast range of applications in numerous fields some of which are mentioned above. The field that intrigued and inspired me to propose and implement an application in the field of law enforcement. So far, we have seen many applications of drones related to law enforcement, but many countries have not implemented them yet. The latter is mainly because of strict legislation and level of complexity I will analyze and present these reasons in detail below. My proposed idea and implementation are divided into key areas which I will analyze and present separately.

4.1 Why haven't we seen existing many applications for law enforcement implemented yet?

Although many existing implementations related to the field of law enforcement have already been developed and tested, they have not been integrated into the system and that is because of two main reasons.

The first one is because of very strict legislation regarding UAVs. A UAV has very identical legislation to a regular aircraft. The latter means that there must be an infrastructure that has to be followed for every application and that included a ground control unit, well-trained personnel, and a very robust infrastructure of networking. These requirements are not easy to have in every law enforcement unit because of cost inefficiency and sometimes complicated bureaucracy. Also, the fact that police drones are dealing with civilian data raises privacy concerns that are hard to deal with and perplexes bureaucracy issues even more. Finally, because these drones will fly in the low aerial space of civilian regions safety concerns are becoming an extra issue (for example a drone falls off the sky and injures a civilian).

The second reason is technical challenges. More specifically, drones especially inexpensive drones have a short flight duration and that is because most of them use lithium batteries that do not meet the requirements of the desired power consumption. Considering the latter and given the fact that police patrols often last many hours, a time issue arises. Furthermore, police drones have mostly been used for surveillance. A surveillance operation requires a pilot to

operate the drone and an officer that is occupied with the surveillance part of the given task. It makes sense to try to automate some of the tasks to reduce the number of personnel required for surveillance. However, if for example, a pilot is replaced with an automated flight system security concerns will arise because a drone will fly differently in different weather conditions and that can lead to unwanted accidents.

4.2 Idea

Many times, law enforcement units patrol specific regions to detect criminals that already exist in the database of police departments. The latter often requires long hours of patrolling and investigation that can be time-consuming, cost-inefficient, and sometimes produce no results. So, it makes sense to try to automate and enhance the procedure of criminal detection by any means. In my opinion, drones can strongly contribute to this area through various methods. Some very high-quality drones have integrated ultra-high-resolution cameras that can record or live stream video directly to a server. The latter can become extremely beneficial and strongly contribute to criminal detection if it is integrated with artificial intelligence.

Artificial Intelligence is a field of computer science that has seen rapid development in recent years and has strongly contributed to the solution of difficult and multiplex problems in the real world. In recent years one area of artificial intelligence that has seen a massive improvement is computer vision. Computer vision is the field of computer science whose purpose is to obtain data consisting of images or videos and apply machine learning or deep learning algorithms to extract useful information that will aid various fields. For example, many smartphones today use computer vision to unlock the screen of the phone. The latter is achieved after the image processing and recognition of the owner's face are completed with the front camera. Computer vision has been used for many other reasons such as object detection object tracking etc. One very interesting research on object detection can be reviewed at Zhu, Z. and Cheng, Y., 2020.

The general idea is to use these drones with super useful cameras and automatically detect the faces of criminals that have already been inserted into the databases of law enforcement units. The application aims to function in real-time, so it has to be as light as possible. This is not a simple task, it requires a very robust software architecture to reduce the processing power that is necessary to achieve these results, I will analyze this architecture in detail in the next chapter. Also, the drones must be directly operated from the ground control unit so there is a need for an input system to control the drone which I will also present in the following chapters. Furthermore, to identify the faces of specific criminals there is a need to utilize an artificial intelligence algorithm that is essential to perform face recognition. To achieve the latter in the next chapters, I will explain the data preprocessing techniques as well as the training methods for the model. Additionally, given the fact that the datasets of criminals are huge, separate datasets will be created for every zone to reduce the processing power and provide efficient results. Finally, a database schema will be proposed that implements a strong infrastructure that aims to improve the data retrieval speed.

4.3 Technologies

Many modern technologies were used for the implementation. Below I will analyze the technologies as well as programming libraries that were included during the development process.

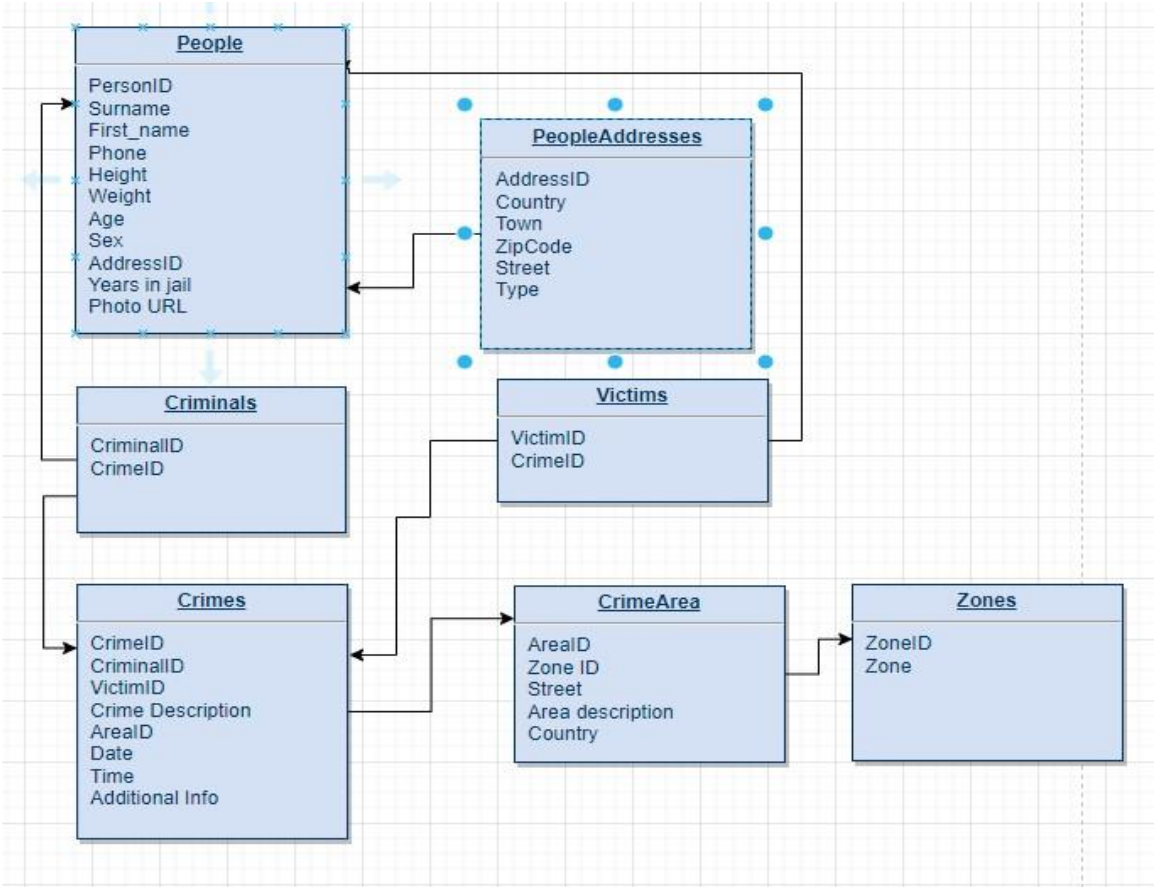
First of all, python is the programming language that the application was written. The latter is because python had already many useful libraries related to computer vision and drone programming. The drone programming library that is used for this application is called tello and it was published by Gabriel Heinzer (https://github.com/gheinzer/python-tello). This library is specifically designed to program the tello drone which is manufactured by Ryze robotics. Furthermore, for the artificial intelligence part of the application, I used OpenCV. Finally, for the database, I used XAAMP, and MySQL combined.

4.4 Application architecture

The application consists of four essential parts the database schema, the artificial intelligence algorithm for face detection-recognition, the training method, and the input system. The user will be able to take off, land, and pilot the drone directly from the keyboard the face recognition part is handled automatically from the application. The database is designed to be fast and easily modifiable to meet the standards of different law enforcement units. The face recognition algorithm can run smoothly on real time. Therefore the architecture aims to be simple for the pilot, the only the thing the pilot has to do is to navigate the drone wherever he wants and everything else will be handled by the application automatically.

4.5 Database schema

The database that is used for this application is MySQL. The database schema has in total 7 tables which are: people, people addresses, criminals, victims, zones, crimes, crime area,



and zones. Below I will analyze the table fields and the relationships between the fields

Figure 7: Database schema

.

To understand better database schema below is a brief description of each table and each field:

Crimes: Crime is a table that records a crime that has been committed and stores the necessary information.

Crimes fields:

- CrimeID: This field is the primary key of the table and uniquely identifies each record.
- CriminalID: This field is a foreign key to the Criminals table, and it uniquely identifies each criminal.
- VictimID: This field Is a foreign key to the victims table, and it uniquely identifies each victim.
- CrimeDescription: This field describes the crime and is not null.
- AreaID: This field is a foreign key to table CrimeArea it can take null values.
- Date: This field stores the date that the crime was committed.

- Time: This field stores the time that the crime was committed.
- AdditionalInfo: This is an optional field, and it stores any additional information related to the crime.

CrimeArea: CrimeArea is the table that stores all the information needed for the location of the crime.

CrimeArea fields:

- AreaID: This is the primary key of the table, and it is also a foreign key to the table Crimes.
- ZoneID: This field is a foreign key to table Zones.
- Street: This field stores the street where the crime was committed.
- AreaDescriprion: This field describes the area where the crime was committed.
- Country: is the field that stores the country where the crime was committed (All the data in this dataset have the same value “Greece”).

Zones:

This table stores the zone in which the crime was committed. Each zone refers to specific regions within a country.

Zones fields:

- ZoneId: This field is the primary key of the table.
- Zone: This field stores the regions in which each zone is stored. An example value is: “Epirus, Thrace, Makedonia, Thessaly.”

Criminals:

This is the table that stores the information needed to identify a criminal.

Criminals fields:

- CriminalId: This field is the primary key of the table and it is also a foreign key in table People.
- CrimeID: This field is a foreign key to table Crimes and it stores the which were committed by this criminal.

Victims:

This is the table that stores the information needed to identify a victims

Victims fields:

- VictimsId: This is the primary key of the table and it is also a foreign key in table People.
- CrimeID: This field is a foreign key to table Crimes and it stores the which were committed by this criminal.

People:

This table stores all the necessary information about both criminals and victims.

People fields:

PersonID: This field store is the primary key of the table and it is also a foreign key in tables criminals and victims.

- Surname: This field stores the person's surname.
- First_name: This field stores the person's first name.
- Phone: This field stores the person's phone, and it is not a mandatory field.
- Height: This field stores the person's height, and it is not a mandatory field.
- Weight: This field stores the person's height, and it is not a mandatory field.
- Age: This field stores the person's age it is, not a mandatory field.
- Sex: This field stores the person's gender it, is not a mandatory field.
- AddressId: This field is a foreign key to the table people address.
- Years in jail: This field refers only to records where the person is a criminal if the person is a victim most likely will be filled with "-". It stores the total years if any a criminal has spent in prison.
- Photo URL: The photo URL is the field that holds the value of the image path that is stored in the server. The photo that is stored in the URL is showing a clear photo of the criminal or victim. This photo is used in the graphical user interface when a criminal is recognized.

People addresses:

This table stores the criminal's and victim's address details.

People addresses fields:

- AddressId: This is the primary key for the table
- Country: This field stores the person's country of origin.
- ZipCode: This field stores the person's zip code of residence.
- Street: This field stores the person's street of residence.
- Type: This field stores the type of person's residence. (For example. A person can have both work and a permanent address).

4.6 Database relations

To make the database more effective relationships have been created between tables, below is a brief description of the database relations.

People table: This table has three relations. The two relations are with the same field personId and the type is many to many with the crimes criminal id field and victims victimId field accordingly. The third is with the people address table and it is a one-to-many relationships.

Crimes: This table has also three relationships. The first relationship is with table criminals on-field crimes.crimeID -> criminals.crimeID. The second relationship is with table victims and it is on field crimes.crimeID -> victims.crimeID. The third relationship is with the table crimes area

and it is with field crimes.AreaID -> crimesArea.AreaId. All three relationships are of type many to many.

CrimeArea: This table has one relationship with table zones on-field CrimeArea.zoneID -> Zones.zoneID. It is a many-to-many type of relationship.

4.7 Database requests

The user can choose two database requests for retrieving criminals depending on zones with two different functions: The first function is "local_known_criminals" and it is used to get the dataset related to the criminals that have committed crimes within the zone. The second function is: "remaining_known_faces" and it is used to get the datasets of criminals that have committed crimes outside the operating zone. Each function will be called according to the desired functionality, but most of the time it makes sense to call the first function.

The datasets of criminals are spitted into zones, in my implementation I have three zones that covered all the land and islands of Greece. More specifically, the zones are:

- Zone 1: Epirus, Thrace, Macedonia, Thessaly.
- Zone 2: Central Greece, Peloponnese, Ionian Islands.
- Zone 3: Crete, Aegean islands.

The distribution of areas in zones aims to improve the efficiency of the machine learning algorithm. If for example, the dataset was the same for all zones it would make the algorithm very dysfunctional due to its enormously large size. By splitting the datasets into zones, the algorithm's efficiency increases significantly because the variations in datasets are much smaller. It makes sense to create even more zones, but this is dependent on the total amount of data that exists in each dataset.

Finally, if a criminal is detected the user retrieves all the necessary information needed to display to the screen with the function "GetCriminalDataById" which returns a dictionary.

4.8 Input system

To make the application directly usable from the ground control unit I provided an input system that can be utilized by any standard keyboard. It is worth mentioning that this is not something mandatory to the general functionality of the software. Nevertheless, below I will analyze the keys and the functionality of each key:

The keys used for the application are: "Q", "E", "P", "W"," A"," S"," D", "LEFT ARROW", "RIGHT ARROW", "UP ARROW", "DOWN ARROW".

- Q: This key is used for drone take-off.
- E: This key is used to land the drone.
- P: This key is used to take a photo if needed. (This functionality is not in the main part of the application).

- W: This key is used to move the drone forward.
- A: This key is used to move the drone backwards.
- S: This key is used to move the drone left.
- D: This key is used to move the drone right.
- RIGHT ARROW: This key is used to rotate the drone clockwise.
- LEFT ARROW: This key is used to rotate the drone anticlockwise.
- UP ARROW: This key is used to raise the altitude of the drone.
- DOWN ARROW: This key is used to lower the altitude of the drone.

To get the user input I used "pygame" which is a very famous python library for game development with python. In this application to get the input the user must click inside a black window that will pop up automatically when the application is initialized. To prompt that window, I used the script in the image below:

```
import pygame

def initializeWindow():
    pygame.init()
    window = pygame.display.set_mode((600,600))

def getKey(keyName):
    ans = False
    for eve in pygame.event.get():pass
    keyinput = pygame.key.get_pressed()
    myKey = getattr(pygame, 'K_{}'.format(keyName))
    if keyinput[myKey]:
        ans = True
    pygame.display.update()

    return ans
```

Figure 8: Shows the code to display the main window of the application.

Moving forward, to take the input user every frame I created a python function that returns a dictionary with the input from the user.

```
def getKeyboardButtonPressed():
    controls = {
        "up_down": 0,
        "forward_backwards": 0,
        "left_right": 0,
        "rotateLeft_right": 0,
        "launch": 0,
        "land": 0,
        "picture": 0
    }
    if dc.getKey('p'):
        controls["picture"] = 1
    if dc.getKey('q'):
        controls["launch"] = 1
    if dc.getKey('e'):
        controls["land"] = 1
    if dc.getKey('UP'):
        controls["up_down"] = 100
    if dc.getKey('DOWN'):
        controls["up_down"] = -40
    if dc.getKey('w'):
        controls["forward_backwards"] = 100
    if dc.getKey('s'):
        controls["forward_backwards"] = -100
    if dc.getKey('a'):
        controls["left_right"] = -40
    if dc.getKey('d'):
        controls["left_right"] = 40
    if dc.getKey('LEFT'):
        controls["rotateLeft_right"] = -40
    if dc.getKey('RIGHT'):
        controls["rotateLeft_right"] = 40

    return controls
```

Figure 9: shows the code for input system.

Finally, to convert the user input into a drone command I used the predefined function from the tello python library:

```
drone.send_rc_control(instructions['left_right'], instructions['forward_backwards'], instructions['up_down'], instructions['rotateLeft_right'])
```

Figure 10: shows drone command that sends the input.

The instructions variable is the dictionary that is been returned from the GetKeyboardPressed function.

4.9 Graphical user interface

When the drone detects a face that has a match equal to or greater than 55% with a criminal face, a graphical user interface is prompted on the screen with all the criminal's known data and crimes. The libraries used for this GUI can be seen in the image below.

```
from tkinter import *
from tkinter import ttk
from PIL import Image, ImageTk
import DatabaseRequests as dr
```

Figure 11: Necessary imported libraries all the data displayed are entirely fictional the face is mine. This image aims only to demonstrate how the app will work if a criminal is detected.

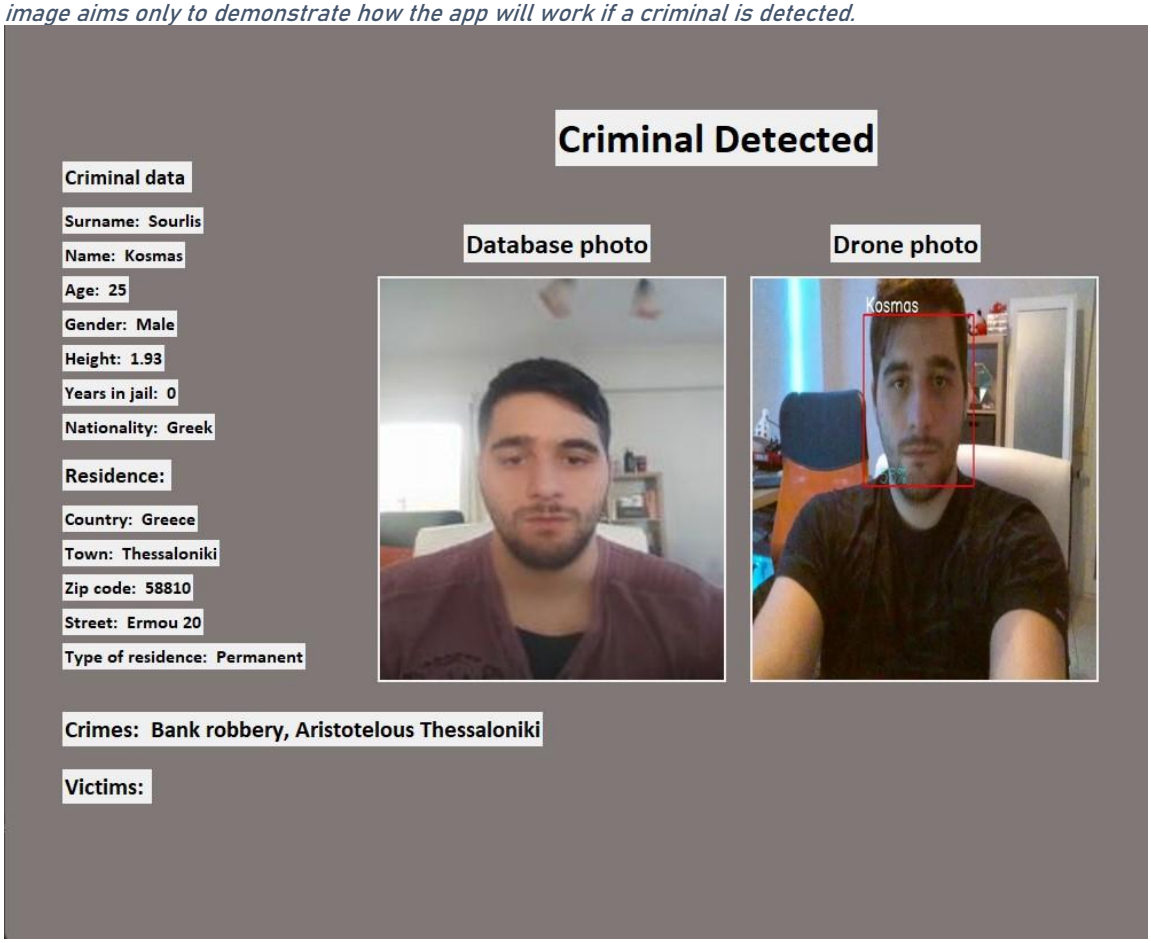


Figure 12: shows the criminal detection popup screen.

4.10 Data preprocessing

Data preprocessing is something every data scientist must do before applying the artificial intelligence algorithm. For this application, the data preprocessing is done with two different techniques. The first acts on an existing dataset, potentially 50 or 60 photos of a criminal face. The second is by recording a video and taking one photo per frame. Nevertheless, below I will analyze and present each way of data preprocessing.

Existing Dataset technique:

This technique requires an already existing dataset of criminal photos that have no other faces in it. The script below is used to crop each photo and modify the color of the picture.

```
import os
from PIL import Image
import cv2
photos = os.listdir('./user_2_Pre_crop_images')
face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
for img in photos:
    imgpath = img
    image = cv2.imread('\\PoliceDroneSoftware\\user_2_Pre_crop_images\\' + imgpath)
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    faces = face_detector.detectMultiScale(gray, 1.3, 5)
    for (x,y,w,h) in faces:
        cv2.rectangle(image, (x,y), (x+w,y+h), (255,0,0), 2)
        cv2.imwrite('\\PoliceDroneSoftware\\user_2_crop_images\\' + imgpath, gray[y:y+h,x:x+w])
```

Figure 13: data preprocessing technique 1

In the first three lines, the libraries need for these operations are imported. Moving forward, the variable photos store the preprocessed images of the criminal. The line with the variable face_detector uses the cv2 algorithm which recognizes the faces of the photo. Finally, the two nested loops iterate through all the preprocessed images, crop all the faces, and store them in the directory “user_userID_images” in grey color. If there is only one face on the photo the second loop will run only once, it should be only one face in the photo, if there are more than one face in the photo the algorithm will be confused.

Video technique:

This technique can function on live video streaming or with an already recorded video. Nevertheless, the script below runs on live video streaming.

```
def PreprocessImgs(img):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    faces = face_detector.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        milliseconds = int(round(time.time() * 1000))
        length = (len(os.listdir('./user_4_crop_images')))
        if (length < 65):
            cv2.imwrite("user_4_crop_images/User_4_" +str(milliseconds) + ".jpg", gray[y:y + h, x:x + w])
```

Figure 14: Data preprocessing technique 2

This function is called every frame from a different part of the application so there is only one for a loop. Similar to the first technique the face detector variables store faces from each frame with the usage of the "cv2.CascadeClassifier" function. The color in which each photo is stored is grey. The name of each photo that is stored is random and therefore it is saved with milliseconds of the current time. The total photos of each user is 65 and that is restricted by the if statement: "if (length < 65)". The length variable stores the number of photos stored for each user.

4.11 Training the model

After preprocessing the data, the desired output of each photograph should be:

Figure 15: Ideal image for training

The algorithm that is used for training and therefore creating a dataset is LBPH which stands for (local binary pattern histogram). More specifically, in this application implementation of OpenCV of this particular algorithm. This algorithm is very famous for providing extremely efficient results on frontal and side face recognition (Zhao, X. and Wei, C., 2017). In this application, the algorithm is only trained for front face recognition. Also, this algorithm works better when there is a lot of light in the image (Jagtap et al, 2019). Bellow the script that is to

create to train the datasets. Each dataset is trained separately for each zone.

```
import cv2
import numpy as np
from PIL import Image
import os
recognizer = cv2.face.LBPHFaceRecognizer_create()
face_detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml");
paths = ['./user_1_crop_images','./user_2_crop_images','./user_4_crop_images','./user_4_crop_images']
faceSamples = []
ids = []
for path in paths:
    photos = os.listdir(path)
    img_prefix_path = os.path.abspath(path)
    for photo in photos:

        img = Image.open(str(img_prefix_path) +'\\'+ photos[0]).convert('L')
        img_numpy = np.array(img, 'uint8')
        array_to_get_id = path.split('_')
        id = array_to_get_id[1][0]
        faces = face_detector.detectMultiScale(img_numpy)
        for (x, y, w, h) in faces:

            faceSamples.append(img_numpy[y:y + h, x:x + w])
            print(img_numpy[y:y + h, x:x + w])
            print()
            ids.append(int(id))

recognizer.train(faceSamples, np.array(ids))
# Save the model into trainer/zone1.yml
recognizer.write('trained_model/zone1.yml')
```

Figure 16: Training

In the first lines the libraries needed for training the model are imported. Moving forward variable recognizer stores the artificial intelligence algorithm that is needed to train the model, the face_detector variable stores the functionality that is required to detect the faces on the photograph, paths variable stores all the paths of photographs that will be trained for each zone dataset, face samples and ids variables store the photos and ids of each criminal (Khan et al, 2019). There are three nested for loops the first one loops through each path of criminal photos, the second loop iterates through all photos of each criminal and the last loop must run only once because it must run for only one face. In the last 2 lines: "recognizer.train(faceSamples, ,nparray(ids))", and "recognizer.write("trained_model/zone1.yml")" the model is trained and the dataset is stored. Notice that each face is stored with a unique Id, this Id is the same as the database criminal Id.

4.12 Recognizing the face in real-time

After training the models and separate datasets have been creating for each zone. Depending on the zone the user wants to detect faces different datasets will be implemented for the function. In the script below you can see the function that recognizes the criminal's faces.

```
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read(zone_dataset)
cascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath)
font = cv2.FONT_HERSHEY_SIMPLEX
def CriminalDetection(img):
    imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(imgGray, 1.2, 4)
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    for (x,y,w,h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h), (0, 0, 255), 2)
        id, confidence = recognizer.predict(gray[y:y+h, x:x+w])
        if (confidence >= 55):
            print(id)
            id = criminal_data["name_criminal"]
            confidence = "  {0}%".format(round(100 - confidence))
            GUI.PopUpWindow(criminal_data, id, photo_url)
        else:
            id = "unknown"
            confidence = "  {0}%".format(round(100 - confidence))
        cv2.putText(img,str(id),(x + 5, y - 5),font, 1,(255, 255, 255),2)
        cv2.putText(
            img,
            str(confidence),
            (x + 5, y + h - 5),font,1, (255, 255, 0),1)
    return img
```

Figure 17: Facial recognition from dataset that was created

In the first two lines the variable recognizer stores the algorithm that can provide that functionality needed to recognize a face from the desired dataset, face cascade variable helps to identify any face from the video. The function CriminalDetection is called every frame and it detects any face present on the frame. The for loop will run only if a face is present on the image frame. If multiple faces exist in the image frame, then it will iterate through all the faces, predict whether the face exists in the dataset, and display a match percentage as well as the name of the criminal. The latter has been achieved with the command “recognizer.predict()” and the popup screen only appears if the match percentage is greater or equal to 55%. If the face is not present in the dataset, then the displayed name will be unknown and the match percentage will either be 0% or a value below 10%.

5. Experimental Results (Testing/Evaluation)

In my implementation I tested two different data preprocessing techniques and concluded that the second with live streaming video was more efficient than the first one. The latter is because the first technique that used an existing dataset of photos had a wide variety of photos, and the lighting was different as well as the angle of the face, therefore it provided less efficient results compared to the live video streaming technique which recorder 65 images, each

one with very similar angle and the same lighting. Also, I found out that with both techniques the best number of images per face was around 50-70 if I trained the algorithm with fewer images per face then the results became significantly less efficient. Furthermore, I noticed that if the person were glasses or a mask the success rate was substantially reduced. Interesting research related to mask detection can be seen at Mohammed Ali and Al-Tamimi, 2022. Moreover, I tested the application with Ryze tech Tello drone, which is a drone for mainly educational purposes, and realized that the drone only recognized faces from a close distance if I moved the drone further away the camera was not able to recognize the face and even if it did the match percentage was really low approximately 20-30%. Therefore, it makes sense to use a drone with a much better camera that can detect faces from greater distances and be more efficient.

Throughout the development process, I tried to implement the face recognition part of the application with many different algorithms and technologies. The problem with these different algorithms was that almost no algorithm could provide satisfactory match percentages, and none was able to run without too much processing power in real-time and that was an important obstacle since processing power is essential for this type of application. Many produced satisfactory results but not on live stream. More specifically, "keras_vggface" has many useful algorithms but could not run well in real-time. It produced very insufficient results. OpenCV cascade classifier for frontal faces appeared to be the best lightweight pre-trained model that could identify faces from the front side. Finally, OpenCV had great algorithms for training, met all the requirements of the application, and worked extremely well on every part of artificial intelligence.

Although OpenCV seem to be the perfect choice for this implementation when I first run the application, I created a single dataset with all the criminal information combined. The problem was that the application could not meet satisfactory matching percentages due to many variations in data. It produced match percentage around 35-45 percent on each face. After the datasets were divided into zones this percentage rose significantly up to 80 percent. So, the most innovative part of this implementation is the distribution of datasets in different zones, it is what made this application radically more successful and efficient on face prediction. The latter is something that must be kept in mind when designing these kinds of datasets because many times data distribution can lead to more efficient results.

6. Conclusion

In conclusion, drones can become incredibly useful in many fields. So far, we have seen numerous applications in many fields such as law enforcement, agriculture, health care, etc. Drones can contribute in many ways, their very high-quality sensors and high-resolution cameras have pushed the scientific community to develop various types of applications. This valuable hardware combined with artificial intelligence and specifically facial recognition algorithms has opened a new field of opportunities. In this study, an implementation related to drones and computer vision was developed. From this implementation, I concluded that running a facial recognition algorithm in real time was complicated and required high energy consumption. To face these challenges the facial recognition algorithm needs to be as light as possible and the dataset on which the algorithm functions must not have many different variations. Hence, many smaller datasets were created, and each dataset was related to a specific zone. Also, data preprocessing proved to be the most essential part of the application.

Two different data preprocessing techniques were tested and the most efficient was chosen after experimenting with both. The future looks bright for drones and many more applications are yet to be created and contribute to the world. The latter is expected to raise privacy and security threats that need to be managed.

7. Future Work and Directions

In my opinion, my implementation is a great combination of emerging technologies and it produced very strong results. Nevertheless, I strongly believe that there are some future works and improvements that can be made that would significantly improve the application. To begin with, the facial recognition algorithm implemented covers only the front of the face, in the future, it can be improved so it can cover the side face. Additionally, the algorithm can expand and also recognize a criminal from his movement (walk, run, etc) but that can be extremely difficult due to high level of complexity. Furthermore, when detecting a criminal an option can be given to the user whether he wants to automatically follow the criminal and track his movement. The latter will require an autonomous flight system and that would raise numerous security concerns, but it would help greatly the operator of the drone. Also, the input system can be significantly improved, it would be better not to use the keyboard and build a dedicated controller to have more efficient control of the drone, control the speed the altitude, etc. Another thing that can be improved is the algorithm itself, in my implementation I managed to get up to 80 percent match on a face, but that worked when the scene was filled with very good lighting. Therefore, it would make sense to develop an algorithm that would work well in darker environments and produce better results. Moreover, the application can be modified to function on regular CCTV cameras, which would be quite tricky because CCTV cameras often do not have great resolution. Also, the database schema must be modified to fit the standards of each law enforcement unit, the existing database schema is generic, and is designed to be flexible and easily convertible. To implement this kind of application in society many obstacles need to be overcome, solutions must be found to bureaucracy and eliminate security and privacy concerns. Finally, drones are extremely prone to cyber-attacks and therefore it is essential to ensure safety during flight time. UAVs often deal with very sensitive information that can lead to very unpleasant situations. For example, in my application it would be very unfortunate for the video transmission to be leaked, therefore it would make sense to apply a cryptographic algorithm during data transformation that would prevent that from happening. It will also make sense to provide good authentication protocols when connecting to the drone to prevent unauthorized access to the drone itself.

8. Bibliography

Ali Alheeti, K., Al-Ani, M., Najem Al-Aloosy, A., Alzahrani, A. and Sattar Rukan, D., 2022. Intelligent mobile detection of cracks in concrete utilising an unmanned aerial vehicle. *Bulletin of Electrical Engineering and Informatics*, 11(1), pp.176-184.

Jagtap, A., Kangale, V., Unune, K. and Gosavi, P., 2019. A Study of LBPH, Eigenface, Fisherface and Haar-like features for Face recognition using OpenCV. *2019 International Conference on Intelligent Sustainable Systems (ICISS)*.

Jones, G., 2019. *The Use of Armed Drones by Law Enforcement: A Need for Essential Elements in Policy - ProQuest*. [online] Proquest.com. Available at: <https://www.proquest.com/openview/5b80ddfcaff6cb434066d641be5ce32b/1?pq-origsite=gscholar&cbl=51922&diss=y&fbclid=IwAR2_8asOhUe1gItA-Df2KClvOrZefRejreU50eN90IVYHhqQ5rKDA_HIEHM> [Accessed 22 May 2022].

Khan, M., Chakraborty, S., Astya, R. and Khepra, S., 2019. Face Detection and Recognition Using OpenCV. *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*.

Labib, N.S., Brust, M.R., Danoy, G. and Bouvry, P. (2021). The Rise of Drones in the Internet of Things: A Survey on the Evolution, Prospects, and Challenges of Unmanned Aerial Vehicles. *IEEE Access*, 9, pp.115466–115487.

Lee, D., Gyu La, W. and Kim, H., 2018. Drone Detection and Identification System using Artificial Intelligence. *2018 International Conference on Information and Communication Technology Convergence (ICTC)*.

Lin, C., He, D., Kumar, N., Choo, K.-K.R., Vinel, A., and Huang, X. (2018). Security and Privacy for the Internet of Drones: Challenges and Solutions. *IEEE Communications Magazine*, 56(1), pp.64–69.

Ling, G. and Draghic, N., 2019. Aerial drones for blood delivery. *Transfusion*, 59(S2), pp.1608-1611.
Merwe, D., Burchfield, D., Witt, T., Price, K. and Sharda, A., 2020. Drones in agriculture. *Advances in Agronomy*, pp.1-30.

Michailidis, E. and Vouyioukas, D., 2022. A Review on Software-Based and Hardware-Based Authentication Mechanisms for the Internet of Drones. *MDPI*, 6(2), p.41.
Mohammed Ali, F. and Al-Tamimi, M., 2022. *Face mask detection methods and techniques: A review*. [online] Dx.doi.org. Available at: <http://dx.doi.org/10.22075/ijnaa.2022.6166> [Accessed 21 May 2022].

Oruc, A., 2022. Potential cyber threats, vulnerabilities, and protections of unmanned vehicles. *Drone Systems and Applications*, 10(1), pp.51-58.

Ouiazzane, S., Addou, M. and Barramou, F., 2021. A Multiagent and Machine Learning Based Denial of Service Intrusion Detection System for Drone Networks. *Geospatial Intelligence*, pp.51-65.

Pulli, K., Baksheev, A., Kornyakov, K. and Eruhimov, V., 2012. Real-time computer vision with OpenCV. *Communications of the ACM*, 55(6), pp.61-69.

Restás, Á., 2022. Drone Applications Fighting COVID-19 Pandemic—Towards Good Practices. *MDPI*, 6(1), p.15.

Shafique, A., Mehmood, A. and Elhadef, M. (2021). Survey of Security Protocols and Vulnerabilities in Unmanned Aerial Vehicles. *IEEE Access*, 9, pp.46927–46948.
Tanveer, M., Khan, A., Nguyen, T., Ahmad, M. and Abdei-Latif, A., 2022. Towards A Secure and Computational Framework for Internet of Drones Enabled Aerial Computing. *IEEE Transactions on Network Science and Engineering*, pp.1-1.

Tutorialspoint.com. 2022. *AI with Python â□" Computer Vision*. [online] Available at: <https://www.tutorialspoint.com/artificial_intelligence_with_python/artificial_intelligence_with_python_computer_vision.htm> [Accessed 21 May 2022].

Veerappan, C., Loh, P. and Chennattu, R., 2022. Smart Drone Controller Framework—Toward an Internet of Drones. *AI and IoT for Smart City Applications*, pp.1-14.

Voulodimos, A., Doulamis, N., Doulamis, A. and Protopapadakis, E., 2018. Deep Learning for Computer Vision: A Brief Review. *Computational Intelligence and Neuroscience*, 2018, pp.1-13.

Whelan, J., Almehmadi, A. and El-Khatib, K., 2022. Artificial intelligence for intrusion detection systems in unmanned aerial vehicles. *Computers and Electrical Engineering*, *99*, p.107784.

Wulfovich, S., Rivas, H. and Matabuena, P., 2018. Drones in Healthcare. *Health Informatics*, pp.159-168.

Yaacoub, J., Noura, H., Salman, O. and Chehab, A., 2020. Security analysis of drones systems: Attacks, limitations, and recommendations. *Internet of Things*, 11, p.100218.
Zhao, X. and Wei, C., 2017. A real-time face recognition system based on the improved LBPH algorithm. *2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*.

Zègre-Hemsey, J., Bogle, B., Cunningham, C., Snyder, K. and Rosamond, W., 2018. Delivery of Automated External Defibrillators (AED) by Drones: Implications for Emergency Cardiac Care. *Current Cardiovascular Risk Reports*, 12(11).

Zhu, P., Wen, L., Bian, X., Ling, H. and Hu, Q., 2018. Vision meets drones: A challenge. arXiv preprint arXiv:1804.07437.

Zhu, Z. and Cheng, Y., 2020. Application of attitude tracking algorithm for face recognition based on OpenCV in the intelligent door lock. *Computer Communications*, 154, pp.390-397.