AUTOMATIC *AEDES AEGYPTI* BREEDING GROUNDS DETECTION USING COMPUTER VISION TECHNIQUES

Wesley Lobato Passos

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: Eduardo Antônio Barros da Silva
Gabriel Matos Araujo

Rio de Janeiro
Fevereiro de 2019

AUTOMATIC *AEDES AEGYPTI* BREEDING GROUNDS DETECTION USING COMPUTER VISION TECHNIQUES

Wesley Lobato Passos

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

RIO DE JANEIRO, RJ – BRASIL
FEVEREIRO DE 2019

Passos, Wesley Lobato

Automatic *Aedes aegypti* Breeding Grounds Detection Using Computer Vision Techniques/Wesley Lobato Passos. – Rio de Janeiro: UFRJ/COPPE, 2019.

XII, 77 p.: il.; $29,7$cm.

Orientadores: Eduardo Antônio Barros da Silva
Gabriel Matos Araujo

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2019.

Referências Bibliográficas: p. 67 – 77.

1. Aedes aegypti. 2. Aerial images. 3. image processing. 4. computer vision. 5. machine learning. I. Silva, Eduardo Antônio Barros da *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

À minha família.

Agradecimentos

Em primeiro lugar agradeço a Deus, sem o qual nada disso seria possível. Eu agradeço a Ele por ter me dado forças durante esta caminhada e me por ter me abençoado colocando pessoas especiais na minha vida.

Aos meus pais, Edson e Penha, e minha irmã, Maysa, que me acompanham e incentivam desde os meus primeiros passos.

À minha amada esposa Thallita por sempre me apoiar e acreditar em mim, mesmo nos piores momentos. Sem sua ajuda, a conclusão deste ciclo não seria possível. Agradeço também aos meus sogros, José Luiz e Átila que são como novos pais para mim.

Aos professores Eduardo A. B. da Silva e Gabriel Araujo por aceitarem me orientar e trabalhar neste projeto tão desafiador.

Aos Eng. Leonardo Nunes e Prof. Sergio Lima Netto por aceitarem o convite para compor a banca de avaliação deste trabalho.

Aos amigos do SMT, em especial, Felipe Barboza, Felipe Ribeiro, Gabriel, Lucas Cinelli, Marcelo Spelta, Matheus, Igor, Rafael Chaves, Rafael Padilla, Roberto e Vinicius.

Aos alunos do laboratório de processamento de sinais do CEFET/RJ *Campus* Nova Iguaçu pela ajuda no processo de anotação base de dados que está sendo desenvolvida.

Por fim, agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro (Código de Financiamento 001).

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

DETEÇÃO AUTOMÁTICA DE CRIADOUROS DE *AEDES AEGYPTI* USANDO TÉCNICAS DE VISÃO COMPUTACIONAL

Wesley Lobato Passos

Fevereiro/2019

Orientadores: Eduardo Antônio Barros da Silva
Gabriel Matos Araujo

Programa: Engenharia Elétrica

Todos os anos, milhares de pessoas são afetadas por doenças como dengue, chikungunya, zika e febre amarela. Todas essas doenças são transmitidas pelo *Aedes aegypti*, que se reproduz em água limpa e parada, usualmente acumulada em recipientes como pneus, garrafas, caixas d'água etc. O uso de ferramentas inteligentes pode auxiliar no trabalho dos agentes de fiscalização dos focos deste mosquito, aumentando, assim, a eficiência e área de cobertura. Esse trabalho aborda o problema de detecção automática de focos de mosquitos através do uso de técnicas de visão computacional e aprendizado de máquina. Nesse contexto, propõe-se um conjunto de vídeos aéreos, adquiridos através de um veículo aéreo não tripulado. O conjunto possui diversos desses objetos em múltiplos cenários: diferentes localidades, altitudes e disposições dos objetos. Os vídeos são devidamente retificados para amenizar distorções da câmera e manualmente anotados quadro-a-quadro, viabilizando o desenvolvimento de um detector automático de objetos de interesse.

Um detector do tipo *Faster Region-based Convolutional Neural Network* é treinado com uma pequena base de dados, e é capaz de encontrar possíveis focos de mosquito de maneira automática. O modelo gerado atinge uma precisão média de 49,31%, o que é promissor, indicando que novos e melhores modelos podem ser treinados para este fim.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

AUTOMATIC *AEDES AEGYPTI* BREEDING GROUNDS DETECTION USING COMPUTER VISION TECHNIQUES

Wesley Lobato Passos

February/2019

Advisors: Eduardo Antônio Barros da Silva
Gabriel Matos Araujo

Department: Electrical Engineering

Every year, thousands of people are infected with diseases such as dengue, chikungunya, zika, and yellow fever. These diseases are transmitted by the *Aedes aegypti*, which usually reproduces in containers with accumulated clean water, such as tires, bottles, water tanks, etc. The use of intelligent tools can be employed to assist health agents in a search for these objects, providing more efficiency and coverage in this process. This work addresses the problem of automatic detection of such mosquito breeding grounds using computer vision and machine learning techniques. In this context, a new aerial videos dataset is devised including such objects in different scenarios: distinct backgrounds, altitudes, object displacement, and so on. The videos are rectified in order to compensate for camera distortions and manually annotated, frame-by-frame, enabling the development of an automatic detector for the target objects.

A Faster Region-based Convolutional Neural Network detector is trained, using a small dataset, and is capable of finding potential mosquito foci. This model achieves 49.31 points of average precision, which is promising, indicating that new and better models can be trained for this task.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The *Aedes aegypti* is the main vector of several diseases caused by the arbovirus (acronym for arthropod-borne virus), such as dengue, zika, chikungunya and, more recently in Brazil, urban yellow fever [1, 2]. Zika virus disease can be quite dangerous for pregnant women due to its correlation with the microcephaly, a congenital fetus brain malformation [3, 4]. Among these diseases, dengue is the one that causes the most deaths, with about 390 million people infected per year in the world [5]. Yellow fever also has a high rate of mortality and chikungunya can incapacitate those infected for long periods.

By considering the high rates of lethality and eradication difficulty, arboviruses transmitted by *Aedes aegypti* are one of the leading global health problems. Wherefore, the World Health Organization (WHO) launched, in 2012, a comprehensive strategy for dengue control and prevention [6], whose one of the goals is to reduce disease cases by 25% by 2020. Unfortunately, combat tools are still limited: the dengue vaccine remains in the improvement phase, and the fumes against mosquitoes are ineffective [7]. Thus, the current best form of combat is through the control and elimination of possible mosquito foci proliferation, which acts directly in the prevention of all these diseases. Given that the *Aedes aegypti* reproduces in clean and stagnant water, the main mosquito foci are open water bowls, gutters, tires, bottles, plant pots, and any container that can collect water.

As a result, monitoring and controlling the mosquito without proper technical support is expensive, time-consuming and therefore inefficient. For that reason, allying the knowledge of an expert with a tool that accelerates the search for potential mosquito foci and towards a more precise work is extremely important in the current scenario. Thus, using images and videos captured by an unmanned aerial vehicle (UAV), better known as a drone, with several sensors and camera may be a reasonable approach. The objective is to identify objects with high potential of being a mosquito breeding site. This technology has already been used by organizations to visually inspect difficult-to-reach sites in order to locate such breeding spots. In this process, the acquired videos are examined by a specialist, which makes the procedure time-consuming and tiring, what may lead to failures.

In this sense, a possible solution to increase efficiency is to apply machine learning techniques to automate the analysis process, helping the specialist in the decision-making action [8]. After this fast analysis, potential breeding sites can be treated or removed by a team of agents, as usual. It is known, through a local study [9], that treating the most productive water container types in a region has a comparable result as treating them all in the same region. This roughly cuts by half the number of containers to be treated while keeping similar effectiveness, drastically reducing the potential for epidemic development. In the case of Nova Iguaçu, a city located in the state of Rio de Janeiro, the reservoirs listed with high potential were, according to [10]: water tanks, glass and plastic bottles, buckets, tires, and external drains. The initial goal then becomes to automatically recognize as many of these objects as possible in videos or images acquired from a UAV to reduce the amount of images or videos the agents would need to visually evaluate. In the future, we plan to expand this work and provide an intelligent decision support tool for agents, generating heat maps highlighting the places with more risk, thereby increasing the effective area of action.

1.1 Main contributions of this work

Throughout this work, we describe the problem of automatic detection of potential mosquito breeding sites using aerial images acquired from a UAV and propose a complete solution. We construct a video dataset containing objects considered as potential mosquito breeding sites of the *Aedes aegypti* in several scenarios. We apply a methodology to calibrate the camera of our UAV and reduce the lens distortions that may be expanded for other drone models. We train a state-of-the-art object detector using a small dataset in order to detect tires, considered big productive water containers.

1.2 Dissertation organization

In Chapter 2, we do a review on the *Aedes aegypti*, including biologic aspects, transmitted diseases and sequelae, and preferred ground sites. Also, we point out some statistics related to the theme and government plans to combat the transmitter. In Chapter 3, we make a literature review of related themes, in order to see how several techniques of machine learning can be used to address the problem. We also present a new dataset containing the main objects considered as potential foci of the *Aedes aegypti* in several scenarios. In Chapter 4, since we are interested in detecting specific objects, we describe the state-of-the-art algorithm employed in order to accomplish this task. We discuss from classical object detections up to recent deep-learning-based models, particularly the one employed in this work. The evaluation method, implementation details, and results for the method used are discussed in Chapter 5; and we finally conclude at Chapter 6.

Chapter 2

The Aedes aegypti

In order to fight *Aedes aegypti*, we must first know it better. In this chapter, we address many aspects of the mosquito, including its biology, diseases transmitted, and potential ground sites. Also, we point some indicators and discuss plans and actions to combat the *Aedes aegypti*.

2.1 The vector biology

The *Aedes aegypti* is considered one of the most dangerous mosquito species according to public health analysis, considering that it is the major vector of arbovirus transmission [1]. This mosquito species is well adapted to urban environment and inhabit mostly domestic and peridomestic environments [11]. The high capacity of this vector to transmit diseases to humans is mostly due to a set of biological, ecological, and behavioral characteristics that promote a more direct contact with humans. One of these characteristics is synanthropic behavior (live near human dwellings) [11].

The dispersion of *Aedes aegypti* around the world began around the 16th century with the Portuguese maritime routes between Africa and the other continents. Since then, always due to human transport, the mosquito has invaded many of the tropical and subtropical regions of the planet, establishing itself in the Americas, Southeast Asia, Southwest of the United States, islands of the Indian Ocean, and the north of Australia. In areas outside the latitudes that comprise these regions, there have been some sporadic occurrences, even though the species shows an apparent difficulty in establishing a viable population in these places. However, considering the anticipated global climate change, the *Aedes aegypti* may be able to expand its presence beyond customary regions [12]. The proliferation of the vector across the globe is related to the circulation of goods and people between various countries and continents. The eggs of this species are particularly resistant and can survive long journeys and inhospitable environments, besides presenting great adaptability to artificial breeding sites [12].

The *Aedes aegypti* is less than 1 cm long and black with white stripes, as depicted in Figure 2.1. The adult specimen lives on average 45 days, usually stings in the first hours of the morning and the late afternoon. Studies by the Oswaldo Cruz Foundation (FIOCRUZ) have shown that the female flies up to 1 km away from its eggs. Moreover, the *Aedes aegypti* feeds on plant sap. The females are hematophagous, that is, they feed on blood as well. As a consequence, when ingesting the blood of an infected host, it contracts the microorganism responsible for the diseases [14].

The life cycle of *Aedes aegypti*, like other mosquito species, comprises four phases: egg, larva, pupa, and adult. The first three phases develop exclusively in the aquatic environment, while the last phase occurs in the terrestrial/aerial environment. Therefore, the existence of water and breeding suitable for their retention are essential for the mosquito development [11]. *Aedes aegypti* breeding grounds are mostly small containers, either artificial or natural, in or near dwelling sites, which allow the storage of water (water tanks, buckets, ornamental fountains, plant dishes, water canisters for animals, tires, etc). The preference for breeding sites near or in domestic environments relates to a set of habits that promote close contact with humans: synanthropic habits; endophilic (resting inside housing/animal facilities); and anthropophilic [11]. Hence, human activity is a determining factor for the reproduction and dissemination of the *Aedes aegypti*.

Furthermore, the mosquito deposits the eggs under appropriate – hot and water-filled – places. Under these conditions, the embryos take from two to three days to develop and hatch. These embryos may weaken or die if, during this period, the eggs dry out, but if in the initial stage a perfect development is ensured, the eggs of the mosquito become resistant to drying and thus survive for periods ranging from some months to a year [14]. This resistance is one of the major barriers to the

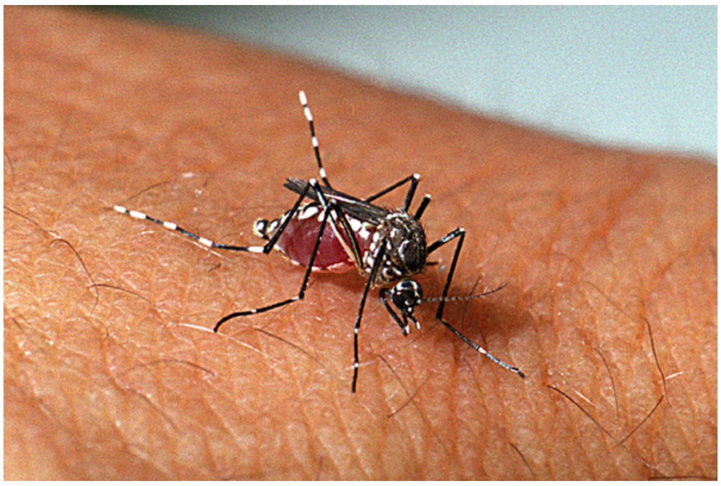


Figure 2.1: The *Aedes aegypti*. Source: [13]

Aedes aegypti elimination. The larval period, which is the feeding and growth stage, depends on temperature, larval density, and availability of food, and in optimal conditions, does not exceed five days. When in low temperature and lack of food, this phase can extend for weeks. The pupa is a phase without nutriment from the embryo to the adult stage (egg, larva, and pupa), which takes on average ten days. On the first or second day after becoming adults, mosquitoes copulate. After mating, females begin to feed on blood since it has the necessary proteins for the development of eggs [14].

The arbovirus can be maintained in the *Aedes aegypti* populations by transovarial transmission in which the female vector passes the infectious agent through the eggs to the next generation [15]. Such transmission is epidemiologically important, as epidemics of dengue, for example, usually enable quick proliferation. That depends on the time it takes for the mosquito to become vector after it stings a viremic person. If mosquitoes already emerge as vectors, *i.e.*, infected and capable of transmitting the virus, without the need to prick an infected person, the probability of vector-human-vector transmission increases, as well as the magnitude of the epidemic. The transovarial transmission also creates the possibility that the *Aedes aegypti* eggs, which carry the virus, spread to other geographic regions by being passively transported [15].

Like other species, *Aedes aegypti* is also particularly sensitive to climatic conditions. Several studies demonstrate the role that factors such as temperature and precipitation have on ecology and vector biology [16]. It is verified that the higher the temperatures, the faster the development of the different phases of the mosquito, and the higher its longevity and fecundity during the adult phase; at lower temperatures, there is a degraded condition to its development, which may jeopardize survival. Concerning precipitation, it favors the creation of potential breeding sites where females lay their eggs and immature forms (larvae and pupa) develop [17].

According to [18], the vector population grows with increasing temperatures. At 32°C the number of mosquito bites is twice as high as at 24°C. Another interesting aspect related to temperature is that the rate of metabolism of the mosquito, its evolutionary cycle, can be extended up to about 22 days in the cold months. On the other hand, in the months of high temperatures, its maturation speed increases. Therefore, the ambient temperature is directly proportional to the time of development of the vector for the adult phase. Nonetheless, temperatures above 40°C decreases the life expectancy of the mosquito [19]. Researches diverge about the relationship between rainfall and the increase of the presence of vectors in the environment. Excessive rainfall could lead to the decline of breeding sites due to flooding, whereas lack of rainfall may lead to the storage of water in domestic reservoirs (vases, dishes, canisters, etc.) and in other locals that can be used for vector reproduction (barrels, wells, water boxes and etc.). However, in abandoned areas with poor structural and sanitary development, rainfall incidence is important to understand the development of breeding sites in fixed deposits (gutters, slabs, glass shards in walls, and other architectural works), solid waste, and other abandoned items. For that reason, there are different correlations between precipitation and the *Aedes aegypti* proliferation along the geographical spaces, being possible to occur positive or negative relations [20].

Finally, the understanding of these biological and ecological aspects of the vector is of great epidemiological importance, since studies of this nature generate information about the reproductive, hematophagy dynamics and vectorial competence, and help to understand the mechanism of arbovirus transmission. This knowledge can assist in the creation of mechanisms for combating and controlling the *Aedes aegypti* .

2.2 Diseases transmitted and sequelae

The *Aedes aegypti* is a transmitter of some diseases, known as arboviruses. Nevertheless, it is important to note that only infected mosquitoes transmit the disease. The main diseases are dengue, zika, chikungunya, and yellow fever. Although not directly relevant to the work developed, this section presents, in a concise way, a panorama of these diseases, their transmission, and their possible sequels.

2.2.1 Dengue

Dengue is the most rapidly spreading mosquito-borne viral disease in the world. During the last 50 years, the incidence of dengue has increased 30 times, and the number of affected countries has been increasing steadily [21]. Today, approximately 3.6 billion people live in more than 100 dengue-endemic countries, and an estimated 284-528 million dengue infections occur annually [22]. It is estimated that 500,000 people with severe dengue require hospitalization each year and about 2.5% of those affected with, die [23].

Dengue fever is an acute febrile disease caused by the arbovirus (arthropod-borne virosis) that has the *Aedes aegypti* as the vector transmission. Dengue virus has four serotypes (DENV-1, DENV-2, DENV-3, and DENV-4) that are genetically and antigenically distinguishable. In Brazil, the four types of virus circulate, being the last one isolated in the State of Roraima since 1991 [24].

This disease can last up to seven days, evolving into spontaneous healing or leaving sequelae. It can manifest itself in dangerous ways, as is the case of hemorrhagic dengue. In a case of suspected classical dengue fever, the patient is diagnosed with acute febrile illness lasting up to seven days accompanied by at least two of the following symptoms: headache; pain around the eyes, in the body and joint; and prostration. The hemorrhagic dengue usually arises, most of the time, when the person is infected more than once by the virus, leading to changes in blood clotting. Therefore, it causes bleeding especially in the eyes, gums, ears, and nose, as well as the appearance of blood in the stool, red skin patinas, vomiting, weak and fast pulse. Classical dengue fever is confirmed by laboratory tests; however, during epidemics, confirmation can be performed through clinical-epidemiological criteria. On the other hand, cases of hemorrhagic dengue need to be confirmed by the laboratory, as well as by specific criteria [25].

Dengue vectors become infected with the virus by feeding on the blood of individuals in the viremia stage. This phase begins one day before the fever and lasts six to eight days after the onset of the disease. In the mosquito, the virus multiplies in its cycle of evolution, discussed in Section 2.1. This is the extrinsic period, after maturation of the mosquito in which it becomes a vector, transmitting the virus through the saliva throughout its lifetime [26].

2.2.2 Zika

Zika is also an arbovirus transmitted by *Aedes aegypti*, being first identified in Brazil, April 2015. The zika virus was given the same name as the place of origin: the Zika forest, in Uganda; identified in sentinel monkeys used to monitor yellow fever, in 1947. The zika virus disease presents a higher risk than other arboviruses, such as dengue, yellow fever, and chikungunya, due to the development of neurological complications such as encephalitis and Guillain Barré syndrome [3]. One well-known neurological complication is the microcephaly; a condition in which the baby's head is smaller than the average. It usually happens when there are problems in the uterus that causes the baby's brain to stop growing properly, which may also occur after birth. Microcephaly-born children often present developmental difficulties. Rarely, children with this condition can develop normally. In addition to congenital microcephaly, many manifestations have been reported among infants up to four months of age exposed to the zika virus in the uterus. These include head malformations, involuntary movements, seizures, irritability, and brain stem dysfunction, with swallowing problems, limb contractures, hearing and vision abnormalities, and brain abnormalities. Other consequences associated with zika virus infection in the uterus may involve spontaneous abortions and stillbirths. The spectrum of congenital abnormalities associated with fetal exposure to this virus during gestation is known as congenital zika virus syndrome [4].

The symptoms of zika are red spots all over the body, red eye, fever, body aches and joints of small intensity. In general, the disease progresses benign, and the symptoms disappear spontaneously after 3 to 7 days. However, joint pain may persist for about a month. According to the Ministry of Health, all sexes and age groups are susceptible to the zika virus; however, pregnant women and older adults are at higher risk of developing the complications aforementioned. These risks are amplified when the person has some chronic illness, such as hypertension and diabetes, even if treated [27].

Thus, like dengue, the primary mode of transmission is by the bite of the vector, but it can also be transmitted through sexual intercourse. According to the Pan American Health Organization (PAHO), the zika virus can be found in semen, blood, urine, amniotic fluid, and saliva as well as fluids found in the brain and spinal cord [27].

The zika's diagnosis is based on the patient's new symptoms and history (such as mosquito bites or trips to areas with virus circulation). Moreover, laboratory tests may confirm the presence of zika in the blood; however, this diagnosis may not be as reliable as the virus could react with other viruses such as dengue and yellow fever [27].

2.2.3 Chikungunya

Chikungunya fever is a viral disease transmitted by the mosquitoes *Aedes aegypti* and *Aedes albopictus*, and its circulation was first identified in Brazil in 2014. Chikungunya means "those who fold" in Swahili, one of the languages of Tanzania, located in East Africa. That refers to the curved appearance of patients who were seen in the first documented epidemic there, between 1952 and 1953 [28]. The main symptoms are rapid onset fever, intense pain in the joints of the feet and hands, in addition to fingers, ankles, and wrists. There may also be a headache, muscle aches and red spots on the skin. It is not possible to have chikungunya more than once. Previously infected, a person becomes immune to lifetime. The symptoms begin between two and twelve days after the mosquito bite. The mosquito gets the CHIKV virus by stinging an infected person during the period the virus is present in the infected organism. The incubation period of the virus in the human is 4 to 7 days [28]. The chikungunya can also be transmitted from the pregnant woman to the fetus, but this only occurs when the mother becomes ill in the last week of gestation. In this case, the child who is born healthy remains hospitalized for observation and immediate treatment. This procedure is adopted because if the disease develops, the child may present severe pictures with neurological and skin manifestations. Besides, some factors contribute to the enduring complications of the disease, as advanced age, being a woman and already possess other diseases such as diabetes and rheumatoid arthritis. The CHIKV virus can generate lasting sequelae such as persistent inflammation in the joints, especially the hands and feet. Although joint pain is the most frequent chronic complication, it is not the only one. There is a possibility that the virus can trigger neurological problems such as Guillain Barré syndrome, encephalitis, and other complications [29].

2.2.4 Yellow fever

Yellow fever is an acute febrile infectious disease, caused by a virus transmitted by mosquito vectors, and has two cycles of transmission: wild (when there is transmission in rural or forest areas) and urban. There is no direct transmission from person to person. Yellow fever is epidemiologically important because of its clinical severity and potential for dissemination in urban areas infested by the *Aedes aegypti* . Early symptoms of yellow fever include sudden onset of fever, chills, severe headache, back pain, general body aches, nausea and vomiting, fatigue and weakness. Most people get better after these initial symptoms. However, according to the Ministry of Health, about 15% have a brief period of hours a day without symptoms and then develop a more severe form of the disease. In severe cases, the person may develop a high fever, jaundice (yellowing of the skin and whites of the eyes), bleeding (especially from the gastrointestinal tract), and eventually multiple organ failure and shock. About 20% to 50% of people who develop severe illness might die [2].

The vaccine is the primary tool for prevention and control, unlike the diseases previously analyzed; the Brazilian government offers the vaccine against yellow fever for the population through the Sistema Único de Saúde (SUS) – the Brazilian public healthcare. As aforementioned, there are two different epidemiological cycles of transmission, the wild and the urban. The disease has the same characteristics from the etiological, clinical, immunological and pathophysiological point of view. In the wild cycle of yellow fever, nonhuman primates (monkeys) are the main hosts and amplifiers of the virus. The vectors are mosquitoes with strictly wild habits, with the genera Haemagogus and Sabethes being the most important in Latin America. In this cycle, man participates as an accidental host when entering forest areas. In the urban cycle, man is the only host with epidemiological importance and transmission occurs from infected urban vectors, the *Aedes aegypti* [30].

2.3 Potential grounds, reproduction, and indicators

As could be observed, both the persistence and the progression of the arboviruses are conditioned to the survival and reproduction of their vector in the environment. Therefore, in the context where there is no vaccine for the diseases presented, with exception of yellow fever, the best prevention is to avoid the vector proliferation. The presence of this mosquito specie is more common in urban areas, and the infestation is more intense in regions with a high population density and low vegetation, where females have more opportunities for food and have more places to spawn [14]. Another critical factor is the lack of infrastructure of some localities. Without a regular supply of water, residents need to store it in large containers that do not receive the necessary care and end up becoming mosquito breeding sites, because they are not entirely closed. Therefore, efforts to control mosquito proliferation are indeed related to government measures (providing regular supply of potable water, specially for poor communities) and population commitment.

Since 2017, the Brazilian Ministry of Health has issued Resolution No. 12, which makes it obligatory for entomological surveys of the *Aedes aegypti* infestation by municipalities and reporting it. Monitoring the numbers and geographical distribution of mosquitoes over time helps in making timely decisions about how best to manage vector populations. Surveillance can be used to identify areas with a mosquito-borne high-density infestation or periods in which its population has increased [31].

The *Aedes aegypti* Rapid Index Survey (LIRAa, from Portuguese *Levantamento Rápido de Índices para Aedes aegypti*) [32] is a methodology used by the Brazilian Ministry of Health that allows estimating the number of properties with the presence of containers with mosquito larvae. In each city, health agents visit at least 8,100 different homes or other types of properties, chosen respecting some criteria, every year, in order to inspect and identify breeding sites [33]. One agent is provided for every 800 to 1,000 propeties and each agent is expected to visit from 20 to 25 properties in a usual day of work (8h).

In the case of finding larvae or pupae, the agents collect them for laboratory analysis. According to the National Guidelines for Prevention and Control of Dengue Epidemics (2009) [33], the parameters for classification of municipalities regarding the predial infestation index (IIP), the ratio of the number of properties where larvae were found to the total of properties visited, are: (i) satisfactory if less than 1%; (ii) alert if between 1% and 3.99%; and (iii) risk if above 3.99%. The results obtained by this methodology allow the managers to evaluate vector control activities, besides indicating the most used deposits by *Aedes aegypti*.

The LIRAa classifies the deposits considered as potential breeding sites for the *Aedes aegypti* in five groups, in order to inform their epidemiological importance, facilitating the targeting of vector control and surveillance actions, as shown in Table 2.1. The state of Rio de Janeiro epidemiological report of 2018 shows that type A2, B, C, and D2 deposits account for 84.7% of the 5,608 breeding sites found in the whole territory. From all 92 municipalities in Rio de Janeiro, 91 (98.9%) performed the LiRAa. From these, 45 (49.5%) are classified as satisfactory, 43

Table 2.1: The classification of potential breeding sites for the *Aedes aegypti*, according to LIRAa.

Code	Description
A1	Water tank connected to the grid (high tanks)
A2	Deposits at ground level (barrel, tub, drum, tank, well)
B	Mobile containers (vases/jars, plates, drippings, drinking fountains, etc)
C	Fixed deposits (tanks, gutters, slabs, etc)
D1	Tires and other rolling materials
D2	Garbage (plastic containers, bottles, cans, scraps)
E	Natural deposits (bromeliads, bark, tree holes)

(47.3%) in the alert, and 3 (3.3%) in risk [34].

At the national level, 5,358 municipalities, 96.2% of the total, performed some transmitter monitoring. The Brazilian Ministry of Health indicates a reduction in the three diseases transmitted by the *Aedes aegypti* between January and October 2018, compared to the same period in 2017; however, some states show a significant increase in cases of dengue, zika, and chikungunya. In 2018, 241,664 cases of dengue were reported over the country, there was a small increase compared to the previous year (232,372). Fortunately, the number of deaths reduced from 176 to 142. Besides, there was a reduction of 54% concerning the previous year in the notification of cases of chikungunya, from 184,344 to 84,294 in 2018. The number of deaths followed this drop, reducing from 191 to 35, a significant fall of 81.6%. Concerning to zika, there was also a reduction in the number of cases, from 17,025 to 8,024 [35] and four deaths were reported.

Regarding costs, the resources for health surveillance actions, which include combating *Aedes aegypti*, have grown in recent years, from R$ 924.1 million in 2010 to R$ 1.94 billion in 2017 [36]. Dengue epidemics place a heavy burden on health services and countries' economies. Despite a few studies on this subject, a recent work conducted in eight countries in the Americas and Asia, including Brazil, has shown that the cost of dengue epidemics in these countries was about US$ 1.8 billion, including only outpatient and hospital expenses, disregarding the costs of surveillance activities, vector control, and population mobilization [33].

The difficulty of mosquito control in Brazil is the non-uniformity of compliance with the guidelines of the dengue control program, zika and chikungunya in all municipalities, as well as the inability of epidemiological and entomological surveillance to eliminate all possible existing outbreaks (breeding sites) in all regions of all Brazilian cities.

2.4 Plans and actions to combat the vector

Dengue, zika, and chikungunya are vector-borne diseases, and due to the lack of vaccines and specific antiviral drugs, up to now, the prevention of these diseases are the control, elimination or eradication of the vector. The interventions aimed at vectorial control have been based on three main sets of mechanisms:

- Mechanical: consists of the adoption of practices capable of eliminating the vector and the breeding sites or reduce the contact of the mosquito with humans. Environmental management in current national control programs has generally been limited to the destruction of potential breeding sites in the domiciles and peridomiciles environments, without intervening in other elements of the urban infrastructure (garbage collection, water supply, etc.) and the way of the population lives [37].
- Biological: based on the use of predators or pathogens with the potential to reduce the vector population. Promising attempts have been made to use biological methods of larval control [37].
- Chemical: consists of the use of chemicals to kill larval or adult forms of the vector. Larvae are eliminated in their habitat (accumulated water) by the use of larvicides, while winged forms are eliminated by spraying the environment with pyrethroid or organophosphorus insecticides at "ultra-low volume" [37]. It is a type of control which the use must be done safely and rationally, complementing the actions of surveillance and environmental management. The irresponsible use may cause selecting vectors resistant to the products and environmental impacts [38].

An alarming fact is that in 2016, 3.31 million cases of dengue were registered on the planet and reported to the WHO, almost half of them in Brazil. This number was the highest in recent history. Taking into account that it does not include data from Africa, where the infrastructure to monitor the incidence of the disease is more precarious, the global number would be even higher. Therefore, the data show the potential need for a multi-scalar *Aedes aegypti* control strategy, both locally and globally. In order to combat the mosquito, the WHO presented a comprehensive strategy of vectorial control in 2004 [39]; it is a way of integrated management, in which Brazil is a member. The WHO strategy builds on the basic concept of integrated vector management with a renewed focus on improving governance capacity at the national and subnational levels. There is an emphasis on strengthening infrastructures and systems (*e.g.* sustainable development, access to potable water, adequate solid waste and excreta management), particularly for vulnerable areas [6]. As pointed by the WHO, for a sustainable impact on vector control, greater intersectoral and interdisciplinary action is needed, linking efforts in environmental management, health education, and reorienting relevant government programs around proactive strategies to control new and emerging threats [6]. Therefore, critical attention is adapting to local contexts; these information along with the global strategies make effective and sustainable the vector control.

Although several countries in the Americas have been considered "free" of the *Aedes aegypti* in the recent past, the discussion about the difficulties or even the impossibility of eradicating a biological species from a large geographical area remains alive. In 2001, the Brazilian government began to consider vector control instead of the eradication goal, with the implementation of the Dengue Control Actions Intensification Plan (PIACD), prioritizing actions in municipalities with the higher transmission of dengue. In 2002, the Brazilian National Plan for Dengue Control (PNCD) was developed due to the increased risk of epidemics, the occurrence of severe dengue cases and the reintroduction and rapid dissemination of serotype 3 in the country [40]. The PNCD also seeks to incorporate lessons from national and international dengue control experiences, emphasizing the need for change in previous models, *i.e.*, it is a reflection of the new global approach incorporated in WHO.

Furthermore, with the support of the Brazilian Ministry of Health and the states, the municipal health secretariats began to manage and implement PNCD actions. These actions involved ten main components: epidemiological surveillance, vector control, patient care, integration with basic care, environmental sanitation actions, integrated actions of health education, communication and social mobilization, training of human resources, legislation, political and social support, and monitoring and evaluation of PNCD [40]. Thus, the program was no longer exclusively directed at combating the vector and suggested adaptations consistent with local specificities, including the possibility of elaborating sub-regional plans.

In Brazil, the community, health and endemics combat agents, in partnership with the population, are responsible for promoting the control of the vector, whose actions are focused on detecting, destroying or properly allocating natural or artificial water reservoirs that may serve as a deposit for mosquitoes' eggs. Another complementary strategy advocated by the Brazilian Ministry of Health is the promotion of educational actions during the home visit by the community agents, with the objective of guaranteeing the sustainability of the elimination of the breeding sites, in an attempt to break the chain of transmission of the diseases [33].

Thereby, new technologies have been developed as alternatives to the control of the mosquito, using different mechanisms of actions, such as social measures, careful monitoring of infestation, dispersion of insecticides, new chemical and biological control agents and genetic procedures for control of the mosquito populations, including combinations of techniques. The adoption of vector control strategies combinations requires continuous evaluation of effectiveness [37], considering the possible synergistic effects between compatible strategies and spatial heterogeneity, based on an assessment of risk areas. Therefore, mapping techniques are also presented as a promising strategy, developed to evaluate and identify areas of increased risk for arbovirus transmission in certain territories using local spatial statistics. By linking spatial data with entomological surveillance data (characteristics, presence, infestation rates, and efficacy evaluation of control methods), epidemiological surveillance, laboratory network, and sanitation, specific vector control actions are directed to priority areas.

2.5 Conclusions

The *Aedes aegypti* is one of the primary biological vector responsible for transmitting a wide range of arboviruses. These infections are on the rise and extending to new geographical areas. As previously discussed, these viruses are causing a tremendous negative impact on the health of the Brazilian and global population. Thus, controlling the vector mosquitoes is critical for preventing these diseases. The control methods should be considered according to the regional context including chemical, biological and environmental means. Furthermore, the governmental effort should be intensified to come up with and stimulate more innovative ways of controlling and surveillance of the mosquito. In this work we propose a methodology to help in the combat of the *Aedes aegypti* by locating locals with potential mosquitoes breeding sites.

Chapter 3

Vision Meets Unmanned Vehicles

In this chapter, we briefly review some works related to our problem, with focus on those ones that detect mosquito breeding sites directly. Particularly, we also talk about those directly interested in detecting mosquito breeding sites. Lastly, we describe in details our new dataset.

3.1 Related works

In this section, we provide a brief review of related works. We first discuss techniques for detecting stagnant water which is the perfect environment for *Aedes aegypti* reproduction. Moreover, we highlight works that directly perform mosquito breeding grounds detection.

3.1.1 Stagnant water detection

Image-based water detection systems can be useful for many applications [41–47]. In our problem specifically, this approach could also be applied since *Aedes aegypti* reproduces in stagnant water.

The method in [41] uses a water detection system to assist in the navigation of autonomous off-road vehicles. The system is based on several types of features, including texture and others using the HSV (hue, saturation and value) color space. It also uses a pair of cameras to separate regions of the image that are candidates to be as reflective as water (regions whose depth is estimated as greater than the depth of the neighborhood). It also presents a fusion rule to combine these features and segment the region of interest containing water.

In [42] one finds a methodology to asses the performance of water detectors used in images from unmanned vehicles. Two types of evaluations are presented: one that considers the intersection between the region of the detector outputs and the ground truth, and another one that assess the georeferencing accuracy. The authors in [43] observed that, to detect water bodies, the reflections of the sky are more useful in images whose objects are distant from the camera while colors have more discriminative power in the shorter range.

The authors of [44] propose a shape descriptor in images which is invariant to scale, rotation, affine transformations, mirroring and non-rigid distortions such as ripple effects. This descriptor is quantitatively and qualitatively compared with other methods, achieving better results.

The work on [45] performs water detection based on sky reflection and has been developed to be applied to unmanned land vehicles. In that work, it is considered that water bodies act as flat mirrors for large incidence angles so that the proposed method seeks to geometrically locate the pixels of the sky that are reflected in a water body candidate. Based on the color similarity and the local characteristics of the terrain, it is decided whether the candidate is water or not, given that it is below the horizon line. Tests performed in open rural areas with distances greater than 7 meters obtained 100% true positives and a maximum of 0.58% false positives for different climatic conditions.

Another way to detect water in videos is through the dynamic texture segmentation [46]. In that work, the proposed technique removes the static background image and even dynamic objects present in the scene. To do so, an entropy measure computed through optical flow in the course of several frames to obtain the water signature is used. In order to detect regions without motion, an image segmentation method based on the propagation of labels is applied. The technique was validated in 12 videos with static and moving camera obtaining 95% of true positives and 10% of false positives.

A new water-reflection recognition technique is presented in [47]. First, they construct a new feature space using moments invariant to motion distortion in low- and high-frequency curvelet space. In this new space of characteristics, they apply the algorithms to minimize the cost of reflection in low frequencies together with discrimination of *curvelets* coefficients in high frequencies. By doing so, they classify the water reflection and detect the reflection axis in the images with a hit rate ranging from 80% to 95%.

3.1.2 Mosquito breeding grounds

Regarding mosquito breeding grounds detection we point out references [48–50]. In [48] a system receives geotagged images generated by the population. Then, the image quality is evaluated in order to reject images that contain high levels of distortions or artifacts. Next, each image is converted into a feature vector using the bag of visual words model through the scale-invariant feature transform (SIFT) descriptor. Afterwards, a support vector machine (SVM) classifier is trained to identify whether the images contain potential mosquitoes breeding sites (stagnant water, open tyres or containers, bushes, etc) or not (running water, manicured lawns, tyres attached to vehicles, etc). Finally, the system outputs a heat map where the regions with the highest risk of having mosquitoes habitats are indicated.

The authors of [49] use a trained SVM classifier to detect water puddles in images obtained from videos that can have stagnant water, acquired by a quadcopter. The work in [50] use thermal and gray level images to detect stagnant water. From each image, they compute a 128-bit vector using speeded-up robust features (SURF) descriptor. This vector is then reduced to a 64-bit vector using principal component analysis (PCA). The computed vectors are used to train an ensemble of naive Bayes classifiers to identify potential mosquitoes breeding grounds.

In this work, we propose a methodology for mosquitoes breeding grounds detection that applies machine learning techniques using videos acquired by an Unmanned Aerial Vehicle (UAV), also known as a drone. To our knowledge, there is not a big public dataset to train a model for this task. However, the dataset proposed in [8] is interesting but with some drawbacks, which we describe in Section 3.2. They train a Random Forest classifier using features (H channel from HSV color space for detecting tires and histograms from S and V channels from the same space for detecting stagnant water) extracted from the images. Taking [8] as inspiration, we propose to design and acquire a new dataset, described in Section 3.3.

3.2 The CEFET dataset

The authors of [8] use the commercial UAV DJI Phantom Vision 2 Plus to capture several videos containing tires, puddles and some other objects with water, like water tanks and pails at different simulated environmental situations. This UAV model has 20 minutes of flight autonomy and is equipped with a camera capable of recording videos up to 1080×720 pixels resolution at 60 Hz. They chose to record the videos using 1080p at 30 frames per second (fps).

This UAV camera model presents high distortions, specially the radial one. In order to mitigate this problem, they reduced the field of view (FoV) to 85 degrees. By doing so, the flight time increases since the drone would take more time to cover a region.

All the videos, with about 15 s duration, were recorded with the camera at downward position with the drone following a specified route at approximately constant 5 m altitude. The speed is set to be 7 km/h. They point out that the parameters may vary over the course due environmental interference such as wind.

The drone telemetry measurements (*e.g.* position, velocity and altitude) are extracted using the third-party application Litchi [51]. For each video, there is a text file containing the frame-by-frame of objects annotations. Another limitation of this dataset is that it only contains tires annotated.

Their dataset, including the 29 videos in `mp4` format and corresponding text annotation files, is publicly available[1]. We show some examples of scenarios of this dataset in Figure 3.1 The train-test split is included in the annotation files. However, in this work, we propose a different split that we mention in Chapter 5.

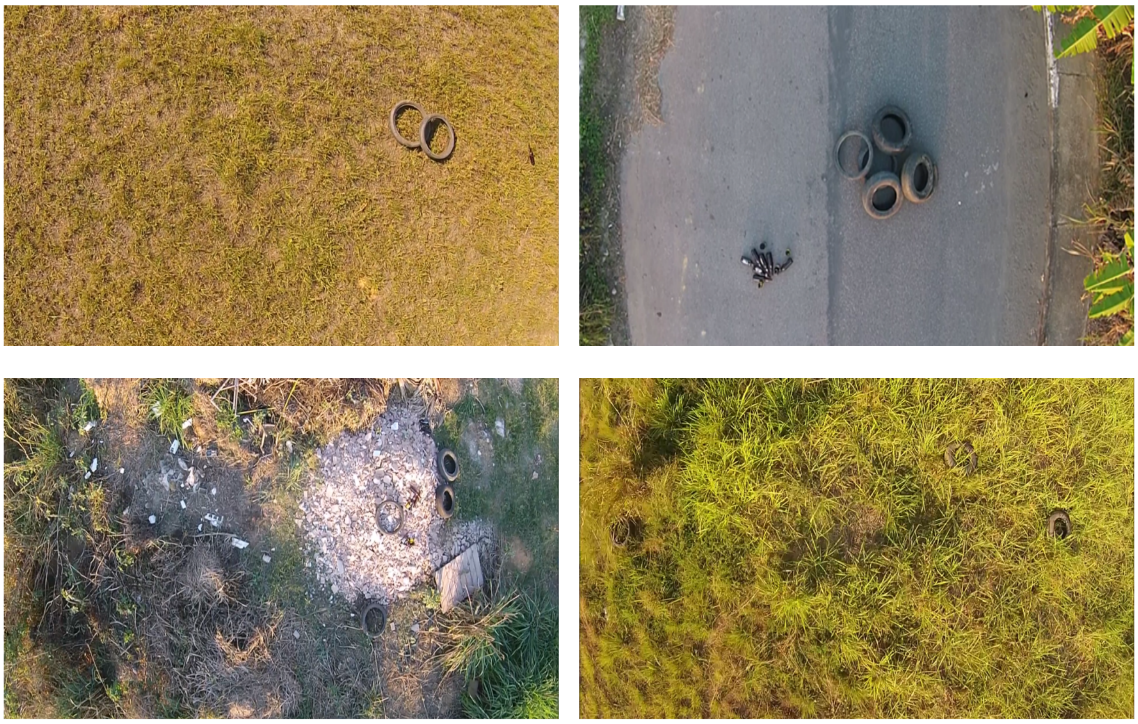


Figure 3.1: Examples of scenarios contemplated by the CEFET dataset.

3.3 The Mosquito Breeding Grounds dataset

In this section we present the description of our dataset named as "Mosquito Breeding Grounds" (MBG). We use the commercial Phantom Vision 4 PRO UAV from DJI Company for acquiring the aerial videos that might depict potential mosquito breeding sites. This UAV has approximately 30 minutes of flight autonomy and is capable of executing a predefined flight plan. One may retrieve all telemetry information (*e.g.* altitude, latitude, longitude, speed, etc) from a `.csv` file using a 3rd party software. It is equipped with a high-definition camera with passive and active stabilization (dampers and gimbal). The camera has many parameters that can be adjusted and can generate videos up to 4096 p at 60 Hz. Since the telemetry sampling rate is higher than the camera frame rate, we employ a decimation in the

[1]`https://drive.google.com/open?id=1tDOVdb_vALUnD_cY31Qf0ggoiM1F63Jl`

signals obtained from the `.csv` file in order to associate each video frame to its respective telemetry measurement.

The dataset has the following technical specifications:

- We automatically predefine a flight plan using the Litchi [51] software. This plane performs a serpentine-like sweep over the entire terrain area autonomously.

- We turn off the camera auto adjustment and set all the parameters manually, keeping the focus fixed at infinity and setting the video scan to 3840 p at 50 frames per second (fps). This step is important because we want to keep all camera parameters constant in order to perform camera calibration.

- Before starting a flight, we record a calibration video using a calibration pattern, as described in Section 3.3.1.

- The altitude is approximately constant in each video. Currently, the dataset has videos acquired at different altitudes, *e.g.* 10, 25, 40 m, all of them predefined in the flight plan through Litchi. Small variations in altitude (± 0.5m, according to the manufacturer [52]) caused by the limited accuracy of telemetry, wind, etc are within acceptable ranges.

- Speed approximately constant of 15 km/h is preset via Litchi. This parameters can suffer small variations caused by wind, for example. However, our drone has a 10m/s maximum wind speed resistance [52].

- The dataset includes different types of terrains. Currently, it contains high and low grass, asphalt, wasteland and buildings, as depicted in Figure 3.2.

- The videos have about 15 objects manually inserted objects, randomly arranged on the recording area, including tires, bottles, and other objects that can accumulate water such as buckets and plastic pools. The videos also have objects that originally were part of the recorded scenes such as water tanks. We show examples of these objects in Figure 3.3.

- Afterward, we compensate the videos' distortions, as described in Section 3.3.1 and manually annotate them using the Zframer software, according to Section 3.5.

The generated dataset contains several sequences of aerial videos containing tires, water puddles, water reservoirs and several other objects filled with water. Currently, the videos were recorded at the Technology Center of UFRJ, *campus* Ilha do Fundão, and at the CEFET/RJ *campus* Nova Iguaçu.

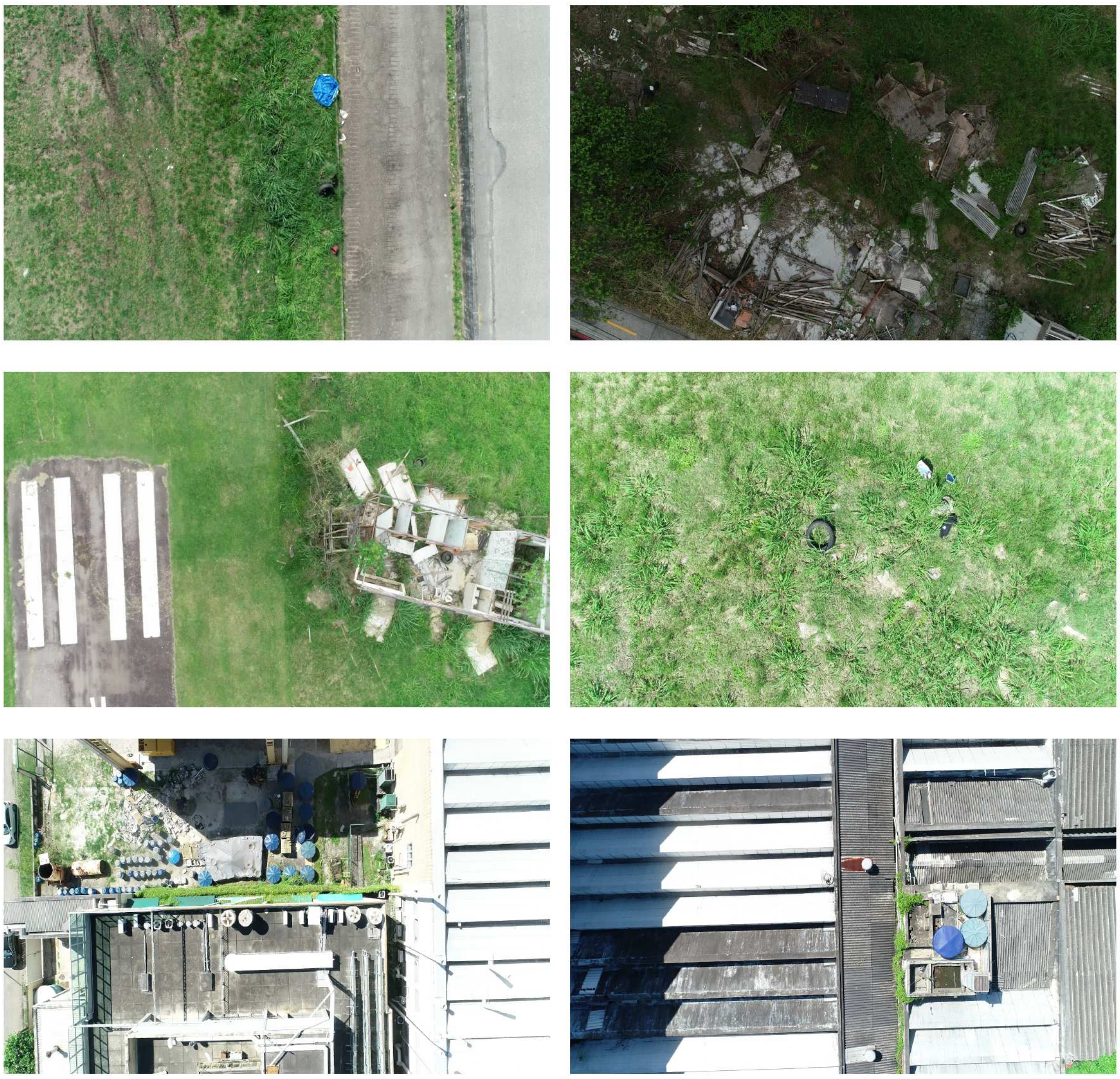


Figure 3.2: Examples of scenarios contemplated by the MBG dataset.

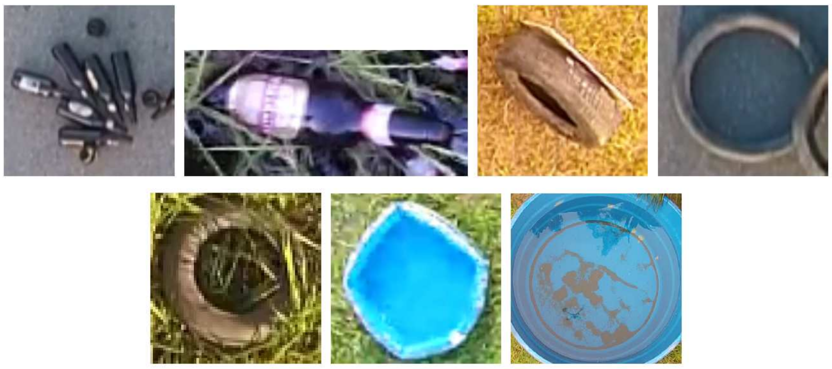


Figure 3.3: Examples of objects in the video dataset.

3.3.1 Camera calibration

Camera calibration is an important process for computer vision applications. This step is critical because three-dimensional (3D) metric information may be extracted from an image if calibration is well performed. Through camera calibration one may find the intrinsic camera parameters, such as focal length and principal point. Many calibration techniques have been developed [53, 54], even for photometry applications [55, 56].

In this work, we apply camera calibration for minimizing the distortions caused by the camera lens, mainly the radial distortions. We have chosen the method proposed in [57] due to its simplicity and low cost. This technique consists in extracting the key points (corners) from an image containing a calibration pattern, usually a chessboard; estimating the camera parameters; estimating the distortion coefficients; and applying the correction.

Keypoints detection

Harris [58] and SIFT [59] are classical detectors of image key points. However, in this work, we use the method implemented by the OpenCV's `findChessboardCorners` function [60], for being more robust in cluttered images containing smoothing artifacts. Initially proposed by Vladimir Vezhnevets [61], this method consists of first converting the image to a binary image based on an adaptive threshold, segmenting the white and black squares of the calibration pattern. Then the borders of black squares are found and these contours are approximated to quadrilaterals, whose corners are selected and grouped in, according to the calibration object.

In this work, we record calibration videos showing the calibration pattern which has a 10×7 checkerboard pattern. Key points detection is performed at every 20 frames of the video. Before performing detection, we filter the image with a Gaussian filter of size 7×7 pixels, 0 mean and standard deviation 1.4 in both directions. We lower the image resolution by 40% to reduce detection time.

Camera model

Having detected the key points in the calibration pattern, one may estimate the camera projection matrix from the coordinates of these points in the real world and the image. Therefore, we consider that a camera maps a world point $\mathbf{M}' = [X, Y, Z, 1]^{\mathrm{T}}$ to an image point $\mathbf{m}' = [u, v, 1]^{\mathrm{T}}$ through a projective transformation of the form [62]

$$s\mathbf{m}' = \mathsf{A}[\mathsf{R} \mid \mathbf{t}]\mathbf{M}', \tag{3.1}$$

where s is an arbitrary scale factor, R and $\mathbf{t}$ the rotation matrix and translation vector, respectively, (extrinsic parameters) that relates the world coordinate system to the camera coordinate system. The matrix A is the calibration camera matrix (intrinsic parameters), defined as

$$\mathsf{A} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.2}$$

where $[u_0, v_0]^{\mathrm{T}}$ denotes the principal point coordinates, α and β the scale factors of u and v image axis, respectively, and γ is the skewness of the two image axis. The intrinsic parameters does not depend on the image viewed. As a result, once computed, they may be used for all images since the focal length is fixed (same zoom level).

Distortion compensation

Conventional cameras usually have significant lens distortion, specifically radial distortion. In order to minimize such distortions, we apply Zhang's algorithm [63] as follows. Let (u, v) be the ideal (nonobservable distortion-free) pixel image coordinates and $(\breve{u}, \breve{v})$ the corresponding coordinates at observed (distorted) image. The ideal points are projections of calibration pattern points according to the model given by Equation (3.1). Likewise, (x, y) and $(\breve{x}, \breve{y})$ are, respectively, the ideal and real normalized image coordinates. Hence, the radial distortion may be modeled as [63]:

$$\begin{cases} \breve{x} = x + x(k_1 r^2 + k_2 r^4) \\ \breve{y} = y + y(k_1 r^2 + k_2 r^4) \end{cases}, \tag{3.3}$$

where k_1 and k_2 are radial distortions coefficients and $r^2 = (x^2 + y^2)$. Higher order distortion models do not present significant improvements and may lead to numerical instability [63]. The center of the radial distortion is located at the principal point. From $\breve{u} = \alpha\breve{x} + \gamma\breve{y} + u_0$ and $\breve{v} = \beta\breve{y} + v_0$, assuming $\gamma = 0$, we may write

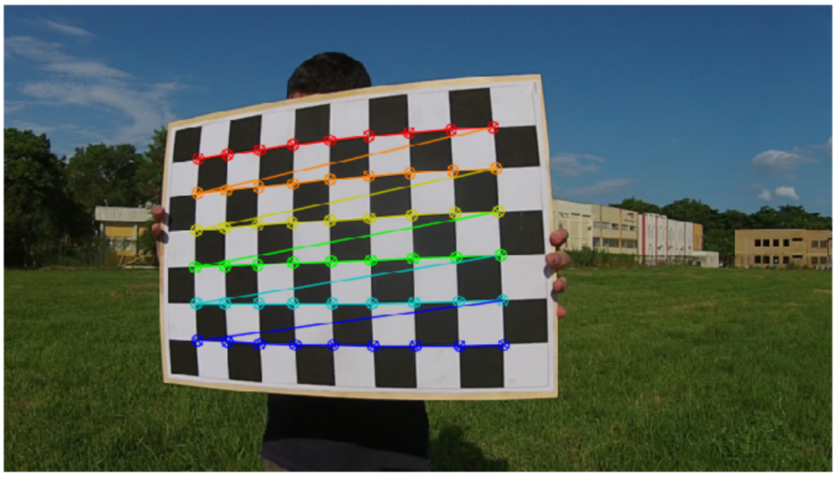
$$\begin{cases} \breve{u} = u + (u - u_0)(k_1 r^2 + k_2 r^4) \\ \breve{v} = v + (v - v_0)(k_1 r^2 + k_2 r^4) \end{cases}. \tag{3.4}$$

Zhang's algorithm [57] first makes a coarse estimation of the camera extrinsic and intrinsic parameters and refine them through maximum likelihood estimation. Given n calibration pattern images, considering we have m points in this pattern, and the image points are corrupted by independent and identically distributed (i.i.d.) noise, the maximum likelihood estimation may be obtained by minimizing the following function [63]

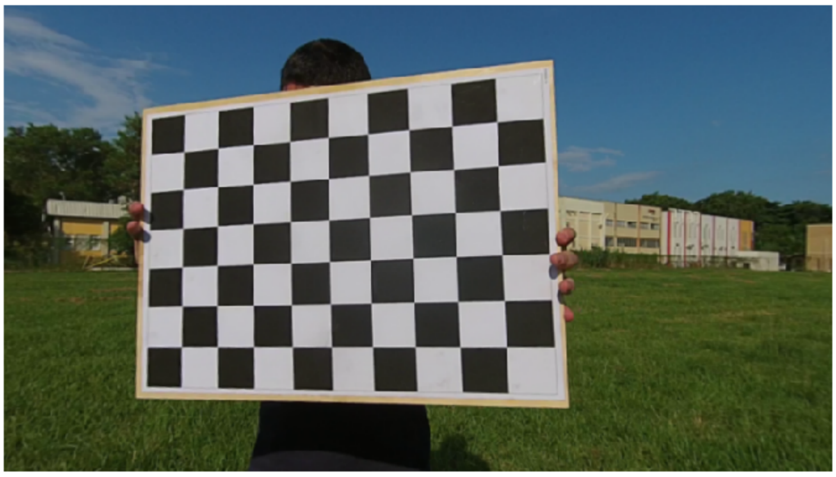
$$\sum_{i=1}^{m}\sum_{j=1}^{n} \|\mathbf{x}_{ij} - \breve{\mathbf{x}}(\mathsf{A}, k_1, k_2, \mathsf{R}_i, \mathbf{t}_i, \mathbf{X}_j)\|^2, \tag{3.5}$$

where $\breve{\mathbf{x}}(\mathsf{A}, k_1, k_2, \mathsf{R}_i, \mathbf{t}_i, \mathbf{X}_j)$ is the projection of point $\mathbf{X}_j$ in image i, according to Equation (3.1), following the distortion model in Equations (3.4). The minimization of the functional given in Equation (3.5), is an nonlinear optimization problem that may be solved with the Levenberg-Marquardt algorithm.

The Figure 3.4a shows an example of the original (distorted) image with detected calibration pattern corners overlaid, and Figure 3.4b shows the corresponding image after applying Zhang's algorithm to undistort the image. It is interesting to note that the "barrel" effect was eliminated of the image, which is clearer from the observation of the chessboard calibration pattern.



(a) Original (distorted) image with detected calibration pattern corners overlaid.



(b) Undistorted image.

Figure 3.4: Example of image undistortion using Zhang's algorithm.

3.4 Generating rectified videos

We apply rectification frame by frame *i.e.*, we need to extract all video frames and apply the rectification transform in each one. To verify how compression compromises the frames quality, we made a brief study that we describe in the sequel.

First, we extract all frames from the original, not rectified videos. We save them as new videos using the same codec (X264) of the original videos, through `imageio` python library [64] (a `FFMPEG` wrapper). `FFMPEG` is a software that can save, convert and create video and audio streams in various formats [65]. Lastly, we compare the videos both objectively and subjectively.

We choose the constant rate factor (CRF) mode that varies from 0 to 51 (using 8 bits), where 0 is lossless and 51 is the worst quality possible. A lower value leads to higher quality, and a subjectively sane range tends to be 17–28 [65]. By using this mode we want to keep the best quality without caring much about the final file size. In this brief experiment, we use a "quality" scale [64] in which the conversion to CRF is done as follows:

$$\text{CRF} = 51\left(1 - \frac{\text{quality}}{10}\right). \tag{3.6}$$

We evaluate the quality varying from 0 to 10 with unity step, according to Table 3.1.

Table 3.1: Quality indexes evaluated and respective CRF values [65].

Quality	0	1	2	3	4	5	6	7	8	9	10
CRF	51	45.9	40.8	35.7	30.6	25.5	20.4	15.3	10.2	5.1	0

To evaluate our results quantitatively, we use the peak signal-to-noise ratio (PSNR), defined as

$$\text{PSNR} = 10\log_{10}\left(\frac{2^B - 1}{\text{MSE}}\right), \tag{3.7}$$

where, B is the number of bits used to represent a pixel (in our case, $B = 8$) and MSE is the mean squared error between original frame I and compressed frame K of same size $m \times n$, defined as,

$$\text{MSE} = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\Big(I(i,j) - K(i,j)\Big)^2. \tag{3.8}$$

Good-quality compressed images present typical values of PSNR between 30 and 50 dB, provided the bit depth is 8 bits, where higher is better [66].

For each video and each quality level, we compare the frame by frame PSNR extracted from original and compressed video and we take their mean. We run this experiment for 4 videos; Video 1 is calibration video and the other 3 are videos recorded on grass area (we choose these last 3 since grass is a texture that is hard to compress). We show the average PSNR values along with the final video compressed size in Figure 3.5. Note that as we increase quality, the file size increases exponentially, but, as video encoders are essentially lossy, the PSNRs reach a saturation level [65].

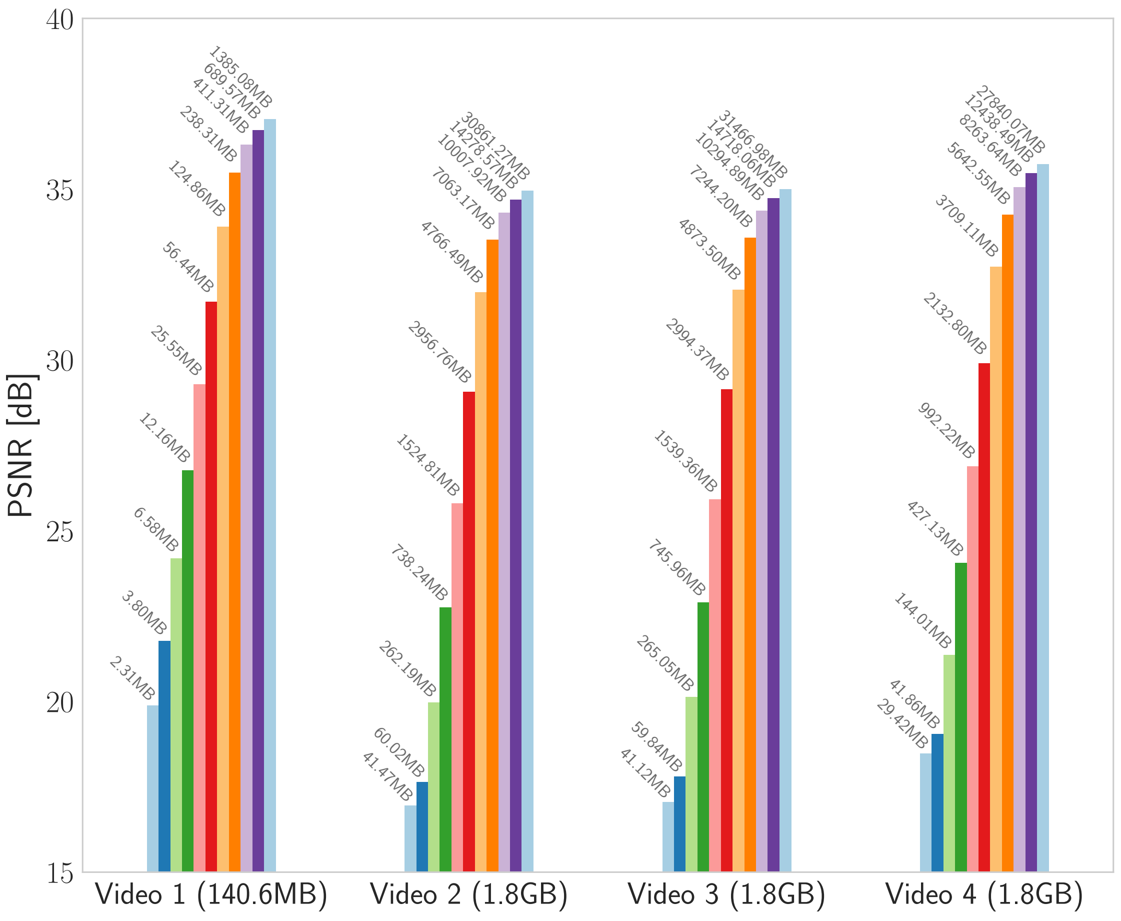


Figure 3.5: Video compression and file size comparison between original and frames after compression. The leftmost bin represents quality= 0 and rightmost quality= 10.

We also visually evaluate the videos generated. As shown in Figure 3.6, at zero quality the image is seriously damaged, as expected. We chose the quality parameter that gives the best balance between quality and file size. The quantitative and qualitative results have shown the quality was not significantly compromised when we use quality factor around 5 (CRF = 25.5). Therefore, we use this factor when storing the new videos with the rectified frames.

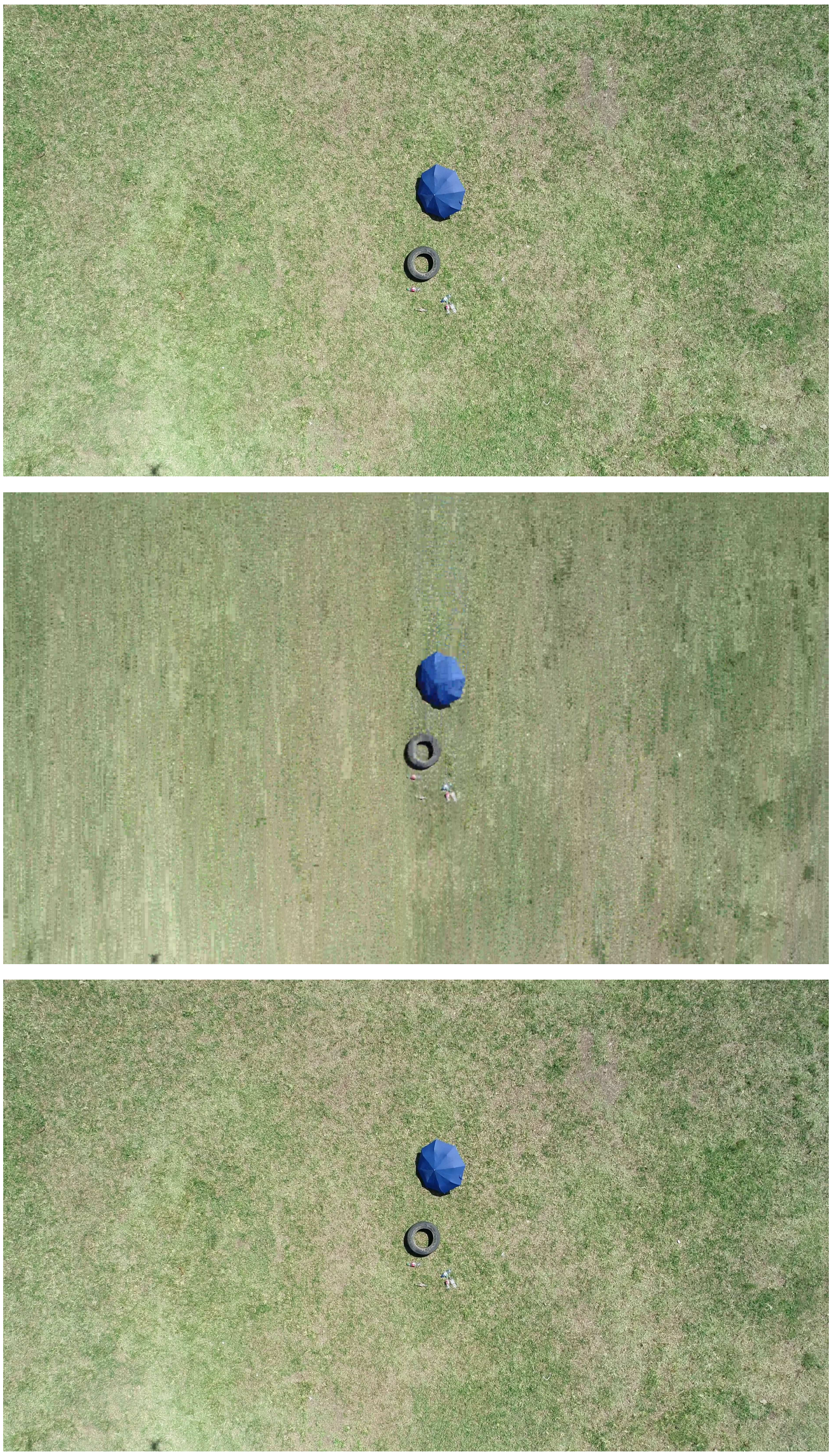
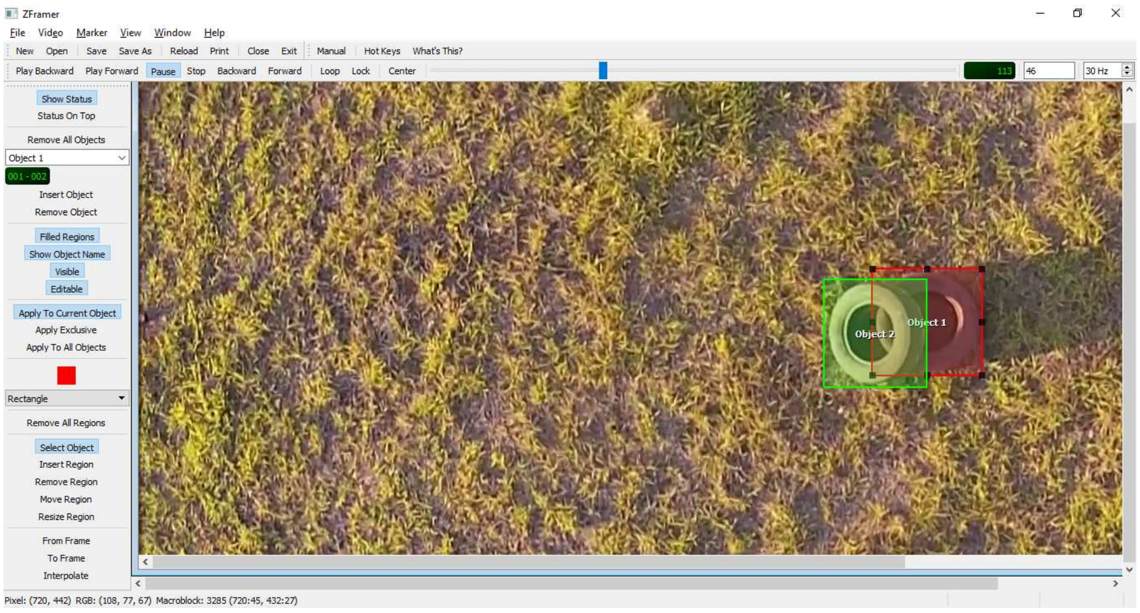


Figure 3.6: Compression quality comparison. From up to down: frame from original video, quality = 0, and quality = 5.

3.5 Database annotation

Our dataset is under manual annotation process and is being performed using the Zframer software[2], developed at the Signals, Multimedia, and Telecommunications (SMT) Laboratory of Coppe/UFRJ. After rectified, the acquired video sequences are labeled frame by frame with Zframer, as depicted in Figure 3.7. The students from signal processing laboratory at CEFET/RJ *Campus* Nova Iguaçu has helped us in this labeling process. Using the Zframer one may annotate, in each frame of the videos, the objects that have been determined as potential mosquitoes breeding grounds (*e.g.* tires, water reservoirs, bottles). Moreover, the software allows interpolation between annotations of selected frames, so that it is not necessary that all frames where the object appears are annotated. The software output is a text file containing, in each line, the annotation format (in our case, rectangles), and the frame number along with the pixel coordinates of the of the upper-left and bottom-right corners of the bounding box of each annotated object, as shown in Figure 3.8.



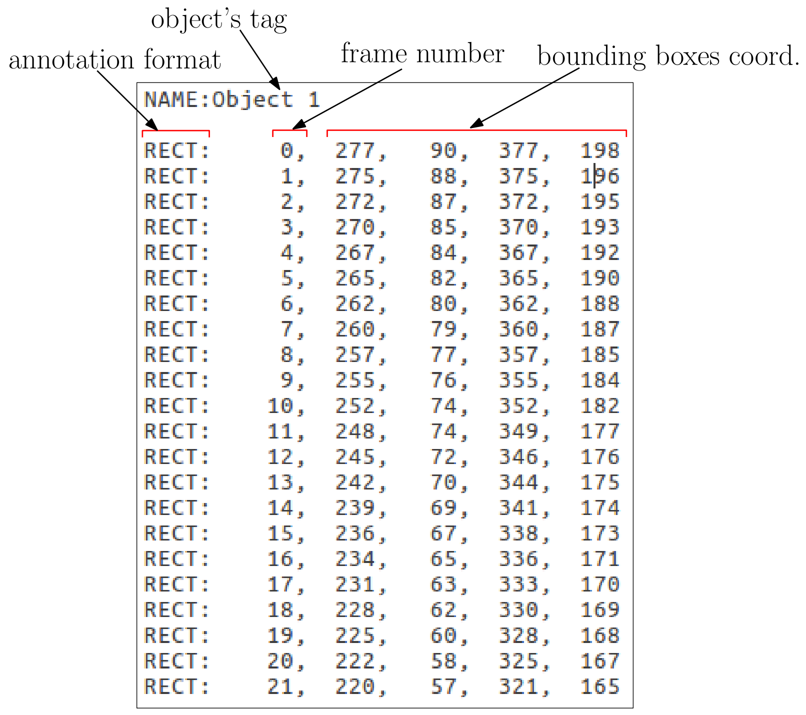
Pixel: (720, 442) RGB: (108, 77, 67) Macroblock: 3285 (720:45, 432:27)

Figure 3.7: A video frame annotation using Zframer.

3.6 Conclusions

In this chapter, we discussed works that are direct and indirectly related to our problem. We have seen that many of them are concerned about helping develop intelligent systems. In particular, we have gone through those that are of interest

[2]`http://www.smt.ufrj.br/~tvdigital/Software/zframer`



```
NAME:Object 1

RECT:     0,  277,   90,  377,  198
RECT:     1,  275,   88,  375,  196
RECT:     2,  272,   87,  372,  195
RECT:     3,  270,   85,  370,  193
RECT:     4,  267,   84,  367,  192
RECT:     5,  265,   82,  365,  190
RECT:     6,  262,   80,  362,  188
RECT:     7,  260,   79,  360,  187
RECT:     8,  257,   77,  357,  185
RECT:     9,  255,   76,  355,  184
RECT:    10,  252,   74,  352,  182
RECT:    11,  248,   74,  349,  177
RECT:    12,  245,   72,  346,  176
RECT:    13,  242,   70,  344,  175
RECT:    14,  239,   69,  341,  174
RECT:    15,  236,   67,  338,  173
RECT:    16,  234,   65,  336,  171
RECT:    17,  231,   63,  333,  170
RECT:    18,  228,   62,  330,  169
RECT:    19,  225,   60,  328,  168
RECT:    20,  222,   58,  325,  167
RECT:    21,  220,   57,  321,  165
```

Figure 3.8: Example of a text file generated by the annotation process.

to detecting mosquitoes foci. However, since none of them provided a consolidated dataset available, we proposed a new one, and described its construction.

Chapter 4

Object Detection with Deep Learning

In this chapter, we discuss object detectors and make a brief comparison among them. Then, we go deeper into the method used to detect the mosquito breeding sites: the region-based convolutional neural network (RCNN), particularly the Faster R-CNN. By the end of this chapter we hope the reader to have insights in object detection methods and understand how the region-based detectors work and how they evolved over time.

4.1 Introduction

Applications such as face recognition [67–69], self-driving cars [70], smart video surveillance [71], among others, have been attracting lots of attention of the computer vision (CV) community. These and other applications require systems that are capable of recognizing, classifying and localizing objects in image or videos.

Image classification, object localization, and object detection are fundamental and challenging problems in CV. In image classification, an algorithm assigns one (or more) label(s) (from a fixed predefined set of categories or classes) to an input image. Image classification has a wide variety of practical applications such as face recognition or even cancer diagnostic. In object localization, we not only want to know what object is in the image but also where it appears on it. An algorithm assigns a class to the "main" object (one image may contain multiple objects) as well as indicates the position of the object in the image by, as an example, drawing a bounding box around it. At last, object detection is the process of finding multiple instances of objects in images or videos instead of the "main" one only. One may interpret an object detector as a function $f : I \rightarrow \{k,\, p,\, b\}$ *i.e.*, an image I receives labels k from a set of predefined class labels, confidence scores p and bounding boxes b. The bounding box $b = \{x, y, w, h\}$ corresponds to a detection at a position (x, y), width w and height h. Typically, object detectors use features and learning algorithms to detect object instances.

4.1.1 Classical object detection

Classical object detectors use sliding windows to densely extract patches from the input image. These patches are warped to a fixed length (since many classifiers take fixed-size images only) and features are computed using, for example, scale-invariant feature transform (SIFT) [59] or histogram of oriented gradients (HOG) [72]. The classification, using methods like support vector machines (SVMs) [73], is performed in the feature space. This approach is (i) computationally expensive because features are computed for every image crop (and these crops highly overlap); and (ii) inaccurate, since sliding windows may not match the object size due to uncontrolled changes in scale, requiring multiple resolution windows.

4.1.2 Deep learning for object detection

State-of-the-art object detectors [74–76] are deep-learning-based [77]. They employ a type of deep neural network specially developed for CV applications: convolutional neural networks, also known as ConvNets or CNNs [78]. The success of deep learning models is due to: (i) the availability of a large amount of data [79, 80]; (ii) the increase in the available computing power, brought particularly by the development of powerful graphics processing units (GPUs) (along with GPU-accelerated libraries); (iii) the development of several powerful optimization methods (*e.g.* backpropagation; weight initialization; stochastic optimization; regularization; and new activation functions).

The deep-learning-based object detectors may be divided into two groups: single-shot (or one-stage) and region-based (or two-stage) detectors.

Region-based object detectors

Region-based or two-stage detector, as the name suggests, performs the detection in two steps. First, it generates a sparse set of region proposals in the image where the objects are supposed to be. The second stage classifies each proposal into one of the foreground classes or background and, in case it outputs an object label, it refines its position. The region-based convolutional neural network (R-CNN) [81] steered object detection to a new era: by employing ConvNets in the second stage, it achieved significant gains in accuracy. R-CNN evolved over time in terms of speed and accuracy [76, 82]. In Faster R-CNN [76] a ConvNet generates the region proposals, thus turning the whole system into a single convolutional network. Other works extend this framework [83, 84].

Single-shot object detectors

The single-shot detectors focus on speed rather than accuracy, aiming to predict both class and bounding box simultaneously. The single-shot detectors are inspired by the sliding-window paradigm. They split the images into a grid of cells so that we have sparse regions. For each cell, it makes bounding boxes guesses of different scales and aspect ratios. Different from traditional sliding-window detectors, it further refines the bounding box prediction instead of simply using the window position. OverFeat [74] was one of the first single-shot detectors followed for most recent YOLO [75] and SSD [85].

Object Detectors Comparison

Comparing object detectors is not an easy task, since we are not always capable of saying which one is the best model. Two aspects must be taken into consideration when choosing a model: accuracy and speed. Many attributes may impact the performance, to point some:

- feature extractors;
- input image resolution;
- hyper-parameters such as batch size, input image resize, learning rate, and weight decay.

In [86], a detailed comparison among single-shot and region based detectors is done. Single-shot detectors are faster than region-based detectors but they cannot beat region-based detectors in accuracy. Nevertheless, if we reduce the number of proposals in Faster R-CNN, for example, we are able to match the speed of SSD without harming its accuracy significantly.

Since speed is not the main concern of our application, we choose Faster R-CNN as our object detector as it achieves most accurate results [86].

4.2 Region-based Convolutional Network (R-CNN)

Before explaining the Faster R-CNN, we shall see its older, rougher around the edges grandfather: the region-based convolutional network (R-CNN) [81] , followed by the middle child, the Fast R-CNN and then finally the Faster R-CNN.

The R-CNN [81] consists of three parts: (i) a region proposal method that generates class-agnostic regions of interest (RoIs). These regions form the set of candidates available to detection. They are warped into a fixed size and then fed into the next module, individually; (ii) a CNN that extract features from each RoI; (iii) a set of class-specific fully connected (FC) layers to classify each region and refine the corresponding bounding box. Figure 4.1 illustrates the R-CNN flow.

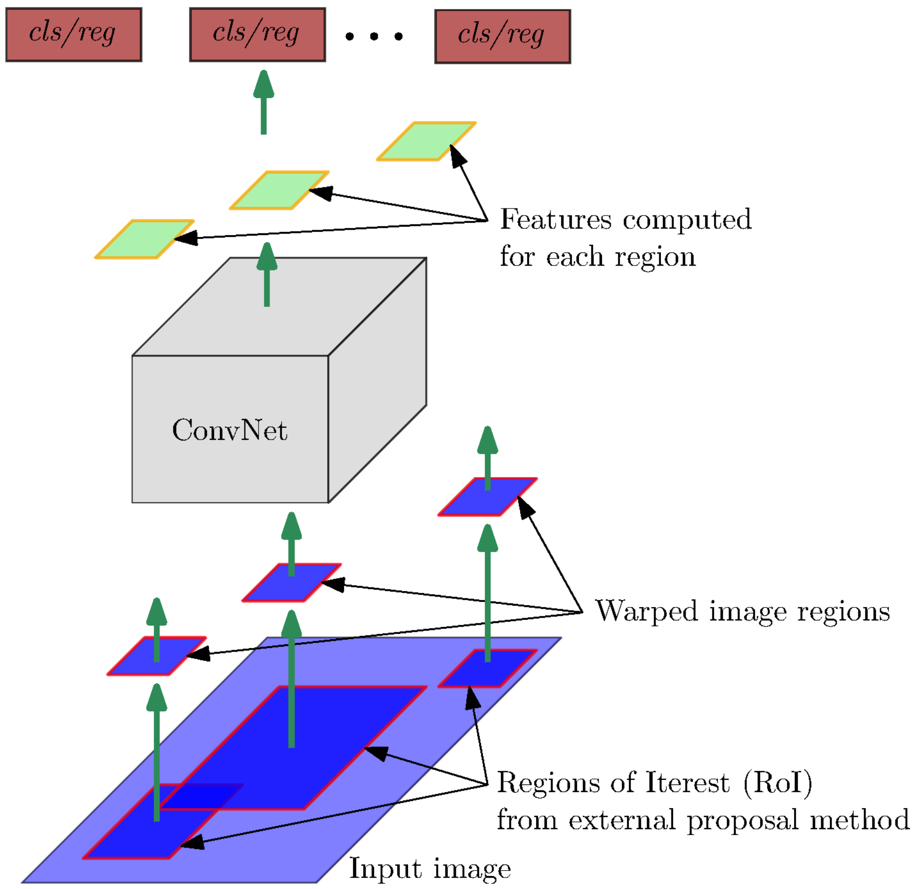


Figure 4.1: R-CNN.

4.2.1 Region proposals

Instead of classifying a huge number of regions, as sliding-window detectors do, R-CNN uses selective search [87] to generate $2,000$ RoIs from the image [81].

The selective search algorithm works by clustering pixels using a similarity measure. First, each pixel of the image is considered as an individual group. Next, it computes a similarity measure (*e.g.* image texture) and combine closest groups into larger ones. It continues merging regions until achieving the number of regions desired. Figure 4.2 shows an example of region proposals generated by the selective search algorithm over an image. The first row is the combined regions, the blue and green boxes in the second row are the possible RoI and detected objects, respectively.

4.2.2 Feature extraction

The features are extracted by forwarding each proposal through a convolutional network. Each proposal is warped to fit the ConvNet's input size, regardless of its size or aspect ratio. Prior to warping, the region is enlarged a bit to include pixels of image context in the warped region [88].

4.2.3 Object classifier and box regression

The region proposals that have an Intersection over Union (IoU – for a more detailed description refer to Section 5.1) with the ground truth smaller than 0.3 are defined as negative samples. Once we have the features and the training sample label, we train a linear SVM per class. After the SVM stage, a class-specific regressor is used to predict a new bounding box for detection [88].

4.2.4 R-CNN drawbacks

Although R-CNN achieved satisfactory results in object detection tasks [81], its authors point some drawbacks in a later work [82]:

- Training is a multi-stage pipeline: first, perform a fine-tuning on a CNN using the RoIs generated by the region proposal method. Then, for each class, fit an SVM. At last, it learns the bounding box regressors.

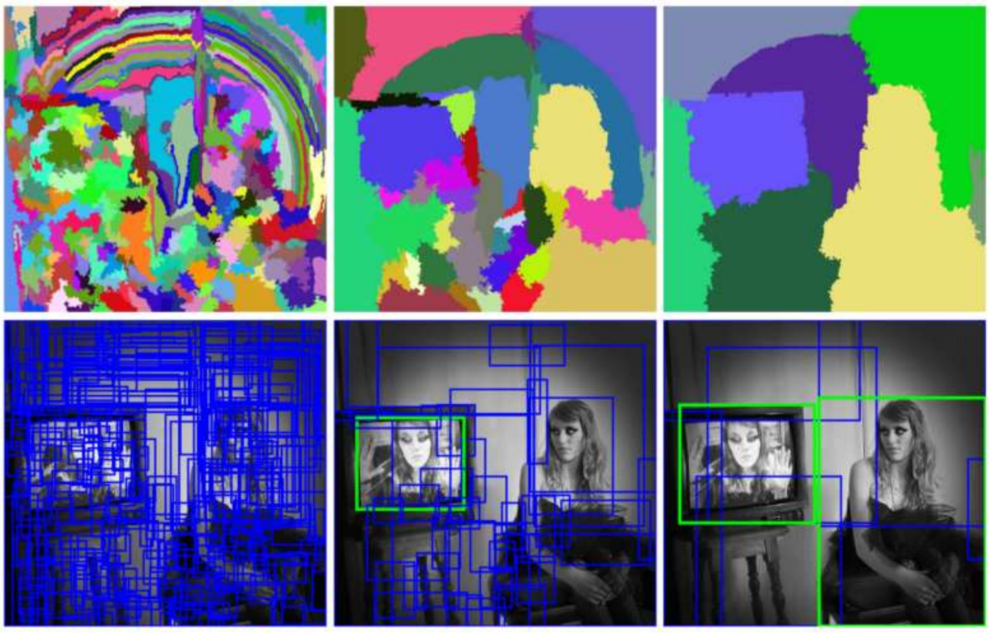


Figure 4.2: Selective search example. The first row are the combined regions, the blue and green boxes in the second row are the possible RoI and detected objects, respectively. Source: [87].

- Training is expensive both in space and time: although selective search reduces the number of RoIs to be analyzed, it stills needed a high number of them to achieve good performance ($\sim 2,000$ for each image). From each RoI in each image, R-CNN extracts the features that are written to disk. The author reports that this process has taken 2.5 GPU-days (Nvidia K40 GPU overclocked to 875 MHz) [82], with VGG-16 [89], for 5,000 images of a specific image set.

- Slow object detection: at test-phase, R-CNN extracts features from each RoI for each image test. For each test image, with VGG-16 [89], detection took about $47s$ on a GPU. This is slow because there are no shared computations during the ConvNet's forward pass.

4.3 Fast R-CNN

In a later work [82], Fast R-CNN authors, solved of some R-CNN's [81] drawbacks in order to build a faster algorithm. Differently from the original work, in which we feed the ConvNet with every single region proposal, the whole image is fed into a ConvNet to produce a single convolutional feature map. Fast R-CNN still uses an external region proposal method (*e.g.* selective search [87]) to generate the proposals. Thus, the input of Fast R-CNN is an image and a set of proposals [82].

The RoIs are the rectangular regions of the feature map bounded by the proposals, each defined by a four-tuple (x, y, w, h) specifying its top-left corner (x, y) and its width and height (w, h) [82]. These RoIs along with the feature maps, extracted from the entire image, form the patches used for object detection. The patches are warped to a fixed-length feature vector, by the using RoI pooling layer [82] (Section 4.3.1), so that they may be fed into a sequence of FC layers. The FC layers branch into two sibling output layers:

1. classification layer: a softmax layer that estimate the class probability of the RoI over K object classes plus a "background" class.

2. localization (regression) layer: outputs a set of 4 real values for each one of the K object classes. These values encode a refined bounding box position, *i.e.*, an offset for the initial proposal.

By extracting the features over the whole image, instead of repeating the feature extraction for each proposal every time, Fast R-CNN reduces the cost time significantly, trains $9\times$ faster than R-CNN (with VGG-16), and takes $\sim 0.3s$ to run detection (not considering object proposal time) *vs.* $47s$ of R-CNN [82]. Another improvement in Fast R-CNN is that one may train the entire network (ConvNet, and softmax and regression layers) end-to-end with the multi-task loss (classification and localization losses, Section 4.3.2), which improves the detection accuracy [82]. Figure 4.3 illustrates the Fast R-CNN flow.

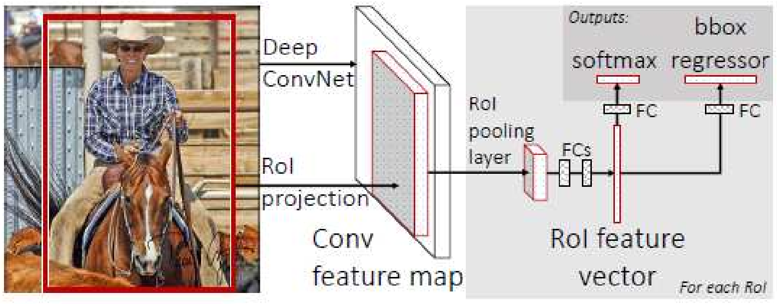


Figure 4.3: Fast R-CNN. Source: [82].

4.3.1 RoI pooling layer

Fast R-CNN uses FC layers for classification and bounding box regression tasks [82]. These layers require an input of predefined size. Since the RoIs generated by region proposal method are variable in size, we warp them into a small spatial fixed extent of $H \times W$ by applying RoI pooling, where H and W are hyper-parameters and independent of the RoIs [82]. The RoI pooling layer is a specific case of [90] where there is only one pyramid level [82].

The RoI pooling layer works as follows: it first divides a given RoI of size $h \times w$ into an $H \times W$ grid of sub-windows of approximate size $h/H \times w/W$. Then, applies pooling (*e.g.* max pooling [77]) to the values of each sub-window, corresponding to the output grid cell. As in standard pooling layers, RoI pooling is applied for every channel in feature map. We illustrate this procedure in Figure 4.4, where we have initially a feature map of size 8×8, and a RoI with $h = 5$ and $w = 7$. In this illustration, we apply RoI pooling to obtain a warped feature map of predefined size with $H = 2$ and $W = 2$.

4.3.2 Multi-task loss

The Fast R-CNN outputs, per RoI, a discrete probability distribution $\hat{\mathbf{p}} = (\hat{p}_0, \ldots, \hat{p}_K)$, over $K + 1$ categories (K object categories +1 for background) and the bounding box regression offset ${}^k\hat{\mathbf{t}} = ({}^k\hat{t}_x, {}^k\hat{t}_y, {}^k\hat{t}_w, {}^k\hat{t}_h)\ \forall\, k \in K$. The offset ${}^k\mathbf{t}$ is parameterized relative to the region proposals and represents a scale-invariant translation and log-space height/width shift relative to an object proposal. We follow the parametrization given in the set of Equations (4.6). We come back to this parametrization scheme later when we talk about RPN and anchor boxes, in Section 4.4.1.

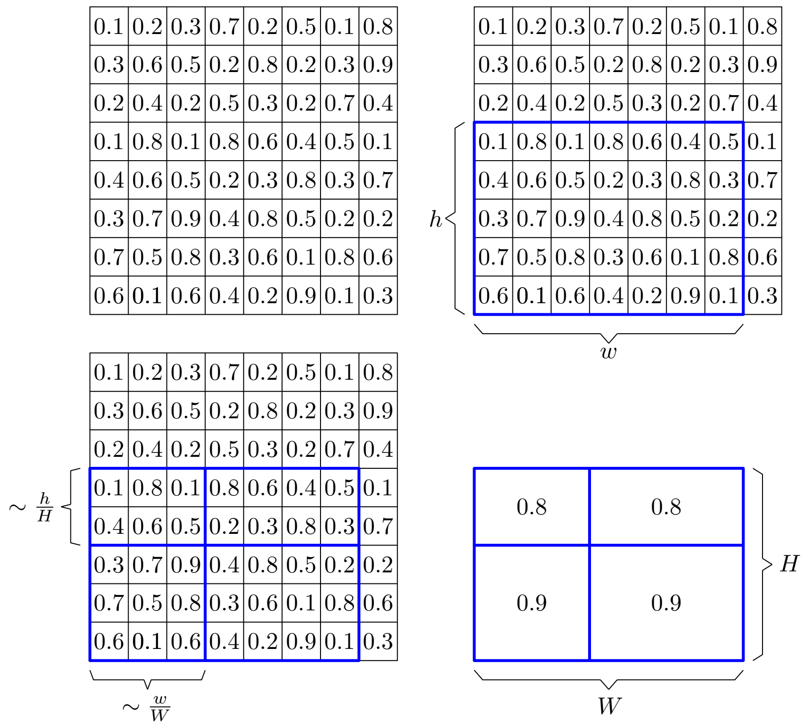


Figure 4.4: RoI pooling layer. Top left: feature maps; top right: RoI (blue) overlap with feature map; bottom left: split RoI into input dimension; bottom right: warped RoI obtained after applying max pooling in each section.

We label each training RoI with a ground truth class u and a bounding box regression target $\mathbf{v}$. We train both classification (*cls*) and regression (*reg*) layers using the multi-task loss given by

$$\mathcal{L}(\hat{\mathbf{p}}, u, {}^{u}\hat{\mathbf{t}}, \mathbf{v}) = \mathcal{L}_{cls}(\hat{\mathbf{p}}, u) + \lambda[u \geq 1]\mathcal{L}_{reg}({}^{u}\hat{\mathbf{t}} - \mathbf{v}) \tag{4.1}$$

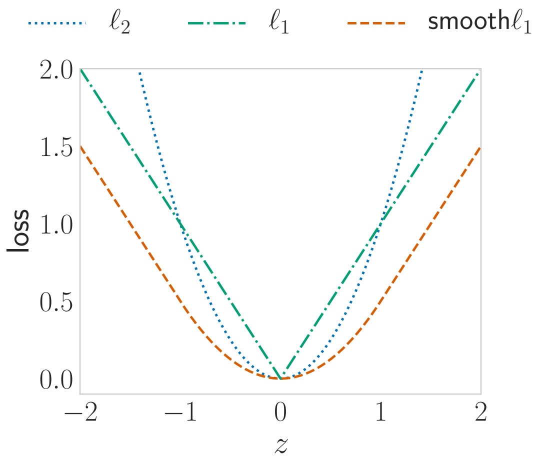
where, $\mathcal{L}_{cls}(\hat{\mathbf{p}}, u) = -\log \hat{p}_u$ is the negative log-likelihood loss for true class u and $\mathcal{L}_{reg}({}^{u}\hat{\mathbf{t}} - \mathbf{v})$ is the regression loss between the predicted tuple ${}^{u}\hat{\mathbf{t}} = ({}^{u}\hat{t}_x, {}^{u}\hat{t}_y, {}^{u}\hat{t}_w, {}^{u}\hat{t}_h)$ and the bounding box regression target $\mathbf{v} = (v_x, v_y, v_w, v_h)$, for class u. The Iverson bracket indicator function allows the regression loss to be activated only for class objects (background class is labeled as $u = 0$). It evaluates to 1 when $u \geq 1$ and 0, otherwise. This is done since there are no ground truth for the background RoIs. Fast R-CNN uses the smoothℓ_1 loss for bounding box regression,

$$\mathcal{L}_{reg}({}^{u}\mathbf{t} - \mathbf{v}) = \sum_{i \in \{x,y,w,h\}} \text{smooth}\ell_1({}^{u}t_i - v_i), \tag{4.2}$$

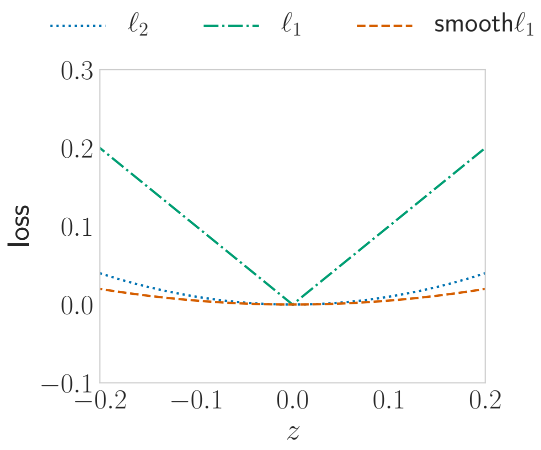
where the smooth$\ell_1(z)$ function is defined as

$$\text{smooth}\ell_1(z) = \begin{cases} 0.5z^2, & \text{if } |z| < 1 \\ |z| - 0.5, & \text{otherwise.} \end{cases} \tag{4.3}$$

The smoothℓ_1 is more robust to outliers than the ℓ_2-loss used in R-CNN [81], which requires a careful learning rate tuning in order to prevent exploding gradients in case of unbounded regression targets [82]. Actually, one may interpret smoothℓ_1 as a combination of ℓ_1- and ℓ_2-loss. When the absolute value of the argument is high (in this case $|z| \geq 1$), it behaves like a linear function (ℓ_1-loss), and when the absolute value of the argument is close to zero (value of $|z| < 1$), it behaves like a quadratic function (ℓ_2-loss), as shown in Figure 4.5. Therefore, it is possible to take advantage of both losses, steady gradients for large values of errors and less oscillation during updates when the error is small. All regression targets $v_i \ \forall i \ \in \{x, y, w, h\}$ are normalized to have zero mean and unit variance. The λ parameter balances the two loss terms, and is usually set to 1 [82].



(a) plot of ℓ_1, ℓ_2, and smoothℓ_1.



(b) Close look at the intersection.

Figure 4.5: Comparison between ℓ_1, ℓ_2, and smoothℓ_1 losses.

4.3.3 Training and testing Fast R-CNN

One may train the Fast R-CNN using back-propagation and stochastic gradient descent (SGD) [78]. Each mini-batch is hierarchically sampled, by first sampling N images and then S/N RoIs from each image, where S is the total number of RoI samples. This sampling scheme reduces the computation, since RoIs from the same image share computations during forward and backward passes [82]. For training Fast R-CNN, we sample up to 1:3 ratio of positive to negative samples, as the background is more common than the foreground in an image. The positive samples are the proposals that have an IoU (for a more detailed description refer to Section 5.1) overlap of at least 0.5 with a ground truth box. These are the foreground examples *i.e.*, $u \geq 1$. The negative samples *i.e.*, background samples ($u = 0$) are the proposals with a maximum IoU overlap in the range $[0.1, 0.5)$ with all ground truth boxes. The FC layers for classification and regression are initialized by drawing the weights from a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ with mean $\mu = 0$ and standard deviation $\sigma = 0.01$ and $\sigma = 0.001$, respectively.

At the test phase, Fast R-CNN takes an image and a set of S pre-computed proposals as input. For each testing proposal s, Fast R-CNN outputs a posterior discrete probability distribution $\hat{\mathbf{p}}$ of the K classes. It also outputs a set of predicted bounding boxes offsets relative to s. For each detection, we assign a confidence score for each class k by using the estimated probability $\hat{p}_k$. Finally, we apply non-maximum suppression (NMS) independently for each class in order to eliminate multiple detections for the same instance of class [81].

4.4 Faster R-CNN

Both R-CNN and Fast R-CNN use an external method to generate the proposals [81, 82]. The region proposal methods generally run on CPU and are time-consuming, affecting the network performance [76]. This stage is, up to now, the bottleneck of the region-based detectors.

The Faster R-CNN [76] presents a new mechanism that eliminates the need for an external region proposal method: the region proposal network (RPN), which is a convolutional network that learns the regions derived from the feature maps.

The Faster R-CNN works akin to Fast R-CNN. First, the image is fed into a ConvNet that outputs the convolutional feature maps. Based on these maps, the RPN then predicts the proposals, which are warped by the RoI pooling layer and then delivered to the FC layers that classify the RoIs and refine them by predicting an offset for the bounding boxes. Hence, Faster R-CNN comprises two modules: RPN, to generate the proposals, and Fast R-CNN detector, as shown in the Figure 4.6.

4.4.1 The region proposal network

The RPN works by sliding $n \times n$ convolutional filters over the feature map from the last feature extractor convolutional layer to generate the class-agnostic region proposals. The hyper-parameter n (typically, $n = 3$) must be chosen by taking the effective receptive field (the region of the input image that a neuron – the filter at a given position – oversees) into consideration [76]. So, each $n \times n$ spatial location is mapped to a lower dimension. The resulting features are fed into two parallel FC layers – a box-regression layer (*reg*) and a box-classification layer (*cls*). These last layers may be implemented with 1×1 convolutional layers [76], as shown in Figure 4.7. Thus, RPN is a fully convolutional network (FCN) [91] so, it is translation invariant up to the network's total stride [76].

At each $n \times n$ sliding filters position, RPN predicts β region proposals. Hence,

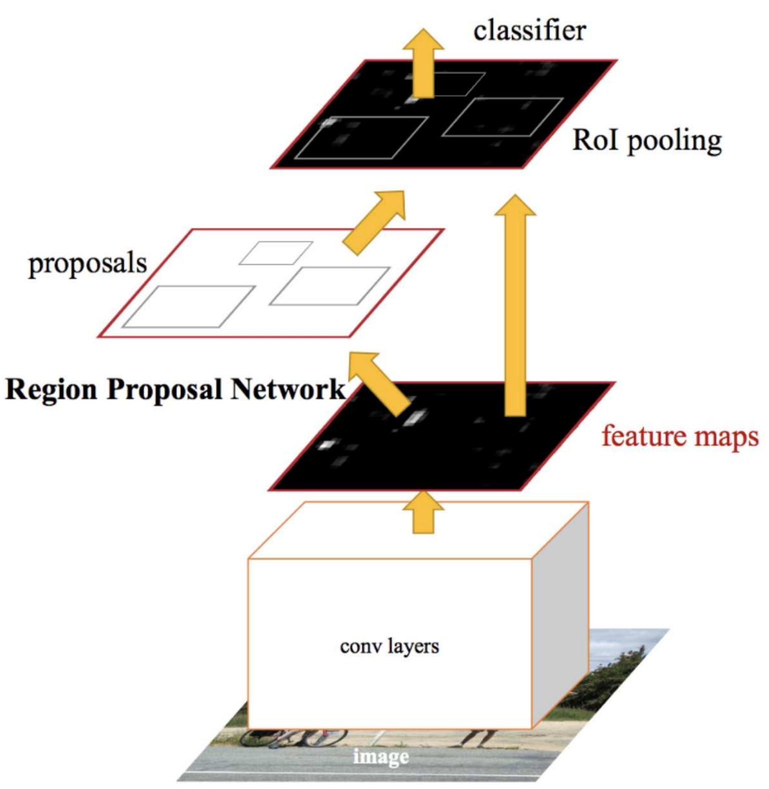
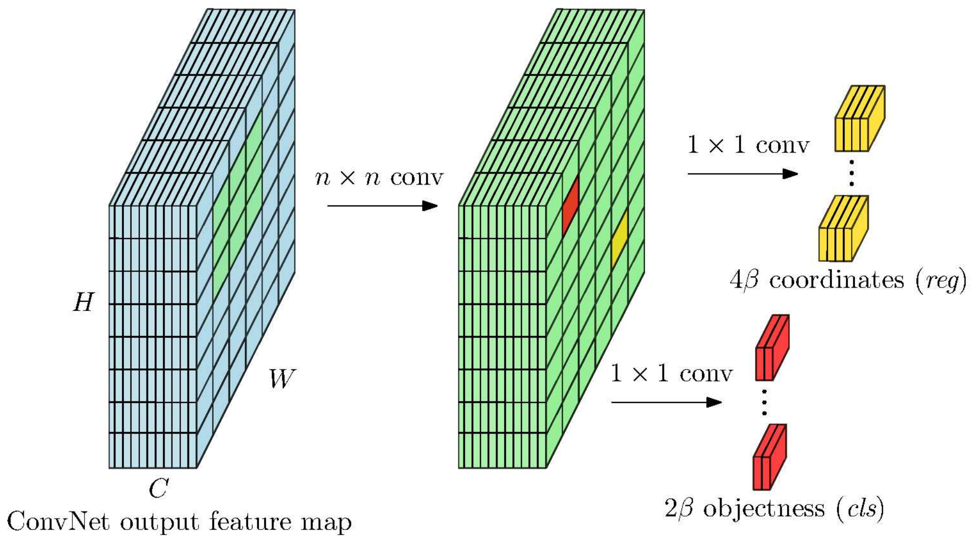


Figure 4.6: Faster R-CNN. Source: [76].

for each position, the *reg* layer outputs 4β encoded coordinates whereas *cls* layer outputs 2β scores that estimate the probability of object or not object, which we call "objectness" (see Figure 4.7). In Faster R-CNN [76], the *cls* layer is implemented as two-class softmax. This could be replaced with logistic regression generating only β class scores [76].



$n \times n$ conv
1×1 conv
4β coordinates (*reg*)
1×1 conv
2β objectness (*cls*)
H
W
C
ConvNet output feature map

Figure 4.7: RPN as a fully convolutional network.

Anchors

As mentioned, at each position of feature map, RPN predicts β proposals. Figure 4.8 illustrates $\beta = 3$ proposals for a specific location in the feature map. The proposals are parameterized relative to prior reference boxes which are called anchors. The anchors are centered at each spatial location of the output feature map, and each anchor is associated with a scale and aspect ratio. Therefore, for a feature map of size $W \times H$ we end with $WH\beta$ anchors in total. The original implementation of Faster R-CNN uses three scales and three aspect ratios, yielding $\beta = 9$ anchors at each position [76]. Faster R-CNN predicts offsets δ_x, δ_y that are relative to those

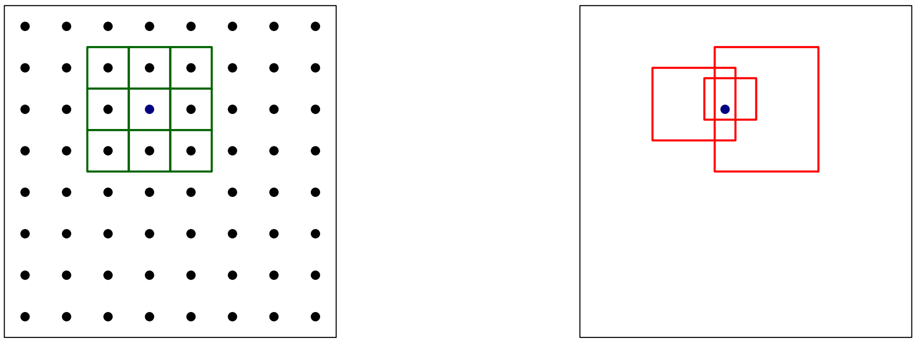
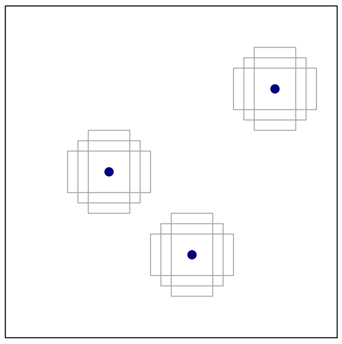

(a) 8×8 feature map and 3×3 filter. (b) RPN proposals for a specific location.

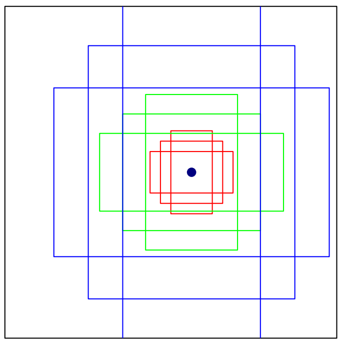
Figure 4.8: RPN proposals for a specific location in the output feature map.

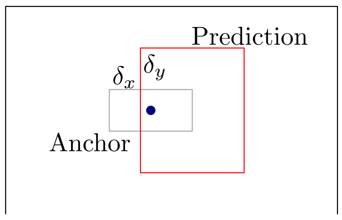
anchors. We illustrate the anchors positions, scales and aspect ratios, and offsets, in Figure 4.9. We represent the anchors at only 3 different spatial locations (that is for illustration purposes; however, anchors exist for every single position in the feature map). In Figure 4.9b, we show the anchors at different scales and aspect ratios for a specific spatial position; where the 3 different colors are different scales (*e.g.* $32^2, 64^2, 128^2$ pixels) and for each color, we have 3 aspect ratios (*e.g.* 1:1, 1:2, 2:1). Lastly, in Figure 4.9c, we illustrate the prediction and anchor offset. Figure 4.10 shows the RPN at a single position along with the β anchors boxes.

Loss function

The RPN outputs object class-agnostic region proposals. It assigns a binary label to each anchor: positive for anchors bounding an object, and negative otherwise. We define as positive the anchors that have (i) the highest IoU; or (ii) an IoU of, at least, 0.7 with the ground truth annotation. We keep the first condition to ensure that we have positive examples in case the first one fails. One should notice that a single ground truth box may assign positive labels to multiple anchors [76]. The negative anchors are those with IoU ratio lower than 0.3 with all ground truth boxes [76]. Anchors outside these conditions do not influence the training phase.


(a) Anchors at specific locations.


(b) Anchors at different scales and aspect ratios for a specific location.



(c) Prediction offset.

Figure 4.9: Anchor boxes at specific locations, different scales and aspect ratios for a specific location, and prediction offset.

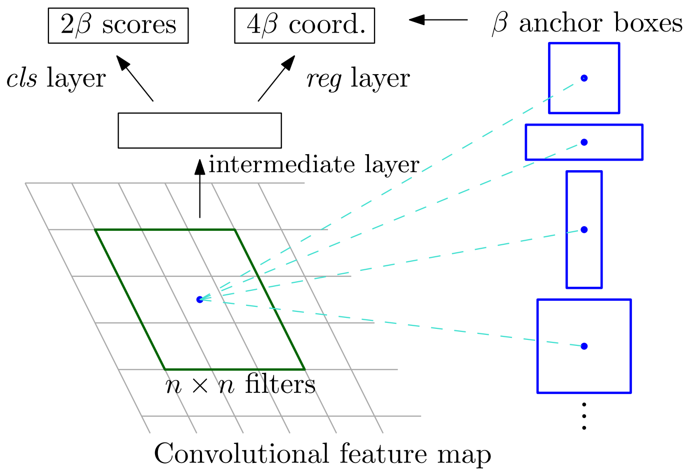


Figure 4.10: Region proposal network (RPN).

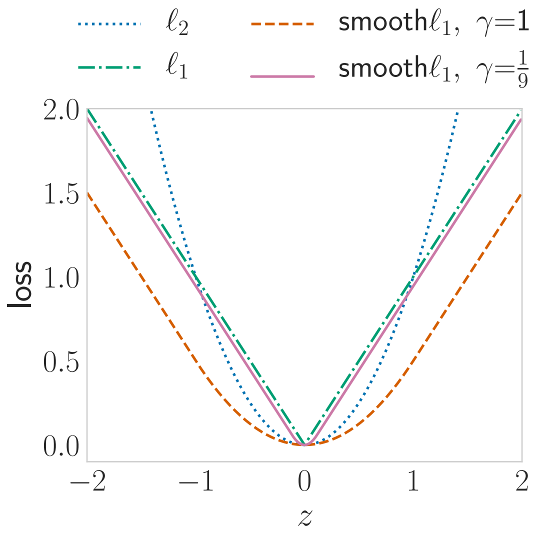
Given these definitions, we aim to minimize the loss function $\mathcal{L}(\{\hat{p}_i\}, \{\hat{\mathbf{t}}_i\})$, defined in Equation (4.4) w.r.t. the outputs $\{\hat{p}_i\}$ and $\{\hat{\mathbf{t}}_i\}$ of the *cls* and *reg* layers, respectively, following the multi-task loss in Fast R-CNN (section 4.3.2) [76],

$$\mathcal{L}(\{\hat{p}_i\}, \{\hat{\mathbf{t}}_i\}) = \frac{1}{N_{cls}} \sum_i \mathcal{L}_{cls}(\hat{p}_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* \mathcal{L}_{reg}(\hat{\mathbf{t}}_i, \mathbf{t}_i^*), \tag{4.4}$$

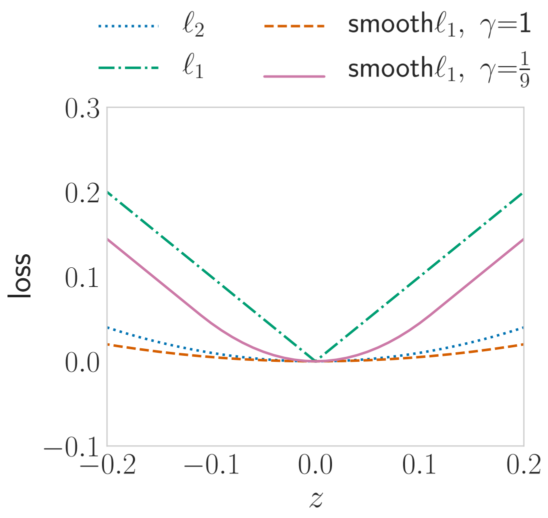
where i is the index of an anchor in a mini-batch; $\hat{p}_i$ the predicted probability of anchor i being an object; p_i^* the ground truth label which is 1 for positive anchors, and 0 for negative anchors; $\hat{\mathbf{t}}_i$ and $\mathbf{t}_i^*$ the vectors representing the 4 parameterized coordinates of the predicted bounding box, and the ground-truth box associated with a positive anchor, respectively; $\mathcal{L}_{cls}$ the classification log loss over two classes (foreground$\times$background); and $\mathcal{L}_{reg}(\hat{\mathbf{t}}_i, \mathbf{t}_i^*) = \text{smooth}\ell_1(\hat{\mathbf{t}}_i - \mathbf{t}_i^*)$ is the regression loss, being smooth$\ell_1(\cdot)$ similar to the robust loss function seen in Section 4.3.2 (Equation (4.2)), except for the inclusion of the parameter γ that controls where the function change from quadratic to linear. Hence, we redefine smooth$\ell_1(z)$.

$$\text{smooth}\ell_1(z) = \begin{cases} \frac{0.5}{\gamma} z^2, & \text{if } |z| < \gamma \\ |z| - 0.5\gamma, & \text{otherwise.} \end{cases} \tag{4.5}$$

As $\gamma \to 0$ the smooth$\ell_1(z)$ approaches to ℓ_1-loss. The reason for doing this is because unlike in Fast R-CNN, in Faster R-CNN the RPN bounding box regression targets v_i are not normalized by their variance since the statistics of the targets are constantly changing throughout learning. By including this γ parameter one may better approximate smoothℓ_1 to ℓ_1-loss (robust to outliers) and maintain the ℓ_2-loss properties for small error values, as shown in Figure 4.11. Although not reported in Faster R-CNN paper [76], the implementation uses $\gamma = \frac{1}{9}$.



(a) plot of ℓ_1, ℓ_2, and smooth–ℓ_1 (for two different values of γ).



(b) Close look at the intersection.

Figure 4.11: Comparison between ℓ_1, ℓ_2, and smooth–ℓ_1 (for two different values of γ) losses.

One may notice that the regression loss $\mathcal{L}_{reg}$ is activated only for positive anchors since $p_i^* = 1$ for positive anchors and $p_i^* = 0$ otherwise. The respective loss terms are normalized by the mini-batch size N_{cls} and the number of anchor locations N_{reg} and weighted by a balancing term λ so that *cls* and *reg* terms have roughly equal contributions. For instance, Faster R-CNN uses $N_{cls} = 256$, $N_{reg} \sim 2,400$, and $\lambda = 10$ [76]. On the other hand, they experimentally show that this normalization is not crucial and the results are insensitive to a wide range of values for λ [76].

For bounding box regression, Faster R-CNN adopts the four coordinates parameterizations, presented in the set of Equations (4.6) [76],

$$\begin{aligned}
\hat{t}_X &= \frac{(\hat{x} - x_\alpha)}{w_\alpha}, & \hat{t}_Y &= \frac{(\hat{y} - y_\alpha)}{h_\alpha}, \\
\hat{t}_W &= \log\left(\frac{\hat{w}}{w_\alpha}\right), & \hat{t}_H &= \log\left(\frac{\hat{h}}{h_\alpha}\right), \\
t^*_X &= \frac{(x^* - x_\alpha)}{w_\alpha}, & t^*_Y &= \frac{(y^* - y_\alpha)}{h_\alpha}, \\
t^*_W &= \log\left(\frac{w^*}{w_\alpha}\right), & t^*_H &= \log\left(\frac{h^*}{h_\alpha}\right),
\end{aligned} \tag{4.6}$$

where x and y denote the box's center coordinates, and w and h its width and height. We differ the predicted, anchor, and ground truth boxes' specifications, using variables $\hat{x}, x_\alpha$, and x^* (likewise for y, w, and h), respectively. One may interpret this as bounding box regression from an anchor box to an adjacent ground truth box [76]. Other RoI-based methods [82] performs bounding box regression on features pooled from arbitrarily sized RoIs and the regression weights are shared by all region sizes. However, in this approach, the features used for regression has the same spatial size $(n \times n)$ as the feature maps. RPN learns a set of β bounding box regressors and each one is responsible for a scale and aspect ratio. These β regressors do not share weights. Thus, the design of anchors allows predicting boxes of various sizes, even though the features are of a fixed size/scale [76].

Training the RPN

We train the RPN end-to-end with back-propagation and SGD [78]. Due to the slow convergence of SGD, one can use the momentum method [92] to accelerate the learning process, and weight decay [77] as a regularization method. Following [82], each mini-batch of size N_{cls} comes from the same image that contains positive and negative anchors. One should notice that we may have many more negative than positive samples, as background boxes are more common than foreground in an image. This fact may cause a bias towards negative samples. Therefore, we randomly sample N_{cls} anchors (usually $N_{cls} = 256$ [76]) with a ratio up to 1:1 of positive to negative. One should pad the batch with negative samples to fill up the mini-batch. As a common practice, all shared convolutional layers are initialized to ImageNet pre-trained weights for classification [79, 80]. The FC layers used for classification and regression are randomly initialized by drawing the weights from a normal distribution $\mathcal{N}(\mu, \sigma)$ with mean $\mu = 0$ and standard deviation $\sigma = 0.01$ and $\sigma = 0.001$, respectively [76].

4.5 Mask R-CNN

Mask R-CNN [84] is a framework for object instance segmentation. It is built on top of Faster R-CNN [76] by adding a third branch in parallel to the regression and classification layers of Fast R-CNN [82]. Recapitulating, the Faster R-CNN extracts the images features by forwarding an image through a ConvNet. Next, it predicts the RoIs on the feature space by using RPN. Then, we warp the proposals to a fixed dimension by applying RoI pooling. Lastly, we feed these features into FC layers to make classification and bounding box regression. Mask R-CNN adds a third branch to Faster R-CNN which outputs the object mask [84], as shown in Figure 4.12. The mask of an object is its pixel-wise segmentation in an image. Instance segmentation is outside the scope of this work. However, we mention it since instance segmentation requires finner spatial layout of an object [84]. The RoI pooling in Faster R-CNN causes misalignments between the RoI and features. Hence, Mask R-CNN proposes RoIAlign that addresses these misalignments.

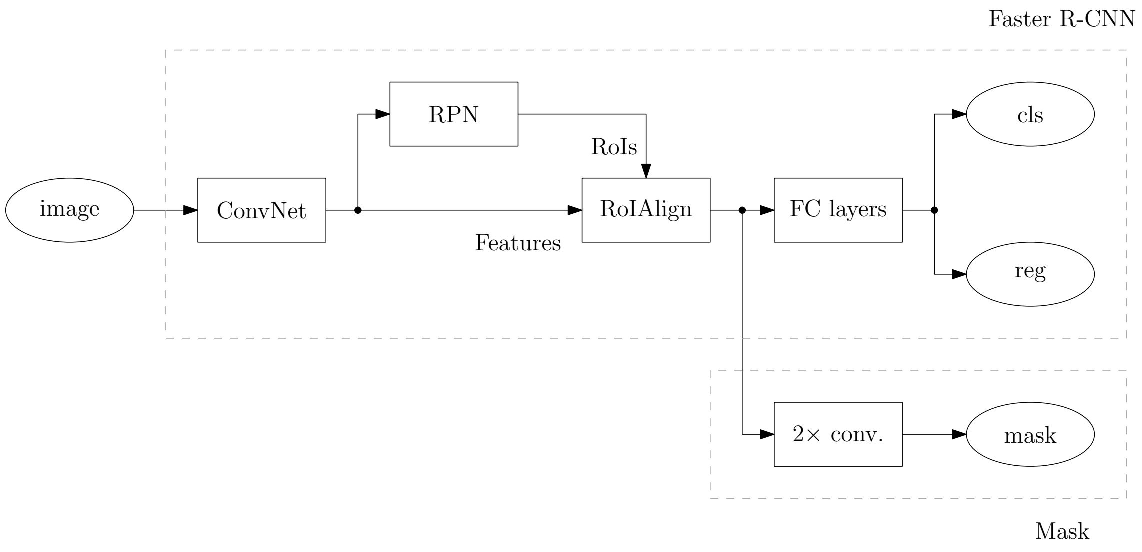


Figure 4.12: Mask R-CNN flow.

4.5.1 RoIAlign

RoI pooling warps the features inside a region proposal to a fixed size. It quantizes the RoI to the discrete granularity of the feature map. Likewise, it performs quantization when dividing the RoI into cells. The cell boundaries of the target feature map are forced to realign with the boundary of the feature map so that the cells might not be of the same size (revisit Figure 4.4). These quantizations result in misalignments between the RoI and features which harms object masks predictions [84].

Mask R-CNN substitutes RoI pooling by RoIAlign to avoid these misalignments. RoIAlign does not perform quantization but makes every target cell to have the same size, properly aligning the features with the input. For instance, it divides a given RoI of size $h \times w$ into an $H \times W$ grid of sub-windows of exactly size $h/H \times w/W$. It applies bilinear interpolation to compute the values in feature map at the cells [84], as shown in Figure 4.13. Then, it aggregates the results using max or average pool [77]. RoIAlign significantly improves the accuracy on both segmentation and localization tasks if compared to RoI pooling [84], so we use RoIAlign instead of RoI pooling.

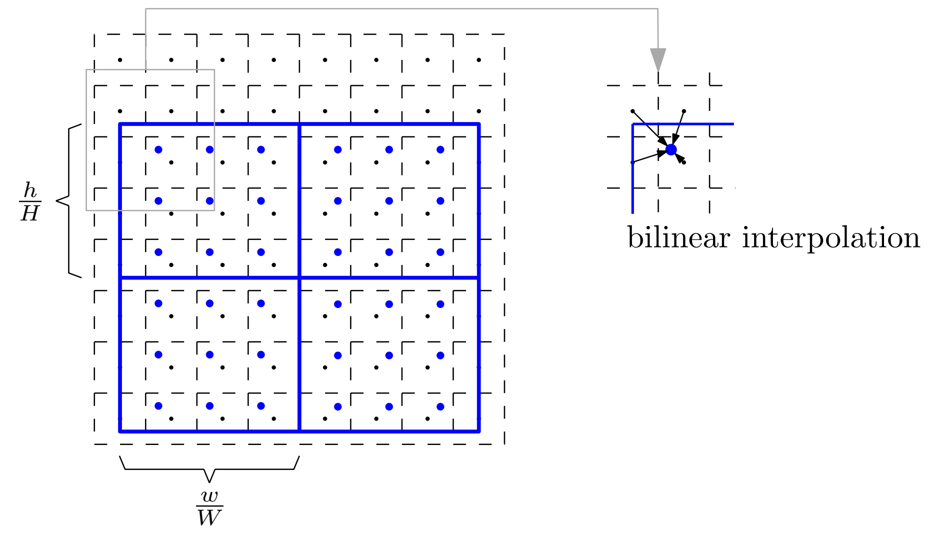


Figure 4.13: RoIAlign: we represent the feature map with dashed lines and small points. The RoI (2×2 cells) is represented with solid lines and 9 sampling points in each cell, that are computed by bilinear interpolation from the nearby grid points on the feature map.

4.6 Conclusions

In this chapter we detailed the method used for detection of potential mosquitoes breeding grounds: the Faster R-CNN. We have chosen this algorithm since it gives good accuracy results if compared to other object detectors. We also discuss how it evolved over the years pointing out the main contributions. For example, how RoIAlign outperforms RoIPooling.

Chapter 5

Results and Discussions

In this chapter, we discuss the experimental procedures adopted. First, we go through how we assess the performance of object detection. After that, we discuss the implementation details, including network architectures, as well as training and inference hyper-parameters set. Finally, we examine the obtained results both numerically and visually.

5.1 Evaluation

Assessing the performance of object detectors is a complex task since the models must be evaluated for both image classification (whether an object is on the image) and localization (where the object appears on the image, *i.e.*, bounding box regression).

Moreover, typical datasets have many classes with significantly nonuniform prior distribution over classes. Thus, a simple accuracy-based metric is not appropriate as it will introduce biases.

Another aspect to be taken into consideration is the risk of misclassification. Missing this information a priori leads to the necessity of introducing a "confidence score" or "model score" associated with each bounding box prediction. This allows to evaluate the model at different levels of confidence, *i.e.*, to regulate the trade-off between different types of classification error.

With these needs in mind, the mean Average Precision (mAP) [93] metric was introduced and it is widely used to evaluate models in object detection and segmentation online challenges. Before talking about the mAP, it is reasonable to understand the precision and recall concepts of a classifier and then discuss about Average Precision (AP) [93]. Before all of that, we need to consider the Intersection over Union (IoU) concept applied in object localization evaluation.

5.1.1 Object localization evaluation

As aforementioned, we need to evaluate our model regarding object localization. In other words, we need to know how closely the object boundary output overlaps the ground truth. To measure that, we use the IoU, which may be used to evaluate any algorithm that outputs bounding boxes.

The IoU is based on the Jaccard index [94], also referred to as the Jaccard similarity coefficient, which is a standard for evaluating the similarity between finite sample sets. One may compute the IoU as follows: given a predicted bounding box B_p associated to a ground truth bounding box B_{gt}, the IoU is the ratio of the area of the intersection to the union of the boxes areas,

$$IoU = \frac{\text{area } (B_p \cap B_{gt})}{\text{area } (B_p \cup B_{gt})}. \tag{5.1}$$

Figure 5.1 illustrates the IoU of a ground truth box (in green) and a prediction box (in red). The area (in blue) is given by the number of pixels inside the bounding boxes.

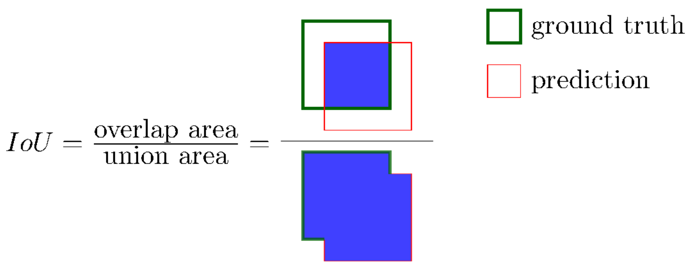


Figure 5.1: Intersection over Union (IoU).

Detections output by the model are considered as true positive (TP) if the IoU between a proposal and the ground truth of the object is equal to or greater than a certain threshold and as false positive (FP) otherwise. Typically, this threshold is set to 50%; however, evaluations with thresholds up to 95% are found in the literature. A false negative (FN) is a not detected ground truth object. In the object detection context, a true negative (TN) does not make sense since it would be all image regions that were correctly considered as background, which would amount to a large number of possible bounding boxes. Although, in this work, we only consider bounding boxes, IoU is also applicable to pixel-wise segmentation [84].

5.1.2 Precision & recall

The precision P measures the ratio of true positive to the total of **predicted detections**,

$$P = \frac{TP}{TP + FP} = \frac{TP}{\text{all detections}}, \tag{5.2}$$

i.e., how accurate the predictions are. The closer to 1.0 the precision score is, the more probable the detector output is correct.

The recall R, also referred to as sensitivity, measures the ratio of true positive detections to the total of **objects in the dataset**,

$$R = \frac{TP}{TP + FN} = \frac{TP}{\text{all ground truths}}, \tag{5.3}$$

i.e., how well it retrieves the objects in the dataset. The closer to 1.0 the recall score is, the more probable it is that the objects in the dataset are detected.

It is worth mentioning that there is an inverse relationship between these metrics as they inversely depend on the IoU threshold previously set.

5.1.3 (Mean) Average Precision for object detection

The procedure to compute the AP follows. For a given class, we rank all predictions by the model score, from highest to lowest. Then, we compute what the precision and recall would be for that output to be a considered as positive by the model. This is equivalent to varying the model score threshold that determines what is counted as a model-predicted positive detection of the class. Then, for calculating the AP score, we take the precision average across all recall values, as follows:

$$AP = \frac{1}{\text{card}(\mathcal{R})} \sum_{R \in \mathcal{R}} P_{\text{interp}}(R), \tag{5.4}$$

where, $\mathcal{R}$ is the set of recalls from 0 to 1 with step size R_s; card($\mathcal{R}$) is cardinality of the set $\mathcal{R}$, and; P_{interp} is defined as

$$P_{\text{interp}}(R) = \max_{\tilde{R} \geq R} P(\tilde{R}), \tag{5.5}$$

where $P(\tilde{R})$ is the measured precision at recall $\tilde{R}$. We execute this interpolation in order to smooth the oscillations caused by small variations in the precision computations. One may view the AP as the area under the curve (AUC) of the precision-recall graph. We approximate this computation by interpolating the precision at each recall level by R taking the maximum precision measured for a method for which the corresponding recall exceeds R, as shown in Equation (5.5).

In [93], the authors vary the recall from 0 to 1, with step size $R_s = 0.1$, so that card($\mathcal{R}$) $= 11$. In this work, following [84], we employ step size $R_s = 0.01$, so that card($\mathcal{R}$) $= 101$. By lowering R_s, we aim to better approximate the AUC of the precision-recall graph.

One may notice that to obtain a high score, a method must have high precision at all recall levels – this penalizes methods which retrieve only a subset of examples with high precision (*e.g.* an object in a certain position) [93]. Also, remember that the IoU has a direct impact on AP since it determines if a detection is a TP or FP.

This computation is exemplified next. Consider a dataset with 5 instances of a given class. We first rank all the model's predictions for that class according to the predicted confidence level (from the highest to the lowest), irrespective of correctness. Table 5.1 shows an example of hypothetical predictions for those 5 instances ranked by their confidence level. The column "Correct?" shows if the

Table 5.1: Example of ranked hypothetical detections.

Rank	Confidence	Correct?	Precision	Recall
1	0.99	True	1.00	0.2
2	0.95	True	1.00	0.4
3	0.82	False	0.67	0.4
4	0.81	False	0.50	0.4
5	0.79	True	0.60	0.6
6	0.78	False	0.50	0.6
7	0.74	True	0.57	0.8
8	0.73	False	0.50	0.8
9	0.63	False	0.44	0.8
10	0.62	True	0.50	1.0

detection match the ground truth for an IoU equal or higher than a threshold of, say 50% [93]. Let us consider the row with rank #3. The precision for that row is the proportion of TP, $P = 2/3 = 0.67$; and the recall is the ratio of TP to total of examples, $R = 2/5 = 0.4$. We can notice that the recall still increases as we include more predictions (lower the confidence model threshold), but the precision goes up and down. The Figure 5.2 shows the precision-recall curve, obtained by computing P and R for all rows.

Again, one may view AP as the AUC of the precision-recall curve. Remember that we approximate the computation by first smoothing the precision oscillations according to Equation (5.5), which is better understandable in Figure 5.3, where we give an example of computing $P_{\text{interp}}(0.7)$. The Figure 5.4 presents the curve of P_{interp} computed across all recall values. Finally, we may compute the AP of our example, using Equation (5.4). Since we varied the recall from 0 to 1 with $R_s = 0.1$,

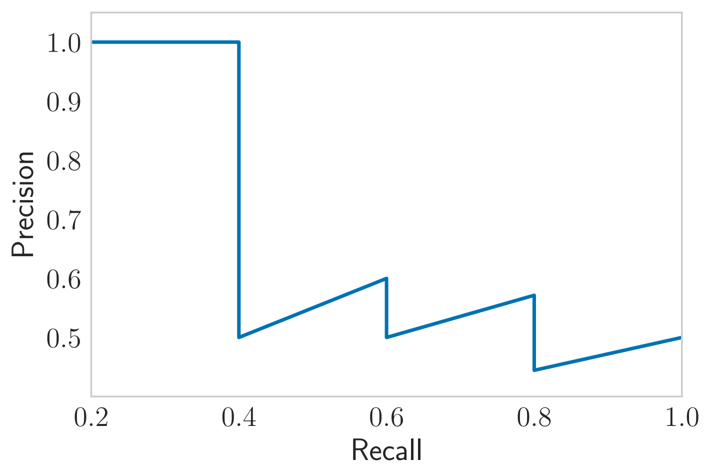


Figure 5.2: Precision-recall curve.

Rank	Confidence	Correct?	Precision	Recall	
1	0.99	True	1.00	0.2	
⋮	⋮	⋮	⋮	⋮	
6	0.78	False	0.50	0.6	$\leftarrow R = 0.7$
7	0.74	True	**0.57**	0.8	
8	0.73	False	0.50	0.8	
9	0.63	False	0.44	0.8	
10	0.62	True	0.50	1.0	

Figure 5.3: Example of computing P_{interp}. In this case, $P_{\text{interp}}(0.7) = 0.57$.

$\text{card}(\mathcal{R}) = 11$.

$$\begin{aligned} AP &= \frac{1}{11}\left(P_{\text{interp}}(0.0) + P_{\text{interp}}(0.1) + ... + P_{\text{interp}}(1.0)\right) = \\ &= \frac{1}{11}\left(1.00 + 1.00 + 1.00 + 1.00 + 1.00 + 0.60 + 0.60 + 0.57 + 0.57 + 0.50 + 0.50\right) = \\ &= 0.7582. \end{aligned}$$

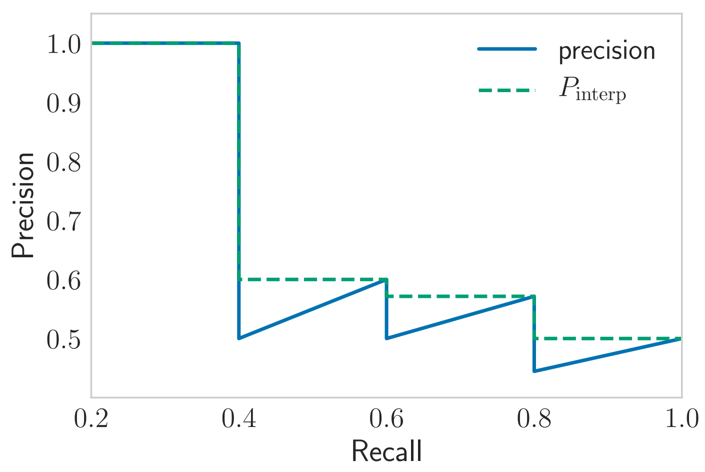


Figure 5.4: Precision-recall curve with P_{interp}.

So far, we have defined the AP and seen the impact of the IoU threshold in its computation. We may now calculate the mAP by computing the AP for all the M classes in the dataset and taking the average over them and/or IoU thresholds, as follows:

$$mAP = \frac{1}{M} \sum_{m \in M} AP_m, \tag{5.6}$$

where AP_m is the Average Precision computed at each class and/or IoU threshold m. Depending on the competition this procedure of computing the mAP may differ. In the next section we discuss two famous online object detections competitions.

5.1.4 Online challenges

The PASCAL Visual Object Classes (VOC) is a dataset for object detection [93]. In this challenge, a prediction is considered a TP if $IoU \geq 0.5$. In the case of multiple detections of the same object, it counts the first one as a positive and others as negatives. So, it is the responsibility of the competitor to deal with multiple detections for the same object. The mAP in PASCAL VOC is calculated by computing the AP as discussed previously, considering $IoU \geq 0.5$, and averaging over all 20 object categories in the dataset.

Latest works [84], tend to report results for the Microsoft Common Objects in Context (MSCOCO) dataset [95] only. There are 12 metrics to assess the performance of an object detector on MSCOCO. Nevertheless, we only focus on the 6 metrics based on AP. The primary challenge metric for this competition averages AP for IoU from 0.5 to 0.95 with a step size of 0.05 (AP at [0.5 : 0.05 : 0.95]). By averaging over the higher IoU thresholds instead of only considering one more tolerant threshold, say $IoU \geq 0.5$, tends to reward detectors with better localization. Other two MSCOCO challenge metrics consider only a single IoU threshold, one $IoU \geq 0.5$ (just like in PASCAL VOC) and another one $IoU \geq 0.75$. For the MSCOCO challenge, the AP is averaged over all 80 object categories to compute mAP.

In MSCOCO dataset 41% of objects are considered small (area $< 32^2$ pixels), 34% medium ($32^2 <$ area $< 96^2$), and 24% large object (area $> 96^2$) [95]. The object size affects the model accuracy substantially [93]. In [84, 93], it is possible to observe the performance of the methods increasing as object size increases. The MSCOCO challenge presents three metrics which considers objects size: $\mathrm{mAP_S}$, $\mathrm{mAP_M}$, $\mathrm{mAP_L}$, in order to evaluate the detections for small, medium and large objects areas. For those metrics, detections of objects with an area outside of a determined threshold are unconsidered. The area is computed as the number of pixels of the ground truth bounding box.

From now on, unless otherwise noted, the (m)AP is averaged over the multiple IoU values [0.50 : 0.05 : 0.95], for simplicity. We summarize the MSCOCO metrics based on AP in Table 5.2.

Table 5.2: Summary of MSCOCO metrics based on AP.

Metric	**Description**
mAP	mAP at IoU at 0.5 : 0.05 : 0.95
mAP_{50}	mAP at $IoU \geq 0.50$
mAP_{75}	mAP at $IoU \geq 0.75$
mAP_{S}	mAP for small objects (area $< 32^2$)
mAP_{M}	mAP for medium objects ($32^2 <$ area $< 96^2$)
mAP_{L}	mAP for large objects (area $> 96^2$)

5.2 Implementation details

Next, we discuss about the implementation details including network architectures and hyper-parameters used on training and test phases. We use the mask R-CNN benchmark implementation available under MIT-license [96]. It is worth mentioning that we do not tune any hyper-parameter since we do not have a validation set. All of them were chosen based on [84], and taking into account the characteristics of the dataset used in our experiments.

The experiments were performed in a machine equipped with 4 GTX 1080 GPUs, 64GB DDR4 2133MHz of RAM, Intel$^{\text{TM}}$ Core i7 6850-K 3.6 GHz processor, and Ubuntu 16.04 as the operational system.

5.2.1 Dataset

Since the MBG dataset described in Section 3.3 is still being labeled, we use the publicly available CEFET dataset[1] to train and evaluate our models. Again, the train-test split of this dataset is included in the annotation files. In the CEFET dataset, each video is cut into several parts. Two parts of the same video, for example, appear one in the training set and the other in the test set. As we are using an approach based on isolated images instead of video, this split may facilitate the task of our detector since in two takes of the same video we have the same background and objects placed at the same place. Therefore, in this work, we adopted a train-test split based on the videos *i.e.*, all the parts of a video are either in train or test set. Having split the videos, we extract images every 30 frames.

[1]from: `https://drive.google.com/open?id=1tDOVdb_vALUnD_cY3lQfOggoiM1F63Jl`.

In total, there are 419 images, containing 374 tires, for training and 144 images, containing 449 tires, for test.

Although we do not have the ground truth bounding boxes for the MBG dataset yet, we run the videos through our trained models and visually analyze some of the obtained results.

5.2.2 Network architectures

We instantiate Faster R-CNN with different network architectures. We define the Faster R-CNN as composed of two parts: (i) the network backbone: the convolutional network architecture (*e.g.* VGG [89], ResNet [97]) responsible for the feature extraction task over images; and (ii) the network head: responsible for image classification and bounding box regression tasks [84].

We use the network depth (number of stacked layers) nomenclature to denote the backbone architecture. We perform experiments by using ResNet [97] with depth of 50 and 101. Following the original implementation of Faster R-CNN with ResNets, we extract features from the final convolutional layer of 4th stage, which we call C4. This is widely used in the literature [86, 97, 98]. We denote this backbone by R-$< 50, 101 >$-C4. The network head follows the architectures presented in Faster R-CNN [76].

5.2.3 Training and inference

During training the positive samples are the RoIs that has an $IoU \geq 0.5$ with the ground truth and all other RoIs are considered as negative samples, as in [82]. We adopt image-centric training [82]. We resize the images, so that the shorter edge is not greater than 800 pixels resolution, while keeping the aspect ratio [84]. Our mini-batch has 4 images (we train on 2 GPUs, 2 images per GPU). From each image, we sample $N = 64$ RoIs with 1:3 ratio of positive to negatives [76, 82]. We train our models for 18k iterations, with learning rate of 0.005 which is decreased by 10 at the 12k and 16k iterations. We use weight decay of 0.0001 and momentum of 0.9.

We use RPN anchors at 5 scales (32, 64, 128, 256, and 512) and 3 aspect ratios (1:2, 1:1, and 2:1) with respect to the resized images input [99].

We set number of proposals of RPN ouput to 600. These boxes highly overlap. In order to reduce the redundancy caused by these overlaps, we use an IoU threshold for NMS at 0.7 and keep only the top-50 ranked proposals, based on *cls* score (see Section 4.4.1), for Fast R-CNN train [76].

We perform batch inference using 2 GPUs and 4 images per batch. We set the number of proposals as 300, run predictions on those, and suppress ambiguous detections by applying NMS [100] at IoU=0.5.

5.3 Results

In this section, we analyze our results both quantitatively and qualitatively. For the first one, we plot the precision-recall curve for results obtained from the test set. Also, we summarize these curves into single numbers as discussed in Section 5.1. Moreover, we make a visual analysis of our results by looking at the detection outputs from our models.

5.3.1 Quantitative results

It took about 2 h to train the models with R-50-C4 backbone while the models with R-101-C4 took about 3.5 h. For inference, the models with R-50-C4 backbone took about 90 ms per image against 140 ms for R-101-C4.

The Table 5.3 shows the result for the CEFET dataset. We report the mAP (averaged over multiple IoUs), mAP_{50}, mAP_{75}, mAP_{M}, mAP_{L} metrics, summarized in Table 5.2. We can notice an improvement of almost 5 points in mAP by only adopting a random horizontal flip, with probability of 50%, as data augmentation strategy in R-50-C4 backbone. Except mentioned otherwise, we keep this data augmentation method for the other experiments.

We also observe that Faster R-CNN with R-101-C4 backbone obtained the best result. This is due to deeper networks being better feature extractors. While deeper networks are more prone to overfitting due to the larger number of parameters, we did not observe any overfitting in our training.

We also evaluate the impact of object size in the results. As expected, we notice better results for large objects than for medium objects. Since we have no objects with area $< 32^2$ in the CEFET dataset, mAP_{S} does not apply.

Moreover, we plot the precision-recall curve for the trained model with R-101-C4 backbone (Figure 5.5) at IoU varying from 0.5 to 0.95 with 0.05 step. As expected, the higher the IoU threshold is, the worse the results are. That happens because as we increase the IoU threshold we only accept more accurate detections as TP, as a consequence, the precision and recall drops drastically. We can observe that

Table 5.3: Main results for CEFET dataset.

		backbone	mAP	mAP_{50}	mAP_{75}	mAP_{M}	mAP_{L}
train	Faster R-CNN (no aug.)	R-50-C4	89.86	90.19	90.19	86.15	92.16
	Faster R-CNN	R-50-C4	89.11	89.84	89.84	87.32	90.34
	Faster R-CNN	R-101-C4	88.95	89.48	89.48	87.52	91.02
test	Faster R-CNN (no aug.)	R-50-C4	43.81	62.25	53.64	34.46	59.56
	Faster R-CNN	R-50-C4	47.38	64.16	57.70	38.42	61.85
	Faster R-CNN	R-101-C4	49.31	66.68	62.61	39.46	65.21

we still can achieve satisfactory results at IoU up to 0.75 from which we achieve precision higher than 0.9 for all models. Unfortunately, all models do not achieve high precisions for high recalls. By analyzing the Equation (5.3), this result may be due to high rate of FN in our results.

We show Figures 5.6 and 5.7 for better analyzing the models at single IoU thresholds. As can be notice, R-101-C4 keeps higher precisions for higher recalls if compared to models with R-50-C4 backbone.

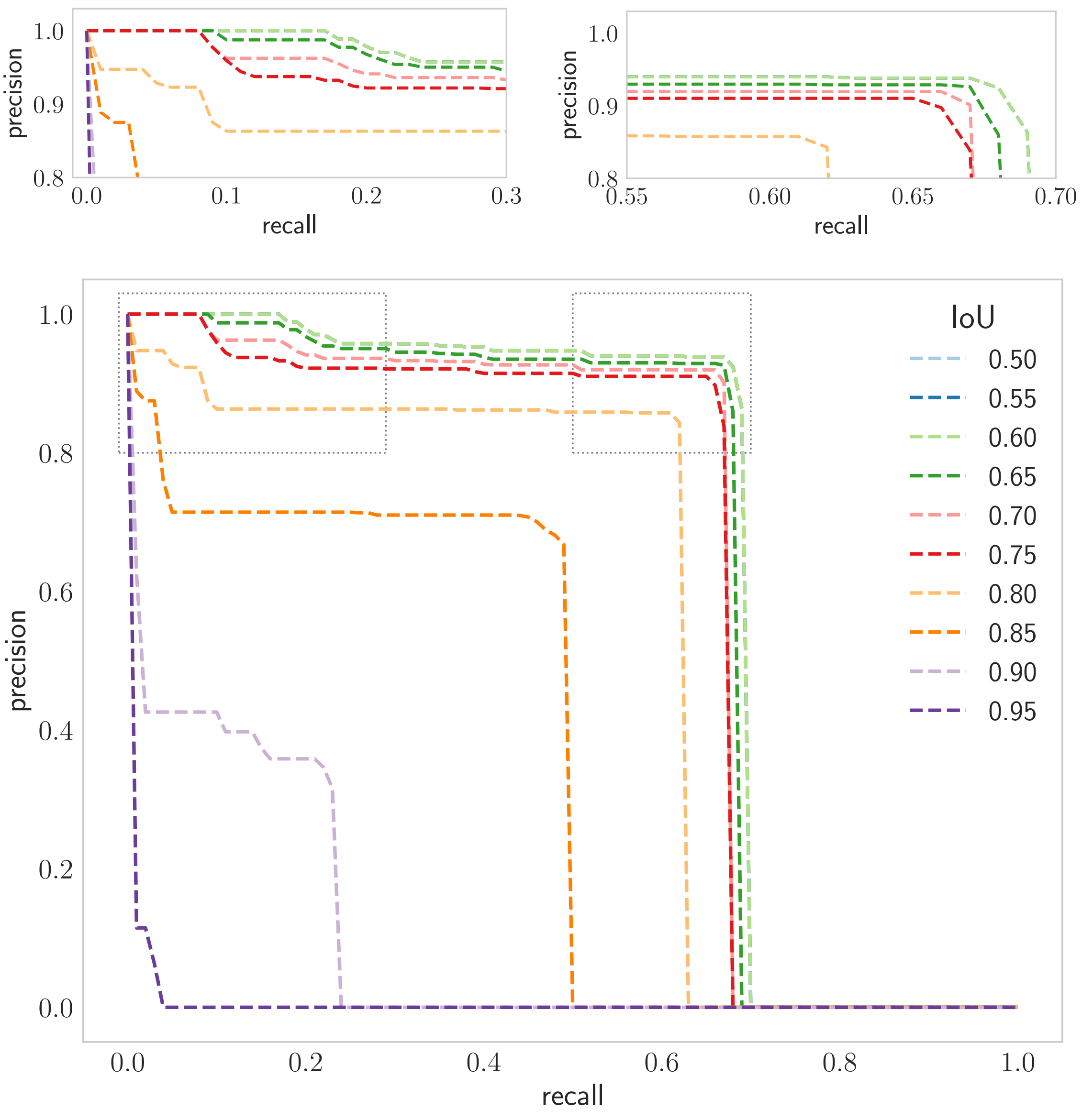


Figure 5.5: Precision-recall curve for R-101-C4 at various IoUs.

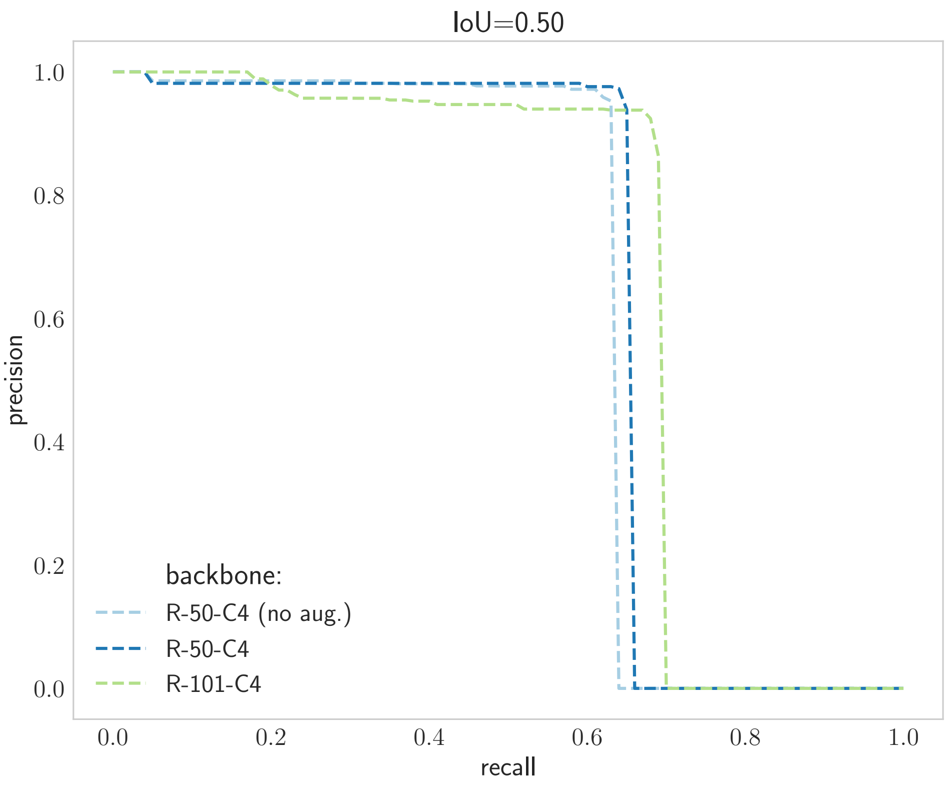


Figure 5.6: Precision-recall curve at IoU = 0.50.

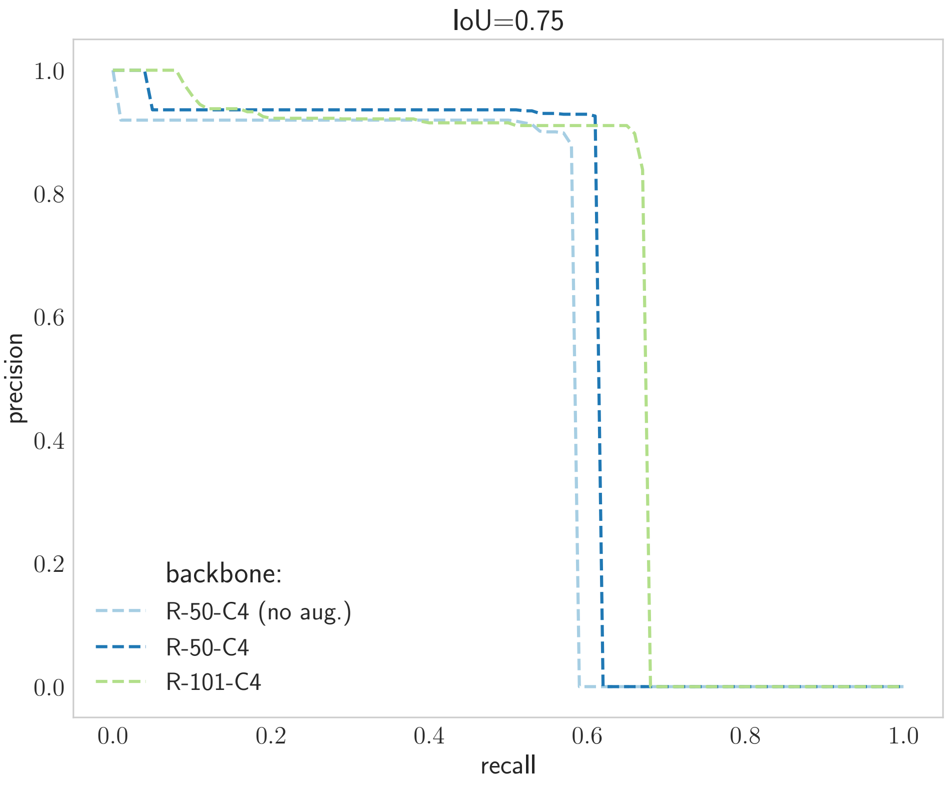


Figure 5.7: Precision-recall curve at IoU = 0.75.

5.3.2 Visual analysis

In this section, we discuss some visual results by analyzing the detections in the images. We try to associate the visualization with the numerical results obtained in the previous section. To do so, we plot the ground truth bounding boxes in blue and overlay the models detection in red, along with the label and confidence scores.

As mentioned, the recall is low for all models, which may be due to high rate of FN in our results. By looking at an example in Figure 5.8, one may notice that there are many hard-to-detect tires in the dataset, even for humans. In this image none of the models were capable to detect any tire.

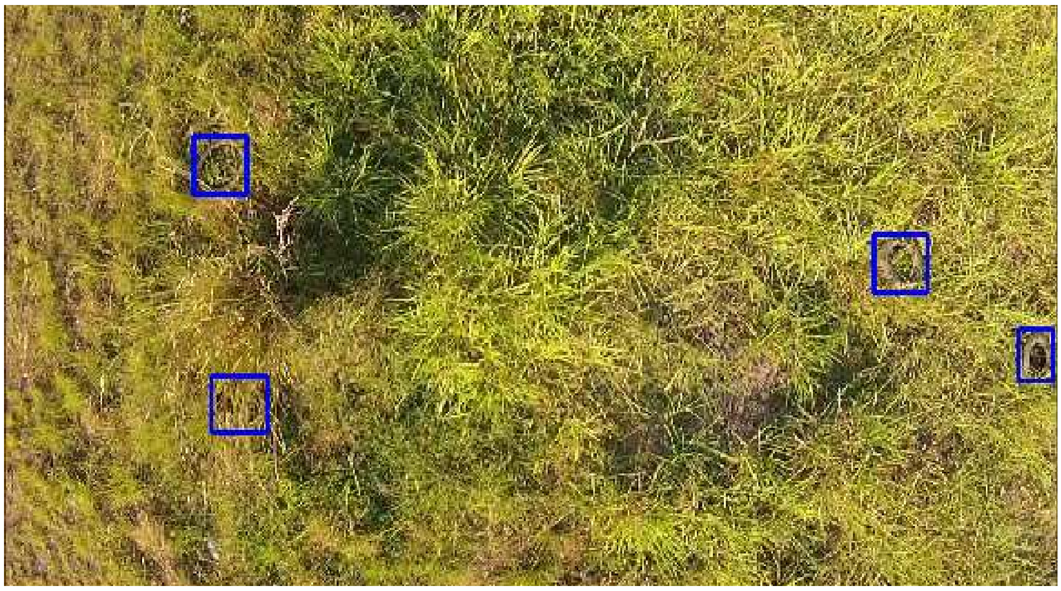
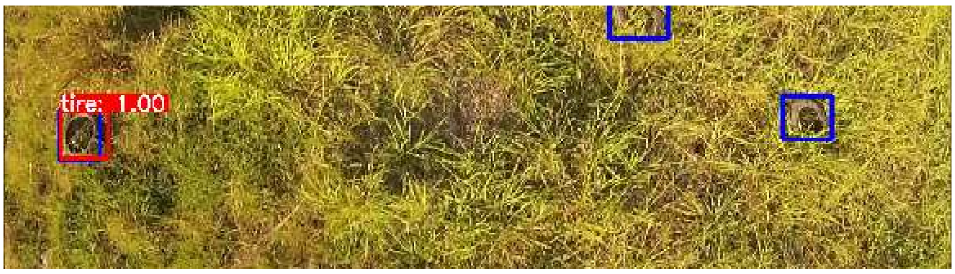


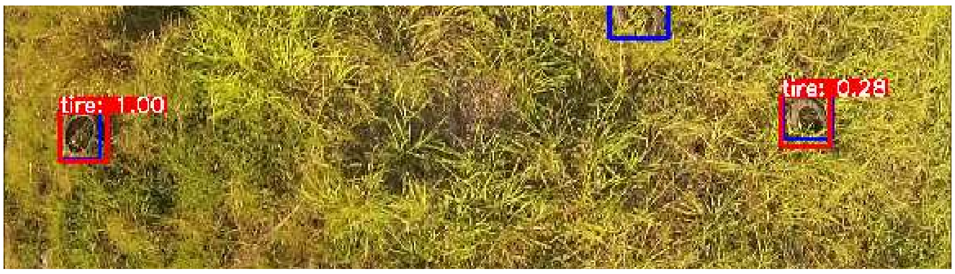
Figure 5.8: Hard example.

Another interesting fact can be observed in Figure 5.9 where the same tire (the rightmost one) was not detected by the R-50-C4 model without data augmentation; detected with a low confidence score (0.28) by the model with R-50-C4, using data augmentation; and detected with a high confidence score (1.00) by the model with R-101-C4, also using data augmentation. All models found the leftmost tire with the same confidence score and none of them found the tire at the top. Following this, we can observe a similar case in Figure 5.10, where both models with R-50-C4 were not capable of finding the tire, while the model with R-101-C4 detected it with a high confidence score (0.97).

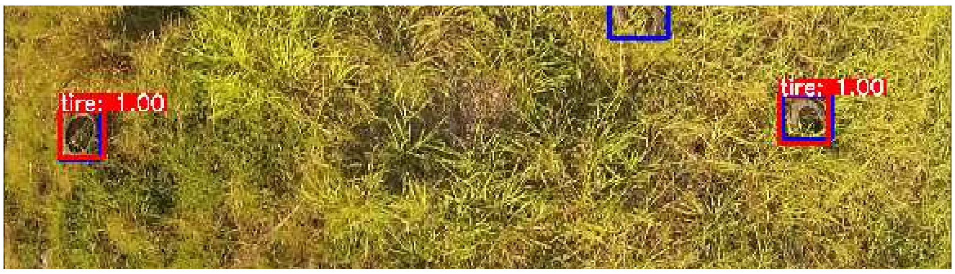
In Figure 5.11, we observe some case of false positives. For the same image, only the model with R-50-C4 backbone and data augmentation outputs the correct detections. The other two models output one FP each. The R-50-C4 (no aug.) wrongly outputs a tire with low confidence score (0.05) between two true tires; while the R-101-C4 predicted the yellow garbage bin as a tire with high confidence score (1.00).



(a) R-50-C4 (no aug.).

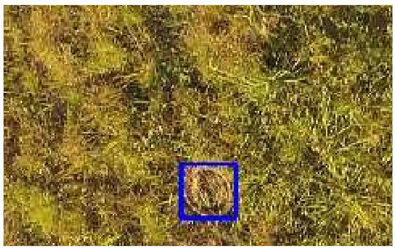


(b) R-50-C4.

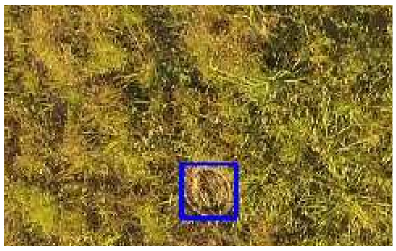


(c) R-101-C4.

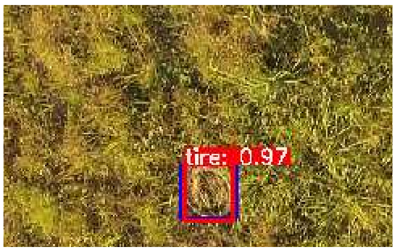
Figure 5.9: Detection improvement over the models (cropped images).



(a) R-50-C4 (no aug.).



(b) R-50-C4.



(c) R-101-C4.

Figure 5.10: Another detection improvement over the models (cropped images).

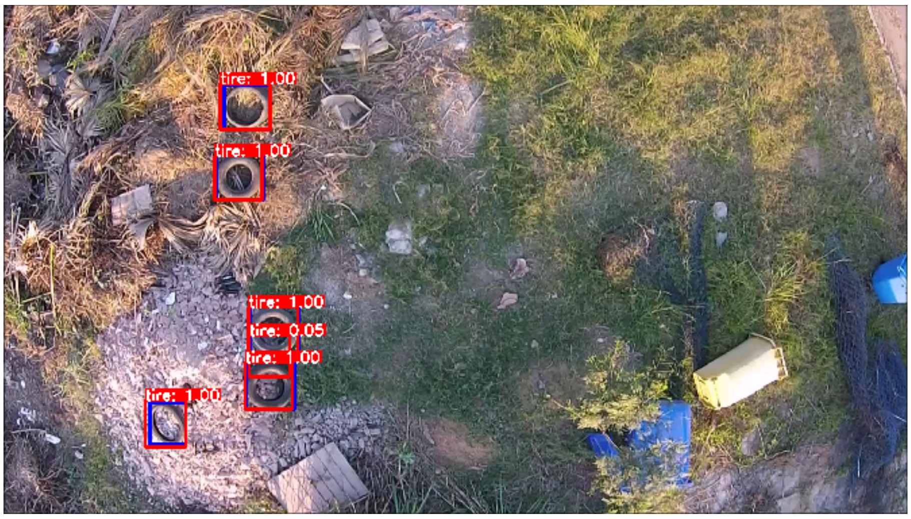
In Figure 5.12, all models could find almost all tires except for the one with high level of occlusion (the very middle one, under all tires). That tire is hard to be found even for humans. Besides, the dataset does not have many occlusions examples, which makes detection of such cases even harder.

In Figure 5.13 all models detect, with a high confidence score (1.00), a tire that had not been annotated in the dataset. Although is has been correctly detect, it was counted as a FP.

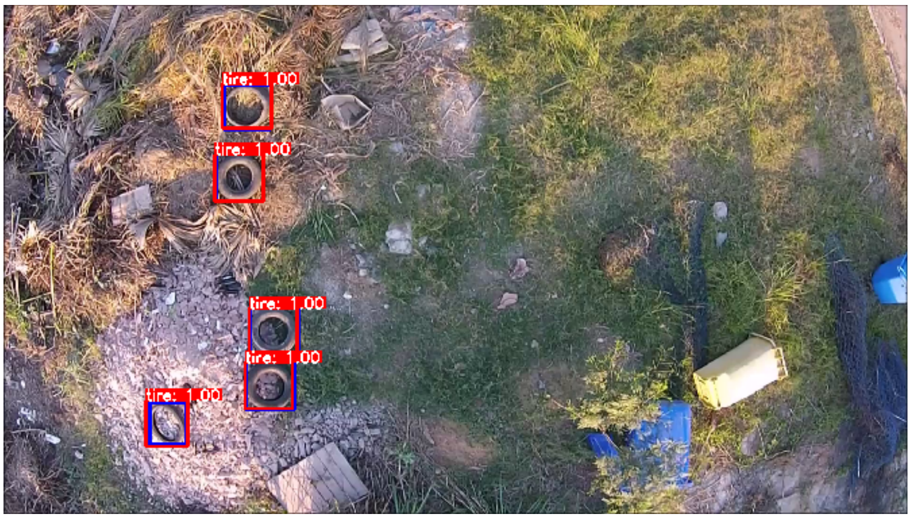
Even tough our models were trained using a small dataset, they were capable to detect tires in the MBG dataset, as depicted in Figure 5.14. However, they also detected a lot of false positives, as shown in Figure 5.15.

5.4 Conclusions

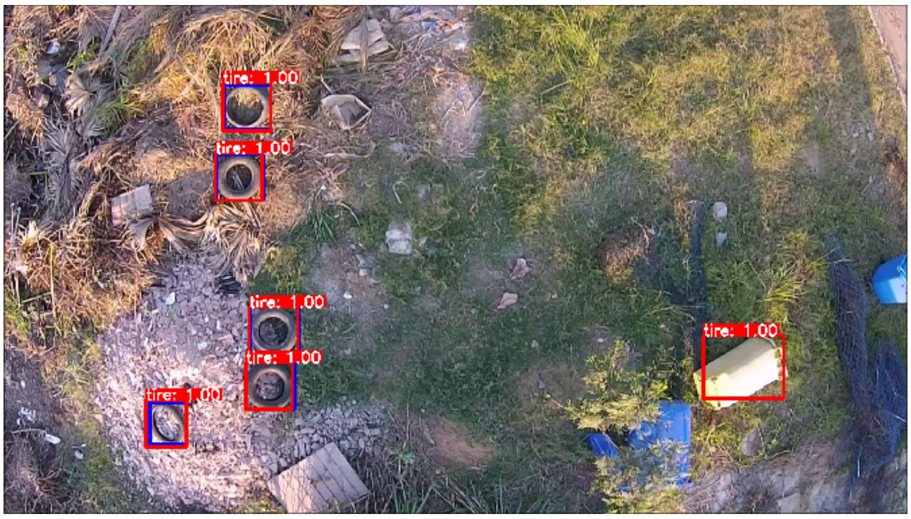
In this chapter we apply deep-learning based models to detect potential mosquito breeding sites, particularly tires. We have seen that deeper models achieved better results. Some further adjustments in the model can improve even more them. Nevertheless, the obtained results have shown promising and that resulting models trained with CEFET dataset may be useful in detecting potential mosquitoes breeding sites.



(a) R-50-C4 (no aug.).

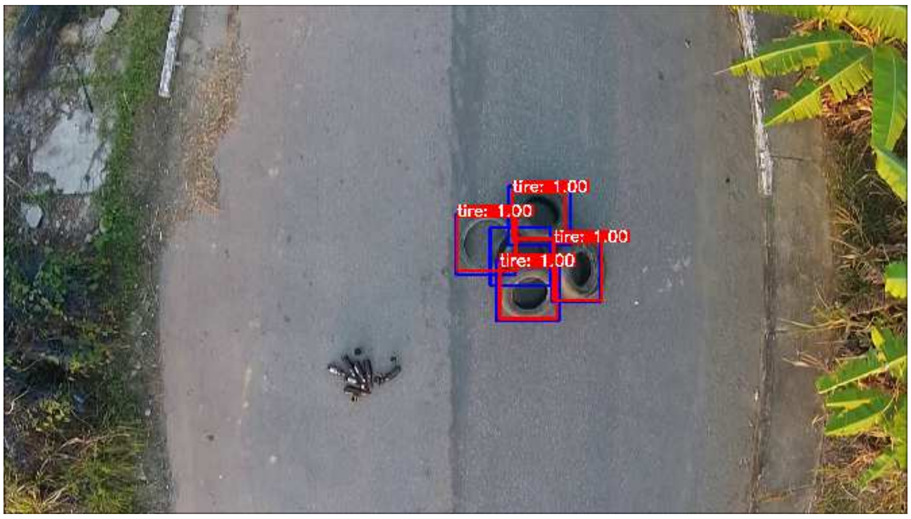


(b) R-50-C4.

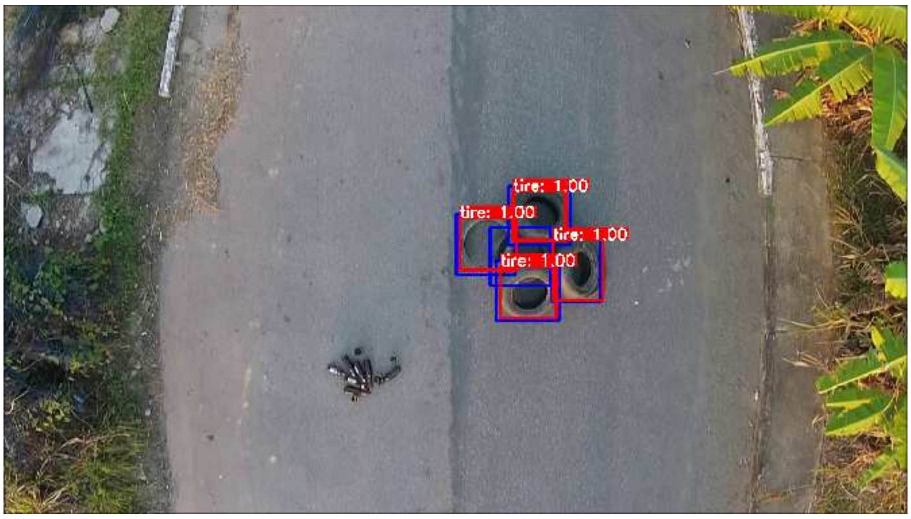


(c) R-101-C4.

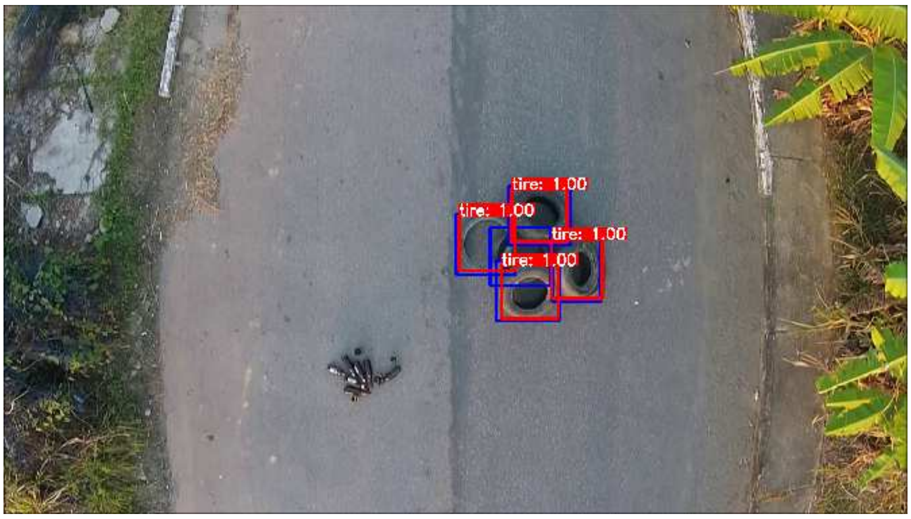
Figure 5.11: Some false positives cases.



(a) R-50-C4 (no aug.).



(b) R-50-C4.



(c) R-101-C4.

Figure 5.12: High occlusion example.

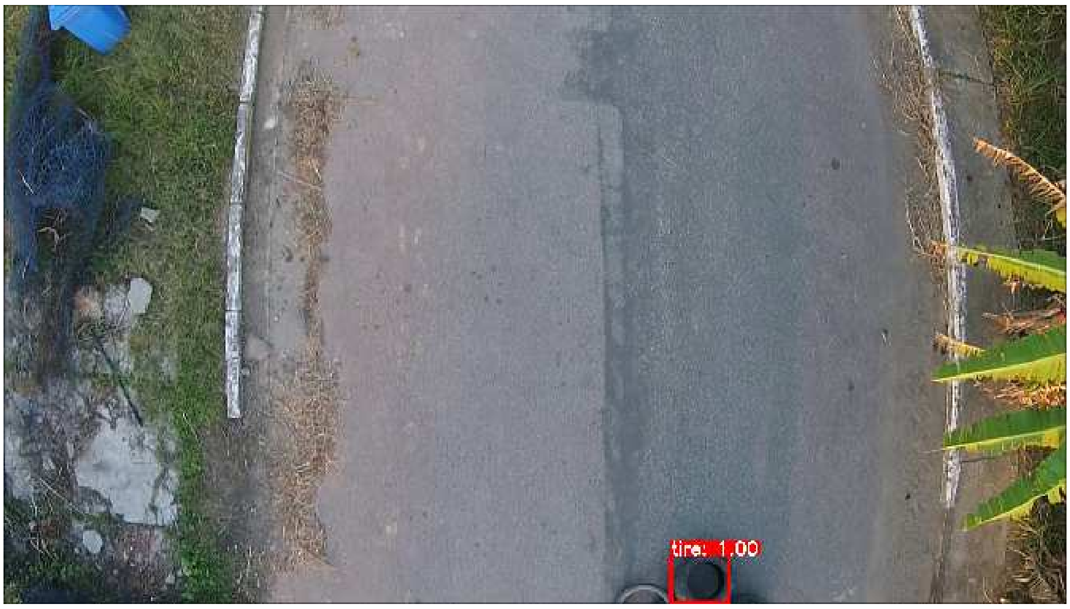


Figure 5.13: Wrong false positive.

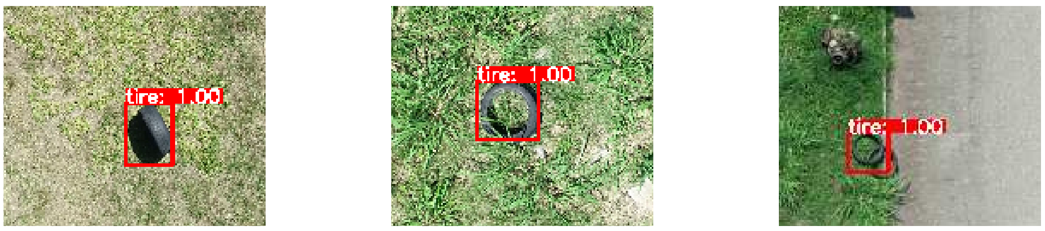


Figure 5.14: Tires from the MBG dataset detected by the models trained using CEFET dataset (cropped images).

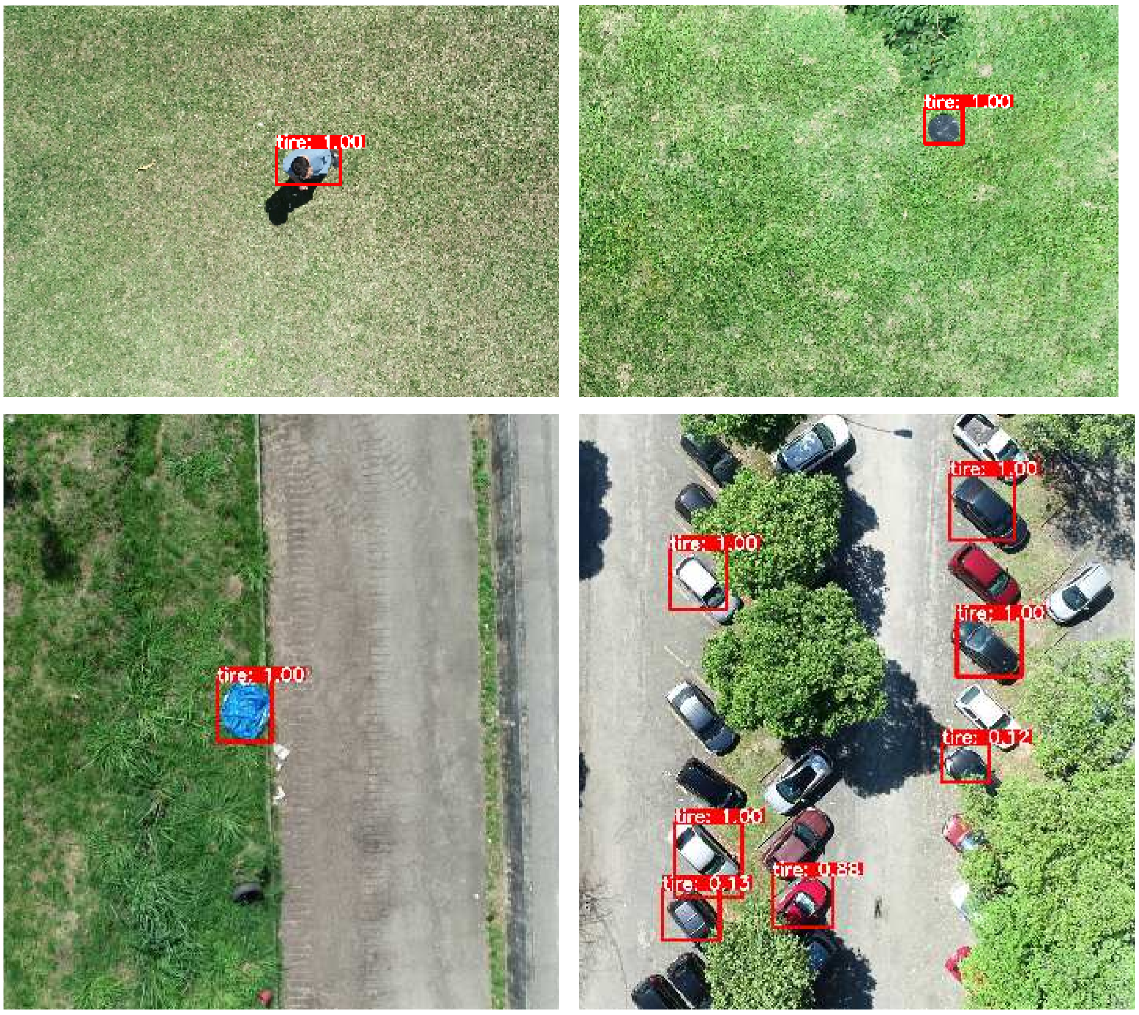


Figure 5.15: Example of objects in the MBG dataset missclassified as tires by the models trained using CEFET dataset (cropped images).

Chapter 6

Conclusions and Future Works

In this work, we started by studying the *Aedes aegypti*, including its biological aspects, the transmitted diseases, and ground sites. We noticed that even though the number of cases of dengue, zika and chikungunya has dropped last year, it is still high. Hence, we proposed a system to automatically detect potential *Aedes aegypti* breeding grounds in order to help health agents to combat its reproduction.

We described the problem of automatic detection of mosquito foci by using computer vision methods. This included a literature review of similar works that use machine learning techniques for this purpose. By the end of this study, we noticed the need for creating a new dataset to train models capable of automatically detecting mosquito sites. Therefore, a new dataset is proposed. In order to collect these data, we used a UAV, to acquire videos with several containers that accumulate clean water in various settings, covering a wide geographic area. Before recording, the camera parameters were manually adjusted, and a calibration procedure was performed. After acquired and rectified, the videos are manually annotated with the Zframer software, allowing to train and test different algorithms for the application of interest.

We then review some object detection algorithms present in the literature. From the classical ones that use sliding windows; up to the most recent ones, which employs deep learning. We performed some experiments using one of the state-of-the-art algorithms for object detection trained with a small dataset. The Faster R-CNN trained with CEFET dataset presented the best result of mAP = 66.68, at IoU = 0.50. The results are considered as promising, indicating that we can employ this methodology to help the government and other institutions in the combat of the *Aedes aegypti*.

As future work, we first intend to expand our dataset by recording new videos at other locations, including more realistic areas, and annotating the maximum possible number of potential mosquito sites. Also, we would like to employ the trained models in real scenarios and analyze their performance. Moreover, we intend to employ shallower feature extractors such as ResNet-18 and ResNet-34. We also intend to exploit and compare other detectors and techniques.

Furthermore, in the future would like to create a final product that can be used as a decision-making support for organizations dedicated to the combat of the *Aedes aegypti*. Our aim is to employ the system for flying over areas and generate heat maps highlighting the places with more risk. We can do so by allying the image detections with the drone telemetry data.

Bibliography

[1] RÜCKERT, C., WEGER-LUCARELLI, J., GARCIA-LUNA, S. M., et al. "Impact of Simultaneous Exposure to Arboviruses on Infection and Transmission by Aedes aegypti Mosquitoes", *Nature Communications*, v. 8, pp. 1–9, may 2017.

[2] MINISTÉRIO DA SAÚDE. "Febre Amarela: Sintomas, Tratamento, Diagnóstico e Prevenção". `http://portalms.saude.gov.br/saude-de-a-z/febre-amarela-sintomas-transmissao-e-prevencao`, 2018. Accessed: 2019-02-05.

[3] PETERSEN, L. R., JAMIESON, D. J., POWERS, A. M., et al. "Zika Virus", *New England Journal of Medicine*, v. 374, n. 16, pp. 1552–1563, apr 2016.

[4] WORLD HEALTH ORGANIZATION. "Zika Virus, Microcephaly and Guillain-Barré Syndrome Situation Report". `http://www.searo.who.int/bhutan/who-zika-situation-report_25-08-2016.pdf`, 2016. Accessed: 2019-02-04.

[5] BHATT, S., GETHING, P. W., BRADY, O. J., et al. "The Global Distribution and Burden of Dengue", *Nature*, v. 496, n. 7446, pp. 504, apr 2013.

[6] WORLD HEALTH ORGANIZATION. "Global Strategy for Dengue Prevention and Control", *Geneva: World Health Organization*, 2012.

[7] NEWTON, E. A., REITER, P. "A Model of the Transmission of Dengue Fever with an Evaluation of the Impact of Ultra-Low Volume (ULV) Insecticide Applications on Dengue Epidemics", *The American Journal of Tropical Medicine and Hygiene*, v. 47, n. 6, pp. 709–720, dec 1992.

[8] ANDRADE, G. S., DIAS, T. M., ALVES, V. C., et al. "Fighting Back Zika, Chikungunya and Dengue: Detection of Mosquito-Breeding Habitats using An Unmanned Aerial Vehicle". IEEE CAS Student Design Competition 2017–2018 Finalist project, may 2018.

[9] TUN-LIN, W., LENHART, A., NAM, V. S., et al. "Reducing Costs and Operational Constraints of Dengue Vector Control by Targeting Productive Breeding Places: a Multi-Country Non-Inferiority Cluster Randomized Trial", *Tropical Medicine & International Health*, v. 14, n. 9, pp. 1143–1153, sep 2009.

[10] LAGROTTA, M. T. F. *Geoprocessamento de Indicadores Entomológicos na Identificação de Áreas, Imóveis e Recipientes "Chaves" no Controle do Aedes Aegypti*. Master's Thesis, Fundação Oswaldo Cruz, 2006.

[11] JANSEN, C. C., BEEBE, N. W. "The Dengue Vector Aedes aegypti: What Comes Next", *Microbes and Infection*, v. 12, n. 4, pp. 272–279, apr 2010.

[12] LIU-HELMERSSON, J., QUAM, M., WILDER-SMITH, A., et al. "Climate Change and Aedes Vectors: 21st Century Projections for Dengue Transmission in Europe", *EBioMedicine*, v. 7, pp. 267–277, may 2016.

[13] BRASIL. INSTITUTO OSWALDO CRUZ. "Dengue, Vírus e Vetor. Curiosidades sobre o Aedes aegypti". `http://www.ioc.fiocruz.br/dengue/textos/curiosidades.html`. Accessed: 2019-02-11.

[14] LAMBRECHTS, L., FAILLOUX, A.-B. "Vector Biology Prospects in Dengue Research", *Memórias do Instituto Oswaldo Cruz*, v. 107, n. 8, pp. 1080–1082, dec 2012.

[15] MONATH, T. P. "Dengue: The Risk to Developed and Developing Countries", *Proceedings of the National Academy of Sciences*, v. 91, n. 7, pp. 2395–2400, mar 1994.

[16] MORIN, C. W., COMRIE, A. C., ERNST, K. "Climate and Dengue Transmission: Evidence and Implications", *Environmental Health Perspectives*, v. 121, n. 11-12, pp. 1264–1272, nov-dec 2013.

[17] HALSTEAD, S. B. "Dengue", *The Lancet*, v. 370, n. 9599, pp. 1644–1652, nov 2007.

[18] FOCKS, D. A., DANIELS, E., HAILE, D. G., et al. "A Simulation Model of the Epidemiology of Urban Dengue Fever: Literature Analysis, Model Development, Preliminary Validation, and Samples of Simulation Results", *The American Journal of Tropical Medicine and Hygiene*, v. 53, n. 5, pp. 489–506, nov 1995.

[19] PATZ, J. A., MARTENS, W., FOCKS, D. A., et al. "Dengue Fever Epidemic Potential as Projected by General Circulation Models of Global Climate

Change", *Environmental Health Perspectives*, v. 106, n. 3, pp. 147–153, mar 1998.

[20] ARCARI, P., TAPPER, N., PFUELLER, S. "Regional Variability in Relationships Between Climate and Dengue/DHF in Indonesia", *Singapore Journal of Tropical Geography*, v. 28, n. 3, pp. 251–272, oct 2007.

[21] RIGAU-PÉREZ, J. G., CLARK, G. G., GUBLER, D. J., et al. "Dengue and Dengue Haemorrhagic fever", *The Lancet*, v. 352, n. 9132, pp. 971–977, sep 1998.

[22] GUZMAN, M. G., HALSTEAD, S. B., ARTSOB, H., et al. "Dengue: a Continuing Global Threat", *Nature Reviews Microbiology*, v. 8, n. 12 Suppl, pp. S7–16, dec 2010.

[23] WORLD HEALTH ORGANIZATION. "Dengue and Severe Dengue". `https://www.who.int/news-room/fact-sheets/detail/dengue-and-severe-dengue`, 2014. Accessed: 2019-02-04.

[24] TEIXEIRA, M. G., COSTA, M. D. C. N., BARRETO, F., et al. "Dengue: Twenty-five Years Since Reemergence in Brazil", *Cadernos de Saúde Pública*, v. 25, n. 1 Suppl, pp. S7–18, jul 2009.

[25] BRASIL. MINISTERIO DA SAUDE. FUNDACAO NACIONAL DE SAUDE. "Dengue: Aspectos Epidemiológicos, Diagnóstico e Tratamento". `http://bvsms.saude.gov.br/bvs/publicacoes/dengue_aspecto_epidemiologicos_diagnostico_tratamento.pdf`, may 2002. Accessed: 2019-02-04.

[26] WATTS, D. M., BURKE, D. S., HARRISON, B. A., et al. "Effect of Temperature on the Vector Efficiency of Aedes aegypti for Dengue 2 Virus", *The American Journal of Tropical Medicine and Hygiene*, v. 36, n. 1, pp. 143–152, jan 1987.

[27] PAN AMERICAN HEALTH ORGANIZATION. WORLD HEALTH ORGANIZATION. BRASIL. "Perguntas e Respostas Sobre o Vírus Zika e suas Consequências". `https://www.paho.org/bra/index.php?option=com_content&view=article&id=5292:perguntas-e-respostas-sobre-o-virus-zika-e-suas-consequencias&Itemid=882`, fev 2017. Accessed: 2019-02-04.

[28] STAPLES, J. E., BREIMAN, R. F., POWERS, A. M. "Chikungunya Fever: An Epidemiological Review of a Re-Emerging Infectious Disease", *Clinical Infectious Diseases*, v. 49, n. 6, pp. 942–948, sep 2009.

[29] WORLD HEALTH ORGANIZATION. “Chikungunya”. `https://www.who.int/news-room/fact-sheets/detail/chikungunya`, 2017. Accessed: 2019-02-05.

[30] CÂMARA, F. P., GOMES, A. L. B. B., CARVALHO, L. M. F. D., et al. “Dynamic Behavior of Sylvatic Yellow Fever in Brazil (1954-2008)”, *Revista da Sociedade Brasileira de Medicina Tropical*, v. 44, n. 3, pp. 297–299, jun 2011.

[31] BRASIL. MINISTÉRIO DA SAÚDE. “Resolução N. 12 de 26 de janeiro de 2017”. `http://www.conass.org.br/wp-content/uploads/2017/02/CIT12-2017.pdf`, jan 2017. Accessed: 2019-02-09.

[32] BRASIL. MINISTÉRIO DA SAÚDE. SECRETARIA DE VIGILÂNCIA EM SAÚDE. DEPARTAMENTO DE VIGILÂNCIA DAS DOENÇAS TRANSMISSÍVEIS. “Levantamento Rápido de Índices para Aedes Aegypti (LIRAa) para vigilância entomológica do Aedes aegypti no Brasil: metodologia para avaliação dos índices de Breteau e Predial e tipo de recipientes”. `http://bvsms.saude.gov.br/bvs/publicacoes/manual_liraa_2013.pdf`, 2013. Accessed: 2019-02-09.

[33] BRASIL. MINISTÉRIO DA SAÚDE. “Diretrizes Nacionais para a Prevenção e Controlede Epidemias de Dengue”. `http://bvsms.saude.gov.br/bvs/publicacoes/diretrizes_nacionais_prevencao_controle_dengue.pdf`, 2009. Accessed: 2019-02-06.

[34] RIO DE JANEIRO. SECRETARIA DE ESTADO DE SAÚDE DO RIO DE JANEIRO. “Informe Epidemiológico 005/2018”. `http://www.riocontradengue.com.br/Publico/MostrarArquivo.aspx?C=1px5ZbL6UIM%3D`, 2018. Accessed: 2019-02-09.

[35] BRASIL. MINISTÉRIO DA SAÚDE. “RN: 131 Municípios em Situação de Alerta ou Risco para Dengue, Zika e Chikungunya”. `http://portalms.saude.gov.br/noticias/agencia-saude/44921-rn-131-municipios-em-situacao-de-alerta-ou-risco-para-dengue-zika-e-chikungunya`, 2018. Accessed: 2019-02-05.

[36] BRASIL. MINISTÉRIO DA SAÚDE. “Mais de Mil Cidades Podem Ter Surto de Dengue, Zika e Chikungunya”. `http://portalms.saude.gov.br/noticias/agencia-saude/43454-brasil-pode-ter-aumento-de-casos-de-dengue-zika-e-chikungunya`, 2018. Accessed: 2019-03-09.

[37] ARAÚJO, H., CARVALHO, D., IOSHINO, R., et al. "Aedes aegypti Control strategies in Brazil: Incorporation of New technologies to Overcome the Persistence of Dengue Epidemics", *Insects*, v. 6, n. 2, pp. 576–594, jun 2015.

[38] WORLD HEALTH ORGANIZATION. "Chemical Methods for the Control of Vectors and Pests of Public Health Importance", *Geneva: World Health Organization*, 1996.

[39] WORLD HEALTH ORGANIZATION. "Global Strategic Framework for Integrated Vector Management". `https://apps.who.int/iris/bitstream/handle/10665/68624/WHO_CDS_CPE_PVC_2004_10.pdf;jsessionid=8EC26B0544508FBA237A98060DC739FC?sequence=1`, 2004. Accessed: 2019-02-06.

[40] BRASIL. "Programa Nacional de Controle da Dengue". `http://bvsms.saude.gov.br/bvs/politicas/programa_nacional_controle_dengue.pdf`, 2018. Accessed: 2018-09-10.

[41] RANKIN, A. L., MATTHIES, L. H., HUERTAS, A. "Daytime Water Detection by Fusing Multiple Cues for Autonomous Off-Road Navigation". In: *24th US Army Science Conference*, pp. 177–184, Orlando, USA, nov 2006.

[42] RANKIN, A., TONISLAV IVANOV, J., BRENNAN, S. "Evaluating the Performance of Unmanned Ground Vehicle Water Detection". In: *10th Performance Metrics for Intelligent Systems Workshop*, pp. 305–311, New York, USA, sep 2010.

[43] RANKIN, A., MATTHIES, L. "Daytime Water Detection Based on Color Variation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 215–221, Taipei, Taiwan, oct 2010.

[44] ZHANG, H., GUO, X., CAO, X. "Water Reflection Detection Using a Flip Invariant Shape Detector". In: *IEEE International Conference on Pattern Recognition (ICPR)*, pp. 633–636, Istanbul, Turkey, aug 2010.

[45] RANKIN, A. L., MATTHIES, L. H., BELLUTTA, P. "Daytime Water Detection Based on Sky Reflections". In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5329–5336, Shanghai, China, may 2011.

[46] SANTANA, P., MENDONCA, R., BARATA, J. "Water Detection with Segmentation Guided Dynamic Texture Recognition". In: *IEEE Interna-*

tional Conference on Robotics and Biomimetics (ROBIO), pp. 1836–1841, Guangzhou, China, dec 2012.

[47] ZHONG, S.-H., LIU, Y., LIU, Y., et al. "Water Reflection Recognition Based on Motion Blur Invariant Moments in Curvelet Space", *IEEE Transactions on Image Processing (TIP)*, v. 22, n. 11, pp. 4301–4313, nov 2013.

[48] AGARWAL, A., CHAUDHURI, U., CHAUDHURI, S., et al. "Detection of Potential Mosquito Breeding Sites Based on Community Sourced Geotagged Images". In: *Geospatial InfoFusion and Video Analytics IV; and Motion Imagery for ISR and Situational Awareness II*, n. 3, p. 90890M, jul 2014.

[49] PRASAD, M. G., CHAKRABORTY, A., CHALASANI, R., et al. "Quadcopter-based Stagnant Water Identification". In: *IEEE National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*, pp. 1–4, Patna, India, dec 2015.

[50] MEHRA, M., BAGRI, A., JIANG, X., et al. "Image Analysis for Identifying Mosquito Breeding Grounds". In: *IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*, pp. 1–6, London, UK, jun 2016.

[51] VC TECHNOLOGY LTD. "Litchi". https://flylitchi.com/, 2018.

[52] DJI. "Phantom 4 Pro Specifications". `https://www.dji.com/phantom-4-pro`. Accessed: 2018-11-05.

[53] GEIGER, A., MOOSMANN, F., CAR, O., et al. "Automatic Camera and Range Sensor Calibration Using a Single Shot". In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3936–3943, Saint Paul, USA, may 2012.

[54] ZHANG, W., JIANG, T., HAN, M. "Digital Camera Calibration Method Based on PhotoModeler". In: *IEEE International Congress on Image and Signal Processing (CISP)*, pp. 1235–1238, Yantai, China, oct 2010.

[55] YUSOFF, A. R., ARIFF, M. F. M., IDRIS, K. M., et al. "Camera Calibration Accuracy at Different UAV Flying Heights", *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)*, v. XLII-2/W3, pp. 595–600, feb 2017.

[56] PÉREZ, M., AGÜERA, F., CARVAJAL, F. "Digital Camera Calibration Using Images Taken from an Unmanned Aerial Vehicle", *International Archives*

of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS), v. XXXVIII-1, n. C22, pp. 167–171, sep 2012.

[57] ZHANG, Z. "A Flexible New Technique for Camera Calibration", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, v. 22, n. 11, pp. 1330–1334, nov 2000.

[58] HARRIS, C., STEPHENS, M. "A Combined Corner and Edge Detector". In: *Alvey Vision Conference*, pp. 147–152, Manchester, UK, sep 1988.

[59] LOWE, D. G. "Distinctive Image Features from Scale-Invariant Keypoints", *International Journal of Computer Vision (IJCV)*, v. 60, n. 2, pp. 91–110, nov 2004.

[60] BRADSKI, G. "The OpenCV Library", *Dr. Dobb's Journal of Software Tools*, 2000.

[61] VEZHNEVETS, V. "OpenCV Calibration Object Detection". `http://graphicon.ru/oldgr/en/research/calibration/opencv.html`, 2005. Accessed: 2019-02-13.

[62] HARTLEY, R., ZISSERMAN, A. *Multiple View Geometry in Computer Vision.* 2 ed. Cambridge, Oxford, UK, Cambridge University Press, 2004.

[63] ZHANG, Z. *A Flexible New Technique for Camera Calibration.* Relatório técnico, Microsoft Research, Redmond, WA, USA, 2008.

[64] IMAGEIO. "imageio: Python Library for Reading and Writing Image Data". `https://github.com/imageio/imageio`, 2014. Accessed: 2018-04-09.

[65] FFMPEG. "H.264 Video Encoding Guide". `https://trac.ffmpeg.org/wiki/Encode/H.264`, 2018. Accessed: 2019-02-13.

[66] WELSTEAD, S. T. *Fractal and Wavelet Image Compression Techniques.* 1 ed. Bellingham, USA, SPIE Press, 1999.

[67] TAIGMAN, Y., YANG, M., RANZATO, M., et al. "DeepFace: Closing the Gap to Human-Level Performance in Face Verification". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1701–1708, Columbus, USA, jun 2014.

[68] SCHROFF, F., KALENICHENKO, D., PHILBIN, J. "FaceNet: A Unified Embedding for Face Recognition and Clustering". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823, Boston, USA, jun 2015.

[69] PASSOS, W. L., QUINTANILHA, I. M., ARAUJO, G. M. “Real-Time Deep-Learning-Based System for Facial Recognition”. In: *Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT)*, pp. 895–899, Campina Grande, Brazil, sep 2018.

[70] CHEN, C., SEFF, A., KORNHAUSER, A., et al. “DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving”. In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 2722–2730, Santiago, Chile, dec 2015.

[71] AFONSO, B. M., CINELLI, L. P., THOMAZ, L. A., et al. “Moving-Camera Video Surveillance in Cluttered Environments Using Deep Features”. In: *IEEE International Conference on Image Processing (ICIP)*, pp. 2296–2300, Athens, Greece, oct 2018.

[72] DALAL, N., TRIGGS, B. “Histograms of Oriented Gradients for Human Detection”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 886–893, San Diego, USA, 2005.

[73] BOSER, B. E., GUYON, I. M., VAPNIK, V. N. “A Training Algorithm for Optimal Margin Classifiers”. In: *Workshop on Computational Learning Theory (COLT)*, pp. 144–152, Pittsburgh, USA, jul 1992.

[74] SERMANET, P., EIGEN, D., ZHANG, X., et al. “OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks”, eprint arXiv:1312.6229v4, feb 2014.

[75] REDMON, J., DIVVALA, S., GIRSHICK, R., et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, Las Vegas, USA, jun 2016.

[76] REN, S., HE, K., GIRSHICK, R., et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, v. 39, n. 6, pp. 1137–1149, jun 2017.

[77] GOODFELLOW, I., BENGIO, Y., COURVILLE, A. *Deep Learning.* Adaptive Computation and Machine Learning Series. Cambridge, England, MIT press, 2016.

[78] LECUN, Y., BOSER, B., DENKER, J. S., et al. “Backpropagation Applied to Handwritten Zip Code Recognition”, *Neural Computation*, v. 1, n. 4, pp. 541–551, dec 1989.

[79] KRIZHEVSKY, A., SUTSKEVER, I., HINTON, G. E. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing systems (NeurIPS)*, pp. 1097–1105, Lake Tahoe, USA, dec 2012.

[80] RUSSAKOVSKY, O., DENG, J., SU, H., et al. "ImageNet Large Scale Visual Recognition Challenge", *International Journal of Computer Vision (IJCV)*, v. 115, n. 3, pp. 211–252, apr 2015.

[81] GIRSHICK, R., DONAHUE, J., DARRELL, T., et al. "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, v. 38, n. 1, pp. 142–158, may 2016.

[82] GIRSHICK, R. "Fast R-CNN". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, Santiago, Chile, dec 2015.

[83] DAI, J., LI, Y., HE, K., et al. "R-FCN: Object Detection via Region-based Fully Convolutional Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 379–387, Barcelona, Spain, dec 2016.

[84] HE, K., GKIOXARI, G., DOLLAR, P., et al. "Mask R-CNN". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, Venice, Italy, oct 2017.

[85] LIU, W., ANGUELOV, D., ERHAN, D., et al. "SSD: Single Shot MultiBox Detector". In: *European Conference on Computer Vision (ECCV)*, pp. 21–37, Amsterdam, Netherlands, oct 2016.

[86] HUANG, J., RATHOD, V., SUN, C., et al. "Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3296–3297, Honolulu, USA, jul 2017.

[87] UIJLINGS, J. R., VAN DE SANDE, K. E., GEVERS, T., et al. "Selective Search for Object Recognition", *International Journal of Computer Vision (IJCV)*, v. 104, n. 2, pp. 154–171, sep 2013.

[88] GIRSHICK, R., DONAHUE, J., DARRELL, T., et al. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 580–587, Columbus, USA, jun 2014.

[89] SIMONYAN, K., ZISSERMAN, A. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations (ICLR)*, pp. 1–14, San Diego, USA, may 2015.

[90] HE, K., ZHANG, X., REN, S., et al. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: *European Conference on Computer Vision (ECCV)*, pp. 346–361, Zurich, Germany, sep 2014.

[91] SHELHAMER, E., LONG, J., DARRELL, T. "Fully Convolutional Networks for Semantic Segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, v. 39, n. 4, pp. 640–651, apr 2017.

[92] QIAN, N. "On The Momentum Term in Gradient Descent Learning Algorithms", *Neural Networks*, v. 12, n. 1, pp. 145–151, jan 1999.

[93] EVERINGHAM, M., VAN GOOL, L., WILLIAMS, C. K. I., et al. "The Pascal Visual Object Classes (VOC) Challenge", *International Journal of Computer Vision (IJCV)*, v. 88, n. 2, pp. 303–338, jun 2010.

[94] JACCARD, P. "The Distribution of the Flora in the Alpine Zone", *New Phytologist*, v. 11, n. 2, pp. 37–50, feb 1912.

[95] LIN, T., MAIRE, M., BELONGIE, S. J., et al. "Microsoft COCO: Common Objects in Context", eprint arXiv:1405.0312v3, feb 2015.

[96] MASSA, F., GIRSHICK, R. "maskrcnn-benchmark: Fast, Modular Reference Implementation of Instance Segmentation and Object Detection algorithms in PyTorch". `https://github.com/facebookresearch/maskrcnn-benchmark`, 2018. Accessed: 2019-02-04.

[97] HE, K., ZHANG, X., REN, S., et al. "Deep Residual Learning for Image Recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, USA, jun 2016.

[98] SHRIVASTAVA, A., SUKTHANKAR, R., MALIK, J., et al. "Beyond Skip Connections: Top-Down Modulation for Object Detection", sep 2017. eprint arXiv:1612.06851v2.

[99] LIN, T.-Y., DOLLAR, P., GIRSHICK, R., et al. "Feature Pyramid Networks for Object Detection". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944, Honolulu, USA, jul 2017.

[100] GIRSHICK, R., IANDOLA, F., DARRELL, T., et al. "Deformable Part Models are Convolutional Neural Networks". In: *IEEE Conference on*

Computer Vision and Pattern Recognition (CVPR), pp. 437–446, Boston, USA, jun 2015.