MS-CALID: a Context-Aware Local Infrastructure Detection and Monitoring System

Building a supervised software system for understanding the local environment based on aerial image processing.

Rares FOLEA
Computer Science & Engineering Department
Faculty Of Automatic Control And Computers
University Politehnica Of Bucharest
Bucharest, Romania
rares.folea@stud.acs.upb.ro

Andrei VATAVU
Computer Science & Engineering Department
Faculty Of Automatic Control And Computers
University Politehnica Of Bucharest
Bucharest, Romania
andrei.vatavu@stud.acs.upb.ro

***Abstract*—This paper presents the work for the Computer Vision Project during the 2020 Computer Vision Class taken in Faculty Of Automatic Control And Computers, University Politehnica Of Bucharest. We aim to present an automated system that is capable of detecting in a supervised manner the local infrastructure of a specific geographic region.**

***Index Terms*—computer vision, image segmentation, road detection, automatic map generation, YOLO, CVAT, convolutional networks**

CONTENTS

Figure 1: Geographical position of Slanic Moldova (Romania)

I. INTRODUCTION

MS-CALID aims to be an automated software system for local infrastructure detection and monitoring built on top of YOLOv4 [2], a tool for Optimal Speed and Accuracy of Object Detection, which can provides insights on the local venues and groundwork and real-time information about local activities and live status information regarding the local infrastructure. We have manually trained MS-CALID on a custom labelled dataset consisting of hundreds of aerial frames obtained from the region of Slanic-Moldova, Bacau, Romania.

We have open-sourced our code and dataset at:

`https://github.com/raresraf/MS-CALID`

The current model has been trained on a hundred manually labelled frames extracted from an 5-minute aerial video taken at Slanic-Moldova, a town in Bacau County, Moldova, Romania. What MS-CALID tries to achieve is identification of the local infrastructure (reports on the number of households, houses, administrative buildings,

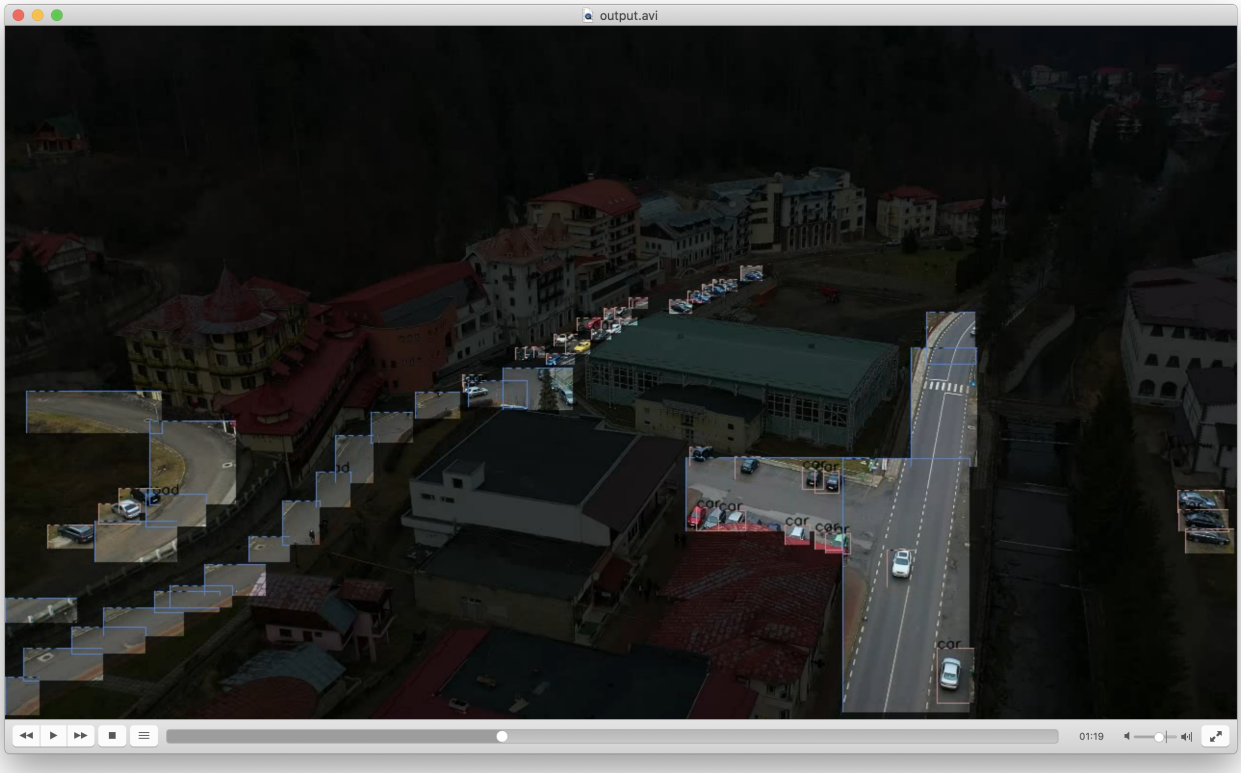

Figure 2: Example of a frame identifying road, alley and cars.

hotels, religious entities, public roads, total number of cars, alleys and parks, vegetation and other features) and offer live monitoring solutions (such as notifications on traffic values).

II. Motivation

A. Authors' moon-shot ideas for Computer Vision

In the first weeks of 2020 Computer Vision Class we were asked by prof. Leordeanu to propose high-level view idea over a new application of Computer Vision in real world. Looking back, we approached the problems we defined there in this work.

1) raresfolea@'s: Generally speaking, **Computer Vision** is a discipline that aims to obtain a *semantic understanding from digital images* for obtaining a specific knowledge that allows *computers to operate and take decisions based on the acquired observations*. One impactful application of Computer Vision is **road-segmentation**, a well-studied, yet unsolved, problem that is encountered both in road-level images, *required by autonomous driving applications, where the computers should reason regarding the road ahead* or *overview photos, taken from satellites, with the objective of obtaining a digital map of the location*. While there are a bunch of notable results for both areas, we want to predominantly analyze satellites images with the objective of creating digital maps of the network of roads. The main difference with general case when analysing urban roads, mostly on flat areas and with a high occurrence of asphalt zones, what we aim to analyze is Mountain Trails and Paths, all obtained from operating on satellite images. Most of the Trails are in zones with high density of forests, high peaks or rocky ground. There are little to none paths asphalted.

How it connects to this project? Road detection is part of local infrastructure detection and looking at the amount of traffic at a particular moment in a specific zone represents context-aware monitoring.

2) andreivatavu@'s: In the last decades, the number of people suffering from obesity and overweight has increased progressively, the main cause being the maintenance of an unhealthy diet. The consequences of this are both physical and mental, but the biggest problem is that this category of people is more prone to chronic diseases such as heart disease, respiratory disease, and cancer. Therefore, we need tools that provide nutritional information on food consumed, based on which we can improve the control of food consumption and we can treat people with nutritional problems.

How it connects to this project? On a general view:

improving the quality of life via live information about the surrounding environment. More particular an integration, advances recommendation systems for restaurants and places to eat healthy.

B. On the nature of duality

This subparagraph is healivy inspired from author's research topics for 2020 in AI-Masters Faculty Of Automatic Control And Computers, University Politehnica Of Bucharest.

The primary structure of the universe is shaped as a perfect balance, that, according to the cultural background of the historical religious sphere, dictates the hidden meaning of everything in the world. Thus, starting from the bottom of the humanity and rebuilding a sense of the being itself, we discover that both man and woman are part of this duality. Initially seen as a whole, the myth of the *androgyne* indicates the subsequent separation of the two autonomic genres of human nature. Furthermore, the nature itself follows the rules of the **duality**, as everything can be divided in two different categories: *appolonian and dionysian*. The first term origins from the Name of the God Apollo and refers to the quite, peaceful and balanced nature whereas the dionysian is linked with the God Dyonis, that, as worshipped by the Bachhae, symbolises the ecstasy and tumult of life (for instance Nietzsche points out the fact that he himself was a dionysiac). Following the same philosophical direction, the greatest and most debatable duality, in the sense of abstraction, is that of the *good and the bad*, *the sacred and the profane* as it is explained in the work of Mircea Eliade[1]: *The Sacred and The Profane*[6].

We will now extend the nature of duality in our *world of computing and information technology*, in terms of **static** *or/and* **dynamic** analysis: for Computer Vision: *images* **AND** *videos*, the evolution of scenes under the evolution of time.

The generic problem of *computer vision*: having an algorithmic that can interfere an understanding similar to our knowledge we obtain during the act of seeing. Serious hard questions arise from this goal, such as finding the true meaning of what vision is. Proven to be a question with deep philosophical consequences, it has been widely studied with ambitious goals of formulating a coherent theory on what vision is and how it can be automatically inherited by computer using dedicated software built on the clear algorithmic solution to the vision problem. The reality has proven to be far more difficult and complex than it has been initially thought as yet, the problem of vision is far away from being solved. Modern approach offers decent results for systems that use advanced machine learning techniques on data that are similar to the environment they have been trained on. However, the view of the worlds might be totally different and using systems that have been trained on a particular world might under-perform on different scenarios. Moreover, best computer vision systems decided to switch from a traditional approach on analyzing bare images individually, having an independent-frame analysis, into analysing videos as a whole, with time causality dependencies between frames. This reaches our point: the field of computer vision managed to evolve and obtain better systems by taking the **advantages of duality**; for such systems, *images* **AND** *videos* are used together for enriching the understanding of vision, linking the evolution of scenes with the evolution of time. Just by taking individual images and having a solo deep analysis is valuable, but by doing so, the system loses the context[2] which is undoubtedly valuable for the general awareness.

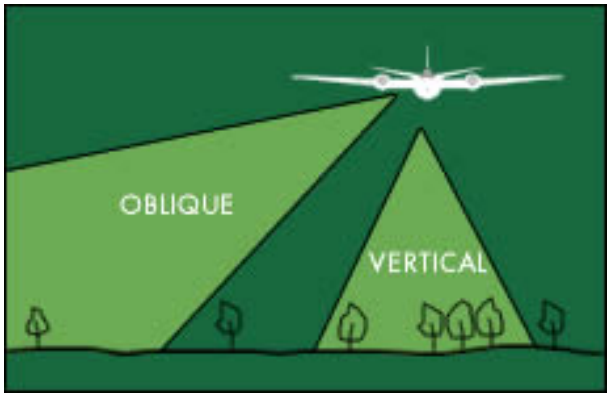

Figure 3: The differences in Vertical and Oblique Aerial Photography: copyrights for National Collection of Aerial Photography.

III. Datasets

The current work has been done during the 2020 Computer Vision Class taken in Faculty Of Automatic Control And Computers, University Politehnica Of Bucharest. For this paper, we have used the datasets provided by [8]. They consist in a stack of 3-5 minutes videos taken around Slanic-Moldova, a town in Bacau County, Moldova, Romania. The city is served by the national road DN12B (road which we will try to detect), which leads to the northeast to Targu Ocna, where it ends in DN12A.

The Videos represents aerial tours of the town of Slanic Moldova taken from a drone on an oblique perspective.

Having images taken from an oblique perspective makes the measurements based on the videos more difficult than it would have been given vertical-taken videos. There are numerous advanced geometry tricks for workarounds this [9].

We have used two movies from our dataset for training(DJI_0956.MP4) and testing(DJI_0957.MP4). The videos captions makes a good overall description of the

[1]Mircea Eliade was a Romanian historian of religion, fiction writer, philosopher, and professor at the University of Chicago.

[2]Which we will further refer as dynamic context.

Figure 4: Sample frame from the movie DJI_0956.MP4.

local environment. It walks around the town in seek of buildings, houses, hotels, culture or religious venues, forest, road, vegetation. Thus, they provide a good input source for our system, that aims to detect the local infrastructure and monitor local resources, such as traffic values or vegetation status.

Understanding the regional area from where is your training dataset represents an important aspect. You cannot train a system on a very specific region and expect it to perform well in different environments. On the other hand, having it trained regionally should provide satisfactory results when deploying it and asking it to analyze real-time scenes which, even though were not provided exactly in the training dataset, are similar enough for the system to draw conclusion.

The dataset also consists of flight logs recorded by the drone. They contain the GPS positions, velocity at any given datapoint, and additional information from which our system can draw context conclusions, such as geolocation of the analyzed zone or the aerial perspective of the taken image.

Moreover, being context-aware when designing a system such as MS-CALID represents an important condition: there is no point in monitoring and trying to identify things that are not specific to the region. Simplifying the conceptual model of the system, adapted to local needs is essential for enriching the results.

A. Labelling frames

Using CVAT, an interactive video and image annotation tool for Computer Vision, we have crafted our own training dataset with manually labelled frames from the training movie(DJI_0956.MP4).

We have a hundred of labelled frames from the video above enclosed in the GitHub repository of this project.

The dataset structure is as follows:

```
.
|--- obj.data
|--- obj.names
|--- obj_train_data
|.. |--- frame{NNN}.jpg
|.. |--- frame{NNN}.txt
```

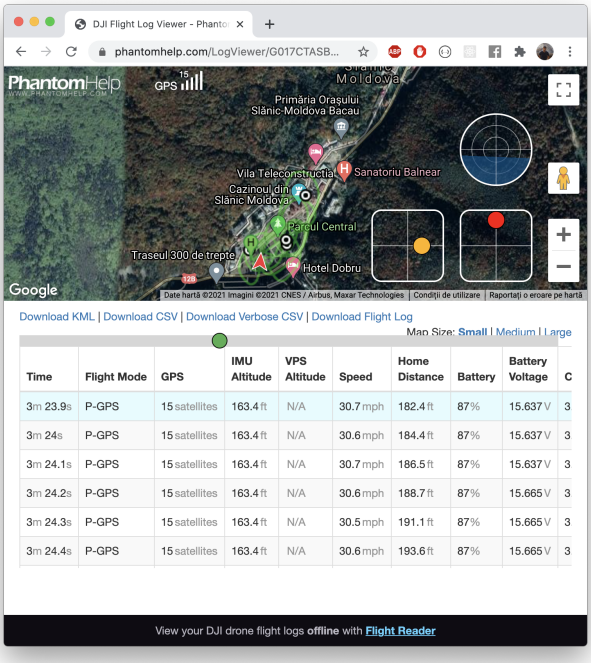

Figure 5: https://www.phantomhelp.com/ provides a good resource for loading the logfiles and presents a nice UI of the flight records. The site provides the service Flight Reader to decrypt and view the flight data inside of the DJI TXT flight logs, Litchi CSV flight logs, or Map Pilot CSV flight logs from DJI drones.

```
...
|-- train.txt
```

IV. Object Recognition

YOLO (You Only Look Once) is a state of the art, real-time object detection. The working principle of Yolo is that it applies a single neural network to the image. The neural network divides the image into regions and predicts bounding boxes and probabilities for each region, after that these bounding boxes are weighted by the predicted probabilities. The author of Yolo has released 3 versions of this state of the art object detection:

- You Only Look Once: Unified Realt-Time Object Detection [11]
- YOLO9000: Better, Faster, Stronger [12]
- YOLOv3: An Incremental Improvement [13]

Two years after YOLOv3, within a period of a couple of months, YOLOv4 [3] and YOLOv5 were released by two different authors, but neither of them by the original author of YOLO. After researching both of them, we decided to use YOLOv4 because it is based on YOLOv3 and has a paper in which are detailed the obtained results. YOLOv5 is different from the other versions because it is not based on the darknet neural network, it is a PyTorch implementation and was released without a paper to sustain the author's claims regarding its performance.

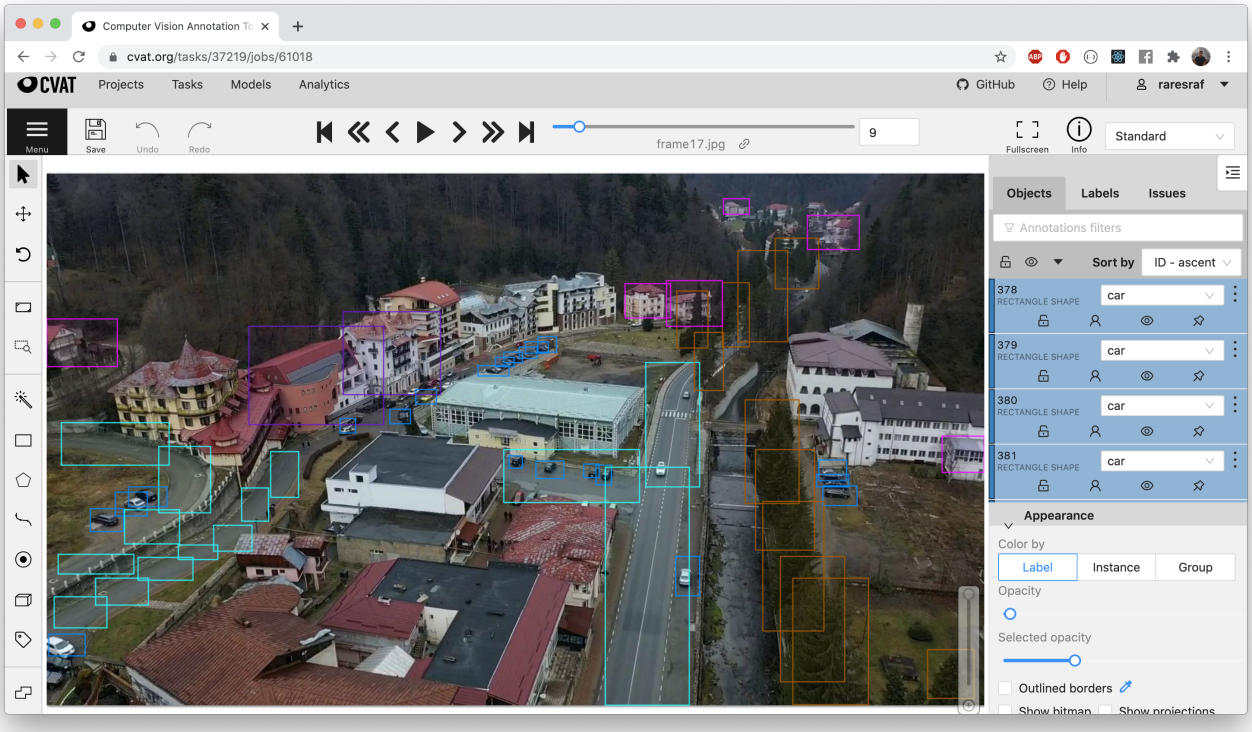

Figure 6: Example of a labelled frame in the UI of the CVAT tool.

V. Object Tracking

This work also presents an application of Object Tracking using build-in OpenCV tracking tools.

VI. Tech stack

Our tech stack contains of (but not limited to) various software Computer Vision specific technologies, such as:

- YOLOv4[1]: neural networks for object detection.
- CVAT[4]: an interactive video and image annotation tool for computer vision.
- opencv-python library: pre-built CPU-only OpenCV packages for Python, used for parsing images and videos in this computer vision problem [10].
- ffmpeg: A complete, cross-platform solution to record, convert and stream audio and video [7].
- Numpy: Powerful n-dimensional arrays. Numerical computing tools. Interoperable. Performant. Open source [5].
- sklearn: Machine learning in Python [14]. Also used for the framework containing neural networks
- C, CUDA and Python3 as most used programming languages.

VII. Training

You only look once (YOLO) is a state-of-the-art, real-time object detection system. Having prepared the dataset in the YOLO specific format, we proceeded to train our MS-CALID network based on YOLOv4. Using the public documentation of the neural network, we trained YOLOv4 for object detection.

YOLO training requires few parameters definition and allows custom architecture for the convolutional networks. We have used a 6k iterations, that took over 24 hours to train on a Nvidia GeForce GTX 1060, 6GB memory, with a learning rate $1e-3$. YOLOv4 is considered to be an fast and accurate network architecture.

Launching a YOLO training using the open-source code can be done using:

```
.\darknet.exe detector train
.\CV-project\obj.data
.\CV-project\yolov4-CV.cfg
.\CV-project\backup\yolov4-CV_last.weights
```

We choose to use YOLOv4 [3] to train a model on our custom dataset. The first step was to prepare the dataset for YOLOv4. For every image, YOLO needs a file with the same name but with the extension ".txt" which contains information about the bounding boxes from that picture. The file must describe each bounding box on a separate line. A line inside this file has the following format:

$$object - class\ x\_center\ y\_center\ width\ height$$

Where:

Figure 7: Example of a generated test frame.

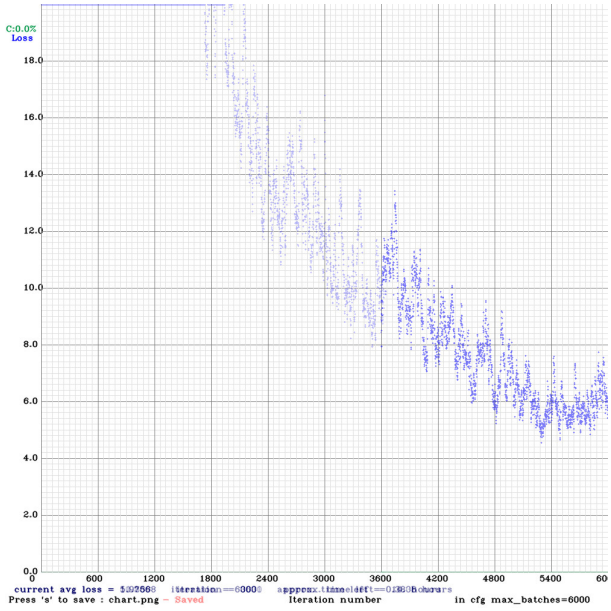

Figure 8: Plot consisting of the representation of the training loss with respect to the number of training iterations.

- $object - class$ - the id of the class [0, number of classes - 1]
- x_center, y_center - (x, y) coordinates of the center of the bounding box relative to width and height of image (float values between 0, 1)
- $width\ height$ - width and height of the bounding box relative to width and height of image (float values between 0, 1)

Lastly, we adjusted the YOLO parameters according to the number of classes from the dataset and the dimension of the training set.

VIII. Testing

We have used the testing(DJI_0957.MP4) movie in order to check that the network is able to adapt to similar situations, rather than only relying on the initially learned frames from the training dataset.

IX. Conclusions

YOLO provides Fast and Real Time Detection and High accuracy for the scenario that we trained the system on.

Even though in this paper we have used the state-of-the-art techniques in object tracking and recognition, we are far away from solving the problem of Computer Vision. There are easy cases in which the system easily gets fooled.

Follow **MS-CALID** development:

`https://github.com/raresraf/MS-CALID/`

References

[1] AlexeyAB. Darknet network: https://github.com/alexeyab/darknet.

[2] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. 2020.
[4] cvat. https://github.com/openvinotoolkit/cvat.
[5] NumPy Official Public Documentation. https://numpy.org/.
[6] Mircea Eliade. *The sacred and the profane: The nature of religion*, volume 144. Houghton Mifflin Harcourt, 1959.
[7] ffmpeg Official Public Documentation. https://ffmpeg.org/.
[8] M. Leordeanu. Computer vision: Homework and project materials.
[9] Massimiliano Molinari, Stefano Medda, and Samir Villani. Vertical measurements in oblique aerial imagery. *ISPRS International Journal of Geo-Information*, 3(3):914–928, 2014.
[10] opencv Official Public Documentation. https://pypi.org/project/opencv-python/.
[11] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. 2016.
[12] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. 2016.
[13] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. 2018.
[14] scikit Official Public Documentation. https://scikit-learn.org/stable/.