Skolkovo Institute of Science and Technology

Mathematical Modelling and Analysis of Intelligent Monitoring Platform for Precision Agriculture

Doctoral Thesis
by

Alexander Menshchikov

Doctoral Program in Computational and Data Science and Engineering

Supervisor
Assistant Professor, Andrey Somov

Moscow - 2020

I hereby declare that the work presented in this thesis was carried out by myself at Skolkovo Institute of Science and Technology, Moscow, except where due acknowledgement is made, and has not been submitted for any other degree.

Candidate (Alexander Menshchikov)

Supervisor (Prof. Andrey Somov)

Mathematical Modelling and Analysis of Intelligent Monitoring Platform for Precision Agriculture

by

Alexander Menshchikov

Friday 2nd October, 2020 04:18

Submitted to the Skoltech Computational and Data Science and Engineering
on July 2020, in partial fulfillment of the requirements for the
Doctoral Program in Computational and Data Science and Engineering

Abstract

The population of humanity is continuously growing. As a consequence, it leads to the growth of human needs in food. Therefore there is increasing demand in chemicals for fertilization, which affects the environment. To sustain the growing population with sufficient food, we will need to use even more fertilizers to cover a greater area. It is expensive and not environmentally friendly. Aerial monitoring opens vast possibilities for precise plant treatment to minimize chemical inputs for bigger farm fields. However, monitoring of bigger areas requires bigger memory storage on board of the monitoring platform as well as bigger computation capabilities for post-processing of the collected data. That is why precision agriculture requires new methods for plant growth monitoring. These new strategies should allow to cover greater areas, work under various weather conditions, and exclude human labor from this activity.

To address these problems, one can rely on modern approaches to monitoring. They include satellite imagery, aerial surveillance by drones, and tracking by Wireless Sensor Networks (WSN). The high-resolution satellite imagery is expensive, doesn't resolve individual plants, and is not allowed by demand. The WSN has smaller coverage and implies the active engagement of humans into the maintenance. Unmanned Aerial Vehicles (UAVs) are cheap, can operate by request, and collect the data in high resolution. It is an appropriate platform to solve the task.

This work is devoted to mathematical modeling and analysis of the electrical Vertical Take-Off and Landing (eVTOL) UAV, empowered by Machine Learning capabilities for semantic segmentation and classification of objects of interest in the real-time onboard. The primary application of the vehicle belongs to the area of smart agriculture - the reconnaissance of harmful plants like hogweed (*lat. Heracleum*). Such an application requires a comprehensive enhancement of UAV range and agility, intelligence, and computer vision capabilities. These challenges are addressed in this work by the development of the morphing wing, and optimization of the semantic segmentation neural network to real-time processing of the video stream onboard of the low-power embedded system.

The methodology behind the morphing wing development includes the Compu-

tational Fluid Dynamics (CFD) investigation and the wind tunnel investigation of the wing's prototype in relevant flight conditions. A primary measurement technique in the experiment is the Particle Image Velocimetry (PIV). The prototype showed $\sim$5% of maximum lift-to-drag ration enhancement and also increased stalling angle for $\sim 4^{\circ}$ for all the flight regimes. It means expanding the range, controllability, and agility. It also means enhancement of the battery charge for at least 3%. This energy could be redistributed for performing the on board data processing.

The computer vision task was addressed by optimization of Fully Convolutional Neural Network (FCNN) architecture for operation on embedded Visual Processing Units (VPU) and Graphical Processing Units (GPU). It resulted in a frame rate of up to 0.64 frames per second (FPS) in real-time operation for a high definition video stream. The ultmiate system is deployed and investigated in real conditions.

Publications

Journal Publications

1. D. Shadrin, **A. Menshchikov**[1], A. Somov, G. Bornemann, J. Hauslage, and M. Fedorov. Enabling precision agriculture through embedded sensing with artificial intelligence. *IEEE Transactions on Instrumentation and Measurement*, pages 1–1, 2019a. ISSN 1557-9662. doi:10.1109/TIM.2019.2947125

2. D. Shadrin, **A. Menshchikov**[1], D. Ermilov, and A. Somov. Designing future precision agriculture: Detection of seeds germination using artificial intelligence on a low-power embedded system. *IEEE Sensors Journal*, 19(23): 11573–11582, Dec 2019b. ISSN 2379-9153. doi:10.1109/JSEN.2019.2935812

3. **A. Menshchikov** and A. Somov. Morphing wing with compliant aileron and slat for unmanned aerial vehicles. *Physics of Fluids*, 31(3):037105, 2019b. doi:10.1063/1.5086976. URL `https://doi.org/10.1063/1.5086976`

4. **A. Menshchikov**. Development of adaptive wing with double hinge aileron for unmanned aerial vehicles. *Austrian Journal of Natural and Technical Science*, pages 150–159, June 2018b. doi:10.29013/ajt-18-5.6-33-40

Conference Proceedings

1. **A. Menshchikov**, D. Lopatkin, E. Tsykunov, D. Tsetserukou, and A. Somov. Realizing body-machine interface for quadrotor control through kalman filters and recurrent neural network. In *IEEE 25th International Conference on Emerging Technologies and Factory Automation (IEEE ETFA 2020)*, September 2020, (in press)

2. **A. Menshchikov**, D. Ermilov, I. Dranitsky, L. Kupchenko, M. Panov, M. Fedorov, and A. Somov. Data-driven body-machine interface for drone intuitive control through voice and gestures. In *IECON 2019 - 45th Annual Conference*

[1]corresponding author

of the IEEE Industrial Electronics Society, volume 1, pages 5602–5609, Oct 2019b. doi:10.1109/IECON.2019.8926635

3. V. Prutyanov, N. Melentev, D. Lopatkin, **A. Menshchikov**, and A. Somov. Developing iot devices empowered by artificial intelligence: Experimental study. In *2019 Global IoT Summit (GIoTS)*, pages 1–6, June 2019. doi:10.1109/GIOTS.2019.8766355

4. **A. Menshchikov** and A. Somov. Mixed reality glasses: Low-power iot system for digital augmentation of video stream in visual recognition applications. In *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*, pages 1–8, June 2018. doi:10.1109/SIES.2018.8442093

Posters

1. **A. Menshchikov**, I. Dranitsky, D. Ermilov, L. Kupchenko, M. Panov, A. Somov, and M. Fedorov. Data-driven body-machine interface for drone intuitive control through voice and gestures. In *Machine Learning Summer School (MLSS-2019)*, September 2019a

2. **A. Menshchikov** and A. Somov. Morphing wing with compliant trailing and leading edges for unmanned aerial vehicles. In *Topology Optimization - Theory, Methods and Applications*, June 2019a

3. **A. Menshchikov**, D. Shadrin, A. Somov, and M. Fedorov. Artificial intelligence in distributed autonomous real time systems for precision agriculture. In *MoNeTec-2018*, October 2018b

4. **A. Menshchikov**, I. Dranitsky, D. Ermilov, L. Kupchenko, M. Panov, A. Somov, and M. Fedorov. Wilco: Drone instantaneous control by voice and gestures. In *IEEE Industrial Instrumentation and Measurement Technologies Conference*, May 2018a

Patents

1. **A. Menshchikov**. Ru 2018618762; "airflow 2.0: 2d computational fluid dynamics simulator and optimizer of airfoils". 2018a

Dedicated to my wife, Maria.

Acknowledgments

Let me thank all my supporters. First of all, I want to thank my Thesis Supervisor Andrey Somov for his passion for my research topic and his deliberate help in my personal development in academia. I also want to thank professor Maxim Fedorov for fruitful discussions and directing me to the proper scientific area. Maxim Panov and Dmitry Ermilov for making drones fly with intelligence. Maria Pukalchik for her unimpeachable expertise in the field of agriculture.

I also want to thank Daniil Lopatkin and Viktor Prutyanov for collaboration, working on research, writing papers, and for the great opportunity of being a co-advisor for these two brilliant masters students.

I want to thank the foundation "Innovations Promotions Fund" and "UMNIK" scientific grant program for young specialists for supporting part of the current research devoted to the Morphing Wing. Grant number is № 9189ГУ/2015.

I show appreciation to the Autonomous Systems Laboratory from the Department of Aeromechanics and Flight Engineering of Moscow Institute of Physics and Technology for the opportunity to perform experiments in their wind tunnel.

And finally, I want to express my gratitude to my colleague and friend, Dmitrii Shadrin, for fruitful collaboration, significant experience of doing science on the cutting edge, and his outstanding capability to help at any time.

Contents

List of Figures

List of Tables

"Before beginning, plan carefully"

Cicero

Chapter 1

Introduction

Let me introduce to the topic of my Ph.D. work at the Skolkovo Institute of Science and Technology. The research topic belongs to the area of Precision Agriculture. Nowadays, there is a need for the platform, which is capable of covering larger areas and perform data-intensive computations in real-time on board. That is why the research aims to mathematical modeling and evaluation of different aspects of the electric Vertical Take-Off and Landing (eVTOL) Unmanned Aerial Vehicle (UAV), empowered by Artificial Intelligence (AI) system deployed on board for effective detection of harmful or ecology-unfriendly plants with the subsequent approbation of the developed approach in real-world conditions. Even though modern UAVs have high-quality multispectral imaging capabilities, the processing of these images is still a challenging task for machine vision algorithms due to the complex structure of plants' topology and broad variety types of background. Partially this problem can be solved by using Fully Convolutional Neural Network (FCNN) for semantic segmentation. However, due to the complexity of the proposed problem, existing algorithms cannot adequately perform semantic segmentation of diverse plants in field conditions. Furthermore, there is still a problem with optimizing high-performance computational algorithms for mobile platforms, and resolving this new fundamental approach should be investigated and implemented.

Currently, there are numerous techniques for aerial imagery segmentation. For example, Haug et al. [2014] proposes to use Random Forest (RF) classifier to estimate crop and weed certainty based an overlapping neighborhood around sparse

pixel positions. At the same time, in Lottes et al. [2016], the authors improved the method. They added a relative plant arrangement based on Markov Random Field (MRF) optimization on exploiting the topological relationships between key-points. Potena et al. [2016] uses a cascade of Convolutional Neural Network (CNN) for crop and weed classification. In the proposed method the primary CNN detects plants, then the output of this network is further classified by a deeper CNN. In a similar research Fawakherji et al. [2019] the authors attempt to overcome the generalization limitations of CNN when the dataset has a few pixel-wise annotated pictures. They also use a cascade of CNNs, where the primary network performs binary segmentation between vegetation and soil terrain, while the secondary network performs a blob-wise classification. Some authors apply Fully Convolutional Neural Network (FCNN) for semantic segmentation of aerial imagery collected by RGB camera Badrinarayanan et al. [2017] as well as multispectral camera Sa et al. [2018]. In similar fashion, Lottes et al. [2018] propose an encoder-decoder network architecture that includes spatial information by considering image sequences. Milioto et al. [2018] uses a similar structure: they propose to create auxiliary input channels, which include various vegetation indices, like NDVI, ExG, etc. These approaches are usually applied to the processing of the data collected onboard the UAV and further transferred to the computer for processing. Moreover, all these researches only state that their approaches could be used in real-time processing onboard the UAV; however, none supports it by test flight data.

That is why the first part of the study creates a generic embedded artificial intelligence system for large-scale detection and monitoring of plants in real-time. Direct monitoring of harmful plants using computer vision methods is a unique study that relies on the latest achievements in the field of AI. The system performs the detection and segmentation of the hotspot distribution areas of harmful plants onboard. It sends the plants' position to the operator, therefore eliminating the bottleneck of the traditional earth observation pipeline. It usually implies data collection onboard, uploading to the computer, and further time-consuming post-processing. Even though it allows us to receive exhaustive multimodal data, it is often overshot for specific missions, like certain breeds of plant classification.

The proposed computer vision system was subsequently tested in various scenarios, including the climate chamber, greenhouse, and aerial monitoring platform. The development of such an intelligent aerial robotics platform also implies the flying UAV itself.

Therefore, the second part of the study is devoted to the mathematical modeling of eVTOL aircraft with Morphing Wing (MW). It focuses on aerodynamic design, development of controller, and estimator. The vehicle has capabilities and a fixed-wing to perform fast, long-range flight with low power consumption. Due to higher dynamics of the atmosphere at low altitudes and due to mission requirements, the vehicle operates under rapidly changing flight regimes. However, aerodynamic surfaces of such type of aircraft usually stay suboptimal in a rapidly changing environment, which leads to losses due to reduced lift-to-drag ratio and control authority. The proposed aircraft performs the flight in multiple rapidly changing regimes. It motivates the search for new means of physical control of the vehicle dynamics.

The Morphing Wing (MW) is the right option for this role. There are multiple approaches to improve the aerodynamic performance of the aircraft using MW. They usually rely on the optimization of various geometric parameters of the wing to better fit specific conditions. That could be an adaptive trailing edge Kota et al. [2009] for saving fuel during take-off and landing. Variable wing span Bashir and Rajendran [2018] and sweep angle Di Luca et al. [2017] apply to multipurpose UAVs for flight in a vast range of flight regimes. Variable dihedral, along with span-wise bending, allows improving lateral stability. Morphing winglets Spivey and Suh [2018] enable the control of aircraft under the smooth turns more effectively. All these research works at most develop morphing structures for large scale aircraft, which operate under large values of Reynolds number. However, the medium-sized UAV, which usually engaged in agricultural monitoring, operates under low Reynolds number. This area of aerodynamics is poorly explored, and there is a need for the development of the morphing structure for a particular application in smart agriculture.

Since MW has the ultimate capability to stay optimal for many flight regimes. This promising technology could significantly improve the performance and maneuverability of the aircraft during the flight. Therefore I assess the performance of the

wing with the traditional and morphing mechanization using computer simulation followed by the experiments in the wind tunnel environment. This work also provides the morphing wing's design: the compliant rib, made from a combination of hard and soft plastic and the skin made from the elastomer. The servo actuators drive the MW. The research demonstrates that the application of the morphing wing mechanization could improve the lift-to-drag ratio by 15% for specific regimes, thereby improving the range and time of operation, which is crucial for eVTOL aircraft.

The study demonstrates the complex approach, which eliminates two key bottlenecks of the modern earth observation drones, used in the precision agriculture domain. (i) Elimination of the data processing bottleneck through the development of the payload with edge computing capabilities. (ii) Prolongation of range and improvement of controllability through numerical and wind tunnel investigation of new physical means of control (the Morphing Wing). This task also implies the derivation of control and estimation algorithms for eVTOL UAV with MW. Even though MW and proposed computer vision system belongs to different research areas, they have essential feature - they are inherent parts of UAV, the platform with limited power. It means that the electric power, saved due to application of the MW, could be applied for the operation of the computer vision system and vice-versa.

1.1 Thesis Structure

The Ph.D. Thesis has the following composition:

Chapter 1 - Introduction The first chapter includes general information on the topic of the dissertation. It contains brief Overview, the Scope, the Motivation, the Goal and Objectives of the Thesis. It also describes Methods and Approaches of the investigation in brief.

Chapter 2 - State-of-the-art The second chapter contains the literature overview. It has three subsections; they relate to edge computing and computer vision with application in precise agriculture, mathematical modeling of eVTOL aircraft, and aerodynamics of the MW, respectively.

Chapter 3 - Methodology The third chapter describes the research behind every thesis objective. It includes the following subtopics:

- Edge Computing: hardware and optimization techniques for particular task:
  - Seed germination task;
  - Tomatoes growth task;
  - Hogweed localization task;
- Description of the testbed for data collection and data processing;
- Aerodynamic design of the drone;
- Mathematical modeling, numerical investigation and wind tunnel experiments of the MW;

Chapter 4 - Conclusion In the last chapter, we discuss the results obtained during the implementation of all the abovementioned thesis objectives.

1.2 The Scope of the Thesis

Modern precision agriculture requires precise and effective monitoring with high resolution. UAVs could be effectively used to address this challenge since they are low cost, cover large areas, and, unlike ground vehicles, do not have a mechanical impact on the field. Unfortunately, modern UAVs, used in precision agriculture, have limited capabilities. They are focused on the traditional approach to fertilization with uniform application of plant solutions to the field. That is why this study is devoted to mathematical modeling and analysis of the intelligent aerial platform with AI capabilities for effective monitoring of harmful plants. Such a platform needs to have a visual detection system onboard to predict localization and geometric parameters of the plant correctly. It should have low power consumption for long-range flights. The first task is addressed by developing a fully convolutional neural network and its optimization for inference on a low-power embedded system. The development of UAV with MW and algorithms of control for it addresses the second task.

As a practical application of such a platform, we determine the Hogweed of Sosnowski (*lat. Heracleum*). It is poisonous for humans, dangerous for farming crops, and local ecosystems. This plant is fast-growing and has already spread all over Eurasia: from Germany to the Siberian part of Russia, and its distribution expands year-by-year. In-situ detection of this harmful plant is a tremendous challenge for many countries. Meanwhile, there are no automatic systems for in-situ detection and localization of hogweed. In this Thesis, I propose, evaluate, and test a new combined approach for fast and accurate detection of hogweed by UAV.

1.3 Thesis Goal and Objectives

The goal of the current research is mathematical modeling and evaluation of a unique intelligent UAV platform for the detection and elimination of harmful plants during the flight. The platform should have capabilities for data processing, classification, and spread assessment of harmful plants using AI on low-power embedded systems onboard. These capabilities should be enhanced by new physical means of drone control, like MW. These capabilities should become an inherent part of the control system for effective spraying and eliminating harmful plants during the flight. For the successful implementation of a complex trajectory, new physical means of control, like MW, should be created. This thesis goal should be addressed by the following thesis objectives, which could be separated into two groups: (I) modeling and development of computer vision system and (II) analysis of aerodynamic surfaces for effective implementation of the flight. These two groups contain detailed sub-tasks.

1. Computer Vision System:

    - Collection of the dataset for further training of the neural networks and computer vision algorithms. It should include birds-eye view images and segmentation of the harmful plants. Dataset collection will engage the usage of multispectral and RGB cameras. The data should be collected in different areas for both individual plants and thickets and should have

different ambient conditions (like weather and lighting). These images should be labeled using a semantic mask for two classes: "hogweed" and "not hogweed."

- Creating a new neural network architecture optimized for embedded electronics. This task implies test of different optimization techniques and development of new optimization technique for inference of FCNN onboard of mobile device, based on Central Processing Unit (CPU), or Graphics Processing Unit (GPU), or Visual Processing Unit (VPU). the computationally effective architecture will allow not only to detect the location of harmful plants but also will be able to process video in real-time during the flight, taking into account image blur and different ambient conditions (light, weather).
- Development of the platform for data collection and data processing. It will be the payload for the drone. Hence it should have both capabilities for onboard data-processing and the data-transferring of the processed data to the operator.
- Analysis of the obtained results. Testing of software and hardware system (UAV) in field conditions. Verification of the work of the proposed algorithms at the test field areas where the data was not previously collected. Thus, the testing of the methodology and the proposed solution will be independent, and the test results will show applicability in real conditions. This objective also includes the optimization of the software and hardware based on the results of field testing.

2. Analysis of aerodynamic surfaces for effective implementation of the mission:

- Creating new physical means and control methods for the effective executing of the flight for precise data collection.
- The range of the vehicle and its controllability should be enhanced by these physical means.
- The investigation should be performed both by computational methods and in relevant flight conditions during the wind tunnel experiment.

1.4 Methods and approaches

The methodology behind the research mostly rely, on the following theoretical, numerical, and experimental approaches:

- Dataset collection will be carried out with the help of a UAV with a camera in manual or autonomous mode in different weather conditions and under different light conditions. Manual dataset labeling as well as with the help of a pre-trained neural network. The list of equipment contains:
  - Portative multispectral cameras: CMS-V (8 color bands + black and white channel, range 550-850 nm), SeekThermal Pro (portable thermal imaging camera), Mapir kernel (19 bands including the near-infrared range).
  - UAVs registered in the manner prescribed by the legislation of the Russian Federation: DJI Phantom 4, DJI Matrice 200, and Geoscan 201 Agro.
  - Mobile laboratory based on Kamaz for the organization of field trips with the drone and its recharging during the day.
- The development of all software will be carried out using the programming languages Python and C++.
- Development and testing of the architecture of the neural networks using Python with Caffe, TensorFlow, OpenCV, scikit libraries based on FCNN for mobile platforms.
- Application of embedded systems and single-board computers to be the core part of the payload's processing unit.
- Testing of existing optimization techniques and development of new optimization techniques for inference of FCNNs onboard of low-power embedded systems.
- Application of the embedded systems and external GPUs to be inherent parts of the autopilot.

- Application of Sensor Fusion methods and methods for processing signals from sensors (such as Kalman filters, particle filters, etc.) in the development and implementation of the navigation, orientation, and stabilization systems, to support reliable flight and good quality data collection.
- Testing of all subsystems separately and in the assembly will be conducted in the laboratory and operating conditions.
- Application of Computational Fluid Dynamics (CFD) software for analysis of MW for the proposed eVTOL system.
- Manufacturing of the model for wind tunnel testing and flying tests, using 3D Printers, Computer Numerical Control (CNC) machines, and carbon manufacturing facilities to produce
- Wind tunnel testing of the MW prototype with Particle Image Velocimetry (PIV).

"Criticizing someone else's, offer your own, and by offering - do it."

Sergei Korolev

Chapter 2

State-of-the-art

Here is a comprehensive review of the literature related to the topic of this work. The research focuses on the development of the UAV platform for effective monitoring and eliminating harmful plants. As an example, we determine the Hogweed of Sosnowski (*lat. Heracleum*). The approach includes the UAV with an embedded system onboard running various FCNN. The UAV airframe also includes the MW. Therefore the Literature Overview contains three parts: (i) Plants' Phenotyping; (ii) Edge Computing in Precision Agriculture; (iii) UAV Control and MW.

2.1 Plants Phenotyping

The worldwide population is continuously growing. As well as the average daily calorie consumption per capita increases annually. Therefore precision agriculture is becoming a hot topic for countries and organizations engaged in agriculture Pahuja et al. [2013a]. Indeed, the growing population, the need to extend the area of arable land, and the increase of the chemical inputs to the soil caused many issues regarding environmental impact and food safety. Hence, the interest in emerging technologies developed for the mitigation of these issues also grows Elijah et al. [2018b], Taylor et al. [2013b]. The greenhouse climate control system continuously advance Gupta and Quan [2018] in many ways, including application of IoT paradigm Elijah et al. [2018a] along with AI had been applied to precision agriculture Muangprathub et al. [2019] Kamilaris and Prenafeta-Boldú [2018]. All these modern technologies along

with the ML techniques are also widely used for modeling the plants for further estimation Ali et al. [2017a] and investigation of agriculture deployments Abouzar et al. [2016]Somov et al. [2018].

According to the International Society of Precision Agriculture ISPAG [2020], the precision agriculture is a management strategy that gathers, processes and analyzes temporal, spatial and individual data and combines it with other information to support management decisions according to estimated variability for improved resource use efficiency, productivity, quality, profitability and sustainability of agricultural production. This scientific area engages the emerging technologies that have been successfully applied recently, e.g. remote sensing Zhou et al. [2016], artificial intelligence Lane et al. [2017c], robotics Chaudhury et al. [2015], sensor networks Eugster et al. [2015] and IoT Alavi et al. [2018]. These promising technologies allow for securing food safety, reduce the impact on the environment, and ensures profit for the economy.

Therefore, precision agriculture is a vast domain and includes almost all areas of agro technologies. However, for the proposed research, there are three relevant cases for application and testing of Edge AI: seed germination in a climate chamber, growth of plants in a greenhouse, growth of plants in an open ground. All of these challenges are addressed by applying and testing different Edge AI solutions with computer vision capabilities. The subsections below overview the most recent and relevant researches behind the topic.

2.1.1 Detection of Seeds During Germination

The controlled seed germination is a crucial problem for modern precision agriculture. As an adult plant, it also has many growing stages that further influence plant growth and health. The controlled seed germination has already been addressed Bello and Bradford [2016]. However, the research in this area is rather fragmented Lee et al. [2017], Ducournau et al. [2005], Awty-Carroll et al. [2018]. From the industrial point of view, the seed germination control by low-power embedded sensors Somov et al. [2015] is a missing link in the greenhouse automation process, especially for the remote areas. This approach helps to realize distributed

intelligent sensing into practice.

Modern ML, along with computer vision algorithms, has become a powerful tool in modern precision agriculture Yao and Ge [2018] and covers many hot topics in this area. A combination of these techniques advances studies in plant phenotyping, plant growth dynamics analysis, large-scale data processing for detailed investigation of the plants, and their characteristics. The image-based approaches behind this area mostly rely on various CNN and FCNN. These architectures and their possible implementation Gu et al. [2018] could be applied for classification, detection, semantic, and instance segmentation tasks. These computer vision techniques demonstrated promising results in various scenarios. For example, the leaves segmentation for plant phenotyping Scharr et al. [2016], Dai et al. [2015], Dai et al. [2016], various tasks in greenhouse control and indoor farming Ghanem et al. [2015], Li et al. [2014], etc. These data-driven algorithms require datasets collection and benchmarking. Nowadays there is plenty of various datasets for multiple tasks in precision agriculture, like phenotyping and growth dynamic evaluation Cruz et al. [2016], Minervini et al. [2016], Silva et al. [2013]. Even though there are several dynamic models of seeds germination Forcella et al. [2000], Bello and Bradford [2016] and a variety of methods of root phenomics Das et al. [2015], there are no publicly available datasets for seed germination task. Therefore, one of the dissertation's objectives is the collection of such a dataset, and it will be described in more detail in the further sections.

The papers that report on the most recent advances in the area of precision agriculture Taylor et al. [2013a], Elijah et al. [2018a] declare that deployment of AI on embedded systems is not a well-studied area yet and there is a high demand in the industry for the implementation of such systems. At the same time, there is no mentioning of autonomous monitoring with AI aboard, as well as the prediction tasks in the agriculture domain. Moreover, for the autonomous operation, it is the limited energy storage and high-power consumption that has been of great concern.

As has been noticed earlier, computer vision, together with machine learning and low-power sensing, could help in addressing the seed germination control problem for industrial automation Zhabelova et al. [2015]. However, squeezing the machine

learning algorithms into the low-power embedded systems is a non-trivial task Lane et al. [2017a].

An essential prerequisite for solving this problem is the AI algorithms that can run on a low-power embedded system. This idea could be realized by applying the edge computing paradigm performing data processing with the artificial intelligence aboard a low-power sensing device. This solution does not require local powerful data processing or complicated data transmission to the cloud, restricted in the Wireless Sensor Networks (WSN) applied to the monitoring tasks in greenhouses Pahuja et al. [2013a].

WSN is a collection of wireless battery-powered sensor nodes deployed over a vast territory Ivanov et al. [2015]. The sensor nodes perform the monitoring tasks and send the data to a user/server over the wireless network. Sensor nodes are equipped with the low-power sensors and inherently avoid applications involving the images or video processing and their transmission as it implies the strict power consumption requirements. Although some works on the use of AI to image processing in WSN already exist Shadrin et al. [2019b], this approach does not allow for the extensive areas coverage, the long-term and autonomous operation of sensing devices. The applications of WSN with the AI are typically limited to the smart monitoring of greenhouses Somov et al. [2018].

Indeed, the general bottleneck of cloud computing technology is the throughout capacity of a data transmission system, e.g., the Wireless Sensor Networks (WSN), the computational capability, and occupancy of the server. This bottleneck could be addressed by the application of the edge computing paradigm Shi et al. [2016]. It performs the data-intensive computation tasks onboard of nodes without involving massive data transmission to the remote server. The ultimate advantage of these systems is a significant reduction in the output data size. For example, instead of sending images to the cloud server (in the range of few Megabytes), edge-computing systems generate segmentation masks (or the text files). They contain coordinates, labels of objects on the image, and quantity characteristics of interest (in the range of few Kilobytes) used for further system control. Such a distribution improves the reliability and prevents the data losses caused by the blackouts or hack attacks on

the server.

2.1.2 Detection of Plants in Greenhouse

The investigations accomplished in the area of precision agriculture in recent years demonstrated significant advances. However, there are still some challenges, that remain unresolved. Indeed, a high number of approaches have already been proposed to automate greenhouses and process environmental data De Silva and De Silva [2016], Li et al. [2014]. However, the problem of automating greenhouses and monitoring plant growth in remote areas (particularly in developing countries) Pahuja et al. [2013b] still exists. It can be explained by the challenges associated with the system autonomous operation and transmission of collected data to high-performance computers/clouds for further processing. In terms of autonomous operation, it is the limited energy storage and high-power consumption that has been of great concern.

Regarding prediction, there is a lack of robust universal models available for the quantitative description of plant biomass changing with the time that can perform the predictive analysis. Although there are mathematical models that can be applied to the direct simulation of plant growth (the so-called "bottom-up" approach Rodríguez et al. [2015]), most of them are based on solving systems of differential equations and involve large numbers of semi-empirical parameters. Therefore, they have to be adapted to each specific type of plant and the cultivation technique. It makes them sensitive to some hidden changes in the environmental and other conditions that are difficult to track Vereecken et al. [2016]. Technically, in the remote areas, it is almost impossible to obtain all the necessary parameters for making a good quality predictive model to assess the plant growth dynamics based on the "bottom-up" approach.

A distributed low-power embedded system with AI on board is required for addressing this problem. This solution is in line with the "edge computing" paradigm aimed at data processing with the AI on board the sensing device without involving complicated data transmission and its further processing in the cloud.

Recent success in smart agriculture domain is mainly powered by the advances in Wireless Sensor Networks (WSN), 2D/3D imaging systems, machine learning, and

cloud computing. Although this research and relevant prototypes are still limited, we provide a concise discussion of the successful attempts in this direction.

The Wireless Sensor Networks (WSN) paradigm is a driving force for monitoring and control applications Somov et al. [2013] including the control of the climate conditions in a greenhouse Pahuja et al. [2013b], Mirabella and Brischetto [2011], Mendez et al. [2012]. Environmental monitoring is a reasonable chip solution for greenhouses and tends towards the *set-to-forget* monitoring option for long periods Lombardo et al. [2018]. Tiny sensors were deployed in difficult-to-access areas at different height and measured the greenhouse relative humidity and temperature. The simple control mechanism characterizes this deployment: the actuators are activated if the threshold values are violated. In this case, the system controls the predefined settings of the greenhouse. Although this study reports on compact wireless sensing devices performing energy-efficient tasks when the measured data are delivered directly to the user or cloud facilities for further processing and data storage, they inherently lack intelligence. It happens due to the limited computation and processing resources on microcontrollers, which are not able to run complicated algorithms.

Up to date, AI has been applied for modeling applications in agriculture. For example, the machine learning approach is successfully applied for modeling and further estimating grasslands in Ireland Ali et al. [2017b]. The authors estimate by processing large volumes of available image data. In terms of modelling, three models were developed: Artificial Neural Network (ANN), Adaptive Neuro-Fuzzy Inference System (ANFIS), and Multiple Linear Regression (MLR).

Imaging approaches include 2D, 3D, and sometimes 2D/3D imaging. The 2D approach is typically helpful in scenarios when a simple structure characterizes a plant and rather large leaves as analyzed in Rajendran et al. [2009]. Simultaneously, computer vision and machine learning-based solutions demonstrated their advantages in performing the assessments of fruit characteristics Pouladzadeh et al. [2014]. The disadvantage of 2D imaging is the complicated software exploited for image analysis. It suffers from the leaf overlap and concavity. A laser scanning approach is often a good option when plant digitization is required. It has been successfully

applied, for instance, to forestry and canopies statistical analysis Yang et al. [2013]. Due to computationally intensive data processing, its application is limited to the extraction of single plant attributes.

3D imaging is used to capture the shape of the plant and its further analysis in three dimensions. Thus it overcomes the problems typical for 2D imaging. One of these methods is reported in Quan et al. [2006]. The authors demonstrate a semiautomatic 3D imaging system for plant modeling where the reconstructed 3D points and the actual images are combined. According to this research, more effective segmentation of the data into individual leaves will be guaranteed for the user.

Another research proposes a 2D/3D system enriched with many sensors Shadrin et al. [2018a], which can find the correlations between the leaf area and biomass. The demonstrated approach assists in predicting the growth rate and the leaf area of the plant. This solution requires a computer or a cloud solution for data processing, though.

Detailed reviews of the state-of-the-art smart approaches with a particular emphasis on the agriculture domain are present in the articles Elijah et al. [2018b], Taylor et al. [2013b]. To the best of our knowledge, these works do not report on the *deployment* of AI on the embedded systems. Also, they do not target *autonomous monitoring*, as well as analysis and prediction tasks in agriculture.

2.1.3 Detection of Plants in Open Ground by UAV

At present, agricultural monitoring is typically realized through the Wireless Sensor Networks (WSN), satellites, or UAV. Since harmful plants usually spread around in the wild and open fields, this subsection is devoted to an overview of the most useful aerial platforms for precision agriculture - satellites and UAVs.

The distribution of Hogweed of Sosnowskyi (*lat.Heracleum*) is a growing problem in agriculture for many countries. This fast-growing weed spreads over Eurasia quickly: from Germany to the Siberian part of Russia, and its distribution area is expanding from year to year. Hogweed is a weed of 3–5 m in height characterized by a straight, firm stem typically reaching a diameter of 12 cm. Its root is very firm and is up to 30 cm diameter; the inflorescence is a big umbel located at the end of each

stem. While blooming, it produces thousands of seeds that are easily distributed by the wind and water. The hogweed is poisonous for humans, dangerous for farming crops, and local ecosystems. It creates another research demand to detect and remove this dangerous weed. The detection task, therefore, imposes many critical requirements:

- monitoring and detection should be realized without the human presence to avoid the distribution of the seeds.
- monitoring is expected to cover vast areas.
- for quick-acting, the data analysis results must contain the positioning data and be available 'real-time.'

UAVs and satellites are another options for collecting the agricultural images and performing smart monitoring. Both drones and satellites have advantages and disadvantages. Some of them are critical for precision agriculture.

The Satellites-UAV tradeoff

For satellites, the selection of the altitude above the ground is always a trade-off between the Field of View (FOV) and Ground Ground Sample Distance (GSD). That is why a satellite can capture thousands of km^2 per orbit pass lasting for around 90 minutes with a low GSD typically equal tens to hundreds of meters per pixel. The best available resolution for satellite image is 30 cm/px Shean et al. [2016]. Drones usually capture the tens of km^2 per flight with the sub-centimeter precision. It is critical for many agriculture-related tasks Rodriguez-Moreno et al. [2017], e.g. weeds classification Liu et al. [2013], identification of crop diseases Zhao et al. [2012], vegetation indices Vega et al. [2015]. That is why the hogweed detection problem requires high GSD as well.

Another critical advantage of drones in comparison to the satellite option is a high temporal resolution. The satellites usually keep the sun-synchronous or polar orbits for the Earth observation, which ensures daily revisiting of the same location on the ground in specific local time. The problem of capturing the same area several

times a day by satellites can only be solved by satellite constellation, distributed over different orbits. Drones can perform a mission by demand along an arbitrary trajectory over the same location at any local time. It is an essential factor for hogweed detection since it is a fast-growing plant, and space imagery could miss the flowering phase.

Nowadays, there are many UAV systems for monitoring agricultural fields Huuskonen and Oksanen [2018]. However, the UAVs accumulate images during the flight about the absolute coordinate system, and the postprocessing of these images, e.g., segmentation and classification, occurs on the computer after the mission. Depending on the task, these operations have to be performed daily or even hourly. It is an inefficient and labor-intensive task. Recent research works have demonstrated the opportunity of creating the autonomous systems and their application in agriculture Andrew et al. [2019]. While none of them perform the semantic segmentation fully on board, many tasks require the semantic image segmentation and the object classification during the mission. At the same time, there are no available relevant multispectral or visual spectrum datasets with a high resolution for plant detection, segmentation, and classification in the field. These datasets are essential for training different Machine Learning (ML) algorithms to deploy the trained algorithms on the embedded systems.

There are examples of detection of thickets of harmful plants based on the blue, green, red, and near-infrared multispectral bands of the satellite Sentinel-2; For example, blooming hogweed stood out in the pictures as pixel scatter spots Tovstik et al. [2018]. A project similar in content is being carried out in the "CosmoInform-Center," the detection of malicious bushes based on the analysis of multispectral high-resolution RGB satellite images. The bright green mask highlights the target object in pictures. In this case, the spectral brightness of the pixels corresponding to the hogweed differs from the background pixel values of the surrounding vegetation and human-made objects in the spectral range of 450-870 nm. The common shortcomings of these methods include the limit of the spatial resolution of images (6.5 - 10 m), which means the impossibility of detecting individual plants and a high error of determination. Besides, it is impossible to obtain images at a given point in time

(data from freely distributed LandSat-8, Sentinel -2A updates with the frequency of shooting at 16 and 10 days, respectively). That is why the UAV development, capable of performing data processing on board, is on high demand.

The cost of satellite design, manufacturing, launch, and operation are usually disparate compared to the cost of the drone. For example, the WorldView-3, the modern reconnaissance satellite by DigitalGlobe, costs 650 million US dollars to build and launch Kakaes et al. [2015]. Satellites perform continuous operation all the time in orbit. The operation cycle can last from several months (for Low Earth Orbit (LEO) nanosatellites) to 15 years (for Geostationary (GEO) satellites). That is why the price of a single shot by the satellite stays constant in the range 250-350 US dollars per 25 km^2. Drones operate by-demand; hence the price per shot depends on the particular drone utilization capacity and captured area. That is why the drones are cost-effective only for the 'small' areas of several hectares. Matese et al. [2015].

All the satellites are the subject of a cloudy sky problem. Clouds cover around 40% of the Earth's surface, which is inaccessible for the carrying out the space imagery Kakaes et al. [2015]. The drones perform the local imagery missions even in the case of the low cloud conditions.

Finally, the satellite imagery can be accessed only during the radio window availability when the satellite and the ground station are in the line of sight. The drone imagery can be obtained during or right after the flight.

To sum up, UAV is a suitable platform for agricultural field monitoring, especially in the case of hogweed and other harmful plants. There could be both the thickets and the individual plants. Individual plants usually occupy a limited area. However, they can produce up to 20 000 seeds a year. That is why the detection of individual plants is crucial for the effective elimination of hogweed. Modern remote sensing technologies can't observe individual plants with sufficient precision from space. That is why UAVs are the most appropriate platform for effective hogweed detection.

Sensors for Plant Phenotyping in Aerial Imagery: Multi-spectral and RGB cameras

Multi-spectral imaging opens a wide vista for performing the precise plant and phenotype detection Dutta et al. [2015] Sodhi et al. [2017] Humplík et al. [2015]. The main difference between the plant detection in multi-spectral range and common RGB is the following: some plants have their unique reflection wavebands, which could be out of visible range. The multi-spectral approach enables the application of simple classification algorithms instead of complex FCNNs for plant shape detection. These algorithms classify the pixels and find those with the intensity representing the required waveband and belonging to a certain plant. Many research works in multi-spectral plant phenotype were dedicated to the remote plant disease detection Martinelli et al. [2015]. Application of multi-spectral and hyper-spectral digital cameras for the plants' disease detection is an exciting research area as a vast amount of diseases on the initial and developing stages have their unique spectral characteristics. This approach allows farmers to detect the diseases at an early stage and prevent the infections from spreading Picon et al. [2019] Große-Stoltenberg et al. [2018]. Multi-spectral imaging is also used for the detection of physical stress Hong et al. [2019]. One of the disadvantages is the cost of high quality commercial multi-spectral cameras comparing to the cost of the typical RGB cameras. The low-cost multi-spectral solutions mostly aimed at measuring the Normalized Difference Vegetation Index (NDVI) Kitić et al. [2019], Kim et al. [2019] and they can not measure the necessary wavebands for specific plant detection. Hyperspectral cameras can measure spectrum in a vast range and have an excellent spectral resolution, but they are too expensive and heavy to put them on the drone as a payload Shuaibu et al. [2018]. Research work reported in Khan et al. [2018] has shown that the trained CNN reconstructs the NDVI index from the RGB images. It is commercially effective to use the RGB camera together with the CNN algorithm in comparison to the multi-spectral camera. Therefor, in the research we use RGB camera.

Methods of Aerial Imagery Segmentation

The problem of phenotyping and estimating the various plants distribution in farms has been addressed by different methods. Several processing algorithms have been proposed for aerial imagery analysis. The localization and phenotyping of various plants have been addressed for many years by different methods relying on the handcrafted features Lottes et al. [2016]Lottes et al. [2017] or the end-to-end principle Potena et al. [2016] Di Cicco et al. [2017]Sa et al. [2018]. The first group of research works involves a reasonably simple approach, e.g., extraction of statistics from the HSL (Hue, Saturation, Lighting) color space and using them as an input for the neural network Burks et al. [2000]. However, recent methods based on classical Machine Learning approaches engage the Random Forest classification Haug et al. [2014] and a variation of this method which includes Random Markov Field Lottes et al. [2016] optimization. The same methodology was applied for the aerial imagery classification, which includes the RGB channels only Lottes et al. [2017]. Due to the limitations associated with flexibility, the algorithms behind the remote sensing analysis in precision agriculture move to the end-to-end approaches. They rely on various deep learning methods to a large extent. These methods include Convolutional Neural Networks (CNN) and Fully Convolutional Neural Networks (FCNN). The first ones proved their efficiency in classification and detection tasks. In the area of precision agriculture, the cascades of CNNs showed high efficiency in the generalization and processing of previously unseen data Potena et al. [2016]Fawakherji et al. [2019]. Whereas the FCNNs became a standard approach for the semantic and instance segmentation tasks Badrinarayanan et al. [2017]Ronneberger et al. [2015]**?**. FCNNs work well for both the RGB Lottes et al. [2018] and the multispectral imagery Sa et al. [2018]Milioto et al. [2018]. However, FCNN is slow for real-time processing tasks, especially while running on the low-power embedded systems. In some cases, it is impossible to make the inference using it. In the current research, we investigate the optimization techniques for making inference of the FCNN architectures running on the low-power embedded systems.

2.2 Edge Computing in Precision Agriculture

In modern science and technology, there is a rapid development of mobile electronics nowadays. Following Moore's law, the number of transistors placed on an integrated circuit increases twice every two years. It leads to a reduction in the size and power consumption of electronics Zaffiro [2015], which means the emergence of increasingly high-performance miniature devices, such as wearable electronics, laptops, smartphones, portable photos and video cameras, devices for the Internet of Things (IoT). Also, a similar trend is in the industry, which leads to the emergence of such embedded systems as miniature autopilot for UAVs Venkatesh et al. [2017] Santoso et al. [2015], microelectronics for nanosatellites and their subsystems Osman and Mohamed [2017] Sánchez-Macián et al. [2017], unmanned vehicles Gurghian et al. [2016] Bojarski et al. [2016], smart cities Misbahuddin et al. [2015], smart houses Kodali et al. [2016], as well as electronics for automated production (the so-called Industry 4.0) Cheng et al. [2016].

On the other hand, in the last decade, there has been a rapid development of artificial intelligence technology. It became possible due to the ubiquitous development of the Internet (which is necessary for collecting large amounts of data for training neural networks) and multiprocessing and multi-threaded processors (for example, graphics chips essential for the effective implementation of neural networks). In turn, this pushed the development of the so-called Convolutional Neural Network (CNN), as well as methods of segmentation of objects using computer vision. They include K-means methods Dhanachandra et al. [2015], methods of growing regions Vo et al. [2015], segmentation using the watershed method Bai and Urtasun [2017], semantic segmentation Shadrin et al. [2018b], and many other methods He et al. [2016a]. Over the past decades, convolutional neural networks have also made great strides. Starting from the classification of handwritten numbers LeCun et al. [1989] and ending with the classification of emotions by facial expression Fan et al. [2016], interpolation of frames in slow motion Jiang et al. [2018], improvement of the quality of Magnetic Resonance Tomograph (MRI) images Milletari et al. [2017], UAV control Smolyanskiy et al. [2017] Carrio et al. [2017], unmanned vehicles Bojarski et al. [2017], etc.

Along with this, the use of semantic segmentation technology in image detection and classification tasks allows determining the boundaries of an object with pixel accuracy in real-time Kulikov et al. [2018]. Besides, a field of deep learning (Deep Neural Network (DNN)) has appeared, which significantly expands and deepens the capabilities of neural networks Schmidhuber [2015].

The development of technologies for embedded electronics and artificial intelligence led to the emergence and development of cost-effective multiprocessing and multi-threaded video processors for systems with low computational power Marantos et al. [2018] Wang et al. [2016]. And also to the emergence of special neural network architectures for efficient and fast data classification on mobile platforms: You Only Look Once (YOLO) Redmon et al. [2016], You Only Look Twice (YOLT) Van Etten [2018], Darknet Redmon and Farhadi [2017], Fast Regions with CNN Features (R-CNN) Girshick et al. [2014], MobileNet Single Shot Detector (SSD) Howard et al. [2017], Region-based Fully Convolutional Neural Network (R-FCN) Dai et al. [2016], etc.

Besides, nowadays the edge computing has become of high demand due to the intensive development of mobile platforms, IoT, robotics, embedded systems, wearables, etc. Edge computing has multiple advantages compared with Cloud computing and Fog computing Marantos et al. [2018] due to the following technological limitations Ignatov et al. [2018]:

- privacy issues;
- dependency on an internet connection;
- delays associated with network latency;
- bottleneck problems: the number of possible clients depends on the server's computational capabilities.

These trends have significantly influenced the development of modern means of obtaining and processing aerial photographs. The development of embedded electronics has contributed to the widespread development of various UAVs for aerial

photography, agriculture monitoring, and infrastructure inspection. The methodology could rely on obtaining depth maps by stereo pairs or lidars as well as on various thermal and multi-spectral cameras. It has led to the accumulation of a significant amount of Earth Remote Sensing (ERS) data, particularly aerial photography data. Such data as Dataset for Object Detection in Aerial Images (DOTA) Xia et al. [2018], INRIA Aerial Image Labeling Dataset Maggiori et al. [2017], SpaceNet Yuan [2017], VIRAT Video Dataset Oh et al. [2011], CLIF 2007 Dataset CLIF [2007], Stanford Drone Dataset Robicquet et al. [2016], EPFL MMSPG Video Drone Dataset Bonetto et al. [2015], as well as many other data from aerial photography and space photography. On the other hand, the accumulation of a large amount of data allowed the use of neural networks to solve problems more effectively in various industries. The exception was not the task of germinating seeds Rasti et al. [2018] and monitoring agricultural land with the help of UAVs Candiago et al. [2015].

At the moment, there are many UAV systems for monitoring agricultural land. However, usually, such UAVs accumulate photographs during the flight relative to the absolute coordinate system. The processing (and classification of objects on them) occurs post-factum on a personal computer or server after the flight. Depending on the task, these operations have to be performed daily or even hourly, inefficient, and labor-intensive. Thus, at the moment, there are no commercially available UAVs that classify images directly on board during the flight. However, there are already many tasks, which require semantic image segmentation and object classification during the flight Chamoso et al. [2014] Pérez et al. [2014] Jordan et al. [2015] Xu et al. [2017] Flammini et al. [2016] Adão et al. [2017].

However, implementing such a smart computer vision system on board of low-power embedded electronics with the constrained size is the problem. The reason is the lack of energy stored in the battery-powered IoT devices and computation capability, which are expected to be deployed everywhere and available anytime. Many research projects tackle the long-term operation problem of low-power sensing devices, e.g., sensor nodes, from a different perspective Minakov and Passerone [2013], Khaled et al. [2015], Kaup et al. [2014]. It is worth noting that most sensing devices are based on a low-power Micro Controller Unit (MCU), and relatively

computational extensive algorithms could be run Somov et al. [2017]. However, the research in adopting artificial intelligence for energy-constrained devices, e.g., embedded systems and mobile phones, is still fragmented Lane et al. [2017b] with the limited number of experimental-based research projects Chauhan et al. [2018].

A solution to address this problem could be to use Single Board Computer (SBC) with powerfull CPU, embedded or external GPU and VPU; or use FPGA or devices for executing these algorithms.

Therefore, the application of various fitting approaches of ML algorithms to mobile and embedded platforms with limited computational capacity is necessary to overcome the issues mentioned above. There are many ways to fit the Machine Learning algorithm on board of edge computing platform. They may include hardware acceleration (HA) by Digital Signal Processor (DSP) Codrescu et al. [2014] or GPU Latifi Oskouei et al. [2016]. The DSP HA is widely used in mobile platforms due to high performance with low power consumption (even in comparison with CPUs and GPUs). The GPU HA implies parallel computations across CPU and GPU. The following libraries could implement it: TensorFlow Mobile TFM [2019], Android Neural Network API (NNAPI) NNA [2019], RenderScrpit-based CNNdroid Latifi Oskouei et al. [2016], and RSTensorFlow. The last one is the GPU-based accelerator of matrix operations, making it possible to accelerate matrix multiplication up to 3 times Alzantot et al. [2017]. Furthermore, some studies show that RenderScript could be used even with CPUs imprecise computing modes to lower execution time of computationally-intensive models Motamedi et al. [2019]. Also, System-on-Chip (SoC) manufacturers propose the SDKs, compatible with their products only. They include SNPE by Qualcomm, HiAI platform by HiSilicon, NeuroPilot SDK by MediaTek, etc.

2.3 Morphing Wing for Control of the UAV

2.3.1 UAV Optimal Control

The subject of UAV optimal control could be divided into two fields: control and estimation. Control usually implies applying different PID (Proportional, Integral,

Differential) controllers, which use the control signals and sensory data, processed by estimators, to send correct commands to the actuators and motors. Even though simple 1D and 2D motion, a single P, PD, or PID controller are sufficient, in case of complex 3D motion, the system of cascaded PID controllers is required.

Estimation of UAV implies evaluating the current state vector (usually consisting of 3 translation, three rotational coordinates, and their first, second derivatives, and biases) from UAV's sensor values and control inputs. Since the data from sensors is usually noisy, this problem becomes challenging. Moreover, due to weight and cost constraints, UAVs have limited on board processing capabilities. Therefore, there is a need to estimate the abovementioned values as quickly as possible. Estimation traditionally relies on application of various Bayes Filters, like Extended Kalman Filter (EKF) Thrun et al. [2005]. It is a nonlinear extension of the Kalman filter, which linearizes a nonlinear transition and measurement model around the current state. However, the Unscented Kalman Filter (UKF) Wan and Van Der Merwe [2000] has comparable runtime, but it is simpler to implement, and it has better accuracy.

Usually, the accelerometer and gyro inputs are used as control inputs. For example, Erdem and Ercan [2015] show, that it is usually better to process the data from both positioning (either Geo Positioning System (GPS) or camera) and attitude, Inertial Measurement Unit (IMU) sensors for the measurement stage. Cristi and Tummala [2000], Quan [2017] also showed in their works, that multi-rate KF improves the precision of estimation. However, the update is performed separately for the GPS, magnetometer, and IMU.

It means that the state vector contains both acceleration and angular velocity. However, it could be neglected to simplify the math and implementation, as performed in the Ardupilot ard [Accessed March 5, 2018]. Due to the limitations of computational capabilities, one may use different state transition models in control compared to the EKF.

It was shown in Erdem and Ercan [2015] that EKF for a mobile device with IMU and camera has the best performance, while the IMU used as a measurement input. It gives 1 cm accuracy. At the same time, the algorithms behind Ardupilot men-

tioned above, rely on the idea of using IMU data to perform control and prediction updates. For example, according to Bry et al. [2012], it is a standard approach.

2.3.2 Morphing Wing for UAV

UAV progressed a lot within the last decades Lim et al. [2012]. This technology enables a high number of monitoring applications Colomina and Molina [2014], as well as commercial services Liu et al. [2014]. At the same time, the UAVs do not carry humans, opening up wide vistas for experimentation on mechanical parts and electronic systems. The Morphing Wing (MW) is a promising technology Noviello et al. [2017], helping improve the flight control and efficient use of the fuel. MW relates to the so-called Intelligent Structures Wada et al. [1990] that have close integration of the actuation, sensing, controlling, and computing capabilities. That kind of structure can sense and perform actions according to the external conditions based on non-trivial controlling and computing algorithms. Highly cognitive properties of those structures make them the kind of bio-inspired embedded neuro systems for engineering applications. To better understand the Intelligent Structures concept, one may refer to the article Wada et al. [1990], where they are described as the subset of a complex intersection of adaptive, sensory, and controlled structures and include the high authority control system Crawley [1994]. The adaptive structures are defined as those that possess the actuators that alter system states and characteristics in a controlled manner Wada et al. [1990]. According to another definition, the Adaptive Structures are the structures that can modify their geometric configurations and physical properties purposefully Larson [1966]. The adaptive structures have a system of distributed actuators. All the wings of conventional aircraft have multiple actuators for flaps, slats, ailerons, etc. That is why the wings of traditional aircraft are the adaptive structures. The sensory structures Mustapha [2017] have the system of distributed sensors. These sensors send back information about the state of the structures or the external conditions (temperature, flow velocity, pressure, current, etc.). The controlled structures Sairajan et al. [2016] are within the overlap of the adaptive and sensory structures. The state of these structures could be influenced by the information from sensors in a simple close-loop or more

advanced control system architecture. The active structures Wu et al. [2016] are within the subset of controlled structures. Their actuators also have load-bearing functionality. Intelligent structures, in turn, is the subset of active structures. They usually have complex computational architecture on top of the control system. Currently, there are many examples of how intelligent structures can be applied to the modern aerospace industry. These examples include: aeroelastic control and maneuver enhancement of a helicopter Dimino et al. [2017], active structure damping Tang et al. [2017], wave propagation control Bergamini et al. [2015], active stabilization (aircraft flutter) Tsushima and Su [2017], vibration and shape control Zhang et al. [2015], creating the adaptive nozzle for the noise reduction Machairas et al. [2014]. Intelligent Structures could be found in many applications including lighter-than-air aircraft Sun et al. [2016], civil Dimino et al. [2017] and military aircraft Marks et al. [2015], UAVs Jenett et al. [2017], flapping wing aircrafts Zhang and Rossi [2017], etc. Nowadays, the crown of that technology is the so-called Smart Intelligent Aircraft Structures (SARISTU) project Wölcken and Papadopoulos [2015]. It is the international collaboration of over 50 organizations aimed at developing the highly-efficient MW for civil airliners. The materials behind the topic of Intelligent Structures are mostly deformable and could implement multiple tasks. Some of them - due to its crystal lattice. Piezo-electric ceramics and polymers, for example, are widely used as both sensors and actuators Gaudenzi [2009]. Smart Memory Alloys (SMA) and Smart Memory Polymers (SMP) Gaudenzi [2009] are used as actuators for smooth and continuous deformation of internal structures or skin. Other materials could perform multiple tasks due to their topology. Corrugated skin Previtali et al. [2016] makes it possible to stretch and contract skin regions, which need to stay highly flexible for the actuator (e.g., skin on the bottom part of the wing in the wing-flap joint). Compliant mechanisms Vasista et al. [2016] are the topologically optimized kind of structures. They usually are single-part joint-less mechanisms that can easily substitute the analogous multi-partial mechanisms. They are designed to distribute mechanical stress over the whole structure and, consequently, sustain billions of deformations. Furthermore, due to the absence of joints, they have higher efficiency of momentum transition, hence significantly lower backlashes and higher grasp preci-

sion than multi-joint tools. Due to these features, compliant mechanisms are widely used as microgrippers in medicine and physics Chen et al. [2016]. These mechanisms do not produce noise and perform smooth deformations. Therefore, they are applicable in soft robotics Liu et al. [2018], aerodynamics, and hydrodynamics. They could be used as part of wings Wakayama and White [2015], wind turbines Alejandro Franco et al. [2017], car spoilers, etc. Structures with the cellular honeycomb cores are usually used as an internal filler of the wing, and they are not new in aeronautics. Nowadays, however, it is possible to apply them as an inner deformable part of the morphing wing: as the under-skin morphing filler for smooth aileron or flap deformation Olympio and Gandhi [2010]. It could also be used as an internal structural part for morphing wings with variable wingspan Bashir and Rajendran [2018]. The skin of MW with a wide range of stable shapes could be manufactured from multi-layered smart laminates, which incorporate multiple layers made from different materials with specific properties Ferreira et al. [2016]. These properties vary depending on the aeronautics system, their purpose, and the particular application of the material. They could have various capabilities: transformable structures, skin with variable electric resistance or thermal conductivity, etc. In some fields, all these smart materials are not applicable or appropriate. That is why the structures made from traditional materials (like aluminum, titanium, steel alloys or carbon and glass fibers) with the optimized topology of the internal structure Aage et al. [2017] could become the lightweight smoothly deformable airframe. The Morphing Wing concept, as part of Intelligent Structures, could be classified into three main categories Sofla et al. [2010]: planform alteration, out-of-plane transformation, and airfoil profile adjustment. Planform alteration includes span-wise, chord-wise, and sweep transformation (swing wing and oblique wing). The out-of-plane transformation includes chord-wise, span-wise bending, and wing twisting. The most notable approaches behind these types of MW are wing sweep and wing twisting. However, the present study focuses on the development of MW with airfoil profile adjustment. The first MW of such type was created in 1986 as an embedded part of F-111 jet fighter Crawley and Lazarus [1991]. However, at that time, it was inefficient and impractical. A few years later, incorporating actuators into the wing

substructure or skin, thereby developing the actual adaptive aeroelastic structure, has been investigated with the promising results Kota et al. [2009]. However, 20 years later, "FlexSys inc." proved, that the adaptive wing with a smooth variable flap for business jets could become effective and commercially attractive Huston and Bond [2017]. Modern UAVs with morphing structures could smoothly deflect their leading edge to increase lifting force and shift separation point downstream on high Angle of Attack (AoA). They do that more effectively than UAVs with traditional mechanization Sodja et al. [2019]. There are also applications with an adaptive trailing edge. The deflection of the trailing edge could be local (adaptive flap or aileron) Kota et al. [2009] or can continuously and smoothly vary over the span Burdette et al. [2015]. Variable wingspan Bashir and Rajendran [2018] and sweep angle Di Luca et al. [2017] wings are used for multi-functional UAVs that need to perform both high-efficient loitering and high-speed maneuverable flight during the mission. UAVs with a variable dihedral angle and span-wise bending are designed to increase lateral stability and implement new approaches in aerodynamic control of the aircraft Guo et al. [2017]. Aircraft with a variable twist angle usually have better controllability and higher authority per degree deflection. These wings are often lightweight due to the advanced structural shape and distribution of the air loads over the wingspan, leading to reducing the weight of the spar Wölcken and Papadopoulos [2015]. The morphing winglet is Boeing's new advance in aeronautics materials Spivey and Suh [2018]. Although it is purposely designed for passenger aircraft, it is currently investigated onboard the UAV 1:11 scale model of passenger aircraft. It could further be used as an inherent part of the advanced morphing UAV. The abovementioned aircraft belong to classes of small, medium, and large UAVs. However, there is also wide domain of Micro Aerial Vehicles (MAV) Zakaria et al. [2012]. It usually includes paper plane-size aircraft and bio-inspired flapping wing machines, which mimic birds, insects, bats, and other flying animals. MAVs mass and wingspan doesn't exceed 100g and 15 cm respectively Shahzad et al. [2018]. The flight mechanics of these tiny machines are complex and unusual for bigger aircraft because flapping motion is usually associated with continuous deformation of flexible airfoil during a single flap Su et al. [2017]. Furthermore, aerodynamic

lift generation mechanisms for natural flyers usually produce unsteady aerodynamic forces, which vary over the phase of a flap; they include clap and fling, delayed stall, rapid pitch rotation, wake capture, etc. Zhu and Sun [2017]. The intensity of these mechanisms depends on many aspects: geometry (planform shape and aspect ratio) Shahzad et al. [2018], position (positional, deviation, and pitch angles) and amplitude of motion Zhu and Sun [2017]. For example, insect flyer with high stroke amplitude (120°-160°) mainly produces high lift by the delayed stall mechanism, whereas in a case with low amplitude (∼60°), so-called "paddling mechanism" would dominate Zhu and Sun [2017]. Numerical investigations of such mechanics rely on high-performance computing because flexible wings with complex topology require moving mesh. Moreover, the periodic motion of lifting surfaces generates sophisticated velocity and vorticity profiles Han et al. [2018]. The design of shape morphing lifting surfaces relies on the so-called aerodynamic-structural coupling. It is an already well-studied technique in the development of MW for medium-size UAVs Gamboa et al. [2009]. This technology would help in the design of MW for bio-inspired robotics and other flapping machines with complex dynamics in the future. Summing up, the Morphing Wing is a promising technology that enables improvements in efficiency, stability, and controllability of multipurpose UAVs. It implies applying multi-functional materials with embedded structural, sensing, and control capabilities, as well as lightweight structures due to their topological, dynamic, or material properties. 'FlexSys inc.' currently owns the most advanced technology, which demonstrates progress in topology optimization for compliant mechanisms, smart materials, and actuators, as well as implantation of corrugated skin. However, it was created for business jets – fast passenger airplanes that fly in the conditions with high Reynolds numbers, low AoA, and perform gentle maneuvers with low g-loads. UAVs, in contrast, operate in conditions with low Reynolds numbers and low to high AoA. Furthermore, they could perform abrupt maneuvers with high g-loads. Such a system is on high demand in application to precision agriculture, for tasks of detection of harmful plants, which usually grow irregularly in remote locations. It significantly influences the path planning of the UAVs, which need to be agile. And MW enhances this capability.

"The method is more important than discovery, because the right research method will lead to new, even more valuable discoveries."

Lev Davidovich Landau

Chapter 3

Methodology

The upcoming sections are devoted to developing and testing the low-power embedded platform with computer vision capabilities for monitoring applications in smart agriculture. The common research question of all the sections is "how to design a self-contained system capable of plant detection by using low-power computational devices and maintaining a high level of accuracy on the results." The first three sections address this question from the hardware and algorithmic perspective, the latter from the power consumption perspective, which is closely connected to the overall platform's aerodynamic aspects. Investigation begins with the computer vision system development (3.1), continues with testing under laboratory conditions in the climate chamber (3.2), then follows by the examination in the greenhouse (3.3) and finalizes by the trial on board of the UAV (3.4). Then the study turns to aerodynamical aspects of the proposed UAV, which include CFD investigation and wind tunnel experiment (3.5.2) and key aspects of the control system development for the UAV (3.5.3).

3.1 Development of the Embedded System with AI Capabilities

3.1.1 Introduction

The current section reports on the development of the monitoring platforms for various computer vision tasks. Such as classification, object detection, and semantic segmentation. The proposed approach demonstrates how to design a low-power autonomous monitoring platform with AI capabilities for various applications in precision agriculture. These applications include seeds germination in the climate chamber, the tomato growth monitoring in the greenhouse, and real-time hogweed monitoring on board of Unmanned Aerial Vehicle (UAV). The approach relies on trading-off and benchmarking the Single Board Computer (SBC) by a variety of characteristics. Besides, the optimization technique for inference of Convolutional Neural Network (CNN) on board of low-power embedded systems is reported. The section's outcomes are the basis for the monitoring platforms, reported in the sections 3.2, 3.3 and 3.4.

The results of this section were published in the conference proceedings Prutyanov et al. [2019].

3.1.2 The Trade-Off Study of the Available Platforms

The key component of the proposed embedded system with AI capability should be a SBC because these platforms are small, lightweight, cheap, and can handle computer vision tasks. However, their computational performance is significantly lower than that for modern desktops and laptops. Therefore the RAM size and the performance of the processor are limited. However, since this platform will become the key part of the testbeds in the laboratory, greenhouse, and flight tests, the characteristics of mass, size, and power become critical. The SBC candidates were chosen among the most popular commercially available computers: Raspberry Pi (RPi), Neural Computer Stick (NCS) (version 1 and 2), Nvidia Jetson Nano, Google Coral, Odroid XU4 and commercially available laptop, used as a reference.

For details see the Table 3.1.

All the proposed platforms have multiple cores and adequate RAM size to perform inference of modern CNNs and FCNNs on board. However, for real-time processing of video streams, this is not sufficient. The platform should have some AI acceleration coprocessor for fast inference. From this point of view, Raspberry Pi with various combinations of external VPU as well as Jetson Nano with powerful multiple cores GPU and Google Coral with mobile TPU seem to be the most suitable platforms for the mobile computer vision task. However, the monitoring platform should have data-intensive computing capabilities, low power consumption, small size, and weight. It will become a core part of a compact autonomous device. From this point of view, plain RPi is the lightest platform with the smallest power consumption. Therefore, there is a need to benchmark all the abovementioned platforms in various computer vision tasks. The Odroid UX4 was discarded from this list since it does not significantly outweigh other platforms.

Criterion	**Laptop ASUS UX305CA**	**RPi 3B**	**RPi 3B + NCS**	**RPi 3B + NCS2**	**Jetson Nano**	**Google Coral**	**Odroid XU4**
Architecture	x86_64	ARM	ARM	ARM	ARM	ARM	ARM
Type	Intel Core m3-6Y30	Cortex A53	Cortex A53	Cortex A53	Cortex A57	Cortex A53	Cortex A15
Cores	4	4	4	4	4	4	8
RAM, Gb	**4**	1	1	1	**4**	1	2
AI acceleration coprocessor	N/A	N/A	NCS with 12 128-bit SHAVE cores @ 0.6 GHz	NCS2 with 16 128-bit SHAVE cores @ 0.7 GHz	128-core NVIDIA Maxwell @ 0.921 GHz	Google Edge TPU ML accelerator coprocessor	N/A
GFLOPS	41.9	3.62	103.62	1003.62	472	**4000**	8.3
Average Power, W	15.9	**2.4**	3.4	3.4	10	15	13.9
GFLOPS/W	2.63	1.51	30.48	**295.18**	47.2	266.67	0.6
Size, WxDxH, mm	325x226x12.7	85.6x56.5x17	145.6x56.5x17	145.6x56.5x17	100x80x29	88.1x59.9x22.4	83x59x18
Weight, grams	907	**45**	80	80	140	138	60

Table 3.1: Single Board Computers Trade-Off Study.

3.1.3 The Benchmarking

The characteristics enlisted in the Table 3.1 are mostly available in datasheets by manufacturers. However, some of these parameters provide only general characteristics, like average power consumption. For a better understanding of the embedded system performance under conditions close to the real-time video stream processing, the dynamics of the following characteristics is required:

- Frames per Second (FPS)
- Power consumption
- CPU Temperature

The framerate could significantly change depending on the neural network type, the input sample's size, and available RAM and CPU. The power consumption and temperature of the CPU will change with the overall computational load of the system. A high CPU's temperature could lead to overheating. The power consumption is the limiting factor of the autonomous monitoring platform's power budget. Moreover, there is the need to improve all these parameters by optimization of the neural networks. Therefore, the objectives of this section are:

1. Find maximum framerate for different pre-trained CNNs for the following computer vision tasks:
    - Classification
    - Detection
    - Segmentation
2. Find the power consumption during various modes, including the inference of CNNs.
3. Find the peak CPU temperature during the inference.
4. Implement an optimization technique for the improvement of the abovementioned parameters.

The CPU and RAM loads are limiting factors of SBCs in several ways. Firstly, all the platforms with AI acceleration coprocessors (see Table 3.1) rely on CPU as the vital element for receiving the data from the camera and sending it to the CNN input. The CNN inferences on the AI accelerator and do not influence the CPU performance. Secondly, the RPi also uses CPU as the critical component for data-intensive calculations. Therefore, the CPU performance as a limiting factor of the overall monitoring platform performance. Besides, the amount of RAM influences an SBC capability to load the neural networks' parameters and provide high-resolution images at high framerate to a CNN input. Both parameters significantly influence FPS, Power Consumption, and CPU temperature. Nevertheless, the latter parameters are the most important in the current research; therefore, they will be outlined in this subsection. However, the CPU and RAM load will be demonstrated in the next subsection in the CNN optimization techniques.

FPS

For the application of SBC in different precision agriculture tasks the framerate or Frames per Second (FPS) characteristic is essential. High FPS is especially important in terms of a further application for the development of monitoring payload for UAV. The FPS in this application will significantly influence the real-time plant detection capability during the flight of the UAV.

The proposed SBCs were tested in various computer vision tasks, including classification, object detection, and segmentation. The results of the experiment are enlisted in Table 3.2. Note, that "N/A" (not applicable) results could occur due to various reasons: not sufficient memory capacity, unsupported layers or frameworks, or hardware and software limitations.

The methodology behind the framerate test is the following. The pre-trained Mobile Net v2 Howard et al. [2017], Inception v1 Szegedy et al. [2015] were used for classification task; Tiny YOLO v3 Redmon and Farhadi [2017] and MobileNet SSD v2 Howard et al. [2017]- for the object detection tasks. These neural networks were trained on the Common Objects in Context (COCO) dataset Lin et al. [2014]. The test image with ten recognizable objects was sent to the neural network's input 10000

Table 3.2: Comparative study of SBCs performance in computer vision tasks

Model	**Jetson Nano**	**RPi 3B**	**RPi 3B + NCS**	**RPi 3B + NCS2**	**Google Coral**
Classification					
MobileNet v1 (300x300)	64 FPS	2.5 FPS	7.5 FPS	30 FPS	**130 FPS**
Inception v1 (224x224)	55 FPS	N\A	5.8 FPS	12 FPS	**243.9 FPS**
Object Detection					
Tiny YOLO v2 (416x416)	**25 FPS**	0.5 FPS	2.57 FPS	5.1 FPS	N\A
MobileNet SSD v2 (300x300)	39 FPS	1 FPS	4.5 FPS	11 FPS	**48 FPS**
Segmentation					
UNet (1x512x512)	**18 FPS**	N \A	N\A	5 FPS	N\A

times; after that, the average FPS was computed. A similar test was performed with UNet Ronneberger et al. [2015]. However, it was pre-trained on the original dataset for cells' membranes segmentation.

According to Table 3.2, the Google Coral outperforms all the other candidate platforms. However, Jetson Nano and Raspberry Pi with various s demonstrate a better capability to run different types of s. In contrast, Google Coral cannot run some types of object detection and semantic segmentation neural networks. Moreover, for most of the precision agriculture tasks addressed in the current research, the framerate greater than 30 FPS is an overshoot. Even for aerial monitoring tasks. Furthermore, Google Coral does not support upsampling operation, which is the subject to discard it from the list of platforms candidates for semantic segmentation of plants from aerial imagery. Moreover, according to the Table 3.1, Google Coral has higher power consumption in comparison to the other platforms, it could become a limiting factor for all the precision agriculture applications, described in the upcoming sections.

Power Consumption

Power consumption is a crucial factor in the evaluation of the energy efficiency of the system. The monitoring platform will be further used in the laboratory, greenhouse environment, and payload for a UAV. In all cases, it will operate autonomously. Therefore it will be powered from a local power source as a battery. That is why the evaluation of power consumption under various conditions is essential. The average power consumption, measured by the manufacturer is not sufficient for monitoring tasks, since the monitoring platform operates in different modes, as hibernation and monitoring modes; therefore, idle and peak power consumption are necessary to estimate.

The methodology behind the investigation is the following. All the platforms were plugged to the 5V power source via the multimeter, connected inline. The multimeter has an accuracy of $\pm 0.01A$. In the case of Raspberry Pi, used with NCS1 and NCS2, the multimeter was used only in line with the Raspberry Pi's power input. Therefore, the power consumption by the whole platform was measured. However, the power consumption by NCS1/2 was not. According to the manufacturer's official data, the power consumption by NCS1/2 is 1W (as depicted in Table3.1, the power consumption by RPi 3B + NCS1/2 is the sum of RPi power consumption and the NCS1/2 power consumption). However, according to the study Sukholeyster [2020], the NCS power consumption depends on architecture and the number of Streaming Hybrid Architecture Vector Engine (SHAVE) cores used in operation. Thus, power consumption of NCS can vary from 0.7 to 1.7 Watts.

In the experiment, the testing platform was unplugged from the keyboard, mouse, monitor, and camera to evaluate the power consumption by the plain SBC only. During the test, the voltage and current under different operation modes were measured. Then, the power consumption was calculated by the formula: $P = I * U$, where P is the power consumption, I is current, and U is the input voltage. The results of the test are enlisted in Table 3.3. In addition three neural network architectures were used for power consumption estimation of the RPi and RPi with NCS: Inception v1 Szegedy et al. [2015], Tiny-YOLO v2 Redmon and Farhadi [2017] and MobileNet v1 Howard et al. [2017], see 3.4.

Table 3.3: Comparative study of SBCs power consumption in different modes

Model	**Jetson Nano**	**RPi 3B**	**RPi 3B + NCS**	**RPi 3B + NCS2**	**Google Coral**
Idle (W)	2.25	2.05	2.5	2.4	3
Peak (W)	6.1	5.25	4.3	4.55	4.8

Table 3.4: Comparative study of SBCs power consumption for different CNN inference on board of RPi and RPi with NCS

Model	**RPi 3B**	**RPi 3B + NCS**
Inception v1 (300x300)	N\A	3.71
MobileNet v1 (300x300)	2.2	3.195
Tiny-YOLO v2 (416x416)	2.659	3.675

According to the investigation, the Raspberry Pi has the minimum power consumption under idle mode. Therefore, it might be efficiently used in various applications, characterized by a prolonged hibernation mode period. Such applications might include seed germination prediction in thermal camera and time series prediction of tomato growth in a greenhouse. Raspberry Pi with NCS1/2 and Google Coral have smaller power consumption in the peak mode than Raspberry Pi. Nevertheless, they have more significant power consumption in idle mode. Therefore, these platforms are favorable in the application, where object detection takes a significant part of operation time. Jetson Nano has the highest power consumption among all the platforms. However, Jetson and RPi + NCS2 seem to be the best candidates for semantic segmentation of aerial data, since they can efficiently process FCNN on board, see Table 3.2.

Temperature Measurements

The evaluation of the CPU temperature is essential from the point of operation under various conditions. Any SBC might face the problem of overheating in service, even being under operating temperature. Overheating may lead to throttling the performance or even end up with switching off or rebooting the SBC. All the cases may end up with undesirable consequences, including data loss and operation interruption. To prevent overheating, one needs to anticipate it on the design stage. Therefore, estimation of the CPU temperature under various conditions is essential.

Table 3.5: Comparative study of SBCs internal CPU temperature

Model	**Jetson Nano**	**RPi 3B**	**RPi 3B + NCS**	**RPi 3B + NCS2**	**Google Coral**
CPU temperature (C^o)	53	83	RPi 51/ NCS 45	RPi 52/ NCS2 45	60
Operation temperature range (C^o)	-25 to 80	-40 to 85	0 to 40	0 to 40	0 to 50

The investigation methodology relies on the measurements by the temperature sensors on board SBCc. All the measurements were performed under the environmental temperature of 24 degrees Celsius. During the investigation, an SBC was isolated from heat sinks and sources. The data from the CPU temperature sensors was accessed from the Linux command line. All the temperature measurements were performed after 10 000 inferences of the MobileNet v2 (300x300). The recommended operating temperature, as well as the results of the experiment, are in the Table 3.5.

According to the results, the Raspberry Pi demonstrates the highest CPU temperature during the inference of the MobileNet v.2. At such temperature, it is exposed to performance throttling due to high temperature. Therefore, for reliable operation, it requires an external heat sink or active cooler. However, the application of an AI accelerator significantly reduces the external temperature from 83 to 51 degrees. It happens due to transferring all the calculations from the RPi CPU to the external VPU. The temperature of other platforms: Google Coral, Jetson Nano, and both NCSs are stable during the operation. However, only Jetson and Coral stay in the operating temperature range. Therefore, they seem to be the most reliable in terms of overheating; the Raspberry Pi requires an additional heat sink or external VPU for reliable operation.

3.1.4 Optimization of CNN for inferencing on low-power embedded device

This subsection reports on the optimization technique implemented for MobileNet v1, pre-trained on ILSVRC2012 dataset, Russakovsky et al. [2015]. It was tested on the Raspberry Pi 3B and Raspberry Pi 3B with NCS, Intel Movidius. The optimization technique relies on an adjustment of depth and input size of CNN. The input size is a resolution of the input image. In the experiment it takes the following values: 128, 160, 192 and 224. The depth is the characteristic, specified by parameter α, which takes the following values in the experiment: 0.25. 0.5, 0.75 and 1.0. This parameter influences the number of input and output channels for each convolutional layer of the MobileNet v1. For example, $\alpha = 1.0$ corresponds to original MobileNet v1 with no changes, 0.5 corresponds to the architecture with half of the input and output channels, and so on. Therefore α significantly influences the number of operations and accuracy of the neural network (see Fig. 3-1), either the input size of the image (see Fig. 3-2).

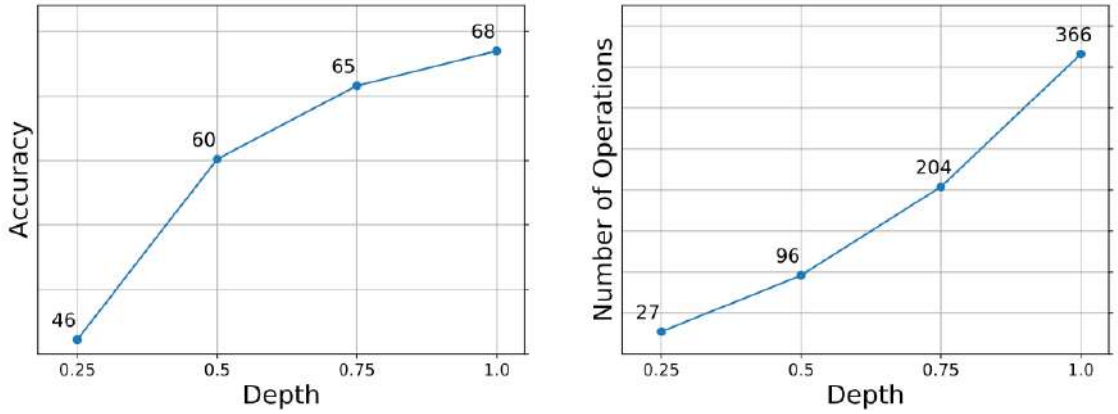


Figure 3-1: MobileNet classification accuracy and computational complexity with different depths.

Consequently, shrinkage of the input size and depth of the CNN results in a reduction of the number of operations. However, it also results in the accuracy reduction. Nevertheless, the right combination of these two parameters would reduce CPU and RAM load and increase FPS.

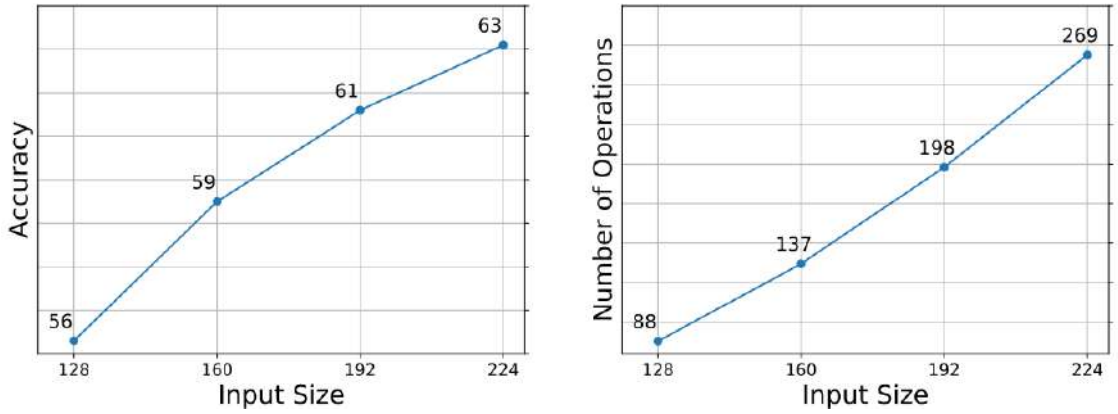


Figure 3-2: MobileNet classification accuracy and computational complexity against different input sizes.

FPS

Figure 3-3 shows the FPS value dependency on the parameters of executable neural networks. All the input images have a square shape. Therefore, the first number corresponds to the depth of the neural network and the second - to the input layer's size.

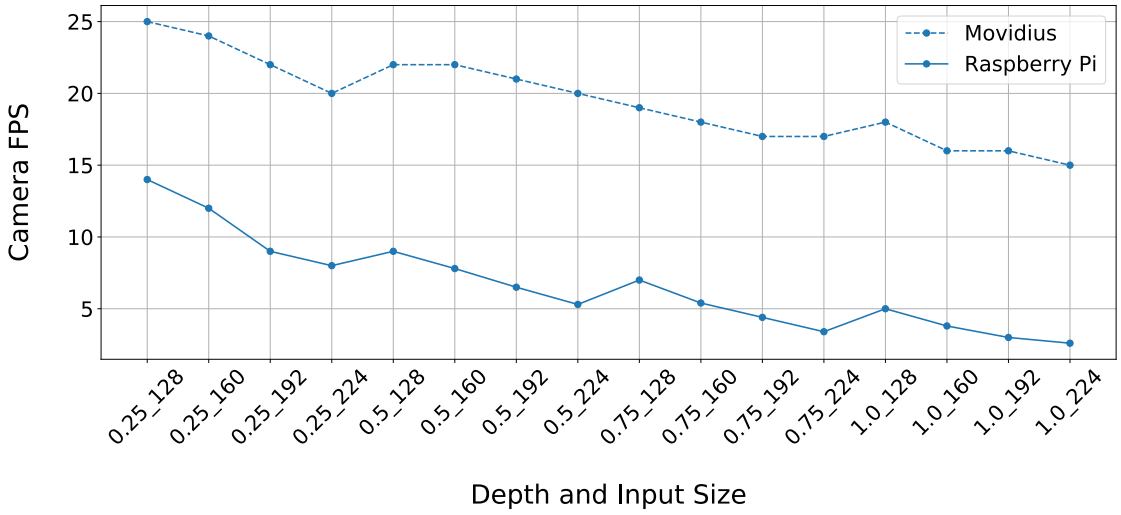


Figure 3-3: Different MobileNet FPS rates with and without Movidius.

For most experiments, the RPi with NCS (Movidius) FPS is around 20, while the mean value of RPi FPS is 7. In general, for an object detection related applications, 1 FPS is a good result. That is why RPi can be used even for more reliable MobileNet neural networks in a frames-per-second metric.

CPU and RAM load

Next, CPU and RAM loads were considered as shown in Figure 3-4 and Figure 3-5, respectively. In real applications, in addition to running a camera with the neural networks predicting labels, the user will most likely run other processes, e.g., storing it to a database or running other system scripts. Therefore, it is essential to consider that only running a heavy neural network on RPi can take up almost all CPU power and a considerable portion of RAM.

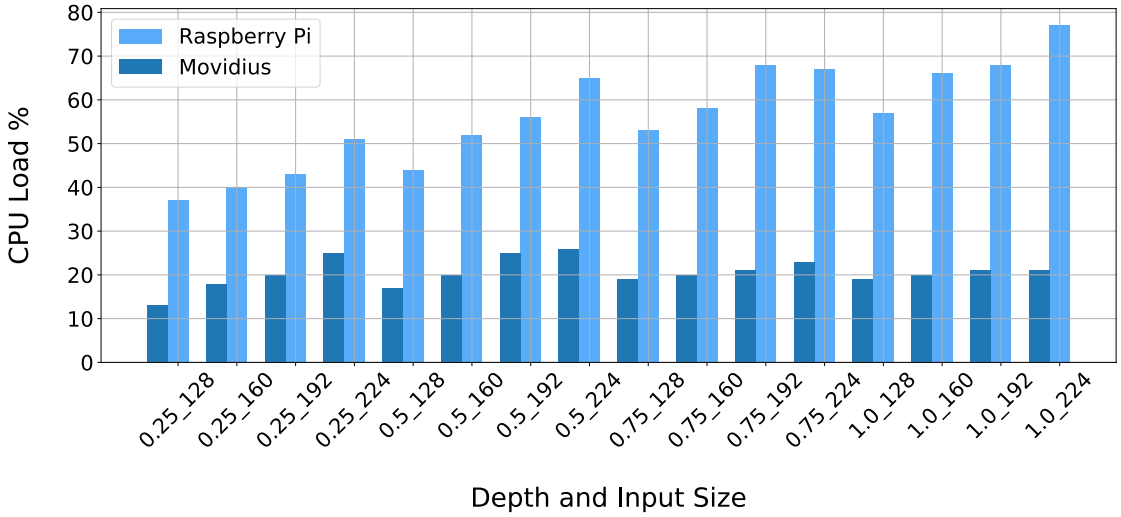


Figure 3-4: Raspberry Pi CPU load during MobileNet execution on Raspberry Pi and Movidius NCS.

Usage of NCS does not affect CPU functioning, and only 20% is used for the pre-processing image task. On the contrary, using only RPi takes an additional 20% to 60% of CPU usage, depending on the neural network size.

Figure 3-5 shows a comparative study on RAM Load while using the lightest and the heaviest models. Movidius requires the same amount of memory for both cases, which is around 22%. If using only RPi, the extra 10% is needed when launching a heavier model.

CPU Temperature

The most critical part of the experiments is to figure out whether the neural networks could be safely executed for an extended period. If the IoT device has suf-

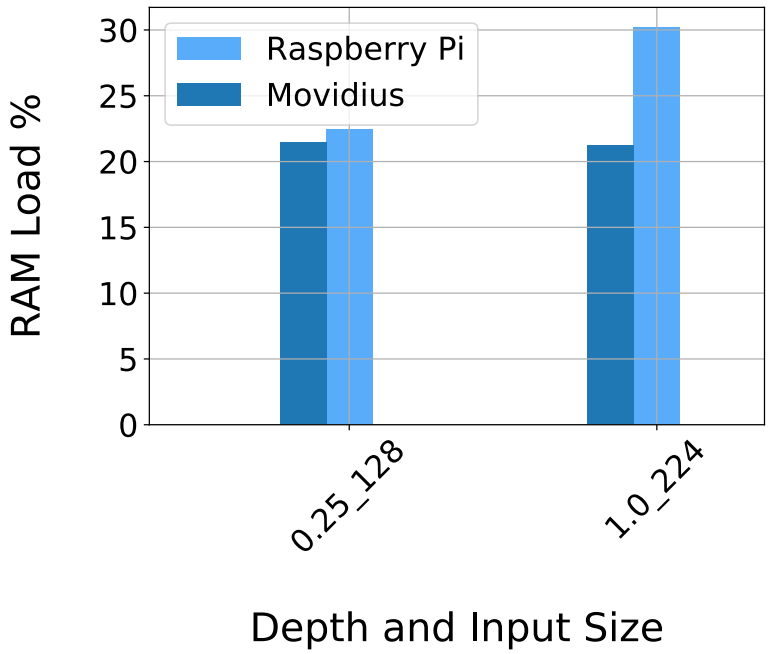


Figure 3-5: Raspberry Pi RAM consumption during the MobileNet execution on Raspberry Pi and Movidius NCS.

ficient power supply, the only thing that should be avoided is overheating. CPU Temperature analysis, while using only RPi and Movidius, is depicted in Figure 3-6.

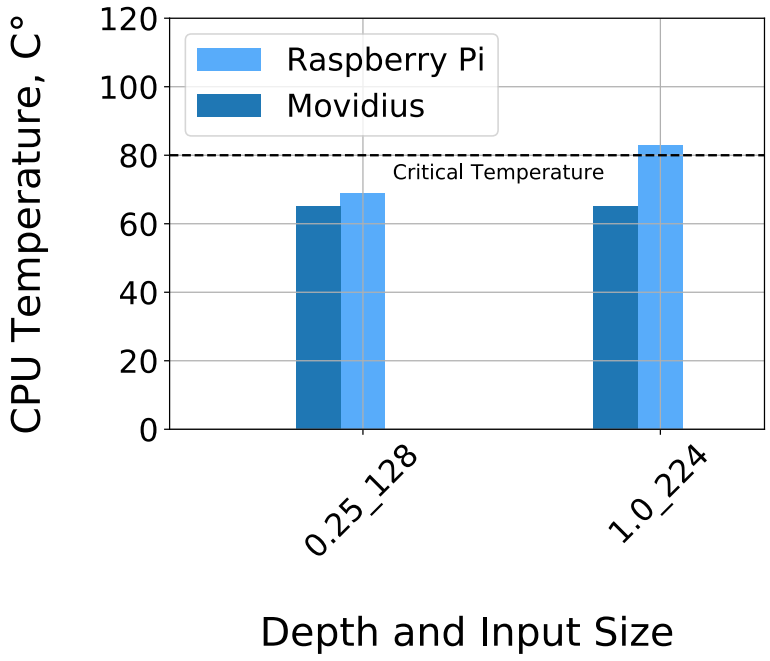


Figure 3-6: Raspberry Pi CPU package temperature during the MobileNet execution.

The CPU Temperature is compared for cases with the lightest and the heaviest models on board. The Movidius CPU in operation heats up to 64° in both experiments. However, executing the same neural networks only on RPi results in overheating in the second experiment with the largest model. It means that if this experiment continues over an extended period, the problem of performance reduction will unavoidably appear. It will also result in increased power consumption and CPU performance throttling.

In the case of the MobileNet v1 set of neural networks, one can execute up to the depth equal to 1 and input size equivalent to 160 models on RPi without Movidius. Heavier models (with the depth 1.0 and input sizes 192 and 224 pixels) result in overheating, and the use of Movidius is necessary in this case.

3.1.5 Conclusions

To sum up, the research demonstrates that even platforms with low computational performance might be useful for specific tasks and that high performance is not necessarily the critical factor. Low power consumption and better cooling efficiency might significantly influence real-life operation. Thus, a platform might be useful in some cases and useless in others. Therefore, it was demonstrated that plain Raspberry Pi might be used as a monitoring platform for applications with prolonged periods of hibernation and low requirements to framerate. It might be useful for object detection and classification tasks like seeds germination monitoring and tomato growth prediction. Similarly, the Raspberry Pi with an external AI accelerator, either NCS1 or NCS2, might be used in the same tasks. It is due to a higher framerate and reduced probability of overheating. However, it results in higher power consumption for both idle and peak modes. Raspberry Pi with NCS2 could also be used for semantic segmentation tasks; however, Jetson Nano was chosen for this role due to higher framerate, which is essential for real-time aerial monitoring tasks. Google Coral demonstrated exceptionally high framerate for classification and object detection tasks. However, such a high framerate is an overshoot for the abovementioned seeds' germination monitoring and tomato growth prediction tasks. Furthermore, this platform has higher power consumption then Raspberry with var-

ious NCSs. On the other hand, such a high framerate could significantly improve the efficiency of aerial monitoring platform. Unfortunately, it does not support the upsampling operation, which is essential for semantic segmentation. Thus, Google Coral was discarded from the list of the platforms candidates for further investigations. Odroid XU4 was also discarded from the list because it does not outweigh other platforms.

The optimization technique demonstrated in the current section resulted in a significant reduction of CPU load (up to 40% for plain Raspberry Pi and up to 17% for Raspberry Pi with NCS). It also resulted in an 8% RAM load and 14 degrees CPU temperature reduction for plain Raspberry Pi. However, the improvement of the RAM usage and CPU temperature for RPi with NCS was insignificant. This technique was applied for pre-trained classification CNN on two platforms only. However, the same technique was applied to the semantic segmentation task on board of Jetson Nano - the results are reported in section 3.4. The results of this section will be used as a basis for further intellectual monitoring platforms development in sections 3.2, 3.3 and 3.4.

3.2 Detection of Seeds During Germination

3.2.1 Introduction

In this section, an intelligent embedded system for the seeds recognition and germination detection is presented. The proposed system is a sensor node characterized by sensing, processing, and communication capabilities with a particular focus on data processing. For this reason, we collected a dataset in an industrial chamber and designed a CNN consisting of 2 convolutional blocks, 2 linear blocks, and a sigmoid block. To process the images, we equipped the embedded system with an external GPU. We managed to achieve 97% accuracy of seeds recognition accuracy and 83% of the average IoU score. Simultaneously, the proposed solution takes advantage of scalability, small size, and the ability to be powered by batteries, therefore ensuring autonomous intelligent operation in the forthcoming IoT era.

The research, described in this section, was accomplished in close collaboration

with Dmitrii Shadrin, a Ph.D. student at Skoltech. He contributed the following parts of the study: setting up a continuous experiment, data annotation, and training the CNN. My part of the research includes squeezing the pre-trained CNN, implementing the embedded monitoring system with AI capabilities, testing the embedded platform, and algorithm in the experimental environment (climate chamber). The abovementioned contribution by Dmitrii Shadrin is also part of his Ph.D. thesis "Data-driven modeling of plant growth dynamics in controlled environments." Also, the results of this section were published in the IEEE Sensors journal Shadrin et al. [2019b].

3.2.2 Methodology

In this section the methodology used in the present work is described.

Approach

The methodology includes a number of important steps needed to detect the seeds germination process:

- Set up of a continuous experiment for the seeds image data collection. Within this step, we combine the pictures with timestamps to follow the germination process. The pictures are collected continuously with a predefined period.
- Data annotation, needed to provide the dataset with the examples of positive and negative samples.
- Training of the CNN algorithm based on the collected data. After the extraction of the patterns from the data process, the algorithm can search for patterns in new previously unseen data.
- Assembling the embedded platform equipped with external VPU-based AI accelerator and running the designed CNN on it.
- Detection of the seeds using CNN, followed by the detection of germinated seeds using computer vision techniques.

- Test of the embedded platform and algorithm in the climate chamber.

3.2.3 Implementation of the Smart Sensing Platform

The goal of the investigation is to create intellectual monitoring platform for seed germination recognition. Seed germination is a relatively slow process, and a reasonable statistics collection might be achieved even daily. However, in this study, we collected the data every 3 hours because HD camera resolution is still too low to detect reasonable changes in subsequent frames. Moreover, data collection within a smaller period would result in error accumulation for time series prediction. Therefore, the predictions will be made within 3 hours period (see Fig.3-7. Thus the platform needs to stay in a hibernation mode for 3 hours than collect the data and make a prediction. The platform should withstand overheating inside the climate chamber, be reliable to operate continuously for 36 hours and perform data-intensive calculations on board. Relying on the experimental results from the section 3.1, the Raspberry Pi (RPi) with Neural Computer Stick (NCS) "Intel Movidius" is the most appropriate platform for this task.

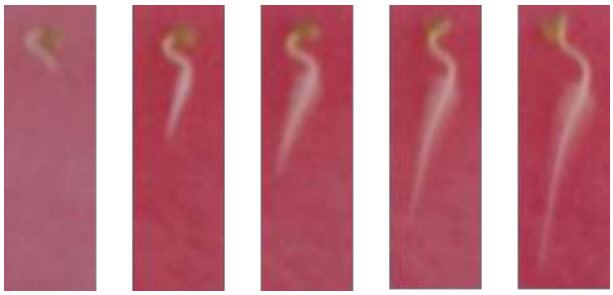


Figure 3-7: Photos of seeds germination process taken with 3 hours period.

In this subsection, the platform with the embedded AI aboard the low-power embedded sensing system (see Figure 3-8) is proposed. It is based on RPi 3B single board computer and NCS (an external VPU). The platform can easily run the pre-trained CNNs. Although it initially had restricted computational capabilities, the external VPU significantly expands the platform performance. It has 12 sharp vector engines, which results in over 100 GFLOPS performance at power consumption 1 W only. Its 450 Gb/s carrying capacity allows to process data from 8 High Definition

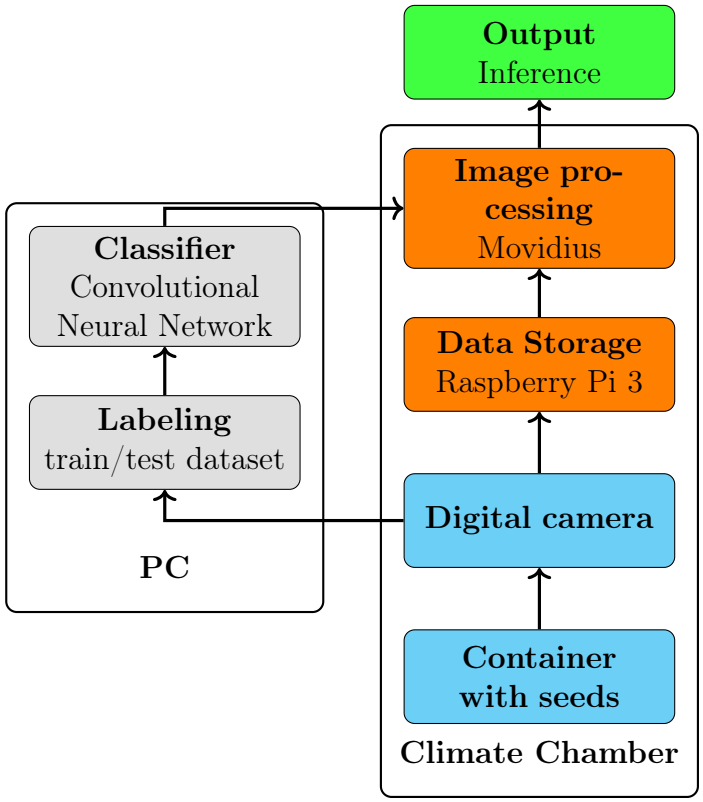


Figure 3-8: System block diagram

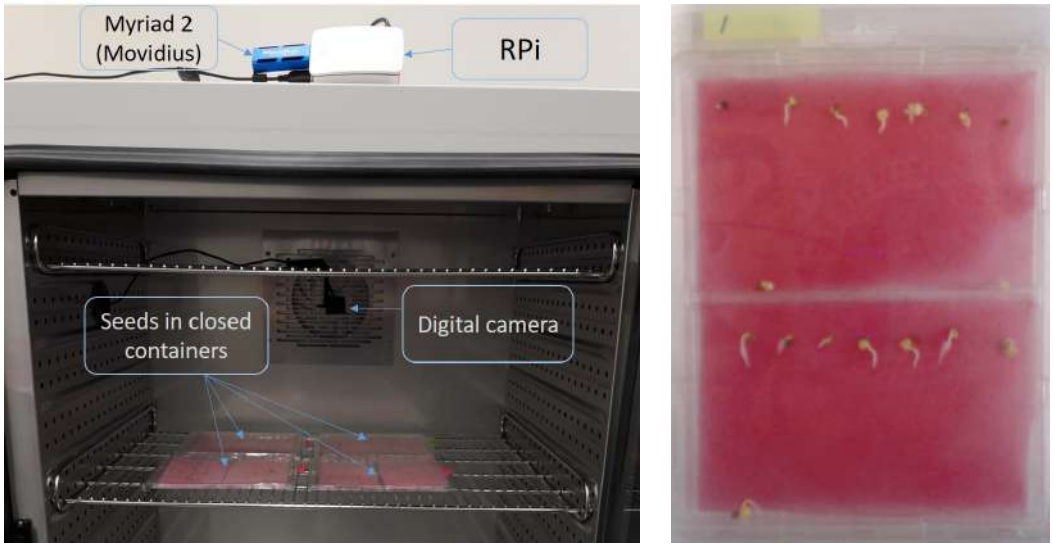
(HD) web camera sensors simultaneously. An external battery powers the prototype. The only sensor used in the prototype is an HD web camera, Fig.3-9.

The proposed platform successfully works with the pre-trained NNs. The CNN used in the investigation was trained on a desktop computer using Python with a PyTorch library. The resultant code was converted to the Caffe model by open source software and was finally compiled into the binary graph file using Intel's OpenVINO library.

The Neural Network Architecture

In this work, the PyTorch framework Paszke et al. [2017] is used to train and evaluate the performance of CNN. The structure of the CNN applied is summarized in Table 3.6.

The CNN showed in Table 3.6 includes 2 convolutional blocks, 2 linear blocks, and a sigmoid block. The convolutional block consists of the following layers: convolutional, batch normalization, max pooling, ReLU activation, and dropout. The



(a) (b)

Figure 3-9: Climate chamber (a) Embedded system assembly for testing of Machine learning (CNN) algorithm; (b) Example of photos of the containers with seeds taken during the experiment.

Table 3.6: Convolutional Neural Network Architecture

#	**Layer**	**Dimension**			**Kernel**	**Stride**
		Width	*Height*	*Depth*		
0	Input	90	90	3	-	-
1	Convolution	40	40	48	11	2
2	Normalization	40	40	48	-	-
3	Pooling	20	20	48	2	2
4	ReLU	20	20	48	-	-
5	Dropout(p=0.1)	20	20	48	-	-
6	Convolution	16	16	96	5	1
7	Normalization	16	16	96	5	1
8	Pooling	8	8	96	2	2
9	ReLU	8	8	96	-	-
10	Dropout(p=0.1)	8	8	96	-	-
11	Fully Connected	1	1	100	-	-
12	Normalization	1	1	100	-	-
13	ReLU	1	1	100	-	-
14	Dropout(p=0.1)	1	1	100	-	-
15	Fully Connected	1	1	100	-	-
16	Normalization	1	1	100	-	-
17	ReLU	1	1	100	-	-
18	Dropout(p=0.1)	1	1	100	-	-
19	Fully Connected	1	1	1	-	-
20	Sigmoid	1	1	1	-	-

linear block includes the fully connected, batch normalization, ReLU activation, and dropout layers. The sigmoid block is composed of the fully connected and sigmoid activation layers. It is assumed that the values from the previous layer are the input for the next one. The convolutional layer takes the input frame and sums the frame values with weights. It helps extract the local features of the input. The batch normalization layer centers the input with the mean and scales it with the dispersion. It is used to make the convergence stable. The max-pooling layer takes the input frame and finds the frame maximum. It helps extract the local peaks and compress the input. The ReLU activation layer turns the negative input values into zeros. It speeds up the convergence process. The dropout layer turns the input values into zeros with the probability p. It makes the network robust to overfitting. The fully connected layer sums all the input values with weights for every value in it. It helps find the meaningful connections between the input values. The Sigmoid activation function of x is $\sigma(x) = \frac{1}{1+e^{-x}}$. It is used for converting the values into probabilities. When, for example, we take the picture frames 90x90x3 ($pixels \times pixels \times channels$) as the input, we get the probability belonging to the seed class as the output.

Image Processing Workflow

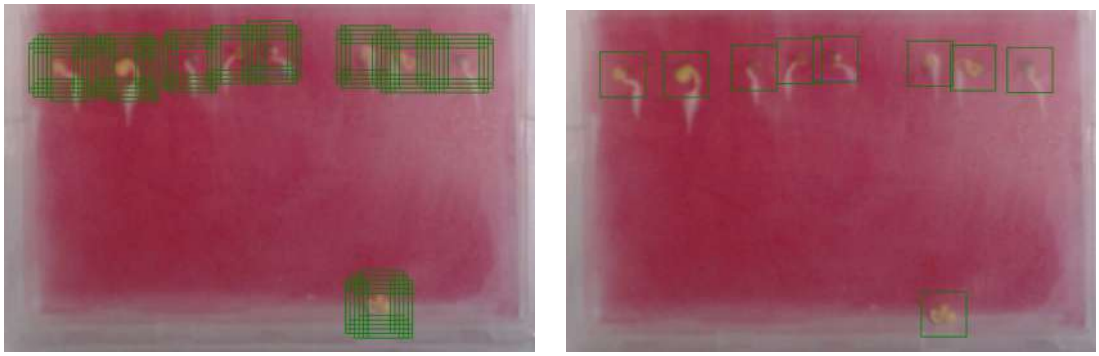
The proposed CNN architecture described in previous sections was trained to perform the seeds recognition task ('exist or not exist') in the predefined window area (90×90). In the current study, the recognition and localization of the seeds were performed by CNN with a sliding window technique Noh et al. [2016]. It had 90×90 windows that overlap less than 90%: every next window was obtained from the previous one by a small shift in the horizontal or vertical direction. After cropping images from the picture, they were provided to the pre-trained CNN input to recover the labels (existing or not existing seed). All the 90×90 pixels windows with positive labels ('seed exists') were combined, and the non-maximum suppression technique Oro et al. [2016] was applied to them. This technique allows us to join close images (windows) to avoid multiple references to the same location in the picture. The result is presented in Fig. 3-10. Once the pictures covered by the windows with seeds are obtained, the Intersection over Union (IoU) got calculated for them.

After that, CNN predicts the germination of the seeds. It was performed with the computer vision techniques that make it possible to calculate the number of white pixels referring to sprouts and compare pixels to the manually defined threshold.

Fitting the Neural Network to the Embedded Platform

In the current study, the Intel Movidius NCS was used, which could be classified as a multicore Very Long Instruction Word (VLIW) AI accelerator with a video fixed function unit. VLIW allows the implementation of Instruction Level Parallelism (ILP), which implies the ability to implicitly specify instructions to run in parallel, whereas CPU can execute instructions in a sequence. VLIW processors use the software to decide in advance what instructions could be run in parallel. That is why the complexity of the instruction scheduling is moved to the compiler, leading to a significant reduction in the hardware complexity. In the current case, it allows shrinking of the size in a volume of USB flash memory stick. The VPU is Myriad 2 processor, its architecture relies on 16 Streaming Hybrid Architecture Vector Engine (SHAVE) cores SHA [2019]. These processors execute instructions over all the elements of 1-D objects - vectors. It significantly increases the speed and throughput of the whole device, especially for video and image processing tasks.

In addition to the CNN inference acceleration by the hardware, the software optimization techniques were also applied. All these techniques were implemented in the OpenVINO toolkit provided by the manufacturer. These techniques include quantization, freezing, and fusion. The quantization reduces the memory required for neural network storage and inference by changing the computer number format. For example, it can convert all the neural network weights from 16-bit Floating Point (FP16) to 8-bit Integer (INT8), hence reducing the memory requirements and inference time. However, it also results in an accuracy reduction, but usually, this accuracy drop is tolerable. Freezing the graph is the technique that reduces all the unnecessary operations, which are usually used during the training, but are not used during the inference. Fusion is the technique, which combines subsequent layers into a single layer. For example, it can combine convolutional, normalization, max-pooling, and ReLU layers into a single one, reducing the overall number of



(a) (b)

Figure 3-10: Non-maximum suppression application. In (a) seeds are covered with multiple windows, while in (b) we used one window per seed. This was obtained by grouping the windows and keeping one window per group.

multiplication and addition operations.

3.2.4 Experimental Results

CNN Performance Evaluation

The CNN was trained for 50 epochs (iterations) with the cross-entropy loss for the train and validation datasets shown in Fig. 3-11a. For each epoch, the random horizontal and vertical flips and color jittering were applied to ensure the data augmentation. The accuracy of this validation is more than 97%. (See '3-11b It means that CNN mismatches 3 windows with seeds or background out of 100. In the test phase, the method described in Section 3.2.3 was applied. The results are shown in Fig. 3-12. The quality of the model was assessed by the average IoU between the predicted windows and the ground truth windows as follows:

$$\frac{\sum_{\forall\{w_g,w_p\}} \mathrm{IoU}(w_g, w_p)}{\#\{w_p\}}, \tag{3.1}$$

where

$$\mathrm{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}, \tag{3.2}$$

and $\forall\{w_g, w_p\}$ is all the possible pairs of ground truth and predicted windows, while $\#\{w_p\}$ is a number of predicted windows. As a result, the average IoU of 0.83

was obtained.

Germination Detection

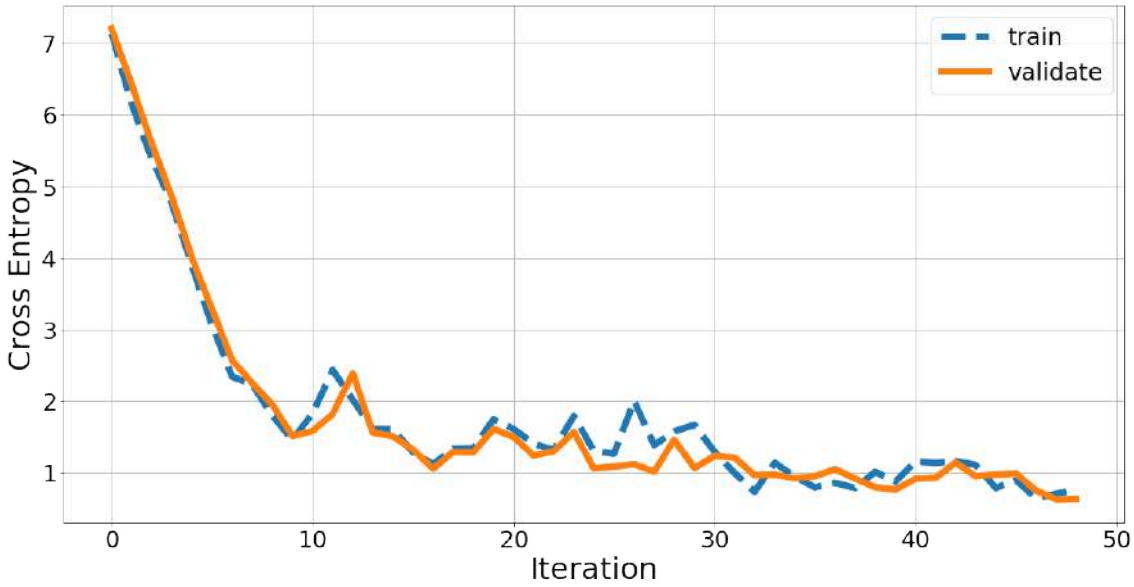
A typical problem associated with the detection of seeds germination by images and the application of traditional computer vision algorithms is to distinguish the white pixels belonging to sprouts. These pixels may belong to recently germinated seeds or other objects close to white pixels, e.g., the objects resulted from humidity and appeared on the container walls. The key feature of the designed CNN is to propose the regions for detecting further germination within the areas identified at an early stage. Figure 3-12 shows that all the seeds germinated within a specified period are characterized by the reasonable quality of the seeds used. In most cases, the germination rate is 80-90%.

Fig. 3-13a demonstrates the example where the seeds have been localized in a container using CNN. Later on, the germinated seeds out of the localized ones were recognized.

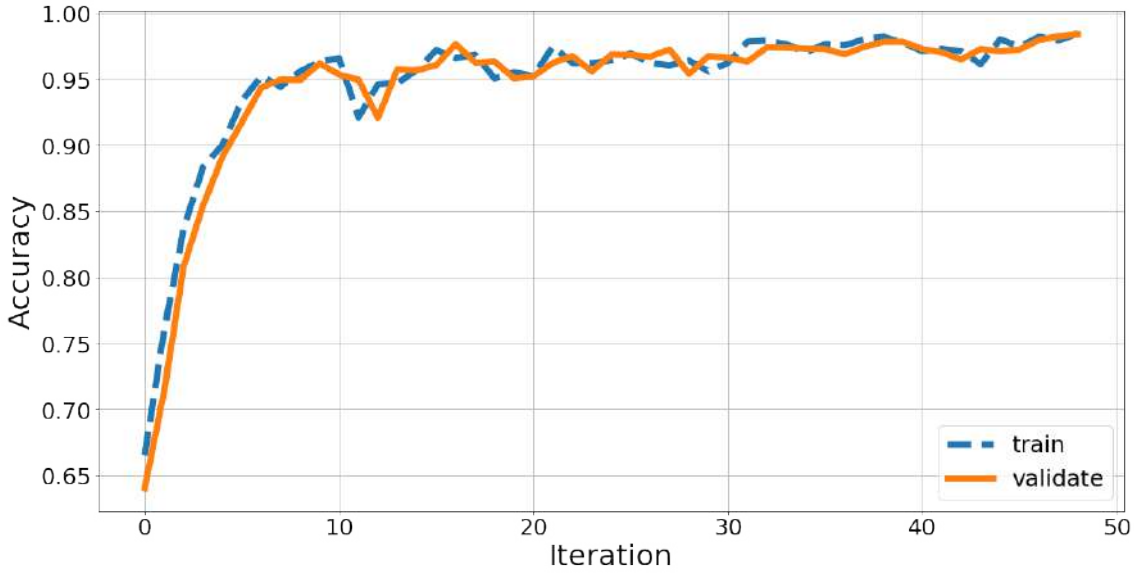
For this reason, the following algorithm was developed. It is based on the Pythons library *skimage*. The algorithm was then applied in each of the bounding boxes and functioned as follows:

- Converting the RGB image into the grey-scale one.
- Using the Otsu algorithm (for each of the proposed regions) for binarization of the image.
- Obtaining the grey-scale morphological closing of the image.
- Obtaining the bounding boxes for each instance. We put the lower 100 pixels threshold while the seeds with more than 100 white pixels are assumed to germinate.
- Presenting the instances: the germinated seeds and background.

Fig. 3-13b shows the proposed methodology outcome: five seeds out of all seeds in the container are recognized as germinated on the 26-th hour after starting the experiment.



(a)



(b)

Figure 3-11: Cross entropy loss(a) and accuracy(b) on CNN training for 50 iterations(epochs)

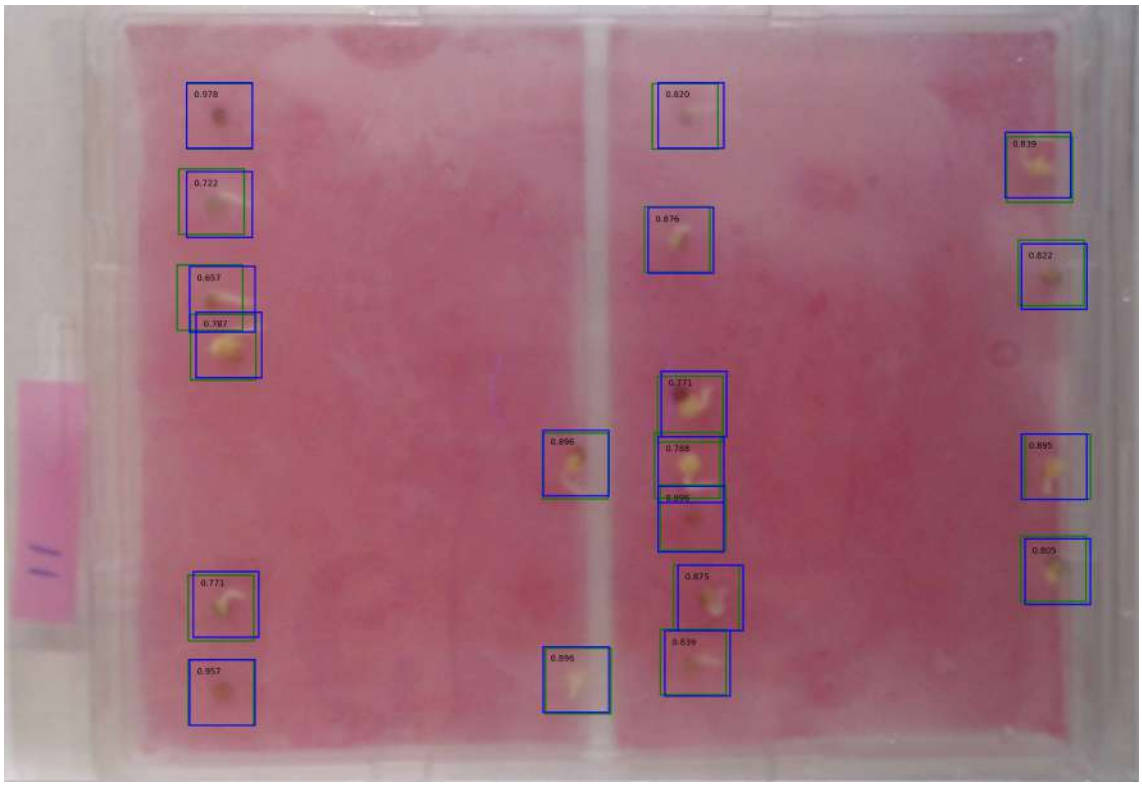
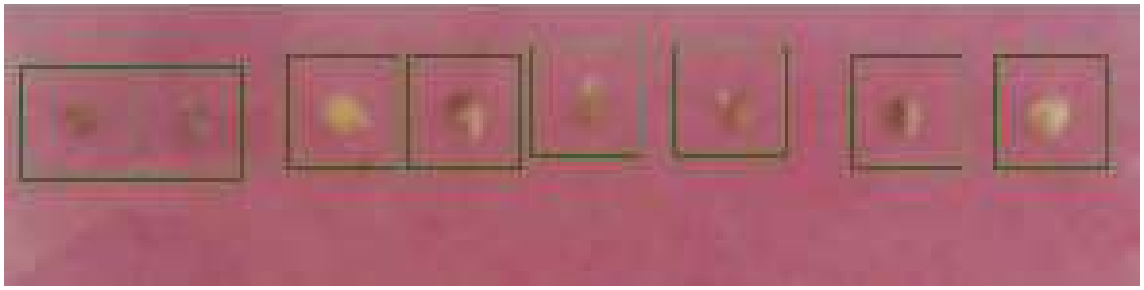
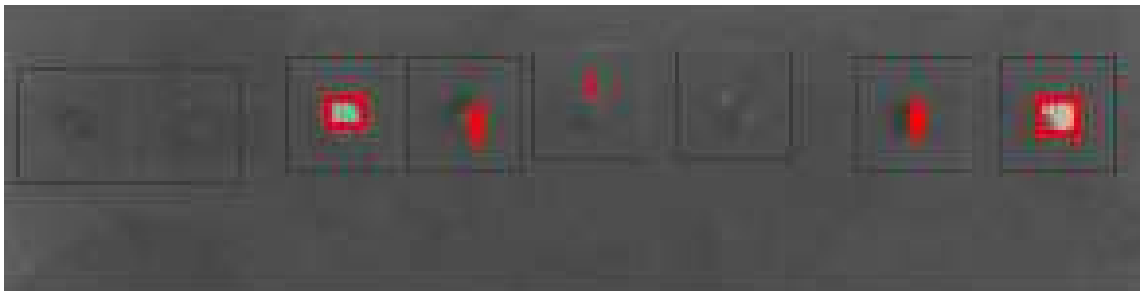


Figure 3-12: Seed recognition in containers. Green boxes - ground truth, blue boxes - estimated bounding boxes



(a)



(b)

Figure 3-13: Detection of seed germination (b) in the regions proposed by CNN (a).

Intellectual Monitoring Platform Performance Evaluation

The CNN described in Section 3.2.3 was trained on the desktop computer. The network was trained on the dataset collected during the seed germination experiment in the climate chamber. The CNN achieved 97% accuracy and 83% mean IoU at the validation and test stages of the experiment. In addition to the evaluation of multiple SBC, demonstrated in section 3.1, an extra performance evaluation was accomplished. The prototype (RPi 3B with NCS) was assessed against the laptop (see specification in the Table 3.1) implementation to show reasonably good computational efficiency.

The experiment provided for the following steps:

- Analysis of the overall computational performance
- Time estimation of a single prediction
- Estimation of power consumption

The test dataset, as well as the pre-trained NN, were uploaded to both platforms. Then the algorithm runs the CNN to perform 1000 iterative predictions. The following criteria were used for performance evaluation:

- CPU and RAM usage.
- Time of prediction.
- Power consumption.

All these characteristics, timestamp, current, and voltage, were extracted simultaneously and were written in a single CSV file during the experiment. The entire code for data collection was created in Python 3 using the "psutils" library to collect CPU and RAM data. The power consumption data were collected in different ways. For the laptop, the "powertop" python library was used to collect power consumption data directly from the battery controller. Since the above approach to power measurement with the software does not apply to Raspberry Pi-based prototypes, similarly to the investigation in section 3.1, the multimeter was connected in line to the power plug of the RPi to collect the current and voltage data.

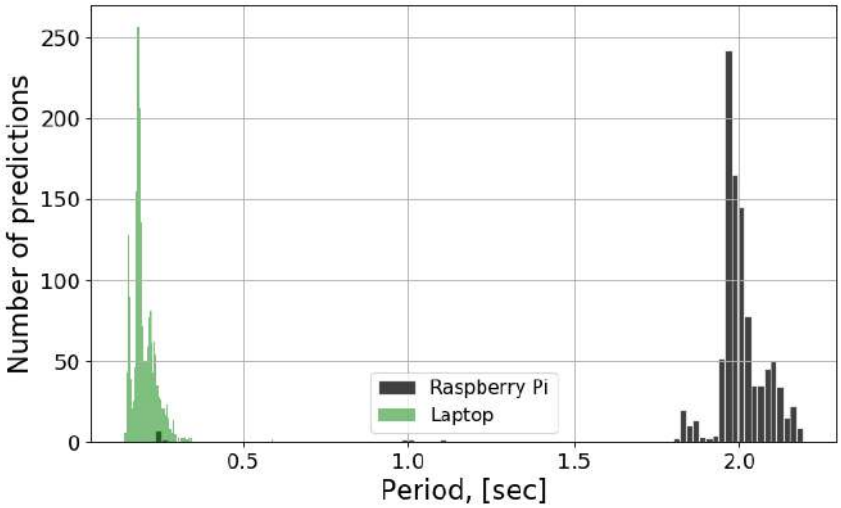


Figure 3-14: Time spent per a single prediction over the entire frame for both platforms

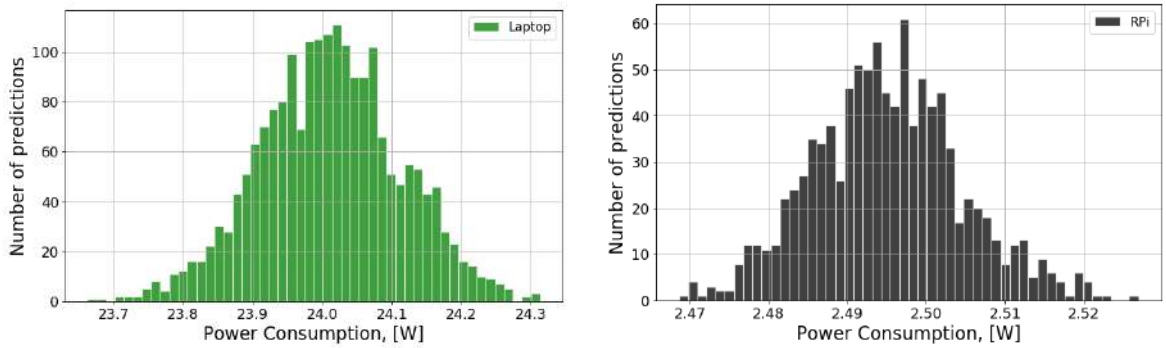
The period is the characteristic that shows the time per single prediction over the whole frame. Moreover, the laptop is 10 times faster (see Fig.3-14), so that we have 0.85 FPS for the embedded system against 4.5 FPS for the laptop when running the same code with NN and germination detection algorithm described above. It is the germination detection algorithm that significantly slows down both of the machines: Raspberry Pi and laptop has 4.5 FPS and 10.8 FPS, respectively, while running the same code with CNN only. That is why the development and training of advanced NN are subjects of future analysis.

Even though the laptop is faster than the proposed testbed, it has higher power consumption: its mean power consumption is 24.01 W in comparison with 2.5 W for the testbed (see Fig.3-15). The advantage of the proposed prototype - the small and low-power platform with AI on-board could be used in tasks where the availability of power supply, mass, and dimensions of the platform the characteristics that count.

The embedded system's computational performance with the NCS is lower than the laptop due to the following reasons. First, the laptop has better computational power capabilities comparing to the embedded system. Second, RPi has the USB interface version 2.0, while the NCS has USB version 3.0. Therefore, the computational performance of the embedded sensing system is restricted by the USB 2.0 data-carrying capacity: USB 2.0 is 480 Mbps, and USB 3.0 is 5 Gbps maximum

Table 3.7: Performance Evaluation for the Computer and Embedded Intelligence Prototype

Parameter	**Laptop**	**Mobile Prototype**
CPU usage, %		
Mean	21.99	37.04
Median	21.81	37.2
Minimum	14.85	16.0
Maximum	35.78	49.0
Mode	18.85	38.0
SD	3.03	2.23
RAM usage, %		
Mean	19.48	30.67
Median	19.5	30.7
Minimum	19.1	29.1
Maximum	19.8	31.5
Mode	19.5	30.7
SD	0.20	0.38
Period, [s]		
Mean	0.20	1.98
Median	0.19	1.99
Minimum	0.14	0.23
Maximum	0.59	2.19
Mode	0.18	1.98
SD	0.035	0.18
Power consumption, [W]		
Mean	24.01	2.50
Median	24.01	2.49
Minimum	23.66	2.47
Maximum	24.31	2.53
Mode	23.66	2.46
SD	0.10	0.01



(a) (b)

Figure 3-15: Histograms of power consumption for (a) the laptop and (b) the embedded system equipped with NCS.

Axelson [2015].

The major disadvantage of desktop computers and laptops compared with SBCs is their high power consumption and low mobility. The proposed prototype has 10 times lower power consumption and can be powered even by an off-the-shelf battery. The following formula could calculate the required capacity of the battery: $E = \frac{T*P}{U}$, where T is the time of operation, P is the power consumption, and U is the input voltage. Assuming that the monitoring platform is plugged into 5 Volts power source and works continuously for 36 hours consuming 2.5 Watts, the battery's required capacity will be 18 mAh. It seems reasonable for a 36 hours experiment, and power banks with such capacity are commercially available.

3.2.5 Conclusion

This work demonstrates the approach to the inference of DNNs on a low-power embedded system for detecting the seed germination dynamics without involving extensive data transmission from the local nodes to a cloud computing server. The proposed approach is scalable and has an industrial impact as a powerful tool for assessing the performance of growing systems and predicting their future harvesting period. Simultaneously, it provides an opportunity for making optimization at the initial stage of plant growth. This optimization will further result in the optimal management of resources in the context of precision agriculture. For implementing the proposed approach, the dataset Shadrin [2018] was collected. It contains the

sequentially time-ordered images of seeds germination process at different stages. With this end in view, we have proposed a custom CNN architecture for the seeds recognition, which achieves the 97% accuracy and 83% of IoU.

Using the CNN and computer vision, the sensing system can first localize them in the container and, second, detect the germinated seeds. This solution is implemented on the embedded system equipped with the NCS, which runs the CNN on board. The proposed monitoring platform's performance with AI capabilities against a laptop was assessed within the experimental work. The laptop is about 10 times faster than the embedded system, but an 18 mAh battery could power the last one for 36 hours of continuous operation. It is beneficial for the emerging autonomous applications in the scope of IoT and precision agriculture.

3.3 Detection of Plants in Greenhouse

3.3.1 Introduction

This section reports on the development of the low-power embedded system enriched with the AI-based on Fully Convolutional Neural Network (FCNN) and Recurrent Neural Network (RNN) Hochreiter and Schmidhuber [1997] called Long-Short Term Memory (LSTM) for continuous analysis and in-situ prediction of the growth dynamics of plant leaves. The monitoring platform is grounded on a low-power embedded sensing system with a GPU and can run the neural networks on board. The proposed approach guarantees the continuous autonomous system operation using a standard Li-ion battery.

This study also shares with the research community the *Tomato Growth* dataset Shadrin et al. [2019] collected during the research. The dataset contains 5514 time-sequenced top-down images of plant growth and growth conditions. It can be used for training the CNN for solving plant detection problems, segmentation, leaf area estimation, and plant growth dynamics assessment. This study opens up a wide vista for various intelligent monitoring applications, especially in the agriculture domain.

The research, described in this section, was accomplished in close collaboration

with Dmitrii Shadrin, a Ph.D. student at Skoltech. He contributed the following parts of the study: setting up a continuous experiment, data annotation, and training the LSTM. My part of the research includes squeezing the pre-trained LSTM, implementing the embedded monitoring system with AI capabilities, testing the embedded platform and algorithm in the experimental environment (the greenhouse). The abovementioned contribution by Dmitrii Shadrin is also part of his Ph.D. thesis "Data-driven modeling of plant growth dynamics in controlled environments." The results of this section were published in the IEEE Transactions on Instrumentation and Measurement Shadrin et al. [2019a].

3.3.2 Methodology

Using RNNs for time-series prediction

The design of industrial plant-growth processes is typically based on the expert knowledge, e.g. of micro-climatic requirements, contents of nutrient solution, irrigation schedule, as well as on the historical experimental data. This approach does not scale and is not generic in terms of production size and crop variability.

In this subsection, the RNNs are used for the prediction of the dynamics of plant growth. The RNN is a class of neural networks where the nodes contain the feedback response and store the information about their internal state. One of RNNs attractive features is that they can potentially link previous knowledge with the current state. The RNN can process the data that is represented as time-dependent sequences by using the internal state information. A typical RNN may have a problem with the processing of long-term dependencies. To overcome this problem, the Long-Short Term Memory (LSTM) NN were introduced as a particular architecture of RNN Hochreiter and Schmidhuber [1997] capable of learning long-term dependencies. The critical element of LSTM is a cell state, which can be changed during training. This feature is essential for modeling the plant growth dynamics since the future dynamics of the plant growth are in strong relation with the previous states passed a long time before.

Recently, lots of applications for the LSTM NN architecture have appeared Stol-

lenga et al. [2015]. However, for the *precision agriculture*, the application of RNN LSTM for crop yield prediction or description of plant growth dynamics based on the environmental growth conditions is a novel research direction Chlingaryan et al. [2018]. In current work, the LSTM is a core of AI. It is described in detail in the next sections.

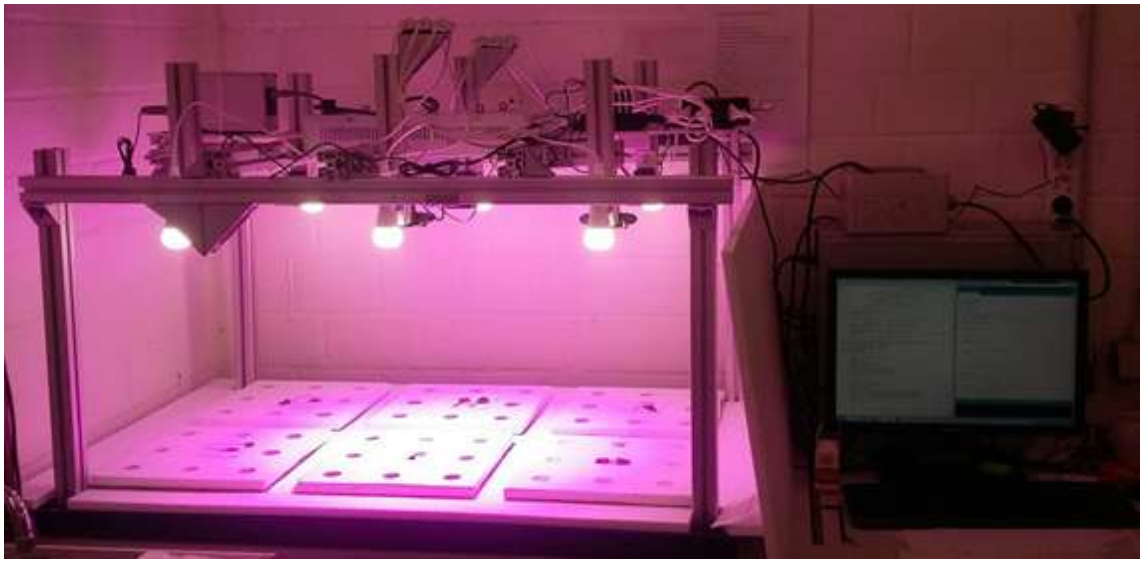
Data Collection

The data collection system was designed to perform plant growth modeling and dynamics prediction (see Figure 3-16a). It is based on the hydroponic approach allowing the simultaneous plant growth in different conditions (nutrient solutions). This testbed was equipped with an automatic image acquisition system and controlled LED illumination. The sequence of raw images was obtained in the one-month experiment on the plant's growth. It contains the data on the plant's growth with the fixed time interval under different conditions. In total, 5514 images with 30 minutes timestep were collected (see Fig. 3-16b). The FCNN was used for calculating changes of the leaves area in time-based on the obtained images. The data on the leaves area was used for training the LSTMs. Afterward, the ablation study was performed. It is required to find the optimal configuration of NNs since they will be squeezed to the low-power embedded system.

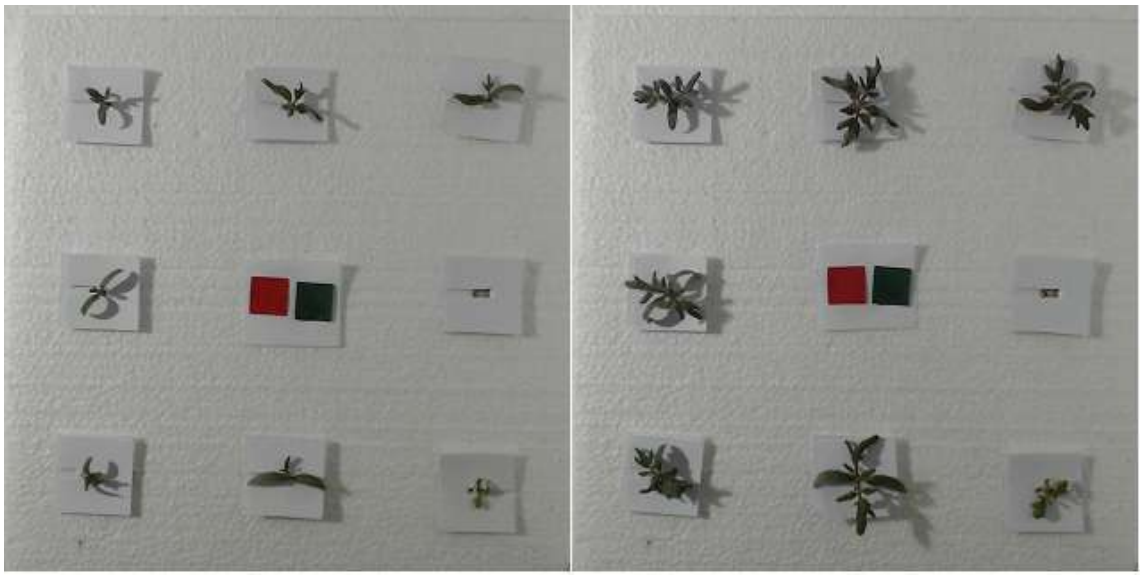
Training the Neural Network

The numerical experiments were conducted using the data for 12 days. Then the LSTM model was trained for each section. The dataset was split into the training and test sets with 400 and 200 elements for each section.

For optimization the adam optimizer Kingma and Ba [2014] was used. And as a loss function the mean squared error was used. For adam optimizer the following hyper-parameters were applied: $lr = 0.001$ (learning rate); $beta_1 = 0.9$ (exponential decay rate for the first moment); $beta_2 = 0.999$ (exponential decay rate for the second moment); $epsilon = 10^{-8}$. The hidden states for each epoch of training were reset. The network was trained for 10 epochs, which is a reasonable amount for stabilizing the loss function for most tested architectures with one hidden layer of LSTM.



(a)



(b)

Figure 3-16: (a) Experimental setup: growing (bottom) and data collection system1 (see video cameras on top); (b) Example of top-down photos obtained in the experiment. On the left: tomatoes on the 5-th day after germination. On the right: tomatoes on the 10-th day after germination.

Different amount of points from the previous steps were used. It was realized that 3 points set are enough for making accurate predictions even for a 3-hour horizon. The train/test dataset was created in the following way. Each train/test data sample contained 13 points: the sequence of 3 points (leaf area) from the previous three steps and 10 points for the next 10 steps. The dataset preparation process also included transformation to stationary time series and scaling to (-1:1) range. In this work, the Root Mean Square Error (RMSE) was evaluated for the time horizon from 30 minutes to 5 hours.

The results of leaf area projection prediction presented for three out of six possible options (6 different solutions were in the experiment) are shown in Figure 3-17a, Figure 3-17b and Figure 3-17c. Leaf area in Figure 3-17a, Figure 3-17b, and Figure 3-17c was taken as a sum of the leaves area of all plants in section. The results of this prediction demonstrate an excellent fit to the ground truth. RMSE in Figure 3-18 shows how the error changes with respect to the size of step prediction. It is worth noting that we have accurate predictions even for the 5-hour prediction horizon. RMSE varies from 9 to 14 for different solutions for a 5-hour prediction horizon, and in the test dataset, the leaves area values vary from approximately 100 to 140. It was demonstrated that the obtained performance withstands the application requirements even for the 5-hour horizon (maximum that is required): 5-10% of relative error. For the lower prediction horizon, the relative error is less. Please note that this experiment was conducted in the closed artificial system.

3.3.3 Implementation of the Smart Sensing Platform

System Overview

The LSTM NN architecture described in Section 3.3.2 was implemented, trained, and tested on a desktop computer. However, the real challenge for NNs is their implementation and running on the low-power embedded and mobile systems which are not initially designed for data-intensive computing.

In this experimental study, the ultimate goal is to demonstrate the feasibility of the proposed approach described in previous sections and the opportunity to imple-

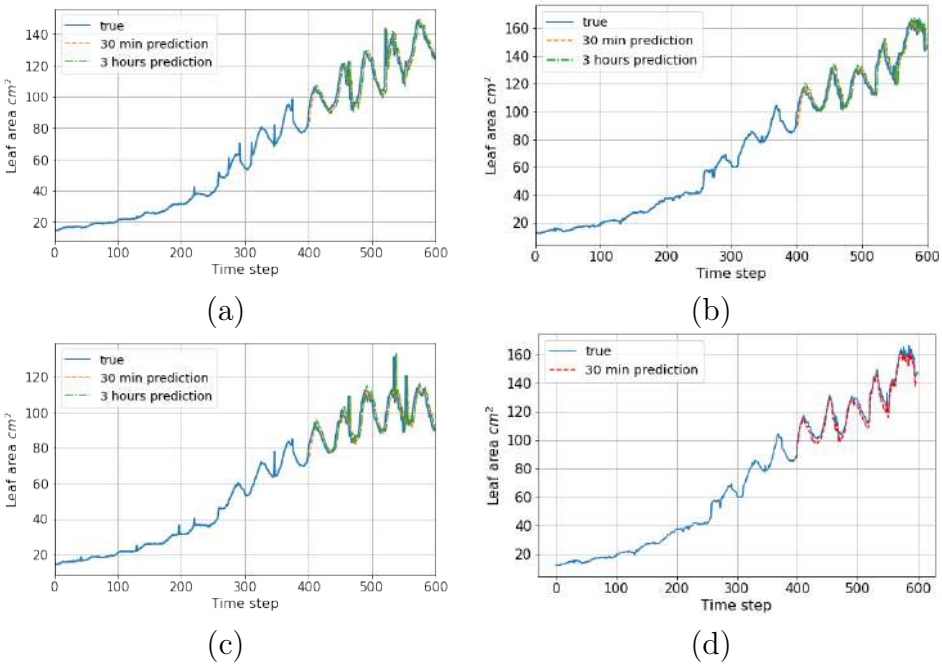


Figure 3-17: (a) Prediction of a leaf area for the section fed with the "Hoagland" nutrient solution. (b) Prediction of a leaf area for the section fed with the "Base + P" nutrient solution. (c) Prediction of a leaf area for the section that fed with "Base + Ca" nutrient solution. (d) Prediction of a leaf area based on autoregression for the section fed with the "Base + P" nutrient solution. Each time step in (a), (b), (d) represents 30 minutes.

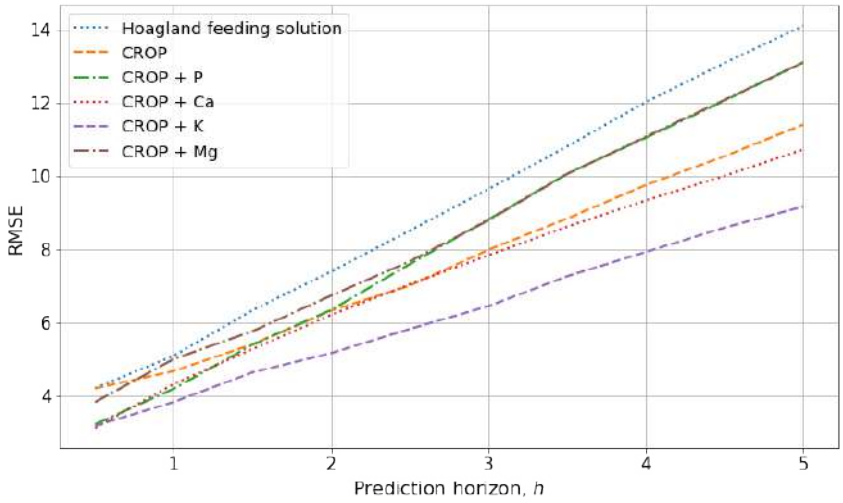


Figure 3-18: Dependence of root means squared errors for prediction of leaf area on the number of prediction steps for all six solutions.

ment the predictive analytics on board of a low-power embedded system. It helps to solve real-world problems in the scope of precision agriculture and, in particular, modeling of the plant growth dynamics.

Since the idea behind precision agriculture is to ensure intelligent, distributed, and autonomous sensing, the following requirements for a tiny sensor node were defined:

- Size [mm]: 100x60x20
- Mass $<$ 100 [g]
- Processor performance: Quad-core 0.9 [GHz]
- RAM $>$ 300 [Mb]
- Power consumption (Idle) $<$ 5 [W]
- OS: Linux
- Interface: USB 2.0 or later
- Wireless connectivity: Wi-Fi
- Price $<$ 100$

The size, mass, and power consumption are essential parameters for the system deployment in a greenhouse or climate chamber. Processor and RAM requirements are necessary for running the AI discussed earlier. The USB interface is required for plugging a camera or connecting external devices, e.g., sensors. The last requirement is wireless connectivity: Wi-Fi is a widely used technology in distributed wireless sensing to connect the sensor nodes to a network. Wireless network evaluation is out of the scope of this work. That is why the operation of a single node is demonstrated. Each node is a consistent part of the future system.

Relying on the benchmarking and evaluation of SBCs in section 3.1, the most appropriate platforms for such an investigation are plain RPi 3B and RPi with NCS. However, only RPi with NCS was chosen due to reliability reasons, as described in Section 3.1.5. Furthermore, it meets most requirements and can be further optimized in terms of power consumption and performance.

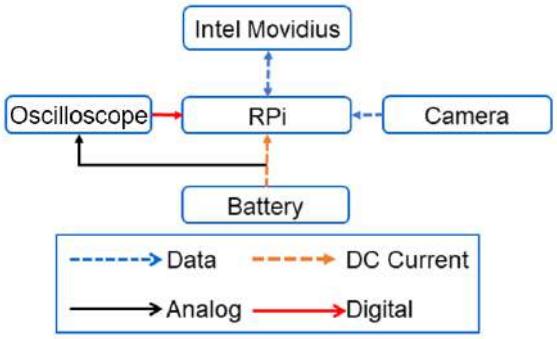


Figure 3-19: Block diagram of the experimental testbed

System Assessment

Experimental testbed is shown in Figure 3-19 a. All the calculations run on Raspberry Pi: it receives and processes the data, then it sends the data to the external VPU and receives back the prediction. This process goes simultaneously with the collection of the power consumption data. The oscilloscope was connected to the power rode of the Raspberry Pi via the 100 mΩ 1% shunt, which receives the Volt-Ampere (VA) characteristics and sends the digital data to the COM-port of RPi. The computational and power consumption related characteristics are collected and recorded into '.csv' table. These characteristics include timestamp, test and train scores, VA, power consumption, CPU, and RAM usage.

The experiment includes the following steps:

- Analysis of the overall computational performance of the system.
- Time estimation of the code execution for a single prediction step.
- Measurement of power consumption.

3.3.4 Experimental Results

The experiment is performed for the proposed low-power embedded system and a laptop. Both platforms are characterized by the parameters summarised in Table 3.1.

Pre-trained LSTM was uploaded to both platforms with the test dataset. After that, the NN iteratively performed the predictions. The total number of iterations is 1000. Every single run, the model and the test dataset were refreshed. During the investigation, the RMSE and the time spent for a single prediction were measured (see Figure 3-20). The 'prediction time' is the difference between the final and the start time of a single run of the prediction algorithm. Figure 3-20 demonstrates that the time per prediction varies over the experiment for both platforms, primarily for the RPi with NCS. It is featured lower computational performance and, hence, the increased possibility of buffer overflow.

The computational performance of the computer as compared to the Raspberry Pi is roughly six times higher, as it is shown in Table 3.8.

Table 3.8: Performance evaluation of computer and embedded intelligence prototype

Parameter	**Computer**	**Prototype**
Median prediction time, s	0.55	2.98
Maximum prediction time, s	1.26	7.82
Minimum prediction time, s	0.49	1.81

RMSE for the experimental testbed remains constant for every iteration for Raspberry Pi and computer and equals to 8.30. Each prediction is made by the NN, trained on a dataset with 400 elements (1 element is a one-time step, which equals 30 minutes), and every prediction is made for the following 200 steps. The operation period of the system is 30 minutes. During that period, the system starts, gets pictures, processes it, and feeds it to the NN. It takes around 30 s. The rest of the time, it is in sleep mode; hence the duty cycle is 3.4 %. Mean time for booting up is around 20 seconds, 7 seconds takes to get and process picture, prediction time varies from 1.8 to 7.8 s with a median value of 3 s.

The computational performance is low, mostly due to the absence of a USB 3.0 interface on board of Raspberry Pi 3B. In contrast, Movidius is equipped with it. Even though the single board computer with Movidius has a higher prediction time, it is still much faster than needed. For the current problem, each prediction should be performed every 30 minutes or less frequently.

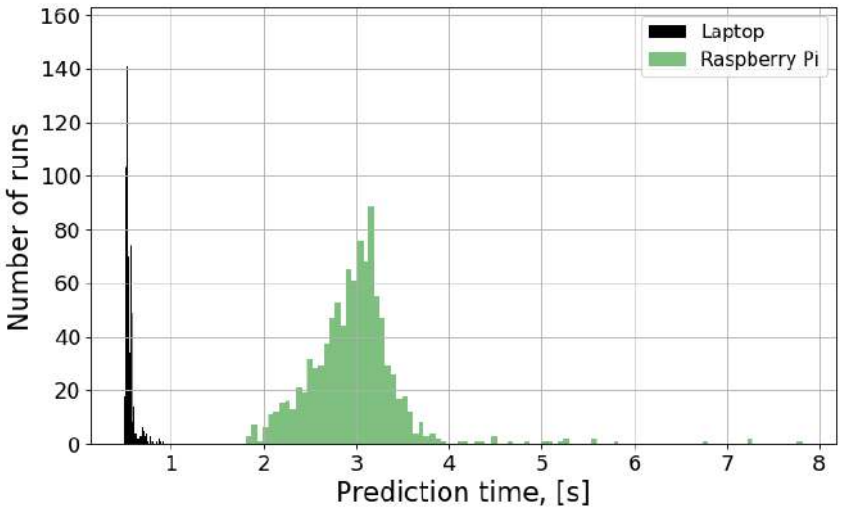


Figure 3-20: Histogram of run time distribution. Raspberry Pi - green; computer - gray.

For assessing the long-term operation of the proposed embedded AI system, we used a 2550 mAh power bank and run the sequence 'load model - load test set - make prediction' in a loop. In this experiment, we measured the power consumption over the iterations (see Figure 3-21).

The prototype of the proposed intellectual monitoring system requires an external source of light. It is a scalable solution that is easy to deploy as a distributed system in the greenhouses without additional hardware and infrastructure. According to the investigation, the prototype can perform 8663 continuous predictions within 4 hours 34 minutes before the battery discharge. The mean prediction time is 1.9 s. During the investigation, the mean power consumption is 2.23 W; the median is 2.184 W, and the modal value of power consumption is 1.7578 W. During the experiment, the CPU load was 52.44 %, with a standard deviation of 3.22. Simultaneously, the RAM had a 50.8 % load with a standard deviation of 0.19. Due to the continuous and slow change of the plants' state, there is no need to perform these predictions continuously: the 3 hours time step is sufficient and still provides high precision of predictions. Since the system is assumed to switch on and off every 30 minutes and perform one prediction within this period, the system can operate autonomously for up to 6 hours until the battery is completely depleted. Here we accomplished the calculation, similar to section 3.2.4. The following formula could

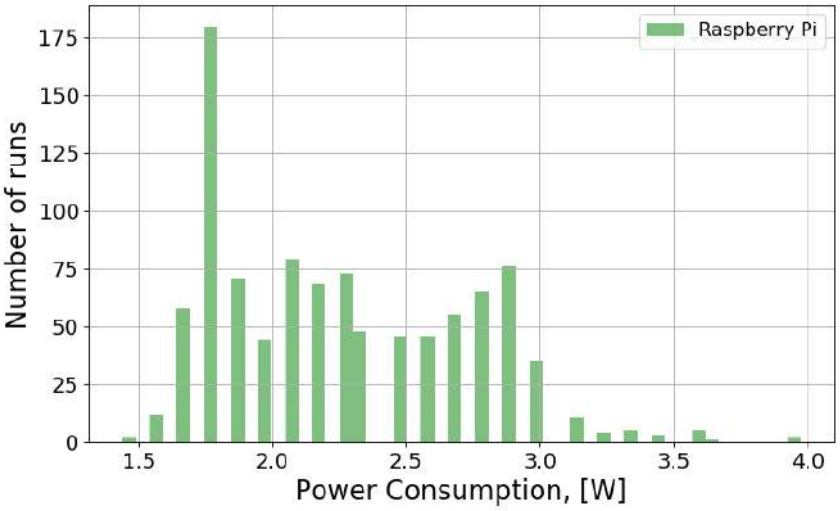


Figure 3-21: Power consumption of the prototype over runs.

calculate the required capacity of the battery: $T = \frac{E*U}{P}$, where T is the time of operation, P is the power consumption, E is the battery capacity in [A*h] and U is the input voltage. Assuming that the monitoring platform is plugged into 5 Volts power source and with 2660 mAh capacity and works continuously consuming 2.23 Watts, the battery's will be completely depleted in 6 hours. The obtained performance withstands the application requirements discussed in Section 3.3.3. It is in case the platform operates in the idle mode - prediction - idle mode routine. However, if it operates the switch-on - make a prediction - switch-off routine with 30 min timestep, the overall period could be prolonged for up to 180 days.

That is why the proposed solution proves to be an efficient low-power system with AI on board. It can be beneficial for the applications where low power consumption, low mass, and size and autonomous operation are required. The system can be easily extended and adapted to mobile platforms, e.g., drones. Once realized into practice, this system will close the gap and will be able to perform the full cycle of plant growth dynamics analysis.

The power consumption of the platform varies within the time of the experiment. This value correlates to the period spent on the prediction task. There could be many unpredicted factors in a real deployment influencing the performance, e.g., reducing the system performance and increasing the power consumption (see Figure

3-20 and Figure 3-21). The primary factor influencing the system performance is limited RAM and cache memory of Raspberry Pi. It leads to memory overflow, which has a negative impact on power consumption.

3.3.5 Conclusion

This work presents a generic low-power embedded platform equipped with the AI on board to assess and predict the plant growth dynamics.

Performance evaluation of the proposed solution has demonstrated that the developed AI architecture based on a Recurrent Neural Network (RNN) called Long-Short Term Memory (LSTM) is characterized by reasonable precision for the prediction horizon. The proposed solution can be used as an autonomous tool for continuous plant growth dynamics monitoring. Together with an actuating capability, the proposed approach is promising for guarantying easy-to-deploy, generic, and robust optimization tools for precision agriculture.

The *Tomato Growth* dataset Shadrin et al. [2019] for training and testing procedures were collected by the designed and assembled experimental setup coupled with the automatic imaging system. This dataset is publicly available for the research community. It can be used in various computer vision tasks to develop and verify new machine learning algorithms. For effectuating the growing stage on the experimental setup, we used a hydroponic approach. It ensures the optimal control of the plant nutrition and provides the opportunity to "drive" the system in a desirable way.

3.4 Detection of Hogweed onboard of UAV

3.4.1 Introduction

This section reports on an aerial drone platform with an embedded system on board capable of real-time image processing using FCNN. The developed prototype can process the data on board and provide the detection results in a few seconds. The output is the mask image or text file with the coordinates and parameters of hog-

weed. The collected dataset is shared hog [2019] with the research community for testing reasons and designing the segmentation algorithms.

This section focuses on the development of the UAV platform for effective monitoring and eliminating harmful plants. As an example, the Hogweed of Sosnowskyi (*lat. Heracleum*) was determined. The approach includes the UAV with an embedded system on board running various FCNNs. The optimal FCNN was proposed. Its architecture optimized for the embedded system relying on the trade-off between the detection accuracy and maximum frame rate. The best performance of hogweed recognition with 47 % Intersection over Union (IoU) in real-time was achieved. The proposed model achieves ROC AUC 0.96 in the hogweed segmentation task, which can process 4K frames at 0.46 FPS on NVIDIA Jetson Nano. The developed system can recognize the hogweed on the scale of individual plants and leaves. This system opens up a wide vista for obtaining comprehensive and relevant data about the spreading of harmful plants allowing for the elimination of their expansion.

3.4.2 Methodology and Platform Overview

The core components of the research include the following three points: Deep Learning (DL) algorithms, an embedded system able to run the FCNN, and provide the inference, and the UAV platform. The research is based on the sequence of the following steps:

- Data collection. Several locations were investigated in the Moscow Region, Russia. The aerial imagery dataset was collected using several flying platforms, with different optical sensors and operating altitudes, to achieve better performance of the FCNN.

- Labeling of the dataset. The manual labeling of the dataset for two classes (hogweed and not-hogweed) was implemented.

- Training of the FCNN. Several FCNNs were trained to solve the semantic segmentation tasks. Their performance was also evaluated.

- Inference of FCNNs on the desktop. The performance of the proposed FCNNs

was checked on the desktop computer. The investigation subject is as follows: the images from the test dataset and previously captured test video. Evaluating the performance using the following characteristics: Frames Per Second (FPS), ROC AUC, the potentially covered region (in m^2) in a mission.

- Inference of the NNs on a Single Board Computer (SBC) on the ground (with previously captured video). In addition to the characteristics from the previous step, the power consumption was also measured.

- Performing the inference of the NNs on the SBC in the flight mode. The SBC was used as a payload of the drone. The performance of NNs was estimated as in the previous step.

Platform Requirements

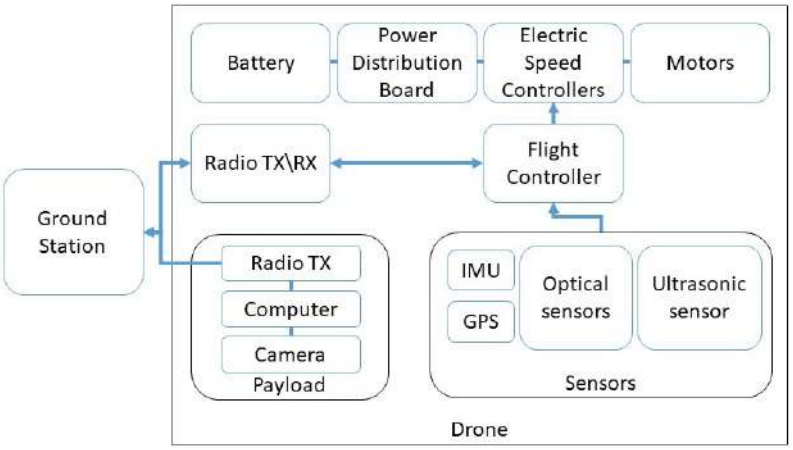


Figure 3-22: The diagram of the proposed platform

- Minimum Area Covered: 15 ha,

- Minimum ROC AUC: 0.90,

- Maximum Power Consumption: 7.5 W.

The evaluation of all investigated CNNs for the semantic segmentation task was accomplished on the Nvidia TX 1080Ti and single board computer Nvidia Jetson Nano. For the CNN's' assessment three parameters were chosen: $ROC\ AUC$, which

can be regarded as a quality metric, *FPS* as a measure of the speed of computations, *power consumption* and *area coverage*. However, it is better to understand which area the user can cover with a flying drone to estimate the proposed solution's performance in practice. It is the reason why we calculated the *Area Covered* parameter.

Nvidia Jetson Nano was used as a payload for the drone DJI Matrice 200. The images are collected using 12 MP camera. The altitude of flight is 10 meters, the range of the mission is limited by 40 minutes, and there is no overlap between the neighbor photos. The drone makes a photo then flies to the next position for taking the next one. We calculate the total area as follows:

$$S = GSD^2 * x * y * FPS * t \tag{3.3}$$

where

$$GSD = \frac{H}{f} = \frac{L}{l} \tag{3.4}$$

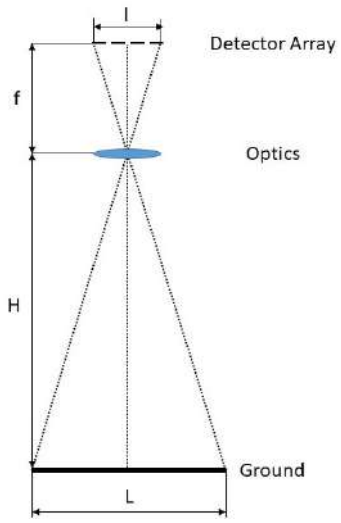


Figure 3-23: Ground Sample Distance.

Ground Sample Distance (GSD) (see Fig. 3-23) is the distance on the ground (L) covered by a single pixel (l) of a camera. This value also equals to the ratio of the altitude (H) and the focal distance of the camera (f). It is measured in cm/pixel and varies for different camera's sensors and altitudes, e.g. for 12 MP camera at 10

m altitude GSD stands for 0.43 cm/px; x and y is the quantity of pixels along the x and y axes of the photo respectively; FPS are the frames per second; t equals to the 40 minutes time period. Higher frame rate (due to the NN performance) allows for flying with higher speed on the same altitude hence allows for coverage of larger area during a single flight.

Dataset Collection and Labelling

Although various species of hogweed exist across Eurasia, Sosnowsky's hogweed represents one of the most widely spread weed both in Europe and Asia Cock and Seier. It is one year plant and could reach a height of up to 5 meters. The leaves could be as long as 0.5 meters and appear at the beginning of May each year. Inflorescence represents an umbel, located at the end of each stem. It reaches up to 0.4 meters in size — the Sosnowsky's hogweed blooms in the middle of each summer. Hogweed usually displaces all other plants.

The extensive distribution area and big leaves of hogweed make an aerial data collection relatively easy. The green color of leaves (visible spectrum) could vary across the season and location, which also has to be considered during the dataset collection. The dataset was collected during the middle of May 2019. At that time, leaves have already reached around 0.4 meters in length, and the plants were approximately 1 meter in height. There was no inflorescence. Hogweed covers significant areas across Russia, including the Moscow region. The current dataset was collected near Yurlovo village in Krasnogorsk city district in Moscow Region (117 meters above the sea level, Latitude: 55 deg 53' 51.78" N, Longitude: 37 deg 16' 16.25" E).

For reaching better generalization, the data was collected from different locations. Some of the received images were near the trees, other ones were in the lowland with some water content, and we also covered the highland places with more dry soil. All of these conditions lead to different lighting and different plant nutrition. Therefore, the collected data has a noticeable variation in the plant growth environment. Also, various backgrounds on the images with and without hogweed allowed for training more robust NNs.

(a) DJI Phantom 3 drone (b) DJI Mavic Pro drone (c) Xiaomi action camera

Figure 3-24: Captured images using different platforms.

For cost-effective and reasonable simplicity reasons, the research was conducted in the visible spectrum and used regular RGB cameras. At the same time, we carried out a comparative study on the UAVs and collected the images using different platforms. It also helped to collect a dataset from heterogeneous sources. Two of them were quadcopters, and one was a handheld action camera. We used DJI Phantom 3 drone (sensor 1/2.3" CMOS, image size 4000x3000 pixels, 92 images collected), DJI Mavic Pro drone (sensor 1/2.3" CMOS, image size 4000x2250 pixels, 102 images collected), and Xiaomi Yi action camera (sensor 16MP CMOS, image size 4608x3456 pixels, 69 images collected).

These platforms allowed for taking the images from a wide variety of positions and orientations. Multiple flights were accomplished to obtain the dataset. Some of these missions were performed in autonomous mode with DroneDeploy application for the DJI drones. Another portion of the images was collected during the manual flight and manual image capture. All platforms used the default camera settings. Operation altitude for the drones was maintained at 8-12 meters above the ground. The action camera was placed at around 1 meter above the hogweed leaves. Examples of the images are shown in Fig. 3-24.

The collected dataset was labeled manually in the *Supervisely*. The dataset contains images, semantic masks, and *json* for two classes and complete orthophotos. It is publicly available and could be used for deep learning investigations in precision agriculture. The number of annotated images is summarized in Table 3.9.

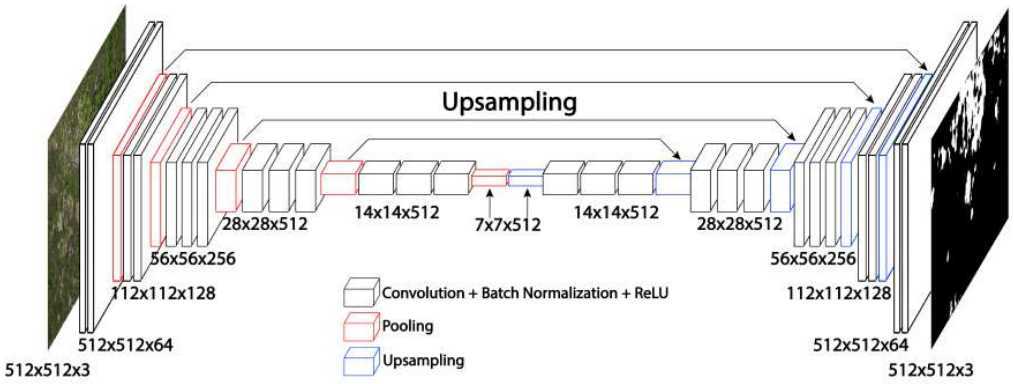


Figure 3-25: The architecture of the modified SegNet, used in the investigation.

Table 3.9: Summary of obtained images

Device	DJI Phantom 3	DJI Mavic Pro	Xiaomi action camera	**Total**
Without objets	3	15	0	**18**
Separate objects	31	30	24	**85**
Many objects	58	57	45	**160**
				263

FCNNs for Semantic Segmentation of Hogweed: a Comparative Study

The general idea behind the FCNN architecture is applying a convolution network followed by the deconvolution layers. A 1x1 convolution layer usually connects these two parts. The convolution part is a classification network such as VGG Simonyan and Zisserman [2014] or ResNet He et al. [2016b]. The deconvolution task is to project the features created by the convolution part onto the high-resolution image.

In this work, various FCNN were used: SegNet Badrinarayanan et al. [2017], variety of U-Net Ronneberger et al. [2015], and RefineNet with the ResNet backbone to compare the performance on the hogweed segmentation. Principal schemes of the used FCNN Noh et al. [2015] for semantic segmentation task are shown in Fig. 3-25. The detailed architecture of the used FCNNs is available in Appendix A.1. For solving hogweed segmentation, different FCNN architectures were tested. The training parameters (number of epochs, batch size, layers' parameters) were varied to obtain the best possible performance. Moreover, different data preparation procedures were used, including cropping, flipping, resizing, rotating, color jittering,

and many others. Next, we describe the training and data preparation procedure that guaranteed us the best performance. The proposed FCNNs were trained on the cluster of four Nvidia 1080Ti. Then their performance was evaluated on both the cluster and *Nvidia Jetson Nano*.

SegNet

SegNet Badrinarayanan et al. [2017] is a neural network architecture for multi-class pixel-wise segmentation. It consists of a sequence of encoding blocks and corresponding decoding blocks followed by the classifier (see Fig. 3-25). Each encoder block includes a convolutional layer, Batch Normalization (BN) and ReLU non-linearity. The first 13 convolutional layers in the encoder were taken from the first 13 convolutional layers in the VGG-16 network.

The SegNet was trained on the hogweed dataset described in the previous subsection. We trained the network on 512x512 frames with a 50% chance of horizontal and vertical flips and random rotation up to 30°. *LogSoftmax* layer is used to predict the hogweed class after the last convolution. The negative log-likelihood loss is applied to calculate the derivatives and make step by Adam optimizer with default parameters ($LR = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$). It took around 9 hours to train this network over 1500 epochs for obtaining acceptable accuracy. This long period is required due to the high number of network trainable parameters: 29,444,166. Fig. 3-26 depicts the example of SegNet prediction.

U-Net

U-Net Ronneberger et al. [2015] is a convolutional network architecture for semantic segmentation. U-Net, as well as SegNet, is an encoder-decoder architecture. In this study, some modifications were applied to the original U-Net architecture. First of all, the BN layer was inserted after each convolutional layer. Thus, in this study, the U-Net with the following architecture was used: 9 double convolution blocks composed of two convolutions, BNs and ReLU activations, and the output convolution layer. Some convolutional channels of the UNet were made adjustable for further application of the width scaling technique Tan and Le [2019]. It is set by

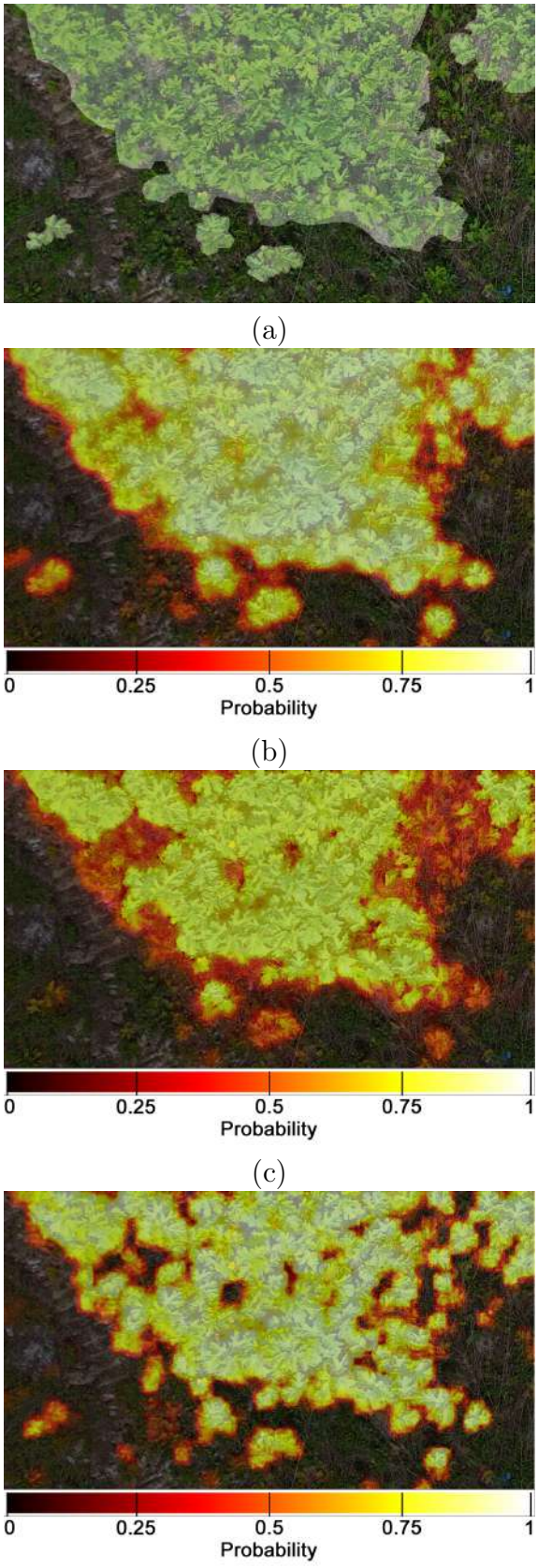
the parameter W, which denotes the number of output channels in the first convolutional layer. Also, to make the depth scaling possible, one encoder block and one decoder block are made removable. It is indicated by the parameter D, which denotes how many blocks are in the encoder. During the training process, we use the augmentation techniques such as the random crop with the size of 512×512, random horizontal flip, and random rotation on the angle from $[-\pi/2; \pi/2]$. We use a weighted binary cross-entropy as the loss function. Classes of background and hogweed are taken into account with weights 1 and 2 to compensate for the imbalance. As an optimizer, we use Adam with the default learning parameters. Fig. 3-26 shows the example of U-Net output.

RefineNet with ResNet backbone

The current research feature is large scale images (commonly 2250x4000 pixels) that must be processed rapidly and accurately. For achieving this goal, one should choose from several unique architectures that can handle these problems. RefineNet Lin et al. [2016] is a popular general framework for building the high-resolution segmentation networks at the top level of some other networks (backbone), e.g. ResNet's family. The main idea of RefineNet is to fuse various levels of details on the different layers of convolutions. On the one hand, this approach allows us to keep the fine structures of the original images. On the other hand, it does not use the large intermediate features' maps, which do not use much memory.

ResNet is a popular architecture proposed by the Microsoft Research team with the residual connections between functional blocks He et al. [2016b]. Residual connections allow for constructing very deep networks that can achieve good quality. The implementation of RefineNet with a pre-trained ResNet-50 network was used in the research as backbone Nekrasov et al. [2018].

The RefineNet output is 1/4 of the original input size by design: the interpolation was performed to restore the original mask size. For data augmentation, the following techniques were used: the random rotation, random frame cut, horizontal and vertical flips with 0.5 probability. The performance of RefineNet on the test image is shown in Fig. 3-26.



(a)

(b)

(c)

(d)

Figure 3-26: (a) The input frame with annotation. Target class probability predictions made by neural network models: (b) Frame and Annotation; (b) U-Net (W=32; D=5); (c) SegNet; (d) RefineNet

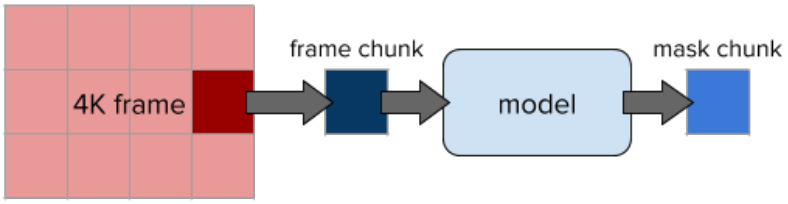


Figure 3-27: Processing pipeline

Implementation of the FCNNs on Embedded System

As mentioned earlier, the ultimate goal is the end-to-end UAV-based platform capable of autonomous real-time detection of dangerous weeds. It requires a lightweight network capable of running at a high frame rate and the computationally powerful low-voltage on board processing unit for running the network.

The results of section 3.1 demonstrate that Nvidia Jetson Nano is the best platform for implementing the intellectual monitoring platform for aerial imagery processing on board the UAV. Apart from many advantages for semantic segmentation task, Jetson also has the standard CUDA interface. Unlike the various revisions of Intel Movidius, it does not require any neural network conversion and allows one to launch a model written on any CUDA-aware framework. Besides, it is a standalone device where CPU-to-GPU communication is implemented most efficiently.

However, this platform has a limited memory - all 4 Gb of RAM is shared between the CPU and GPU. Each 4K input frame and temporal memory for convolution result of the same size takes about 100 Mb. There are two possible approaches to fit 4K images to the FCNN's input: (i) rescaling and (ii) cutting the frame into tiles. The second approach is better for two reasons. Firstly, training of the neural network (especially FCNN type) usually requires much more training parameters, therefore, a bigger computational capacity of the computer, which is a limiting factor. Furthermore, it leads to a bigger size of the resulting pre-trained FCNN, which can easily reach several Gb. However, in our case, it is not a problem, since UNet occupies 100 Mb RAM. Secondly, rescaling leads to a reduction in the input sample's number of features, which easily leads to underfitting. In the case of cutting the image into tiles, we preserve the features from the original photo. Therefore, we have better output results - higher Intersection over Union (IoU). It is essential in the case

of semantic segmentation of hogweed, which usually has a green background. This approach also reduces the number of trainable parameters, reducing the required RAM and the time of inference. Therefore, the image is split into smaller chunks for realizing one-by-one processing. The processing pipeline is shown in Fig. 3-27, where the red and blue areas depict the CPU and GPU memory, respectively.

Dividing the frame into parts and using a sliding window does not speed up the network. It helps in reducing the used memory. To achieve truly high performance requires investigation and changing the networks. For example, the original U-Net with 4K resolution input frames runs at 0.025 FPS on NVIDIA Jetson Nano. This result is inadmissible in the real-time application. However, it contains many parameters and may be exposed to the model scaling without a significant loss of accuracy. Thus, the model scaling was applied:

- Width scaling: reduce the number of output channels in the convolutional layers. Parameter W indicates the degree of scaling. Original U-Net is characterized by $W = 64$.
- Depth scaling: reduce the number of convolutional blocks. Original U-Net has $D = 5$ and U-Net scaled by depth has $D = 4$.
- Compound scaling: apply width and depth scaling simultaneously.

These techniques were applied to UNet training to identify the best combination of D and W parameters.

Metrics

For the recognition quality evaluation, the Area Under the Curve (AUC) of a ROC curve was used. The following formulas define the true positive rate and false positive rate:

$$TPR = \frac{TP}{TP + FN} \tag{3.5}$$

$$FPR = \frac{FP}{FP + TN} \tag{3.6}$$

Table 3.10: Model quality and size among with Nvidia Jetson Nano inference results

Model	Parameters ($\times 10^6$)	FPS	ROC AUC	Area covered, ha	Power, W
U-Net (W=32,D=5)	3.4	0.074	0.967	3.0	7.0
U-Net (W=16,D=5)	0.84	0.15	0.965	6.0	7.0
U-Net (W=8,D=5)	0.21	0.27	0.963	10.8	6.5
U-Net (W=4,D=5)	0.053	0.46	0.958	18.4	5.5
U-Net (W=3,D=5)	0.03	0.53	0.946	23.5	5.5
U-Net (W=2,D=5)	0.013	0.68	0.938	27.6	5.5
U-Net (W=32,D=4)	0.84	0.085	0.956	3.4	6.5
U-Net (W=16,D=4)	0.21	0.16	0.954	6.4	6.5
U-Net (W=8,D=4)	0.053	0.28	0.952	11.2	6.0
U-Net (W=4,D=4)	0.014	0.47	0.948	18.8	5.5
U-Net (W=3,D=4)	0.0077	0.55	0.939	24.0	5.5
U-Net (W=2,D=4)	0.0035	0.70	0.938	28.3	5.0
SegNet	29.4	0.020	0.969	0.8	7.0
RefineNet	27.3	0.20	0.968	8.2	7.5

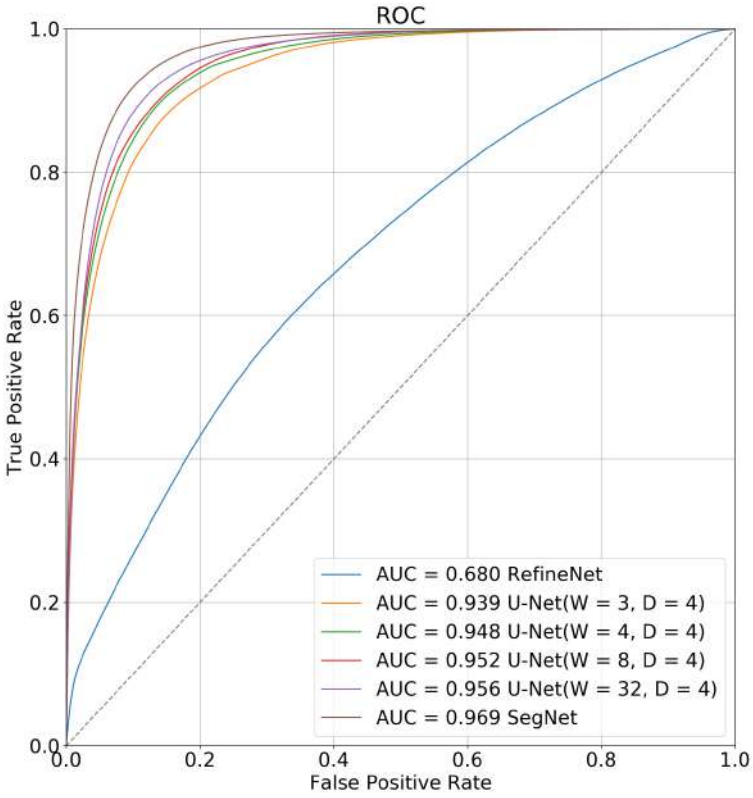
where TP, TN, FP, FN are the numbers of true positive, true negative, false positive, and false negative classifications for target class, respectively. For the inference performance assessment, the frame rate, or FPS, was used. The input frame has a 4000×2250 resolution. Apart from the recognition quality and inference performance, the embedded system power consumption during the inference was assessed.

3.4.3 Experimental Results

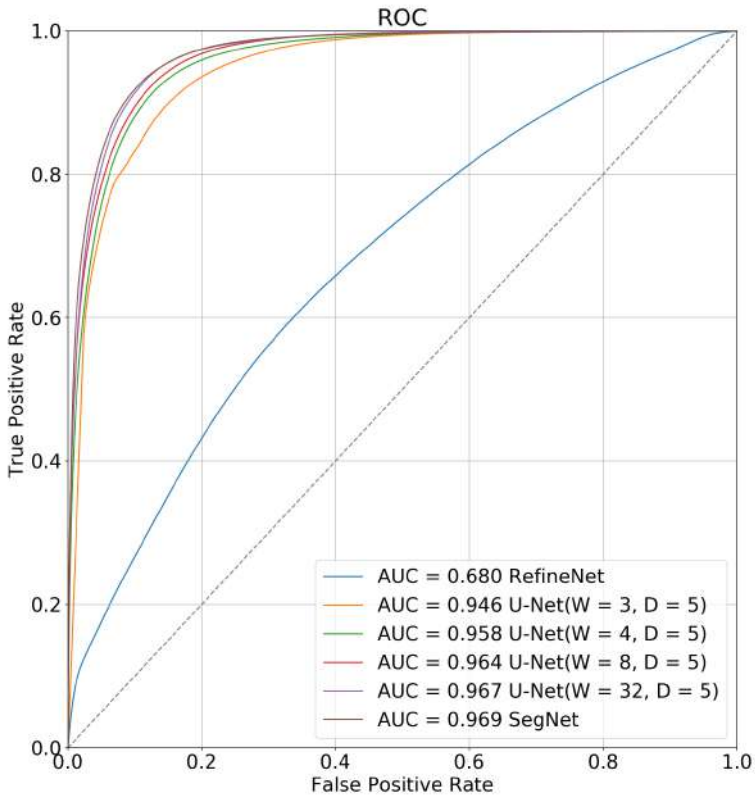
Comparison of the models in terms of recognition quality, inference performance, number of parameters, and current consumption is shown in Table 3.10. It demonstrates that the more parameters the model has, the more time inference takes.

As mentioned earlier, the recognition quality is expressed using the area under the ROC-curves. This metric helps in demonstrating a detailed view of some of the trained models (see Figure 3-28). It shows that the best models in ROC AUC also have the highest true-positive rate and the lowest false-negative rate compared with other models with the same threshold value.

Figure 3-29 shows how inference performance and recognition quality are related to some trained models. In particular, it shows that complex models, e.g., SegNet, U-



(a)



(b)

Figure 3-28: (a) ROC AUC for UNet versions with D = 4; (b) ROC AUC for UNet versions with D = 5

Net($W = 32$, $D = 5$), and RefineNet, provide high recognition quality, but operate too slowly. We assumed that by reducing the size of the model, it is possible to significantly increase its operation speed while not reducing the quality significantly. This hypothesis is generally true as it is made evident by U-Net($D = 5$) width scaling results. In comparison with U-Net($W = 32$, $D = 5$), the 4 times scaled U-Net($W = 8$, $D = 5$) is 3.6 times faster and only 0.5% worse in quality. The 8 times scaled U-Net($W = 4$, $D = 5$) is 6.1 times faster, and only 1.0% worse in terms of quality. Since further width scaling leads to a significant loss of quality, U-Net($W = 4$, $D = 5$) is an acceptable trade-off between the quality and the frame rate. This investigation also shows that the depth scaling (models with $D = 4$) has almost no effect on the inference speed, but leads to significant quality drop. It can be explained by the fact that as the network depth decreases, the receptive field for pixels in the U-Net bottleneck feature maps decreases.

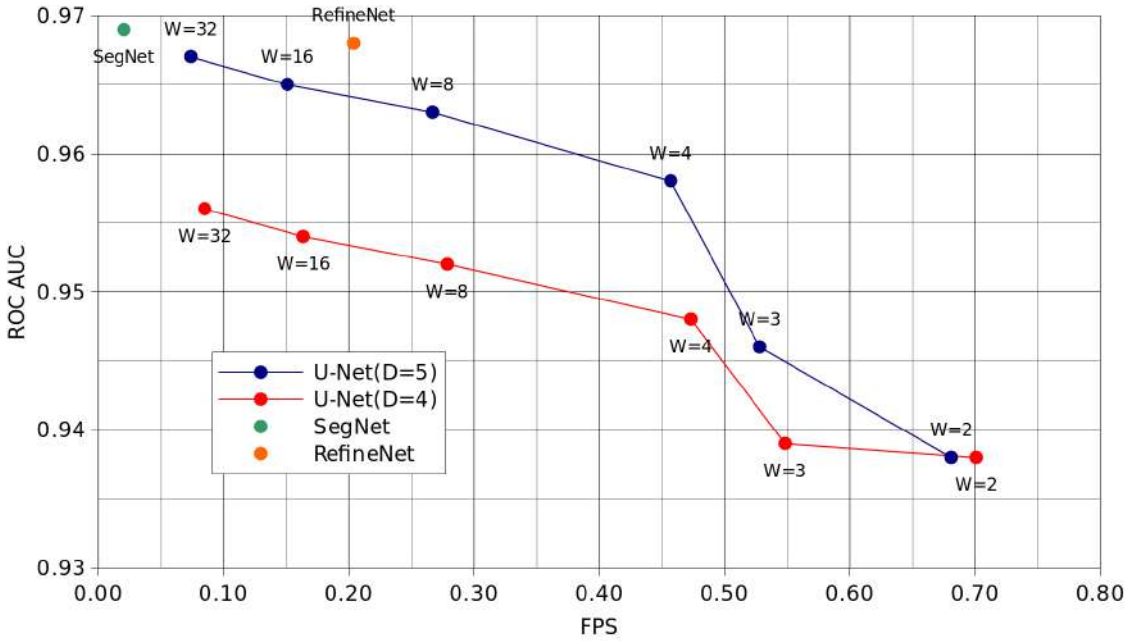


Figure 3-29: Performance and quality comparison on NVIDIA Jetson Nano

Besides that, Figure 3-30 shows the power efficiency comparison of the same models. Despite different current values, the inference speed makes a significant contribution to power efficiency.

Carried out analysis of FCNNs output showed that in some cases, the NNs predict correct masks even if they were not labeled by hand (see Figure 3-26). At this point,

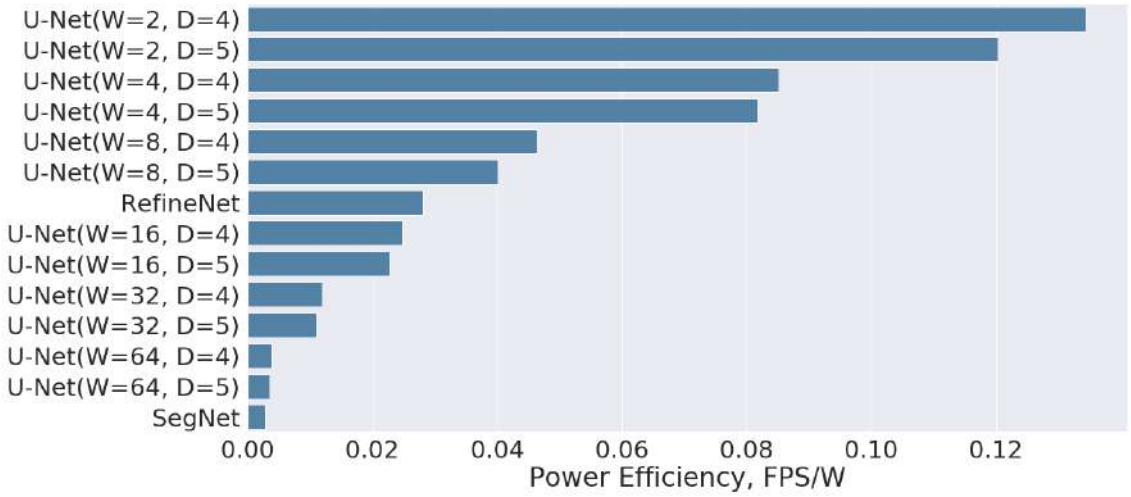


Figure 3-30: Power efficiency comparison

we demonstrate that different architectures return different results in several cases, as it is shown in Figure 3-32. Our models work correctly for a variety of scenarios:

- Single plant. In this case, the NN takes the input image with the single object of interest or with a few objects spread over the photo.
- Brushwood. In this situation, the drone identifies multiple plants covering more than 50% of the photo.
- Hogweed thickets mutual spatial arrangements.

Using equation (3.3) and frame rate results, we calculate the potential covered area for a typical industrial drone, e.g., DJI Matrice 200 equipped with the 12MP 4000×2250 camera, for a 40-minutes flight. The results are shown in Figure 3-31.

Figure 3-28 and Figure 3-26 show that the SegNet, U-Net($W = 32$, $D = 5$) and RefineNet models have the best results in recognition quality. On the one hand, these NNs return very detailed results (highlighting even tiny features on the mask). On the other hand, its performance is not sufficient for the large area coverage (see Figure 3-31). However, in terms of power consumption (see Figure 3-30), the result is not among the best solutions. It is worth noting that the power consumption is an important point in terms of NN trade-off. We found out that NVIDIA Jetson Nano is sensitive to the input voltage. Also, the voltage drop due to the low battery charge

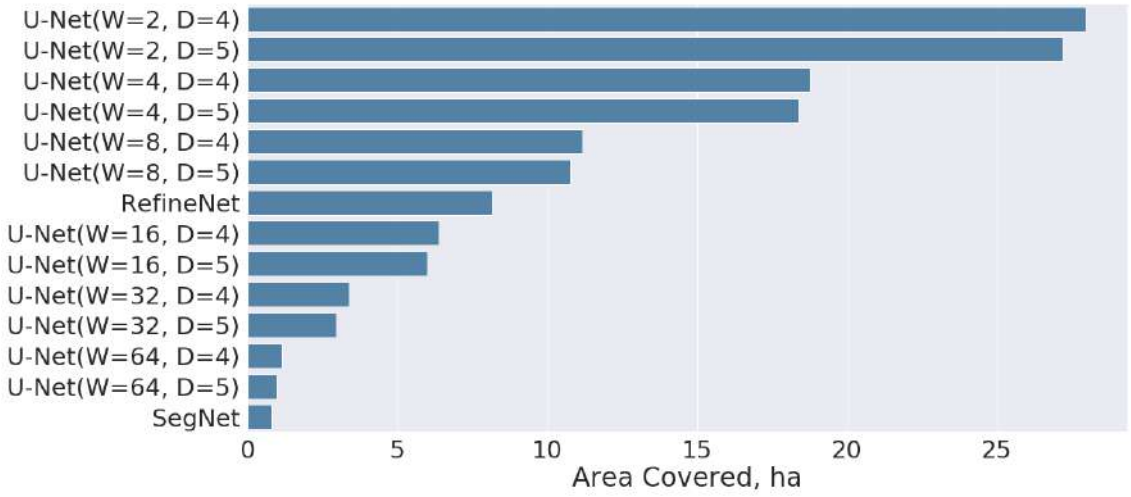


Figure 3-31: The potential area covered while using different neural networks

at the end of the mission could switch off the computer. That is why U-Net($W = 4$, $D = 5$) is the best option for the mission in real conditions.

All the proposed AI algorithms and optimization techniques were tested in the simulation. The aerial data, collected during the flight, was stitched into orthophoto mosaics. Using Surface From Motion (SFM) algorithm, the series of overlapped aerial images were converted into a textured 3D model, which repeats the original landscape. Both the orthophoto mosaic and 3D models were created in the WebODM software. The 3D model was further used in the AirSim simulator Shah et al. [2017]. It is a platform for AI-related research to experiment with deep learning algorithms for autonomous vehicles, e.g., drones and cars. AirSim allows the user to load 3D models of orthophoto maps (see Figure 3-33) of areas in the Moscow region, Russia, where hogweed is growing and perform a simulation of the drone flight. The simulation engaged two computers - the desktop, which ran the simulation environment, and the SBC, which performed the simulator's flight and processed all the input data in real-time. It allowed us to make all the relevant measurements depicted in the Results section: FPS, ROC AUC, Area coverage, and Power Consumption.

The research methodology includes the data collection, labeling, training, and optimization of FCNNs for embedded devices. However, several assumptions could influence the scaling and could limit the proposed system. First of all, the data could be obtained in wavebands behind the visual spectrum. All the demonstrated

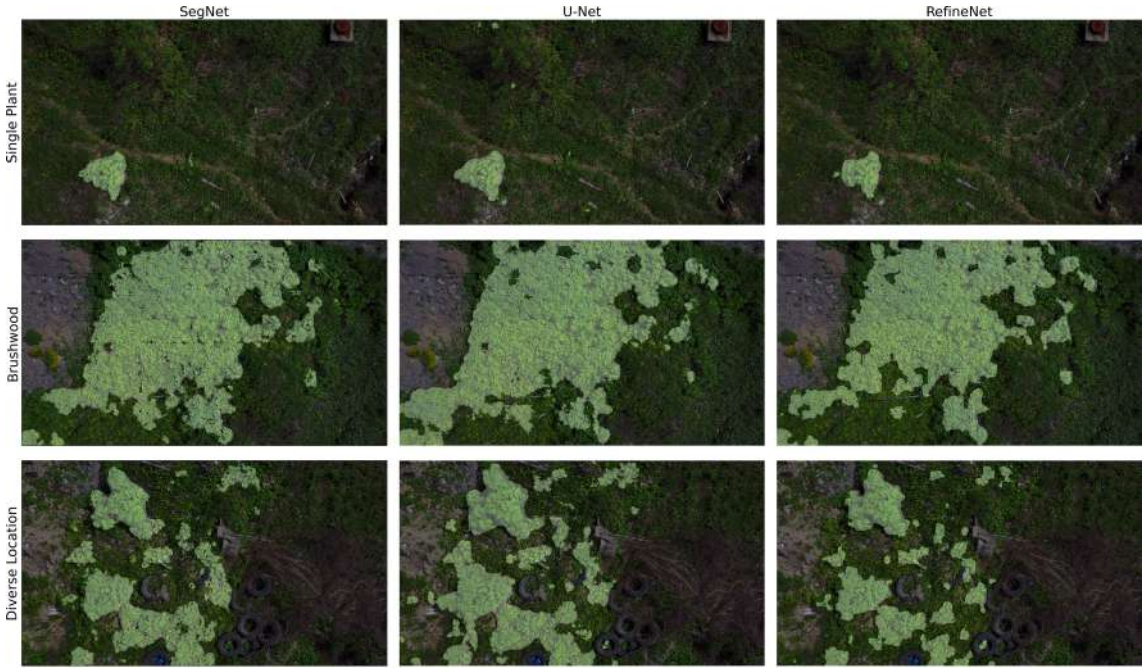


Figure 3-32: Predicted masks for SegNet, U-Net and RefineNet for different situations: the image with the single plant on it, with multiple plants and with variety of other objects

(a) (b)

Figure 3-33: (a) Aerial images from simulator. (b) Simulator window screenshot.

digital cameras could work in the red-edge and near-infrared bands using special optics. These cameras, along with the multispectral cameras, could provide additional features for better plant segmentation. It could lead to the development of faster segmentation algorithms.

Indeed, multispectral imaging opens a wide vista for performing the precise plant and phenotype detection Dutta et al. [2015]Sodhi et al. [2017] Humplík et al. [2015]. The main difference between the plant detection in multi-spectral range and common RGB is the following: some plants have unique reflection characteristics outside of the visual range. Multispectral imagery may involve simple classification algorithms instead of FCNNs for plant phenotyping. These algorithms classify the pixels and find those with the intensity representing the required waveband and belonging to the certain plant Pignatti et al. [2019]. However, one of the disadvantages is the cost of high-quality commercial multi-spectral cameras compared to the RGB cameras' cost.

An RGB camera for hogweed detection was used as a cost-effective solution that meets the work requirements. We can detect the hogweed on the vegetation stage, one of the most complex detection problems. As the hogweed is green and blends in with the background, the only specific hogweed shape can be detected in the vegetation stage. In the later stages, e.g., flowering, it is much easier for the AI algorithms to detect the hogweed since its white flowers can be easily recognized in the RGB range even without the application of complex FCNN.

Although it is out of this research's scope, there is still an open issue about the instance segmentation for two and more classes. The proposed platform was tested in real conditions and a 3D simulator in real-time. The orthophotos were created out of the imagery captured in the first phase of the investigation. ROC-curves showed in Figure 3-28 and images shown in Figure 3-26 demonstrate that all networks perform the hogweed segmentation with reasonably high quality.

The experimental evaluation confirms the existence of the U-Net modifications, which preserve the acceptable segmentation quality rate and outperform the original U-Net, SegNet, and RefineNet in terms of power efficiency and inference speed. These modifications are obtained with the model scaling method. Also, we assume

the U-Net ($W = 4$, $D = 5$) as a NN of choice for our further research. It outperforms other networks by frame rate, except for the modifications with 3 or fewer channels in the first convolution layer, which have significantly lower recognition quality. Compared to RefineNet (best baseline model), it is only 1% worse in terms of the target quality metric, but it works 2.3 times faster and 2.9 times more power-efficient.

The obtained results could be compared with relevant research work Sa et al. [2017], where the tasks are similar. The authors reported ROC AUC 0.945 for 'crop' class and 0.787 for 'weed' class, ROC AUC 0.968 for hogweed is a comparable result in terms of recognition quality. That system requires about 550 ms per 480×360 frames on NVIDIA Jetson TX2 (twice as many GPU cores than Jetson Nano) when images are loaded from a disk. In the same inference scenario, U-Net ($W = 4$, $D = 5$) result is 65 ms per frame on NVIDIA Jetson Nano, which is 8.5 times faster.

Finally, it is worth noting that our solution is scalable to other applications, including plants' disease detection, forest monitoring, and building construction monitoring.

3.4.4 Conclusion

In this work, an aerial platform for real-time hogweed detection is proposed. The platform consists of a UAV with the embedded system able to run AI on board. We performed the analysis and implemented several FCNN architectures for the hogweed detection problem. A comparative study followed it in terms of quality of detection (ROC AUC), power consumption, frame rate, and the area covered by a typical mission. Afterward, the aerial platform for real-time detection of harmful plants considering hogweed was designed and tested. This study included data collection, training the NN, inference on the embedded platform, experimentation, and evaluation. This study shows that different NN architectures successfully solve the semantic segmentation task for the aerial hogweed detection of two classes. For example, the *SegNet* has the best ROC AUC 0.969. Along with other networks, it can detect the hogweed, which was not initially labeled.

Modified *U-Net* architecture is characterised by the high frame rate (up to 0.7 FPS) and reasonable recognition quality (ROC AUC $\geq$ 0.938). This archi-

tecture demonstrates its applicability for real-time scenarios and its running on edge-computing devices along with low power consumption. Experimental results showed that one of the U-Net modifications could achieve 0.46 FPS on the NVIDIA Jetson Nano platform with the ROC AUC 0.958. The pilot study demonstrated that the proposed solution could explore 18.4 hectares within a 40-minute flight by the UAV equipped with a 12 MP 4000×2250 camera at 10 m altitude in detection mode. The proposed approach is promising for effectuating the weeds' removal and enables disruptive innovation in precision agriculture.

3.5 Morphing Wing for Control of the UAV

3.5.1 Introduction

Aircraft performs flight in multiple regimes with different speeds, Angle of Attack (AoA), sideslip angles, and at different altitudes. Designers usually choose the airfoil having the best performance for the cruise mode only or being able to stay suboptimal for all the flight regimes. It leads to a reduction of maximum lift-to-drag ratio for a specific regime and the deterioration of the overall performance. That is why the adaptive wing, with its ability to stay optimal for any of the flight regimes is a promising technology that could significantly improve the aircraft's performance and maneuverability during the flight. In this section, the performance of the wing with the traditional and adaptive mechanization is assessed. These wings have a flap and a slat. They were investigated using computer simulation followed by the experiments in the wind tunnel environment. This section also provides the design of the adaptive wing with adaptive flap and slat. All the investigations were performed for the two-dimensional (2D) airfoil under different Reynolds numbers and AoA. The results of the research were published in the journals **A. Menshchikov** and Somov [2019b], **A. Menshchikov** [2018b].

3.5.2 Methodology

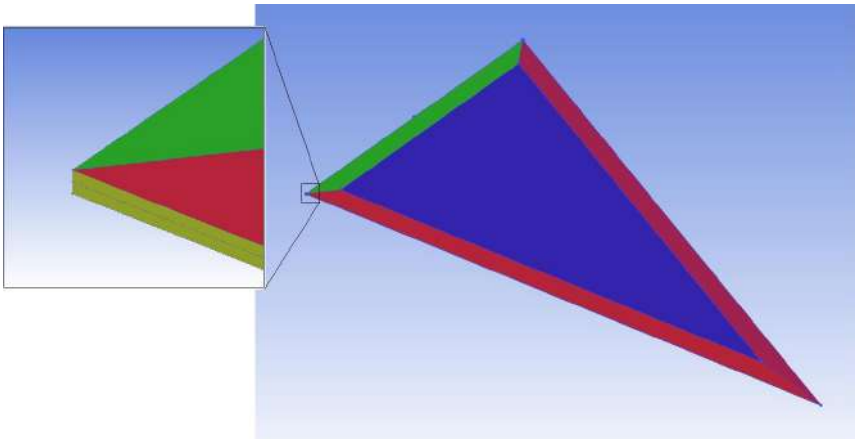
The research is aimed at investigating and evaluating the aerodynamic performance of the morphing wing with the adaptive slat and aileron in low Reynolds numbers. As it is focused on aerodynamics, the section of the wing used as a testbed is bulky, but can smoothly deform and achieve the proper outer shape. The research is split into three parts: the software benchmarking, the two-dimensional (2D) CFD analysis, and the wind tunnel experiment. In both cases, the wing was investigated separately in the undeflected and deflected modes for flap and slat.

Mesh validation and benchmarking

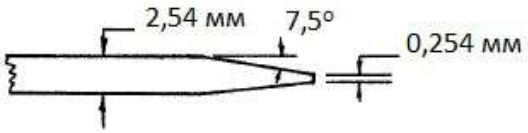
The validation and benchmarking of the mesh was performed in OpenFoam software, relying on the results of the wind tunnel experiment Kohlman and Wentz Jr [1968]. It represents an investigation of vortex breakdown over slender delta wing for high subsonic speeds. The investigated geometry (Fig. 3-34) has the following characteristics:

- The sweep angle is $\chi = 75°$
- The aspect ratio is $\lambda = 1.07$
- The relative thickness of the airfoil is $\bar{c} = 0.55\%$
- The wingspan is L = 244.9 mm
- The root chord is b = 457.3 mm
- The $b_a = 304.8mm$
- Edge chamfer is 7.5°
- The thickness of the edge is 0.254 mm

During the experiment, multiple aerodynamical phenomena occur: separation of flow, vortex generation, vortex breakdown, shockwave generation. Even though the geometry and flight regimes are irrelevant for the present study, this validation



(a) The geometry of the delta wing



(b) The blunt edge of the delta wing

Figure 3-34: The validatiob model

study shows the overall computational performance, possibilities, and reliability of the solver. The solver relies on Finite Element Analysis (FEA) methodology and solves a system of Navier-Stokes equations for 3D case:

$$\frac{\partial \rho}{\partial t} + div(\rho \bar{u}) = 0 \tag{3.7}$$

$$\frac{\partial \rho u}{\partial t} + div(\rho u \bar{u}) = -\frac{\partial p}{\partial x} + div(\mu \nabla u) + S_{M_x} \tag{3.8}$$

$$\frac{\partial \rho v}{\partial t} + div(\rho v \bar{u}) = -\frac{\partial p}{\partial y} + div(\mu \nabla v) + S_{M_y} \tag{3.9}$$

$$\frac{\partial \rho w}{\partial t} + div(\rho w \bar{u}) = -\frac{\partial p}{\partial z} + div(\mu \nabla w) + S_{M_z} \tag{3.10}$$

$$\frac{\partial \rho i}{\partial t} + div(\rho i \bar{u}) = -p div(\bar{u}) + div(k \nabla T) + \Phi + S_i \tag{3.11}$$

$$p = p(\rho, T) \tag{3.12}$$

$$i = i(\rho, T) \tag{3.13}$$

Where Φ is dissipation function and S_M - is mass forces.

Nowadays there are numerous of scientific approaches to model the turbulence.

The most relevant are the following:

1. Direct Numerical Simulation (DNS). This methodology works for low Reynolds numbers, nonstationary Navier-Stokes equations, and continuity equations. The disadvantage of DNS is that it is limited by computer's characteristics even nowadays. From a practical point of view, the statistics received from DNS model studies could be used to test and calibrate models, which focus on averaging Reynold's equations.

2. Large Eddy Simulation (LES). The core idea here is that large scale turbulence is computed using explicit equations. At the same time, the effects of smaller vortices could be resolved using low-passing filtering of the Navier-Stokes equations. It is still time-consuming; however, that methodology is much faster than DNS.

3. Reynolds averaged Navier – Stokes (RANS). The most well-known methodology of turbulence modeling. It is based on Navier-Stokes equations. The idea behind RANS is Reynolds decomposition, whereby an instantaneous quantity is decomposed into its time-averaged and fluctuating quantities Wilcox et al. [1998].

The most well-known RANS models are: $k-\epsilon, k-\omega$, Shear Stress Transport (SST). $k-\epsilon$ is the best for free flow far from walls. $k-\omega$ model is the best for near-the-wall flows. Whereas SST is the mixture of $k-\omega$ and $k-\epsilon$ model. It uses a $k-\epsilon$ model for free flow simulation (it is written in equations of a $k-\omega$ model) and $k-\omega$ model for near-the-wall precise flow simulation. Then these two solutions are united for different types of flow. SST model is similar to the traditional $k-\omega$ model. However, it has some significant advantages:

1. It includes advanced switching functions, which allows us to solve only $k-\omega$ equations near the wall and only $k-\epsilon$ equations far from the wall.

2. The definition of turbulent viscosity is changed to allow to make a model of the shift of shear stress.

Table 3.11: Properties of test meshes, used in validation study.

# of mesh	Type of mesh	Number of nodes
1	Unstructured	2 342 377
2	Structured	6 300 126
3	Hybrid	53 313 633

3. SST model includes cross-diffuse derivative.

4. There are many modeling constants.

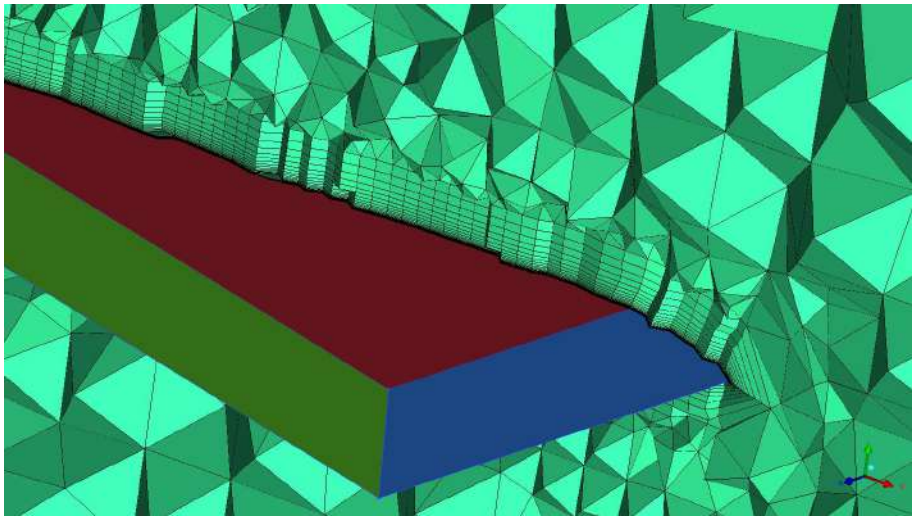
These features make the SST model precise and reliable for modeling a wide range of viscous flows apart from $k-\omega$ and other RANS turbulent models. That is why it is used in a numerical investigation. General mathematical representation of the SST model is similar to $k-\omega$ model:

$$\frac{\partial \rho k}{\partial t} + \frac{\partial \rho k u_i}{\partial x_i} = \frac{\partial}{\partial x_j}\left[\Gamma_k \frac{\partial k}{\partial x_j}\right] + G_k - Y_k + S_k \tag{3.14}$$

$$\frac{\partial \rho \omega}{\partial t} + \frac{\partial \rho \omega u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\Gamma_\omega \frac{\partial \omega}{\partial x_j}\right] + G_\omega - Y_\omega + D_\omega + S_\omega \tag{3.15}$$

Where G_k represents a generation of turbulent kinetic energy, and it could be defined in the same way as in the $k-\omega$ model. G_ω represents a generation of ω, and it is absolutely the same as in the standard $k-\omega$ model. Γ_k and Γ_ω characterize effective diffusion, k, and ω. Y_k and Y_ω characterize dissipation of k and ω due to turbulence. D_ω represents a cross-diffuse element. The calculation will be shown below. The elements S_k and S_ω are defined according to the particular case. The validation was performed for three types of meshes: unstructured, structured, and hybrid (Table 3.11).

Unstructured mesh (Fig. 3-35) is typically used in cases with complex geometry or when an adaptive mesh is needed. It is built automatically and fits the near-wall layer with layers of prismatic cells, whereas the rest of the volume is fit by tetrahedral cells. The detailing of the mesh strongly depends on the performance of the computer. That is why sometimes it is better to spend more time to create a structured mesh manually but make it bigger and precise. Unstructured meshes



(a) Prismatic layer over the surface of test model and tetrahedral cells around the model

(b) Structured mesh on the surface of the model

Figure 3-35: Structured and unstructured meshes

mostly fit for problems with complex geometry. Because creating such a mesh manually is usually a daunting task, but a computer could solve the problem in a matter of an hour. In contrast, structured mesh (Fig. 3-35b) is usually made manually. Even though it takes more time, any feature of mesh and blocking could be edited manually. Elements of structured mesh have a 3D box shape. However, in the mesh validation study, hybrid mesh proved to be the most precise.

The hybrid mesh is a mixture of structured and unstructured meshes. Detailed structured mesh sticks to the surface of the model to resolve the boundary layer, and the rest of the test volume is fit by automatically generated unstructured tetrahedral

mesh. The near-the-wall layer was 10^{-5} order of magnitude of the root chord to resolve boundary laminar sublayer by a single-cell near-the-wall layer. Hence, y^+ parameter is equal to 1. The rest of the boundary layer was resolved by 15 layers with a ratio = 1.2.

To perform numerical investigation the following regimes of flight from article Kohlman and Wentz Jr [1968] were chosen, AoA: 0°, 10°, 20°, 30°, 35°, 37.5°, 40°, 42.5°, 50°, 52.5°, with velocity of M = 0.15. Thus, 10 investigations were performed for each mesh. The turbulence level was set to high = 10%. The solver ran on the cluster computer with the following convergence criteria:

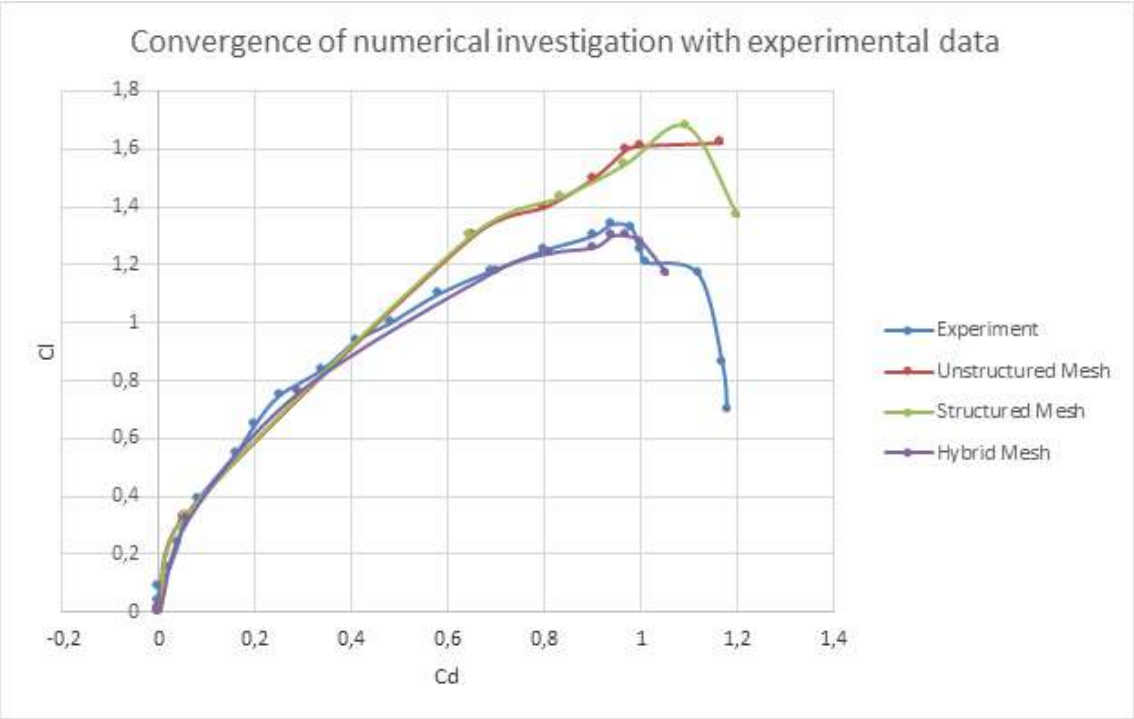
- Physical timescale: 0.001 [s]
- Residual Type: Root Mean Square Error (RMSE)
- Residual Target: 0.000001.
- Most of computations were resolved in 300 iterations. The results are the following (Fig. 3-36).

The following conclusions could be made from this verification study. Firstly, the hybrid mesh gives the closest to the practice results even for non-linear part of C_L vs. AoA. Secondly, the number of cells also matters: structured and unstructured meshes in the present study have the lowest quantity of cells. Thirdly, a linear part of C_L vs. AoA dependency could be resolved by any type of cells, which are built according to the abovementioned approach. Fourthly, hybrid mesh allows modeling even rare aerodynamic phenomena as asymmetric vortex breakdown, which usually occurs due to the influence of velocity profile, generated by one vortex, on the core of another vortex (Fig. 3-38).

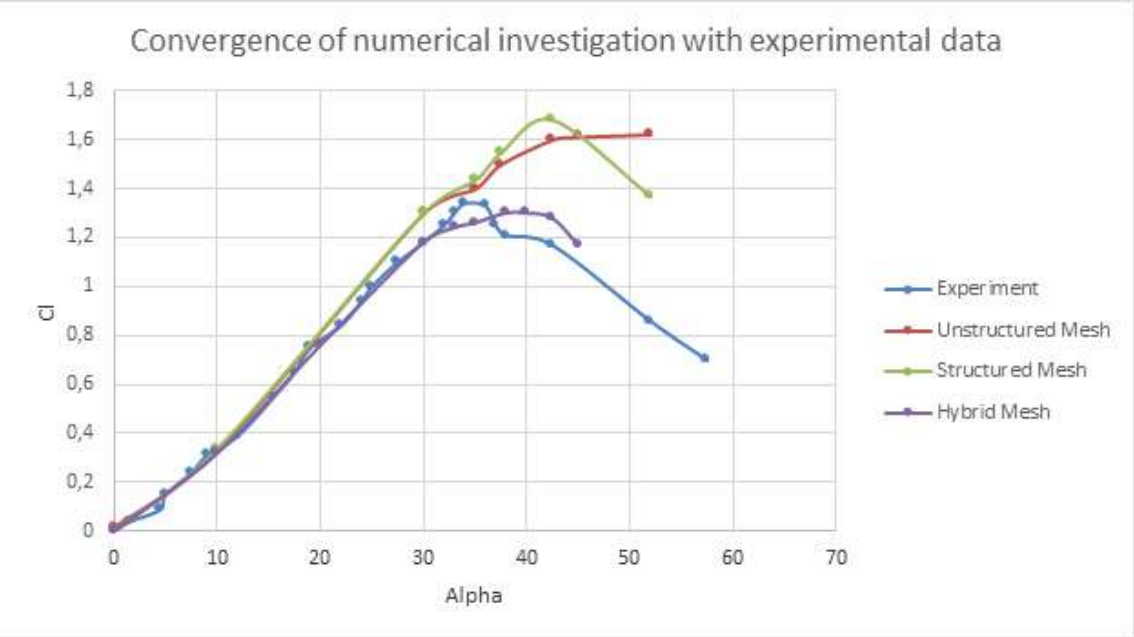
All the principles and options used in the current verification study were later used to design mesh around the wing with adaptive flap and aileron.

CFD Analysis of Morphing Wing

The model of investigation is the Clark YH airfoil with the following characteristics:

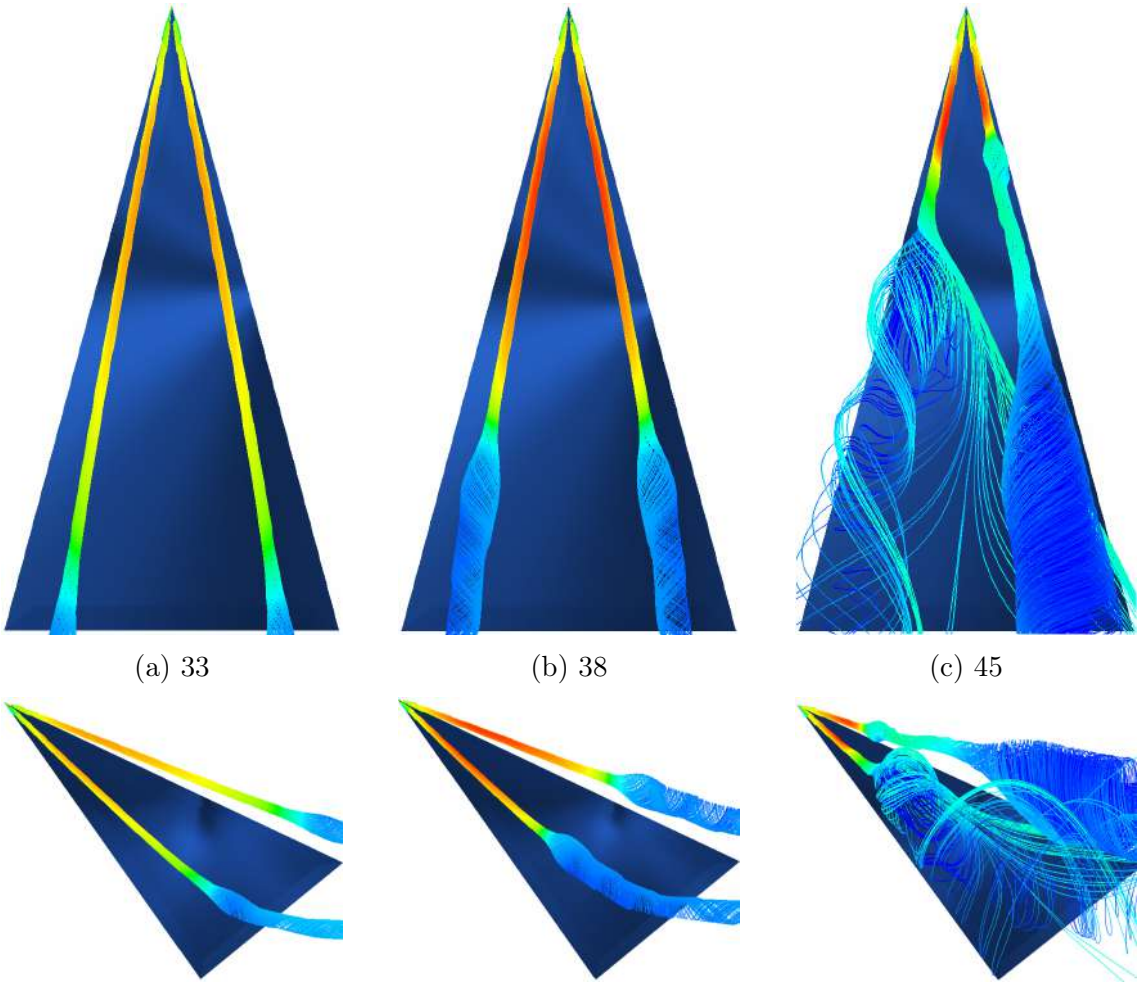


(a) $C_L vs. C_D$



(b) $C_L vs. AoA$

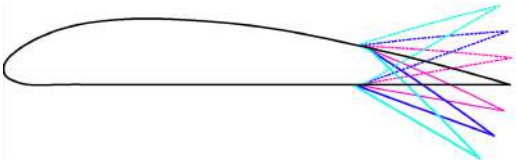
Figure 3-36: Convergence of numerical investigation with experimental data



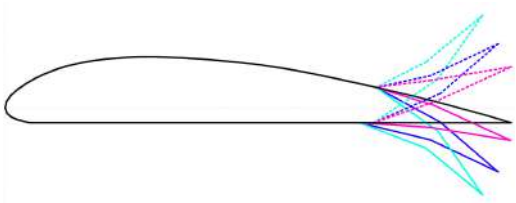
(a) 33 (b) 38 (c) 45

(d) 33 (e) 38 (f) 45

Figure 3-37: Vortex breakdown over slender delta wing with different AoA



(a) Traditional mechanization



(b) Adaptive mechanization

(c) Adaptive leading edge

Figure 3-38: The airfoil Clark YH with different types of mechanization. The color depicts angle of deflection: 0°-black; 10°-purple; 20°-blue; 30°-cyan. Solid line is for negative angles, dotted line is for positive angles.

- 11.9% thickness at 30% chord
- Max camber is 2.5% at 30% chord
- Chord is 11.2 cm

In all the experiments the flap is deflected from +30° to 30° with the 10° increment. The slat has only the 20° deflection (see Fig. 3-38).

The investigation was performed in the OpenFOAM software with the unstructured mesh. It has the 1 000 000 nodes 2D unstructured tetrahedral mesh with a prismatic near-wall layer. The wall has no-slip conditions for the proper modeling of viscous phenomena, e.g., the vortex generation or separation of the flow. The mesh has the $y^+ = 1$ parameter to resolve the boundary layer. This parameter represents the ratio of the laminar sublayer thickness to the first cell height and can be expressed in two ways:

$$y^+ = \frac{\rho U_\tau h}{\mu} \tag{3.16}$$

$$y^+ = \frac{y}{h} \tag{3.17}$$

where U_τ is the velocity in laminar sublayer, ρ is the density of the gas, μ is viscosity, h is the first cell height, and y is the thickness of the boundary sublayer. The height of the first cell was 10^{-5} order of magnitude of the chord to resolve the laminar sublayer of the boundary layer. The increment ratio was 1.2 to make a slow growth of the cell size away from the airfoil model. On the one hand, this made it possible to represent near-surface processes with high detalization. On the other hand, it reduces the computational difficulty for distant regions with the free stream. All the simulations were time-averaged. The turbulence was modeled with the RANS approach with the use of the SST model of turbulence. The numerical investigation was performed for all the geometrical models of adaptive and traditional mechanization configurations for different conditions:

- Reynolds number varies from 100 000 to 300 000 with the 100 000 increment.
- AoA varies from $-5°$ to $35°$ with the $0.5°$ increment.

Wind tunnel experiment

All the experiments were conducted in the open-return wind tunnel with low-Reynolds numbers (Fig. 3-39). Its test chamber is of octahedral shape in cross-section. The characteristics of the wind tunnel are the following:

- Maximum airflow velocity: 30 m/s
- Test chamber width x length: 0.8 m × 1.0 m
- Mean turbulence level: 7%

It is equipped with the particle generator and high FPS optical system for the particle dynamics detection. It consists of two high-speed cameras and a green

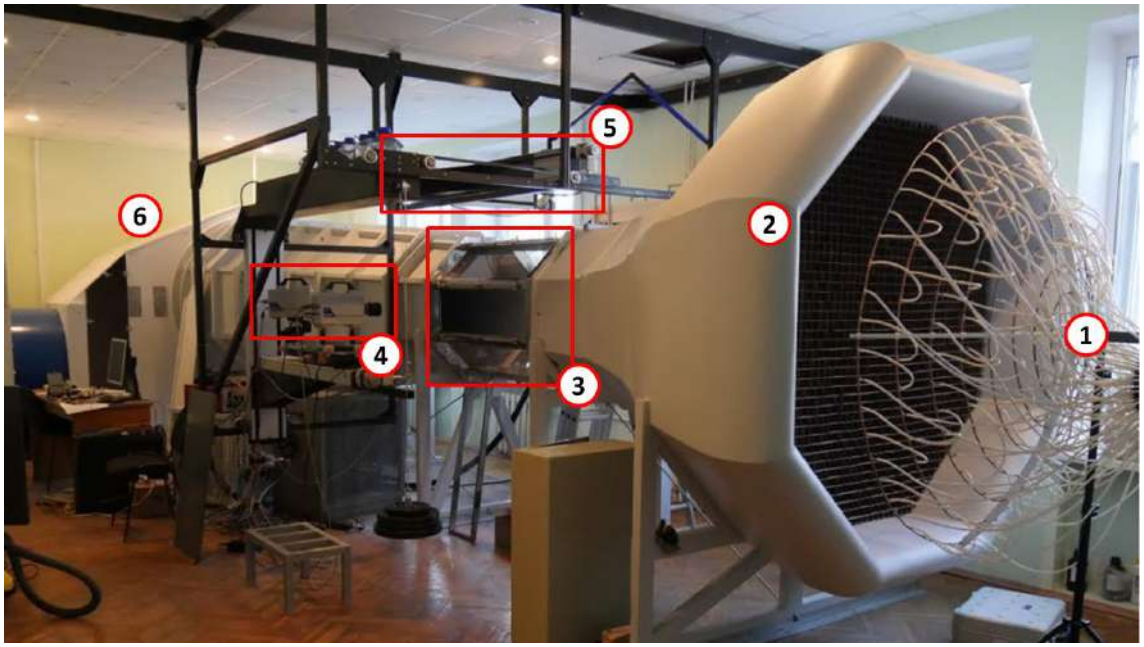
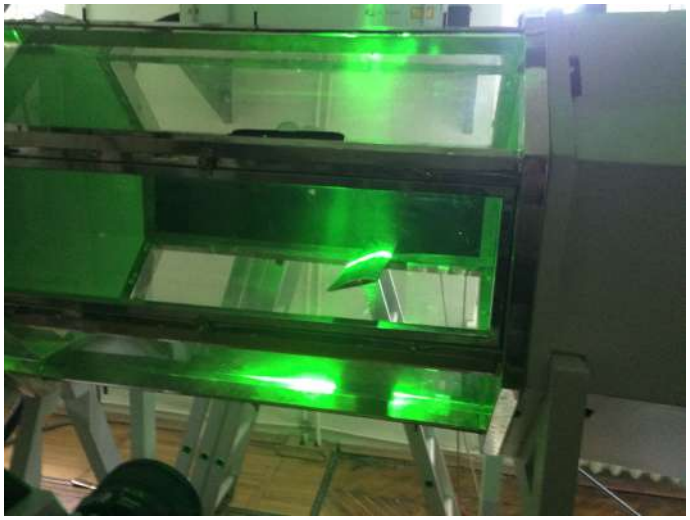


Figure 3-39: Wind tunnel; 1 – source of particles; 2 – inlet; 3 – test chamber; 4 – high speed cameras for videogrammetry; 5 – laser mounting system; 6 – electric motor and outlet.

laser. This methodology is widely known as Particle Image Velocimetry (PIV). In our case, particles are produced by the fume generator, and then they go to the inlet of the wind tunnel. A unique lens transforms the laser beam into the light sheet. Then it illuminates particles inside the test chamber. The reflected light goes to the high-speed cameras that capture the picture from different angles. The resultant video becomes an input file for the advanced videogrammetry software, which calculates every particle's position and velocity in the flow for every frame. The kinematic characteristics of these particles over time allow to calculate the pressure distribution over the object of investigation, e.g., airfoil, as it is shown in Fig. 3-40

The advantage of such a methodology is no touch measurements that do not produce any disturbance of the flow. The advanced software makes it possible to calculate the flow characteristics at every point with high accuracy. The disadvantage is that the setup only returns the 2D pictures of the flow. Furthermore, it works not simultaneously for the entire object, but the top or bottom surface at a time. The section of the adaptive wing has the 11.2 cm chord and the 40 cm wingspan. The morphing wing model for the wind tunnel experimentation was designed and

(a) Videogrammetry system in front of the wind tunnel



(b) The test chamber of the wind tunnel with the model during operation (the flow goes from the right side)

Figure 3-40: Experimental setup

Figure 3-41: The design of the adaptive wing segmentin SolidWorks software

tested in SolidWorks software (Fig. 3-41). The wing consists of 40 parts, made from ABS plastic using additive manufacturing. The skin of the wing was made from silicon. The mechanization of the wing has the following characteristics:

- Slat can deflect up to 30°
- Flap can deflect from −30° to +30°
- The thickness varies from 11.9% to 22%

All the control surfaces are deflected using 6 servos situated inside the wing. The Arduino Uno microcontroller controls them. The proposed construction is heavy for real flight; however, it is suitable for the wind tunnel experiment. Servomotors were chosen as actuators for the following reasons. Firstly, they are widespread in aviation

due to rapid and precise motion, high reliability. Secondly, they have a short latency. Thirdly, they are easy to substitute in case of malfunction. However, servomotors have significant disadvantages, which make them hard to use for adaptive structures – they have a high mass, are bulky, and cannot perform any other function apart from actuation. The last point makes it impossible to use servomotors as part of intelligent structures. On the other hand, there are smart materials, which, in contrast to servomotors, could have structural, actuation, and sensing capabilities. These materials include piezoceramics, piezopolymers, SMA, SMP, electrostrictors, magnetostrictors. Most of them are used in experimental models of MW and other intelligent structures Wada et al. [1990] Crawley [1994]. However, they have some features, extrinsic for servomotors. For example, piezomaterials, electrostrictors, and magnetostrictors have low deflection amplitude; they depend on the external electromagnetic field and have low stiffness and strain. Even though SMA and SMP have high stiffness and strain and potentially higher amplitude of deflection, they still require an additional heat source and heat sink, hence additional mass and power consumption. Along with that, SMA and SMP depend on external temperature and have high relaxation time. Even though they are used on practice for gentle maneuvering and landing (e.g., flaps of B-2 and prototype of Boeing's adaptive winglets, made from SMA), they do not apply to actuation in cases, where rapid response to control signal is required.

The prototype of the adaptive wing is demonstrated in Fig. 3-42. All these servos could work independently and could twist the trailing edge to redistribute the airload over the wingspan during the flight. However, the current study is focused on the 2D experiments, while the airload redistribution is a subject for future work and pre-flight experiments.

3.5.3 Control of the Morphing Wing

All the topics mentioned in the previous sections relate to the computer vision algorithms based on DNNs and their optimization for inference on the low-power embedded system. All this data is further used for path planning and trajectory reconstruction of the UAV. Even though this is a vast topic with many nuances,

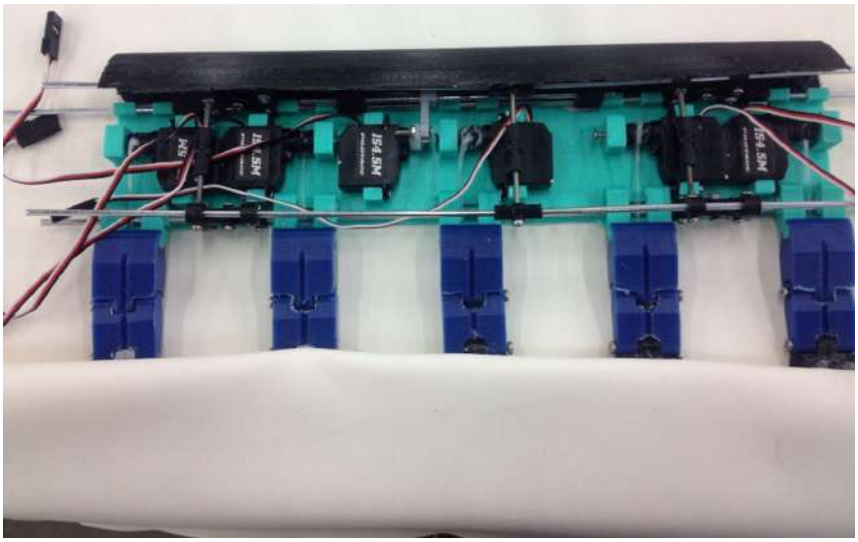


Figure 3-42: The manufacturing of the section of the morphing wing

the current research relies on well-known algorithms like RRT^* and A^*. What is more relevant for the study is the control of VTOL drone, fixed-wing drone, and fixed-wing drone with the MW.

The control of the eVTOL with morphing wing relies on the cascaded controller and an Extended Kalman Filter (EKF). The controller and are described in Appendix(A.2). There is a description of several flight modes enlisted below. The control entirely relies on the so-called Unified Control Model, which described in detail in the Appendix.

- Roll attitude hold

$$
\begin{aligned}
E_{\phi_t} &= \phi_t^c - \phi_t \\
D_t &= \frac{E_{\phi_t} - E_{\phi_{t-1}}}{\Delta t} \\
\delta a_t &= k_{p_\phi} E_t - k_{d_\phi} D_t
\end{aligned}
$$

- Course hold

$$E_{\chi_t} = \chi_t^c - \chi_t$$
$$I_{\chi_t} = I_{\chi_{t-1}} + E_{\chi_t}\Delta t$$
$$\phi^c = k_{p_\chi} E_{\chi_t} + k_{i_\chi} I_{\chi_t}$$

- Sideslip Hold

$$E_{\beta_t} = \beta_t^c - \beta_t$$
$$I_{\beta_t} = I_{\beta_{t-1}} + E_{\beta_t}\Delta t$$
$$\delta r = -k_{p_\beta} E_{\beta_t} - k_{i_\beta} I_{\beta_t}$$

- Pitch attitude hold

$$E_{\theta_t} = \theta_t^c - \theta_t$$
$$D_t = \frac{E_{\theta_t} - E_{\theta_{t-1}}}{\Delta t}$$
$$\delta e = k_{p_\theta} E_{\theta_t} + k_{d_\theta} D_{\theta_t}$$

- Altitude hold

$$E_{z_t} = z_t^c - z_t$$
$$I_{z_t} = I_{z_{t-1}} + E_{z_t}\Delta t$$
$$\theta^c = k_{p_z} E_{z_t} + k_{i_z} I_{z_t}$$

- Airspeed hold using commanded pitch

$$E_{V_t} = V_t^c - V_t$$
$$I_{V_t} = I_{V_{t-1}} + E_{V_t}\Delta t$$
$$\theta^c = k_{p_{V_2}} E_{V_t} + k_{i_{V_2}} I_{V_t}$$

- Airspeed hold using commanded throttle

$$E_{V_t} = V_t^c - V_t$$
$$I_{V_t} = I_{V_{t-1}} + E_{V_t}\Delta t$$
$$\delta t = \delta^* t + k_{p_V} E_{V_t} + k_{i_V} I_{V_t}$$

3.5.4 Experimental Results

The CFD study of the adaptive wing with the adaptive flap showed the improvement of aerodynamic characteristics. It is depicted in the polar curve - dependency of lift coefficient (C_L) from drag coefficient (C_D) C_L vs C_D (see Fig. 3-43) and C_L vs AoA (see Fig. 3-44) characteristics. The C_L vs. C_D characteristics show that in the same conditions, the adaptive wing has slightly lower C_L than the traditional wing. However, C_D for the adaptive wing is significantly lower than for the conventional wing under the similar condition and configuration. That is why the lift-to-drag ratio of the adaptive wing substantially increases. It leads to the reduction of power consumption of the aircraft. The C_L vs. C_D characteristic is in the polar reference frame; therefore, the modulus of the radius-vector and its inclination represent physical characteristics. The modulus of the radius-vector shows the total aerodynamic force, and the inclination angle shows the lift-to-drag ratio for specific conditions. If the radius vector is the tangential line which origins in the reference point, its inclination angle will represent the maximum lift-to-drag ratio possible in such configuration:

$$K_{max} = tan(\theta); \tag{3.18}$$

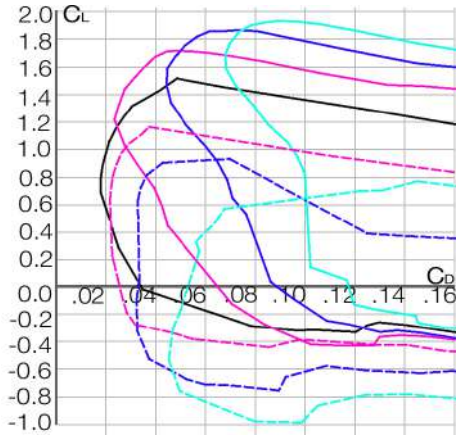
Where θ is inclination of the tangential.

It is shown (see Fig. 3-43) that the graphs are slightly higher for the adaptive wing than for the traditional one - hence the maximum achievable lift-to-drag ratio is also increased. All the C_L vs AoA (see Fig. 3-44) graphs slightly moved in the positive direction of the AoA (horizontal) axis. Hence, the stalling angle increased for 4° for all the configurations. It means an improvement in stability and maneuverability. The critical values of C_L (for stalling angle) are also increased in every case. It means the higher values of the lifting force before stalling and, consequently, the better controllability for the near-critical regimes. All the C_L vs. AoA characteristics in the stalling angle region are low-sloped. It means that aircraft don't lose controllability under stalling conditions.

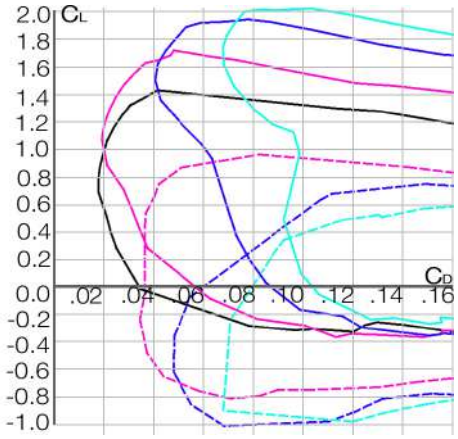
The experiment was performed for every deflection angle of the flap and the AoA from $-5°$ to $+35°$ with 5° step. The flow velocity had the following values: 10 m/s, 20 m/s, 30 m/s to achieve the 100 000, 200 000, and 300 000 Reynolds numbers. The pictures of the illuminated airflow over the wing were received (Fig. 3-45).

These pictures and videos were the input data for the OpenPIV software. As a result, all the important aerodynamic characteristics were calculated: lift, drag, pressure, and velocity distribution (see Fig. 3-46).

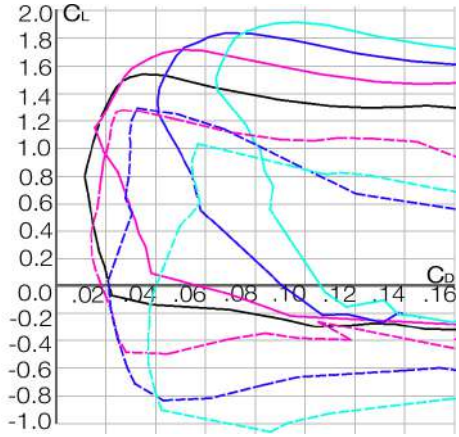
The analysis of the experimental results proved the initial hypothesis. Furthermore, the experiment demonstrates that the flap deflection ensures the increase of the lifting force and the lift-to-drag ratio. It also increases the torque, which means an improvement in the controllability and maneuverability of the aircraft. The deflection of the adaptive flap also results in a decrease of the drag force, thereby lowering the aircraft's power consumption. The experiment also showed a significant reduction of pressure over the wing's upper surface, which, in turn, increases the lifting force.



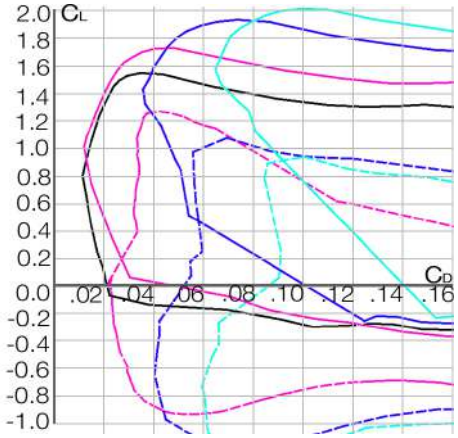
(a) Traditional Configuration; Re = 100 000



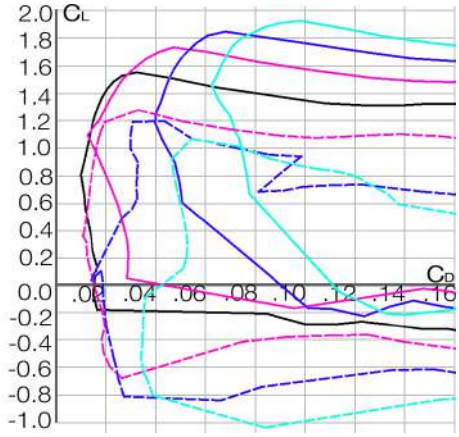
(b) Adaptive Configuration; Re = 100 000



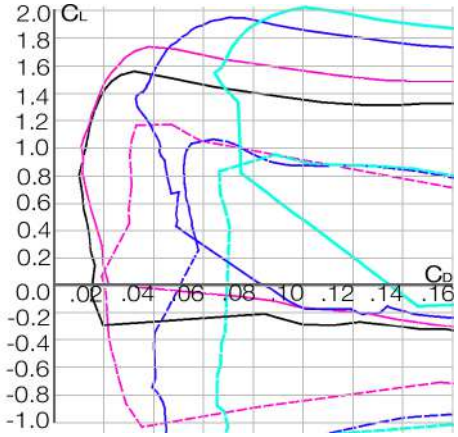
(c) Traditional Configuration; Re = 200 000



(d) Adaptive Configuration; Re = 200 000

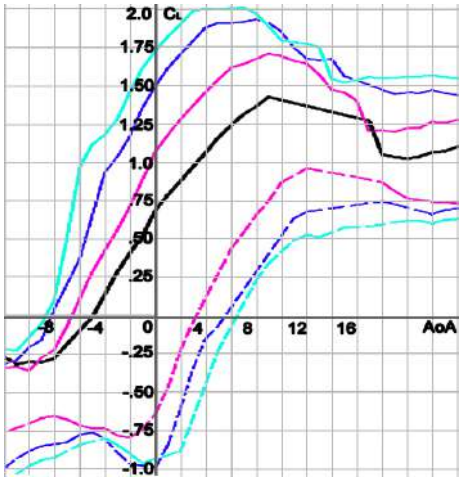


(e) Traditional Configuration; Re = 300 000

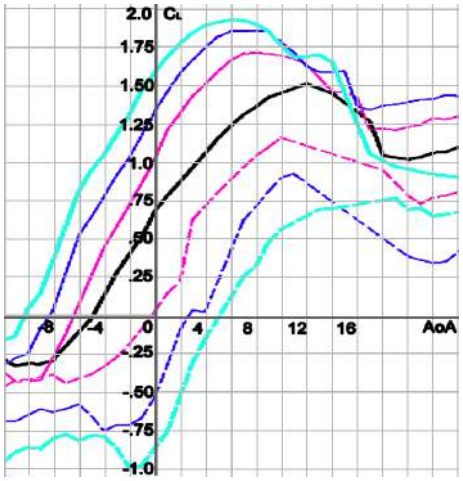


(f) Adaptive Configuration; Re = 300 000

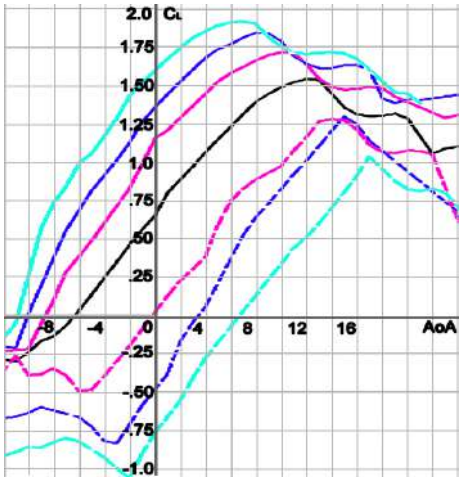
Figure 3-43: C_L vs C_D characteristics for different Reynolds numbers and configurations of the wing. Angles of deflection: 0°-black; 10°-purple; 20°-blue; 30° -cyan. The solid line is for negative angles, the dotted line is for positive angles.



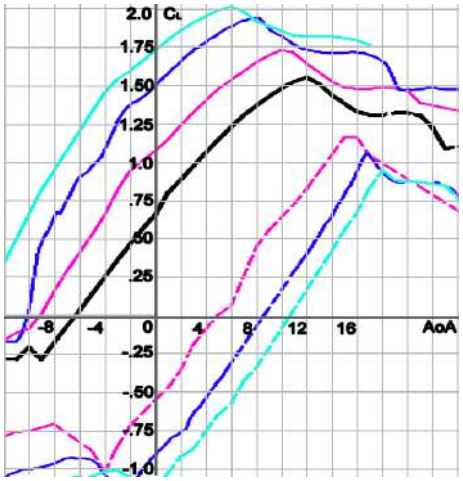
(a) Traditional Configuration; Re = 100 000



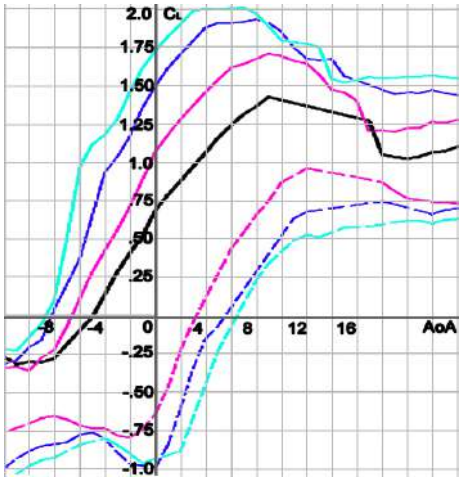
(b) Adaptive Configuration; Re = 100 000



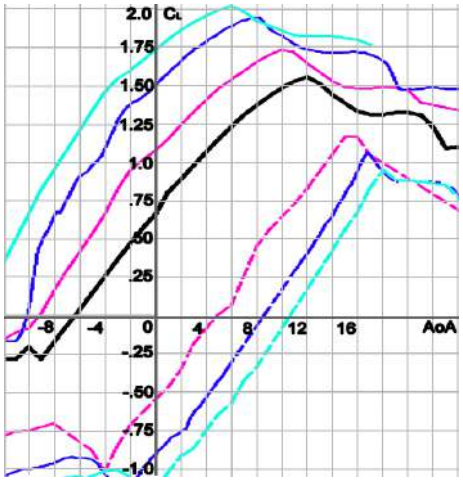
(c) Traditional Configuration; Re = 200 000



(d) Adaptive Configuration; Re = 200 000



(e) Traditional Configuration; Re = 300 000



(f) Adaptive Configuration; Re = 300 000

Figure 3-44: C_L vs AoA for different Reynolds numbers and configurations of the wing. Angles of deflection: 0

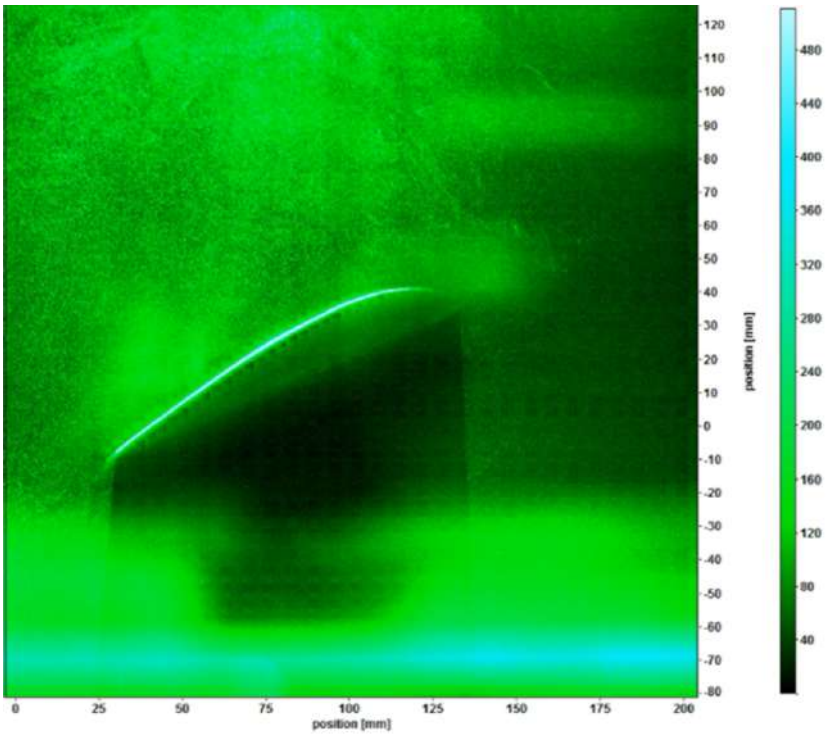


Figure 3-45: The test chamber of the wind tunnel with the laser, turned on

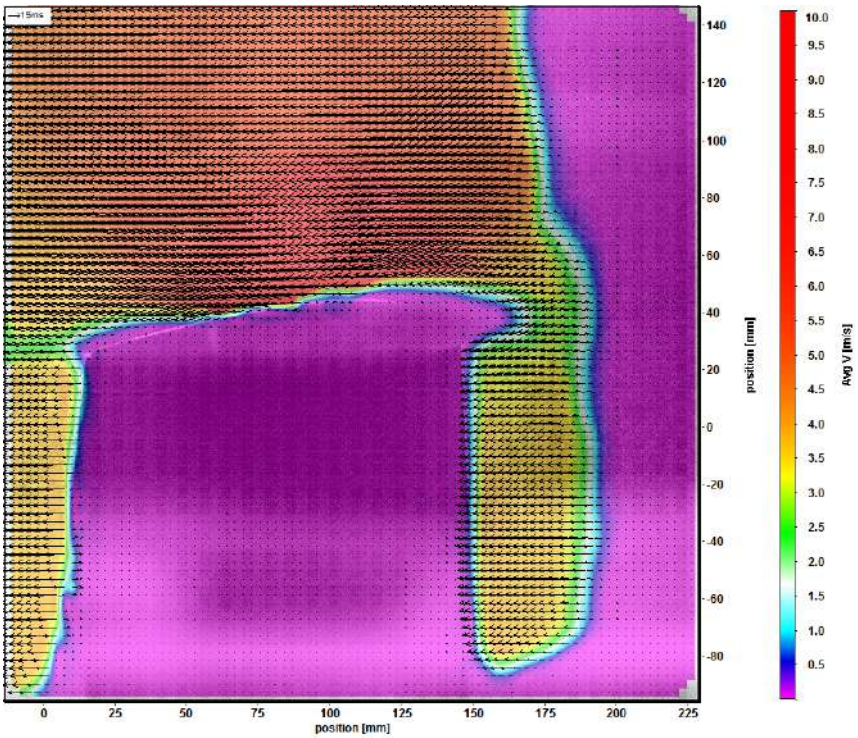


Figure 3-46: Visualization of the pressure distribution of the airflow over the segment of the adaptive wing

3.5.5 Conclusion

Adaptive mechanization demonstrated multiple advantages. First, the lift-to-drag ratio of the adaptive wing significantly increased. The result is the lower power consumption of the aircraft. Second, the maximum lift-to-drag ratio increased by up to 5% for all the configurations and regimes. The adaptive mechanization enhances the stalling angle for 4^o for all the configurations. Thus, it improves stability and maneuverability. The critical values of C_L increased in every case. Thus, the lifting force was increased. The controllability was improved for near-critical regimes. Therefore, it means an improvement in controllability under stalling conditions.

Moreover, in the investigation, the segment of the adaptive wing was successfully designed and manufactured. It reliably worked during the experiment in the wind tunnel. As a result of the experiment, the initial hypothesis (that the adaptive flap and slat might significantly improve the wing's aerodynamic characteristics) was successfully proved.

"Real generosity toward the future consists in giving all to the present."

Albert Camus

Chapter 4

Conclusion

This work demonstrates different steps of mathematical modeling and analysis of the airborne platform for analytics in precision agriculture. It includes the following steps:

- Development of neural networks and optimization for low-power embedded systems.
- Testing these platforms in various conditions: in the climate chamber, greenhouse, onboard of the UAV.
- Mathematical modeling of the control system for the UAV.
- Mathematical modeling and experimental investigation of the Morphing Wing for the UAV.

All these steps are necessary since the state-of-the-art UAVs for smart agriculture have to be equipped with the capabilities for fast data processing on board. However, such an ability significantly influences the UAV dynamics and control. The new requirement arises, which implies the UAV to be both agile and economical. That is why the development of the adaptive wing as a new way to lessen the energy consumption and increase the flight range, and controllability comes to the stage. The primary goal of this platform is the detection of hogweed and other harmful plants. However, it is a particular task for the dissertation, which opens a wide

vista for further development and application of such platforms for future precision agriculture.

4.1 Low-Power Embedded System for Monitoring in Precision Agriculture

In this work, multiple low-power embedded systems with AI capabilities were investigated. The experimental assessment on how to use a single embedded system efficiently and the embedded system empowered with an external GPU was carried out. It included several Single Board Computers with either external or internal GPU or VPU. All the systems showed their capability to be used as nodes for processing data in different conditions: a stationary environment, like a climate chamber in a lab or greenhouse, and on a mobile platform, like UAV. The autonomous system proved to be reliable for both scenarios. It might be used as an isolated system without an external power source or internet connection for a prolonged period of up to 180 days.

The system proved its applicability to run the neural networks of various types. They include CNN, FCNN, RNN LSTM. All of these networks were demonstrated.

- The first CNN for seed germination was optimized for inference on a low-power embedded system. It has capabilities for detecting the seed germination dynamics without extensive data transmission from the local nodes to a cloud computing server. The proposed approach is scalable and has a strong industrial impact as a powerful tool for assessing the performance of growing systems and predicting their future harvesting period. At the same time, it provides an opportunity for making optimization at the initial stage of plant growth. This optimization will further result in the optimal management of resources in the context of precision agriculture. With this end in view, a custom CNN architecture was proposed. Its primary function is seed recognition, which achieves the 97% accuracy and 83% of IoU. Using the CNN and computer vision, the sensing system can, first, localize them in the container

and, second, detect the germinated seeds. Even though the desktop computer is about 10 times faster than the embedded system, but a battery could power the last one. It is beneficial for the emerging autonomous applications in the scope of IoT.

- The second network, RNN LSTM, demonstrated its operation in the greenhouse environment onboard a similar system - single board computer, Raspberry Pi with external VPU, Intel Movidius. Performance evaluation of the proposed solution has demonstrated that the developed AI architecture based on an RNN LSTM is characterized by reasonable precision for the horizon of prediction. The proposed solution can be used as an autonomous tool for continuous plant growth dynamics monitoring for up to 180 days. With an actuating capability, the proposed approach promises to guarantee easy-to-deploy, generic, and robust optimization tools for precision agriculture.

- And finally, the FCNN for inference onboard of the airborne platform was demonstrated. Its primary goal is real-time hogweed detection. The platform consists of a UAV with the embedded system on board able to run AI. Several FCNN architectures for the hogweed detection problem were created and evaluated. They were evaluated in terms of IoU, power consumption, frame rate, and the area that can be covered by a typical mission. Afterward, I tested the aerial platform for real-time detection of harmful plants was designed and tested. This study included data collection, training the NN, inference on the embedded platform, experimentation, and evaluation. The performance of FCNNs varies.

  - For example, the *RefineNet* has the best IoU equal to 47% and the area coverage. It detects the hogweed, which was not initially labeled with high detalisation. However, this solution is not the best in terms of power consumption and frame rate.

  - However, modified *UNet* architecture is characterised by the high frame rate (FPS = 0.64), reasonable accuracy (IoU = 37%) and detalisation.

Along with the low power consumption, this architecture has a high potential for its application in a real-time scenario.

– *SegNet* has a smooth mask and works well in terms of accuracy and detalisation. However, it is the largest and the heaviest NN. these two-point prevent its application in real-time.

The proposed platform is scalable to other detection applications and opens opportunities for harmful plant elimination, enhancing the capabilities for future precision agriculture.

4.2 Control System and Morphing Wing

As it was shown in part, devoted to hogweed detection, the successful application of such a system in practice requires an advance control system and new physical means to perform the agile and economical flight. To meet this requirement, I propose the control system for the eVTOL drone along with Morphing Wing.

The control system consists of two parts: estimator and a cascaded controller. The estimator relies on the EKF algorithm and allows to implement the data fusion system, which aggregates heterogeneous data from multiple UAV sensors: IMU, GPS, magnetometer. All this data is further used as a part of the input data of the cascaded controller. The approach reported in this work is tested in simulations and experiments with a calculated standard deviation, which is up to 10 cm. Further, these algorithms, along with the Unified Control Model for fixed-wing aircraft, was implemented to control the Morphing Wing.

Adaptive mechanization of the Morphing Wing demonstrated multiple advantages. Firstly, the lift-to-drag ratio of the adaptive wing significantly increased. The result is the lower power consumption of the aircraft. Secondly, the maximum lift-to-drag ratio also increased up to 5% for all the configurations and regimes. The application of adaptive mechanization resulted in the stalling angle enhancement for $\sim 4^\circ$ for all the configurations. It means an improvement in stability and maneuverability. The critical values of C_L increased in every case. It means the higher values of the lifting force and better controllability for near-critical regimes.

It means an improvement in controllability under stalling conditions. Moreover, in the course of the investigation, the adaptive wing segment was successfully designed and manufactured. It reliably worked during the experiment in the wind tunnel. As a result of the experiment, the initial hypothesis (that the adaptive flap and slat might significantly improve the wing's aerodynamic characteristics) was successfully proved.

To sum up, this dissertation is devoted to mathematical modeling and analysis of next-generation UAV platforms for smart agriculture. It was shown that the ability to process data-intensive computing immediately onboard the UAV is crucial for modern farming. However, implementation of such functionality requires advances both in terms of algorithms and physical means of control. All these challenges were addressed employing FCNN, optimized for the low-power embedded system, control system, and morphing wing. The task of hogweed detection was selected to demonstrate the capabilities of such a platform. However, the proposed platform and the methodology of developing such a platform open new opportunities for IoT, UAV, and AI development in the area of precision agriculture.

Glossary

ABS Acrylonitrile Butadiene Styrene. 119

AI Artificial Intelligence. 15, 19, 20

ANFIS Adaptive Neuro-Fuzzy Inference System. 29

ANN Artificial Neural Network. 29

AoA Angle of Attack. 44, 107, 113

AUC Area Under the Curve. 98

BN Batch Normalization. 94

CFD Computational Fluid Dynamics. 23, 46, 108

CNC Computer Numerical Control. 23

CNN Convolutional Neural Network. 16, 26, 36, 47, 48

CPU Central Processing Unit. 21, 39, 51

DL Deep Learning. 88

DNN Deep Neural Network. 37, 75

DNS Direct Numerical Simulation. 110

DOF Dimensions of Freedom. 162

DOTA Dataset for Object Detection in Aerial Images. 38

DSP Digital Signal Processor. 39

EKF Extended Kalman Filter. 40, 121, 163

ERS Earth Remote Sensing. 38

eVTOL electric Vertical Take-Off and Landing. 15, 17, 23

FCNN Fully Convolutional Neural Network. 15, 16, 22, 24, 26, 48, 54, 76, 87, 88

FEA Finite Element Analysis. 109

FOV Field of View. 31

FPGA Field-Programmable Gate Array. 39

FPS Frames per Second. 50, 51, 99, 117

GEO Geostationary. 33

GPS Geo Positioning System. 40

GPU Graphics Processing Unit. 21, 22, 39, 48, 76

GSD Ground Sample Distance. 31, 90

HA hardware acceleration. 39

HD High Definition. 63, 64

ILP Instruction Level Parallelism. 67

IMU Inertial Measurement Unit. 40

IoT Internet of Things. 36

IoU Intersection over Union. 66, 68, 72, 88, 97

IR Infrared. 174

LEO Low Earth Orbit. 33

LES Large Eddy Simulation. 110

LSTM Long-Short Term Memory. 76, 77, 87

MAV Micro Aerial Vehicles. 44

ML Machine Learning. 25, 26

MLR Multiple Linear Regression. 29

MoCap Motion Capture. 173

MRF Markov Random Field. 16

MRI Magnetic Resonance Tomograph. 36

MW Morphing Wing. 17–20, 23, 24, 41, 121

NCS Neural Computer Stick. 47, 49, 56, 63, 67, 72, 73, 82

NDVI Normalized Difference Vegetation Index. 34

NED North-East-Down. 162

PIV Particle Image Velocimetry. 23, 118

R-CNN Regions with CNN Features. 37

R-FCN Region-based Fully Convolutional Neural Network. 37

RAM Random Access Memory. 48

RANS Reynolds averaged Navier – Stokes. 110, 117

RF Random Forest. 15

RMSE Root Mean Square Error. 80

RNN Recurrent Neural Network. 76, 87

ROC Receiver Operating Characteristic. 98

ROS Robot Operating System. 174

RPi Raspberry Pi. 47, 48, 51, 63, 72, 82

SBC Single Board Computer. 39, 47, 51, 72, 75, 82, 89

SHAVE Streaming Hybrid Architecture Vector Engine. 49, 53, 67

SMA Smart Memory Alloys. 42, 120

SMP Smart Memory Polymers. 42, 120

SoC System-on-Chip. 39

SSD Single Shot Detector. 37

SST Shear Stress Transport. 110, 117

TPU Tensor Processing Unit. 48

UAV Unmanned Aerial Vehicle. 10, 15, 19, 20, 24, 30, 40, 46, 47, 51, 53, 88, 107–128

UKF Unscented Kalman Filter. 40

USB Universal Serial Bus. 73

VA Volt-Ampere. 83

VLIW Very Long Instruction Word. 67

VPU Visual Processing Unit. 21, 39, 48, 62, 63, 67

WSN Wireless Sensor Networks. 27–30

YOLO You Only Look Once. 37

YOLT You Only Look Twice. 37

Bibliography

Fotokite simulator. https://github.com/udacity/FCND-Controls-CPP, 2018.

NNAPI. https://developer.android.com/ndk/guides/neuralnetworks, 2019. [Online; accessed 07-May-2019].

Intel Movidius Myriad 2 SHAVE cores. https://en.wikichip.org/wiki/movidius/microarchitectures/shave_v2.0, 2019. [Online; accessed 08-May-2019].

TensorFlow Mobile. https://www.tensorflow.org/lite, 2019. [Online; accessed 07-May-2019].

Heracleum dataset. https://github.com/DLopatkin/Heracleum-Dataset, 2019.

Extended kalman filter navigation overview and tuning, Accessed March 5, 2018. http://ardupilot.org/dev/docs/extended-kalman-filter.html#extended-kalman-filter.

Niels Aage, Erik Andreassen, Boyan S Lazarov, and Ole Sigmund. Giga-voxel computational morphogenesis for structural design. *Nature*, 550(7674):84–86, 2017.

P. Abouzar, D. G. Michelson, and M. Hamdi. Rssi-based distributed self-localization for wireless sensor networks used in precision agriculture. *IEEE Transactions on Wireless Communications*, 15(10):6638–6650, Oct 2016. ISSN 1536-1276. doi:10.1109/TWC.2016.2586844.

Telmo Adão, Jonáš Hruška, Luís Pádua, José Bessa, Emanuel Peres, Raul Morais, and Joaquim Sousa. Hyperspectral imaging: A review on uav-based sensors, data processing and applications for agriculture and forestry. *Remote Sensing*, 9(11): 1110, 2017.

Amir H. Alavi, Pengcheng Jiao, William G. Buttlar, and Nizar Lajnef. Internet of things-enabled smart cities: State-of-the-art and future trends. *Measurement*, 129:589 – 606, 2018. ISSN 0263-2241. doi:https://doi.org/10.1016/j.measurement.2018.07.067.

Jesus Alejandro Franco, Juan Carlos Jauregui, Andres Carbajal, and Manuel Toledano-Ayala. Shape morphing mechanism for improving wind turbines performance. *Journal of Energy Resources Technology*, 139(5), 2017.

I. Ali, F. Cawkwell, E. Dwyer, and S. Green. Modeling managed grassland biomass estimation by using multitemporal remote sensing data-a machine learning approach. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(7):3254–3264, July 2017a. ISSN 1939-1404. doi:10.1109/JSTARS.2016.2561618.

Iftikhar Ali, Fiona Cawkwell, Edward Dwyer, and Stuart Green. Modeling managed grassland biomass estimation by using multitemporal remote sensing data—a machine learning approach. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(7):3254–3264, 2017b.

Moustafa Alzantot, Yingnan Wang, Zhengshuang Ren, and Mani B Srivastava. Rstensorflow: Gpu enabled tensorflow for deep learning on commodity android devices. In *Proceedings of the 1st International Workshop on Deep Learning for Mobile Systems and Applications*, pages 7–12. ACM, 2017.

William Andrew, Colin Greatwood, and Tilo Burghardt. Aerial animal biometrics: Individual friesian cattle recovery and visual identification via an autonomous uav with onboard deep inference. *arXiv preprint arXiv:1907.05310*, 2019.

Danny Awty-Carroll, John Clifton-Brown, and Paul Robson. Using k-nn to analyse images of diverse germination phenotypes and detect single seed germination in miscanthus sinensis. *Plant methods*, 14(1):5, 2018.

Jan Axelson. *USB complete: the developer's guide*. Lakeview research LLC, 2015.

Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5221–5229, 2017.

M Bashir and P Rajendran. Static structural analysis of a variable span morphing wing for unmanned aerial vehicle. In *IOP Conference Series: Materials Science and Engineering*, volume 370, page 012040. IOP Publishing, 2018.

Pedro Bello and Kent J Bradford. Single-seed oxygen consumption measurements and population-based threshold models link respiration and germination rates under diverse conditions. *Seed Science Research*, 26(3):199–221, 2016.

Andrea E Bergamini, Manuel Zündel, Edgar A Flores Parra, Tommaso Delpero, Massimo Ruzzene, and Paolo Ermanni. Hybrid dispersive media with controllable wave propagation: A new take on smart materials. *Journal of Applied Physics*, 118(15):154310, 2015.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

Mariusz Bojarski, Philip Yeres, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Lawrence Jackel, and Urs Muller. Explaining how a deep neural network trained with end-to-end learning steers a car. *arXiv preprint arXiv:1704.07911*, 2017.

Margherita Bonetto, Pavel Korshunov, Giovanni Ramponi, and Touradj Ebrahimi. Privacy in mini-drone based video surveillance. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 4, pages 1–6. IEEE, 2015.

Adam Bry, Abraham Bachrach, and Nicholas Roy. State estimation for aggressive flight in gps-denied environments using onboard sensing. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1–8. IEEE, 2012.

David A Burdette, Gaetan K Kenway, Zhoujie Lyu, and Joaquim Martins. Aerostructural design optimization of an adaptive morphing trailing edge wing. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1129, 2015.

TF Burks, SA Shearer, Richard S Gates, and KD Donohue. Backpropagation neural network design and evaluation for classifying weed species using color image texture. *Transactions of the ASAE*, 43(4):1029, 2000.

Sebastian Candiago, Fabio Remondino, Michaela De Giglio, Marco Dubbini, and Mario Gattelli. Evaluating multispectral images and vegetation indices for precision farming applications from uav images. *Remote sensing*, 7(4):4026–4047, 2015.

Adrian Carrio, Carlos Sampedro, Alejandro Rodriguez-Ramos, and Pascual Campoy. A review of deep learning methods and applications for unmanned aerial vehicles. *Journal of Sensors*, 2017, 2017.

Pablo Chamoso, William Raveane, Victor Parra, and Angélica González. Uavs applied to the counting and monitoring of animals. In *Ambient intelligence-software and applications*, pages 71–80. Springer, 2014.

Ayan Chaudhury, Christopher Ward, Ali Talasaz, Alexander G Ivanov, Norman PA Huner, Bernard Grodzinski, Rajni V Patel, and John L Barron. Computer vision based autonomous robotic system for 3d plant growth measurement. In *2015 12th Conference on Computer and Robot Vision (CRV)*, pages 290–296. IEEE, 2015.

J. Chauhan, S. Seneviratne, Y. Hu, A. Misra, A. Seneviratne, and Y. Lee. Breathing-based authentication on resource-constrained iot devices using recurrent neural networks. *Computer*, 51(5):60–67, May 2018. ISSN 0018-9162. doi:10.1109/MC.2018.2381119.

Weilin Chen, Xianmin Zhang, and Sergej Fatikow. A novel compliant orthogonal displacement amplification mechanism and its application in micro-grasping.

In *2016 International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS)*, pages 1–8. IEEE, 2016.

Guo-Jian Cheng, Li-Ting Liu, Xin-Jian Qiang, and Ye Liu. Industry 4.0 development and application of intelligent manufacturing. In *2016 international conference on information system and artificial intelligence (ISAI)*, pages 407–410. IEEE, 2016.

Anna Chlingaryan, Salah Sukkarieh, and Brett Whelan. Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review. *Computers and Electronics in Agriculture*, 151:61–69, 2018.

AFRL CLIF. dataset over ohio state university, 2007.

M. J. W. Cock and M. K. Seier. The scope for biological control of giant hogweed, heracleum mantegazzianum. In *Ecology and management of giant hogweed (Heracleum mantegazzianum)*, pages 255–271. CABI. doi:10.1079/9781845932060.0255.

Lucian Codrescu, Willie Anderson, Suresh Venkumanhanti, Mao Zeng, Erich Plondke, Chris Koob, Ajay Ingle, Charles Tabony, and Rick Maule. Hexagon dsp: An architecture optimized for mobile multimedia and communications. *IEEE Micro*, 34(2):34–43, 2014.

Ismael Colomina and Pere Molina. Unmanned aerial systems for photogrammetry and remote sensing: A review. *ISPRS Journal of photogrammetry and remote sensing*, 92:79–97, 2014.

John L Crassidis, F Landis Markley, and Yang Cheng. Survey of nonlinear attitude estimation methods. *Journal of guidance, control, and dynamics*, 30(1):12–28, 2007.

Edward F Crawley. Intelligent structures for aerospace-a technology overview and assessment. *AIAA journal*, 32(8):1689–1699, 1994.

Edward F Crawley and Kenneth B Lazarus. Induced strain actuation of isotropic and anisotropic plates. *AIAA journal*, 29(6):944–951, 1991.

Roberto Cristi and Murali Tummala. Multirate, multiresolution, recursive kalman filter. *Signal Processing*, 80(9):1945–1958, 2000.

Jeffrey A Cruz, Xi Yin, Xiaoming Liu, Saif M Imran, Daniel D Morris, David M Kramer, and Jin Chen. Multi-modality imagery database for plant phenotyping. *Machine Vision and Applications*, 27(5):735–749, 2016.

Jifeng Dai, Kaiming He, and Jian Sun. Convolutional feature masking for joint object and stuff segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3992–4000, 2015.

Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

Abhiram Das, Hannah Schneider, James Burridge, Ana Karine Martinez Ascanio, Tobias Wojciechowski, Christopher N Topp, Jonathan P Lynch, Joshua S Weitz, and Alexander Bucksch. Digital imaging of root traits (dirt): a high-throughput computing and collaboration platform for field-based root phenomics. *Plant methods*, 11(1):51, 2015.

PC P De Silva and PC A De Silva. Ipanera: An industry 4.0 based architecture for distributed soil-less food production systems. In *Manufacturing & Industrial Engineering Symposium (MIES)*, pages 1–5. IEEE, 2016.

Nameirakpam Dhanachandra, Khumanthem Manglem, and Yambem Jina Chanu. Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 54:764–771, 2015.

Maurilio Di Cicco, Ciro Potena, Giorgio Grisetti, and Alberto Pretto. Automatic model based dataset generation for fast and accurate crop and weeds detection. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5188–5195. IEEE, 2017.

Matteo Di Luca, Stefano Mintchev, G Heitz, Flavio Noca, and Dario Floreano. Bioinspired morphing wings for extended flight envelope and roll control of small drones. *Interface focus*, 7(1):20160092, 2017.

Ignazio Dimino, Leonardo Lecce, Rosario Pecora, et al. *Morphing Wing Technologies: Large Commercial Aircraft and Civil Helicopters*. Butterworth-Heinemann, 2017.

S Ducournau, A Feutry, P Plainchault, P Revollon, Bertrand Vigouroux, and MH Wagner. Using computer vision to monitor germination time course of sunflower (helianthus annuus l.) seeds. *Seed Science and Technology*, 33(2):329–340, 2005.

Siddhartha Dutta, Jeffrey A Cruz, Yuhua Jiao, Jin Chen, David M Kramer, and Katherine W Osteryoung. Non-invasive, whole-plant imaging of chloroplast movement and chlorophyll fluorescence reveals photosynthetic phenotypes independent of chloroplast photorelocation defects in chloroplast division mutants. *The Plant Journal*, 84(2):428–442, 2015.

O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow, and M. N. Hindia. An overview of internet of things (iot) and data analytics in agriculture: Benefits and challenges. *IEEE Internet of Things Journal*, pages 1–1, 2018a. ISSN 2327-4662. doi:10.1109/JIOT.2018.2844296.

Olakunle Elijah, Tharek Abdul Rahman, Igbafe Orikumhi, Chee Yen Leow, and MHD Nour Hindia. An overview of internet of things (iot) and data analytics in agriculture: Benefits and challenges. *IEEE Internet of Things Journal*, 2018b.

Arif Tanju Erdem and Ali Özer Ercan. Fusing inertial sensor data in an extended kalman filter for 3d camera tracking. *IEEE Transactions on Image Processing*, 24 (2):538–548, 2015.

Patrick Eugster, Vinaitheerthan Sundaram, and Xiangyu Zhang. Debugging the internet of things: The case of wireless sensor networks. *IEEE Software*, (1):1–1, 2015.

Yin Fan, Xiangju Lu, Dian Li, and Yuanliu Liu. Video-based emotion recognition using cnn-rnn and c3d hybrid networks. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pages 445–450, 2016.

Mulham Fawakherji, Ali Youssef, Domenico Bloisi, Alberto Pretto, and Daniele Nardi. Crop and weeds classification for precision agriculture using context-independent pixel-wise segmentation. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pages 146–152. IEEE, 2019.

Andre Duarte BL Ferreira, Paulo RO Novoa, and Antonio Torres Marques. Multifunctional material systems: a state-of-the-art review. *Composite Structures*, 151: 3–35, 2016.

Francesco Flammini, Riccardo Naddei, Concetta Pragliola, and Giovanni Smarra. Towards automated drone surveillance in railways: State-of-the-art and future directions. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 336–348. Springer, 2016.

Frank Forcella, Roberto L Benech Arnold, Rudolfo Sanchez, and Claudio M Ghersa. Modeling seedling emergence. *Field Crops Research*, 67(2):123–139, 2000.

P Gamboa, Jose Vale, FJP Lau, and Afzal Suleman. Optimization of a morphing wing based on coupled aerodynamic and structural constraints. *AIAA journal*, 47 (9):2087–2104, 2009.

Paolo Gaudenzi. *Smart structures: physical behaviour, mathematical modelling and applications*. John Wiley & Sons, 2009.

Michel Edmond Ghanem, Hélène Marrou, and Thomas R Sinclair. Physiological phenotyping of plants for crop improvement. *Trends in Plant Science*, 20(3): 139–144, 2015.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

A Große-Stoltenberg, C Hellmann, J Thiele, C Werner, and J Oldeland. Early detection of gpp-related regime shifts after plant invasion by integrating imaging spectroscopy with airborne lidar. *Remote sensing of environment*, 209:780–792, 2018.

Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354 – 377, 2018. ISSN 0031-3203. doi:https://doi.org/10.1016/j.patcog.2017.10.013. URL http://www.sciencedirect.com/science/article/pii/S0031320317304120.

Tianhao Guo, Zhongxi Hou, and Bingjie Zhu. Dynamic modeling and active morphing trajectory-attitude separation control approach for gull-wing aircraft. *IEEE Access*, 5:17006–17019, 2017.

G. S. Gupta and V. M. Quan. Multi-sensor integrated system for wireless monitoring of greenhouse environment. In *2018 IEEE Sensors Applications Symposium (SAS)*, pages 1–6, March 2018. doi:10.1109/SAS.2018.8336723.

Alexandru Gurghian, Tejaswi Koduri, Smita V Bailur, Kyle J Carey, and Vidya N Murali. Deeplanes: End-to-end lane position estimation using deep neural networksa. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 38–45, 2016.

Jiakun Han, Zongjing Yuan, and Gang Chen. Effects of kinematic parameters on three-dimensional flapping wing at low reynolds number. *Physics of Fluids*, 30 (8):081901, 2018.

Sebastian Haug, Andreas Michaels, Peter Biber, and Jörn Ostermann. Plant classification system for crop/weed discrimination without segmentation. In *IEEE winter conference on applications of computer vision*, pages 1142–1149. IEEE, 2014.

H-J He, C Zheng, and D-W Sun. Image segmentation techniques. In *Computer Vision Technology for Food Quality Evaluation*, pages 45–63. Elsevier, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016b.

Walter T Higgins. A comparison of complementary and kalman filtering. *IEEE Transactions on Aerospace and Electronic Systems*, (3):321–325, 1975.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Kan Hong, Xiaoling Liu, Guodong Liu, and Wentao Chen. Detection of physical stress using multispectral imaging. *Neurocomputing*, 329:116–128, 2019.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Jan F Humplík, Dušan Lazár, Alexandra Husičková, and Lukáš Spíchal. Automated phenotyping of plant shoots using imaging methods for analysis of plant stress responses–a review. *Plant methods*, 11(1):29, 2015.

Dryver R Huston and Justin M Bond. Shape sensing of inflatable aerospace structures with fiber optic curvature rosettes. In *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2017*, volume 10168, page 101681P. International Society for Optics and Photonics, 2017.

Janna Huuskonen and Timo Oksanen. Soil sampling with drones and augmented reality in precision agriculture. *Computers and Electronics in Agriculture*, 154:25 – 35, 2018. ISSN 0168-1699. doi:https://doi.org/10.1016/j.compag.2018.08.039. URL http://www.sciencedirect.com/science/article/pii/S0168169918301650.

Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. Ai benchmark: Running deep neural networks on android smartphones. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

ISPAG. International society of precision agriculture, 2020. URL https://www.ispag.org/.

S. Ivanov, K. Bhargava, and W. Donnelly. Precision farming: Sensor analytics. *IEEE Intelligent Systems*, 30(4):76–80, July 2015. ISSN 1941-1294. doi:10.1109/MIS.2015.67.

Benjamin Jenett, Sam Calisch, Daniel Cellucci, Nick Cramer, Neil Gershenfeld, Sean Swei, and Kenneth C Cheung. Digital morphing wing: active wing shaping concept using composite lattice-based cellular structures. *Soft robotics*, 4(1):33–48, 2017.

Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9000–9008, 2018.

Tor-Aleksander Johansen and Raymond Kristiansen. Quadrotor attitude estimation using adaptive fading multiplicative ekf. In *American Control Conference (ACC), 2017*, pages 1227–1232. IEEE, 2017.

Benjamin R Jordan et al. A bird's-eye view of geology: The use of micro drones/uavs in geologic fieldwork and education. *GSA today*, 25(7):50–52, 2015.

Konstantin Kakaes, Faine Greenwood, Mathew Lippincott, Shannon Dosemagen, Patrick Meier, and Serge Wich. Drones and aerial observation: New technologies for property rights, human rights, and global development: A primer. *New America*, pages 6–103, 2015.

Andreas Kamilaris and Francesc X. Prenafeta-Boldú. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70 – 90, 2018. ISSN 0168-1699. doi:https://doi.org/10.1016/j.compag.2018.02.016. URL http://www.sciencedirect.com/science/article/pii/S0168169917308803.

Fabian Kaup, Philip Gottschling, and David Hausheer. Powerpi: Measuring and modeling the power consumption of the raspberry pi. In *39th Annual IEEE Conference on Local Computer Networks*, pages 236–243. IEEE, 2014.

F. Khaled, O. Ondel, and B. Allard. Optimal energy harvesting from serially connected microbial fuel cells. *IEEE Transactions on Industrial Electronics*, 62(6): 3508–3515, June 2015. ISSN 0278-0046. doi:10.1109/TIE.2014.2371437.

Zohaib Khan, Vahid Rahimi-Eichi, Stephan Haefele, Trevor Garnett, and Stanley J Miklavcic. Estimation of vegetation indices for high-throughput phenotyping of wheat using aerial imaging. *Plant methods*, 14(1):20, 2018.

Jongmin Kim, Youngryel Ryu, Chongya Jiang, and Yorum Hwang. Continuous observation of vegetation canopy dynamics using an integrated low-cost, near-surface remote sensing system. *Agricultural and forest meteorology*, 264:164–177, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Goran Kitić, Aristotelis Tagarakis, Norbert Cselyuszka, Marko Panić, Slobodan Birgermajer, Dušan Sakulski, and Jovan Matović. A new low-cost portable multispectral optical device for precise plant status assessment. *Computers and Electronics in Agriculture*, 162:300–308, 2019.

Ravi Kishore Kodali, Vishal Jain, Suvadeep Bose, and Lakshmi Boppana. Iot based smart security and home automation system. In *2016 international conference on computing, communication and automation (ICCCA)*, pages 1286–1289. IEEE, 2016.

DL Kohlman and WH Wentz Jr. Wind tunnel investigations of vortex breakdown on slender sharp edged wings final report. 1968.

Sridhar Kota, Russell Osborn, Gregory Ervin, Dragan Maric, Peter Flick, and Donald Paul. Mission adaptive compliant wing–design, fabrication and flight test. In *RTO Applied Vehicle Technology Panel (AVT) Symposium*. RTO-MP-AVT-168, Evora, Portugal, 2009.

Victor Kulikov, Victor Yurchenko, and Victor Lempitsky. Instance segmentation by deep coloring. *arXiv preprint arXiv:1807.10007*, 2018.

N. D. Lane, S. Bhattacharya, A. Mathur, P. Georgiev, C. Forlivesi, and F. Kawsar. Squeezing deep learning into mobile and embedded devices. *IEEE Pervasive Computing*, 16(3):82–88, 2017a. ISSN 1536-1268. doi:10.1109/MPRV.2017.2940968.

N. D. Lane, S. Bhattacharya, A. Mathur, P. Georgiev, C. Forlivesi, and F. Kawsar. Squeezing deep learning into mobile and embedded devices. *IEEE Pervasive Computing*, 16(3):82–88, 2017b. ISSN 1536-1268. doi:10.1109/MPRV.2017.2940968.

Nicholas D Lane, Sourav Bhattacharya, Akhil Mathur, Petko Georgiev, Claudio Forlivesi, and Fahim Kawsar. Squeezing deep learning into mobile and embedded devices. *IEEE Pervasive Computing*, (3):82–88, 2017c.

R Larson. Flight control system development and flight test experience with the f-111 mission adaptive wing aircraft. In *Astrodynamics Conference*, page 2237, 1966.

Seyyed Salar Latifi Oskouei, Hossein Golestani, Matin Hashemi, and Soheil Ghiasi. Cnndroid: Gpu-accelerated execution of trained deep convolutional neural networks on android. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1201–1205. ACM, 2016.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Hoonsoo Lee, Tran Quoc Huy, Eunsoo Park, Hyung-Jin Bae, Insuck Baek, Moon S Kim, Changyeun Mo, and Byoung-Kwan Cho. Machine vision technique for rapid measurement of soybean seed vigor. *Journal of Biosystems Engineering*, 42(3): 227–233, 2017.

Lei Li, Qin Zhang, and Danfeng Huang. A review of imaging techniques for plant phenotyping. *Sensors*, 14(11):20078–20111, 2014.

Hyon Lim, Jaemann Park, Daewon Lee, and H Jin Kim. Build your own quadrotor: Open-source projects on unmanned aerial vehicles. *IEEE Robotics & Automation Magazine*, 19(3):33–45, 2012.

Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation. *arXiv:1611.06612 [cs]*, November 2016.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Chih-Hsing Liu, Chen-Hua Chiu, Ta-Lun Chen, Tzu-Yang Pai, Yang Chen, and Mao-Cheng Hsu. A soft robotic gripper module with 3d printed compliant fingers for grasping fruits. In *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 736–741. IEEE, 2018.

Mark W Liu, Mutlu Ozdogan, and Xiaojin Zhu. Crop type classification by simultaneous use of satellite images of different resolutions. *IEEE Transactions on geoscience and remote sensing*, 52(6):3637–3649, 2013.

Peter Liu, Albert Y Chen, Yin-Nan Huang, Jen-Yu Han, Jihn-Sung Lai, Shih-Chung Kang, Tzong-Hann Wu, Ming-Chang Wen, Meng-Han Tsai, et al. A review of rotorcraft unmanned aerial vehicle (uav) developments and applications in civil engineering. *Smart Struct. Syst*, 13(6):1065–1094, 2014.

Luca Lombardo, Simone Corbellini, Marco Parvis, Ahmed Elsayed, Emma Angelini, and Sabrina Grassini. Wireless sensor network for distributed environmental monitoring. *IEEE Transactions on Instrumentation and Measurement*, 67(5):1214–1222, 2018.

Philipp Lottes, Markus Hoeferlin, Slawomir Sander, M Müter, P Schulze, and Lammers C Stachniss. An effective classification system for separating sugar beets and weeds for precision farming applications. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5157–5163. IEEE, 2016.

Philipp Lottes, Raghav Khanna, Johannes Pfeifer, Roland Siegwart, and Cyrill Stachniss. Uav-based crop and weed classification for smart farming. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3024–3031. IEEE, 2017.

Philipp Lottes, Jens Behley, Andres Milioto, and Cyrill Stachniss. Fully convolutional networks with sequential information for robust crop and weed detection in precision farming. *IEEE Robotics and Automation Letters*, 3(4):2870–2877, 2018.

Theodoros Machairas, Alexandros Solomou, and Dimitris Saravanos. A morphing chevron actuated by shape memory alloy wires for noise reduction. In *las actas, proceedings, de 3AF/CEAS Conference Greener Aviation: Clean Sky Breakthroughs and Worldwide Status*, 2014.

Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 3226–3229. IEEE, 2017.

Charalampos Marantos, Nikolaos Karavalakis, Vasileios Leon, Vasileios Tsoutsouras, Kiamal Pekmestzi, and Dimitrios Soudris. Efficient support vector machines implementation on intel/movidius myriad 2. In *2018 7th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pages 1–4. IEEE, 2018.

F Landis Markley. Attitude error representations for kalman filtering. *Journal of guidance, control, and dynamics*, 26(2):311–317, 2003.

Christopher R Marks, Lauren Zientarski, Adam J Culler, Benjamin Hagen, Brian M Smyers, and James J Joo. Variable camber compliant wing-wind tunnel testing. In *23rd AIAA/AHS Adaptive Structures Conference*, page 1051, 2015.

Federico Martinelli, Riccardo Scalenghe, Salvatore Davino, Stefano Panno, Giuseppe Scuderi, Paolo Ruisi, Paolo Villa, Daniela Stroppiana, Mirco Boschetti, Luiz R Goulart, et al. Advanced methods of plant disease detection. a review. *Agronomy for Sustainable Development*, 35(1):1–25, 2015.

Alessandro Matese, Piero Toscano, Salvatore Di Gennaro, Lorenzo Genesio, Francesco Vaccari, Jacopo Primicerio, Claudio Belli, Alessandro Zaldei, Roberto

Bianconi, and Beniamino Gioli. Intercomparison of uav, aircraft and satellite remote sensing platforms for precision viticulture. *Remote Sensing*, 7(3):2971–2990, 2015.

Gerard Rudolph Mendez, Mohd Amri Md Yunus, and Subhas Chandra Mukhopadhyay. A wifi based smart wireless sensor network for monitoring an agricultural environment. In *2012 IEEE International Instrumentation and Measurement Technology Conference Proceedings*, pages 2640–2645. IEEE, 2012.

Andres Milioto, Philipp Lottes, and Cyrill Stachniss. Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in cnns. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2229–2235. IEEE, 2018.

Fausto Milletari, Seyed-Ahmad Ahmadi, Christine Kroll, Annika Plate, Verena Rozanski, Juliana Maiostre, Johannes Levin, Olaf Dietrich, Birgit Ertl-Wagner, Kai Bötzel, et al. Hough-cnn: deep learning for segmentation of deep brain regions in mri and ultrasound. *Computer Vision and Image Understanding*, 164:92–102, 2017.

Ivan Minakov and Roberto Passerone. Pases: An energy-aware design space exploration framework for wireless sensor networks. *Journal of Systems Architecture*, 59(8):626 – 642, 2013. ISSN 1383-7621. doi:https://doi.org/10.1016/j.sysarc.2013.05.020. URL http://www.sciencedirect.com/science/article/pii/S138376211300101X.

Massimo Minervini, Andreas Fischbach, Hanno Scharr, and Sotirios A Tsaftaris. Finely-grained annotated datasets for image-based plant phenotyping. *Pattern recognition letters*, 81:80–89, 2016.

Orazio Mirabella and Michele Brischetto. A hybrid wired/wireless networking infrastructure for greenhouse management. *IEEE Transactions on Instrumentation and Measurement*, 60(2):398–407, 2011.

Syed Misbahuddin, Junaid Ahmed Zubairi, Abdulrahman Saggaf, Jihad Basuni, A Sulaiman, Ahmed Al-Sofi, et al. Iot based dynamic road traffic management for smart cities. In *2015 12th International Conference on High-capacity Optical Networks and Enabling/Emerging Technologies (HONET)*, pages 1–5. IEEE, 2015.

Mohammad Motamedi, Daniel Fong, and Soheil Ghiasi. Cappuccino: efficient cnn inference software synthesis for mobile system-on-chips. *IEEE Embedded Systems Letters*, 11(1):9–12, 2019.

Jirapond Muangprathub, Nathaphon Boonnam, Siriwan Kajornkasirat, Narongsak Lekbangpong, Apirat Wanichsombat, and Pichetwut Nillaor. Iot and agriculture data analysis for smart farm. *Computers and Electronics in Agriculture*, 156:467 – 474, 2019. ISSN 0168-1699. doi:https://doi.org/10.1016/j.compag.2018.12.011. URL http://www.sciencedirect.com/science/article/pii/S0168169918308913.

Faizal Mustapha. innovation in smart materials and structural health monitoring for composite applications. Materials Research Forum LLC, 2017.

Vladimir Nekrasov, Chunhua Shen, and Ian Reid. Light-Weight RefineNet for Real-Time Semantic Segmentation. *arXiv:1810.03272 [cs]*, October 2018.

Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

S. Noh, D. Shim, and M. Jeon. Adaptive sliding-window strategy for vehicle detection in highway environments. *IEEE Transactions on Intelligent Transportation Systems*, 17(2):323–335, Feb 2016. ISSN 1524-9050. doi:10.1109/TITS.2015.2466652.

Maria Chiara Noviello, Rosario Pecora, Francesco Amoroso, Francesco Rea, Maurizio Arena, and Ignazio Dimino. Experimental shape reconstruction of a morphing wing trailing edge in simulated operative conditions. In *2017 8th International Conference on Mechanical and Aerospace Engineering (ICMAE)*, pages 249–256. IEEE, 2017.

Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, Larry Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR 2011*, pages 3153–3160. IEEE, 2011.

Kingnidé R Olympio and Farhan Gandhi. Flexible skins for morphing aircraft using cellular honeycomb cores. *Journal of intelligent material systems and structures*, 21(17):1719–1735, 2010.

D. Oro, C. Fernández, X. Martorell, and J. Hernando. Work-efficient parallel non-maximum suppression for embedded gpu architectures. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1026–1030, March 2016. doi:10.1109/ICASSP.2016.7471831.

Duaa AM Osman and Sondos WA Mohamed. Hardware and software design of onboard computer of israsat1 cubesat. In *2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*, pages 1–4. IEEE, 2017.

R. Pahuja, H. K. Verma, and M. Uddin. A wireless sensor network for greenhouse climate control. *IEEE Pervasive Computing*, 12(2):49–58, April 2013a. ISSN 1536-1268. doi:10.1109/MPRV.2013.26.

Roop Pahuja, HK Verma, and Moin Uddin. A wireless sensor network for greenhouse climate control. *IEEE Pervasive Computing*, (2):49–58, 2013b.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

Alberto Pérez, Pablo Chamoso, Víctor Parra, and Antonio Juan Sánchez. Ground vehicle detection through aerial images taken by a uav. In *17th International Conference on Information Fusion (FUSION)*, pages 1–6. IEEE, 2014.

Artzai Picon, Aitor Alvarez-Gila, Maximiliam Seitz, Amaia Ortiz-Barredo, Jone Echazarra, and Alexander Johannes. Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild. *Computers and Electronics in Agriculture*, 161:280–290, 2019.

S. Pignatti, R. Casa, A. Harfouche, W. Huang, A. Palombo, and S. Pascucci. Maize crop and weeds species detection by using uav vnir hyperpectral data. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 7235–7238, 2019.

Ciro Potena, Daniele Nardi, and Alberto Pretto. Fast and accurate crop and weed identification with summarized train sets for precision agriculture. In *International Conference on Intelligent Autonomous Systems*, pages 105–121. Springer, 2016.

Parisa Pouladzadeh, Shervin Shirmohammadi, and Rana Al-Maghrabi. Measuring calorie and nutrition from food image. *IEEE Transactions on Instrumentation and Measurement*, 63(8):1947–1956, 2014.

Francesco Previtali, Giulio Molinari, Andres F Arrieta, Michel Guillaume, and Paolo Ermanni. Design and experimental characterisation of a morphing wing with enhanced corrugated skin. *Journal of Intelligent Material Systems and Structures*, 27(2):278–292, 2016.

V. Prutyanov, N. Melentev, D. Lopatkin, **A. Menshchikov**, and A. Somov. Developing iot devices empowered by artificial intelligence: Experimental study. In *2019 Global IoT Summit (GIoTS)*, pages 1–6, June 2019. doi:10.1109/GIOTS.2019.8766355.

Long Quan, Ping Tan, Gang Zeng, Lu Yuan, Jingdong Wang, and Sing Bing Kang. Image-based plant modeling. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 599–604. ACM, 2006.

Quan Quan. *Introduction to multicopter design and control*. Springer, 2017.

Karthika Rajendran, Mark Tester, and Stuart J Roy. Quantifying the three main components of salinity tolerance in cereals. *Plant, cell & environment*, 32(3): 237–249, 2009.

Pejman Rasti, Didier Demilly, Landry Benoit, Etienne Belin, Sylvie Ducournau, Francois Chapeau-Blondeau, David Rousseau, and Station Nationale d'Essais de GEVES. Low-cost vision machine for high-throughput automated monitoring of heterotrophic seedling growth on wet paper support. In *BMVC*, page 323, 2018.

Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision*, pages 549–565. Springer, 2016.

Francisco Rodríguez, Manuel Berenguel, José Luis Guzmán, and Armando Ramírez-Arias. *Modeling and control of greenhouse crop growth*. Springer, 2015.

F Rodriguez-Moreno, J Kren, F Zemek, J Novak, V Lukas, and M Pikl. Advantage of multispectral imaging with sub-centimeter resolution in precision agriculture: generalization of training for supervised classification. *Precision Agriculture*, 18 (4):615–634, 2017.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi:10.1007/s11263-015-0816-y.

Inkyu Sa, Zetao Chen, Marija Popović, Raghav Khanna, Frank Liebisch, Juan Nieto, and Roland Siegwart. weednet: Dense semantic weed classification using multispectral images and mav for smart farming. *IEEE Robotics and Automation Letters*, 3(1):588–595, 2017.

Inkyu Sa, Marija Popović, Raghav Khanna, Zetao Chen, Philipp Lottes, Frank Liebisch, Juan Nieto, Cyrill Stachniss, Achim Walter, and Roland Siegwart. Weedmap: a large-scale semantic weed mapping framework using aerial multispectral imaging and deep neural network for precision farming. *Remote Sensing*, 10(9):1423, 2018.

KK Sairajan, GS Aglietti, and KM Mani. A review of multifunctional structure technology for aerospace applications. *Acta astronautica*, 120:30–42, 2016.

Alfonso Sánchez-Macián, Pedro Reviriego, Jesús Tabero, Alberto Regadío, and Juan Antonio Maestro. Sefi protection for nanosat 16-bit chip onboard computer memories. *IEEE Transactions on Device and Materials Reliability*, 17(4): 698–707, 2017.

Fendy Santoso, Matthew A Garratt, and Sreenatha G Anavatti. Fuzzy logic-based self-tuning autopilots for trajectory tracking of a low-cost quadcopter: A comparative study. In *2015 International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA)*, pages 64–69. IEEE, 2015.

Hanno Scharr, Massimo Minervini, Andrew P French, Christian Klukas, David M Kramer, Xiaoming Liu, Imanol Luengo, Jean-Michel Pape, Gerrit Polder, Danijela Vukadinovic, et al. Leaf segmentation in plant phenotyping: a collation study. *Machine vision and applications*, 27(4):585–606, 2016.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

D. Shadrin, **A. Menshchikov**[1], A. Somov, G. Bornemann, J. Hauslage, and M. Fedorov. Enabling precision agriculture through embedded sensing with artificial intelligence. *IEEE Transactions on Instrumentation and Measurement*, pages 1–1, 2019a. ISSN 1557-9662. doi:10.1109/TIM.2019.2947125.

D. Shadrin, **A. Menshchikov**[1], D. Ermilov, and A. Somov. Designing future precision agriculture: Detection of seeds germination using artificial intelligence on a low-power embedded system. *IEEE Sensors Journal*, 19(23):11573–11582, Dec 2019b. ISSN 2379-9153. doi:10.1109/JSEN.2019.2935812.

Dmitrii Shadrin. SeedGermination. https://github.com/DmitriiShadrin/SeedsGermination/, 2018. [Online; accessed 19-May-2019].

Dmitrii Shadrin, Andrey Somov, Tatiana Podladchikova, and Rupert Gerzer. Pervasive agriculture: Measuring and predicting plant growth using statistics and 2d/3d imaging. In *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 1–6. IEEE, 2018a.

Dmitrii Shadrin, Alexander Menshchikov, Andrey Somov, Gerhild Bornemann, Jens Hauslage, and Maxim Fedorov. Tomato Growth Dataset. https://github.com/DmitriiShadrin/TomatoesGrowth, 2019. [Online; accessed 21-January-2019].

Dmitrii G Shadrin, Victor Kulikov, and Maxim Fedorov. Instance segmentation for assessment of plant growth dynamics in artificial soilless conditions. In *BMVC*, page 329, 2018b.

Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. URL https://arxiv.org/abs/1705.05065.

Aamer Shahzad, Fang-Bao Tian, John Young, and Joseph CS Lai. Effects of hawkmoth-like flexibility on the aerodynamic performance of flapping wings with different shapes and aspect ratios. *Physics of Fluids*, 30(9):091902, 2018.

David E Shean, Oleg Alexandrov, Zachary M Moratto, Benjamin E Smith, Ian R Joughin, Claire Porter, and Paul Morin. An automated, open-source pipeline for mass production of digital elevation models (dems) from very-high-resolution commercial stereo satellite imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 116:101–117, 2016.

[1]corresponding author

Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

Mubarakat Shuaibu, Won Suk Lee, John Schueller, Paul Gader, Young Ki Hong, and Sangcheol Kim. Unsupervised hyperspectral band selection for apple marssonina blotch detection. *Computers and Electronics in Agriculture*, 148:45–53, 2018.

Pedro FB Silva, Andre RS Marcal, and Rubim M Almeida da Silva. Evaluation of features for leaf discrimination. In *International Conference Image Analysis and Recognition*, pages 197–204. Springer, 2013.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Nikolai Smolyanskiy, Alexey Kamenev, Jeffrey Smith, and Stan Birchfield. Toward low-flying autonomous mav trail navigation using deep neural networks for environmental awareness. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4241–4247. IEEE, 2017.

Paloma Sodhi, Srinivasan Vijayarangan, and David Wettergreen. In-field segmentation and identification of plant structures using 3d imaging. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5180–5187. IEEE, 2017.

Jurij Sodja, Marcias J Martinez, John C Simpson, and Roeland De Breuker. Experimental evaluation of a morphing leading edge concept. *Journal of Intelligent Material Systems and Structures*, 30(18-19):2953–2969, 2019.

AYN Sofla, SA Meguid, KT Tan, and WK Yeo. Shape morphing of aircraft wing: Status and challenges. *Materials & Design*, 31(3):1284–1292, 2010.

A. Somov, E. F. Karpov, E. Karpova, A. Suchkov, S. Mironov, A. Karelin, A. Baranov, and D. Spirjakin. Compact low power wireless gas sensor node with thermo compensation for ubiquitous deployment. *IEEE Transactions on Industrial Informatics*, 11(6):1660–1670, Dec 2015. ISSN 1551-3203. doi:10.1109/TII.2015.2423155.

A. Somov, A. Karelin, A. Baranov, and S. Mironov. Estimation of a gas mixture explosion risk by measuring the oxidation heat within a catalytic sensor. *IEEE Transactions on Industrial Electronics*, 64(12):9691–9698, Dec 2017. ISSN 0278-0046. doi:10.1109/TIE.2017.2716882.

A. Somov, D. Shadrin, I. Fastovets, A. Nikitin, S. Matveev, I. Oseledets, and O. Hrinchuk. Pervasive agriculture: Iot-enabled greenhouse for plant growth control. *IEEE Pervasive Computing*, 17(04):65–75, oct 2018. ISSN 1558-2590. doi:10.1109/MPRV.2018.2873849.

A. Somov, D. Shadrin, I. Fastovets, A. Nikitin, S. Matveev, I. seledets, and O. Hrinchuk. Pervasive agriculture: Iot-enabled greenhouse for plant growth control. *IEEE Pervasive Computing*, 17(4):65–75, Oct 2018. ISSN 1536-1268. doi:10.1109/MPRV.2018.2873849.

Andrey Somov, Alexander Baranov, Denis Spirjakin, Andrey Spirjakin, Vladimir Sleptsov, and Roberto Passerone. Deployment and evaluation of a wireless sensor network for methane leak detection. *Sensors and Actuators A: Physical*, 202:217 – 225, 2013. ISSN 0924-4247. doi:https://doi.org/10.1016/j.sna.2012.11.047. URL http://www.sciencedirect.com/science/article/pii/S0924424712007297.

David Spivey and Peter Suh. Spanwise adaptive wing. 2018.

Marijn F Stollenga, Wonmin Byeon, Marcus Liwicki, and Juergen Schmidhuber. Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation. In *Advances in neural information processing systems*, pages 2998–3006, 2015.

Xiaohui Su, Zhen Yin, Yuanwei Cao, and Yong Zhao. Numerical investigations on aerodynamic forces of deformable foils in hovering motions. *Physics of Fluids*, 29 (4):041902, 2017.

Oleksandr Sukholeyster. fpga-gpu-benchmarking, 2020. URL https://github.com/softserveinc-rnd/fpga-gpu-benchmarking/blob/master/README-MOVIDIUS.md.

Jian Sun, Qinghua Guan, Yanju Liu, and Jinsong Leng. Morphing aircraft based on smart materials and structures: A state-of-the-art review. *Journal of Intelligent material systems and structures*, 27(17):2289–2312, 2016.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.

Wei Tang, Bo Wang, Bin Bao, and Jing Jun Cao. Experimental comparisons of two detection methods for semi-passive piezoelectric structural damping. *J. Vib. Eng. Technol*, 5(4):367–379, 2017.

K. Taylor, C. Griffith, L. Lefort, R. Gaire, M. Compton, T. Wark, D. Lamb, G. Falzon, and M. Trotter. Farming the web of things. *IEEE Intelligent Systems*, 28 (6):12–19, Nov 2013a. ISSN 1541-1672. doi:10.1109/MIS.2013.102.

Kerry Taylor, Colin Griffith, Laurent Lefort, Raj Gaire, Michael Compton, Tim Wark, David Lamb, Greg Falzon, and Mark Trotter. Farming the web of things. *IEEE Intelligent Systems*, 28(6):12–19, 2013b.

A. Menshchikov. Ru 2018618762; "airflow 2.0: 2d computational fluid dynamics simulator and optimizer of airfoils". 2018a.

A. Menshchikov. Development of adaptive wing with double hinge aileron for unmanned aerial vehicles. *Austrian Journal of Natural and Technical Science*, pages 150–159, June 2018b. doi:10.29013/ajt-18-5.6-33-40.

A. Menshchikov and A. Somov. Mixed reality glasses: Low-power iot system for digital augmentation of video stream in visual recognition applications. In *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*, pages 1–8, June 2018. doi:10.1109/SIES.2018.8442093.

A. Menshchikov and A. Somov. Morphing wing with compliant trailing and leading edges for unmanned aerial vehicles. In *Topology Optimization - Theory, Methods and Applications*, June 2019a.

A. Menshchikov and A. Somov. Morphing wing with compliant aileron and slat for unmanned aerial vehicles. *Physics of Fluids*, 31(3):037105, 2019b. doi:10.1063/1.5086976. URL `https://doi.org/10.1063/1.5086976`.

A. Menshchikov, I. Dranitsky, D. Ermilov, L. Kupchenko, M. Panov, A. Somov, and M. Fedorov. Wilco: Drone instantaneous control by voice and gestures. In *IEEE Industrial Instrumentation and Measurement Technologies Conference*, May 2018a.

A. Menshchikov, D. Shadrin, A. Somov, and M. Fedorov. Artificial intelligence in distributed autonomous real time systems for precision agriculture. In *MoNeTec-2018*, October 2018b.

A. Menshchikov, I. Dranitsky, D. Ermilov, L. Kupchenko, M. Panov, A. Somov, and M. Fedorov. Data-driven body-machine interface for drone intuitive control through voice and gestures. In *Machine Learning Summer School (MLSS-2019)*, September 2019a.

A. Menshchikov, D. Ermilov, I. Dranitsky, L. Kupchenko, M. Panov, M. Fedorov, and A. Somov. Data-driven body-machine interface for drone intuitive control through voice and gestures. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 5602–5609, Oct 2019b. doi:10.1109/IECON.2019.8926635.

A. Menshchikov, D. Lopatkin, E. Tsykunov, D. Tsetserukou, and A. Somov. Realizing body-machine interface for quadrotor control through kalman filters and recurrent neural network. In *IEEE 25th International Conference on Emerging Technologies and Factory Automation (IEEE ETFA 2020)*, September 2020, (in press).

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

EV Tovstik, TA Adamovich, VV Rutman, G Ya Kantor, and T Ya Ashikhmina. Identification of the thickets of heracleum sosnowskyi using earth remote sensing data. , (2):35–37, 2018.

Natsuki Tsushima and Weihua Su. Flutter suppression for highly flexible wings using passive and active piezoelectric effects. *Aerospace Science and Technology*, 65:78–89, 2017.

Adam Van Etten. You only look twice: Rapid multi-scale object detection in satellite imagery. *arXiv preprint arXiv:1805.09512*, 2018.

Srinivas Vasista, Alessandro De Gaspari, Sergio Ricci, Johannes Riemenschneider, Hans Peter Monner, and Bram van de Kamp. Compliant structures-based wing and wingtip morphing devices. *Aircraft Engineering and Aerospace Technology: An International Journal*, 2016.

Francisco Agüera Vega, Fernando Carvajal Ramirez, Monica Perez Saiz, and Francisco Orgaz Rosua. Multi-temporal imaging using an unmanned aerial vehicle for monitoring a sunflower crop. *Biosystems Engineering*, 132:19–27, 2015.

G Anup Venkatesh, P Sumanth, and KR Jansi. Fully autonomous uav. In *2017 International Conference on Technical Advancements in Computers and Communications (ICTACC)*, pages 41–44. IEEE, 2017.

Harry Vereecken, Andrea Schnepf, Jan W Hopmans, Mathieu Javaux, Dani Or, Tiina Roose, Jan Vanderborght, MH Young, Wulf Amelung, Matt Aitkenhead, et al. Modeling soil processes: Review, key challenges, and new perspectives. *Vadose Zone Journal*, 15(5), 2016.

Anh-Vu Vo, Linh Truong-Hong, Debra F Laefer, and Michela Bertolotto. Octree-based region growing for point cloud segmentation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 104:88–100, 2015.

Ben K Wada, James L Fanson, and Edward F Crawley. Adaptive structures. *Journal of Intelligent Material Systems and Structures*, 1(2):157–174, 1990.

Sean R Wakayama and Ed V White. Evaluation of adaptive compliant trailing edge technology. In *33rd AIAA Applied Aerodynamics Conference*, page 3289, 2015.

Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. Ieee, 2000.

Ying Wang, Huawei Li, and Xiaowei Li. Re-architecting the on-chip memory subsystem of machine-learning accelerator for embedded devices. In *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–6. IEEE, 2016.

David C Wilcox et al. *Turbulence modeling for CFD*, volume 2. DCW industries La Canada, CA, 1998.

Piet Christof Wölcken and Michael Papadopoulos. *Smart intelligent aircraft structures (SARISTU): proceedings of the final project conference*. Springer, 2015.

J Wu, C Yuan, Z Ding, M Isakov, Y Mao, T Wang, ML Dunn, and HJ Qi. Multi-shape active composites by 3d printing of digital shape memory polymers sci. *Rep*, 6:24224, 2016.

Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3974–3983, 2018.

Yongzheng Xu, Guizhen Yu, Yunpeng Wang, Xinkai Wu, and Yalong Ma. Car detection from low-altitude uav imagery with the faster r-cnn. *Journal of Advanced Transportation*, 2017, 2017.

Xiaoyuan Yang, Alan H Strahler, Crystal B Schaaf, David LB Jupp, Tian Yao, Feng Zhao, Zhuosen Wang, Darius S Culvenor, Glenn J Newnham, Jenny L Lovell, et al. Three-dimensional forest reconstruction and structural parameter retrievals using a terrestrial full-waveform lidar instrument (echidna®). *Remote sensing of environment*, 135:36–51, 2013.

L. Yao and Z. Ge. Deep learning of semisupervised process data with hierarchical extreme learning machine and soft sensor application. *IEEE Transactions on Industrial Electronics*, 65(2):1490–1498, Feb 2018. ISSN 0278-0046. doi:10.1109/TIE.2017.2733448.

Jiangye Yuan. Learning building extraction in aerial scenes with convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 40(11): 2793–2798, 2017.

Gianluca Zaffiro. A market analysis of technologies towards the disappearing of electronics. In *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*, pages 399–400. IEEE, 2015.

MY Zakaria, Moatassem M Abdallah, and M Adnan Elshafie. Design and production of small tailless unmanned aerial vehicle (sakr 2). In *Proceedings of the 15th AMME Conference*, page 1, 2012.

G. Zhabelova, V. Vyatkin, and V. N. Dubinin. Toward industrially usable agent technology for smart grid automation. *IEEE Transactions on Industrial Electronics*, 62(4):2629–2641, April 2015. ISSN 0278-0046. doi:10.1109/TIE.2014.2371777.

C Zhang and C Rossi. A review of compliant transmission mechanisms for bio-inspired flapping-wing micro air vehicles. *Bioinspiration & biomimetics*, 12(2): 025005, 2017.

Shun-Qi Zhang, Ya-Xi Li, and Rüdiger Schmidt. Active shape and vibration control for piezoelectric bonded composite structures using various geometric nonlinearities. *Composite Structures*, 122:239–249, 2015.

Yun Zhao, Yong He, and Xing Xu. A novel algorithm for damage recognition on pest-infested oilseed rape leaves. *Computers and electronics in agriculture*, 89: 41–50, 2012.

Lianjie Zhou, Nengcheng Chen, Zeqiang Chen, and Chenjie Xing. Roscc: An efficient remote sensing observation-sharing method based on cloud computing for soil

moisture mapping in precision agriculture. *IEEE Journal of selected topics in applied earth observations and remote sensing*, 9(12):5588–5598, 2016.

Hao Jie Zhu and Mao Sun. Unsteady aerodynamic force mechanisms of a hoverfly hovering with a short stroke-amplitude. *Physics of Fluids*, 29(8):081901, 2017.

Appendix A

Additional Resources

A.1 FCNN Architectures

In this section, we present detailed diagrams of the proposed neural network architectures.

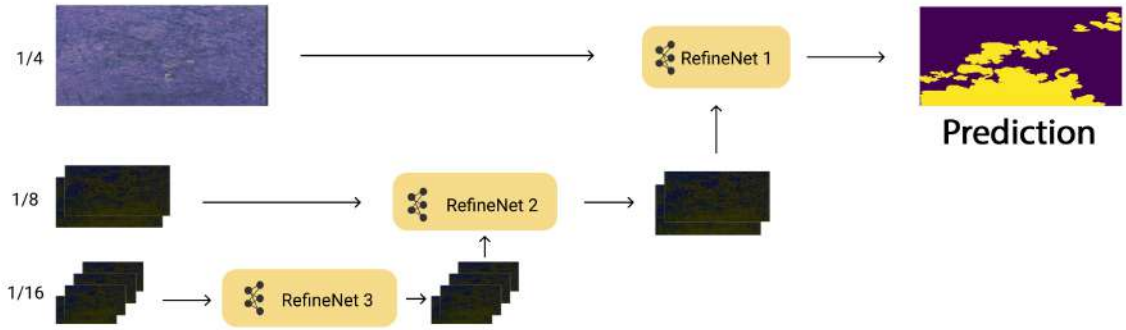


Figure A-1: General architecture of RefineNet, used in the research

A.2 Estimation and Control for eVTOL

The upcoming section describes the mathematics and algorithms behind the estimation and control systems of the eVTOL drone. It was made as a part of the research devoted to the intuitive drone control. The research paper was accepted to the reputed conference by the IEEE society EFTA. However, the part, which describes the NNs, Kalman Filter, and math behind gesture control was not included in this section due to low relevance to the dissertation.

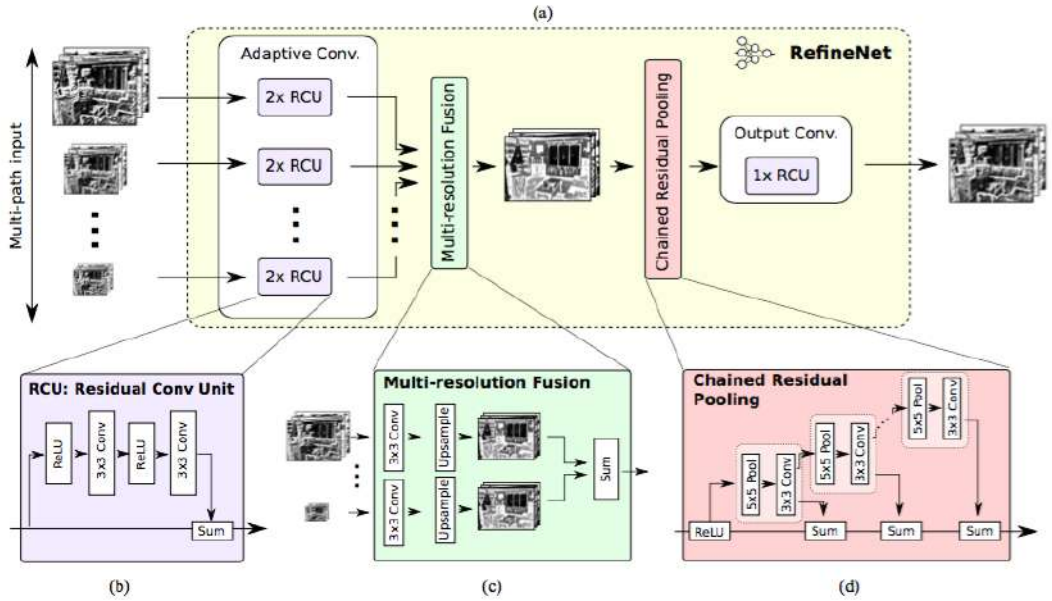


Figure A-2: Refinenet backbone block

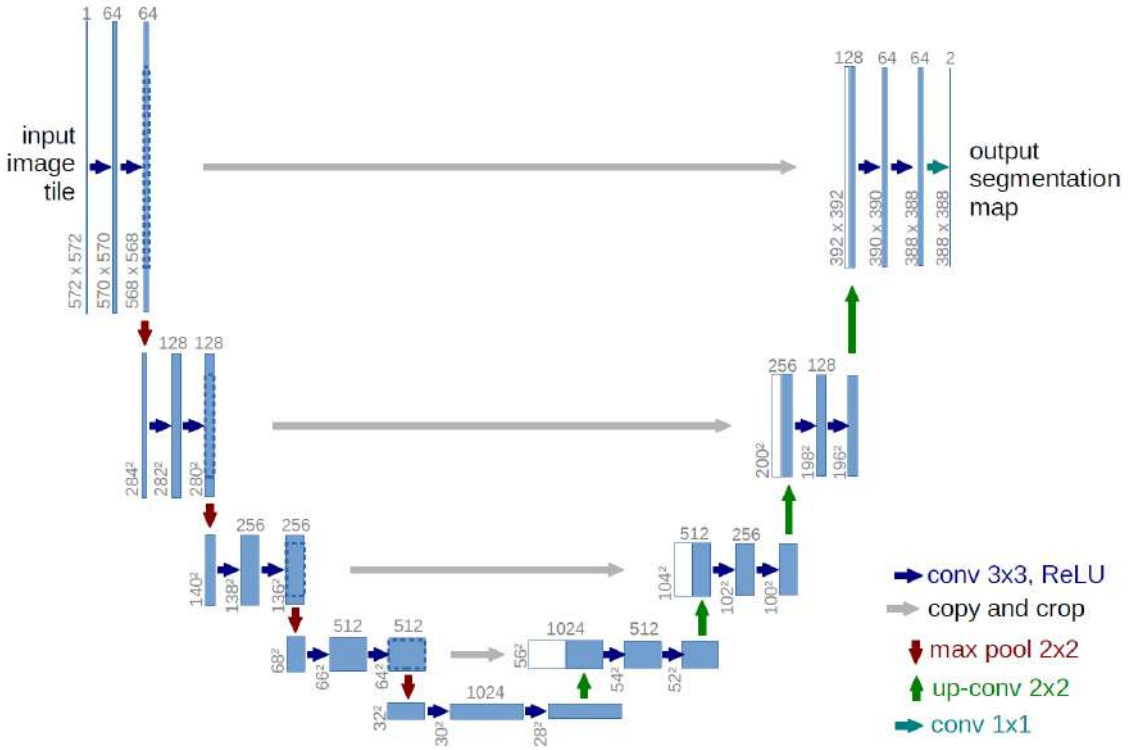


Figure A-3: UNet, used in the research

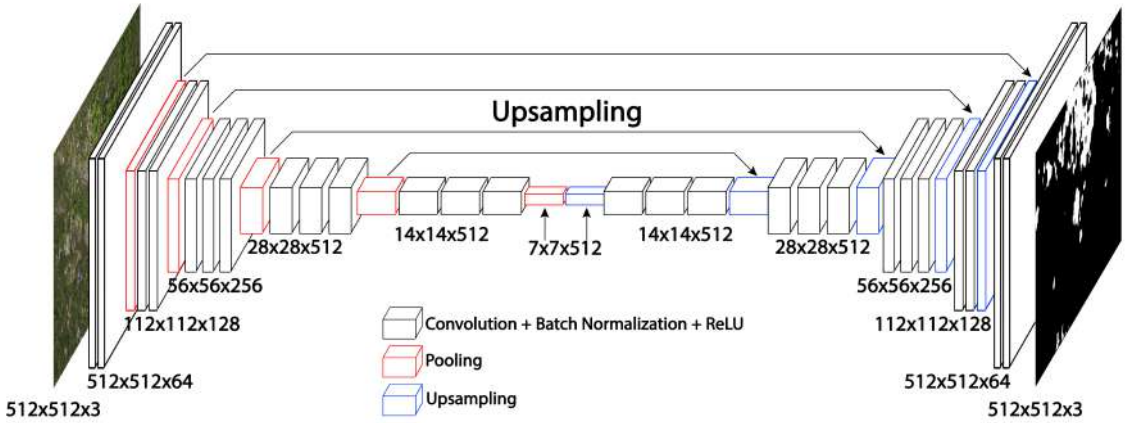


Figure A-4: SegNet, used in the research

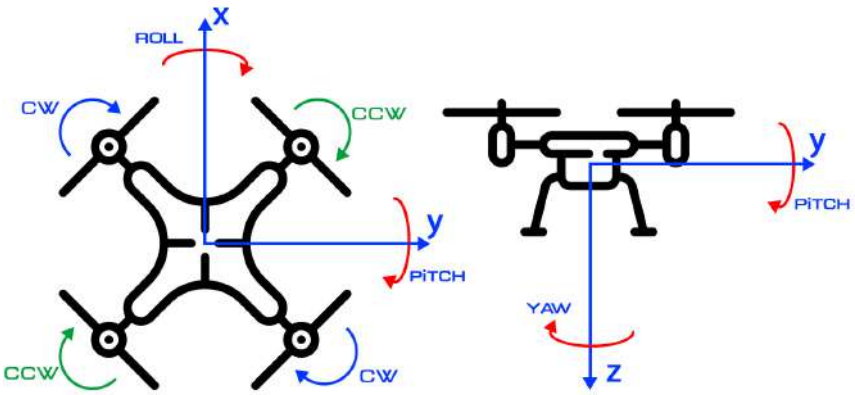


Figure A-5: Body-fixed coordinates (NED) and rotational DOFs of the UAV.

A.2.1 Drone Controller

The results of this section was published in **A. Menshchikov** et al. [2020, (in press].

The drone is a physical system with six Dimensions of Freedom (DOF), which includes three coordinates for the position and three angles for the attitude in the world's frame (roll, pitch, and yaw). In this research, I rely on the North-East-Down (NED) reference frame associated with the drone (see Fig. A-5). However, the state vector of such a system has 12 dimensions:

$$x_t^T = \begin{bmatrix} x & y & z & \phi & \theta & \psi & \dot{x} & \dot{y} & \dot{z} & \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix} \tag{A.1}$$

The control system of the drone includes both controller and estimator implemented using *C++* programming language. The cascaded controller is a system of *PD* and *P* controllers, effectuating the control of the specific parameters in the drone

state vector. The estimator is the Extended Kalman Filter (EKF). It is the non-linear version of Kalman Filter. Its operation relies on the assumption that all the input data has Gaussian distribution, but the estimate of the current mean and covariance need to be linearized. The development and testing of the controller and estimator were performed in the open-source simulator and the experimental setup (see Section A.2.3).

Controller

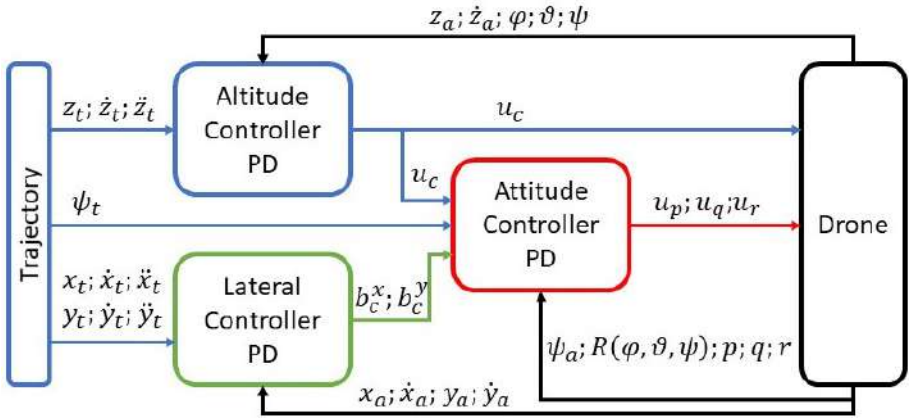
This part is devoted to the implementation of a 3D controller for a quadrotor. It includes the position, altitude, and attitude controllers; each of them is a *PD* or *P* controller. The drone has only 4 control channels assigned one per motor. The thrust and torque generated by the motors depend on the square of the motors rotation rate and are expressed as follows:

$$F_i = k_f \omega_i^2; \quad \tau_i = (-1)^i k_m \omega_i^2 \tag{A.2}$$

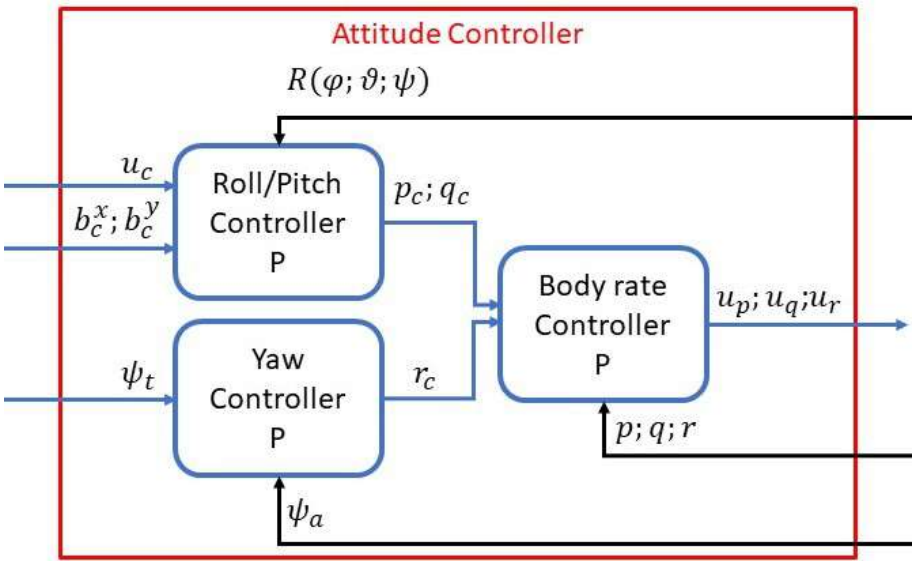
where k_f and k_m are the constants of motors. To control such a system, one needs the controllers for attitude, altitude, and position. Hence, the overall architecture of the controller for the 3D motion of a drone is the following (see Fig. A-6). Here the values with index "a" state for the actual values received from the estimator. The values with index "t" are the target values that are yielded from the trajectory parameters. The values *p, q, r* are the body rates measured by the gyroscope (these values are measured in the body reference frame whereas $\dot{\phi}, \dot{\theta}, \dot{\psi}$ - in the global reference frame). u_c, u_p, u_q, u_r are the control inputs for the total thrust, roll, pitch, and yaw.

Command vector

The motor commands are accurately calculated from the total collective thrust and total torques over the different axes. In this initial step we assume that the following parameters are known: l is the distance from the motor to the x and y axes, parameters of the motors k_f and k_m, moments of inertia I_x, I_y, I_z. To calculate the



(a) The cascaded controller diagram.



(b) Detailed diagram of the attitude controller.

Figure A-6: The controller architecture.

individual commands for each motor from these values, we rely on equation A.3.

$$\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \bar{c} \\ \bar{p} \\ \bar{q} \\ \bar{r} \end{bmatrix} \tag{A.3}$$

$$\bar{c} = \frac{F_\Sigma}{k_f}; \quad \bar{p} = \frac{I_x \bar{u}_p}{l k_f}; \quad \bar{q} = \frac{I_y \bar{u}_q}{l k_f}; \quad \bar{r} = \frac{-I_z \bar{u}_r}{l k_m}; \tag{A.4}$$

Body rate controller

The P controller takes the moments of inertia and body rates of the drone for calculating the commanded moments. It relies on equation A.5.

$$\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} k_p^p(p_t - p_a) \\ k_p^q(q_t - q_a) \\ k_p^r(r_t - r_a) \end{bmatrix} \tag{A.5}$$

where k_p^p, k_q^p, k_r^p are the proportional coefficients for the body rates.

Roll and Pitch controller

The controller uses the acceleration, thrust commands, and the vehicle attitude for output the body rates commands. Also, it accounts for the non-linear transformation from local acceleration to the body rates. We note here that z-axis coincides with the direction of the local $\overrightarrow{g}$. Hence, the lifting force has a negative value (A.6).

$$c = -\frac{F_\Sigma}{m} \tag{A.6}$$

$$\begin{bmatrix} p_c \\ q_c \end{bmatrix} = \frac{1}{R_{33}} \begin{bmatrix} R_{21} & -R_{11} \\ R_{22} & -R_{12} \end{bmatrix} \begin{bmatrix} k_p(\frac{\ddot{x}_c - \ddot{x}_a}{c}) \\ k_p(\frac{\ddot{y}_c - \ddot{y}_a}{c}) \end{bmatrix} \tag{A.7}$$

where indexed "R" are the corresponding rotation matrix elements.

Altitude Controller

The altitude controller is a feed-forward PID controller that takes the data on vertical position, velocity, and acceleration as an input, and returns the required total thrust. The acceleration of the drone along the z-axis is calculated from the linear equation (A.9).

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ c \end{bmatrix} \tag{A.8}$$

$$\ddot{z} = cR_{33} + g \tag{A.9}$$

$$c = \frac{\bar{u} - g}{R_{33}} \tag{A.10}$$

In our case $\bar{u} = \ddot{z}$ and is received from the PID controller equation:

$$\bar{u} = k_p(z_t - z_a) + k_d(\dot{z}_t - \dot{z}_a) + \ddot{z}_{ff} + k_i e \tag{A.11}$$

where $z_t, \dot{z}_t$ are the target altitude and the vertical component of velocity; $z_a, \dot{z}_a$ are the actual altitude and the vertical component of velocity; k_p, k_i, k_d are the PID controller coefficients; e is the integrated altitude error; z_{ff} is the feed-forward component of the controller. At the end of the day the final value of the total thrust is calculated by (A.12).

$$F_\Sigma = -mc \tag{A.12}$$

Lateral Controller

The lateral position PD controller uses the local position and velocity as an input to generate a commanded local acceleration (A.14).

$$\ddot{x} = k_p(x_t - x_a) + k_d(\dot{x}_t - \dot{x}_a) \tag{A.13}$$

$$\ddot{y} = k_p(y_t - y_a) + k_d(\dot{y}_t - \dot{y}_a) \tag{A.14}$$

Yaw Controller

The yaw controller is proportional and is described by (A.15).

$$r_c = k_p(\psi_t - \psi_a) \tag{A.15}$$

A.2.2 Estimator

This section is devoted to the implementation of EKF for a quadrotor. It is a type of Bayes Filter which has two general steps: predict and update. The proposed filter works both with the IMU, Positioning System, and Magnetometer. IMU returns the data from accelerometer and gyroscope, positioning system - the coordinates and velocity data, magnetometer - the data about heading. Altogether these sensors allow for estimating the values for 12 variables for the quadrotor position and attitude. The state vector consists of position, velocity (both received from positioning system), and yaw angle (received from the magnetometer).

$$x_t^T = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \psi \end{bmatrix} \tag{A.16}$$

The vector of control input at time t, u_t is the acceleration in the body frame, where $\dot{\psi}$ is global frame yaw.

$$u_t^T = \begin{bmatrix} \ddot{x}^b & \ddot{y}^b & \ddot{z}^b & \dot{\psi} \end{bmatrix} \tag{A.17}$$

EKF algorithm implemented in the current study performs the update step for IMU, positioning system, and magnetometer separately Cristi and Tummala [2000], Quan [2017]. The Complementary Filter processes the attitude data received from the IMU before the EKF. The EKF algorithm includes the following steps:

Sensor Noise

For the sensor noise estimation, the simulated data have been collected. It includes the data for positioning and accelerometer sensors along x axis. The sensors have different measurement frequencies: 100 Hz for positioning and 200 Hz for accelerom-

Algorithm 1 E(KF) algorithm.

function PREDICT($\mu_{t-1}, \Sigma_{t-1}, u_t, \Delta t$)
  $\bar{\mu}_t = g(u_t, \mu_{t-1})$
  $G_t = g'(u_t, x_t, \Delta t)$
  $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + Q_t$
  return $\bar{\mu}_t, \bar{\Sigma}_t$
function UPDATE($\bar{\mu}_t, \bar{\Sigma}_t, z_t$)
  $H_t = h'(\bar{\mu}_t)$
  $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + R_t)^{-1}$
  $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$
  $\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$
  return μ_t, Σ_t
function EXTENDEDKALMANFILTER
  $u_t =$ COMPUTECONTROL(μ_{t-1}, Σ_{t-1})
  $\bar{\mu}_t, \bar{\Sigma}_t =$ PREDICT($\mu_{t-1}, \Sigma_{t-1}, u_t, \Delta t$)
  $z_t =$ READSENSOR()
  $\mu_t, \Sigma_t =$ UPDATE($\bar{\mu}_t, \bar{\Sigma}_t, z_t$)

eter. The standard deviation values for positioning sensor and accelerometer are the following:

$$\begin{bmatrix} \sigma_x & \sigma_y & \sigma_z \end{bmatrix} = \begin{bmatrix} 0.718 & 0.706 & 2.05 \end{bmatrix} \tag{A.18}$$

$$\begin{bmatrix} \sigma_{\ddot{x}} & \sigma_{\ddot{y}} & \sigma_{\ddot{z}} \end{bmatrix} = \begin{bmatrix} 0.488 & 0.01 & 0.109 \end{bmatrix} \tag{A.19}$$

Attitude Estimation

IMU usually includes a gyroscope, accelerometer, magnetometer, and some optional sensors (barometer, thermometer, etc.). The gyroscope returns the turn rates around body axes (in body frame), but this data is usually noisy and biased. The bias depends on temperature and could drift over with the temperature change. Gyroscope is modeled as follows:

$$\hat{\omega}_t = \omega_t + b + \eta; \quad \eta \sim N(0, \sigma^2_{gyro}) \tag{A.20}$$

The Gaussian noise also varies with time, since the uncertainty grows over time according to the formula:

$$\sigma_n^2 = n\sigma^2\Delta t^2 \tag{A.21}$$

where n is the number of measurements over time. Angle measurements rapidly shift in the range $\pm 3\sigma$ in a matter of seconds. That is why there is always the need in the source of additional information about the attitude. Usually, the accelerometer provides this additional data since it can measure the acceleration's components in the body frame. The relation of these components returns the trigonometric functions of the Euler angles:

$$\hat{\phi}_{accel} = \arctan \frac{a_y}{a_z} \tag{A.22}$$

$$\hat{\theta}_{accel} = \arcsin \frac{a_x}{g} \tag{A.23}$$

However, there is always the issue of how to combine this data with improving the gyroscope measurements. There are many available options which include various EKF Markley [2003] and MEKF (Multiplicative Extended Kalman Filters) Johansen and Kristiansen [2017] Crassidis et al. [2007], Complimentary Filters Quan [2017] Higgins [1975]. Typically, both algorithms have similar computational loads. Kalman filters can reach better accuracy while the complementary filter is usually easier to implement. That is why we used the linear complementary filter Quan [2017] for improving attitude estimation. The idea behind it is the following. The attitude measurement described by the vector z_t, which incorporates data from the accelerometer equation (A.22) and the bodyrates from the gyroscope:

$$z_t^T = \begin{bmatrix} \hat{\phi}_{accel} & \hat{\theta}_{accel} & p & q \end{bmatrix} \tag{A.24}$$

The attitude state vector includes the pitch and roll angles. The yaw angle is not included because it is received from the magnetometer measurements as follows:

$$x_t = \begin{bmatrix} \phi \\ \theta \end{bmatrix} \tag{A.25}$$

Since the gyroscope returns the bodyrates, one needs to perform the transformation from the body to the global reference frame.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{A.26}$$

We perform the Euler Forward Method for the numerical integration to calculate the Euler angles as follows:

$$\begin{bmatrix} \phi_t \\ \theta_t \\ \psi_t \end{bmatrix} = \begin{bmatrix} \phi_{t-1} \\ \theta_{t-1} \\ \psi_{t-1} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} dt \tag{A.27}$$

Then all these values are used in the complementary filter to estimate pitch and roll angles as follows:

$$\hat{\phi}_t = \frac{\tau}{\tau + T_s}\left(\hat{\phi}_{t-1} + T_s\dot{\phi}\right) + \frac{T_s}{\tau + T_s}\hat{\phi}_{accel} \tag{A.28}$$

$$\hat{\theta}_t = \frac{\tau}{\tau + T_s}\left(\hat{\theta}_{t-1} + T_s\dot{\theta}\right) + \frac{T_s}{\tau + T_s}\hat{\theta}_{accel} \tag{A.29}$$

Here $\tau = 0.95$ is the time constant and the T_s is the filter sampling period. This new and better gyro rates integration scheme helped to receive attitude estimator precision < 0.1 rad.

Prediction Step

First, the predicted state vector was calculated out of current state vector and acceleration vector.

$$\begin{bmatrix} x_{upd} \\ y_{upd} \\ z_{upd} \\ \dot{x}_{upd} \\ \dot{y}_{upd} \\ \dot{z}_{upd} \end{bmatrix} = \begin{bmatrix} x_{cur} \\ y_{cur} \\ z_{cur} \\ \dot{x}_{cur} \\ \dot{y}_{cur} \\ \dot{z}_{cur} \end{bmatrix} + \begin{bmatrix} \dot{x}_{cur} \\ \dot{y}_{cur} \\ \dot{z}_{cur} \\ \ddot{x}_{cur} \\ \ddot{y}_{cur} \\ \ddot{z}_{cur} \end{bmatrix} dt + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -g \end{bmatrix} dt \tag{A.30}$$

where x_{upd} is the updated state vector, x_{cur} is the current state vector, $\dot{x}_{cur}$ is the first derivative of the state vector. To access the correct acceleration values it was necessary to convert the acceleration vector from the body to the inertial frame. After that the derivative of the rotation matrix R'_{bg} was calculated:

$$R'_{bg} = \begin{bmatrix} -c_\theta s_\psi & -s_\phi s_\theta s_\psi - c_\phi c_\psi & -c_\phi s_\theta s_\psi + s_\phi c_\psi \\ c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ 0 & 0 & 0 \end{bmatrix} \tag{A.31}$$

Here $c_\theta = \cos\theta; s_\phi = \sin\phi$, etc. Finally, the updated covariance g' was calculated as follows:

$$g'(x_t, u_t, \Delta t) = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & R'_{bg_1}\Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & R'_{bg_2}\Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 & R'_{bg_3}\Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.32}$$

where R'_{bg_i} is the *i-th* row of the matrix multiplication $R'_{bg}u_t$. The covariance matrix is predicted by the following equation.

$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + Q_t \tag{A.33}$$

where G_t equals to $g'(x_t, u_t, \Delta t)$, Σ_{t-1} and $\bar{\Sigma}_t$ are the current and predicted covariance matrices, Q is the uncertainty matrix, associated with the measurements.

Heading Update

To properly update the magnetometer data it is assumed that the readings (yaw) obtained from the magnetometer in the global frame. Also, the observation z_t and the observation function $h(x_t)$ are as follows:

$$z_t = \begin{bmatrix} \psi \end{bmatrix} \tag{A.34}$$

$$h(x_t) = \begin{bmatrix} x_{t,\psi} \end{bmatrix} \tag{A.35}$$

Since the observation function is linear, the derivative is a matrix of zeros and ones.

$$h'(x_t) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.36}$$

The yaw error is less than 0.1 rad in 1 minute of the simulation.

Position Update

Positioning System let us estimate both the position and velocity of the vehicle. One may consider using the heading from the GPS, but it does not take into account the drone's orientation - only the direction of travel. For the orientation, the magnetometer and IMU data were used. Hence it is removed from the observation. The observation vector and the measurement function are as follows:

$$z_t^T = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix} \tag{A.37}$$

Figure A-7: Crazyflie 2.0 quadrotor with four IR reflective markers and 2.4 GHz Crazyradio.

$$h^T(x_t) = \begin{bmatrix} x_{t,x} & x_{t,y} & x_{t,z} & x_{t,\dot{x}} & x_{t,\dot{y}} & x_{t,\dot{z}} \end{bmatrix} \tag{A.38}$$

Then the partial derivative is the identity matrix augmented with a vector of zeros for $\frac{\partial}{\partial x_{t,\phi}} h(x_t)$:

$$h'(x_t) = I_{6x6} \tag{A.39}$$

All the math implemented in the code allowed us to perform the stable flight within the simulated environment.

A.2.3 Results

The development and testing of the control system was performed in the open-source simulator Fot [2018] and in the experimental setup containing the Motion Capture (MoCap) system and Crazyflie drone.

First of all, the system was investigated in the simulator. The proposed controller and estimator were coded and tested in the simulator by Fotokite. This simulator was used in the investigation because the underlying dynamical model is

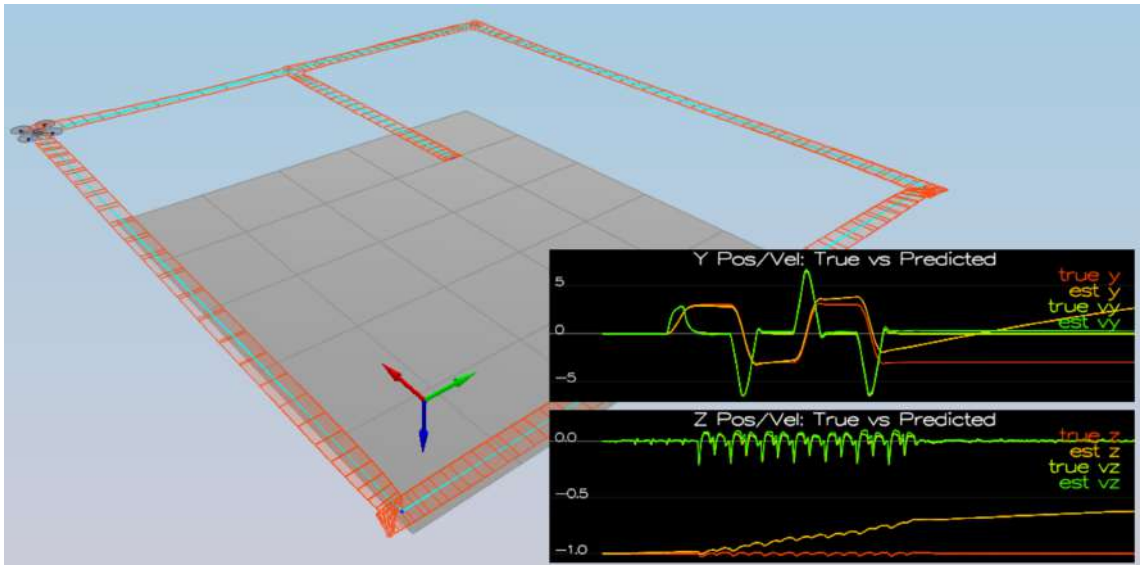


Figure A-8: The simulation of the drone dynamics, following the square trajectory.

very detailed and allows for receiving realistic results. The dynamical parameters of simulated drone, as well as the noisy sensors data, are adjusted to mimic the parameters of the experimental setup (see Fig. A-8).

To track the quadrotor in the real environment a Vicon motion capture system was deployed. It consists of 12 Infrared (IR) cameras (Vantage V5) covering 5 m x 5 m x 5 m space. We used Robot Operating System (ROS) Kinetic framework to run the custom software and ROS stack for Crazyflie 2.0. Four IR reflective markers lead to the total weight of 33 grams reducing the flight time for up to 5 minutes. Crazyflie 2.0 is supplied with two controllers. The first one is the 32 bits ARM Cortex-M4 (STM32F405) for the main applications. The second one is ARM CortexM0 (nRF51822) for power and communication purposes. Crazyflie 2.0 supports Bluetooth, but for the communication, we used the 2.4 GHz Crazyradio showed in Fig. A-7.

Both simulation and the experiments in the real environment resulted in the close coincidence of the trajectories (Fig. A-9). We note that the experimental trajectories contain the take-off and landing parts, which were not included in the evaluation. The standard deviation of experimental trajectories from the simulated trajectories varies in the range of 10 cm. However, the deviation is very low for the "line" trajectory, it is high for the "eight"-like trajectory (Standard deviation of x,

y, z coordinates in the experiment from the simulated data) (A.42).

$$\sigma_{square} = \begin{bmatrix} 0.101 & 0.078 & 0.058 \end{bmatrix} \tag{A.40}$$

$$\sigma_{eight} = \begin{bmatrix} 0.1 & 0.099 & 0.086 \end{bmatrix} \tag{A.41}$$

$$\sigma_{line} = \begin{bmatrix} 0.037 & 0.027 & 0.01 \end{bmatrix} \tag{A.42}$$

A.3 Fixed Wing Control

A.3.1 Notation

$$\text{Position in body frame is } \vec{x}_B = [x_B\ y_B\ z_B]^T \tag{A.43}$$

$$\text{Position in inertial frame is } \vec{x}_I = [x_I\ y_I\ z_I]^T \tag{A.44}$$

$$\text{Attitude is } \Theta = [\phi\ \theta\ \psi]^T = [\text{roll pitch yaw}]^T \tag{A.45}$$

$$\text{Rotation from frame 1 to frame 2 is } H_1^2 \tag{A.46}$$

$$\text{Rotation from inertial to body frame is } H_I^B \tag{A.47}$$

$$\text{Velocity in body frame is } \vec{v}_B = [u\ v\ w]^T \tag{A.48}$$

$$\text{Angular rate in body frame is } \vec{\omega}_B = [p\ q\ r]^T \tag{A.49}$$

$$\text{Total velocity in body frame is } V = \sqrt{u^2 + v^2 + w^2} \tag{A.50}$$

$$\text{Angle of attack in body frame is } \alpha = \tan^{-1}(w/u) \tag{A.51}$$

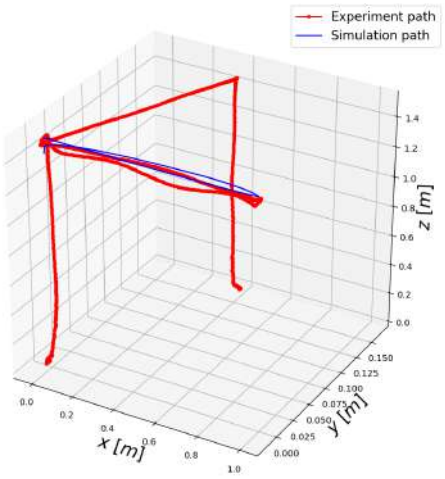
$$\text{Flight path angle is } \gamma = \theta - \alpha \tag{A.52}$$

$$\text{Sideslip angle in body frame is } \beta = \sin^{-1}(v/V) \tag{A.53}$$

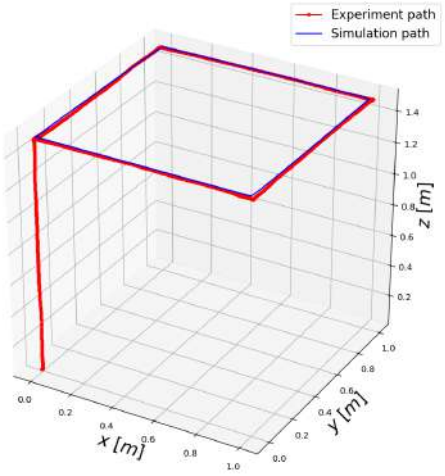
$$\text{Flight path heading is } \xi = \psi + \beta \tag{A.54}$$

$$\text{Bank angle is } \beta \tag{A.55}$$

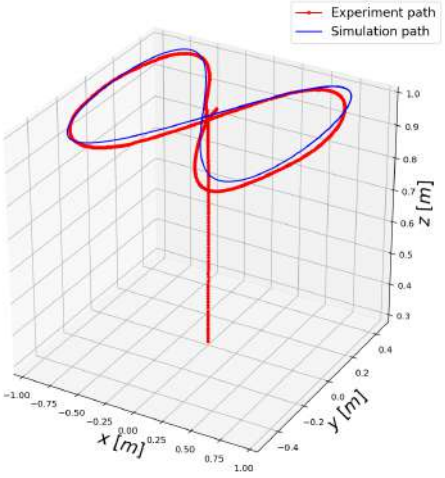
$$\text{Magnitude of the thrust vector is } T \tag{A.56}$$



(a) Collected position data during the flight along the line.



(b) Collected position data during the flight along the square.



(c) Collected position data during the flight along the "eight" figure.

Figure A-9: Flight trajectories during simulations and real-life experiment.

If the thrust of vehicle is aligned with the centerline (x_B axis), then

$$[T_x\ T_y\ T_z]^T = T\cos\alpha, 0, -T\sin\alpha \tag{A.57}$$

$$\text{Aerodynamic forces are } X, Y, Z_B = H_I^B - D, SF, -L \tag{A.58}$$

where L is lift, D is drag, and SF is side force.

$$\tag{A.59}$$

$$\text{Aerodynamic moments are } L, M, N_B = H_I^B L, M, N_I \tag{A.60}$$

Yes, the roll moment and lift force are both denoted by L...

$$\text{Mass of vehicle is } m \tag{A.61}$$

$$\text{Reference area (e.g., wing area) is } S \tag{A.62}$$

$$\text{Wing span is } b \tag{A.63}$$

$$\text{Mean aerodynamic chord } \bar{c} \tag{A.64}$$

$$\text{Elevator deflection is } \delta E \tag{A.65}$$

$$\text{Aileron deflection is } \delta A \tag{A.66}$$

$$\text{Rudder deflection is } \delta R \tag{A.67}$$

$$\text{Dynamic pressure is } \bar{q} = \frac{1}{2}\rho V^2 \tag{A.68}$$

A.3.2 Longitudinal Model:

Longitudinal equations of motion are

$$\dot{x}_I = u\cos\theta + w\sin\theta \tag{A.69}$$

$$\dot{z}_I = -u\sin\theta + w\cos\theta \tag{A.70}$$

$$\dot{\theta} = q \tag{A.71}$$

$$\dot{u} = F_X/m - qw \tag{A.72}$$

$$\dot{w} = F_Z/m + qu \tag{A.73}$$

$$\dot{q} = M_m/I_{yy} \tag{A.74}$$

To get forces and moments (assuming zero wind)

Moment of inertia about the y axis is I_{yy} (A.75)

Aerodynamic forces in stability frame: $\begin{bmatrix} -D \\ -L \end{bmatrix}$ (A.76)

Aerodynamic forces in body frame: $\begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \times \begin{bmatrix} -D \\ -L \end{bmatrix}$ (A.77)

Force in x_B direction $F_X = L\sin\alpha - D\cos\alpha + T - mg\sin\theta$ (A.78)

Force in z_B direction $F_Z = -L\cos\alpha - D\sin\alpha + mg\cos\theta$ (A.79)

We're going to simplify by ignoring the forces and moments from the fuselage and fuselage-wing interference, and compute the forces and moments as:

$$\text{Lift:} \quad L = C_L \bar{q} S \tag{A.80}$$

$$C_L = C_{L_0} + C_{L_\alpha}\alpha + C_{L_{\delta E}}\delta E \tag{A.81}$$

$$\text{Drag:} \quad D = C_D \bar{q} S \tag{A.82}$$

$$C_D = C_{D_0} + \epsilon C_L^2 \tag{A.83}$$

$$= (C_{D_0} + \epsilon C_{L_0}^2) + C_{D_\alpha} + C_{L_\alpha^2}\alpha^2 \tag{A.84}$$

$$\text{with induced drag factor } \epsilon \tag{A.85}$$

$$C_{D_\alpha} = 2\epsilon C_{L_0} C_{L_\alpha} \tag{A.86}$$

$$C_{D_\alpha^2} = \epsilon C_{L_\alpha^2} \tag{A.87}$$

$$\text{Pitch:} \quad M = C_M \bar{q} S \bar{c} \tag{A.88}$$

$$C_M = C_{M_0} + C_{M_\alpha}\alpha + C_{M_\delta E}\delta E \tag{A.89}$$

$$\text{where the elevator deflection is } \delta E \tag{A.90}$$

Quantities needed to implement this model:

Initial conditions:	x_0, y_0, z_{0I}, $\dot{x}_0, \dot{y}_0, \dot{z}_{0I}$, Θ_0, $\vec{\omega}_0$
Vehicle properties:	mass m, wing area S, mean aerodynamic chord $\bar{c}$, inertial matrix I
Coefficients:	C_{L_0}, C_{L_α}, C_{D_0}, ϵ, $C_{L_\alpha^2}$,
Air density:	$\rho(x)$
Gravity:	g
Controls:	thrust T, elevator deflection δE

A.3.3 Linearized Longitudinal Model:

$$
\begin{aligned}
\dot{x}_I &= u\cos\theta + w\sin\theta \\
\dot{z}_I &= -u\sin\theta + w\cos\theta \\
\dot{\theta} &= q \\
\dot{u} &= -g\sin\theta + \frac{\rho V^2 S}{2m}\left[C_{X_0} + C_{X_\alpha}\alpha + C_{X_q}\frac{\bar{c}q}{2V} + C_{X_{\delta_e}\delta_e}\right] + (T+\delta T)/m - qw \\
\dot{w} &= -g\cos\theta + \frac{\rho V^2 S}{2m}\left[C_{Z_0} + C_{Z_\alpha}\alpha + C_{Z_q}\frac{\bar{c}q}{2V} + C_{Z_{\delta_e}\delta_e}\right] + qu \\
\dot{q} &= \frac{\rho V^2 \bar{c} S}{2I_{yy}}\left[C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{\bar{c}q}{2V} + C_{m_{\delta e}}\delta e\right] \\
w &= V\sin\alpha \\
\bar{w} &= V^*\cos\alpha^*\bar{\alpha}
\end{aligned}
$$

$$
\begin{bmatrix} \dot{\bar{x}}_I \\ \dot{\bar{z}}_I \\ \dot{\bar{\theta}} \\ \dot{\bar{u}} \\ \dot{\bar{\alpha}} \\ \dot{\bar{q}} \end{bmatrix} =
\begin{bmatrix}
0 & 0 & -u^*\sin\theta^* + w^*\cos\theta^* & \cos\theta^* & V^*\sin\theta^*\cos\alpha^* & 0 \\
0 & 0 & -u^*\cos\theta^* - w^*\sin\theta^* & -\sin\theta^* & V^*\cos\theta^*\cos\alpha^* & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & -g\cos\theta^* & X_u & X_w V^*\cos\alpha & X_q \\
0 & 0 & \frac{-g\sin\theta^*}{V^*\cos\alpha^*} & \frac{Z_u}{V^*\cos\alpha^*} & Z_w & \frac{Z_q}{V^*\cos\alpha^*} \\
0 & 0 & 0 & M_u & M_w V^*\cos\alpha^* & M_q
\end{bmatrix}
\cdot
\begin{bmatrix} \bar{x_I} \\ \bar{z_I} \\ \bar{\theta} \\ \bar{u} \\ \bar{\alpha} \\ \bar{q} \end{bmatrix} +
$$

$$
+
\begin{bmatrix}
0 & 0 \\
0 & 0 \\
0 & 0 \\
X_{\delta e} & X_{\delta t} \\
\frac{Z_{\delta e}}{V\cos\alpha} & 0 \\
M_{\delta e} & 0
\end{bmatrix}
\cdot
\begin{bmatrix} \bar{\delta} e \\ \bar{\delta} T \end{bmatrix}
$$

Coefficient	Formula
X_u	$\frac{u^*\rho S}{m}\left[C_{X_0}+C_{X_\alpha}\alpha^*+C_{X_{\delta e}}\delta e^*\right]-\frac{\rho S w^* C_{X_\alpha}}{2m}+\frac{\rho S\bar{c}C_{X_q}u^*q^*}{4mV^*}$
X_w	$-q^*+\frac{w^*\rho S}{m}\left[C_{X_0}+C_{X_\alpha}\alpha^*+C_{X_{\delta e}}\delta e^*\right]+\frac{\rho S u^* C_{X_\alpha}}{2m}+\frac{\rho S\bar{c}C_{X_q}w^*q^*}{4mV^*}$
X_q	$-w^*+\frac{\rho V^* S C_{X_q}\bar{c}}{4m}$
$X_{\delta e}$	$\frac{\rho V^{*2} S C_{X_{\delta e}}}{2m}$
$X_{\delta T}$	$\frac{1}{m}$
Z_u	$q^*+\frac{u^*\rho S}{m}\left[C_{Z_0}+C_{Z_\alpha}\alpha^*+C_{Z_{\delta e}}\delta e^*\right]-\frac{\rho S w^* C_{Z_\alpha}}{2m}+\frac{\rho S\bar{c}C_{Z_q}u^*q^*}{4mV^*}$
Z_w	$\frac{w^*\rho S}{m}\left[C_{Z_0}+C_{Z_\alpha}\alpha^*+C_{Z_{\delta e}}\delta e^*\right]+\frac{\rho S u^* C_{Z_\alpha}}{2m}+\frac{\rho S\bar{c}C_{Z_q}w^*q^*}{4mV^*}$
Z_q	$u^*+\frac{\rho V^* S C_{Z_q}\bar{c}}{4m}$
$Z_{\delta e}$	$\frac{\rho V^{*2} S C_{Z_{\delta e}}}{2m}$
M_u	$\frac{u^*\rho S\bar{c}}{I_{yy}}\left[C_{m_0}+C_{m_\alpha}\alpha^*+C_{m_{\delta e}}\delta e^*\right]-\frac{\rho S w^* C_{m_\alpha}}{2I_{yy}}+\frac{\rho S\bar{c}^2 C_{m_q}u^*q^*}{4I_{yy}V^*}$
M_w	$\frac{w^*\rho S\bar{c}}{I_{yy}}\left[C_{m_0}+C_{m_\alpha}\alpha^*+C_{m_{\delta e}}\delta e^*\right]-\frac{\rho S u^* C_{m_\alpha}}{2I_{yy}}+\frac{\rho S\bar{c}^2 C_{m_q}w^*q^*}{4I_{yy}V^*}$
M_q	$\frac{\rho V^* S\bar{c}^2 C_{m_{\delta e}}}{4I_{yy}}$
$M_{\delta e}$	$\frac{\rho V^{*2} S\bar{c} C_{m_{\delta e}}}{4I_{yy}}$

A.3.4 Lateral-Directional motion:

Lateral-directional equations of motion are

$$\dot{x}_I = u(\cos\theta\cos\psi) + v(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) + w(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \tag{A.91}$$

$$\dot{y}_I = u(\cos\theta\sin\psi) + v(\sin\phi\sin\theta\cos\psi + \cos\phi\cos\psi) + w(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \tag{A.92}$$

$$\dot{z}_I = -u\sin\theta + v\sin\phi\cos\theta + w\cos\phi\cos\theta \tag{A.93}$$

$$\dot{\phi} = p + r\cos\phi\tan\theta \tag{A.94}$$

$$\dot{\theta} = -r\sin\phi \tag{A.95}$$

$$\dot{\psi} = r\cos\phi\sec\theta \tag{A.96}$$

$$\dot{u} = -g\sin\theta + \frac{\rho V^2 S}{2m}\left[C_{X_0} + C_{X_\alpha}\alpha\right] + T/m + rv \tag{A.97}$$

$$\dot{v} = g\cos\theta\sin\phi + \frac{\rho V^2 S}{2m}\left[C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{bp}{2V} + C_{Y_r}\frac{br}{2V} + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r\right] + pw - ru \tag{A.98}$$

$$\dot{w} = g\cos\theta\cos\phi + \frac{\rho V^2 S}{2m}\left[C_{Z_0} + C_{Z_\alpha}\alpha\right] - pv \tag{A.99}$$

$$\dot{p} = \left(I_{zz}L + I_{xz}N\right)/\left(I_{xx}I_{zz} - I_{xz}^2\right) \tag{A.100}$$

remembering that L and N are the rolling and yawing moments.

$$\dot{q} = \frac{\rho V^2 \bar{c} S}{2I_{yy}}\left[C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{\bar{c}q}{2V} + C_{m_{\delta e}}\delta e\right] \tag{A.101}$$

$$\dot{r} = \left(I_{xz}L + I_{xx}N\right)/\left(I_{xx}I_{zz} - I_{xz}^2\right) \tag{A.102}$$

And we can write the rolling (not lift) and yawing moments as

$$L = \frac{1}{2}\rho V^2 Sb\left[C_{l_0} + C_{l_\beta}\beta + C_{l_p}\frac{bp}{2V} + C_{l_r}\frac{br}{2V} + C_{l_{\delta a}}\delta a + C_{l_{\delta r}}\delta r\right] \tag{A.103}$$

$$N = \frac{1}{2}\rho V^2 Sb\left[C_{r_0} + C_{r_\beta}\beta + C_{r_p}\frac{bp}{2V} + C_{r_r}\frac{br}{2V} + C_{r_{\delta a}}\delta a + C_{r_{\delta r}}\delta r\right] \tag{A.104}$$

$$\tag{A.105}$$

If we are in level flight, with a small sideslip angle β, then our forces and moments are:

$$Y = C_{Y_\beta}\bar{q}S\beta + C_{Y_{\delta R}}\delta R \tag{A.106}$$

It's weird that the sideforce should be given with respect to the reference area S and dynamic pressure $\bar{q}$. Unpacking the co-efficient, it gets renormalised for the tail, as

$$C_{Y_\beta} = \left(\frac{\bar{q}_{vt}}{\bar{q}}\right)\left(1 + \frac{\partial\sigma}{\partial\beta}\right)\eta_{vt}\left(\frac{S_{vt}}{S}\right)\left(C_{Y_{\beta_{vt}}}\right). \tag{A.107}$$

where η_{vt} is a tail efficiency parameter, σ is the sidewash angle which we can ignore, and the parameters $(\cdot)_{vt}$ are the relevant parameters of the vertical tail. Similarly, the rolling and yawing moments get translated to the tail, for example as

$$C_{N_{\beta_{vt}}} = -C_{Y_{\beta_{vt}}}\eta_{vt}\frac{S_{vt}l_{vt}}{Sb} \tag{A.108}$$

where l_{vt} is the vertical tail length, i.e., the distance from centre of mass to tail centre of pressure. But if we are banked, then the longitudinal forces will have lateral-directional effect too.

$$\tag{A.109}$$

Additional quantities needed to implement this model:

Coefficients:	$C_{Y_0},\ C_{Y_\beta},\ C_{Y_p}$,$C_{Y_r}$,$C_{Y_{\delta a}}$,$C_{Y_{\delta r}},\ C_{l_0},\ C_{l_\beta},\ C_{l_p}$,$C_{l_r}$,$C_{l_{\delta a}}$,$C_{l_{\delta r}}$, $C_{r_0},\ C_{r_\beta},\ C_{r_p}$,$C_{r_r}$,$C_{r_{\delta a}}$,$C_{r_{\delta r}}$
Wing parameters:	b, S
Tail parameters:	S_{vt}, $\bar{q}_{vt}$, η_{vt}, l_{vt}.

A.3.5 Linearized Lateral-Directional Model:

Because we have the trim state and the longitudinal model, for the lateral dynamics case we can basically forget about the x, z and pitch variables in the state. Given a trim state that contains all the relevant variables, we can build and analyze a linear model that only contains the lateral velocity v, the roll and yaw ϕ and ψ, and the corresponding rates p and r.

$$\dot{v} = g\cos\theta\sin\phi + \frac{\rho V^2 S}{2m}\left[C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{bp}{2V} + C_{Y_r}\frac{br}{2V} + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r\right] + pw - ru$$

$$\dot{p} = \left(I_{zz}L + I_{xz}N\right)/\left(I_{xx}I_{zz} - I_{xz}^2\right)$$

$$\dot{r} = \left(I_{xz}L + I_{xx}N\right)/\left(I_{xx}I_{zz} - I_{xz}^2\right)$$

$$\dot{\phi} = p + r\cos\phi\tan\theta$$

$$\dot{\psi} = r\cos\phi\sec\theta$$

But

$$v = V\sin\beta$$

Linearizing around $\beta = \beta^*$:

$$\bar{v} = V^*\cos\beta^*\bar{\beta}$$

$$\dot{\bar{\beta}} = \frac{1}{V^*\cos\beta^*}\dot{\bar{v}}$$

$$\begin{bmatrix} \dot{\bar{\beta}} \\ \dot{\bar{p}} \\ \dot{\bar{r}} \\ \dot{\bar{\phi}} \\ \dot{\bar{\psi}} \end{bmatrix} = \begin{bmatrix} Y_v & \frac{Y_p}{V^* \cos\beta^*} & \frac{Y_r}{V^* \cos\beta^*} & \frac{g \cos\theta^* \cos\phi^*}{V^* \cos\beta^*} & 0 \\ L_v V^* \cos\beta^* & L_p & L_r & 0 & 0 \\ N_v V^* \cos\beta^* & N_p & N_r & 0 & 0 \\ 0 & 1 & \cos\phi^* \tan\theta^* & q^* \cos\phi^* \tan\theta^* & \\ & & & -r^* \sin\phi^* \tan\theta^* & 0 \\ 0 & 0 & \cos\phi^* \sec\theta^* & p^* \cos\phi^* \sec\theta^* & \\ & & & -r^* \sin\phi^* \sec\theta^* & 0 \end{bmatrix} \cdot \begin{bmatrix} \bar{\beta} \\ \bar{p} \\ \bar{r} \\ \bar{\phi} \\ \bar{\psi} \end{bmatrix} +$$

$$+ \begin{bmatrix} \frac{Y_{\delta a}}{V^* \cos\beta^*} & \frac{Y_{\delta r}}{V^* \cos\beta^*} \\ L_{\delta a} & L_{\delta r} \\ N_{\delta a} & N_{\delta r} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \bar{\delta} a \\ \bar{\delta} r \end{bmatrix}$$

Aircraft are often symmetric about the plane spanned by x_b and z_b, which means that

$$I = \begin{bmatrix} I_{xx} & 0 & -I_{xz} \\ 0 & I_{yy} & 0 \\ -I_{xz} & 0 & I_{zz} \end{bmatrix}$$

We can further define the following terms

$$\begin{aligned}
\Gamma &= I_{xx}I_{zz} - I_{xz}^2 \\
\Gamma_1 &= \frac{I_{xz}(I_{xx} - I_{yy} + I_{zz})}{\Gamma} \\
\Gamma_2 &= \frac{I_{zz}(I_{zz} - I_{yy}) + I_{xz}^2}{\Gamma} \\
\Gamma_3 &= \frac{I_{zz}}{\Gamma} \\
\Gamma_4 &= \frac{I_{xz}}{\Gamma} \\
\Gamma_5 &= \frac{I_{zz} - I_{xx}}{I_{yy}} \\
\Gamma_6 &= \frac{I_{xz}}{I_{yy}} \\
\Gamma_7 &= \frac{(I_{xx} - I_{yy})I_{xx} + I_{xz}^2}{\Gamma} \\
\Gamma_8 &= \frac{I_{xx}}{\Gamma}
\end{aligned}$$

We can therefore rewrite the angular rate derivatives as follows:

$$\begin{aligned}
\dot{p} &= \Gamma_1 pq - \Gamma_2 qr \\
\dot{q} &= \Gamma_5 pr - \Gamma_6 (p^2 - r^2) \\
\dot{r} &= \Gamma_7 pq - \Gamma_1 qr
\end{aligned}$$

Coefficient	Formula
Y_v	$\frac{v^*\rho Sb}{4mV^*}\left[C_{Y_p}p^* + C_{Y_r}r^*\right] + \frac{v^*\rho S}{m}\left[C_{Y_0} + C_{Y_\beta}\beta^* + C_{Y_{\delta a}}\delta a^* + C_{Y_{\delta r}}\delta r^*\right] + \frac{\rho SC_{Y_\beta}}{2m}\sqrt{u^{*2}+w^{*2}}$
Y_p	$w^* + \frac{\rho V^* Sb}{4m}C_{Y_p}$
Y_r	$-u^* + \frac{\rho V^* Sb}{4m}C_{Y_r}$
$Y_{\delta a}$	$\frac{\rho V^{*2} S}{2m}C_{Y_{\delta a}}$
$Y_{\delta r}$	$\frac{\rho V^{*2} S}{2m}C_{Y_{\delta r}}$
L_v	$\frac{v^*\rho Sb^2}{4V^*}\left[C_{p_p}p^* + C_{p_r}r^*\right] + v^*\rho Sb\left[C_{p_0} + C_{p_\beta}\beta^* + C_{p_{\delta a}}\delta a^* + C_{p_{\delta r}}\delta r^*\right] + \frac{\rho SbC_{p_\beta}}{2}\sqrt{u^{*2}+w^{*2}}$
L_p	$\Gamma_1 q^* + \frac{\rho V^* Sb^2}{4}C_{p_p}$
L_r	$-\Gamma_2 q^* + \frac{\rho V^* Sb^2}{4}C_{p_r}$
$L_{\delta a}$	$\frac{\rho V^{*2} Sb}{2}C_{p_{\delta a}}$
$L_{\delta r}$	$\frac{\rho V^{*2} Sb}{2}C_{p_{\delta r}}$
N_v	$\frac{v^*\rho Sb^2}{4V^*}\left[C_{r_p}p^* + C_{r_r}r^*\right] + v^*\rho Sb\left[C_{r_0} + C_{r_\beta}\beta^* + C_{r_{\delta a}}\delta a^* + C_{r_{\delta r}}\delta r^*\right] + \frac{\rho SbC_{r_\beta}}{2}\sqrt{u^{*2}+w^{*2}}$
N_p	$\Gamma_7 q^* + \frac{\rho V^* Sb^2}{4}C_{r_p}$
N_r	$-\Gamma_1 q^* + \frac{\rho V^* Sb^2}{4}C_{r_r}$
$N_{\delta a}$	$\frac{\rho V^{*2} Sb}{2}C_{r_{\delta a}}$
$N_{\delta r}$	$\frac{\rho V^{*2} Sb}{2}C_{r_{\delta r}}$

A.3.6 Unified Model without Forces:

$$\dot{x_I} = (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi)v + (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w$$

$$\dot{y_I} = (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi - \cos\phi\cos\psi)v + (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w$$

$$\dot{z_I} = u\sin\theta - v\sin\phi\cos\theta - w\cos\phi\cos\theta$$

$$\dot{u} = rv - qw$$

$$\dot{v} = pw - ru$$

$$\dot{w} = qu - pv$$

$$\dot{\phi} = p + (q\sin\phi + r\cos\phi)\tan\theta$$

$$\dot{\theta} = q\cos\phi - r\sin\phi$$

$$\dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta$$

$$\dot{p} = \left(-[I_{xz}(I_{yy} - I_{xx} - I_{zz})p + [I_{xz}^2 + I_{zz}(I_{zz} - I_{yy})]r]q\right)/(I_{xx}I_{zz} - I_{xz}^2)$$

$$\dot{q} = \frac{-(I_{xx} - I_{zz})pr - I_{xz}(p^2 - r^2)}{I_{22}}$$

$$\dot{r} = \left(-[I_{xz}(I_{yy} - I_{xx} - I_{zz})r + [I_{xz}^2 + I_{xx}(I_{xx} - I_{yy})]p]q\right)/(I_{xx}I_{zz} - I_{xz}^2)$$

A.3.7 Unified Model:

$$\dot{x_I} = (\cos\theta\cos\psi)u + (\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi)v + (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)w$$

$$\dot{y_I} = (\cos\theta\sin\psi)u + (\sin\phi\sin\theta\sin\psi - \cos\phi\cos\psi)v + (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)w$$

$$\dot{z_I} = u\sin\theta - v\sin\phi\cos\theta - w\cos\phi\cos\theta$$

$$\dot{u} = rv - qw - g\sin\theta + \frac{\bar{q}}{m}\left[C_X(\alpha) + C_{X_q}(\alpha)\frac{\bar{c}q}{2V_a} + C_{X_{\delta e}}(\alpha)\delta_e\right] + T$$

$$\dot{v} = pw - ru + g\cos\theta\sin\phi + \frac{\bar{q}}{m}\left[C_{Y_0} + C_{Y_\beta}(\beta) + C_{Y_p}\frac{bp}{2V_a} + C_{Y_r}\frac{br}{2V_a} + C_{Y_{\delta a}}\delta_a + C_{Y_{\delta r}}\delta_r\right]$$

$$\dot{w} = qu - pv - g\cos\theta\cos\phi + \frac{\bar{q}}{m}\left[C_Z(\alpha) + C_{Z_q}(\alpha)\frac{\bar{c}q}{2V_a} + C_{Z_{\delta e}}(\alpha)\delta_e\right]$$

$$\dot{\phi} = p + (q\sin\phi + r\cos\phi)\tan\theta$$

$$\dot{\theta} = q\cos\phi - r\sin\phi$$

$$\dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta$$

$$\dot{p} = \Gamma_1 pq - \Gamma_2 qr + \bar{q}b\left[C_{p_0} + C_{p_\beta}\beta + C_{p_p}\frac{bp}{2V_a} + C_{p_r}\frac{br}{2V_a} + C_{p_{\delta a}}\delta_a + C_{p_{\delta r}}\delta_r\right]$$

$$\dot{q} = \Gamma_5 pr - \Gamma_6(p^2 - r^2) + \bar{q}\frac{\bar{c}}{I_{yy}}\left[C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{\bar{c}q}{2V_a} + C_{m_{\delta e}}\delta_e\right]$$

$$\dot{r} = \Gamma_7 pq - \Gamma_1 qr + \bar{q}b\left[C_{r_0} + C_{r_\beta}\beta + C_{r_p}\frac{bp}{2V_a} + C_{r_r}\frac{br}{2V_a} + C_{r_{\delta a}}\delta_a + C_{r_{\delta r}}\delta_r\right]$$

where

$$
\begin{aligned}
C_{p_0} &= \Gamma_3 C_{l_0} + \Gamma_4 C_{n_0} \\
C_{p_\beta} &= \Gamma_3 C_{l_\beta} + \Gamma_4 C_{n_\beta} \\
C_{p_p} &= \Gamma_3 C_{l_p} + \Gamma_4 C_{n_p} \\
C_{p_r} &= \Gamma_3 C_{l_r} + \Gamma_4 C_{n_r} \\
C_{p_{\delta_a}} &= \Gamma_3 C_{l_{\delta_a}} + \Gamma_4 C_{n_{\delta_a}} \\
C_{p_{\delta_r}} &= \Gamma_3 C_{l_{\delta_r}} + \Gamma_4 C_{n_{\delta_r}} \\
C_{r_0} &= \Gamma_4 C_{l_0} + \Gamma_8 C_{n_0} \\
C_{r_\beta} &= \Gamma_4 C_{l_\beta} + \Gamma_8 C_{n_\beta} \\
C_{r_p} &= \Gamma_4 C_{l_p} + \Gamma_8 C_{n_p} \\
C_{r_r} &= \Gamma_4 C_{l_r} + \Gamma_8 C_{n_r} \\
C_{r_{\delta_a}} &= \Gamma_4 C_{l_{\delta_a}} + \Gamma_8 C_{n_{\delta_a}} \\
C_{r_{\delta_r}} &= \Gamma_4 C_{l_{\delta_r}} + \Gamma_8 C_{n_{\delta_r}}
\end{aligned}
$$

and we push the dependence on the angle of attack into the lift and drag coefficients, so that

$$
\begin{aligned}
C_X(\alpha) &= -C_D(\alpha)\cos\alpha + C_L(\alpha)\sin\alpha \\
C_{X_q}(\alpha) &= -C_{D_q}(\alpha)\cos\alpha + C_{L_q}(\alpha)\sin\alpha \\
C_{X_{\delta_e}}(\alpha) &= -C_{D_{\delta e}}(\alpha)\cos\alpha + C_{L_{\delta_e}}(\alpha)\sin\alpha \\
C_Z(\alpha) &= -C_D(\alpha)\sin\alpha - C_L(\alpha)\cos\alpha \\
C_{Z_q}(\alpha) &= -C_{D_q}(\alpha)\sin\alpha - C_{L_q}(\alpha)\cos\alpha \\
C_{Z_{\delta_e}}(\alpha) &= -C_{D_{\delta e}}(\alpha)\sin\alpha - C_{L_{\delta_e}}(\alpha)\cos\alpha
\end{aligned}
$$