Radio Communication to Control and Run an Autonomous Mission for UAVs

Thiago R. F. Cavalcante
Master Degree in
Electrical Engineering
Federal University of Amazonas
Amazonas, Manaus
Email: thiagorodrigoengcomp@gmail.com

Erickson H. S. Alves
Master Degree in
Electrical Engineering
Federal University of Amazonas
Amazonas, Manaus
Email: erickson.higor@gmail.com

Celso B. Carvalho
Department of Electronics
and Computer
Federal University of Amazonas
Amazonas, Manaus
Email: celsocarvalho75@gmail.com

***Abstract*—This paper presents the development of mission planners in intralogistics for a commercial unmanned aerial vehicle equipped with a robotic gripper in an industrial environment, which consists of an input warehouse, production lines, and a product depot. In this study, the planner produces the needed commands for carrying out a given mission, which includes the delivery of inputs brought from the warehouse to the production line until the final product is delivered to the client (product depot). It was developed two different approaches for mission planning: in the first approach, a simple heuristic is used to solve the mission problem, where a UAV gets the necessary inputs to produce a product, from the warehouse, and bring to the respective production line and the UAV waits in production line site to finish the production of the product; in the second approach, a technique with task scheduling (production process) is employed; both approaches follow a set of production rules. An evaluation of the developed mission planners is performed, verifying the cost of both approaches, measuring the execution time, and comparing those results with the optimum cost obtained with the IBM ILOG CPLEX optimizer.**

I. Introduction

Logistics has become a competitive and fundamental factor for organizations, involving the management, conservation, and supervision of freight transport. In addition, excellent logistics means customer satisfaction; so speed is still an important factor in a successful logistics process [1]. Currently, one of the solutions to this type of problem is the use of unmanned aerial vehicles (UAVs). Nowadays, UAVs are mostly remotely piloted vehicles (RPV), since their operations are carried out by ground operators. If the tasks performed by a UAV are performed autonomously, it would relieve the work of these operators, since they perform tedious and repetitive tasks [2].

One possible improvement of these logistics systems is the increase of the UAVs automation, which results in costs minimization. Consequently, investments and studies related to stand-alone UAVs are important to the smart factories development [3]. However, one of the main problems for using autonomous UAVs is the system's reliability and intelligence. Thus, increased employment of autonomous UAVs requires the development of devices, which are able to perform tasks and interact with the environment in an intelligent and reliable way.

Autonomous UAVs need to know what will happen in a future instant and what is the best decision to make at the present time; therefore, they require strategies not only to decompose their missions into meaningful subtasks, but also to track progress toward mission goals and the evolution of these tasks relative to the autonomous UAVs capabilities [4]. As a consequence, in order to successfully perform a mission, it is recommended to perform task planning [4]. Mission planning problems consist of planning events to meet certain requirements and objectives [5]. Therefore, planning events is one of the main challenges faced in solving this problem.

Both academy and industry have done researches about evaluation and optimization of mission planning in the last years. In [6] is used ant colony to optimize UAV missions. Another paper investigates energy consumption for a factory and evaluates the logistic planning processes using statistical metrics of evaluation [7].

To evaluate mission planning strategies, evaluation metrics must be employed. An evaluation metrics consists of a set of measures that follow a common underlying evaluation methodology. It is used to evaluate the efficacy of information retrieval systems and to justify theoretical and/or pragmatical developments of these systems [8]. In this work, we use an optimal measure to compare with a calculated value of a mission cost.

This paper presents a methodology that evaluates the cost of mission planners for a commercial UAV. We developed an evaluation metrics that evaluates the relative cost of a planning strategy related with the optimal cost generated by the CPLEX optimizer [9]. In summary, the main contributions of this study are:

- a novel evaluation methodology for UAV intralogistics mission planners algorithms, which allows predicting the planners performance and also obtaining optimal algorithms and missions;
- development of an intralogistcs mission planner framework that provides mission commands for a

UAV system;
- use of a commercial UAV system in intralogistics missions to demonstrate the evaluation methodology efficiency.

As a result to this work, we have verified that the development and experiments of the mission planner evaluation methodology were done successfully as sown in Section V. Additionally, the framework to command and control a UAV was gratefully developed and tested in simulated and real environment.

Outline. Section **??** shows previous studies related to mission planning, optimization, and evaluation. Section II provides the fundamentals of mission planning and optimization problems. Section III describes the UAV movement system used in this work. Section IV explains the proposed evaluation methodology in further details. Section V describes the experimental procedures and results in order to explore and demonstrate the potential of methodology, and, finally, Section VI concludes this study and describes future work.

II. Preliminaries

A. Terminology

Key definitions related to the case study and the application developed in this study need to be clarified. All definitions below are adopted in the remainder of this study.

Definition 1: **(Mission Command)** Mission Command is a command created to execute a task such as to go from one location to another, get a package using a robot gripper, and land a UAV.

Definition 2: **(Mission)** Mission is the set of steps and mission commands that the UAV executes to produce the customer's order.

Definition 3: **(Warehouse)** Warehouse is the set of stored raw material available until the moment of entering the productive process. The raw materials, *i.e.*, the inputs available in this work are inputs A, B, and C.

Definition 4: **(Order)** Order is the requisition of products made by the client. In this study, the products are of type X and Y.

Definition 5: **(Production Time)** Production time is the time required to produce a product X or Y, after making available all the needed inputs for the production, given by the production rule.

Definition 6: **(Production Rule)** Production rule describes what and how many inputs are needed to produce a particular product.

Definition 7: **(Mission Planner)** Mission planner is the agent who performs the planning of a mission, that is, produce all steps and commands needed to carry out a given mission.

Definition 8: **(Mission File)** The mission file is a file that is created for the context of this work, with the extension .MISSION containing the mission itself.

Definition 9: **(Movement Function)** The movement function are functions created in Python using the drokekit API to send commands to the UAV by MAVLink protocol.

Definition 10: **(Production Mission)** The production mission is the set of steps to produce all the product required by the client.

B. UAV Iris+

The Iris+, introduced by 3DRobotics in Fall 2014, is a 550mm class (measured motor-to-motor) ready-to-fly quadcopter [10]. It is powered by a 5100mAh battery driving 4 950kV motors through a 4-in-1 electronic speed controller. It includes a Pixhawk flight control board (described below), an internal uBlox GPS with integrated 3-axis compass for navigation, and a 915MHz radio (or 433MHz, depending on country) for communication with a ground control station. Also included is a full-featured remote control transmitter with a preconfigured receiver onboard the drone. The Iris+ has an advertised flight time of 16-22 min. As of January 2016, the Iris+ retails for $600, very reasonable for a vehicle with its capabilities.

Fig. 1: The 3DR Iris+.

Control of the Iris+ is facilitated via the Pixhawk flight control board [11]. Developed by researchers at ETH Zurich, the Pixhawk is open source and very widely adopted. It is suitable for use in copters, planes and ground-based rovers. Connectivity options are numerous, allowing for use of a wide array of external devices such as GPS, range finders and companion computers. Internally, the unit integrates a three-axis accelerometer, a three-axis gyroscope, a three-axis compass and a barometer. These sensors allow for determination of motion, orientation, heading and relative altitude, respectively.

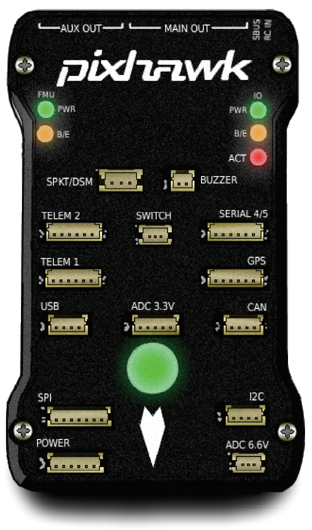

Fig. 2: The Pixhawk flight control board. Numerous ports allow for connection of a multitude of external sensors and devices.

C. Software

In this section we discuss the third-party open-source software that is integral to the vehicle. It is distinct from that we have developed as part of subsequent research projects. Firmware for the Pixhawk comes in the form of two distinct, but cooperating, open source efforts: APM [12] and PX4 [13]. Both flight stacks have robust and supportive development communities and are updated frequently. MAVLink [14], a message-based communication protocol, underlies both efforts. In our vehicles, we use the APM flight stack.

MAVLink provides messages that allow for accessing and changing vehicle parameter values, checking vehicle status and navigation, changing, for example, vehicle position, attitude and velocity. The protocol is extensible, allowing users to define new messages for their purposes. However, the provided message set is sufficiently robust that we have not had the need to do this.

In 2015, 3DR introduced DroneKit-Android, a Android API for developing MAV applications [15]. DroneKit frees researchers and developers from many of the low-level aspects of MAVLink, providing a high-level interface for connecting to, monitoring and controlling a vehicle.

Vehicle configuration and sensor calibration are easily facilitated by ground control station software (GCS). Mission Planner (Windows only) and APM Planner (multi-platform) are both freely available. We use Mission Planner due to its ease of installation.

III. UAV Movement System

The core hardware of the UAV IRIS+ is the Pixhawk and we can control it using a Python library [15], which uses Micro Air Vehicle Link (MAVLink) protocol [11]. MAVLink is a protocol for communicating with small unmanned vehicle, which is designed as a header-only message marshalling library.

The IRIS+ UAV is integrated into a robot gripper to take and leave packages during missions (cf. Definition 2). We have connected a servo motor to the Pixhawk by one of the pulse width modulation (PWM) outputs. Figure 3 shows the system hardware architecture and the interconnections between each component module. In the hardware architecture shown in the Figure 3, we can see the UAV hardware component connections where there is the Pixhawk (flight controller) and its connections between other components such as the compass, GPS, PWM outputs, battery and etc. Moreover, it shows the connection with a robot gripper using a PWM outputs as a signal control for the servo motor in the robot gripper. Finally, it shows the communication between a personal computer (PC) and the UAV via radio control (RC) signal.

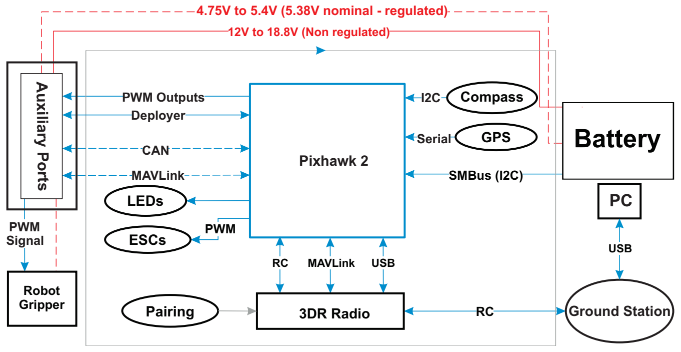

Fig. 3: System Hardware Architecture.

In the software architecture, the Mission Planner (cf. Definition 7) reads the warehouse inputs and client order and produces a `.mission` file, which contains the list of mission commands needed for producing the required client order. This `.mission` file is used by a UAV Control Program to control the UAV and to produce the low-level movement commands using MAVLink protocol (cf. Definition 1). Figure 4 shows the mission planning framework software components.

In order to control the UAV from a PC, we have used the dronekit API that translates MAVLink commands to a Python function. In the ground station, the PC is running the UAV Control Program that controls the UAV using a radio module connected to the PC via USB. We have created a bunch of functions in the control program for the most common UAV actions. The movement functions (cf. Definition 9) are described in Table I.

Command	Description
`TakeOff`	takes off the UAV
`GoTo`	moves the UAV to a certain location
`TakePackage`	takes an input/product (gripper)
`LeavePackage`	leaves an input/product (gripper)
`Wait`	makes the UAV to hover (wait)
`Land`	lands the UAV

TABLE I: Description of movement functions

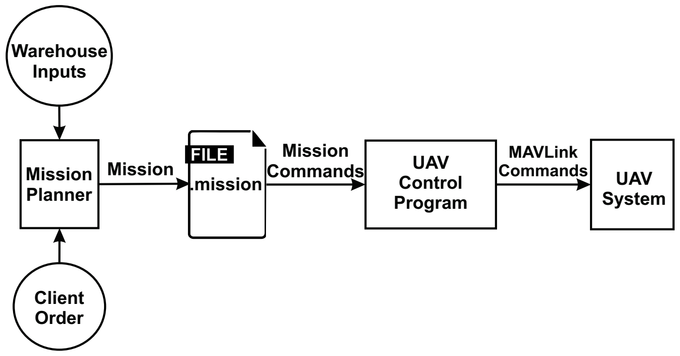

Fig. 4: System's Architecture.

IV. Methodology of Time Cost Evaluation, UAV Use and Mission Planning

A. Case Study: UAV Intralogistics Mission

In order to model the mission planning problem as an optimization problem, the case study shown in Figure 5 is used.

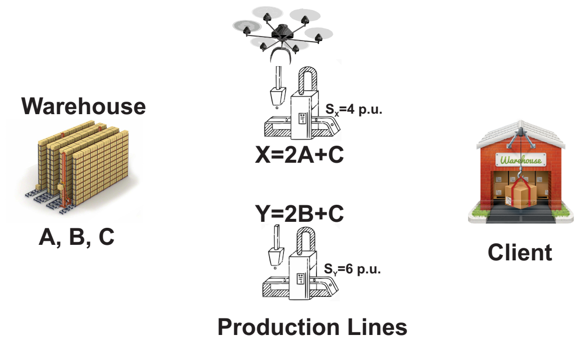

Fig. 5: Case Study Representation.

Figure 5 shows that there are three types of inputs in the warehouse (*i.e.*, A, B, and C) and two production lines that produce two different products (*i.e.*, X and Y). Each production line produces only one type of product and has a characteristic production time (cf. Definition 5). Figure 5 shows that to produce a product of type X, two inputs of type A and one input of type C are required, and to produce a product of type Y, two inputs of type B and one input of type C are required. The production time of a X product is $4p.u.$ and the time of production of product Y is $6p.u.$. A production unit ($1p.u.$) is considered to be a `GoTo` command performed by the UAV.
The task to be performed is the production of the client order (cf. Definition 4), where a given UAV collects supplies from the warehouse, takes that to the production line, and once the production of a certain product is finished, the UAV delivers it to the client.

B. Modeling a UAV Intralogistics Mission as an Optimization Problem

The purpose of this subsection is to make the mission planning problem into an optimization problem, creating a modeling for the problem; in order to find, afterwards, the shortest execution time of all tasks (minimization), based on the case study explained in Section IV-A. The notation used is given below:

- $\mathcal{T} = \{T_j | j \in \mathbb{N}^*, j \leq N\}$ is the set of N tasks;
- $\mathcal{M} = \{m_i | i \in \mathbb{N}^*, i \leq M\}$ is the set of M production lines (machines);
- $\mathcal{P} = \{p_j | j \in \mathbb{N}^*, j \leq N\}$ is the processing time of each j-th task;
- $S = \{s_i | s \in \mathbb{N}^*, i \leq M\}$ is the setup (production) time of each i-th production line;

a) Decision variable: The variable x_{ij} is a binary decision variable that takes the value 1 if the task j is running on the machine i; and 0 otherwise. The variable $C_{mission}$ is the variable that we want to optimize.

$$x_{ij} = \begin{cases} 1, & \text{if the task } j \text{ is running in} \\ & \text{the machine (production line) } i \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

b) Objective function: The objective function is the total mission cost $C_{mission}$ (total process execution time) that can be modeled as follows.

$$C_{mission} = \sum_{i=1}^{M} \sum_{j=1}^{N} (p_j + s_i) x_{ij}, \tag{2}$$

Eq. (2) represents the sum of the duration time p_j of each travel from one place to another in the case study explained in Section IV-A, considering the production time (cf. Definition 5) s_j in each production line.

c) Constraints:

- Each task must be executed/processed in a unique machine:

$$\sum_{i=1}^{M} \sum_{j=1}^{N} x_{ij} = 1 \tag{3}$$

- Execution time of each machine:

$$C_{mission} \leq C_{max} \tag{4}$$

Eq. (4) indicates that the mission cost is always less or equal than a maximum cost denoted by C_{max}, obtained empirically.

d) Resulting optimization problem: The resulting optimization problem consists in minimizing C_{max} w.r.t. the decision variable (1) constrained to the condition in Equations (3) and (4). Thus, the optimization problem is represented as follows:

$$\begin{aligned} \min \quad & C_{mission}, \\ \text{s.t.} \quad & \sum_{i=1}^{M}\sum_{j=1}^{N} x_{ij} = 1, \\ & C_{mission} \leq C_{max} \end{aligned} \tag{5}$$

C. Planner Evaluation Methodology

The main contribution of this study is a methodology to evaluate UAV mission planner algorithms and to find minimum cost planner. For this purpose, a generalized evaluation metrics is developed. The objective (cost) function modeled in Section IV-B is related to the total time spent for the mission execution. Our evaluation metrics compares the cost of a planner algorithm with the best cost computed by the CPLEX solver [9]. This work proposed a novel metrics called Mission Planner Cost Index ($MPCI$). Firstly, the optimal cost of the problem is obtained by means of the CPLEX solver, which returns the optimum value (minimum mission execution time). The model proposed in Section IV-B is implemented using the CPLEX solver library available for C++.

The cost of each planner strategy (Algorithms 1 and 2) c_X is obtained by counting the number of `GoTo` commands which represents a process (task).

Finally, the evaluation of each mission planner is computed w.r.t. the optimal cost, therefore, the $MPCI_X$ of a planner X is computed as follows:

$$MPCI_X = \frac{c_o}{c_X}, \tag{6}$$

where c_o is the optimal cost obtained by a C++ program that converts a given instance to a file in a format known by CPLEX, such as the LP format, *i.e.* the developed algorithm generates the model to be solved by CPLEX with the help of the mathematical programming tool UFFLP. c_X is the cost of the solution generated by planner X, and $0 \leq MPCI_X \leq 1$. Note that as close to 1 the $MPCI_X$ is, the solution cost becomes smaller.

D. Mission Planners

In this study, we considered that mission planner is a software that generates a production mission (cf. Definition 10) given the warehouse and customer order. This program generates a `.mission` extension file containing a set of mission commands (cf. Definition 1), as described in Section III. Two examples of planners are presented in this work and are employed to demonstrate the cost evaluation methodology.

1) Planner 1: In Algorithm 1, we show a strategy to solve the mission planning problem and we denoted it as Planner 1. In this particular algorithm, the production of X products has a higher priority over Y, *i.e.*, the inputs are firstly allocated to production of X orders, and the production of Y products begins if there is no other X to be produced. The general steps of planner 1 are described in Algorithm 1.

Algorithm 1 Planner 1

```
Input: warehouse
Input: order
Output: mission file .mission
begin
    check the order;
    repeat
        go to the warehouse;
        repeat
            get input A;
            bring to the production line X;
        until bring 2 A elements;
        go to the warehouse;
        get the input C;
        bring to the production line X;
        wait X to be produced;
        bring X to the client;
    until production of all X elements finish;
    repeat
        go to the warehouse;
        repeat
            get input B;
            bring to the production line Y;
        until bring 2 B elements;
        go to the warehouse;
        get the input C;
        bring to the production line Y;
        wait Y to be produced;
        bring Y to the client;
    until production of all Y elements finish;
end
```

2) Planner 2: The strategy for Planner 2 is a bit more complex than Planner 1. In the Planner 2 strategy, the UAV starts to bring all the necessary inputs to make the first X product, taking all the A, B and C inputs, respectively, to the production line X. After bring all the necessary inputs to produce the first X product, the production line X starts to produce the X product and while the production line X is producing, the UAV goes to the warehouse to get the necessary inputs to produce the next product (either Planner 1 and Planner 2 produce firstly the X products e then all the Y products). However, when the X product finishes producing, the UAV knows the instant and goes to the production line to get the X product to bring to the client place; and after that, the UAV goes back to bring the rest of the inputs. The UAV keeps work in the same way until it brings all the products to the client place. Differently to the Planner 1, the Planner 2 does not wait the production in the production line. The UAV works as a scheduler and executes the mission faster than Planner 1 strategy, because it does not enter in a busy wait state. The general steps of planner 2 are shown in the Algorithm 2.

Where t_X and t_Y are the production time (cf. Definition 5) of the production lines X and Y, respectively.

V. Experimental Evaluation

This section describes the experimental results obtained in this project, as well as the cost evaluation of two

Algorithm 2 Planner 2

```
Input: warehouse
Input: order
Output: mission file .mission
begin
    initialize $t_x$;
    initialize $t_y$;
    check the order;
    repeat
        if the counter of that X is not $t_X$ then
            go to the warehouse;
            repeat
                get the input A;
                bring to the production line X;
            until until bring 2 A elements;
            go to the warehouse;
            get the input C;
            bring to the production line X;
            start the counter of this X (production time);
            keep producing;
        else
            go back to the production line X;
            bring X to the client;
            go back to producing;
        end
    until production of all X elements finish;
    repeat
        if the counter of that Y is not $t_Y$ then
            go to the warehouse;
            repeat
                get the input B;
                bring to the production line Y;
            until until bring 2 B elements;
            go to the warehouse;
            get the input C;
            bring to the production line Y;
            start the counter of this Y (production time);
            keep producing;
        else
            go back to the production line Y;
            bring Y to the client;
            go back to producing;
        end
    until production of all Y elements finish;
end
```

techniques used, which are compared to the optimum cost implemented with the CPLEX solver.

A. Experimental Environment and Objectives

In order to verify the efficiency of the metrics shown in Section IV, our experimental evaluation aims to answer the following research questions:

RQ1 Does the framework for mission planning, command and control for intralogistics mission using a UAV produce the expected results?

RQ2 Is the metrics of mission evaluation efficient?

The mission planning algorithms were executed in a computer running Linux Mint OS, core i7 processor and 8 GB of RAM. In order to control the UAV, we run the control program, which uses the dronekit API as an interface between a high level program language (Python) and the protocol that the UAV understands (MAVLink), in the same computer where there is a radio module connected via USB communicating with the UAV radio module.

B. Cost Evaluation

The results of each mission planner is compared to the optimal solution obtained with the branch-and-cut algorithm of the IBM/ILOG CPLEX 12.4 tool developed in C++ [9]. In order to obtain better results to perform the comparison, it was considered only the time in which the UAV takes to finish the production of a product.

	Planner 1	**Planner 2**	**CPLEX**
Time (s)	420	404	134

TABLE II: Planners and optimal (CPLEX) times

The Table II shows the mission execution times obtained using the planner algorithms 1 and 2, and the minimum value provided by CPLEX. Using the metrics shown in the section IV, then:

$$MPCI_1 = \frac{134}{420} = 0.319 \tag{7}$$

$$MPCI_2 = \frac{134}{408} = 0.328 \tag{8}$$

The MPCI indicates (cf. Eqs. (1) and (2)) that planner 2 performs the mission more quickly and has a lower cost than planner 1.

C. Practical Results

In order to verify the practical results, as well as a cost comparison between the different approaches of mission planners developed in this work, the flight time measurement was performed using the two mission planning algorithms developed, using the case study shown in IV-A. The experiments were performed in both, simulator and real UAV system. The simulations are performed with DroneKit-SITL [15], that is an resource of Dronekit Pythn API that allows to simulate the behavior and movement of a plane, a copter, or a rover, without the hardware, *i.e.* a real UAV. Additional experiments are performed with the real UAV system (3DR IRIS+). Figure 6 shows the map of experimental environment (Faculty of Physical Education and Physiotherapy of Federal University of Amazonas).

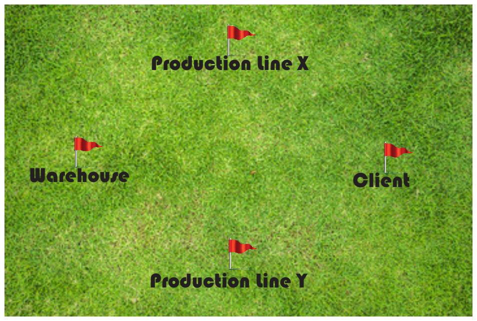

Fig. 6: Warehouse, Production Line X, Production Line Y and Costumer in the Map.

Table III presents the performance (total flight time) of both planners algorithms for simulations and tests with the (real) 3DR Iris+ UAV.

Test #	**Flight Time of Planners**			
	Simulator		**3DR Iris+**	
	1	***2***	***1***	***2***
1	460.41	436.08	455.12	441.72
2	460.69	436.89	456.93	440.18
3	460.08	441.68	457.19	447.51
4	460.72	441.03	460.25	438.19
5	460.23	451.87	459.47	445.85

TABLE III: Mission Planners Flight Times.

Table III indicates that the mission time of planner 2 is lower than the time of planner 1 in all five tests with simulator and 3DR Iris+, ensuring the results of the evaluation methodology.

The aforementioned results confirmed the prediction provided by the planner evaluation methodology, and the planner 2 is faster than planner 1 in all the tests with simulations and IRIS+ tests.

D. Threats to Validity

In order to compile the experiments, we have assembled a favourable environment to apply our evaluation metrics. In this way, we considered the use case in Section IV.

However, in case we change the scenario where the number of UAVs increases, our algorithms will not work as expected because we did not adapted the algorithms for cooperative work and consequently our metrics will not work as expected. Further works may be implemented to make the metrics works in a cooperative work environment where the number of UAVs is greater than two.

VI. Conclusion

We have presented an evaluation methodology for UAV mission planner in an industrial production scenario, as described in Section IV. We have used a evaluation methodology to perform two different algorithms to verify the performance of them. Our approach uses an optimizer tool to generate an optimal cost for a mission (w.r.t. flight time) and compares to different algorithm strategies using the index defined in Eq. 6.

In addition, we have developed a framework for mission planning, command and control for intralogistics mission using a commercial UAV. We used an UAV to solve intralogistics problems using the dronekit API to control and command adopting a high-level programming language.

Summary, our experiments to test the evaluation methodology for mission planner were done successfully as expected in simulated environment as well as in a real environment using a real UAV, and consequently, the methodology is able to evaluate mission planner algorithms for intralogistics problems. Moreover, the framework can solve the necessity of control and command a commercial UAV. Thus, our work is a new tool to verify a specific problem of intralogistics.

Future work includes the use of computational vision for the recognition of inputs, and improvements of the optimization problem modeling for better results in cost evaluation. Supplementary, we will make experiments in a cooperative work environment where the number of UAVs is greater than two. In order to improve our results, we will develop more planner strategies such as an algorithm that produces different types of products simultaneously.

Acknowledgment

The authors would like to thank...

References

[1] Service-drone, "Drones for logistic and transport | multirotor," 2014, accessed in: 12/02/2016. [Online]. Available: \url{http://www.service-drone.com/en/production/logistics-and-transport}

[2] D. Pascarella, S. Venticinque, and R. Aversa, "Autonomic agents for real time UAV mission planning," in *IEEE 10th International Conference on Autonomic and Trusted Computing.* IEEE, 2013, pp. 410–415.

[3] A. Hern, "DHL launches first commercial drone 'parcel-copter'delivery service," *The Guardian*, 2014.

[4] A. Finn and S. Scheding, *Developments and challenges for autonomous unmanned vehicles.* Springer, 2012.

[5] J. A. Krozel, "Search problems in mission planning and navigation of autonomous aircraft," Tech. Rep., 1988.

[6] C. Schwarz and J. Sauer, "Towards decentralised agv control with negotiations." in *STAIRS*, 2012, pp. 270–281.

[7] E. Müller, H. Hopf, and M. Krones, "Analyzing energy consumption for factory and logistics planning processes," in *IFIP International Conference on Advances in Production Management Systems.* Springer, 2012, pp. 49–56.

[8] J. Pehcevski and B. Piwowarski, "Evaluation metrics for structured text retrieval," in *Encyclopedia of Database Systems.* Springer, 2009, pp. 1015–1024.

[9] S. CPLEX, "Ilog," *Inc., Armonk, NY*, 2003.

[10] D. Robotics, "Iris+," 2014, accessed in: 13/06/2017. [Online]. Available: \url{3drobotics.com/iris-plus}

[11] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Pixhawk: A system for autonomous flight using onboard computer vision," in *IEEE international conference on Robotics and automation*, 2011, pp. 2992–2997.

[12] A. Developers, "Ardupilot development site," 2015, accessed in: 13/06/2017. [Online]. Available: \url{dev.ardupilot.com}

[13] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," 2015.

[14] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Mavlink common message set," 2015, accessed in: 13/06/2017. [Online]. Available: \url{pixhawk.ethz.ch/mavlink}

[15] I. 3D Robotics. Dronekit-android. [Online]. Available: android.dronekit.io