Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 55-14 (2022) 139–144

Software-in-the-loop simulation of the forerunner UAV system

Antal Hiba * **Peter Bauer** ** **Mihaly Nagy** ** **Erno Simonyi** ** **Adam Kisari** ** **Gergely Istvan Kuna** *** **Istvan Drotar** ***

* *Research Center of Vehicle Industry, Széchenyi István University, Győr, Hungary and Computational Optical Sensing and Processing Laboratory, Institute for Computer Science and Control (SZTAKI), ELKH, Budapest, Hungary (e-mail: hiba.antal@sztaki.hu).*

** *Research Center of Vehicle Industry, Széchenyi István University, Győr, Hungary and Systems and Control Laboratory, Institute for Computer Science and Control (SZTAKI), ELKH, Budapest, Hungary (e-mail: bauer.peter@sztaki.hu)*

*** *Széchenyi István University, Győr, Hungary*

Abstract: The forerunner UAV means a camera equipped drone flying in front of the advancing first responder units to increase driver situational awareness with an aerial view of the traffic situation and notification about imminent dangers. This article presents the software-in-the-loop (SIL) simulation of the concept including UNREAL4-Carla as the virtual reality environment with a firetruck driven through a game controller, the Matlab simulation of the DJI M600 forerunner hexacopter with UDP communication between firetruck and M600 and the real-time AI processing of synthetic images to detect ground vehicles and pedestrians. The target of SIL development is threefold. First, to test M600 autopilot and AI-based object detection in close to realistic conditions before the real flights. Second, to make an exhaustive feasibility study of the whole forerunner concept with several simulated situations. Third, to generate the required large amount of image data for AI object detection tuning. After introducing all parts of the SIL simulation the article presents an illustrative example evaluating the tracking of the ground vehicle with the M600 and the inference system results.

Keywords: first responders, forerunner UAV, SIL simulation

1. INTRODUCTION

The combination of ground and aerial vehicles can be advantageous as their capabilities can complement each other see e. g. Arbanas et al. [2018]. The concept of the forerunner UAV is to have a drone flying in front of an emergency ground vehicle (EGV) with a downward looking camera and notifying the driver about any approaching danger. This idea was introduced by the authors in Nagy. et al. [2021] and Nagy [2021] as the first responders on their way to the scene can suffer traffic accidents (Donoughe et al. [2012], NHTSA [2014]) due to the lack of situational awareness. An aerial view of traffic in front of the vehicle can show approaching vehicles or other dangers otherwise hidden from the driver of the EGV.

The realisation of this concept requires the drone flying in front of the EGV at least at or beyond braking distance to be able to notify the driver in time. The main challenge is the fact that the EGV driver can alter the route at any time so it is not enough to fly along a predefined route.

Free driving of the EGV, adaptive tracking with the DJI M600 multicopter (DJI [2017]) and the collection and processing of the aerial images are complex tasks the integration of which needs X-in-the-loop testing (Shokry and Hinchey [2009]). Model-in-the-loop (MIL) testing was done for the EGV and hexacopter in Nagy [2021] by simulating EGV motion on a fixed route and testing the tracking control of the hexacopter.

The goal of this article is to describe the next step which is our Software-in-the-loop (SIL) test approach including the free driving of the EGV by a game controller in UNREAL4-Carla environment (see Section 2), the high fidelity simulation of the M600 in Matlab Simulink (see Section 3), the tracking of the EGV with the same communication between EGV and M600 as in the real experiments (see Section 4), the collection of synthetic images in UNREAL4-Carla and their processing on the Jetson Xavier NX NVIDIA [2020] target hardware (see Section 5). The article finally includes an illustrative example about the proper operation in Section 6 and the conclusion (Section 7).

As the flight speed of M600 is limited to 65 km/h besides the realistic testing of the whole system another goal of SIL simulation is to explore the resulting EGV speed limitation required to be able to track it by the M600. A further goal is to collect training data for neural network development and later include a higher speed helicopter model exploring high speed extra-urban situations finally publishing a feasibility study from the overall forerunner concept.

10.1016/j.ifacol.2022.07.596

2. THE UNREAL4-CARLA SIMULATION

The SIL simulation aims for a real-time test and demonstration of the on-board image processing, decision making and M600 EGV tracking control. The environment, sensor data and vehicle states are simulated while the payload computer of the UAV and its software are similar to the real mission setup.
UNREAL4-Carla simulator (Dosovitskiy et al. [2017]) is a rich simulation environment for autonomous ground vehicles. It consists of a high quality 3D city with AI-controlled cars, motorbikes, and pedestrians. It also provides detailed sensor models for RGB cameras (semantic segmentation available), LIDAR (semantic segmentation possible), depth camera, GNSS, IMU. Python API is provided for manipulation of the simulation, and for acquisition of sensor data. The SIL of this paper is the improved realization of the simulator concept presented in Nagy. et al. [2021]. The driver can use a gamer steering wheel to control the EGV, while the scaled down image of the drone camera is presented in the right bottom corner of its view. The Matlab M600 simulation is on a separate computer to also realize UDP wireless communication with which the real drone gets the EGV state. In SIL a wired UDP channel is applied instead of the wireless, but the protocol is the exact same. This separation also decreases the computational burden of the main simulator laptop. We only have the Nvidia Jetson Xavier NX on-board computer from the real hardware setup, which performs real-time detection of 4-wheeled, 2-wheeled vehicles and pedestrians to predict dangerous traffic situations. The Xavier NX also presents its results on a separate screen.

There are two driving modes related to two different tracking concepts of the EGV. In freedrive mode, the drone flies forward to the next (known) intersection, stops and waits until the EGV leaves it and then overtakes the EGV to the following intersection. However, in this case, the EGV many times leaves behind the drone which has not enough speed advantage to reach the next intersection in time. In routedrive mode the drone flies only braking distance ahead of the EGV and as approaching an intersection it 'pulles back' above the EGV to be able to follow any route change. In this concept the drone does not have to stop allowing for a smoother tracking of the EGV and the reduction of stop/start pitching motion possible. This concept is introduced in detail in Section 4. SIL tests also helped us to identify design parameters such as field of view of camera and relative altitude before real flight tests.

Fig. 1 presents the main components of the SIL. The UNREAL4-Carla simulator is manipulated through python modules. The non-player character(NPC) behaviour is responsible for traffic generation and manipulation. In the SIL mode, it is the built-in NPC AI of Carla, however, in the data collection mode, we describe perturbed trajectories for specific traffic situations for an intersection. The second python module is the main component of the SIL that collects data from simulated sensors and integrates the drone simulator into the Carla environment. At each update of the simulation, this main module sends the delta_t time with the EGV pose and velocity to the Matlab Hexacopter model which calculates the dynamics and control of the drone and gives back its state after the update. The drone is moved in the Carla side according to the Matlab simulation. In the next section the M600 Matlab simulator is briefly introduced.

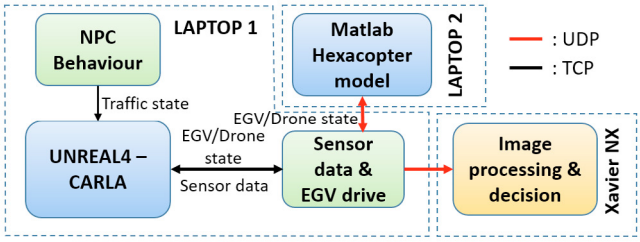

Fig. 1. Software-in-the-loop simulator setup.

3. MATLAB SIMULATION MODEL OF THE DJI M600 HEXACOPTER UAV

The simulation of the DJI M600 hexacopter is implemented in Matlab Simulink because it provides numerous tools for system modeling, an easy-to-use interface to run simulations and option to generate C or C++ code from the models. The overall scheme of the hexacopter model can be seen in Fig. 2.

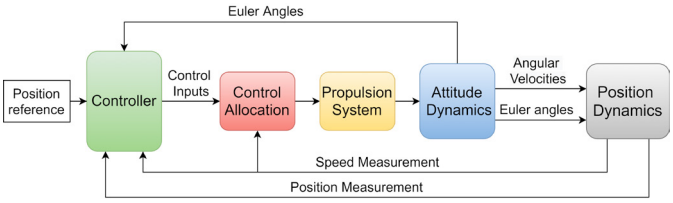

Fig. 2. Block diagram of the hexacopter model

Considering the drone as a rigid body, the common governing equations (Beard and McLain [2012]) can be applied. These are implemented in the Position Dynamics block in Fig. 2 considering the rotor, gravity and aerodynamic translational forces. The rotor and aerodynamic rotational forces are considered in Attitude dynamics block having as main component the torques produced by the rotating blades (Mostafa Moussid [2015]):

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} bl\frac{\sqrt{3}}{2}\left(-\Omega_2^2 - \Omega_3^2 + \Omega_5^2 + \Omega_6^2\right) \\ bl\left(-\Omega_1^2 + \Omega_4^2 + \frac{1}{2}\left(-\Omega_2^2 + \Omega_3^2 + \Omega_5^2 - \Omega_6^2\right)\right) \\ d\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2 - \Omega_5^2 + \Omega_6^2\right) \end{bmatrix}.$$

The Propulsion System block incorporates six BLDC motor models with parameters from the DJI M600 propulsion system manual DJI [2016]. Weighted Control allocation (Guillaume J. J. Ducard [2011]) is a method which defines angular speed references for each motor while taking predefined constraints like vertical speed into account. The Control Allocation block receives control inputs, namely altitude and orientation reference signals provided by the Controller (including low level control tuned to have realistic dynamics (on the real M600 provided by DJI) and high level EGV tracking control introduced in Section 4).

Besides the BLDC motor parameters most of the other parameters like dimensions, mass, torque and thrust coef-

ficients were derived from DJI manual DJI [2018]. Others which could not be found were set according to engineering intuition. However, since the actual M600 Pro arrived we are working on a detailed system identification of the model relying on test flights done at end of 2021 and beginning of 2022.

Regarding the high level control the freedrive concept was introduced in Nagy [2021] (and demonstrated as Project Forerunner [2021]). The Controller can track waypoints while paying attention to the position of its car companion. It utilizes simple PID controllers to achieve reference tracking and a state machine to successfully adapt to the car's speed and navigate between the predefined waypoints. However, facing the limitations of the freedrive concept we developed the routedrive mode introduced in the next section.

4. ROUTEDRIVE EGV TRACKING WITH THE M600 HEXACOPTER

One of the greatest challenges in the forerunner concept is to track the EGV flying in front of it with the possibility of sudden route changes by its driver. Flying in front of the EGV is inevitable if one wants to provide information about the traffic on the route in advance. With a downward looking camera the M600 should be at least at braking distance ahead to provide information in time. This means that in case of sudden route changes (unplanned turn in an intersection) the M600 gets well behind the EGV and so aerial view coverage is not provided until it overtakes the EGV. Our approach to solve this is an adaptive tracking control considering the planned route of the EGV transmitted through wireless UDP communication between EGV and M600 together with current EGV position and speed.

In the routedrive concept the M600 flies in front of or at least above the EGV considering its actual braking distance (speed dependent) and the proximity of the intersection. Knowing the position of the next intersection the M600 is 'pulled back' to be above the EGV before it reaches the intersection and so to be able to follow any unplanned turn.

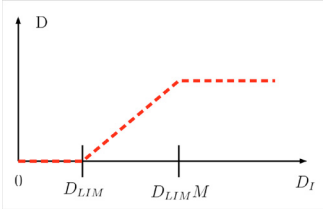

Fig. 3. Adaptive distance from EGV (D)

The modeling of ground vehicle braking distance was done in detail in e. g. Bauer et al. [2019] leading to the formula (D offset distance [m], V vehicle speed [m/s]):

$$D = 0.06346V^2 + 0.4727V$$

Far from the intersection the M600 tries to keep this distance in front of the EGV in the actual moving direction of the EGV. As the M600 approaches the intersection its distance from the intersection D_I is continuously evaluated and below a given limit $D_{LIM}M$ the reference distance is gradually decreased to zero. Below D_{LIM} the M600 should stay above the EGV. The whole scaling can be seen in Fig. 3, the current parameter values for the simulated urban scenario are: $D_{LIM} = 30m$ and $M = 3$.

The DJI M600 can be controlled through forward and side velocity, yaw angle and altitude references, that's why our simulation also makes control through these variables. The altitude reference is fixed, the yaw angle is aligned with the current moving direction of the EGV and the forward V_x and side V_y velocity references are generated considering the speed of the EGV, the offset distance (aligned with EGV moving direction) and position errors also:

$$V_{x_{ref}} = CV_x + 0.5(CP_x + D_x - HC_x)$$
$$V_{y_{ref}} = CV_y + 0.5(CP_y + D_y - HC_y)$$

Here CV_x, CV_y is EGV velocity, CP_x, CP_y is EGV position and HC_x, HC_y is M600 position vector all in M600 body coordinate system (see Nagy [2021]). D_x, D_y are the M600 body coordinates of the offset distance and 0.5 is a tuned constant.

Besides the UNREAL4-Carla environment and the M600 simulation and control a crucial part is the processing of the generated synthetic aerial images on the target Jetson Xavier NX hardware to detect any dangerous traffic situation in front of the EGV. Related results are summarized in the next section.

5. OBJECT DETECTION AND HARDWARE EXPERIENCES

Object detection was done using trained Yolo neural networks. Two separate network architectures were trained with birds-eye view images, one with simulated data from the UNREAL4-Carla simulator trained for 3 classes (vehicles, bicycles, pedestrians) and a second network trained only for vehicles with real-life data obtained from publicly available databases as a preparation for real-life experiments. Inference was done on a Jetson Xavier NX embedded device.

The database for the simulated network consisted of 1000 frames labeled by Carla, and another 500 was generated with data augmentation by flipping the input images. Real-life birds-eye view images were obtained from the KIT AIS (Karlsruher Institute fũr Technologie [2012]), PUCPR (Almeida et al. [2015]) and CARPK datasets, amounting to roughly 2000 frames. During both trainings a train/test split distribution of 70/30% was used.

A Yolov3 and a Yolov4 neural networks (see Redmon and Farhadi [2018] and Bochkovskiy et al. [2020]) were trained for the Carla dataset with Yolov4 showing better inference results with only a marginal decrease in inference fps compared to the v3 version. A Yolov4-tiny network was also tested and although inference speed was much higher reaching an average of 35 fps the accuracy, especially on the real-life images was very low (below 40.0% mAP@0.5). Table 1 shows the training results of the two networks including the average precision (AP) of each class, the overall mean average precision (mAP) and the Precision,

Recall and F1 scores of the respective networks. Inference results can be seen in Fig. 4 and Fig. 5 class labels 0,1,2 represent vehicles, bicycles and pedestrians respectively. Considering the real images Yolov4 showed comparable training results to it's simulated counterpart. Evaluation of a Yolov5 network and fine-tuning of the trained networks with further training data especially for real-life images is in progress.

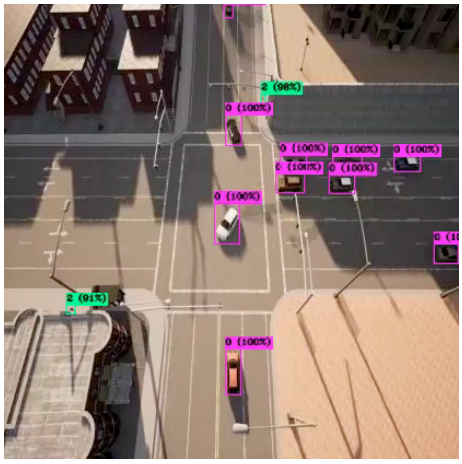

Fig. 4. Simulation inference results

Fig. 5. Simulation inference results 2

Table 1. Training results

Neural network	Yolov3	Yolov4
Vehicle AP	97.84%	98.29%
Bycicle AP	86.38%	93.09%
Pedestrian AP	59.75%	71.51%
Overall mAP@0.50	81.32%	87.63%
Precision	0.9	0.89
Recall	0.9	0.93
F1 score	0.9	0.91
Inference avg fps	15	12

Besides detection of the objects their tracking is also important to be able to make a decision if they decelerate to give way to the EGV or not. As a possibility to implement additional tracking of objects for the previously trained Yolov4 network, experiments were made with Yolov4 Deep SORT (Wojke et al. [2017]), an object tracking module for Yolov4 neural networks. Early results showed that although for the majority of the frames object tracking was accurate there were occasions where Deep SORT failed to detect the originally identified objects. It also drastically reduced inference speed to an average of 6 fps. Fig. 6 depicts the tracking results of Deep SORT where labels include the class number and the unique ID of the tracked object. The future developments will target to have a less computationally demanding tracker possibly integrated with vehicle speed and acceleration estimation.

Fig. 6. Yolov4 Deep SORT inference results

Fig. 7. The Forerunner SIL simulation at a demonstration event.

6. ILLUSTRATIVE EXAMPLE

The whole SIL test setup can be seen in Fig. 7 showing the Matlab simulation of M600 (developed in Nagy [2021]) on a laptop, the UNREAL4-Carla virtual reality simulation with the game controller and our colleague driving the firetruck and the Jetson Xavier NX board with the processed aerial images on the screen. A short presentation of the Forerunner project and a video of the working SIL setup can be seen on Youtube Project Forerunner [2021].

Fig. 8 shows the tracks of the human driven firetruck and the waypoints placed at the intersections and one on the curved road. Fig. 9 shows the tracking of the second track with the routedrive method introduced in Section 4. Note that the background images were only approximately placed below the track in both figures.

Fig. 9 shows that along the straight streets the tracking is pretty good, only the turns at the intersections (and the waypoint on the curve) cause larger deviations from the

track. The source of these deviations is that the tracking of the velocity references by the M600 is slower than required as shown in Fig. 10 despite the increase of the maximum allowed pitch (and roll) angle to 40° from the DJI limit 25° in the simulation.

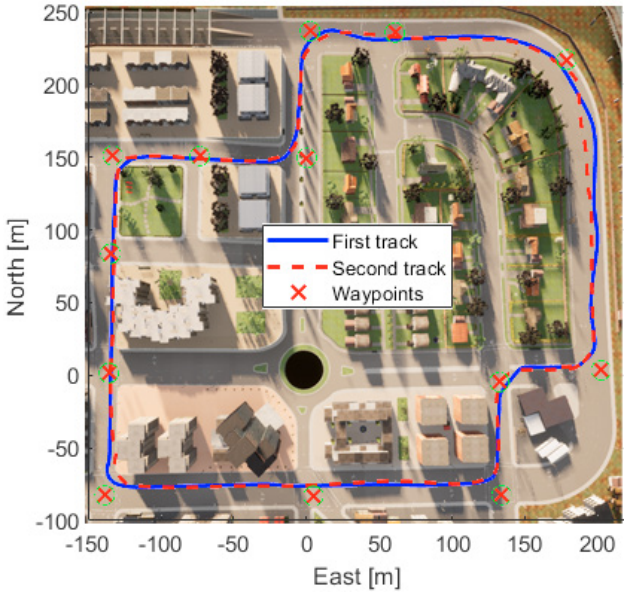

Fig. 8. Two firetruck trajectories with waypoints in UNREAL4-Carla

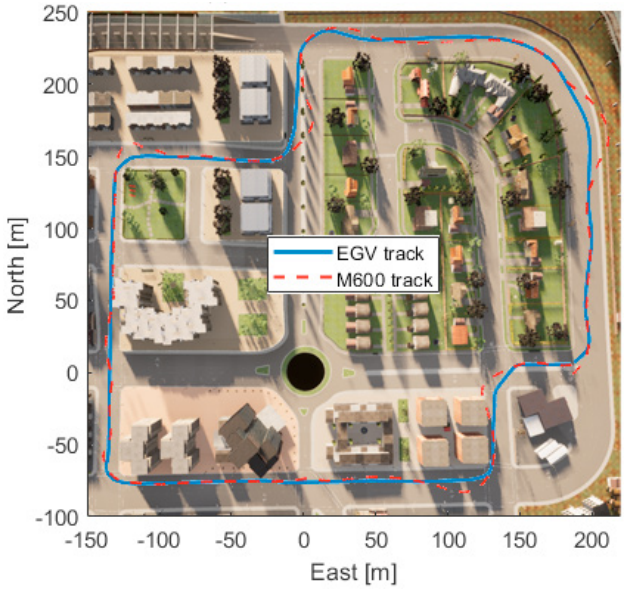

Fig. 9. Tracking of EGV trajectory

The figure also shows that starting from zero velocity the M600 can reach the EGV and then track it. Another important aspect of this figure is that the reference speeds are often above 65 km/h which is the maximum speed of the M600 (see DJI [2017]). Note that at the same time the car rarely exceeds 50 km/h which means that significant speed advantage of the forerunner UAV is required (of course this also depends on control tuning). These facts clearly show that speed limitations of multicopters can be a major limitation in the forerunner concept. However, there exist unmanned drones e. g. Schiebel Corporation [2022] with airspeeds well above 200km/h so potentially even highway forerunner applications can be covered. Our goal with the DJI M600 is only a moderate cost proof of concept of the forerunner UAV in low speed (possibly 20-30 km/h) urban situations.

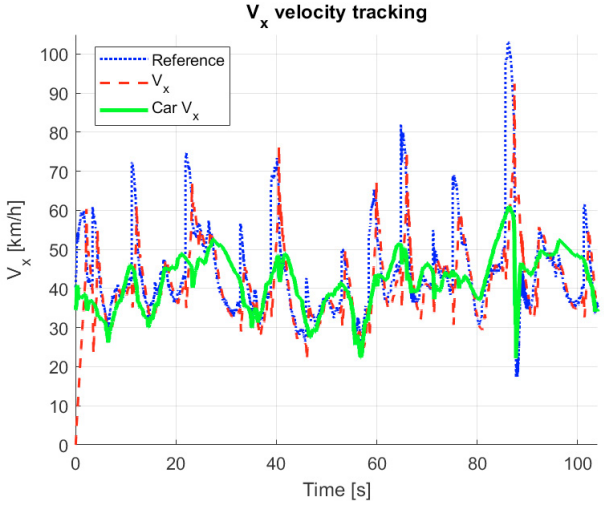

Fig. 10. Tracking of X body velocity with M600 together with car velocity

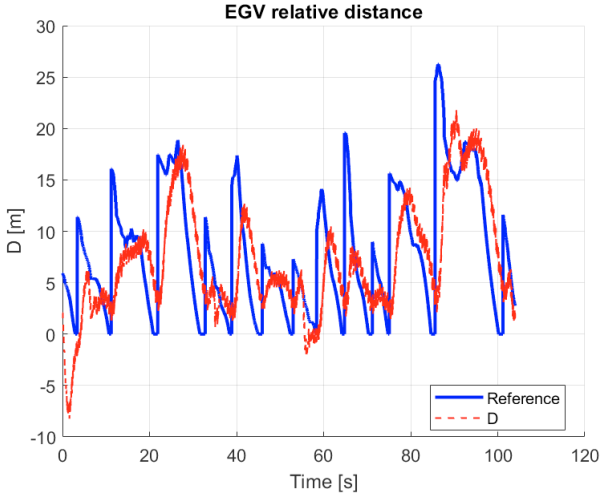

Fig. 11. Reference and real distance from EGV

Finally Fig. 11 shows the reference (adaptive distance from EGV is shown in Fig. 3) and real distance between the EGV and M600 along the driving direction of the EGV. The figure shows that after the initial transient the distance is positive in most of the time so the M600 do not get behind the EGV which means that it can always cover some distance in front of it depending on the setting of the camera.

Regarding the simulation the DJI M600 model (Section 3) was developed without having access to the vehicle. The first flight tests with the real vehicle were done at end of 2021 performing also maneuvers for system identification which is currently in progress. The future plan is to integrate the identified flight model of the M600 into the

SIL and fine tune and evaluate the methods on the realistic model strictly considering its 25° pitch (roll) and 65 km/h speed limitations.

7. CONCLUSION

The article presents the Software-in-the-loop (SIL) simulation of the forerunner UAV concept. It first introduces the UNREAL4-Carla virtual reality solution where the firetruck emergency ground vehicle (EGV) is driven by a game controller. Then it briefly introduces the Matlab simulation of the DJI M600 hexacopter (derived from datasheets and literature) together with the tracking of the EGV. The latter is based-on UDP communication of planned route, actual position and velocity of the EGV to the M600 flying forward of the EGV at least at braking distance. The possibility of sudden EGV route changes is handled by 'pulling' the M600 above the EGV when they approach an intersection. A downward looking virtual camera is considered for traffic detection in front of the EGV. Vehicle detection results are presented running the algorithms on the target hardware.

The first SIL simulation results show that the object detection can be well tuned and can run on 12-15 fps which can be satisfactory to decide about the motion of any observed vehicle. However, the EGV tracking results show that even in the urban scenario (EGV speed mostly between 30-50 km/h) as large as 80-100 km/h hexacopter speeds can be required to complete the task. Unfortunately the speed limitation of the M600 is 65 km/h so its limits were extended in the simulation to attempt to cover the presented track.

System identification of the M600 dynamic and control model based-on real flight tests is currently in progress. After integrating the resulting M600 model into SIL the system will be fine tuned and the realistic capabilities of the M600 forerunner will be evaluated leading finally to an exhaustive feasibility study.

ACKNOWLEDGEMENTS

This work was supported by the project "Developing innovative automotive testing and analysis competencies in the West Hungary region based on the infrastructure of the Zalaegerszeg Automotive Test Track" GINOP-2.3.4-15-2020-00009.

Part of the research was supported by the Ministry of Innovation and Technology NRDI Office within the framework of the Autonomous Systems National Laboratory Program.

REFERENCES

Almeida, P., Soares de Oliveira, L., Jr, A., Jr, E., and Koerich, A. (2015). Pklot - a robust dataset for parking lot classification. *Expert Systems with Applications*, 42.

Arbanas, B., Ivanovic, A., Car, M., Orsag, M., Petrovic, T., and Bogdan, S. (2018). Decentralized planning and control for uav—ugv cooperative teams. *Auton. Robots*, 42(8), 1601â1618.

Bauer, P., Hiba, A., and Zarandy, A. (2019). Comparison of Mono Camera-based Static Obstacle Position Estimation Methods for Automotive Application. In *2019 27th Mediterranean Conference on Control and Automation (MED)*, 386–391.

Beard, R.W. and McLain, T.W. (2012). *Small Unmanned Aircraft*. Princeton University Press.

Bochkovskiy, A., Wang, C.Y., and Liao, H.Y.M. (2020). Yolov4: Optimal speed and accuracy of object detection.

DJI (2016). *E2000 Pro Tuned Propulsion System User Manual*.

DJI (2017). DJI M600 Pro hexacopter. URL `https://www.dji.com/hu/matrice600-pro`.

DJI (2018). *Matrice 600 Pro User Manual*.

Donoughe, K., Whitestone, J., and Gabler, H.C. (2012). Analysis of firetruck crashes and associated firefighter injuries in the united states. *Annals of advances in automotive medicine. Association for the Advancement of Automotive Medicine. Annual Scientific Conference*, 56, 69–76. URL `https://pubmed.ncbi.nlm.nih.gov/23169118`.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, 1–16.

Guillaume J. J. Ducard, M.D.H. (2011). Discussion and practical aspects on control allocation for a multi-rotor helicopter. *1st International Conference on UAVs in Geomatics*.

Karlsruher Institute fũr Technologie (2012). KIT AIS dataset. URL `https://www.ipf.kit.edu/-downloads_data_set_AIS_vehicle_tracking.php`.

Mostafa Moussid, Adrija Bagchi, H.M. (2015). Dynamic modeling and control of a hexarotor using linear and nonlinear methods. *International Journal of Applied Information Systems (IJAIS)*.

Nagy, M. (2021). *Development and simulation testing of a forerunner UAV system*. Master's thesis, Budapest University of Technology and Economics.

Nagy., M., Bauer., P., Hiba., A., Gáti., A., Drotár., I., Lattes., B., and Kisari., A. (2021). The Forerunner UAV Concept for the Increased Safety of First Responders. In *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems - VEHITS*, 362–369. INSTICC, SciTePress.

NHTSA (2014). The National Highway Traffic Safety Administration and Ground Ambulance Crashes.

NVIDIA (2020). Jetson xavier nx. URL `https://www.nvidia.com/en-us/-autonomous-machines/embedded-systems/-jetson-xavier-nx/`.

Project Forerunner (2021). Forerunner SIL simulation setup video. URL `https://youtu.be/0OHGYtJSFLs`.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv*.

Schiebel Corporation (2022). Schiebel camcopter s-100. URL `https://schiebel.net/products/-camcopter-s-100-system-2/`.

Shokry, H. and Hinchey, M. (2009). Model-based verification of embedded software. *Computer*, 42(4), 53–59.

Wojke, N., Bewley, A., and Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, 3645–3649. IEEE.