Quadcopter Simulation Model for Research of Monitoring Tasks

Artemii Zenkin, Ivan Berman, Kanstantsin Pachkouski,
Igor Pantiukhin, Vyacheslav Rzhevskiy
ITMO University
Saint Petersburg, Russia
{a.zenkin, iaberman, k.pachkouski, zevall, vrzhevskiy}@itmo.ru

***Abstract*—The article describe a simulation of the drone designed to monitor a large area. Three scenarios are considered: detection of the fire source in the forest, detection of intruders in the forbidden territory and detection o their cars. Open source software is used for the simulation: robotic simulator Gazebo, framework for robotic applications Robot Operating System, communication protocol for small unmanned vehicles MAVLink and autopilot system PX4. Iris+ quadcopter is chosen as a prototype for the simulation, its mathematical model is presented as well as the vision model based on the PX4FLOW monocular camera and optical flow. Algorithms for detecting objects of interest are described. As a result, the successful tests of simulation are presented, in which the classification and localization accuracy is tested.**

I. Introduction

In the last decade, the field of mobile aerial robotics has attracted considerable attention from the scientific community and the commercial sector. Prospects for the use of unmanned aerial vehicles (UAVs) for tasks, that were previously difficult and costly to automate, forces researchers and developers to improve unmanned technology. Innovative developments allow the use of drones in many existing industries: for rescue services [1], in logistics [2], in building and construction [3], in agriculture [4], in telecommunications [5]. The potential of unmanned aerial vehicles is most revealed in the field of monitoring, especially in cases of a large area or difficult to access infrastructure. The deployment of round-the-clock monitoring of such objects is one of the costly tasks for automation. For natural areas or agricultural land, mobile monitoring is more effective than conventional monitoring methods [6], [7].

An important example of the use of drones is fire monitoring of forests [8]. Currently, fire services use three main methods for monitoring fires: fire towers, the helicopter monitoring and satellite surveillance. Each of them has disadvantages in cost and effectiveness. Fire towers have a limited range of vision, continuous monitoring with helicopters is expensive and satellite surveillance is not an accurate method for the rapid detection of fires [9]. On the other hand, monitoring by drones can be deployed to automatically search for fires in real time, and at a lower cost than conventional methods [10], [11], [12].

Another example of the use of drones in monitoring is the recording of illegal activities. Among the examples are: illegal logging, illegal digging of solid waste landfills, damage to oil and gas pipelines [13], [14], [15]. In such cases, offenders often manage to leave the area / destroy evidence while the security services react. Monitoring with the help of UAVs allows detecting zones of vegetation oppression, violation of sealing of solid waste landfills and other signs of illegal activity.

The academic community has done a lot of research on the use of drones to detect objects of interest [16]. However, the mass introduction of UAVs in monitoring services has not yet occurred. One of the reasons is that various monitoring scenarios impose their limits on the design of a monitoring system. For a large area, organizational questions arise [17]. How many UAV units should be involved in the mission? Which trajectory is optimal? At what altitude and at what velocity drones should fly? There are also issues of optimal fleet management and visual capture of the area with computer vision [18], [19]. Solving these issues in one way or another required tests and experiments. Since real testing carries risks of equipment loss, it is easier and more beneficial to use simulation programs.

This article is dedicated to creating a convenient and undemanding to the computational power simulation for testing monitoring missions with UAVs. The development of such simulations was carried out earlier, and this article differs from similar ones in a number of points. It differs from such works as [20] in that it considers primarily the task of monitoring a large area, which has been little studied [21]. Another example is the articles [22], [23] which present platforms for simulation based on Unreal Engine 4 with support for realistic physics. However, such works are for the most part intended for testing vision, rather than for simulating the entire monitoring system. Articles show that Unreal Engine software is very demanding on computing power, which can be critical when testing simulation in the field (for example, before launching an UAV at a monitoring object). In our case, a less demanding Gazebo simulator is used, which can work on computers with a small amount of video memory.

In paper [24], the authors developed a simulation platform based on the Gazebo simulator and the Robot Operating System framework. To simulate computer vision, the authors added inertial measurement unit (IMU) and sonar to the UAV model. However, sonar is already built into most drone models and the nuances of its functioning on UAVs are quite well

understood. IMU cameras are ineffective without gimbal. In our case, we simulate a realistic camera with a low resolution for working with the optical flow.

Thus, the main advantages of our work for testing of area monitoring system are:

- realistic computer vision;
- great attention to the mathematical model of the drone and computer vision;
- the use of open software that is not demanding on computing power.

The article is organized as follows. Section II describes the open source software used. Section III presents the mathematical model of a drone. A description of the vision system and its algorithms is presented in Section IV. The Section V is devoted to three monitoring scenarios, and the Section VI presents the simulation test results. The conclusion are summarized in the last section.

II. Software Description

The monitoring system is organized using the following software:

- Gazebo is a robotic simulator that allows to simulate robot's operation both indoors and outdoors [25].
- Micro Air Vehicle Link (MAVLink Developer Guide, Web: mavlink.io) is a communi-cation protocol for Micro Air Vehicle or MAV. The protocol establishes interaction between the MAV and the ground control station, as well as their constituent parts and components. Basic telemetry information is packaged in a special message.
- Robot Operating System (ROS) is an open framework for creating distributed robotic systems. The main abstractions in ROS are nodes, messages, topics, and services. Some nodes publish messages in topics, while others subscribe to topics to obtain the necessary information. This creates a publisher-subscription relationship between nodes. Services also perform the function of communication, but operate on the basis of requests / responses.
- For ROS, the MAVROS package (mavros — ROS Wiki, Web: wiki.ros.org/) was written, which provides the ability to control drones using the MAVLink protocol. MAVROS nodes subscribe to spe-cific topics awaiting commands and publish telemetry to other topics. To control the drone, it is possible to set the target position and yaw angle in the ENU (East North Up) coordinate system, linear and angular velocity and orientation.
- PX4 is a flight controller working on Pixhawk, Pixracer and others boards [26]. It is an autopilot system for autonomous devices consisting of two levels. The first level is a set of modules designed for flight control, the second level consists mainly of device drivers. In addition, the second level includes a simulation layer that allows to use PX4 for simulation.

III. Model of Drone

Iris + is chosen as a prototype for the drone model [27]. It is equipped with a battery that feeds 4 brushless motors with a capacity of 950 revolutions per volt (maximum number of revolutions without load). Thrust of motors allows to fly with the payload to 400 grams. The choice of this model is due to the fact that quadcopters of this class are the most common on the market and they perfectly perform video recording of objects and areas.

The frame of the IRIS + quadcopter consists of 4 frame arms in the transverse configuration and two sets of propellers (Fig. 1). One set rotates clockwise and the other set rotates counterclockwise.

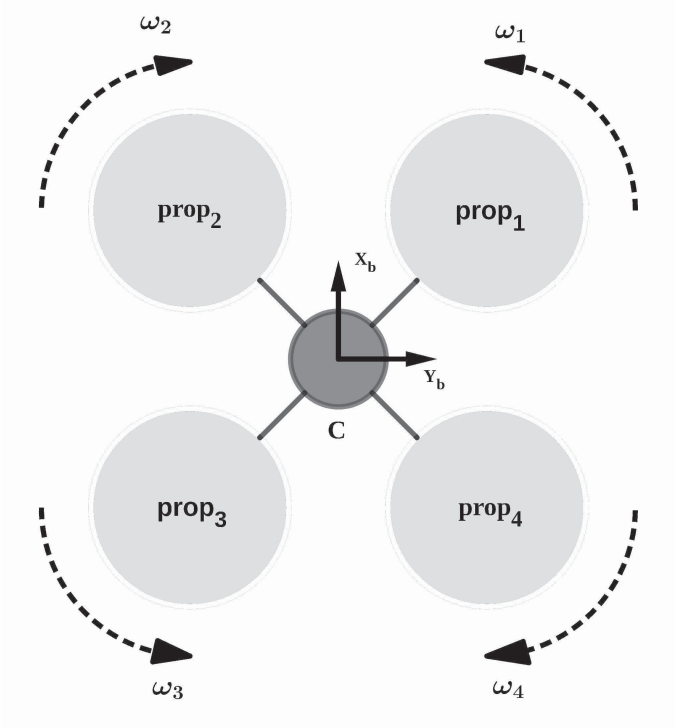


Fig. 1. Quadcopter X_bCY_b plane in transverse configuration

To develop the simulation, the following properties of the quadcopter equipment is taken into account:

- Lithium-ion polymer battery with $5100\ \text{mA h}$. To simulate maximum flight duration, a battery consumption algorithm is provided.
- Four DC motors mounted on the body shell. Their masses and maximum velocity are taken into account.
- Telemetry is implemented using the PX4 autopilot.
- Inertial measurement units (gyro sensor, accelerometer, magnetic meter), GPS module, barometer, laser rangefinder.
- Model of monocular camera PX4FLOW with 800×800 resolution and optical flow.

Below is a mathematical model of a quadcopter. Units are assumed to be standard according SI: m for coordinates,

rad for angles, $\mathrm{m/s}$ for linear velocities, $\mathrm{rad/s}$ for angular velocities, $\mathrm{m/s^2}$ for linear accelerations, $\mathrm{rad/s^2}$ for angular accelerations, N for forces, $\mathrm{N\,m}$ for force moments.

The mathematical model is based on a number of premises:

- The quadcopter design is symmetrical.
- Quadcopter propellers are solid.
- The quadcopter body shell is solid.
- The free air velocity is 0.
- The dynamics of motors can be neglected.
- The flexibility of the propellers is small and it can be neglected.
- Drag force acts linearly according to Stokes' law.
- The general center of mass coincides with the center of mass of the body shell.

Two coordinate frames (c.f.) that describe the orientation and position of the quadcopter are presented below (Fig. 2).

1) Inertial frame $\{i\}$. Axis X_i of inertial c.f. is directed to the north, axis Y_i is directed to the east, and axis Z_i is directed along the radius to the center of the earth. The flight paths and GPS position are determined relative to the inertial frame.
2) The fixed-body coordinate frame $\{b\}$, rigidly connected with the body shell. It has a beginning at the center of gravity of the quadcopter. C.f. rotates with the quadcopter and describes the motion relative to the inertial frame. Axis X_b always points to the front of the quadcopter, axis Y_b points to its right side, and axis Z_b points down. Such an arrangement of the axes is chosen because calculations are performed in this c.f. for the drone controller and on-board sensors. Also in this c.f., aerodynamic forces and moments are measured.

The angle between axes X_i and X_b is the pitch angle θ, between axes Y_i and Y_b is the yaw angle ψ, and between axes Z_i and Z_b is the roll angle ϕ.

In the model, the following forces and moments are taken into account: gravity force; thrust of the quadcopter; roll moment; pitch moment; yaw moment. Rotational motion creates a gyroscopic effect that acts on the quadcopter and depends on the inertial characteristics of the rotor. But since these characteristics are very small, the gyroscopic effect can be neglected.

The nonlinear equation of motion of the quadcopter was taken from work [28] and it is as follows:

$$\begin{bmatrix} \dot{x}^i \\ \dot{y}^i \\ \dot{y}^i \end{bmatrix} = R_b^i \begin{bmatrix} u \\ v \\ w \end{bmatrix},$$

$$R_b^i = \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & C_\phi C_\psi S_\theta + S_\phi S_\psi \\ C_\theta C_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & C_\phi S_\psi S_\theta + S_\phi C_\psi \\ -S_\theta & S_\theta C_\theta & C_\phi C_\theta \end{bmatrix},$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} p + q \cdot S_\phi \cdot \tan\theta + r \cdot C_\phi \cdot \tan\theta \\ q \cdot C_\phi - r \cdot S_\phi \\ q \cdot S_\phi \cdot \sec\theta + r \cdot C_\phi \cdot \sec\theta \\ v \cdot r - q \cdot w - g \cdot S_\phi \\ p \cdot w - u \cdot r + g \cdot C_\theta \cdot S_\phi \\ u \cdot q - p \cdot v + g \cdot C_\theta \cdot C_\phi \\ q \cdot r \cdot (I_{yy} - I_{zz})/I_{xx} \\ p \cdot r \cdot (I_{zz} - I_{xx})/I_{yy} \\ q \cdot r \cdot (I_{xx} - I_{yy})/I_{zz} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ F_z/m \\ L/I_{xx} \\ M/I_{yy} \\ N/I_{zz} \end{bmatrix},$$

where u, v, w — linear velocities along axes X, Y, Z in c.f. $\{b\}$; v — linear velocity along axis Y in c.f. $\{b\}$; $\dot{p}$ — quadcopter roll acceleration along the axis X in c.f. $\{b\}$; $\dot{q}$ — quadcopter pitch acceleration along the axis Y in c.f. $\{b\}$; $\dot{r}$ — quadcopter yaw acceleration along the axis Z in c.f. $\{b\}$; F_z, L, M, N — aerodynamic forces and moments: total thrust, torque effect of roll, pitch and yaw; m — quadcopter mass; I_{xx}, I_{yy}, I_{zz} — moments of inertia about the axes X_i, Y_i, Z_i; ϕ, θ, ψ — roll, pitch, yaw angles in c.f. $\{i\}$; x^i, y^i, z^i — quadcopter position along the axes X, Y, Z in c.f. $\{i\}$; g — gravitational acceleration; S_*, C_* — sine and cosine of the corresponding angles.

The state-space model can be written as:

$$\dot{X}(t) = AX(t) + Bu(t),$$

$$X(t) = \begin{bmatrix} \phi & \theta & \psi & p & q & r \end{bmatrix}^T,$$

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

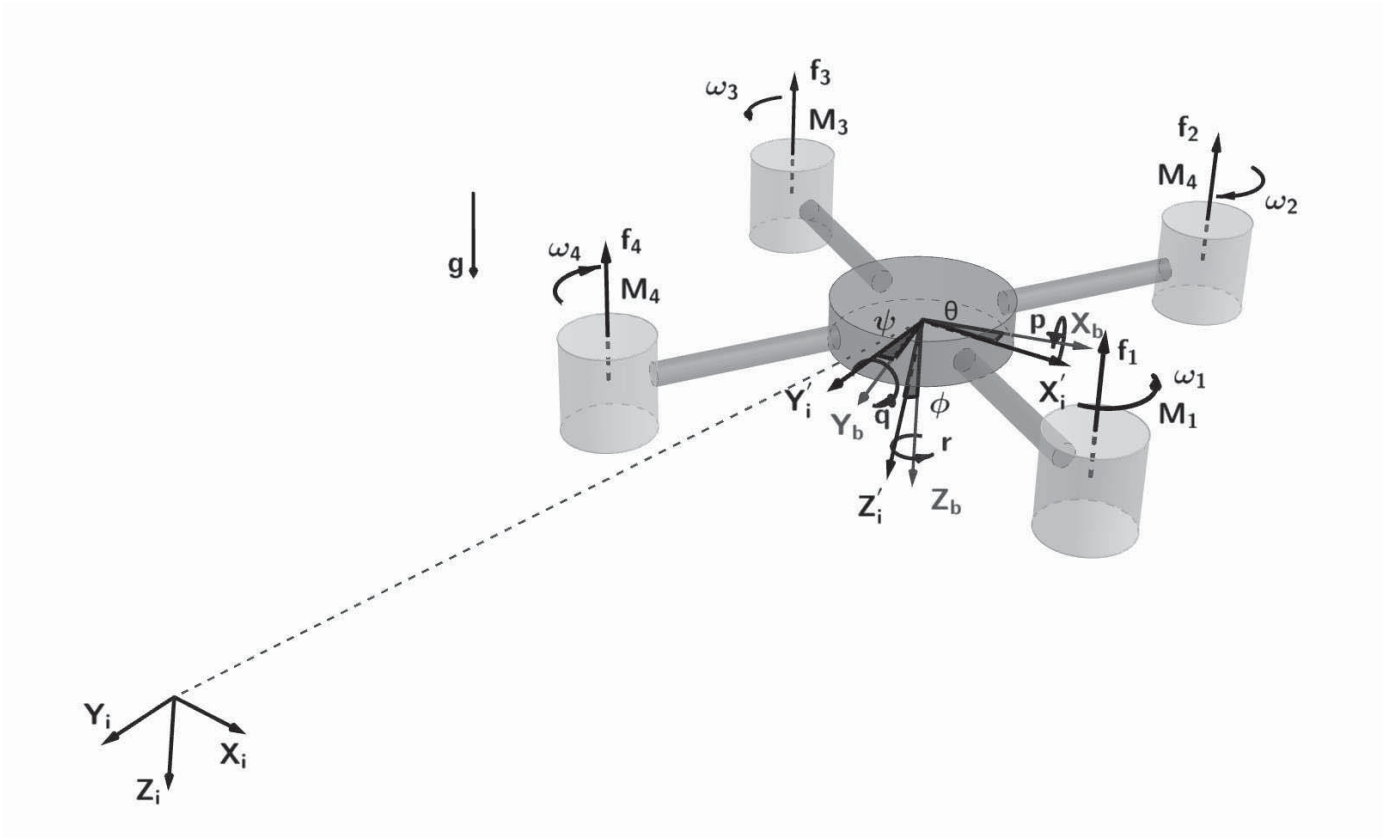


Fig. 2. Quadcopter scheme used in a mathematical model

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1/I_{xx} & 0 & 0 \\ 0 & 1/I_{yy} & 0 \\ 0 & 0 & 1/I_{zz} \end{bmatrix} \times$$

$$\times \begin{bmatrix} -bL_{xB} & bL_{xF} & bL_{xB} & -bL_{xF} \\ bL_{yF} & -bL_{yB} & bL_{yF} & -bL_{yB} \\ d & d & -d & -d \end{bmatrix},$$

$$u(t) = \begin{bmatrix} \omega_1^2 & \omega_2^2 & \omega_3^2 & \omega_4^2 \end{bmatrix}^T,$$

where ω_i — rotation velocity of the i-th motor; b — thrust coefficient; d — propeller drag coefficient; $L_{xB}, L_{xF}, L_{yB}, L_{yF}$ — moment arms. Moment arms are shown in Fig. 3.

Since the accuracy of the mathematical model depends on physical parameters, we decided to use the real parameters of the IRIS + quadcopter. Moment arms lengths and quadcopter mass were found by measuring tape and electronic scales, propeller thrust and drag coefficients were taken from the

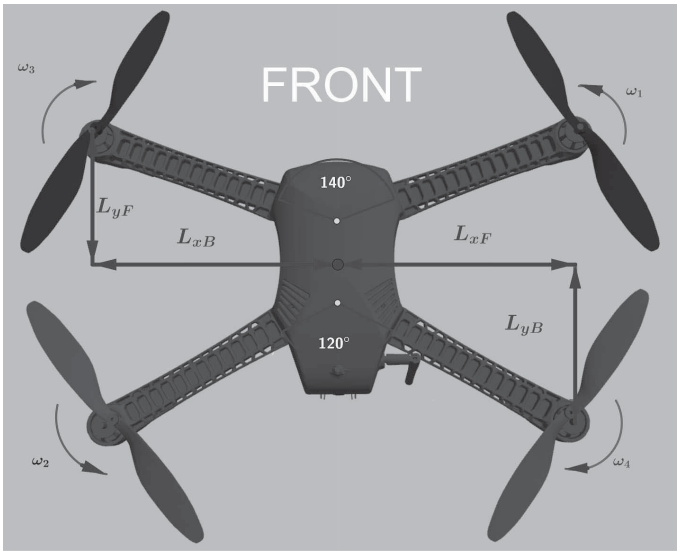


Fig. 3. Illustration of moment arms of IRIS+

quadcopter specification. The moment of inertia was determined twice using analytical and experimental methods, then the results of both methods were analyzed.

For the experimental determination of the moments of inertia, the approach described in [27] was applied. In this approach a trifillary pendulum is used to measure quadcopter oscillations along each of its axes. For the analytical determi-

nation, the components of the quadcopter were approximated as geometric shapes: quadcopter body shell and its motors as cylinders, quadcopter rays as thin rods (Fig. 2). Thus, the moment of inertia for each of the geometric shapes was found individually, and then we found the moments of inertia I_{xx}, I_{yy} and I_{zz} using the Huygens–Steiner theorem [27]. All found quadcopter parameters are listed in the Table I.

TABLE I. IRIS + PHYSICAL PARAMETERS

	Parameter	Value
m	quadcopter mass, kg	1.689
I_{xx}	moment of inertia, kg m^2	0.0220
I_{yy}	moment of inertia, kg m^2	0.0108
I_{zz}	moment of inertia, kg m^2	0.0309
b	thrust coefficient, kg m	$7.2115 \cdot 10^{-6}$
d	drag coefficient, kg m^2	$1.6473 \cdot 10^{-7}$
L_{xB}	arm, m	0.228
L_{xF}	arm, m	0.228
L_{yF}	arm, m	0.128
L_{yB}	arm, m	0.128

IV. COMPUTER VISION

A vision system is represented by a monocular camera. It is modeled taking into account internal parameters such as camera weight, matrix resolution, viewing angle, culvilinear and tangential distortion. The camera is fixed to the bottom of the quadcopter under the center of mass and is directed downward, perpendicular to the ground. To obtain an image from the camera and transmit processed information about the position of the object, the ROS framework is used. Computer vision is implemented for two tasks: detecting a fire in the forest and detecting intruding into a forbidden area.

A. Detection

The majority voting algorithm is used to detect a fire source [29]. The algorithm operates on the basis of several of the most robust conditions for detecting pixels of a fire image. In our case, if at least seven conditions out of eleven are triggered, then the pixel is considered to belong to the fire, otherwise — to the background. This algorithm has low computational complexity, since the belonging of a pixel to a fire image is determined using a pre-calculated table, where a known belonging value is assigned to each color. As a dataset, 300 images of fires in the forest are used. All conditions of the algorithm are defined in the following format:

$$r_i(x) = f_i(x) - c_i,$$

where x — pixel coordinate in the image; $f_i(x)$ — function of a condition i; $r_i(x)$ — resulting function of a condition i; c_i — a constant. As conditions, the most effective conditions from [29] were selected.

As the first condition, the algorithm from [30] is used, in which a three-dimensional histogram is constructed in the RGB space based on segmented images from the dataset. The result is a function $f(r, g, b)$, where r, g, b — color components of red, green and blue. This function takes positive values for points that probably belong to the fire, and negative for points that belong to the background. The first condition can be described as follows:

$$r_1(x) = f[I(x)],$$

where $I(x)$ — pixel color at point x in the image.

The second condition uses $a*$ and $b*$ channels from the $L*a*b$ color space and looks like this [29]:

$$r_2(x) = I_{a*}(x) + I_{b*}(x) - t_1,$$

where $I_{a*}(x)$ and $I_{a*}(x)$ — saturation of channel $a*$ and $b*$ in point x; $t_1 = 32$ — threshold constant.

The third condition uses the RGB color space and the fact that for fire the saturation of red will be much greater than the saturation of blue or green:

$$r_3(x) = I_R(x) + \min[I_R(x), I_G(x), I_B(x)] - t_2,$$

where $t_2 = 72$ — threshold constant.

The fourth and fifth conditions use the RGB and YUV color spaces [31]. Using the fourth condition, high saturation areas are searched in the image:

$$r_4(x) = I_v(x) - t_3,$$

where $I_v(x)$ — saturation value for channel V of the color space YUV; t_3 — threshold value that is found using the Otsu method [32]. Then, the selected areas are analyzed using the RGB color space. This is how a three-dimensional Gaussian model is built on the basis of the color model of fire using labeled images from the dataset.

The fifth condition is as follows:

$$r_5(x) = -\sqrt{\sum_{C \in \{R,G,B\}} (I_C(x) - m_C)^2} + \tau_\sigma \cdot \sigma,$$

where m_C — average value of channel C for pixels satisfying the condition r_4; $\sigma = \max\{R, G, B\}(\sigma_C)$ — standard deviate of channel C for pixels satisfying the condition r_4. The coefficient $\tau_\sigma = 2.5$ was found empirically.

The six remaining conditions use YC_bC_r color space [33]. The sixth and seventh conditions use the fact that in most cases the saturation of red and the brightness component in th YC_bC_r color space are greater than the saturation of blue:

$$r_6(x) = I_Y(x) - I_{C_b}(x),$$
$$r_7(x) = I_{C_r}(x) - I_{C_b}(x),$$

In the following three conditions, the average value of the brightness component is used to search for bright areas in the image:

$$r_8(x) = I_Y(x) - Y^m,$$
$$r_9(x) = C_b^m - I_{C_b}(x),$$
$$r_{10}(x) = I_{C_r}(x) - C_r^m,$$

where Y^m, C_b^m and C_r^m — saturation average values of channels Y, C_b and C_r.

The following rule uses the fact that there is a difference between the pixel saturations on channels C_b and C_r.

$$r_{11}(x) = |I_{C_b}(x) - I_{C_r}(x)| - t_4,$$

where the coefficient $t_4 = 40$ was found empirically.

The maximum numbers at which the values of the resulting function of the conditions take positive values in the image areas belonging to the fire are chosen as coefficients t_1, t_2, t_3, t_4. Minimum number is chosen as a coefficient τ_σ.

For detection of intruders from a drone's altitude, a machine learning method is used, based on boosting feature of histogram of oriented gradients [34]. It is robust to light changes, effective in describing the shape of objects and undemanding in computing power.

First, features are extracted from the image, and then they are used to train the classifier using the AdaBoost method. At the output of the trained classifier, a set of boosting features is obtained. The vector of boosting features has a dimension five times smaller than a regular vector of oriented gradients, and this significantly reduces the time spent on the algorithm. Then these features are used to obtain a classifier based on support vectors. Its main advantage is the effectiveness for this classification task. The final classifier is used to detect people and cars.

The described methods of computer vision are chosen taking into account the fact that after the simulation, the monitoring system should be tested on real equipment installed on the UAV. Preliminary tests showed that on a single-board computer with a Cortex-A53 processor (ARM v8) and a frequency of 1.4 GHz, the image processing time takes 41 ms, which corresponds to 24 frames per second.

B. Determining the Position of Objects on the Map

ROS topics provide information about the current position and orientation of the quadcopter relative to the inertial coordinate frame. Using this data and camera data, we need to localize the objects of interest (Fig. 4).

The coordinate transformation matrix from the c.f. of the drone to the inertial coordinate frame has the following form:

$$M_b^i = \begin{bmatrix} \ddots & & \iddots & \vdots \\ & R_b^i & & P^i \\ \iddots & & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where R_b^i — rotation matrix from c.f. $\{b\}$ to $\{i\}$; P^i — position of the quadcopter in inertial c.f.

Since the camera is fixedly mounted on the UAV, the transformation matrix from the camera coordinate frame to the drone coordinate frame looks like:

$$M_{\text{cam}}^b = \begin{bmatrix} \ddots & & \iddots & \vdots \\ & R_{\text{cam}}^b & & P^b \\ \iddots & & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where R_{cam}^b — rotation matrix from the camera c.f. to the drone c.f.; P^b — camera position in drone c.f.

Then the coordinate of the point P^{map} in a inertial frame looks like:

$$P^{\text{map}} = M_b^i \cdot M_{\text{cam}}^b \cdot P^{\text{cam}}$$

where P^{cam} — coordinates of the point relative to the camera.

Since a monocular camera is used, and only the position of the object on the image plane is known, to determine the three-dementional position of the object, it is necessary to project its image back to the ground. Let the surface of the earth be represented as a discrete function $z = h(x, y)$. The matrix of internal parameters of the camera is as follows

$$M_{\text{int}} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}.$$

There are two focal lengths in this matrix: $f_x = F \cdot sx$ and $f_y = F \cdot sy$, where F — focal length in mm; sx and sy — the number of pixels per millimeter along the camera axes X and Y; c_x and c_y — coordinates of the center of the image in pixels. All four parameters f_x, f_y, c_x and c_y are determined by calibrating the camera.

Then the vector that determines the possible position of the point, that is projected to the image point $P_{im} = (p_x; p_y; 1)$, has the form:

$$\bar{a}_{\text{cam}} = M_{\text{cam}}^{-1} \cdot \bar{p}_{\text{im}},$$

where $\bar{p}_{\text{im}}$ — radius vector.

By adding the coefficient k and the homogeneous coordinate to the coordinates of the vector $\bar{a}_{\text{cam}}$, the point is obtained:

$$b_{\text{cam}}(k \cdot a_x; k \cdot a_y; k \cdot a_z; 1).$$

With a certain coefficient k, the point b_{cam} will be on the plane X_iOY_i of the inertial coordinate frame. The search for this coefficient is carried out using the equality:

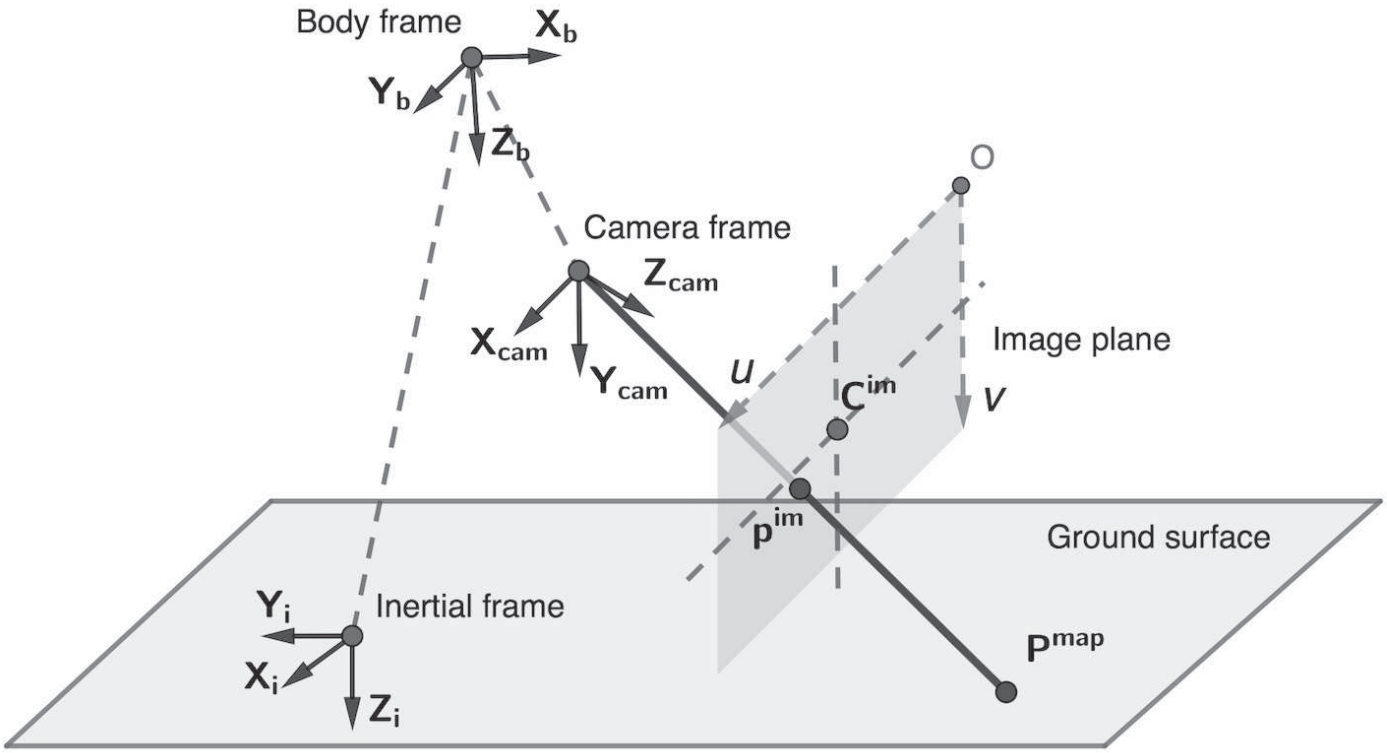


Fig. 4. Projecting a point from an image onto the surface of the ground: P^{map} — point of interest on the ground, P^{im} — image of point P^{map}

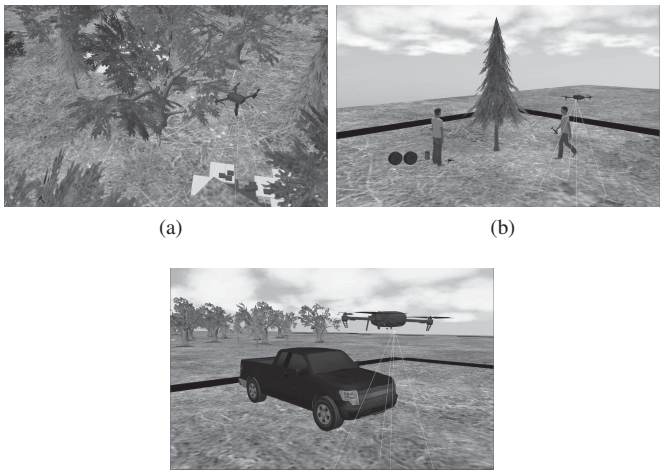
$$(M_b^i \cdot M_{\text{cam}}^b \cdot b_{\text{cam}})_z = (M_b^i \cdot M_{\text{cam}}^b \cdot \begin{bmatrix} k \cdot a_x \\ k \cdot a_y \\ k \cdot a_z \\ 1 \end{bmatrix})_z = (b_{\text{map}})_z = 0.$$

Expressing k and substituting in b_{cam}, we can find the position of the object for the case when the function $h(x, y)$ of the ground's surface is zero throughout the plane X_iOY_i. Then, using the iterative method, the position of the object is refined taking into account the values of the function $h(x, y)$. Thus, the position of objects of interest on the map can be obtained by projecting their contours from the image into three-dimensional space.

V. Simulation Scenarios

All objects were modeled in Blender, a software for creating 3D images. In Gazebo simulator, an environment was created that included three zones: a forest in fire (Fig. 5a), the area with illegal activity (Fig. 5b), the area with a car of intruders in a forbidden area (Fig. 5c).

The general simulation scenario is as follows. For each task, flight boundaries and altitude are determined in advance. The quadcopter control algorithm performs take off, then directs the quadcopter to the study area and begin monitoring. As soon as the object of interest is detected, the quadcopter will hang over it and will launch the algorithms of computer vision. Further information about the objects is transmitted to the user console.



(a) (b)

(c)

Fig. 5. Simulation Scenarios: a) a forest in fire, b) illegal activity, c) car of intruders

VI. Simulation Results

F-score coefficient is used as a metric for evaluation the recognition efficiency of objects.

$$F = \frac{2 \cdot P \cdot R}{P + R},$$

$$P = \frac{TP}{TP + FP},$$

$$R = \frac{TP}{TP + FN},$$

where TP — number of true positive results; FP — number of false positive results; FN — number of false negative results.

The quadcopter was commanded to fly around at seven different heights (H) with velocity of 15, 30 and 45 $\mathrm{km/h}$ an area in the form of a square with 600×600 m dimensions. The time for which the quadcopter flies around the area, the error in determining the position of objects and the F-score were determined. The error e was determined as the difference between the calculated coordinate of the object and the real coordinate from Gazebo. The simulation results are shown in Table II, III and IV.

TABLE II. RESULTS AT THE VELOCITY OF 15 $\mathrm{km/h}$

N	H, m	t, min	e, m	F-score		
				fire	car	human
1	30	35.35	1.79	0.963	0.965	0.959
2	40	27.00	1.75	0.985	0.956	0.950
3	50	22.62	1.84	0.983	0.948	0.938
4	60	19.93	1.90	0.984	0.932	0.927
5	70	17.33	2.03	0.970	0.925	0.921
6	80	15.12	2.13	0.969	0.922	0.904
7	90	14.37	2.21	0.976	0.916	0.899

TABLE III. RESULTS AT THE VELOCITY OF 30 $\mathrm{km/h}$

N	H, m	t, min	e, m	F-score		
				fire	car	human
1	30	17.68	1.80	0.975	0.968	0.962
2	40	13.48	1.79	0.976	0.959	0.953
3	50	11.33	1.83	0.980	0.930	0.930
4	60	9.93	1.94	0.977	0.924	0.919
5	70	8.67	2.05	0.969	0.921	0.912
6	80	7.58	2.12	0.982	0.921	0.895
7	90	7.20	2.25	0.962	0.914	0.879

TABLE IV. RESULTS AT THE VELOCITY OF 45 $\mathrm{km/h}$

N	H, m	t, min	e, m	F-score		
				fire	car	human
1	30	11.78	1.85	0.972	0.959	0.950
2	40	8.98	1.82	0.984	0.953	0.945
3	50	7.29	1.89	0.967	0.941	0.931
4	60	6.48	2.01	0.974	0.930	0.918
5	70	5.77	2.10	0.981	0.923	0.903
6	80	5.02	2.19	0.982	0.920	0.881
7	90	4.83	2.31	0.978	0.908	0.865

VII. CONCLUSION

The simulation model has successfully demonstrated its operability for monitoring tasks. Simulation testing showed that with height, the quadcopter covers the area faster, but at the same time, the accuracy of the algorithm for detecting a car and a human deteriorates. This is due to the reduction in the size of objects in the image. The error in the coordinates of the object is caused by the inaccuracy of the localization sensors, and it amplifies with height. The fire recognition algorithm at altitudes from 30 to 90 m and at speeds from 15 to 45 $\mathrm{km/h}$ shows approximately the same results. This is due to fact that the basis of this algorithm is the color palette of fire.

Further experiments of the simulation model will be devoted to the use of more modern algorithms for computer vision, primarily deep learning methods. The performance of a multi-agent monitoring case and various algorithms for their optimal management will be tested. Other monitoring scenarios will also be explored, in particular the search for missing people during natural disasters.

REFERENCES

[1] R. Grodi, D.B. Rawat, "UAV-assisted broadband network for emergency and public safety communications", in *Proceedings of 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2015, pp. 10–14.

[2] H. Menouar, et al. "UAV-enabled intelligent transportation systems for the smart city: Applications and challenges", *IEEE Communications Magazine*, vol. 55, no. 3, 2017, pp. 22–28.

[3] J. Howard, V. Murashov, C.M. Branche, "Unmanned aerial vehicles in construction and worker safety", *American Journal of Industrial Medicine*, vol. 61, no. 1, 2018, pp. 3–10.

[4] P. Tokekar, et al. "Sensor planning for a symbiotic UAV and UGV system for precision agriculture", *IEEE Transactions on Robotics*, vol. 32, no. 6, 2016, pp. 1498–1511.

[5] Y. Gu, et al., "Airborne WiFi networks through directional antennae: An experimental study", in *Proceedings of 2015 IEEE Wireless Communications and Networking Conference (WCNC)*, 2015, pp. 1314–1319.

[6] G. Lindner, et al. "UAV monitoring and documentation of a large landslide", *IEEE Applied Geomatics*, vol. 8, no. 1, 2016, pp. 1–11.

[7] Y. Zhang, et al. "Remote monitoring of heading rice growing and nitrogen content based on UAV images", *International Journal of Smart Home*, vol. 10, no. 7, 2016, pp. 103–114.

[8] C. Yuan, Y. Zhang, Z.Liu, "A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques", *Canadian Journal of Forest Research*, vol. 45, no. 7, 2015, pp. 783–792.

[9] L. Yu, N. Wang, X. Meng, "Real-time forest fire detection with wireless sensor networks", in *Proceedings of 2005 International Conference on Wireless Communications, Networking and Mobile Computing*, vol. 2, 2005, pp. 1214–1217.

[10] K.A. Ghamry, M.A. Kamel, Y. Zhang, "Cooperative forest monitoring and fire detection using a team of UAVs-UGVs", in *Proceedings of 2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2016, pp. 1206–1211.

[11] B.R. Christensen, "Use of UAV or remotely piloted aircraft and forward-looking infrared in forest, rural and wildland fire management: evaluation using simple economic analysis", *New Zealand Journal of Forestry Science*, vol. 45, no. 1, 2015, p. 16.

[12] L. Merino, J.R. Martinez-de Dios, A. Ollero, "Cooperative unmanned aerial systems for fire detection, monitoring, and extinguishing", in *Handbook of Unmanned Aerial Vehicles*, 2016, pp. 2693–2722.

[13] R.K. Rangel, A.C. Terra, "Development of a surveillance tool using UAV's", in *Proceedings of 2018 IEEE Aerospace Conference*, 2018, pp. 1–11.

[14] D. Gasperini et al, "Potential and limitation of UAV for monitoring subsidence in municipal landfills", *International Journal of Environmental Technology and Management*, vol. 17, no. 1, 2014, pp. 1–13.

[15] A. Shukla, H. Xiaoqian, H. Karki, "Autonomous tracking of oil and gas pipelines by an unmanned aerial vehicle", in *Proceedings of 2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2016, pp. 1–4.

[16] L. Gonzalez et al. "Unmanned aerial vehicles (UAVs) and artificial intelligence revolutionizing wildlife monitoring and conservation", *Sensors*, vol. 16, no. 1, 2016, p. 97.

[17] Y. Altshuler, A. Pentland, A.M. Bruckstein, "Optimal dynamic coverage infrastructure for large-scale fleets of reconnaissance UAVs", in *Swarms and Network Intelligence in Search*, 2018, pp. 207–238.

[18] M.A. Messous, S.M. Senouci, H. Sedjelmaci, "Network connectivity and area coverage for UAV fleet mobility model with energy constraint", in *Proceedings of 2016 IEEE Wireless Communications and Networking Conference*, 2016, pp. 1–6.

[19] R. Bailon-Ruiz, S. Lacroix, A. Bit-Monnot, "Planning to monitor wildfires with a fleet of UAVs", in *Proceedings of 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4729–4734.

[20] F. Wan et al, "Multi-sensor devices for UAV in both simulation platform and realistic tests", in *Proceedings of 12th World Congress on Intelligent Control and Automation (WCICA)*, 2016, pp. 2986–2990.

[21] S. Siebert, J. Teizer, "Mobile 3D mapping for surveying earthwork projects using an Unmanned Aerial Vehicle (UAV) system", *Automation in Construction*, vol. 41, 2014, pp. 1–14.

[22] M. Mueller, N. Smith, B. Ghanem, "A Benchmark and Simulator for UAV Tracking", in *Proceedings of European Conference on Computer Vision*, 2016, pp. 445–461.

[23] O. Ganoni, R. Mukundan, "A Framework for Visually Realistic Multi-robot Simulation in Natural Environment", *arXiv e-prints*. 2017. Web: http://arxiv.org/abs/1708.01938.

[24] F. Wan et al, "Multi-sensor devices for UAV in both simulation platform and realistic tests" in *Proceedings of the 2016 12th World Congress on Intelligent Control and Automation (WCICA)*, 2016, pp. 2986–2990.

[25] J. Meyer, et al. "Comprehensive simulation of quadrotor UAVs using ROS and Gazebo", in *Proceedings of International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, 2012, pp. 400–411.

[26] L. Meier, D. Honegger, M. Pollefeys, "PX4: A node-based multi-threaded open source robotics framework for deeply embedded platforms", *Proceedings of 2015 IEEE international conference on robotics and automation (ICRA)*, 2015, pp. 6235–6240.

[27] W.Z. Fum, "Implementation of Simulink controller design on Iris+ quadrotor", Ph.D. dissertation, Naval Postgraduate School, Monterey, 2015.

[28] A. Freddi, A. Lanzon, S. Longhi, "A feedback linearization approach to fault tolerance in quadrotor vehicles", *IFAC Proceedings Volumes*, vol. 44, no. 1, 2011, pp. 5413–5418.

[29] T. Toulouse et al, "Automatic fire pixel detection using image processing: a comparative analysis of rule-based and machine learning-based methods", in *Signal, Image and Video Processing*, vol. 10, no. 4, 2016, pp. 647–654.

[30] W. Phillips, M. Shah, N. Da Vitoria Lobo, "Flame recognition in video", *Pattern Recognition Letters*, vol. 23, no. 1-3, 2002, pp. 319–327.

[31] L. Rossi, M. Akhloufi, "Dynamic fire 3D modeling using a real-time stereovision system", in *Technological Developments in Education and Automation*, 2010. pp. 33–38.

[32] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, 1979, pp. 62–66.

[33] T. Celik, D. Hasan, "Fire detection in video sequences using a generic color model", *Fire Safety Journal*, vol. 44, no. 2, 2009, pp. 147–158.

[34] X. Cao et al. "Linear SVM classification using boosting HOG features for vehicle detection in low-altitude airborne videos" in *Proceedings of 18th IEEE International Conference on Image Processing*, 2011, pp. 2421–2424.