# Drone SLAM Using TDOA-RSS Signal with Applied EKF on PF Data

**Damir Šoštarić**
Graduate Student
Genesis Mining Ltd., Barrage d.o.o.
Óbuda University
Doctoral School of Safety and Security Sciences Budapest
Hungary

**Gyula Mester**
Professor
Óbuda University
Doctoral School of Safety and Security Sciences Budapest
Hungary

*This paper describes the integration of Particle filter data earlier recorded in simulation and tested in real-time scenario. Those data are parsed from beacon listener using TDOA-RSS inverse method. Drone SLAM has non-linear behavior and for the better precision its applied EKF. IMU and camera odometry with optical flow presents input parameters for EKF block. Initial conditions are focused on KF of drone IMU.Monovision camera odometry is used for optical flow on overall odometry covariance of insecurity.To improve optical flow in covariance of insecurity, it used landmarks on the ground. Specific case scenario of EKF is presented in the paper with drone/listener and beacon positions, which belongs to research area of Range-Only SLAM. Mathematical non-linear expressions are presented between previous position of drone and controling signal.Initial conditions are arbitrary,while drone elevation is constant.Energy function has been considered to implement in energy optimizing scenario.*

***Keywords:*** *Particle filter, TDOA-RSS, EKF, SLAM, Optical flow, Energy function.*

## 1. INTRODUCTION

Localization and mapping using SLAM (Simultaneous Localization And Mapping) algorithm will be presented in indoor drone flight scenario. With corresponding drone hardware ARDrone 2.0, its created scenario which is based on TDOA-RSS (Time Differential Of Arrival – Receiver Signal Strength) inverse method of localization [1,2]. General idea is to use six ultrasonic beacons with listener. This research and development platform (Crossbow Cricket) is modified for earlier scenario described in paper [2-11]. Simulations in Matlab are extended from 2D equations to 3D equation system. Sampling of particles in 3D and resampling after moving drone /listener is represented as 3D particle belt, belt-ring and convergence to final solution. Final solution was already defined in that shape after third iteration of movement, where we have two small particle group of solution. When drone command path has trend to move to specific beacon then this solution will be earlier defined. That kind of prepared particle data can be used for input parameters for EKF (Extended Kalman Filter). In sim0-ulation, locally saved data are ready for additional transformation from file to EKF block. In real-time behaviour scenario are used data from locally saved records. Controlling drone, integration of parser beacon data and algorithms integration are realized with LabView, National Instruments [2]. For writing/reading method is used FIFO (First – In – First – Out) Toolkit.

With positioning of drone for SLAM, its necessary to ensure accurate orientation [12–14]. Since that IMU (Inertial Measurement Unit) have 3D accelerometer and 3D gyro sensors, usually we have scenario of gyro drift. To improve and to correct this settings its added TDOA-RSS method which can make position very accurate and precise. To ensure precise orientation there is a couple of methods. One of the methods is to use second listener on object/drone, which wasn't option for us because object dimensions. Second method was based on monovision where odometry based on optical flow can optimize additionally position but also orientation [7].

Integrated camera on drone bottom is limited with SDK (Software Development Kit), while ARDrone Toolkit doesn't have option to transfer video via WiFi. Instead of this camera, its mounted additional WiFi camera. Additional WiFi adapter on workstation controller laptop is integrated especially to make tunnel bridge and to receive video in the same controller software. Improvements of position and orientation are integrated on this way inside of EKF algorithm block like additional average value of sum.

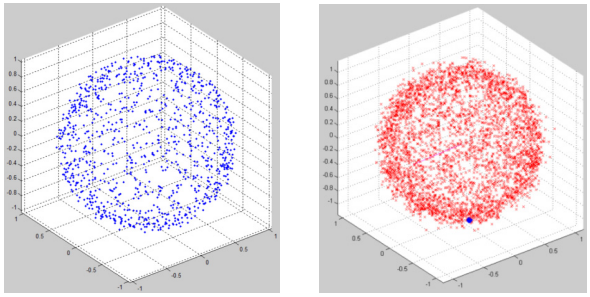
Optical flow can be based on three most used methods. The first method is based on a general approach of BCA (Brightness Constancy Assumption), where is used HS (Horn – Schunck). The second method is focused on ageneral optical flow based on pyramids of frames. This method is called Hierarchical method, where the smallest frame of the pyramid can represent flow vector in shape of one pixel.

The third method represents LK (Lucas and Kanade) algorithm which is used in our case. This method is based on the smoothest term as a box window inside of the frame represented by a group of pixels. Imperfection of this method can be insufficient texture inside the blocks and lots of edges which can cause non-accurate measurements. Flat patch/floor can bring us to the issue where it is hard to estimate a motion. HS method is more global oriented, while LK is local. Best practice shows to use both methods, also known as Bruhn method, while in our case just the LK method is used.

Received: May 2019, Accepted: July 2019
Correspondence to: Damir Šoštarić, Óbuda University, Doctoral School of Safety and Security Sciences Budapest, Népszínház str. 8, 1081 Budapest, Hungary
E-mail: sostaric.damir@gmail.com
**doi:10.5937/fmet1904914S**

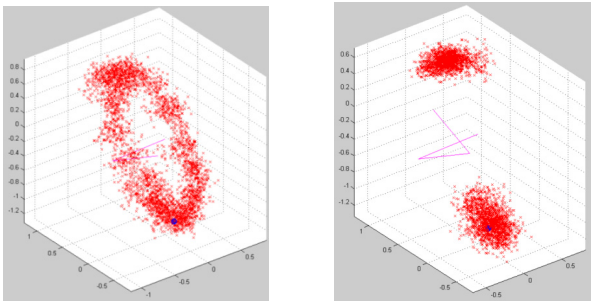
## 2. 3D MATHEMATICAL MODEL OF APPLIED PARTICLE FILTER ON BEACONS

Approach to 3D mathematical model is defined on hypothesis settings. Predefined scene is defined for the drone in indoor environment. With a simulation we isolate variables which can influence control parameters in real – time drone environment. The test case scenario was to use a testbed, where a listener on the drone receives signals from beacons based on TDOA-RSS (Time Differential of Arrival – Receiver Signal Strength) method [2,5]. In this way, we have calculated the distance between each node and a listener. Simulation concept was to use Bayes filter definition with probabilistic methods or algorithms [15,16]. Orientation and position of beacons are observed as random variables, where we have considered deterministic and probabilistic case. Since that deterministic approach is based on the procedures with values which are treated as certain, we use probabilistic procedures and methods. The test case scenario in simulation use Particle filter where is considered that we have one beacon. Applied method is using the numeric approximation based on discarding samples. First step describe 3D sampling of particles where we have 3D cloud, Figure1. a). Blue marker/dot particle, Figure 1. b), represent our hypothesis of beacon position. The position is known to us, but the system is not aware of final solution. With minor movements of drone (listener), its created new step where we have resampling of particles and new 3D cloud, Figure 2. a). This cloud now represent 3D belt of possible solutions. Additional shift of drone requires the next step where approximation is done on discarding samples from previous step. The fourth step is sequence where we can see two unimodal solutions, Figure 2. b).



a) b)

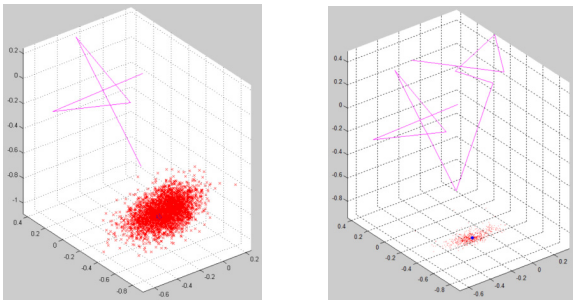
**Figure 1. a) Particle cloud 3D sampling, b) Hypothesis of generated beacon position**



a) b)

**Figure 2. a) Third step: drone shift and resampling of particles – getting cloud belt, b) Fourth step: getting two unimodal solutions**

The fifth step represents one unimodal solution after four iteration, Figure 3. a). The process convergence represents one "Rugby" ball of possible solutions. Iteration after ten steps can define our desired end result of Particle filter, Figure 3. b).



a) b)

**Figure 3. a) Fifth step: getting a unimodal solution, b) The end result of Particle filter**

In simulation case, where we have six beacons, which represent randomised variables, first step requires again sampling, Figure 6. a). Simulation code is represented in two parts in Figure 4 and Figure 5. The first part of *.m Matlab code is based on the distance measurement between drone and beacon where particle circuit/cloud generation is defined on z= +/- 3sigma_b. Since that we have IMU drone insecurity, new particles are added in simulation.

```
%Simulation of drone and 3D cloud
clear all; close all; clc;
%Standard deviation of beacon position (measured noise)
sigma_b = 0.02;
%Standard deviation of IMU
sigma_IMU = 0.005;
%Number of particles
N = 1000;
%Number of added particles
N_add = 3;
%Rnd moving of drone om round square grid m=25
m = 25;
W = [0 0 0.5;[0.5*(2*rand(2,m-1)-1); 0.5*ones(1,m-1)]']'
%Number of beacons
Nb = 6;
%Vector position of beacon
b = rand(3,Nb)-1;
%Simulation of distance measurement between drone/listener and beacon
for i=1:Nb
    tmp = (W(:,1) - b(:,i)).^2;
    z(i,1) = sqrt(sum(tmp)) + sigma_b*randn(1,1);
end
%Particle circuit generation around drone on distance z +- 3sigma_b
%..for each beacon..
X_n = []
for i=1:Nb
    br = 0;
    X_temp = [];
    while(br<N)
        x_rand = (z(i,1)+3*sigma_b)*(2 * rand(3,1) - 1);
        r_c = sqrt(x_rand'*x_rand);
        if( (r_c > (z(i,1)-3*sigma_b)) && (r_c < (z(i,1)+3*sigma_b)))
            X_temp = [X_temp x_rand];
            br = br + 1;
        end
    end
    X_n = [X_n; X_temp];
end
figure;hold on; grid on;
for i=1:Nb
    plot3(X_n(1+3*(i-1),:),X_n(2+3*(i-1),:),X_n(3+3*(i-1),:),'b .','Color',[i*1/Nb 0 0]);
end
axis equal;
We = [0 0 0.5]';
X_rr = X_n;
crtanje = 0;
for quadp = 2:m
   X_n = X_rr;
   X_d = [];
   for k=1:Nb
       %adding new particles because of IMU system insecurity
       X_db = [];
       for i = 1:N
           temp = mvnrnd([0 0 0], sigma_IMU*eye(3,3), N_add)';
           for j = 1:N_add
               Xnew(:,j) = X_n((1+3*(k-1)):(3+3*(k-1)),i) + temp(:,j);
           end
           X_db = [X_db Xnew];
       end
       X_d = [X_d;X_db];
       clear Xnew X_db;
   end
   X_n = [X_n X_d];
   %Simulation of drone - beacon distance measurement
   for i=1:Nb
       tmp = (W(:,quadp) - b(:,i)).^2;
       z(i,1) = sqrt(sum(tmp)) + sigma_b*randn(1,1);
   end
```

**Figure 4. First code part of simulated scenario**

The distance vector in this case can be represented with position vector of beacons, where conditional probability of drone position and beacon positions are one part of substitution. Distance probability is defined in this way as position of drone and beacons where new measurement is everything except last measurement.

By simultaneous measurement, with applied Particle filter on six beacons, we can get final solution of distance vector in 3D test case environment. During simulation with Particle filter, drone altitude was not always the same. This is not the case in real-time environment where we use drone in altitude hold mode with constant hovering height.

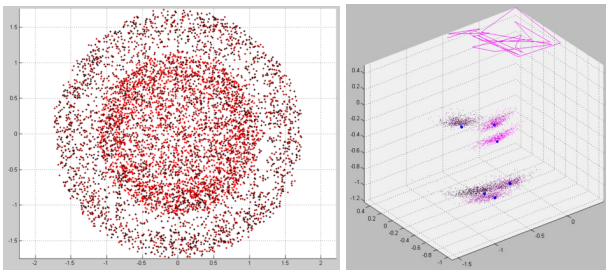
Resampling and weights normalisation is defined in second part of code, Figure 5., where average value of each unimodal distribution is presented in Figure 6. a) including covariance matrix.

Conclusion is next; once when we have movements of drone to one of the beacons we have faster convergence. With bigger number of iterations we have more paths. When those paths cover or when they are closer to "unknown" position of beacons, our process is faster to define unique unimodal solution for each beacon, Figure 6. b).

```
    if(crtanje)
        figure; plot3(X_n(1,:),X_n(2,:),X_n(3,:),'r x');hold on; grid on;
        plot3(b(1),b(2),b(3),'b o','MarkerSize',10, 'MarkerFaceColor','b');
        plot3(W(1,1:quadp),W(2,1:quadp),W(3,1:quadp),'m');
        axis equal
        pause(1);
    end
    We = We + [(W(1:2,quadp) - W(1:2,quadp - 1)) + sigma_IMU * randn(2,1); 0];
    %resample
    X_rr = [];
    for k=1:Nb
        for i = 1:size(X_n,2)

            diff = (z(k,1) - norm(X_n((1+3*(k-1)):(3+3*(k-1)),i)-We))/sigma_b;
            w_c(i) = 1/sqrt(2*pi*sigma_b)*exp(-0.5*diff^2);
        end
        br = 0;
        X_r = [];
       %normalize weights
       w_c_min = min(w_c);
       w_c_min = 0;
       w_c_max = max(w_c);
       w_c_n = zeros(1,length(w_c));
       for i=1:length(w_c)
           w_c_n(i) = (w_c(i) - w_c_min)/(w_c_max-w_c_min);
       end
       [val, ind] = sort(w_c_n);
       val_e = cumsum(val);
       while(br < N)
           r = val_e(end)*rand(1,1);
           for i=1:length(val_e)
               if(r > val_e(i))
                   continue;
               else
                    break;
               end
           end
           X_r = [X_r X_n((1+3*(k-1)):(3+3*(k-1)),ind(i))];
           br = br + 1;
       end
       clear w_c w_c_n val ind val_e;
       X_rr = [X_rr; X_r];
    end
    clear X_n;
    disp(['Pomak quada: ' num2str(quadp)]);
end
figure; plot3(W(1,1:quadp),W(2,1:quadp),W(3,1:quadp),'m');hold on; grid on;
for i=1:Nb
    plot3(b(1,i),b(2,i),b(3,i),'b o','MarkerSize',5, 'MarkerFaceColor','b');
    plot3(X_rr(1+3*(i-1),:),X_rr(2+3*(i-1),:),X_rr(3+3*(i-1),:),'b .',-->
    <--'Color',[i*1/Nb 0 i*1/Nb],'MarkerSize',3, 'MarkerFaceColor','r');
end
axis equal
%Avg. value of each unimodal distribution and covariance matrix
mu_b_0 = mean(X_rr')';
Sigma_b_0 = [];
for k=1:Nb
    sigma_temp = cov(X_rr((1+3*(k-1)):(3+3*(k-1)),:)');
    if(k==1)
        Sigma_b_0 = sigma_temp;
    else
        temp = size(Sigma_b_0,2);
        Sigma_b_0 = [Sigma_b_0 zeros(temp,3)];
        Sigma_b_0 = [Sigma_b_0; zeros(3,temp) sigma_temp];
    end
end
mu_0 = [We(1:2,1); mu_b_0];
Sigma_0 = [[10^2 0; 0 10^2] zeros(2,Nb*3)];
tempp = [zeros(3*Nb,2) Sigma_b_0];
Sigma_0 = [Sigma_0; tempp];
save cesticni_rez sigma_b sigma_IMU Nb mu_0 Sigma_0 W b We
```

**Figure 5. Second code part of simulated scenario**



a) b)

**Figure 6. a) Fifth step: getting a unimodal solution, b) The end result of Particle filter with position vector of six beacons**

## 3. RANGE-ONLY DRONE SLAM – 3D MATHEMATICAL MODEL BEHAVIOR OF APPLIED EXTENDED KALMAN FILTER

EKF is applied on Particle filter data because of non-linear component of the function h. The second reason why EKF is used is because of fast optimisation for positions and distance between drone/listener and beacons. At the same time we got higher position accuracy and distance of the whole six beacons. With unimodal distribution for each beacon from earlier logged data from Particle filter we can estimate the positions with Gauss or normal distribution.

The general idea and understanding of simulation can be presented with Figure 7. where we have real-time scenario ready for real parsing data of measured distances. The 6$^{th}$ beacon in this case scenario is mobile beacon which will be moved/shifted during real measurements around the square grid. Simulation focused facts should stay on the listener and beacons which are positioned as follow in isometric view in Figure 7. Drone IMU (Inertial Measurement Unit) can bring a rough error in measuring, which is caused by gyroscope drift in sampling. That kind of noise can be eliminated with optical flow where total odometry insecurity is reduced. Recorded data from Particle filter simulation are ready for the next step where EKF will be applied.

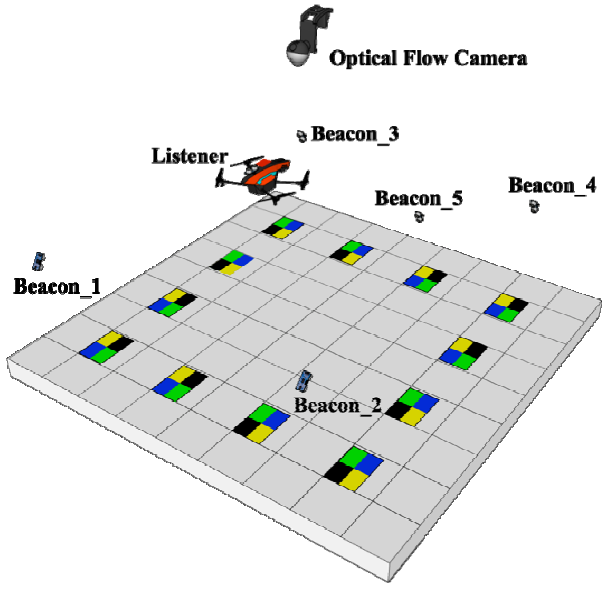



**Figure 7. Real-time test case scenario for measurement and logging data**

Starting with recorded data analyses from Particle filter require initial conditions before we define the next steps. Measured insecurity of odometry is defined with expression (1), where we are considering IMU and camera odometry in the case of Kalman filter. The result representation is state vector $x_t$ which is the output parameter. Insecurity of measurements in the case of beacons is defined with observation vector $z_t$ with measured noise Δ, expression (2). That noise is later represented as covariance matrix $Q_t$. Both parameters are input values of extended Kalman filter, Figure 8. In Figure 8, we also have $R_t$, which is the input parameter of insecurity, strictly for odometry while $u_t$ is insecurity of new measurement. Considered nonlinear system is described with difference equation and the observation model with additive noise (1) and (2). The process nonlinear vector function $g(x_{t-1},u_{t-1})$ and observation nonlinear vector function $h(x_t)$ is given by expression (3) and (4).

$$\mathbf{x_t} = \mathbf{A}\cdot\mathbf{x_{t-1}} + \mathbf{B}\cdot\mathbf{u_{t-1}} + \mathbf{v} \tag{1}$$

$$\mathbf{z_t} = \mathbf{C}\cdot\mathbf{x_t} + \mathbf{\Delta} \tag{2}$$

$$\mathbf{x_t} = \mathbf{g}(\mathbf{x_{t-1}},\mathbf{u_{t-1}}) \tag{3}$$

$$\mathbf{z_t} = \mathbf{h}(\mathbf{x_t}) \tag{4}$$

Extended Kalman filter excerpt for our specific case, with position of drone/listener and each beacon is expressed with (5) and (6). where the first two column elements of the matrix x represent localization and other six group is drone mapping. Nonlinearity between previous position and controlling signal is represented with shift $z_t$ (6) in the direction to drone IMU.

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ x_1 \\ y_1 \\ z_1 \\ x_2 \\ y_2 \\ z_2 \\ x_3 \\ y_3 \\ z_3 \\ x_4 \\ y_4 \\ z_4 \\ x_5 \\ y_5 \\ z_5 \\ x_6 \\ y_6 \\ z_6 \end{bmatrix} \tag{5}$$

$$\mathbf{z} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{bmatrix} \tag{6}$$

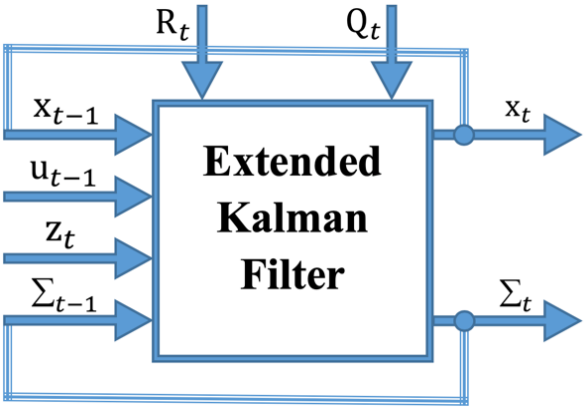



**Figure 8. Applied extended Kalman filter with input and output parameters**

Linearization in some cases can cause a problem because we don't stay in the Gaussian world if motion and/or measurement models are nonlinear functions of the state. Generally, there is no closed-form solution for Bayes filter [5]. The aim is to keep the functions and to approximate distributions. Linearization of the function $g(x_{t-1},u_{t-1})$ around $x_{t-1}$ is prediction where we have g matrix (9) and Jacobian matrix (10). Linearization of the function $h(x_t)$ is given also by first order Taylor expansion (8), where expressions of partial derivations are excerpted from (12). Linearization around $x_t$ is correction where we have the second expression for Jacobian matrix (11). Simulations in our specific case using constant drone altitude $z_0$.

Extended Kalman filter algorithm is defined in (7) through steps. 2nd and 3rd step is prediction while 4th, 5th, and 6th step are correction. The seventh step is resampling and iteratively trough all steps we aimed linearization.

$$\begin{aligned}
&1\text{: Extended Kalman filter algorithm } (\mu_{t-1},\Sigma_{t-1},u_t,z_t)\text{:}\\
&2\text{: } \overline{\mu_t} = g(u_t,u_{t-1})\\
&3\text{: } \overline{\Sigma_t} = G_t\Sigma_{t-1}G_t^T + R_t\\
&4\text{: } K_t = \Sigma_t H_t^T\left(H_t\Sigma_t H_t^T + Q_t\right)^{-1}\\
&5\text{: } \mu_t = \overline{\mu_t} + K_t\left(z_t - h\left(\overline{\mu_t}\right)\right)\\
&6\text{: } \Sigma_t = \left(I - K_t H_t\right)\overline{\Sigma_t}\\
&7\text{: return } \mu_t,\Sigma_t\\
&\overline{\mu_t} \to x_t
\end{aligned} \tag{7}$$

$$\mathbf{h} = \begin{bmatrix} \sqrt{(x_{1,t}-x_t)^2+(y_{1,t}-y_t)^2+(z_{1,t}-z_0)^2} \\ \sqrt{(x_{2,t}-x_t)^2+(y_{2,t}-y_t)^2+(z_{2,t}-z_0)^2} \\ \sqrt{(x_{3,t}-x_t)^2+(y_{3,t}-y_t)^2+(z_{3,t}-z_0)^2} \\ \sqrt{(x_{4,t}-x_t)^2+(y_{4,t}-y_t)^2+(z_{4,t}-z_0)^2} \\ \sqrt{(x_{5,t}-x_t)^2+(y_{5,t}-y_t)^2+(z_{5,t}-z_0)^2} \\ \sqrt{(x_{6,t}-x_t)^2+(y_{6,t}-y_t)^2+(z_{6,t}-z_0)^2} \end{bmatrix} \tag{8}$$

Initial conditions of EKF needs to be declared before applied EKF algorithm. Expression (14) is initial condition from (5), [20], where we have initial covariance matrix $\Sigma_0$, (13). For better understanding of initial covariance matrix we have defined shape of fields (15), (16)

and (17). Those initial conditions are arbitrary data. Initial covariance matrix of the first beacon looks like $\Sigma_{b_{1,0}}$, which represent accumulation point (convergence) from Particle filter [16–19].

$$\mathbf{g} = \begin{bmatrix} x_{t-1} + u_{t-1,x} \\ y_{t-1} + u_{t-1,y} \\ x_{1,t-1} \\ y_{1,t-1} \\ z_{1,t-1} \\ x_{2,t-1} \\ y_{2,t-1} \\ z_{2,t-1} \\ x_{3,t-1} \\ y_{3,t-1} \\ z_{3,t-1} \\ x_{4,t-1} \\ y_{4,t-1} \\ z_{4,t-1} \\ x_{5,t-1} \\ y_{5,t-1} \\ z_{5,t-1} \\ x_{6,t-1} \\ y_{6,t-1} \\ z_{6,t-1} \end{bmatrix} \tag{9}$$

$$G = \frac{\partial g}{\partial x_{t-1}} = I^{20x20} \tag{10}$$

$$\begin{aligned} h_i &= \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z_0)^2} \\ \frac{\partial h_i}{\partial x} &= \frac{-1}{\cancel{2}h_i} \cdot \cancel{2}(x_i - x) = -\frac{x_i - x}{h_i} \\ \frac{\partial h_i}{\partial y} &= -\frac{y_i - y}{h_i} \\ \frac{\partial h_i}{\partial x_i} &= \frac{x_i - x}{h_i} \\ \frac{\partial h_i}{\partial y_i} &= \frac{y_i - y}{h_i} \\ \frac{\partial h_i}{\partial z_i} &= \frac{z_i - z_0}{h_i} \end{aligned} \tag{12}$$

$$\Sigma_0 = \begin{bmatrix} \begin{bmatrix} 10^2 & 0 \\ 0 & 10^2 \end{bmatrix} & \overset{2x3}{\mathbf{0}} & \overset{2x3}{\mathbf{0}} & \overset{2x3}{\mathbf{0}} & \overset{2x3}{\mathbf{0}} & \overset{2x3}{\mathbf{0}} & \overset{2x3}{\mathbf{0}} \\ \overset{3x2}{\mathbf{0}} & \mathbf{\Sigma_{b_{1,0}}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \overset{3x2}{\mathbf{0}} & \mathbf{0} & \mathbf{\Sigma_{b_{2,0}}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \overset{3x2}{\mathbf{0}} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma_{b_{3,0}}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \overset{3x2}{\mathbf{0}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma_{b_{4,0}}} & \mathbf{0} & \mathbf{0} \\ \overset{3x2}{\mathbf{0}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma_{b_{5,0}}} & \mathbf{0} \\ \overset{3x2}{\mathbf{0}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma_{b_{6,0}}} \end{bmatrix} \tag{13}$$

Arbitrarily, initial conditions are defined with (13) and (14) where $p_i$ is position of $i$ particle. Constellations (18), (19) and (20) are defined as stabile results from Particle filter where we can see on Figure 9. a) different isometric view when Particle filter algorithm is applied. Recorded data from Particle filter simulation are saved to *.mat file and in parallel read from the same file with EKF code algorithm, Figure 10. Some examples in practice are not good to use this approach because of big error mistake during measurement in covariance of insecurity. Ensurance that we have the isolated scenario without errors, its additionally increased insecurity, Figure 9. b).

$$\boldsymbol{\mu_0} = \begin{bmatrix} x_0 \\ y_0 \\ x_{b_1,0} \\ y_{b_1,0} \\ z_{b_1,0} \\ x_{b_2,0} \\ y_{b_2,0} \\ z_{b_2,0} \\ x_{b_3,0} \\ y_{b_3,0} \\ z_{b_3,0} \\ x_{b_4,0} \\ y_{b_4,0} \\ z_{b_4,0} \\ x_{b_5,0} \\ y_{b_5,0} \\ z_{b_5,0} \\ x_{b_6,0} \\ y_{b_6,0} \\ z_{b_6,0} \end{bmatrix} \tag{14}$$

$$\overset{2x3}{\mathbf{0}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{15}$$

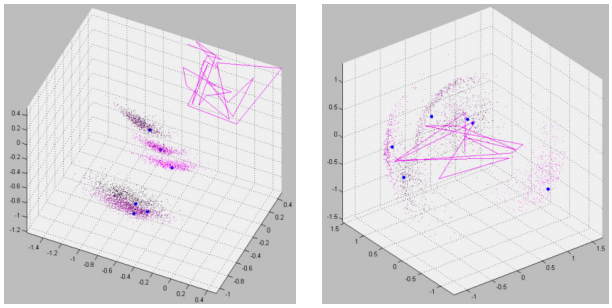
$$\mathbf{0} = \overset{3x3}{\mathbf{0}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{16}$$

$$\mathbf{\Sigma_{b_{i,0}}} = \begin{bmatrix} . & .. & ... \\ . & .. & ... \\ . & .. & ... \end{bmatrix} \tag{17}$$

$$p_1, p_2, \ldots \quad \ldots p_N \tag{18}$$

$$\bar{p} = \frac{\sum p_i}{N} \tag{19}$$

$$\Sigma = \sum_i \left(p_i - \bar{p}\right) \cdot \left(p_i - \bar{p}\right)^T \tag{20}$$



a) b)

**Figure 9. a) The end result of Particle filter (different isometric view), b) The end result of Particle filter with increased insecurity**

Measured noise covariance matrix is defined as (22), where (21) represent units (identity) matrix with ones on

the main diagonal and zeros elsewhere. Complete EKF algorithm is done in this way from mathematical model and its approached to writing Matlab code. Code, *.m file, Figure 11., is written in the steps as is defined in EKF algorithm (7). Simulation shows good results, Figure 10. where it is necessary to reduce insecurity caused by drone IMU gyroscope drift. Improvements can be done with integration of external odometry insecurity, which is presented with total insecurity added from optical flow.

$$\mathbf{V} = \frac{\partial g}{\partial u_{t-1}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{21}$$

$$\mathbf{R}_T = \mathbf{V} \cdot \mathbf{R}_t^{'} \cdot \mathbf{V}^T = \begin{bmatrix} \mathbf{R}_t^{'} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{22}$$

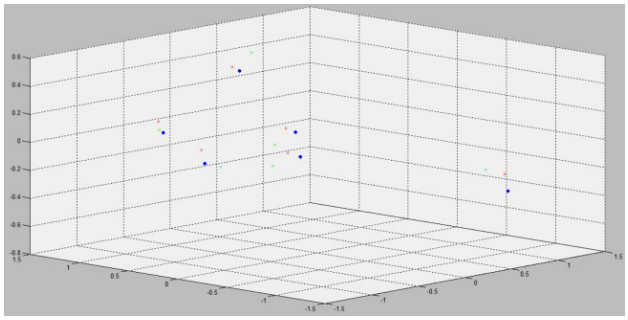


**Figure 10. Applied algorithm of extended Kalman filter**

## 4. OPTICAL FLOW AND ENERGY FUNCTION

Regular consistency from simulation, Figure 10. requires additional parameter to reduce noise error. This can be done by applying one of the optical flow methods. In introduction a couple of methods are mentioned, while is selected for implementation Lucas and Kanade [21]. Smoothness along edges is not considered since we used landmark in real-time scenario. Flow vector then can be better realized once when we have role oriented landmarks in the environment. Direction of gradient is scaled to the point of view where boxes of markings should be relatively large to make accurate measurement. Optical flow simulation in this case scenario is not simulated but will be integrated in real-time measurements. Workstation which is controlling drone from LabView, National Instruments Toolkit for ARDrone have only one WiFi for communication.

Integration of LK optical flow block requires to install additional WiFi adapter on workstation for integration of the video stream, which comes from WiFi camera on the top of the ceiling in indoor environment [22–30]. Representation of that kind of environment is defined with Figure 12.

```
close all; clear all; clc;
load cesticni_rez.mat
Q_t = sigma_b^2*eye(Nb,Nb);
%Rnd shifting of drone on square grid
m = 50;
nIter = 1;
W = [We';[1*(2*rand(2,m-1)-1);0.4*(2*rand(1,m-1)-1)]'];
z0 = We(end);
%Changed drone position
Z = zeros(Nb, 1);
U = zeros(2, 1);
Z0 = 0;
for quadp = 2:m
    %step 2:
    u_t = W(1:2,quadp) - W(1:2,quadp - 1) + sigma_IMU * randn(2,1);
    z0 = W(3,quadp) + sigma_IMU * randn(1,1);
   %Measurements simulation of distance between drone and beacon
    for i=1:Nb
        tmp = (W(:,quadp) - b(:,i)).^2;
        z_t(i,1) = sqrt(sum(tmp)) + sigma_b*randn(1,1);
    end
    U = [U, u_t];
    Z = [Z, z_t];
    Z0 = [Z0, z0];
end
for k = 1:nIter
mu_t_1 = mu_0;
Sigma_t_1 = Sigma_0;
for quadp = 2:m
    u_t = U(:,quadp);
    z_t = Z(:,quadp);
    z0 = Z0(quadp);
    mu_t_mean = mu_t_1 + [u_t;zeros(3*Nb,1)];
    %step3:
    G_t = eye(2+3*Nb,2+3*Nb);
    V = [1 0;0 1];
    V = [V; zeros(3*Nb,2)];
    Rtt = sigma_IMU^2*eye(2,2);
    R_t = V*Rtt*V';
    Sigma_t_mean = G_t*Sigma_t_1*G_t' + R_t;
        H_t = [];
        for i=1:Nb
            h(i,1) = sqrt(sum(([mu_t_mean(1:2,1); z0]-mu_t_mean((3+3*(i-1)):-->
            <--(5+3*(i-1)),1)).^2));
            H_t = [H_t; [-(  mu_t_mean((3+3*(i-1)),1)-mu_t_mean(1,1))/h(i,1)-->
            <-- -(mu_t_mean((4+3*(i-1)),1)-mu_t_mean(2,1))/h(i,1)]];
            temppp(i,:) = [zeros(1,3*(i-1))  (mu_t_mean((3+3*(i-1)),1)-->
            <-- -mu_t_mean(1,1))/h(i,1) (mu_t_mean((4+3*(i-1)),1)-->
            <-- -mu_t_mean(2,1))/h(i,1) (mu_t_mean((5+3*(i-1)),1)-z0)/h(i,1)-->
            <-- zeros(1,3*Nb-3*i)];
        end
        H_t = [H_t temppp];
        %step 4:
        K_t = Sigma_t_mean*H_t'*inv(H_t*Sigma_t_mean*H_t' + Q_t);   %Z0 x Nb
        %step 5:
        mu_t_mean = mu_t_mean + K_t*(z_t - h);
    %end
    mu_t = mu_t_mean;
    %step Nb:
    I=eye(2+Nb*3,2+Nb*3);
    Sigma_t = (I - K_t*H_t)*Sigma_t_mean;
    %step 7:
    mu_t_1 = mu_t;
    Sigma_t_1 = Sigma_t;
    %disp('proso')
end
figure(1);
for i=1:Nb
    plot3(b(1,i),b(2,i),b(3,i),'b o','MarkerSize',5, 'MarkerFaceColor','b');-->
    <--hold on;
    plot3(mu_0((3+3*(i-1)),1),mu_0((4+3*(i-1)),1),mu_0((5+3*(i-1)),1),'g x')
    plot3(mu_t((3+3*(i-1)),1),mu_t((4+3*(i-1)),1),mu_t((5+3*(i-1)),1),'r x')
end
hold off
E=0;
for i = 1:Nb;
    e=(mu_t_mean(3+3*(i-1):5+3*(i-1))-b(:,i));
    E=E+sqrt(e'*e);
end
E = E/Nb;
dW=[mu_t_mean(1:2); z0]-W(:,quadp);
E2=0;
for i = 1:Nb ;
    e=(mu_t_mean(3+3*(i-1):5+3*(i-1))-dW-b(:,i));
    E2=E2+sqrt(e'*e);
end
E2 = E2/Nb;
[E E2]
mu_0(3:2+3*Nb) = mu_t(3:2+3*Nb);
end
```

**Figure 11. Applied code algorithm of extended Kalman filter**

Drone IMU (gyroscope drift) can be compensated in three ways. The first method is to put ultrasonic transmitter (listener) out of the central point of drone. This method is not used but in this case optical flow is not necessary because orientation and position drift can be solved.

The second tested option was to install small WiFi router with the regular USB HD camera. Since that all hardware parts are heavy, measurements with installed camera on that way had lots of instability during the point of view tour.

The third tested option was shown as quality method where camera was mounted on the top of the ceiling. Defined trajectory in this test case scenario (just for testing optical flow) cover round from PoV1 (Point of View) to PoV4. Real time test was with a very accurate and precise orientation and position, where flow vector defines good gradient direction. During the first test are used smaller markers, while in the second test all four color marker squares are 100mm x 150mm. Detection of drone/listener is realized with a red marker 50 mm x 75mm on the top of the drone chassis. The accuracy of measurement with larger block of markers was very precise during square tour around points of view.

Energy function is considered as a future work with specifically defined locations of four beacons. The Distributed base is suggested to use Blockchain technology.

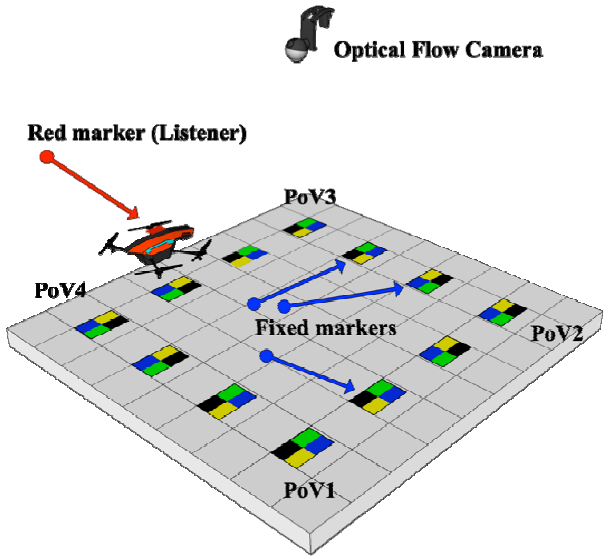



**Figure 12. Real-time test case scenario with optical flow – landmarks point of view**

### 4.1 Lucas and Kanade method of optical flow with landmarks

Using LabView as a tool for LK optical flow is simplified with a block named: "(LK) IMAQ Optical Flow", Figure 13. With applied block on the camera live view, it's computed optical flow (velocity flow), given from drone IMU in the shape of speed in abscissa and ordinate. Those positions are crucial to integrate in the future real-time scenario with PF and EKF. With ROI Descriptor we can define the region of interest within which the optical flow is computed. Current frame defines reference to the current source image, where previous frame is a previous source image. Velocity component "1" In, represents the reference to the image that will contain the horizontal component or the magnitude component of the computed velocity at each pixel. Velocity component "2" In, represent reference to the image that will contain the vertical component or the phase component of the computed velocity at each pixel. Window size is important to define because of the size of a rectangular block of pixels. In that case, its assumed to be moving with the same velocity in the image. The first option is to use a larger window size for smoother motion in the image and smaller window sizes for more turbulent motion.

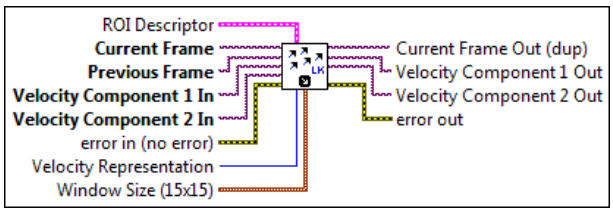



**Figure 13. Optical flow LK block**

Mathematical behaviour of LK optical flow is expressed through energy function in (23), where LK method is a very local method instead of HS method which is global. Optical flow as a process of matching local pixels, first defines small boxes of pixels. From energy term (24), for window or box is a general idea to minimise energy function. The assumption is that there is enough texture under a region of interest.

$$E_{LK}(u,v)=\sum_{(x,y)} w(x,y)\cdot\left(I(x+u,x+v,t+1)-I(x,y,t)\right)^2 \quad (23)$$

$$E_{LK}=\begin{bmatrix}\sum_w\left(\frac{\partial I}{\partial x}\right)^2 & \sum_w\left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right)\\ \sum_w\left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \sum_w\left(\frac{\partial I}{\partial y}\right)^2\end{bmatrix}\cdot\begin{bmatrix}u\\ v\end{bmatrix}=\begin{bmatrix}-\sum_w\frac{\partial I}{\partial x}\cdot\frac{\partial I}{\partial t}\\ -\sum_w\frac{\partial I}{\partial y}\cdot\frac{\partial I}{\partial t}\end{bmatrix} \quad (24)$$

Modifications from initial test scenario Figure 7. and Figure 12. are represented in 3D to 2D scenario, Figure 14. a), where six beacons are reduced to four beacons for computing energy function with a new algorithm, Figure 14. b).

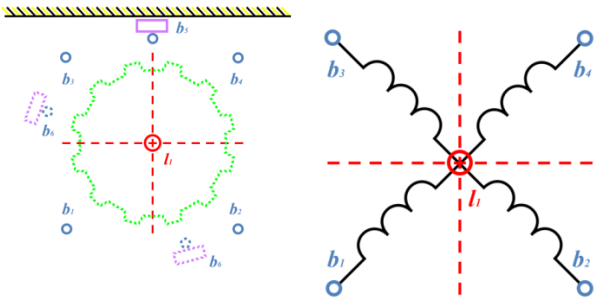



**Figure 14. a) Test case scenario of experiment (listener and beacons), b) Experimental scenario with new algorithm (listener and four beacons)**

### 4.2 Energy function

The role of "energy function" in generative models is the same as probability. It is as an indicator of the system in current configuration. In our specific test case scenario we have three levels of configuration from Particle filter, EKF and optical flow. Every configuration process of variables correspond to an energy value and the least it is the most possible the system (the most probability of the probability graph) is. The variables corresponding to the lowest energy is the most

likely configuration of the system, as well as the most stable state of the system. To find the most likely variables of the system is to find the most likely state of the graph when some variable is fixed or given, this procedure is predicting.

Applied on the main set of configuration process relations are in this shape:

$$\frac{\Delta E}{\Delta x}$$

$$E = \sum_i \left(\bar{x} - \overline{p_i}\right)^2$$

$$\frac{\partial E}{\partial \bar{x}} = 2 \cdot \Sigma \left(x - p_i\right)$$

## 5. CONCLUSION

The paper is oriented to previous work which covers hardware architecture used for the platform. With Crossbow Cricket platform, all measurements are realised with earlier mathematical model definition [31]. Used method is TDOA-RSS, where the listener is located on the drone and collects all data. Those data are parsed wirelessly to workstation, where with WSN (Wireless Sensor node – Network) is parsed to control drone application. The application is realised in LabView National instruments with all additional implemented algorithms. The first integration part is to use simulations from Matlab based on 3D data of Particle filter [20]. Increasing accuracy of the listener position is done with integration of EKF [17]. Confirmation in simulation is given in Figure 11., where implication in real-time environment be can obtained. Rejection of that kind of samples and in our case measured noise can be done with LK optical flow integration like a third step of the configuration process.

Real time measurements and computing in the case of optical flow was required to choose proper approach, where LK method of optical flow is used, because its locally oriented in small boxes of pixels.

Energy function is considered for implementation inside the main application like a new algorithm, where we can make generative model in the direction of probability.

Future work will be focused on data analysis, which needs to be confidential and secure. That kind of base should be decentralised, where insecurity is not present. In PhD thesis, as a future work, this paper will be expanded and additionally integrated with measured values. Those values will be recorded in Blockchain through nodes in dependency of the chain type [32]. In this way data can be recorded but also subsequently analysed in real-time.Upgrading of the presented system can be even more precisely done in a way of integration of a fuzzy-partition model element in a dynamic model, first as a simulation and then verification [33], [34].Classifications of the scientist's based on earlier defined area of research, Range-Only SLAM, are considered with high ranking index in specific area like indoor localization, outdoor localization with adequate subversions [35]. Configuration process has minimum three levels of code through our scenarios, where software reliability should follow smart/intelligent vehicles focus of complexity [36].

Future work is to create the general model of drone system in indoor environment with integration of SolidWork project and Matlab linearization [37].

## ACKNOWLEDGMENT

I would like to thank Prof.dr.sc. Robert Cupec (FERIT) for mathematical model behavior. Also big thanks to dr.sc. Andrej Kitanov (FER then, now Department of Aerospace Engineering Technion - Israel Institute of Technology), who arranged lend of equipment; Crossbow Cricket for testing. Thanks to Doc.dr.sc. Ratko Grbić (FERIT) who helped me with Matlab code parts like beacon parser in real time measurements. I would like to thank Reitz Rabie who started with Point of View approach with LabView markers and controlling ARDrone with toolkit.

## REFERENCES

[1] Drone Parrot ARDrone 2.0: http://www.parrot.com, accessed 1st September 2018.

[2] Šoštarić, D. and Mester, G.: Drone Localization using Ultrasonic TDOA and RSS Signal – Integration of Inverse Method of Particle Filter, Interdisciplinary Description of Complex Systems, 2019, *Accepted for publishing*.

[3] Šoštarić, D.: Modeling,Control and Navigation of an Autonomous Quad-Rotor Helicopter, Interdisciplinary Description of Complex Systems, Vol. 14, No. 3, pp. 322-330, ISSN 1334-4684, DOI: 10.7906/indecs.14.3.5, 2016.

[4] Šoštarić, D. and Mester, G.: ECG Simulation and Integration of Kalman Filter in CardioPediatric Cases, Interdisciplinary Description of Complex Systems, ISSN 1334-4684, DOI: 10.7906/indecs. 7.3.122019, *Accepted for publishing*.

[5] Kitanov, A., Tubin, V. and Petrović I.:*Extending functionality of RF Ultrasound positioning system with dead-reckoning to accurately determine mobile robot`s orientation*, IEEE Control Application, (CCA) & Intelligent Control, (ISIC), 2009.

[6] Eckert, J., German, R., Dressler, F.: *On autonomous indoor flights: High-quality real-time localization using low-cost sensors*, IEEE International Conference on Communications (ICC), 2012.

[7] Eckert, J., Dressler, F. and German, R.: *Real-time indoor localization support for four-rotor flying robots using sensor nodes*, IEEE International Workshop on Robotic and Sensors Environments (ROSE), 2009.

[8] Mester, G.: Backstepping Control for Hexa-Rotor Microcopter, Acta TechnicaCorviniensis – Bulletin of Engineering, Tome VIII, Fascicule 3 (July – September), ISSN 2067–3809, pp. 121-125, 2015.

[9] Mester, G.: *Modeling of Autonomous Hexa-Rotor Microcopter*, Proceedings of the IIIrnd International Conference and Workshop Mechatronics in

Practice and Education, MechEdu 2015, ISBN 978-86-918815-0-4, May 14-16, 2015, Subotica, Serbia, pp. 88-91.

[10] Rodic, A., Mester, G.: *Ambientally Aware Bi-Functional Ground-Aerial Robot-Sensor Networked System forRemote Environmental Surveillance and Monitoring Tasks*, Proceedings of the 55th ETRAN Conference, SectionRobotics, Vol. RO2.5, ISBN 978-86-80509-66-2, Banja Vrućica, Bosnia and Herzegovina, Jun 6-9, 2011, pp. 1-4.

[11] Priyantha, N. B., Chakraborty, A. Balakrishnan, H..:*The Cricket Location-Support System*, 6th ACM International Conference on Mobile Computing and Networking (ACM MOBICOM), 2000.

[12] Newman, P., Cole, D. and Ho, K.: *Outdoor SLAM using visual appearance and laser ranging*, IEEE International Conference on Robotics and Automation (ICRA), 2006.

[13] Hahnel, D., Burgard, W., Fox, D. and Thrun S.: *An Efficient Fast SLAM Algorithm for Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range Measurements*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems,Vol.1, 2003.

[14] Kitanov, A., Cupec, R. and Petrovic, I.: Fast planar surface 3D SLAM using LIDAR, RoboticsandAutonomous Systems 92:197-220, (RAS), Elsevier Journal, June 2017, pp. 197-220.

[15] Ristic, B., Arulampalam, S. and Gordon N.: *Beyond the Kalman Filter – Particle Filters for Tracking Applications*, Library of Congress Cataloging-in-Publication Data, British Library Cataloguing in Publication Data, Book: DSTO, 2004.

[16] Thrun, S., Burgard, W. and Fox D.: *Probabilistic Robotics*, Early draft 1999-2000.

[17] Chen, M., Cheng, F. and Gudavalli, R.: *Precision and Accuracy in an Indoor Localization System*,

DC294-1/2 Course Project, 2003.

[18] Chen, Z.: *Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond*, Manuscript – Technical report, McMaster University, 2003.

[19] Dale, A. I.: *A History of Inverse Probability: From Thomas Bayes to Karl Pearson*, New York: Springer-Verlag, ISBN:9780387878683, 1991.

[20] Kasac, J., Milic, V., Stepanic, J. and Mester, G.:*A computational approach to parameter identi-fication of spatially distributed nonlinear systems with unknown initial conditions*,Proceedings of the IEEE Symposium on Robotic Intelligence in Informationally Structured Space RiiSS (2014), DOI: 10.1109/RIISS.2014.7009170, Vol. 1, ISBN: 978147 9944637, December 9-12, 2014, Orlando, Florida, USA, pp. 55-61.

[21] Babayan, P., Buiko, S., Vdovkin, L., Ershov, D.M., Muraviev, S.V., Sirenko, A. and Smirnov, S.: *Real-time pyramidal Lukas-Kanade tracker performance estimation,*Real-Time Image Processing and Deep Learning, 2019.

[22] Mester, G. and Rodic, A.: Sensor-Based Intelligent Mobile Robot Navigation in Unknown Environments, International Journal of Electrical and Computer Engineering Systems, Vol. 1, Issue No. 2, 1-8, ISSN: 1847-6996, 2010.

[23] Rodic, A. and Mester, G.: The Modeling and Simulation of an Autonomous Quad-Rotor Microcopter in a Virtual Outdoor Scenario, Acta PolytechnicaHungarica, Journal of Applied Sciences, Vol. 8, Issue No. 4, 107-122, ISSN 1785-8860, 2011.

[24] Rodic, A., Mester, G., Stojković, I., Qualitative Evaluation of Flight Controller Performances for Autonomous Quadrotors, Intelligent Systems: Models and Applications, Endre Pap edit., Topics in Intelligent Engineering and Informatics, Vol. 3, Part. 2, ISBN 978-3-642-33958-5, DOI 10.1007/978-3-642-33959-2_7, Springer-Verlag, Berlin Heidel-berg, pp. 115-134, 2013.

[25] Mester, G., Rodic, A.: *Simulation of Quad-rotor Flight Dynamics for the Analysis of Control, Spatial Navigation and Obstacle Avoidance*, Proceedings of the 3rd International Workshop on Advanced Computational Intelligence and Intelligent Informatics, IWACIII 2013, ISSN: 2185-758X, October 18 to 21 in 2013, Shanghai, China, pp. 1-4.

[26] Rodic, A., Mester, G.: *Control of a Quadrotor Flight*, Proceedings of the ICIST Conference, ISBN: 978-86-85525-12-4, 03-06.03.2013, Kopaonik, Serbia, pp. 61-66.

[27] Stepanic, J., Mester, G., Kasac, J.: *Synthetic Inertial Navigation Systems: Case Study of Determining Direction*, Proceedings of 57th ETRAN Conference, Zlatibor, Serbia, June 3-6, 2013, pp. RO 2.7.1-3.

[28] Mester, G., Rodic, A.: Modeling and Navigation of an Autonomous Quad-Rotor Helicopter, E-society Journal: Research and Applications, Vol. 3, No. 1, ISSN 2217-3269, pp. 45-53, July 2012.

[29] Mester, G., Rodic, A.: *Navigation of an Autonomous Outdoor Quadrotor Helicopter*, Proceedings of the 2nd International Conference on Internet Society Technologie and Management, ICIST, ISBN: 978-86-85525-10-0, 01-03.03.2012, Kopaonik, Serbia, pp. 259-262.

[30] Rodic, A., Mester, G.: *Modeling and Simulation of Quad-Rotor Dynamics and Spatial Navigation*, Proceedings of the 9th IEEE International Symposium on Intelligent Systems and Informatics, SISY 2011, ISBN: 978-1-4577-1973-8, DOI:10.1109/SISY.2011.6034325, 8–10 September 2011, Subotica, Serbia, pp. 23-28.

[31] Moravek, P., Komosny, D., Simek, M. and Girbau, D.: Measurement with the Cricket localization system, ElektrorevueEng – Časopis Pro Elektrotechniku, 2011.

[32] Nakamoto, S.: *Bitcoin: A Peer-to-Peer Electronic Cash System,* Whitepaper, October 2008. http: //bitcoin.org accessed 1st September 2018.

[33] Nemes, A., Mester, G.:Unconstrained Evolutionary and Gradient Descent-Based Tuning of Fuzzy-partitions for UAV Dynamic Modeling, FME Transactions, ISSN: 1451-2092, DOI: 10.5937/fmet1701001N, Vol. 45, No. 1, pp. 1-8, 2017.

[34] Nemes A., Mester G.:*Energy Efficient Feasible Autonomous Multi-Rotor Unmanned Aerial Vehicles Trajectories*, Proceedings of the 4th International Scientific Conference on Advances in Mechanical, Engineering, ISCAME 2016, ISBN 978-963-473-944-9, Debrecen, Hungary, 13-15 October 2016, pp. 369-376.

[35] Mester, G.: Rankings Scientists, Journals and Countries Using h-index, Interdisciplinary Description of Complex Systems, Vol. 14, No. 1, ISSN 1334-4684, DOI: 10.7906/indecs.14.1.1, pp. 1-9, 2016.

[36] Gy. Schuster, D. Tokody, I.J. Mezei, "*Software reliability of complex systems focus for intelligent vehicles*" In: K. Jármai, B. Bolló (eds) Vehicle and Automotive Engineering. Lecture Notes in Mechanical Engineering, 2017. ISSN: 2195-4356, Print ISBN 978-3-319-51188-7, pp. 309–321., Springer, Cham, DOI 10.1007/978-3-319-51189-4_28

[37] Stevanović, I., Rašuo, B.: Development of a Miniature Robot Based on Experience Inspired by Nature, FME Transactions, Volume 45 No 1, pp. 189-197, 2017.

## КОПТЕР SLAM КОРИШТЕЊЕМ TDOA-RSS СИГНАЛА СА ПРИМЈЕЊЕНИМ EKF НА PF ПОДАЦИМА

**Д. Шоштарић, Ђ. Мештер**

Овај рад описује интеграцију података честичног филтра раније снимљено у симулацији и тестирано у стварно-временском сценарију. Такви подаци су парсирани из примопредајног уређаја коришћењем TDOA-RSS инверзнеметоде. Коптер SLAM нема линеарну позадину и за бољу прецизност је примјењен EKF. IMU и одометрија камере са оптичким током представља улазни параметар за EKF блок. Почетни услови су фокусирани на KF од IMU коптера. Одометрија моновизијске камере је кориштена за оптички ток на укупној ковариjанци несигурности. За побољшање оптичког тока у ковариjанци несигурности, коришћене су ознаке на поду. Специфични тест сценарио од EKF је представљен у раду са коптер/примопредајником и примопредајним чворовима на позицијама које припада Range-Only SLAM. Нелинеарни математички изрази су представљени између прошле позиције коптера и контролног сигнала. Почетни услови су промјењиви док је коптер висина фиксна. Енергетска функција је размотрена да се имплементира у енергетски оптимизирани сценарио.

**Extended matrix equations: G (10) and H (11).**

$$G=\frac{\partial g}{\partial x_{t-1}}=I^{20x20}=\begin{bmatrix}1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix} \quad (10)$$

$$\mathbf{H} = \frac{\partial h}{\partial x} = \begin{bmatrix} \frac{\partial h_1}{\partial x} & \frac{\partial h_1}{\partial y} & \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial y_1} & \frac{\partial h_1}{\partial z_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\partial h_2}{\partial x} & \frac{\partial h_2}{\partial y} & 0 & 0 & 0 & \frac{\partial h_2}{\partial x_2} & \frac{\partial h_2}{\partial y_2} & \frac{\partial h_2}{\partial z_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\partial h_3}{\partial x} & \frac{\partial h_3}{\partial y} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial h_3}{\partial x_3} & \frac{\partial h_3}{\partial y_3} & \frac{\partial h_3}{\partial z_3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\partial h_4}{\partial x} & \frac{\partial h_4}{\partial y} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial h_4}{\partial x_4} & \frac{\partial h_4}{\partial y_4} & \frac{\partial h_4}{\partial z_4} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\partial h_5}{\partial x} & \frac{\partial h_5}{\partial y} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial h_5}{\partial x_5} & \frac{\partial h_5}{\partial y_5} & \frac{\partial h_5}{\partial z_5} & 0 & 0 & 0 \\ \frac{\partial h_6}{\partial x} & \frac{\partial h_6}{\partial y} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial h_6}{\partial x_6} & \frac{\partial h_6}{\partial y_6} & \frac{\partial h_6}{\partial z_6} \end{bmatrix} \tag{11}$$