VERSITA
Applied Computer Systems
2013/ 14
doi: 10.2478/acss-2013-0006

Single Robot Localisation Approach for Indoor Robotic Systems through Integration of Odometry and Artificial Landmarks

Agris Nikitenko[1], Aleksis Liekna[2], Martins Ekmanis[3], Guntis Kulikovskis[4], Ilze Andersone[5] [1-5]*Riga Technical University, Faculty of Computer Science and Information Technology, Institute of Applied Computer Systems*

***Abstract* – we present an integrated approach for robot localization that allows to integrate for the artificial landmark localization data with odometric sensors and signal transfer function data to provide means for different practical application scenarios. The sensor data fusion deals with asynchronous sensor data using inverse Laplace transform. We demonstrate a simulation software system that ensures smooth integration of the odometry-based and signal transfer – based localization into one approach.**

***Keywords* – robotics, single robot systems, robot localization**

I. INTRODUCTION

Localization is one of the most important issues that have to be solved in mobile robotic platforms today. If the robot operates autonomously its pose estimation is extremely important in order to avoid obstacles, plan actions properly, follow planned trajectories or complete other specific tasks.

In this paper we propose an integrated indoor robot localization approach that is based on use of artificial landmarks that can be visually tracked by robots. Unfortunately, as it is explained later, not always the landmarks are within the sight of sensors. Therefore we propose to combine the landmark-based localization with signal transfer-based and wheel encoders-based localization method [1,2]. This allows overcoming limitations of the methods while used alone. Throughout the paper we assume all of the incoming data being noisy and normally distributed around its mean value. This important assumption allows estimating robot pose and its variance using kinematic model of the robot.

Input data is provided by robot cameras and odometric sensors – wheel encoders. For practical implementation we use iRobot Roomba560 vacuum cleaning robots that are complimentary equipped with Intel Atom CPU (*Central processing unit*) based embedded computing node as well as WiFi (*Wireless network trademark*) and web camera [3].

The rest of the paper is organized as follows: Section II outlines the addressed problem, Section III describes related work in fuse estimations from different data sources, Section IV describes the technique used for robot localization based on artificial landmarks mounted on the ceiling, Section V describes use of other sensors and their error estimation through the algebra of random variables, Section V proposes to use signal response function as the tool for robot pose estimation and forecasting without using sensors, Section VI describes use of multidimensional signal response function to reduce the complexity of calculations and error estimation, Section VII demonstrates use of the proposed indoor localization method while Section VIII gives conclusions and insight of future research goals.

II. PROBLEM STATEMENT

We assume that the robotic system will be used indoors only and the environment itself is available for necessary minor modifications in order to provide infrastructure for robots i.e., it is possible and feasible to install some special markings that provide significant information to robots about their pose and translations within environment. However it is important to emphasize that these data sources are not always available to the robot due to environment configurations as depicted in figure 1.

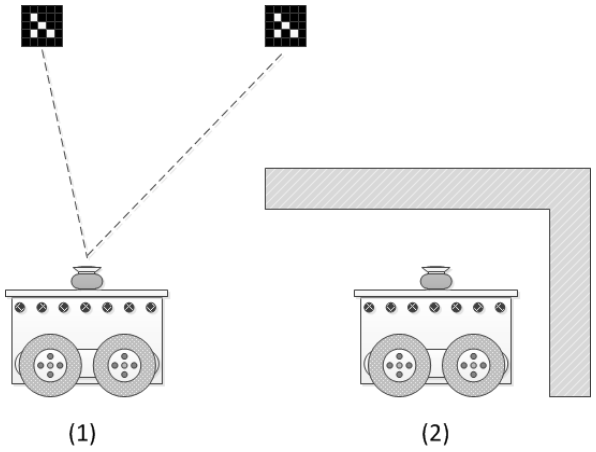

Fig. 1 Robot and landmarks possible positions: (1) – landmark can be detected by sensors and (2) landmark is out of sensors sight

Therefore, another important source of pose and motion information is a set of odometric sensors. In case of Roomba each of the driving wheels is equipped with rotary encoders, which data are available via Roomba serial interface [4].

If more than one information source is used for the same estimation, information from all sources must be fused into a single measurement to provide position estimation [1,2]. Due to their specifics, data from some sensors are available at one moment of time while data from others are delayed. The delays are mainly caused by communication latencies and specifics of the sensors used like maximum refresh rate. It

means that the sensor data fusion should be addressed by appropriate sensor data models.

Another important source of information that might be used for pose estimation is robot mutual collisions where pose estimations from robots involved in the collision can be fused into single estimation. This final estimation than can then be used for individual pose estimation error correction.

Therefore, in our case there are at least three possible pose estimation data sources that have to be fused into a single estimation, which is passed to Kalman filter (*KF further in the text)* or another similar algorithm for noisy time series processing [5]. Only then the estimation is passed to the robot or another system that need this estimation.

Schematically it is depicted in Fig. 2:

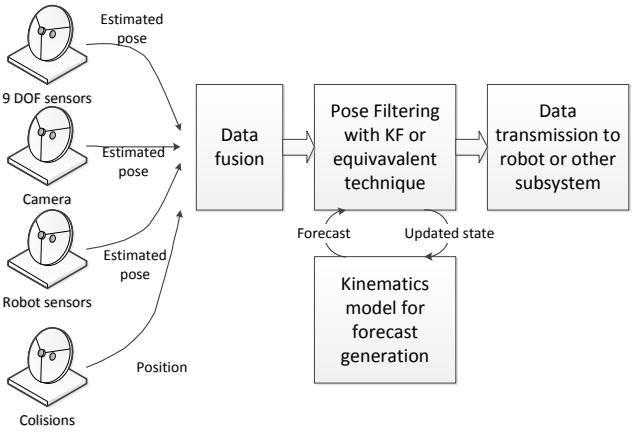

Fig 2. Pose estimation schema

In this paper we focus on data fusion and forecast steps that skip discussion on KF or equivalent techniques and collisions data processing.

Since the pose estimation is a common problem in mobile robotics, some related work has to be analysed.

III. RELATED WORK

In general, having two noisy measurements received from sensors, where each of them is described by its mean μ value and variance σ, i.e. assuming that the measurements are Gaussian variables, we can describe each of the measurements by a Gaussian distribution function of the form [6]:

$$G\ y,\mu,\sigma^2\ = \frac{1}{\sigma\ \sqrt{2\pi}}\exp\ -\frac{(y-\mu)^2}{2\sigma^2} \tag{1}$$

The merging procedure in this case is rather straightforward because of the feature of the Gaussians i.e. the product of the two Gaussians is Gaussian [6].

$$_i\, G(y,\mu_i,\sigma_i^2) \propto G\ y,\mu,\sigma^2\ , \tag{2}$$

where

$$\frac{1}{\sigma^2} = \ _i\ \frac{1}{\sigma_i^2};\quad \frac{\mu}{\sigma^2} = \ _i\ \frac{\mu_i}{\sigma_i^2}$$

This useful feature of Gaussian representations in used in various applications including sensor data fusion. A good example is PGM (*Abbreviation given by authors*) algorithm that employs k-nearest neighbours approach to select the closest incoming measurements in order to combine them pair-wisely in order to produce the best estimate and cluster them using a cluster threshold [6]. This approach is very efficient in terms of saving computing power and effective for use in embedded application but unfortunately does not provide the means for multi-dimensional estimation fusion. However the idea behind the PGM is the assumption that more than one independent data source for the same estimation can provide more accurate final estimation.

In [7] the multidimensional measurements are described by appropriate multidimensional Gaussian distributions with mean values - centre corresponding to estimated value and standard deviation along axes (x and y in this case) correspond to estimated noise of the measurement. Therefore the noise is described by the appropriate matrix where squared standard deviations initially are determined from each dimension axes in the local coordinates frame and aligned on the main diagonally. For the two-dimensional case with two measurements 1 and 2 the matrix for each of the measurements is formed as depicted in equation (3) [7].

$$C = \begin{matrix} \sigma_y^2 & 0 \\ 0 & \sigma_x^2 \end{matrix} \tag{3}$$

The final estimation is accomplished by combining covariance matrixes and estimation mean values like as outlined in the equations (4) and (5) [7].

$$C = C_1 - C_1\ C_1 + C_2\ ^{-1} C_1 \tag{4}$$

$$X = X_1 + C_1\ C_1 + C_2\ ^{-1}\ X_2 - X_1 \tag{5}$$

Here X_i represents mean values of i-th measurement, but C_i represents its noise covariance.

Finally, if the local coordinate frame is rotated in respect to the global coordinate frame then appropriate rotation R(θ) to covariance matrix is applied[7]:

$$C_f = R\ \theta\ ^T C R(\theta) \tag{6}$$

A slightly different approach is proposed by [8], where a *Simple Convex Combination* algorithm is used. In this case the final estimation is calculated as follows [8]:

$$X = C_{err}\ C_1^{-1} X_1 + C_2^{-1} X_2 \tag{7}$$

Here the noise covariance matrix C_{err} is calculated by addition and inversion of two noise covariance matrixes [8]:

$$C_{err} = \ C_1^{-1} + C_2^{-1}\ ^{-1} \tag{8}$$

Both methods assume that measurements are independent of each other. In case they are not independent the calculations are significantly more complex due to cross correlation matrixes calculation [1]. However in our application case we assume that all measurements are independent.

While both methods are presented as very effective by the authors they do not correspond directly to our particular case. The first method presented in [7] is used for robot football, where every robot acts as a separate sensor. The ball in this case is the tracked object. In general, mathematically there is no difference whether a single robot tracks multiple landmarks or multiple robots track a single object because the error sources are related to the multiple landmarks in one case and to multiple sensors in another case.

The method presented in [8] uses multiple stationary installed sensors that track a single robot. Again, here the problem statement in general is the same. The most significant difference between both approaches is sensor covariance matrices that initially might be different for each of the sensors. Similarly we initialize covariance matrices but all of them are equivalent because each of the landmark is recognized with the same sensor – web camera – causing the same sensed data variations.

Another important issue that has to be considered is time delayed incoming data. Unfortunately, none of the presented methods provide means for delayed data. As presented in [1,2] each of the sensors is modelled by using their time dependent models to estimate their values until the predefined time instants where all of the sensor data is fused into a single estimation.

Because of the error being time dependent, i.e. the longer time interval the larger standard deviation or variance of the measurement, both mean value and error estimation time models have to be considered.

IV. Use of Landmarks

Our approach is based on the assumption that indoor environment is available for landmark use – it means that it is possible to install specially designed landmarks for robot positioning in global coordinates, thereby in comparison with the landmarks that can provide pose and position data only in local coordinates frame decreasing positioning complexity and increasing pose estimation accuracy.

Use of artificial landmarks or dedicated robot markings is not new in robotics. In [9] robot pose estimation is based on T shape marking on top of the robots. A single camera installed on robot arena ceiling and the appropriate markings recognitions algorithm is used to estimate the poses and headings. In [10] a "cat paw" markings are proposed that allow to recognize robots pose and heading in the same way as it is done in [9] by ceiling-mounted camera.

Some earlier scientific efforts presented in [11] propose to use rotation variant landmarks in the form of circles with black borders of different thickness. The proposed method allows calculating the actual robot disposition relative to the landmark via calculating deformation of the landmark image in landmarks coordinates frame at various viewing angles, i.e. the circle transforms into ellipse. In contrast, the method presented in [12] defines landmarks as rotation variants and uses colour coding thus enabling robot pose estimation with a single omnidirectional camera.

We propose to use unique (in terms of similar landmarks within the same area of robot operation) rotation variant landmarks – glyphs (see Fig. 3) installed on the ceiling at known positions thereby providing useful information about the position and pose of the robot when appropriate landmark is noticed and recognized [13].

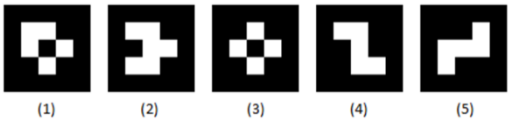

Fig 3. Landmark examples: (1) and (2) rotation variants while (3) – (5) are rotation invariants [13].

As it is depicted in Fig.3, a landmark is a set of white and black squares aligned in a way that they can produce image that in fact is a 2D binary representation. We use this feature to ensure that in a single frame no two similar glyphs can even appear.

Initially the frame produced by the on-board web camera is used to recognise those glyphs that are in the "area of sight", where every recognized glyph is represented by its centre and polar coordinates - relative to the frame geometric centre, i.e. distance **R** and angle **θ** (see Fig. 4).

As emphasised in [14], before using cameras for object recognition it is recommended to use appropriate distortion elimination algorithms to eliminate distortions caused by wide angle cameras lenses. In our case the used cameras are with rather narrow angle of view and therefore the actual distortions are not significant for glyph position estimation.

Having the recognised glyphs and knowing either dimensions of the used glyphs or distance from camera to the glyph centre, when both are aligned on the same line, which is orthogonal to the ceiling, it is possible to express every pixel of the image in mm that corresponds to the actual distance represented by the pixel.

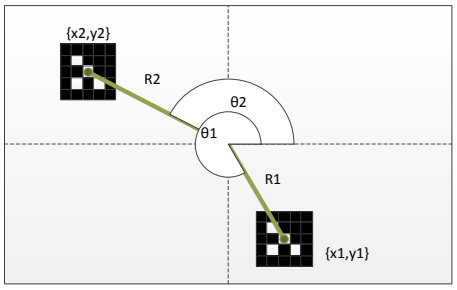

Fig 4. Landmark representations in web camera frame.

With this correspondence it is possible to calculate the actual disposition of the robot from a particular glyph. If each of the glyphs is installed at some known position in global coordinate frame, then every one of them will provide means for robot position estimations in global coordinate frame.

It means that every recognised glyph provides robot position (x,y), its heading (θ) and noise. In this case only one sensor is used, but due to minor impact of lens distortion, camera resolution and recognition algorithm approximations and applied filtering techniques [13] the noise has to be

considered. Using the notion introduced in previous section the calculated state is represented by multidimensional Gaussian, where noise is described by covariance matrix (9) that is associated with the appropriate i-th glyph state (10). Initially all matrixes are the same due to the use of single sensor.

$$C_i = \begin{matrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\theta^2 \end{matrix} \quad (9)$$

$$X_i = \begin{matrix} x_i & y_i & \theta_i \end{matrix}^T \quad (10)$$

The actual values of standard deviations for x,y and θ components of the state are determined experimentally based on multiple robot runs and actual disposition measurements.

For calculation of the final estimation by using multiple estimations they are fused by employing the proposed technique [7], mainly due to its simplicity of implementation. This approach is justified by the fact that all of the estimations corresponding to individual glyphs are simultaneously available. Because of controlled glyph installation all of the glyphs are oriented in one direction, which enables to eliminate rotation calculations as presented in [7]. If only one glyph is observed then fusion is not required and initial covariance matrix of the estimation is used. During the initial tests the glyph data allows to localise the robot with rather high accuracy within few centimetres. The most significant feature of this approach is constant variance that is not growing over time.

V. USE OF OTHER SENSORS

As mentioned before, an important source of robot position information are wheel encoders. Unfortunately due to data interface delays of the used robotic platforms actual data from encoders is not available at present time instants. This situation is depicted in Fig 5, where T_i represents i-th time interval when the robot position is being estimated. In our case, for interval reference we use camera frame processing interval that is constant. D_i represents i-th incoming data for one sensor packet. Δt_i represents varying time interval caused by data transfer delays, which actual length is known only at i-th time instant.

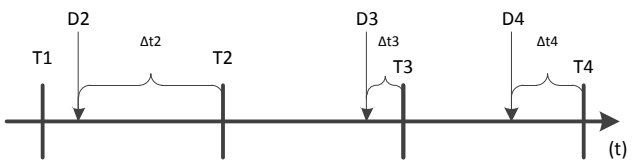

Fig 5. Sensor data availability relative to predefined time intervals of position and pose estimation

While wheel encoder values play significant role in position and pose estimation in situations when landmarks are not observed, it is necessary to estimate the actual value of the encoders having previous observation at a time $T_i - \Delta t_i$. Due to significance of the data, simple linear models have to be replaced by more tailored models to the actual robotic platform. We propose to use signal response function instead of the offered by Linear time-invariant system theory [15]. The actual mechanism behind that is based on Laplace transformation of functions from time domain to Laplace domain. By definition Laplace transformation F(s) of time function f(t) is given by integral (11) [15]:

$$\mathcal{L}\{f(t)\} = F(s) = \int_{-\infty}^{\infty} f(t)e^{-st}dt, \quad (11)$$

where $\mathcal{L}\{f(t)\}$ – indicates Laplace transform of the time function f(t), s – complex Laplace variable of the form $s = \sigma + j\omega$, F(s) – the transformed function in Laplace domain.

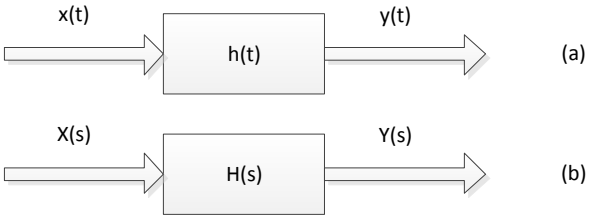

Fig 6. Signal response in time domain (a) and in Laplace domain (b) [15].

As shown in figure Fig 6, the system response is defined by input signals x(t) in time domain or X(s) in Laplace domain, impulse response h(t) or signal transfer functions H(s) and output signals y(t) and Y(s) in corresponding domains.

In Laplace domain, signal response is defined by the ratio between output Y(s) and input X(s), while in time domain the multiplication corresponds to convolution [15]. This correspondence is indicated in equation (12):

$$y(t) = h(t) * x(t) \Rightarrow Y(s) = H(s)X(s) \quad (12)$$

The signal transfer function H(s) is acquired by using (11) and replacing f(t) by h(t). The convolution itself is defined by the equation (13) [15]:

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \quad (13)$$

For modelling the input signal x we use a simple step function that fully corresponds to the incoming robot control signals.

Use of the impulse function in practice is difficult as it is infinitely short. Instead, we can use step function, which is defined by integral of impulse function.

$$u(t) = \int_{-\infty}^{t} \delta(t)dt, \; u'(t) = \delta(t) \quad (14)$$

As shown in (14), the derivative of the step function is an impulse at the time instant t. This allows to transform the equation (13):

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} \delta(\tau)h(t-\tau)d\tau \quad (15)$$

Having expression (15), to get the actual output it is necessary to incorporate the observed values at previous observations by applying another convolution step. While there is no prior information about the control signals available we model the input impulses $\delta(\tau)$ as constant during interval Δt_i.

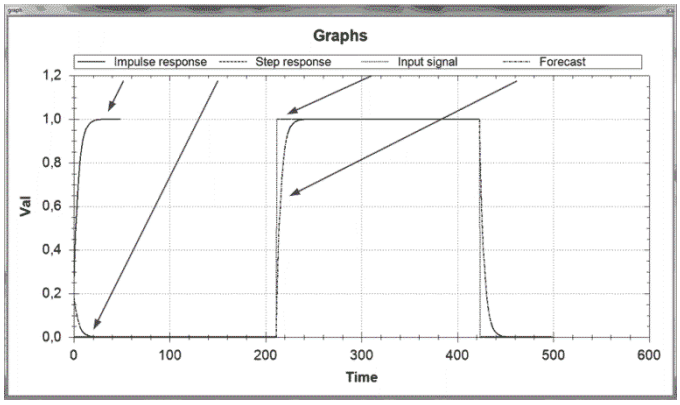

Fig 7. Simulated system response and output forecast without noise.

Figure Fig.7 presents a simulated output forecast having a particular impulse response function and input. Step response represents the derivative of the response function that is actually used in calculations. Input signal represents a typical trapezoidal shape of the control signal for robot manoeuvring.

The actual signal response function has to be acquired experimentally due to technical specification of the robot, its imperfection and environmental constraints, such as ground cover, traction of the surface, etc. Another issue is robot's response latency to the input signal that also differs from robot to robot and therefore has to be tailored experimentally to the particular robot. The response function has to be extracted from a series of experimental raw data and filtered to smooth the noise and reduce possible forecast error.

The abovementioned steps of response function acquisition are repeated for each odometric sensor – in our case for both of two wheels. The input signal is control signal in the form of PWM (*Pulse wide modulated signal*) coded by floating point value normalized to a unit (see Fig 7). Direct wheel speed measured by encoders is used as the output . Thus, the output of response function can be translated to robot's actual speed in m/s.

However, it is necessary to emphasise that while the actual measurements have errors they also have to be estimated over the time. It means that their values – values of variances have to be modelled with the help of time dependant models. In our case we use the linear model:

$$\sigma^2 = \sigma_t^2 + \sigma_t^2 \Delta t \tag{16}$$

In (16) σ_t^2 represents variance estimated at the time instant t and Δt represents varying time interval caused by data transfer delays as depicted in Fig.5. In order to calculate the final estimation error the variance σ^2 has to be estimated through kinematic calculations.

The assumed kinematic model of the differential drive robot is depicted in Fig. 8.

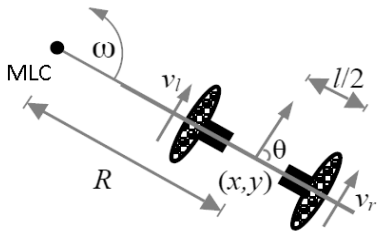

Fig 8. Differential drive robot model

The main parameters of the robot motion are disposition of the robot in polar coordinates:

$$R = \frac{l}{2}\frac{(v_r+v_l)}{(v_r-v_l)}, \quad \sim N(\mu_R, \sigma_R^2) \tag{17}$$

$$\omega = \frac{(v_r-v_l)}{l}, \quad \sim N(\mu_\omega, \sigma_\omega^2) \tag{18}$$

In equations (17) and (18) R – distance from the robot's mass centre to instantaneous rotation centre MLC, v_l and v_r – robot left and right side wheel linear speeds characterised by normal distributions $N(v_l,\sigma_l^2)$ and $N(v_r,\sigma_r^2)$, (x,y) – position of the robot's mass centre in global coordinates frame and l – distance between wheel centres.

$$MLC = MLC_x, MLC_y = x - R\sin\theta, y + R\cos\theta \tag{19}$$

The overall kinematic model is depicted in (20) [16].

$$\begin{matrix} x' \\ y' \\ \theta' \end{matrix} = \begin{matrix} \cos(\omega\Delta t) & -\sin(\omega\Delta t) & 0 \\ \sin(\omega\Delta t) & \cos(\omega\Delta t) & 0 \\ 0 & 0 & 1 \end{matrix} \begin{matrix} x - MLC_x \\ y - MLC_y \\ \theta \end{matrix} + \begin{matrix} MLC_x \\ MLC_y \\ \omega\Delta t \end{matrix} \tag{20}$$

Using this model over time makes it possible to estimate the robot's position expressed in form (9) and (10). With the help of random variable algebra it is rather easy to calculate σ_ω^2 variance of angular speed, which according to (18) is linear transformation of normally distributed random variable. Therefore:

$$\sigma_\omega^2 = \left(\frac{1}{l}\right)^2 \sigma_l^2 + \sigma_r^2 \tag{21}$$

Estimation of σ_R^2 is more challenging because ration of the two normally distributed variables is Cauchy distribution that has limited practical application. In [17] an approximated approach (see (22) and (23)) is proposed that allows to accept R as normally distributed random variable under certain constraints:

$$\mu_R \cong \frac{l}{2}\frac{v_r+v_l}{v_r-v_l} + \frac{\sigma_{r-l}^2(v_r+v_l)}{v_r-v_l^{\,3}} - \frac{\rho\sigma_{r-l}\sigma_{r+l}}{v_r-v_l^{\,2}} \tag{22}$$

$$\sigma_R^2 \cong \left(\frac{l}{2}\right)^2 \frac{\sigma_{r-l}^2\,(v_r+v_l)^2}{v_r-v_l^{\,4}} + \frac{\sigma_{r+l}^2}{v_r-v_l^{\,2}} - 2\frac{\rho\sigma_{r-l}\sigma_{r+l}\,v_r+v_l}{v_r-v_l^{\,3}} \tag{23}$$

In (22) and (23) according to random variable algebra $\sigma_{r+l}^2 = \sigma_l^2 + \sigma_r^2$, $\sigma_{r-l}^2 = \sigma_l^2 + \sigma_r^2$. For calculation of variance of MLC it is necessary to combine several transformations of variances:

$$\sigma_{MLC}^2 = \begin{matrix} \sigma_x^2 + \overline{\frac{\sigma_R^2\sigma_\theta^2 cos^2\theta}{\sigma_R^2+\sigma_\theta^2 cos^2\theta}} \\ \sigma_y^2 + \overline{\frac{\sigma_R^2\sigma_\theta^2 sin^2\theta}{\sigma_R^2+\sigma_\theta^2 sin^2\theta}} \end{matrix} \quad (24)$$

$$\mu_{MLC} = \begin{matrix} x - \frac{R\sigma_\theta^2 cos^2\theta + sin\theta\sigma_R^2}{\sigma_R^2+\sigma_\theta^2 cos^2\theta} \\ y + \frac{R\sigma_\theta^2 sin^2\theta + cos\theta\sigma_R^2}{\sigma_R^2+\sigma_\theta^2 sin^2\theta} \end{matrix} \quad (25)$$

In (24) σ_x^2, σ_y^2 and σ_θ^2 are determined from previous time intervals.

In order to estimate variance of the new pose of the robot after time interval Δt, it is necessary to replace variance of the term ωΔt in (20) by the used time model (16) that results in:

$$\sigma_{\omega\Delta t}^2 = \sigma_\omega^2 + \sigma_\omega^2 \Delta t \quad (26)$$

After applying the replacement (26) in kinematic model (20) the next steps of the future pose estimation is rather straightforward, if we keep in mind that the mean value of the product of two normally distributed variables is affected both by the product mean values and variances like in (25). As said above, the result of those calculations is pose forecast expressed in the form (9) and (10). For shorter notation, let us substitute $\sigma_{MLCx}^2 + \sigma_x^2$ by **a**:

$$\sigma_{X_i}^2 = \begin{matrix} \overline{\frac{\sigma_{\omega\Delta t}^2 \sin\ \omega\Delta t\ a}{\sigma_{\omega\Delta t}^2 \sin\ \omega\Delta t\ +a}} + \overline{\frac{\sigma_{\omega\Delta t}^2 \cos(\omega\Delta t) a}{\sigma_{\omega\Delta t}^2 \cos\ \omega\Delta t\ +a}} + \sigma_{MLCx}^2 \\ \overline{\frac{\sigma_{\omega\Delta t}^2 \cos(\omega\Delta t) a}{\sigma_{\omega\Delta t}^2 \cos\ \omega\Delta t\ +a}} + \overline{\frac{\sigma_{\omega\Delta t}^2 \sin(\omega\Delta t) a}{\sigma_{\omega\Delta t}^2 \sin\ \omega\Delta t\ +a}} + \sigma_{MLCy}^2 \\ \sigma_{\omega\Delta t}^2 + \sigma_\theta^2 \end{matrix} \quad (27)$$

$$X_i = \begin{matrix} \frac{\cos(\omega\Delta t)a + (x - MLC_x)\sigma_{\omega\Delta t}^2}{\sigma_{\omega\Delta t}^2 \sin\ \omega\Delta t\ +a} - \frac{\sin(\omega\Delta t)a + (x - MLC_x)\sigma_{\omega\Delta t}^2}{\sigma_{\omega\Delta t}^2 \cos\ \omega\Delta t\ +a} + MLC_x \\ \frac{\cos(\omega\Delta t)a + (x - MLC_x)\sigma_{\omega\Delta t}^2}{\sigma_{\omega\Delta t}^2 \sin\ \omega\Delta t\ +a} - \frac{\sin(\omega\Delta t)a + (x - MLC_x)\sigma_{\omega\Delta t}^2}{\sigma_{\omega\Delta t}^2 \cos\ \omega\Delta t\ +a} + MLC_y \\ \theta + \omega\Delta t \end{matrix} \quad (28)$$

Term (27) in further calculations is transformed into covariance matrix in the form (9) by replacing the main diagonal with the appropriate values of variances.

Other sensor data processing requires less effort because they are not passed through the kinematic model.

While providing means for error and position estimation equations (27) and (28) are rather complex, which means that every unnecessary operation may lead to increase of the error. Therefore we propose to use more direct approach that is described in the next section.

VI. Use of Multiple Signal Response Functions

To describe the approach it is necessary to look at the actual problem once again – here the input is defined by a vector $[v_l\ t\ , v_r\ t\ ]$, and the expected output from the forecast model is $[v\ t\ , \omega(t)]$, where v(t) – linear speed, and $\omega(t)$ – angular speed of the robot. This problem statement eliminates the unnecessary operations that were required to actually calculate the linear and angular speeds.

Now, having in mind the definition of the model (12), it is possible to define the model in Laplace domain (29):

$$\begin{matrix} v\ s \\ \omega\ s \end{matrix} = \begin{matrix} H_{Lv}\ s & H_{Rv}\ s \\ H_{L\omega}\ s & H_{R\omega}\ s \end{matrix} \cdot \begin{matrix} L\ s \\ R\ s \end{matrix} \quad (29)$$

Here, for a simpler notation the input vector is rewritten as $[L(t), R(t)]$.

Knowing that Laplace transformation of the matrix (30) is matrix (31) with the transformed elements and having the transformations of product and convolutions i.e. (12) and (13) it is possible to rewrite (29) in time domain as (32):

$$x\ t\ = \begin{matrix} x_1(t) \\ x_2(t) \end{matrix} \quad (30)$$

$$\mathcal{L}\ x\ t\ = \begin{matrix} \mathcal{L}\ x_1(t) \\ \mathcal{L}[x_2\ t\ ] \end{matrix} = X(s) \quad (31)$$

$$\begin{matrix} v(t) \\ \omega(t) \end{matrix} = \begin{matrix} \frac{d}{dt}u_{Lv}\ t\ * L\ t\ + \frac{d}{dt}u_{Rv}\ t\ * R(t) \\ \frac{d}{dt}u_{L\omega}\ t\ * L\ t\ + \frac{d}{dt}u_{R\omega}\ t\ * R(t) \end{matrix} \quad (32)$$

In the equation (32) step functions (see 14) $u_{Lv}(t)$, $u_{Rv}(t)$, $u_{L\varphi'}(t)$ and $u_{R\varphi'}(t)$ can be obtained empirically by simply switching on each motor at a full speed and collecting the actual $v(t)$ and $\omega(t)$ data until the speed difference over a single time step achieves 0 meaning that the top speed is reached and transient processes are finished (see figure Fig 10).

Here again we assume that both output functions are normally distributed random variables: $\sim N(\mu_\omega, \sigma_\omega^2)$ and $\sim N(\mu_v, \sigma_v^2)$. Now it is possible to rewrite the actual kinematic model:

$$X_{t+1} = \begin{matrix} x_t + v \cdot \Delta t \cdot \cos(\theta_t) \\ y_t + v \cdot \Delta t \cdot \sin(\theta_t) \\ \theta_t \end{matrix} \ , \quad \omega = 0$$
$$\begin{matrix} x_t - R \cdot \sin\ \theta_t\ + R \cdot \sin(\theta_t + \omega \cdot \Delta t) \\ y_t + R \cdot \cos\ \theta_t\ - R \cdot \cos(\theta_t + \omega \cdot \Delta t) \\ \theta_t + \omega \cdot \Delta t \end{matrix} \ , \omega \neq 0 \quad (33)$$

In (33) R = v/ω. To estimate the variance and the final values of the model it is necessary to use algebra of random variables as it was described above.

This approach allows to eliminate operations (17), (18) and (19), thus simplifying the overall calculations and reducing increase of the variance of the final pose estimation.

VII. Experimental Test of the Model

To validate the proposed integrated localisation method we have developed an experimental simulation software that models differential drive robot in 2D environment by implementing the kinematic model (33). All noisy data is

assumed to be normally distributed random variables as it is stated throughout the paper.

The developed software allows to set regions covered by glyphs and thus providing robot position data without growth of the estimation variance as it is explained in section IV. Along with the possibility to track a set of 100 possible robot positions according to the estimation variances, the software also provides the functionality to acquire signal transfer functions thereby simulating the noise caused by encoder inaccuracy and mechanical imperfection of the robotic system.

Examples of signal transfer functions (29) are depicted in Fig. 9, where t11 represents $H_{Lv}\ s$, t12 represents $H_{Rv}\ s$, t21 represents $H_{L\omega}\ s$ and t22 represents $H_{R\omega}\ s$.

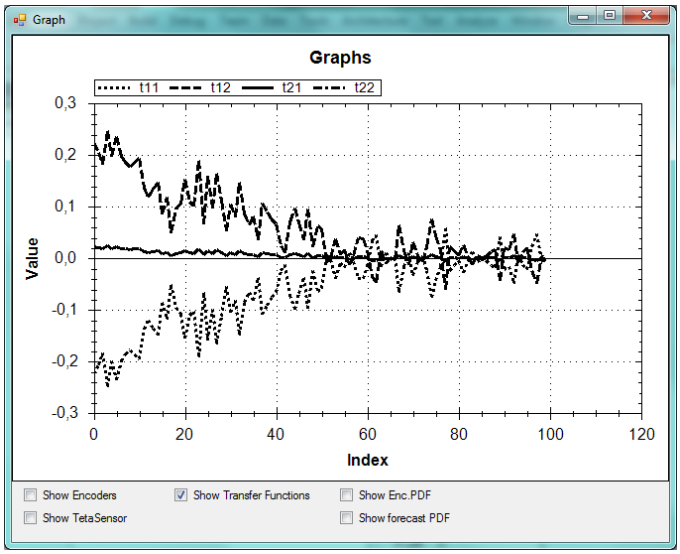

Fig 9. Signal transfer functions acquired from noisy encoders data.

All four functions were generated from the noisy encoders data depicted in Fig.10.

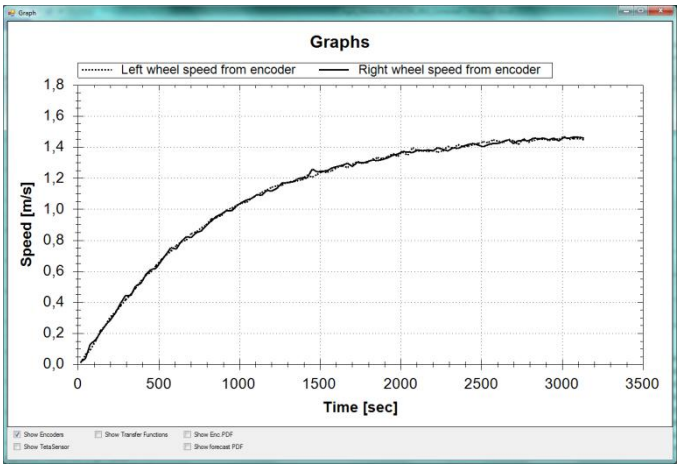

Fig. 10. Noisy encoders readings.

By using data readings from the encoders and passing through kinematics calculation we obtain one position estimation (black crosses in Fig. 11), while using signal transfer functions we acquire another position estimation (blue crosses in Fig. 11). In Fig. 11 pink rectangles are the regions covered by glyphs.

As can be noticed in figure Fig 11, both estimations are comparable in terms of estimation distance from real position and in terms of confidence of the estimations, which variances are depicted in figures Fig 12 and Fig 13.

In both cases the same encoder variances are used. Currently, the tests are performed by using numerical data of probabilities propagation, that is, the tests do not use the data from actual robots. Therefore, the expressions (27) and (28) are replaced by numerically close time series, thus simplifying the initial testing of the whole estimation mechanism.

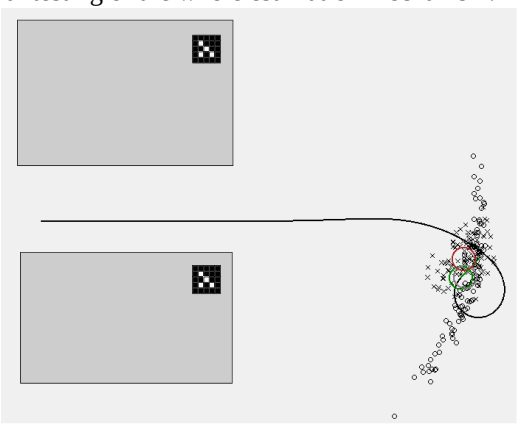

Fig. 11. Positions estimations based on encoders and signal transfer data (more than 9m travelled)

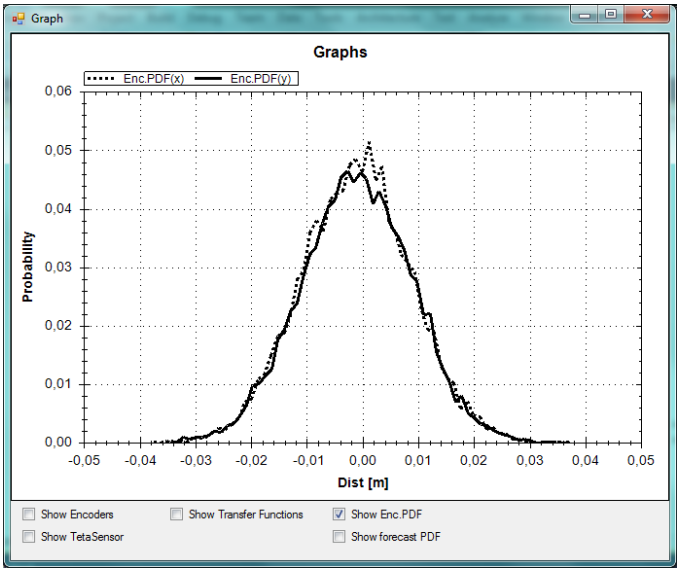

Fig. 12. Variances of estimation acquired from encoders (PDF – probability distribution function)

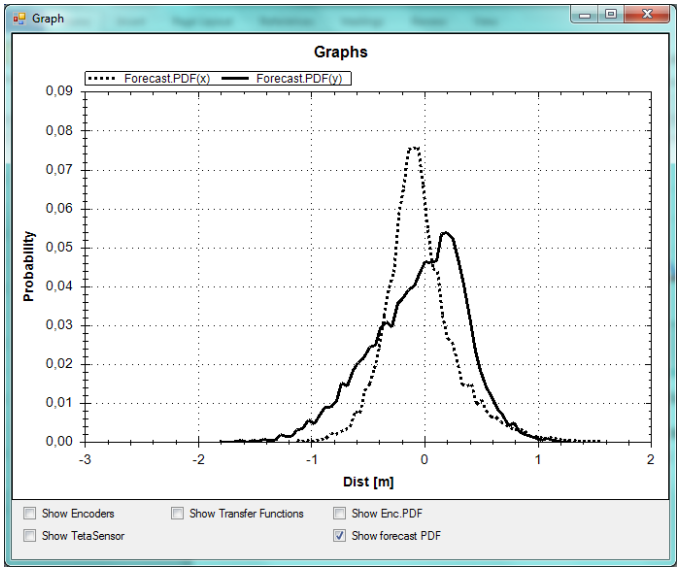

Fig. 13. Variances of estimation acquired from signal transfer functions (PDF – probability distribution function)

As it is described above, the proposed integrated method allows to combine landmark-based localisation with signal transfer based and wheel encoders based localisation techniques. The test run results are depicted in Fig. 14.

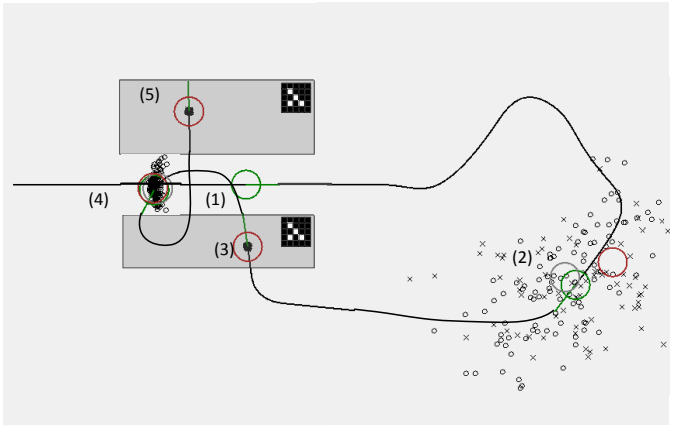

Fig. 14. Integrated position estimation: (1) – starting point, (2) – estimation based on signal transfers function (cross) and wheel encoders (circles), (3) – estimation based on landmarks, (4) – estimation based on signal transfers function and wheel encoders, (5) - estimation based on landmarks. More than 20m travelled.

In order to get the final estimation as explained above, we use equation (4) and (5) proposed by [7], which allows to fuse them if they are assumed to be normally distributed. A rather large distribution variance in position (3) is caused by sensor errors for simulation purposes only. These variances in real application are estimated empirically or are obtained from sensor specifications.

VIII. Conclusions and Future Work

The proposed integrated robot localization method provides the means for localising robots in indoor environment through combination of artificial landmarks and other sensor data. We have proposed to use the signal transfer function along with other sensors providing means for position estimations from control signals as well as deal with sensor data reading latencies, where actual readings are replaced by sensor data time models based on signal transfer functions.

According to the first numerical tests the proposed application of signal transfer functions can provide the estimation accuracy and confidence that is comparable to traditional sensor-based methods such as use of wheel encoders because signal transfer function based estimation takes into account all imperfections of the system including specifics of the environment like, for example, ground surface roughness.

This enables to widen available positioning data sources as well as provides a back-up positioning method in domains where high reliability is required or cost effectiveness is crucial. We intend to apply this method in multi-robot system being developed for indoor use whenever glyph data is not available.

Our future developments will focus on method field tests by using real differential drive robots as well as more accurate random noise distributions evaluations. Currently we assume them being normally distributed, which might not always be the case.

IX. References

[1] J.R.Raol Multi-Sensor Data fusion with MATLAB, CRC Press 2010, 534 pages.

[2] D.L.Hall, J.Llinas Handbook of multisensory data fusion, CRC Press, 2001, 537 pages.

[3] http://www.irobot.com/us/robots/home/roomba.aspx cited: 16.08.2012.

[4] iRobot Roomba 500 Open interface specification, iRobot, Cited: 16.08.2012.

[5] H.B. Mitchell Data Fusion: Concepts and Ideas, Springer – Verlag, 2010,348 pages.

[6] C.Chen and company A Pairwise-Gaussian-Merging Approach towards Genome Segmentation for Copy Number Analysis, World Academy of Science, Engineering and Technology, 2009, 54p.

[7] A.W.Stroupe, M.C.Martin, T.Balch Distributed Sensor Fusion for Object Position Estimation by Multi-Robot Systems, Proceedings of ICRA IEEE vol.2., 2001, pp 1092 - 1098

[8] S.Tennina, M. Valletta, F. Santucci, M.D.Renzo, F. Graziosi, R.Minutolo, Entity Localization and Tracking: A Sensor Fusion-Based Mechanism in WSNs, Proceedings of IEEE 13th International Conference on HPCC, 2011, pp 983 – 988.

[9] A.Prieto, J.A. Becerra, F. Bellas, R.J. Duro Open-ended evolution as a means to self-organize heterogeneous multi-robot systems in real time, Robotics and Autonomous Systems 58, 2010, pp 1282–1291

[10] L.K.Jensen, B.B.Kristensen, Y. Demazeau, FLIP: Prototyping multi-robot systems, Elsevier, Robotics and Autonomous Systems 53, 2005, pp 230–243

[11] B.Zitov, J.Flusser Landmark recognition using invariant features, Elsevier, Pattern Recognition Letters 20, 1999, pp 541 - 547

[12] G.Jang, S.Kim, J.Kim, I.Kweon Metric Localization Using a Single Artificial Landmark for Indoor Mobile Robots, Proceedings of IEEE International Conference on IROS, 2005, pp 2857 – 2862

[13] Open source augmented reality project GRAFT (http://www.aforgenet.com/articles/glyph_recognition/) Cited: 16.08.2012.

[14] A.Bradski, A.Kaehler Learning OpenCV: Computer vision with the OpenCV Library, O'Reilly Media, 2008, 555 pages.

[15] C.L.Phillips, J.M.Parr Signals, Systems and Transforms: 4th edition, Pearson Education, 2008, 795 pages

[16] G. Dutek and M. Jenkin *Computational Principles of Mobile Robotics*, Cambridge: Cambridge University Press, 2000

[17] Jack Hayya, Donald Armstrong and Nicolas Gressis Management Science Vol. 21, No. 11, Theory Series, 1975, pp 1338-1341

X. ACKNOWLEDGEMENT

The research described in this paper is supported by funding of the ERDF Project «Development of technology for multiagent robotic intelligent system », project implementation contract No. 2010/0258/2DP/ 2.1.1.1.0/10/ APIA/VIAA/005.

Agris Nikitenko has received his Dr.sc.ing. in 2006 from Riga Technical University as well as Bc.sc.ing. and Mg.sc.ing. focusing on artificial intelligence in his thesis.

Currently he is docent in the Department of Systems Theory and Design of Riga Technical University and vice dean of study affairs in the Faculty of Computer Science and Information Technology. His scientific interests cover artificial intelligence and autonomy as well as their application in robotics.

He has received award of Verner fon Siemens for his doctoral thesis in 2006. At present he is member of IEEE and ACM as well as represents Latvia in NATO RTO AVT panel.

A. Liekna has received his Bc.sc.ing degree in 2008 and his Mg.sc.ing. degree in 2010 from Riga Technical University. At the moment he is a PhD student at Riga Technical University. His major field of study is computer science.

He is working as a Research Assistant at Riga Technical University. His research interests include artificial intelligence and multi-agent systems.

He was awarded by the Latvian Foundation for Education for his bachelor's thesis “Development and Implementation of Reinforcement Learning Model”.

Martins Ekmanis is master of engineering sciences in telecommunications, he graduated from Riga Technical University in 2005. He continues studies at the faculty of Computer Science and Information Technology as Ph.D. student. His professional interests are focused on robotic systems, empirical model building and machine learning. He is member of IEEE and has participated in several international scientific conferences and research projects.

Guntis Kulikovskis has received his Mg.sc.ing. in Riga Technical University focusing on driving mechanisms of underwater vehicles. At the moment he is a PhD student in the Faculty of Transport and Mechanical Engineering.

Currently he is researcher in the Division of Autonomous Robotic Systems of Riga Technical University. His main scientific focus areas are highly mobile ground vehicles and underwater robotic systems.

Ilze Andersone has received a B.Sc. degree in computer control and computer science in 2007. In 2009 she received a M.Sc. degree in computer systems. Both degrees were acquired at Riga Technical University, Riga, Latvia. She is now a PhD student at Riga Technical University.

In 2007 – 2009 she worked as a Laboratory Assistant at the Department of System Theory and Design, Riga Technical University. Since 2009 she has been working as a Researcher at the Institute of Applied Computer Systems, Riga Technical University. Her research interests include robotic mapping and robot teams.

.