Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 100 (2015) 737 – 745

Procedia Engineering

www.elsevier.com/locate/procedia

25th DAAAM International Symposium on Intelligent Manufacturing and Automation, DAAAM 2014

The Universal Onboard Information-Control System for Mobile Robots

Vladimir Filaretov[a,b], Dmitry Yukhimets[a,b], Eduard Mursalimov[a,b]*

[a]Far Eastern Federal University, 8, Sukhanova str.,Vladivostok, Russia
[b]Institute of Automation and Control Processes FEB RAS, 5, Radio str.,Vladivostok, Russia

Abstract

The approach to creation of universal software platform for realization of information-control system of mobile robot is proposed in this work. The main feature of this platform is universal structure of software components which consist of functional part and communication part. This structure allows provide the independence of functional part from hardware of mobile robot. The minimal set of software components which allows create the core of information-control system of mobile robot is proposed. Also the algorithms realizing of functions of these components are described.

Keywords: mobile robot; control system; software architecture; data fusion; trajectory forming

1. Introduction

Now during of creation of new mobile robots (MR) (unmanned ground vehicle (UGV), autonomous underwater vehicles (AUV), unmanned aerial vehicle (UAV) and etc.) the developers spend a lot of time for developing of MR onboard information control system (ICS). Herewith the main complexity of development of these systems is a wide variety of devices (sensors, actuating device etc.), data exchange interfaces included in the MR and functions that must be implemented in the ICS.

There are some approaches for development of ICS MR. The first approach is development of ICS as a single

* Corresponding author. Tel.: +7-924-242-3293; fax: +7-423-231-0452.
E-mail address: murs@iacp.dvo.ru

doi:10.1016/j.proeng.2015.01.427

program which consistently executes all necessary operations [1]: polling sensors, processing of incoming data and the control function generation. However, if ICS contains a computationally complex control algorithms and data processing algorithms then the generating time of control signals increases significantly that significantly degrades the control quality of the MR. This fact makes such ICS little useful in modern MR.

Another approach to the development of MR ICS is the creation of distributed ICS [1, 2], which consists of some programs, interacting with each other and implementing specified functions of ICS.

Herewith one of the most important and difficult tasks at development of these ICS is development of communication protocols between individual control programs. The composition of onboard computing resources of MR usually containing some individual computing devices [3–6], and distribution strategy of functions of ICS for these devices significantly affect on special aspects of realization these ICS.

Besides the difficulties of development of these ICS it should take into account the need of reusing developed components in other MR with other hardware.

In addition, most existing approaches to development of MR ICS focus on the implementation of the stabilization modes and movement on a sequence of target points. However, for more effective performance of the many tasks it should provide the movement of MR along trajectories generated on the based of using data from its onboard sensors and cameras.

Also often, it is necessary to use more complex control systems then linear controllers for providing of high quality movement of MR. But for synthesis and realization of such control systems it is necessary to provide of forming during work of MR the information about current MR state vector and its parameters. It can be complicated by limited quantity of onboard sensors.

Besides realization of algorithms of control and navigation the testing of these algorithms and whole ICS is very important. It is wise to use the hardware-in-loop simulation mode which allows safely perform tuning of control system that essentially important for such MR as underwater and aerial vehicles.

Now the approaches for solution of each separate task have been developed, but universal approach to realization of ICS MR having all listed functional properties was not developed. In this paper the structure of such ICS and approaches to realization of its main components are offered.

2. Description of the base architecture

One of the main task at development of ICS MR is development of universal architecture which allow to implement the ICS on the different hardware and also provide the extension of its functional properties.

Now there are universal architecture of ICS MR which are successfully use in practice. As a base architecture it is expedient to use Joint Architecture for Unmanned Systems (JAUS) [7, 8], which is well-proven at development of ICS for complex autonomous MR. The block-diagram of ICS which create on the base of this architecture is shown in Fig.1.

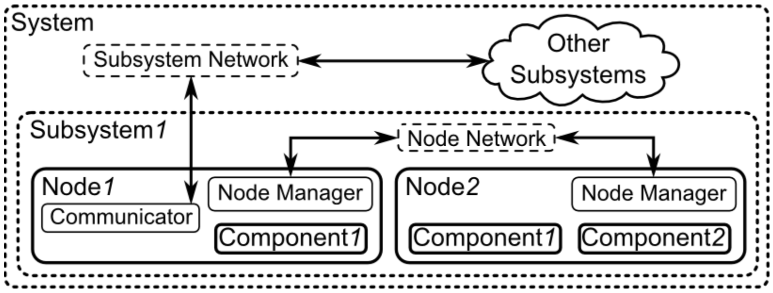

Fig. 1. The block-diagram of ICS on the base JAUS.

The ICS MR implemented on the base of JAUS architecture consists of subsystems including separate devices (for example, operator control unit, mobile robot and etc.). Each subsystem contains the set of interacting nodes usually constituting the real devices of MR, and each node is the set of interacting components each of them implements the separate function (control law, trajectory forming and etc.). Herewith each component included in ICS has address consisting of number of subsystem, number of node in subsystem and number of component in node.

Each component is implemented as object which consist of two part: communication interface and functional part. The structure of component of ICS is shown in Fig. 2. The feature of implementation of ICS component is organization of its work like work of controller program: the serial performance of function in infinity loop. This approach allows without any difficulties to implement the component both on onboard computer and on single controller.

The interaction between different component of ICS realize on the base of exchanging messages of JAUS standard that allows provide the expendability and compatibility of ICS with different MR as components performing equal functions (for example, control of movement MR) in ICS of different MR exchange the equal messages.

Furthermore, in this case it became possible to provide the simple integration of different ICS MR been implemented on the base of this architecture.

The special type of component using in ICS MR with JAUS architecture is manager which perform a routing of messages. Herewith two type of manager are used: the node manager which does routing of messages between nodes of subsystem and communicator which does routing of messages between different subsystems.

The class described the component manager in whole correspond to structure shown in Fig. 2. The difference lies in that component-manager can perform data exchange by some interfaces. For example, it can provide the data exchange between components located in onboard computer by protocols UDP or TCP and with components located in controllers by serial interface. Herewith the manager has to have information about addresses of components located in one node and addresses of nodes in one subsystem. The structure of component manager is shown in Fig. 3.

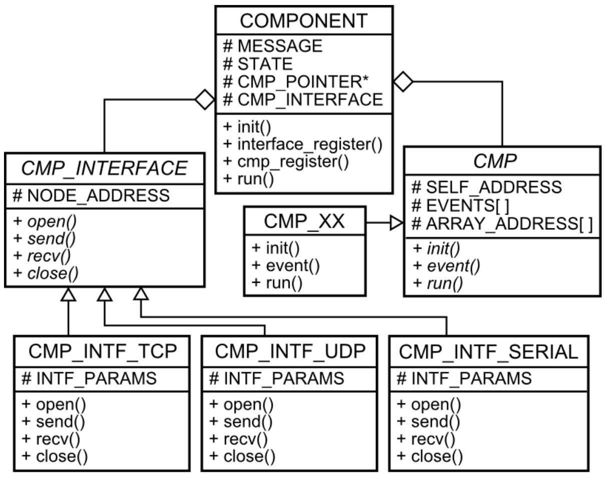

Fig. 2. The structure of component of ICS MR.

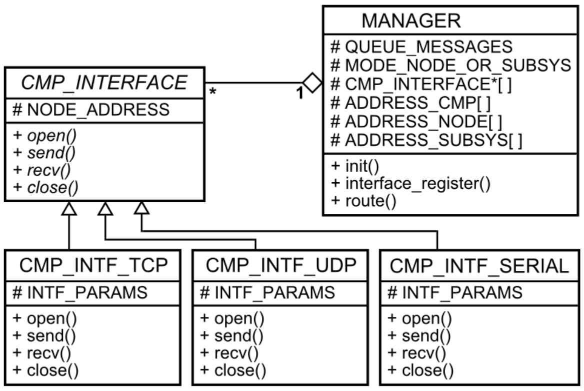

Fig. 3. The structure of component manager.

With taking into account of described architecture the structure and set of components of offered ICS are shown in Fig. 4.

As we can see from Fig. 4 the ICS MR includes the following main components:

1. the component of trajectory forming which allows to form the smooth trajectory on the base of the set of waypoints;
2. the component of data fusion which allows to estimate of state vector of MR on the base of received data from onboard sensors;
3. the component of MR parameters identification which allows to estimate the MR parameters during its working

and uses its parameters for build of MR mathematic model;

4. the component of integration of ICS MR with simulation software which allows to provide of hardware-in-loop simulation and tuning of MR control system;
5. the component of control system which implements of control algorithms on the base of data receiving from components of data fusion, trajectory forming and identification.

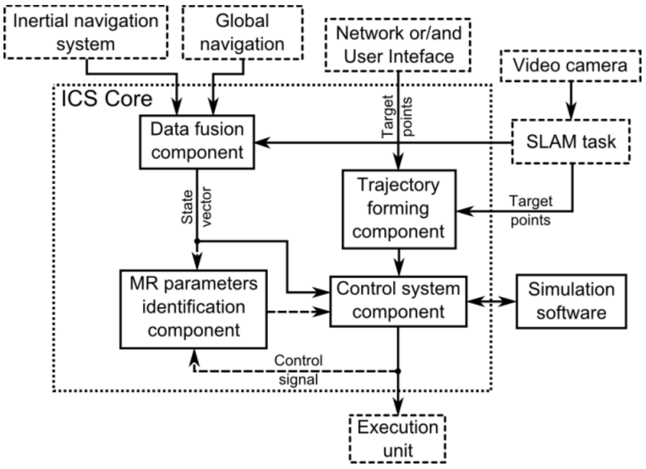

Fig.4 Structure of ICS MR.

Further we describe some features of implementation of each component.

3. Description of components of ICS MR

3.1. Data fusion component

This component solves the following problems: noisiness of data received from MR onboard sensors, incomplete measurements of MR state vector elements and different periods of sensors data updating. Therewith the recovery of MR state vector will be made on the base of its kinematic model, which have same form for many kinds of MR:

$$\dot{\eta} = J(\eta)\upsilon \, , J(\eta) = \left[\begin{array}{c:c} J_1(\eta) & 0 \\ \hdashline 0 & J_2(\eta) \end{array}\right], \tag{1}$$

where $\eta = [x, y, z, \varphi, \theta, \psi]^T$; $\upsilon = [\upsilon_x, \upsilon_y, \upsilon_z, \omega_x, \omega_y, \omega_z]^T$ is the vector of projections of linear and angular velocities of MR on the axes of joined coordinate system; $J(\eta)$ is the matrix of transition from jointed to absolute coordinate system which has a view:

$$J_1(\eta) = \left[\begin{array}{c:c:c} \cos\psi\cos\theta & \begin{array}{c}\cos\psi\sin\theta\sin\varphi - \\ -\sin\psi\cos\varphi\end{array} & \begin{array}{c}\sin\psi\sin\varphi + \\ +\cos\psi\cos\varphi\sin\theta\end{array} \\ \hdashline \sin\psi\cos\theta & \begin{array}{c}\cos\psi\cos\varphi + \\ +\sin\varphi\sin\theta\sin\psi\end{array} & \begin{array}{c}\sin\theta\sin\psi\cos\varphi - \\ -\cos\psi\sin\varphi\end{array} \\ \hdashline -\sin\theta & \cos\theta\sin\varphi & \cos\theta\cos\varphi \end{array}\right], \; J_2(\eta) = \left[\begin{array}{c:c:c} 1 & \sin\varphi\tan\theta & \cos\varphi\tan\theta \\ \hdashline 0 & \cos\varphi & -\sin\varphi \\ \hdashline 0 & \dfrac{\sin\varphi}{\cos\theta} & \dfrac{\cos\varphi}{\cos\theta} \end{array}\right].$$

As model (1) is significantly nonlinear then it is viable to use Unscented Kalman Filter (UKF) [9, 10] which can work with nonlinear models without their linearization.

The suggested algorithm of recovery MR state vector which is built on the base of UKF works following manner. On the each step of its work the measurement vector consisting only of actual data received from onboard sensors is formed. Actual data is a data which was updated on the current step of discretization. As updating periods of signals from different sensors are different the measurement vector has variable size. This vector enters on the input of UKF and its parameters change dynamically depends on dimension of measurements vector. On the output of UKF on the base of model (1) the estimate of UV state vector is formed.

Let consider more detailed the work of UKF on the first stage of algorithm. The UKF work with following model:

$$X_{k+1} = F(X_k) + \xi_k \,, \; Y_k = H(X_k) + \zeta_k \,, \tag{2}$$

where $X_k \in R^n$ is the state vector having dimension *n*; $Y_k \in R^m$ is the measurement vector; $F(X_k) \in R^n$ is the vector-function describing the transition of system from state X_k to state X_{k+1} ; $H(X_k) \in R^m$ is the vector-function of measurement; $\xi_k \in R^n$ и $\zeta_k \in R^m$ are the noise vectors of system and measurements correspondingly which are Gaussian probability processes.

The state vector of MR has view:

$$X_k = [a_k^T, \upsilon_k^T, \eta_k^T]^T \in R^{18} ,$$

where $a_k = [\dot{\upsilon}_x^k, \dot{\upsilon}_y^k, \dot{\upsilon}_z^k, \dot{\omega}_x^k, \dot{\omega}_y^k, \dot{\omega}_z^k]^T$ is the vector of linear and angular accelerations of MR in the joint coordinate system. As in offered algorithm the kinematic model of MR movement (see expressions (1)) is used then vector-function $F(X_k)$ has view:

$$F(X_k) = \left[\begin{array}{c} a_k \\ \hline a_k \Delta t + \upsilon_k \\ \hline \dfrac{J(\eta_k) a_k \Delta t^2}{2} + J(\eta_k) \upsilon_k \Delta t + \eta_k \end{array} \right],$$

where a_k, υ_k, η_k are value of corresponding vectors for state X_k ; Δt is the discretisation period.

In general case vector-function $H(X_k)$ describes nonlinear interconnections between measurements and MR state vector elements and its form depend on set of MR onboard sensor. Therewith if all MR sensors measure the elements of MR state vector then equation (2) can be rewritten in following view:

$$Y_k = H(\overline{X}_k) + \zeta_k = \overline{H}(\eta_k) \overline{X}_k + \zeta_k ,$$

where $\overline{H}(\eta_k)$ is the matrix of coefficients which size depends on measure vector size.

After formation of vector Y_k and matrix $\overline{H}_k$ one can do estimate of MR state vector by using of UKF.

3.2. Identification component

In this component the estimation of MR parameters is performed on the base of information about MR state vector. The analysis of mathematical models of MR different types (AUV, UAV, UGV and etc.) shows that these models for many types of MR are linear relatively of its parameters. Because for their estimation we can use the

Linear Kalman Filter (LKF) [11], and MR model must be presented in following view:

$$q(k+1) = q(k),$$
$$S(X_k) = F(X_k)q(k) + e(k)\,, \tag{3}$$

where $F(X_k)$ is the matrix being formed on the base of model of MR dynamic, $q(k)$ is the vector of identified parameters of MR. As model (3) is linear then for identification of MR parameters one can use LKF which doesn't have large calculation complexity.

3.3. Component of trajectory forming

Herewith, it is convenient to form the trajectory of the AUV as a Bezier spline. For this purpose, the coordinates of two base points which are waypoints and two control points which set the tangent line to the trajectory in these waypoints has to be set [12]. An example of a spline plot is shown in Fig. 5.

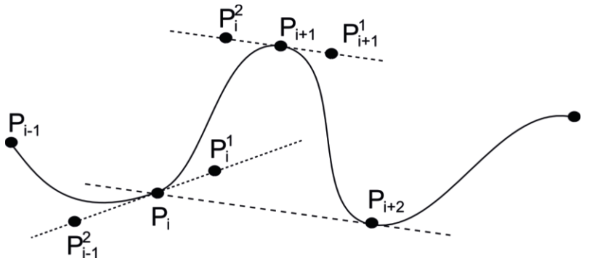

Fig. 5. The process of spline formation on the set points.

The algorithm of defining a point $X^*(t)$ moving along the desired spatial trajectory consists of the following stages.

1. Calculation of the coordinates of the spline control points:

$$\Delta P_i = \frac{P_i - P_{i-1}}{\| P_i - P_{i-1} \|}, \Delta P_{i+1} = \frac{P_{i+1} - P_i}{\| P_{i+1} - P_i \|},$$
$$\Delta P_i^C = k_{pi} \| P_i - P_{i-1} \| \frac{(k_{ci}\Delta P_i + (1-k_{ci})\Delta P_{i+1})}{\| (k_{ci}\Delta P_i + (1-k_{ci})\Delta P_{i+1}) \|}, \tag{4}$$
$$P_i^1 = P_i + \Delta P_i^C, P_i^2 = P_{i+1} - \Delta P_{i+1}^C,$$

where $P_i = (x_i, y_i, z_i)$ is the coordinates of a base point, to which the desired trajectory has to go; P_i^j, $j = \overline{(1,2)}$ are the coordinates of the spline control points; k_{pi}, k_{ci} are the coefficients setting the positions of the spline control points.

It should be noted that the coefficient $0 \le k_{ci} \le 1$ sets the slope of the tangent line to the trajectory in a waypoint (for most cases the value $k_{ci} = 0.5$ can be considered, and coefficient $k_{pi} \ge 0$ sets the curvature of the trajectory (if $k_{pi} = 0$, then the trajectory is a straight line)).

2. Calculation of the current position of point $X^*(t)$ on the desired trajectory [12].

Parameterization of the AUV trajectory equation formed on the basis of the Bezier spline is done by the following expression [12]:

$$X^*(\delta) = (1-\delta)^3 P_i + 3\delta(1-\delta)^2 P_i^1 + 3\delta^2(1-\delta)P_i^2 + \delta^3 P_{i+1} = f_B(\delta)\,, \tag{5}$$

where $\delta \in [0,1]$ is the normalized time of motion of point $X^*(t)$ along the spline between points P_i and P_{i+1}.

The speed of motion of point $X^*(t)$ along the trajectory is defined by the rate of normalized time. Since the distances from neighbouring waypoints are different and unknown, it is difficult to receive the relation in an analytical view between the values δ of normalized and real t times. The following iteration algorithm was developed to provide the motion of point $X^*(t)$ along the spline with the desired speed:

a. $i = 0 : T^* = \upsilon^*\tau,\ X_i^* = X^*(t),\ \delta_i = \delta(k)$

b. $i > 0 : \delta_{i+1} = \delta_i + \Delta\delta,\ X_{i+1}^* = f_B(\delta_{i+1}), T_{i+1} = T_i + \|\ X_{i+1}^* - X_i^*\ \|,\ i = i+1,\ if\ T_{i+1} < T^*$ (6)

c. $X^*(t+\tau) = f_B(\delta_i + \dfrac{T^* - T_i}{T_{i+1} - T_i}\Delta\delta),\ if\ T_{i+1} \geq T^*$

where υ^* is the desired speed of AUV motion along the trajectory; T^* is the way length which the point $X^*(t)$ goes along the trajectory with speed υ^* during a discretization period τ ; $\Delta\delta$ is the increment of normalized time for each algorithm iteration; $\delta(k)$ is the value of normalized time corresponding to the previous position of point $X^*(t)$.

Our research showed that this algorithm (see expressions (4)-(6)) allows definition of the next position of point $X^*(t)$ on the desired trajectory in 2-3 iterations and provides the desired accuracy of speed formation.

3.4. Component of integration of ICS with simulation software

This component allows implement of hardware-in-loop simulation mode of ICS MR and tuning of parameters of control algorithm. If this mode is active then control system sends data about control signals to simulation software where the MR mathematical model is implemented on the base of received data during identification. Herewith the ICS receive information about MR state from this model as result of simulation of MR response to control signals.

It is expedient to use Matlab As simulation software because it is standard engineering software for developing and simulation of control systems.

The structure schema of component of integration with simulation software during its work in hardware-in-loop mode is shown in Fig. 6.

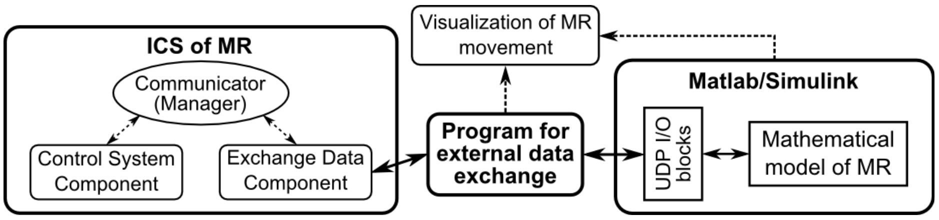

Fig. 6. Structure schema of component of integration with simulation software.

In the schema in Fig.6 it can use both program realization of ICS MR working in local network and MR onboard computer. In the first case the using of hardware-in-loop simulation allows provide the testing of correctness of ICS

MR software implementation, and in second case it allows provide tuning of control algorithm with taking into account the real delays and onboard computer speed.

For adapting of data exchange interfaces between mathematical model and ICS the additional software module is used (see Fig. 6). This software module receive data about MR movement calculating in mathematical model, transforms it in form which is necessary for using in MR control system and sends it to ICS. Also this module receives information from ICS about control signal and sends it to mathematical model. The using of this additional software module allows both to adapt formats and interfaces of data exchange and to provide work of ICS and simulation software with different discretization frequency.

The main feature of this simulation system is the presence of two parallel computing processes executing in two different applications with different speeds. In addition, the process of the data block transferring, especially in the case of using a local network, takes time. As a result, a situation can arise when data on external action computed in the ICS MR will be obsolete for the current step of computer time in Matlab. It will result in an inadequate simulation of MR motion and work its control system. Therefore, to elimination of this situation, the synchronizing problem of the speed of the computing processes.

For soling of this problem it should use the special communication blocks which are realized in Matlab/Simulink. The main elements in these Simulink blocks are s-functions performing importation and transmission data by UDP. Initialization of the protocol in these functions is executed at the beginning of the simulation process when creating the objects of UDP type, which parameters are parameters that are defined by user for data exchange. Sending and receiving data by UDP protocol is exercised by the instrumentality of library functions of Matlab such as fwrite(·) and fread(·). The instances of creating UDP-classes are the parameters of these functions, as well as vector of sending data (for function fwrite(·)) and count bytes containing in one datagram (for function fread(·)).

These communication block are realized the following synchronization algorithm. Let the computer time of simulation in Matlab relates to real time as follows:

$$\tau = k_t t \,, \tag{7}$$

where t is interval of real time that elapsed from the beginning of the simulation process, $k_t > 0$ is coefficient determining simulation speed.

Let at some moment of computer time τ_{si} from Matlab data are sent to external program. These data are sent to external program at time t_{mp}, then data are processed in it at time t_v and then the result is sent to Matlab at time t_{pm}. Thus, the moment of computer time τ_r^{si}, when data calculated for time τ_{si}, come in to Matlab, with a glance (7), will be equal

$$\tau_r^{si} = \tau_{si} + k_t(t_{mp} + t_v + t_{pm}) \,.$$

Obviously, in case when the difference between the moments of computer time τ_{si} and τ_r^{si} is great, the data calculated by an external program are obsolete for using in the current moment of computer time and, consequently, errors in the simulation process are possible. In order to avoid this effect, it is necessary to provide fulfillment of following conditions:

$$\tau_r^{si} - \tau_{si} = k_t(t_{mp} + t_v + t_{pm}) = \tau_\varepsilon \le \tau_{\max} \,,$$

where $\tau_{\max}$ is permissible value of delay for data receiving from external program.

In order to decrease value τ_ε, it is necessary to decrease coefficient of k_t, i.e. the speed of simulation in Matlab. This slowdown of simulation process one may exercise, if to suspend simulation process on some small time interval t_p in certain time moments, for example, when receiving data. The value of t_p is calculated by follows expression:

$$t_p = \begin{cases} k_p(\tau_\varepsilon - \tau_a), & if\ \tau_\varepsilon > \tau_a \\ 0, & if\ \tau_\varepsilon \le \tau_a \end{cases},$$

where k_p is positive coefficient.

Conclusion

In this work the new approach to creation of universal ICS MR which include the base set of components which allow implement the control system for most types of MR independently on their hardware is proposed. The following components of this ICS MR was proposed as a basic: data fusion component, identification component, trajectory forming component, component of integration with simulation software, control system component. Herewith the universal implementation of these components which doesn't depend on MR hardware is proposed and also the algorithms of work of these components are described.

In future it is planned to implement the ICS for group MR which consist of autonomous underwater vehicle, ground wheeled robot and unmanned aerial vehicle for performance various group operations. Furthermore we want test of this ICS on the multilink manipulator. Also is planned to be placed on the Web implemented library on C++ language.

Acknowledgements

This work is supported by Russian Science Foundation (grant No. 14-19-00081).

References

[1] B. Siciliano, O. Khatib, Handbooks of robotics, Springer-Verlag, Berlin, Heidelberg, 2008.

[2] J.J. Borrelly, E. Coste-Maniere, B. Espiau and etc., The ORCCAD architecture, International Journal Robotics Researches, No.4, 17 (1998), pp. 338-359.

[3] B. Seçkin, T. Ayan, E. Germen, Autopilot project with unmanned robot, Procedia Engineering, Vol. 41(2012), pp. 958-964.

[4] I.-W. Park, J.-Y. Kim, B.-K. Cho, J.-H. Oh, Control hardware integration of a biped humanoid robot with an android head, Robotics and Autonomous Systems, 56 (2008), pp. 95-103.

[5] K.bin Hasnan, L. B. Saesar, M.S. Ikhamatiar, T.Herawan, JOMS:system architecture for telemetry and visualization on unmanned vehicle, Procedia Engineering, Vol. 29 (2012), pp. 3899-3903.

[6] J. Jin, Ch. Jung, D. Kim, W. "Chung Development of an autonomous outdoor patrol robot in private road environment", Proceedings of International Conference on Control, Automation and Systems ICCAS 2010, Seoul, Korea, 2010, pp. 1918-1921.

[7] The Joint Architecture for Unmanned Systems, A Set of SAE Interoperability Standards, 2009, http://papers.sae.org/2009-01-3250.

[8] J. Sheng, S. Chung, L. Hansel, D. McLane, J. Morrah, S.-H. Baeg, S. Park, JAUS to EtherCAT Bridge: Toward Real-Time and Deterministic Joint Architecture for Unmanned Systems, Journal of Control Science and Engineering, Vol. 2014 (2014), Article ID 631487, 20 p.

[9] S. Haykin, Kalman filtering and neural networks, John Wiley and Sons, 2001.

[10] S. Julier, J. Uhlmann. "A new extension of the Kalman filter to nonlinear systems", Proceeding of AeroSense: The 11th International Symposium on Aerospace: Defence Sensing, Simulation and Controls, 1997.

[11] E. Ikonen, K. Najim, Advanced process identification and control, Marsel Dekker Inc., New York, NY, USA, 2002.

[12] C. De Boor, A practical guide to splines, Springer, 2001.