Hindawi Publishing Corporation
Modelling and Simulation in Engineering
Volume 2012, Article ID 291318, 4 pages
doi:10.1155/2012/291318

Research Article

An Autonomous Underwater Vehicle Simulation Using Linear Quadratic Servo Based on Open Control Platform

Nanang Syahroni[1] and Jae Weon Choi[2]

[1] *Telecommunication Department, Politeknik Elektronika Negeri Surabaya, Surabaya 60111, Indonesia*
[2] *School of Mechanical Engineering, Pusan National University, Busan 609-735, Republic of Korea*

Correspondence should be addressed to Nanang Syahroni, nanang@eepis-its.edu

Received 4 April 2012; Accepted 10 July 2012

Academic Editor: Ahmed Rachid

This paper presents an optimal regulator for depth control simulation of an autonomous underwater vehicle (AUV) using a new approach of decentralized system environment called open control platform (OCP). Simulation results are presented to demonstrate performance of the proposed method.

1. Introduction

To allow an efficient operation of AUV, a new algorithm with a decentralized structure is developed. A new software infrastructure called Open Control Platform (OCP) accommodates changing navigation information and control components [1–3]. An OCP extends real-time middleware technology via CORBA middleware. In [4], many examples of nice control algorithms for AUV had done in several platforms, but in the implementation are widely opened for software engineering.

2. Equations of Motion

The simplified 6DOF equations of motion in the dimensionless form are considered in this paper [5]:

$$\begin{aligned}
X &= m[\dot{u} - vr + wq - x_G(q^2 + r^2) \\
&\quad + y_G(pq - \dot{r}) + z_G(pr + \dot{q})], \\
Y &= m[\dot{v} + ur - wq + x_G(pq + \dot{r}) \\
&\quad - y_G(p^2 + r^2) + z_G(qr - \dot{p})], \\
Z &= m[\dot{w} - uq + vp + x_G(pr - \dot{q}) \\
&\quad + y_G(qr + \dot{p}) - z_G(p^2 + q^2)], \\
K &= I_x\dot{p} + \left(I_z - I_y\right)qr + I_{xy}(pr - \dot{q}) - I_{yz}(q^2 - r^2) \\
&\quad - I_{xz}(pq + \dot{r}) \\
&\quad + m[y_G(\dot{w} - uq + vp) - z_G(\dot{v} + ur - wp)], \\
M &= I_y\dot{q} + (I_x - I_z)pr - I_{xy}(qr + \dot{p}) - I_{yz}(pq - \dot{r}) \\
&\quad - I_{xz}(p^2 - r^2) \\
&\quad + m[x_G(\dot{w} - uq + vp) - z_G(\dot{u} - vr + wq)], \\
N &= I_z\dot{r} + \left(I_y - I_x\right)pq - I_{xy}(p^2 - q^2) - I_{yz}(pr + \dot{q}) \\
&\quad + I_{xz}(qr - \dot{p}) \\
&\quad + m[x_G(\dot{v} + ur - wp) - y_G(\dot{u} - vr - wq)],
\end{aligned} \tag{1}$$

where, X, Y, and Z are surge, sway, and heave force, respectively; K, M, and N are roll, pitch, and yaw moment, respectively; p, q, and r are roll, pitch, and yaw rate, respectively; u, v, and w are surge, sway, and heave velocity, respectively; x, y, and z are body fixed axes in positive forward, starboard, and down, respectively; I_x, I_y, and I_z are moment of inertia at x-axis, y-axis, and z-axis, respectively; x_G, y_G, and z_G are longitudinal, athwart, and vertical position of center of gravity, respectively; ϕ, θ, and ψ are roll,

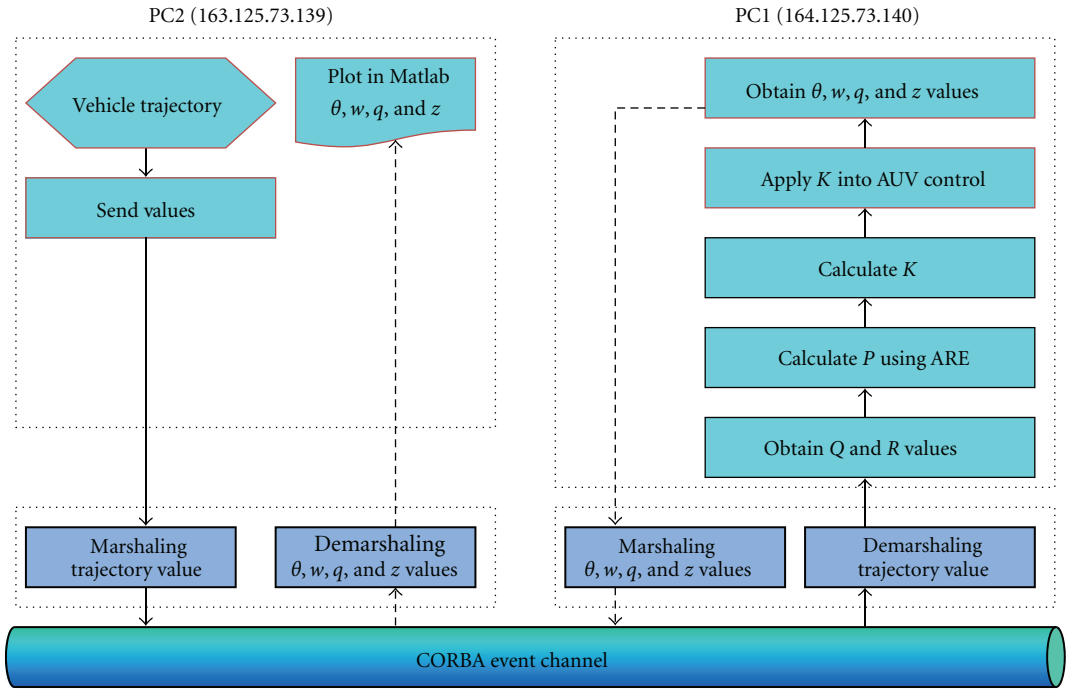

Figure 1: OCP infrastructure.

pitch, and yaw angle, respectively. After several steps of linearization as in [6, 7], vertical motion equations with a control input δ are given as the following:

$$\begin{aligned}
\dot{\theta} &= q,\\
(m - Z_{\dot{w}})\dot{w} - \left(mx_G + Z_{\dot{q}}\right)\dot{q} &= Z_w U w + \left(m + Z_q\right) U q + U^2 Z_\delta \delta,\\
(-M_{\dot{w}} - mx_G)\dot{w} + \left(I_y - M_{\dot{q}}\right)\dot{q} &= -(z_G W - z_B)\theta + M_w U w + \left(M_q - mx_G\right) U q - M_\delta U^2 \delta,\\
\dot{z} &= -U\theta + w.
\end{aligned} \tag{2}$$

Finally, refer to the physical parameters of NPS AUV1 in [8], the state space equation is obtained by:

$$\begin{bmatrix} \dot{\theta} \\ \dot{w} \\ \dot{q} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0.0175 & -1.273 & -3.559 & 0 \\ -0.052 & 1.273 & -2.661 & 0 \\ -5 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ w \\ q \\ z \end{bmatrix} + \begin{bmatrix} 0 \\ 0.085 \\ 21.79 \\ 0 \end{bmatrix} \delta. \tag{3}$$

3. Controller Design

Linear quadratic (LQ) servo is adopted to perform command following regarding to the reference input. Consider an nth-order system with r inputs and m outputs. Let the state vector be $x(t) = [x_r(t); y_p(t)]$, where $y_p(t) \in R^{m\times 1}$ is output and $x_r(t) \in R^{(n-m)\times 1}$ is rest of the system states. Let $e(t) = r(t) - y(t) \in R^{m\times 1}$ be the error vector with $r(t) \in R^{m\times 1}$ which is planed as output and $y(t) \in R^{m\times 1}$ is reference. Then the state space model can be rewritten as the following:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{4}$$

with $y(t) = C_p x(t)$, $x_r(t) = D_p x(t)$, C_p is $[0_{m\times(n-m)} \;\; I_{m\times m}]$, and D_p is $[I_{(n-m)\times(n-m)} \;\; 0_{(n-m)\times m}]$.

LQ optimal control problem is to find a control law $u(t) = -Kx(t)$ which minimizes the performance index of $J = \int_0^\infty [x^T(t)Qx(t) + u^T(t)Ru(t)]dt$, where $K = R^{-1}B^T P$ and P is the solution of Riccati equation $A^T P + PA + PBR^{-1}B^T P + Q = 0$ with weighting matrices of $Q \geq 0$ and $R > 0$.

The control input is given by $u(t) = -Gx(t)$, where the control gain G is consisted by $G = [G_y \; G_r]$, where $G_y \in R^{m\times m}$ and $G_r \in R^{m\times(n-m)}$. Then the control law can be obtained by:

$$u(t) = -G_y y(t) - G_r x_r(t) + G_y r(t). \tag{5}$$

Substituting (5) into (4), we have differential equation of close-loop system as follows:

$$\dot{x}(t) = \left[A_r - BG_y C_p - BG_y D_p\right]x(t) + BG_y r(t). \tag{6}$$

Solution of (6) can be obtained by using Runge-Kutta 4th order approximation:

$$x(t+1) = x(t) + \frac{(k1 + 2k2 + 2k3 + k4)}{6}, \tag{7}$$

where $k1 = f(t, x(t))$, $k2 = f(t + (h/2), x(t) + (h/2)k1)$, $k3 = f(t + (h/2), x(t) + (h/2)k2)$, and $k4 = f(t + h, x(t) + hk3)$.

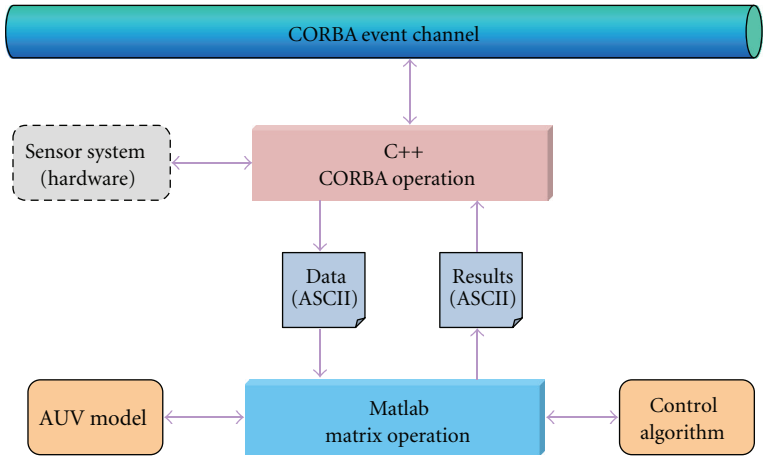

Figure 2: Matlab and C++ interfaces.

4. System Environment

In Figure 1, an OCP infrastructure for linear quadratic servo simulations using two nodes PC as test beds with a general 10 Mb Ethernet, PC1 as a server and PC2 as client, connected via middleware using CORBA event channel [9].

PC1 is used for running a Matlab simulation of the vehicle model and control algorithm, while PC2 is utilized for running a Matlab simulation as mission control station. Figure 2 illustrates an operation of interface between C++ and Matlab. Through this interface, the C++ collects data from PC2 through CORBA and creates data files in ASCII format. The Matlab first picks up the data saved by C++ and then performs matrix operations. The results are sent back to C++ in the ASCII file format, and the C++ continues to execute the communication task to send the results.

In Figure 3, a CORBA event service provides a flexible model for asynchronous and group communication among distributed objects. Consumers are the ultimate targets of events generated by suppliers. Suppliers and consumers can both play active and passive roles. An active push supplier pushes an event to a passive push consumer; passive pull supplier waits for active pull consumer to pull an event from it.

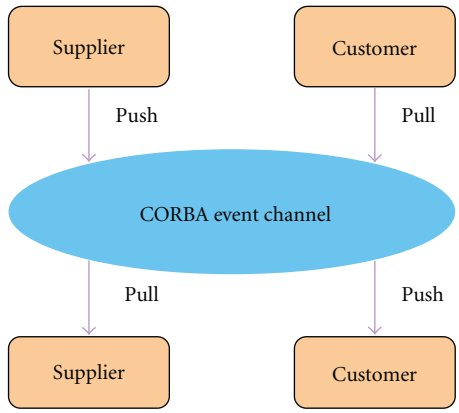

Figure 3: Event channel architecture.

5. Simulation Results

In this simulation, our objective is to control the AUV with speed at 2 m/sec, θ near zero, and depth z near -5 meters with 15 times counter duration, and then change a depth to -3 meters with 15 times counter duration more.

Then AUV returns to -5 meters depth with 15 times counter duration, and finally returns to -3 meters depth, as depicted in Figure 4. Assume 4° dive planes when pitch angle deviates to 5° from zero, the AUV reaches a depth of -5 feet with 0.95 feet deviation. Therefore, all terms in $Q \to 0$ and $R \to 0$, except $q_{11} = (4/57.2958)^{-2} = 205.21$, $q_{44} = (5/57.2958)^{-2} = 131.31$, and $r_{11} = (0.95)^{-2} = 110$.

In Figure 5, control input history is presented. It is based on depth reference input and output feedback. Compulsory

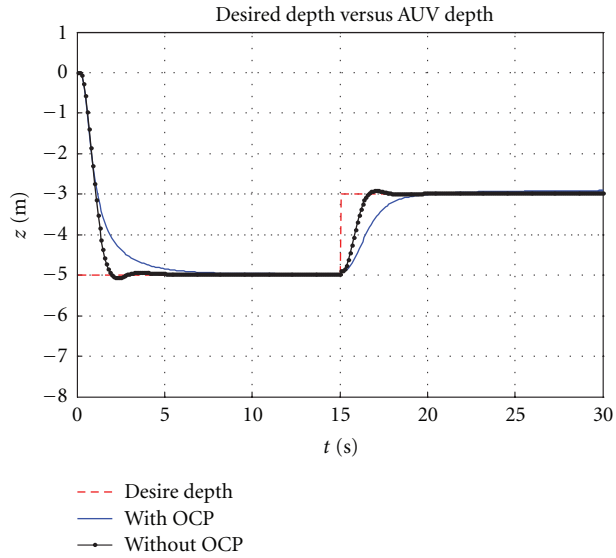

Figure 4: Depth control trajectory.

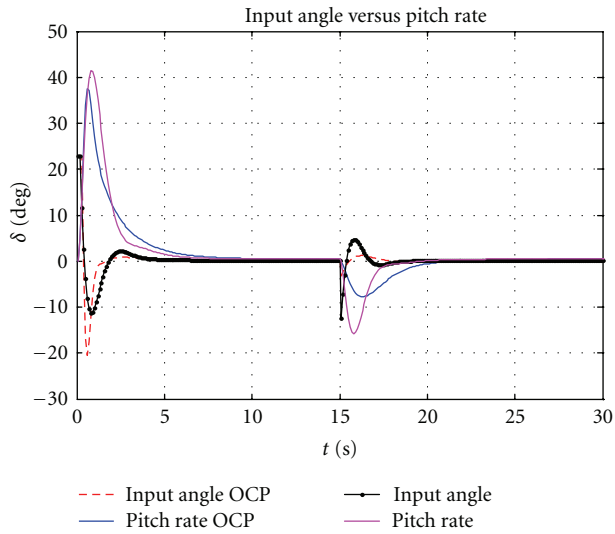

FIGURE 5: Input angle, pitch rate versus pitch angle.

control input value represents the differences between depth reference input and output feedback. The presented state variable history consists of input angle, pitch rate during moving, input angle with OCP, and pitch rate with OCP. We can also see that the control algorithm is optimal because of the input angle value is 12° and 14° to produce pitch rate to reach more than 40°/s and 15°/s; however by using OCP, input angle value is 20° and 5° to produce pitch rate to reach more than 36°/s and 8°/s.

The weighting matrices selection can be performed during run time in OCP environment. In order to increase the degree of machine intelligence, we perform a standard simulation using single weighting matrices, comparative analysis of property of gain matrix, we have $\|K\|_F = 0.4754$ using standard simulations, and $\|K\|_F = 0.5814$ with OCP.

6. Concluding Remarks

This paper presented a new approach of decentralized system environment using Matlab and CORBA event channel on several machines; we believe it will emerge more investigation in the current trends of real-time control system or bilateral control system.

Our proposed method captures the simulation results demonstrated that OCP could be used to provide the additional delivery method for distributing any navigation message among AUV implementation. Our future works would be conducted on additional experiments and measurements in this area, to increase application scalability to distribute the large navigation information.

Acknowledgments

This paper was financially supported by the Ministry of Knowledge Economy (MKE) and Korea Industrial Technology Foundation (KOTEF) through the Human Resource Training Project for Strategic Technology.

References

[1] T. Samad and G. Balas, *Software-Enabled Control: Information Technology for Dynamical Systems*, John Wiley & Sons/IEEE Press, Hoboken, NJ, USA, 2003.

[2] J. L. Paunicka, W. E. Corman, and B. R. Mendel, "A CORBA-based middleware solution for UAVs," in *Proceedings of the 4th International Symposium on Object-Oriented Real-Time Distributed Computing*, Germany, 2001.

[3] L. Wills, S. Kannan, B. Heck et al., "Open software infrastructure for reconfigurable control systems," in *Proceedings of the American Control Conference*, pp. 2799–2803, Chicago, Ill, USA, June 2000.

[4] K. P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru, and G. A. Demetriou, "Control architectures for autonomous underwater vehicles," *IEEE Control Systems Magazine*, vol. 17, no. 6, pp. 48–64, 1997.

[5] The Society of Naval Architects and Marine Engineers, "Nomenclature for treating the motion of a submerged body through a fluid," *Research Bulletin*, no. 1–5.

[6] J. S. Riedel, *Pitchfork bifurcations and dive plane reversal of submarines at low speeds [Engineer's Thesis]*, Naval Postgraduate School, Calif, USA, 1993.

[7] R. Cristi, F. A. Papoulias, and A. J. Healey, "Adaptive sliding mode control of autonomous underwater vehicles in the dive plane," *IEEE Journal of Oceanic Engineering*, vol. 15, no. 3, pp. 152–160, 1990.

[8] A. J. Healey, F. A. Papoulias, and R. Cristi, "Design and experimental verification of a model based compensator for rapid AUV depth control," in *Proceedings of the 6th International Symposium on Unmanned Untethered Submersible Technology*, pp. 458–474, Washington, USA, June 1989.

[9] D. C. Schmidt, "An overview of the real-time CORBA specification," *Computer*, vol. 33, no. 6, pp. 56–63, 2000.