The design of an embedded multi-sensor data fusion system for unmanned surface vehicle navigation based on real time operating system

Wenwen Liu
Mechanical Engineering
University College London
London, UK
w.liu.11@ucl.ac.uk

Yuanchang Liu
Mechanical Engineering
University College London
London, UK
yuanchang.liu.10@ucl.ac.uk

Rui Song
Mechanical Engineering
University College London
London, UK
r.song.11@ucl.ac.uk

Richard Bucknall
Mechanical Engineering
University College London
London, UK
r.bucknall@ucl.ac.uk

***Abstract*—This paper describes the design and implementation of a practical multi-sensor data fusion system for unmanned surface vehicle (USV) navigation. The system employs an embedded Linux board as the main on-board control module to extract and preprocess raw measurements from various navigational sensors using the real time operating system (RTOS). An unscented Kalman Filter (UKF) based data fusion algorithm has been developed to fuse the obtained and preprocessed sensor measurements and provide more reliable and accurate estimations of USV's navigational data in real time. The results demonstrate the effectiveness of the data fusion algorithm in reducing unpredicted errors of a standalone sensor.**

Keywords—multi-sensor data fusion, real time operating system, unmanned surface vehicle

I. INTRODUCTION

Unmanned Surface Vehicles (USVs) have gained increasing attention due to their capabilities of completing various missions of ocean exploitation. While undertaking these missions, the autonomous navigation of USV including real-time self-localization is crucial. Benefiting from the advance of navigational devices such as the Global Positioning System (GPS) and other marine electronics, the essential navigational data, i.e. positions, velocities and headings of an USV can be measured. However, sensors on their own still suffer from signal lost and uncertainties due to the environment disturbances and equipment limitations [1]. To address these issues, one popular approach is to use the multi-sensor navigation method by integrating various sensors as complementary devices.

Multi-sensor data fusion system for real-time navigation is a complex information processing system. With various navigational sensors that have different functions and different sampling times on-board the USV, the system should be capable of synchronizing all the sensors, acquiring different sensor measurements, and processing the measurements for further use. Moreover, achieving real-time localization leads to critical time constraints for the system, especially when the sensor measurement is sensitive to the sampling time. Therefore, a Real Time Operating System (RTOS) on a Single Board Computer (SBC) is more applicable to such data fusion system. Compared to General Purpose Operating System (GPOS), RTOS has faster response time and it is a dedicated system that only processes assigned tasks. In addition, the use of RTOS based SBC brings benefits such as cost-effective and low power consumption.

Processing the raw measurements from sensors is the core to implement the sensoring system for USV navigation. Currently, Kalman filter (KF) has been widely used in sensor data fusions [2]-[5]. However, the multi-sensor integration introduces nonlinearity, which is beyond the capability of the conventional KF. Researchers are perusing different variants of KF such that vehicles can robustly operate in various environments under different dynamic constraints. Motwani [6] developed an interval Kalman filter "(IKF) for the heading estimations of an uninhabited surface vehicle. Jose [7] has her sights on Extended Kalman filter (EKF). Xu [8] developed a low-cost sensors system for a USV's cooperative navigation and localization based on an unscented Kalman filter (UKF) techniques. According to Merwe and Wan [9], UKF can achieve an approximate 30% error reduction performance when comparing to the EKF.

Therefore, a dedicated embedded multi-sensor system based on real time operating system including a UKF based data fusion algorithm has been proposed to implement the real-time self-localization for USVs navigation.

This paper is organized as follows. Section II describes a typical autonomous navigation system of an USV. Both hardware and software implementations of the proposed embedded multi-sensor data fusion system are demonstrated in section III. The details of the UKF based data fusion algorithm are presented in section IV. Section V shows the test results and discussions, which is followed by the conclusion in section VI.

This work was sponsored by ACCeSS group under grant N0014-03-0160.

II. Navigation System of an Unmanned Surface Vehicle

The navigation system of an USV should include three different aspects, i.e. navigational data acquisition, path planning and autopilot, to enable fully autonomous operation. As shown in Fig.1, the first step to navigate an USV is to extract various raw sensor measurements and apply data fusion techniques to obtain reliable navigational data. Based on such information, a path planning algorithm can then generate a safe trajectory with a set of waypoints to guide the USV. Finally, an autopilot is used to ensure that the USV adheres to the generated trajectory by controlling its propulsion system. This research aims to develop a practical embedded multi-sensor data fusion system that is able to implement the data acquisition feature.

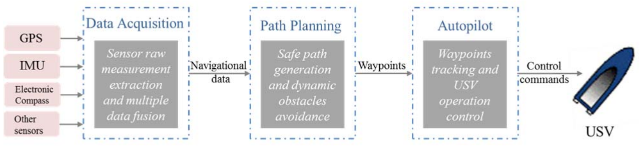
GPS
IMU
Electronic Compass
Other sensors
Data Acquisition
Sensor raw measurement extraction and multiple data fusion
Navigational data
Path Planning
Safe path generation and dynamic obstacles avoidance
Waypoints
Autopilot
Waypoints tracking and USV operation control
Control commands
USV

Fig. 1 Autonomous navigation system for an USV.

III. Implementation of the Embedded Sensoring System

A. Hardware Implementation

In order to measure the required navigational data of an USV, such as positions and headings, various sensors are employed: a GPS module that provides absolute measurements of the USV's locations in longitude/latitude; an IMU module that can measure the USV's acceleration and rotation rate in short terms; and an electronic compass that is able to provide absolute measurements of USV's headings. An embedded Linux board called Pandaboard ES is employed to connect all the aforementioned sensors. A control computer is used as the user interface to monitor on-board sensor measurements and operates the data fusion algorithms. Therefore, as the on-board hosting platform, Pandaboard ES's tasks include interfacing to each sensor, acquiring sensors' data and communicating with the control computer. The connections of the hardware system are illustrated in Fig.2. The navigational sensors are connected to the on-board hosting platform via wired serial communication whereas the communication between the control computer and the Pandaboard ES uses the wireless Wi-Fi connection.

B. Software Implementation

According to the hardware installations that demonstrated in Fig.2, three aspects of the software implementation should be considered: sensor data extraction, wireless communication and operating system of on-board hosting platform.

1) Sensor data extraction and pre-processing

Digital interfaces for each sensor is required to obtain sensor signal and extract useful information from their output signals. Most of the marine electronic devices support the National Marine Electronics Association (NMEA) 0183 standard [10]. Accordingly, measurements of the GPS module and the electronic compass can be easily extracted referring to the standard NMEA0183 sentences. The IMU module features an ATMega328 microcontroller [11] and its output format can be edited by the user.

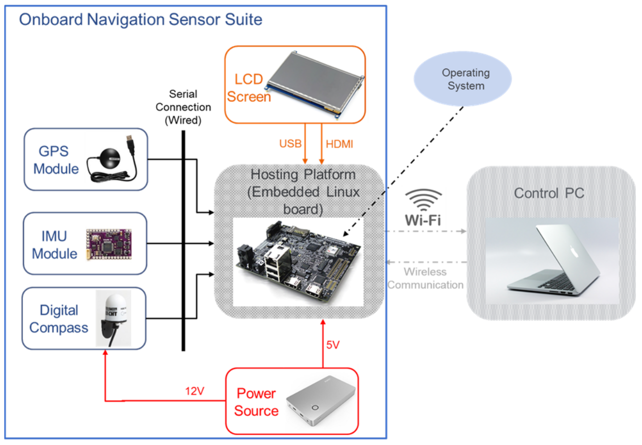

Fig. 2 Hardware connections.

The absolute measurements obtained from the GPS module and the electronic compass are along the earth frame and can be easily converted to a pre-designed navigation frame. Whereas, the acceleration rates provided by IMU are along the inertial frame. It therefore should convert the IMU data from the inertial frame to the same navigation frame. As shown in Figure 3, the pre-designed navigation frame uses the North direction as y axis and the east direction as x axis. The inertial frame can be approximated to the body frame when inertial sensors are installed in the center of the USV. The IMU data can then be converted by applying the rotation matrix as blew:

$$\begin{bmatrix} a_{nx} \\ a_{ny} \end{bmatrix} = \begin{bmatrix} \cos\emptyset & -\sin\emptyset \\ \sin\emptyset & \cos\emptyset \end{bmatrix} \begin{bmatrix} a_{bx} \\ a_{by} \end{bmatrix} \tag{1}$$

where $\emptyset$ is the rotation angle from body frame (inertial frame) to the navigation frame, which is equal to the USV's heading. a_{nx}, a_{ny} are the USV's acceleration in navigation frame and a_{bx}, a_{by} are the USV's acceleration in body frame.

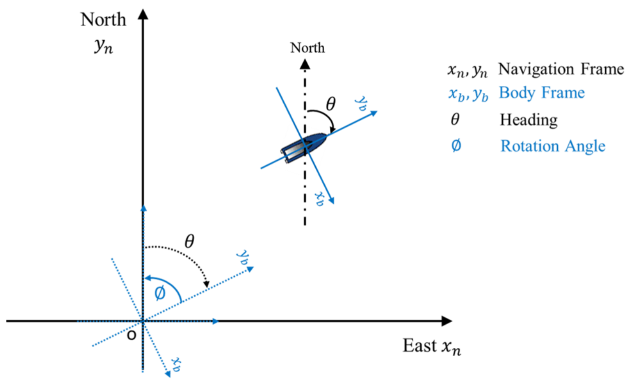

Fig. 3 Coordinate frames.

2) *Wireless communication*

Pandaboard ES features a wireless local network (WLAN)/Bluetooth module that provides a Bluetooth interface and a 2.4 GHz WLAN 802.11b/g/n interface [12], where 802.11 is the IEEE Wi-Fi Standard. Therefore, a wireless connectivity between the on-board hosting platform and the offshore control PC can be established via the Wi-Fi by building a wireless base station, called the access point (AP). After enabling the Pandaboard as a wireless AP and assigning a static internet protocol (IP) address, the control PC can connect to it and establish a bi-directional wireless communication via the socket programming. A socket program involves at least a pair of client and server. Here Pandaboard acts as the Server that waits for the connection request from the control PC (Client) and sends the sensor data when requested. The relationships and connections of the main devices are illustrated in Fig.4.

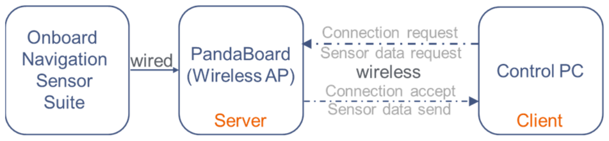

Fig. 4 Relationship and communications between main devices.

3) *Operating system*

Generally, an operating system (OS) is a group of programs that can accomplish different tasks on a computer. It is the key to enable the hardware functions. There are three advantages use an OS, it can reduce the difficulties of the application development, it can increase the readability and portability. A general purpose OS (GPOS), such as Windows, Mac OS and Linux is able to accomplish most of the common applications. However, it require higher-end hardware and as such a complex OS, its CPU are always occupied by a number of threads and processes. In comparison, a real time OS (RTOS) is a dedicated system that only operates the designed tasks according to their priority levels. Therefore, the fast response time on a low cost lower-end hardware with low power consumption makes the RTOS ideal in unmanned vehicle applications.

The Pandaboard ES uses an OMPA4460 processor that features a dual-core ARM Cortex-A9 CPU [12]. Hence, both GPOS and RTOS can be installed. In this study, the FreeRTOS, an open source real time kernel, is used. It features preemptive scheduler so that system can decide whether to suspend the current task by evaluating the priority order of the incoming task. [3] By integrating the RTOS, the system is then able to assign those tasks in adjustable priority level and execute each task in a limited time.

Fig.5 demonstrates the design aspects of a dedicated freeRTOS based system. The freeRTOS kernel creates different tasks and defines their priority level (PL) to meet the requirements of the designed system. The tasks will have four states during the operation, ready, running, suspension or blocked. All the new tasks are placed in Ready lists according to their PL. The kernel will have to maintain the state of each task automatically [13].

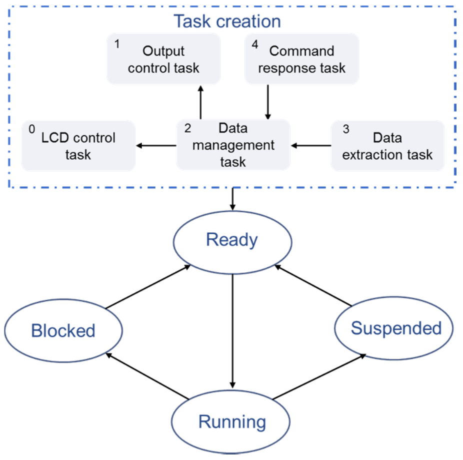

Fig. 5 Design aspects of the freeRTOS for the designed tasks.

The designed tasks are listed below in their priority order, 0 indicates the lowest priority and the priority becomes higher as the number increases.

- Command response: (PL=4) this task has the highest level of priority since it takes commands from the control computer, such as start/stop obtaining sensor signals;
- Data acquisition: (PL=3)after receiving the start command, system starts to read data from sensors via the serial communication;
- Data management: (PL=2) this task is to extract useful information from raw sensor measurements and prepare them in a designed format for future use;
- Output control: (PL=1) this task is to send back the preprocessed and formatted data back to control computer;
- LCD control: (PL=0) a user interface to display the progress of the developed system. It won't affect the whole loop to share sensor data with the control computer. Hence, is has the lowest level of priority.

All lower level tasks will be suspended when encounters a higher level task. The sleep time and execution time for each task are carefully designed. Table I presents useful codes to develop the dedicated RTOS based embedded sensoring system [14].

TABLE I MAIN CODES USED TO IMPLEMENT THE DEDICATED RTOS

Code	Descriptions
xTaskCreate()/xTaskCreateStatic()	Create a task
define Taskname_TASK_PRIO	Assign priority order
vTaskSuspend()	Suspend a task
vTaskResume()	Resume a task
xTimeToWake	Designed wake time of a task
xTimeIncrement	Designed time increment of a task

IV. Data Fusion Algorithm based on Unsecented Kalman Filter

Position, velocity and headings are essential navigational data to locate the USV itself during autonomous navigation. Thus, the desired system vector **x** is defined as following:

$$\mathbf{x}=[p_x \quad p_y \quad v_x \quad v_y \quad \theta]^T \tag{2}$$

where p_x, p_y are the USV's position, v_x and v_y are the velocity and θ is the heading.

As aforementioned, the IMU measures the USV's motion in the inertial navigation frame, which can be approximated to the body frame, whereas the absolute measurements obtained from GPS module and electronic compass are along the designed navigation frame. Considering the nonlinearity error may occur by the frame conversion of the IMU measurements, the UKF is employed and the system vector is governed by (3), and the non-linear dynamic model of the system is demonstrated in (4).

$$\mathbf{x}(k)=f(\mathbf{x}(k-1),\mathbf{w}(k-1)) \tag{3}$$

$$\dot{f}(x)=\begin{pmatrix}\dot{p}_x\\ \dot{p}_y\\ \dot{v}_x\\ \dot{v}_y\\ \dot{\theta}\end{pmatrix}=\begin{pmatrix}v_x\\ v_y\\ \cos\theta\times a_{ix} - \sin\theta\times a_{iy}\\ \sin\theta\times a_{ix} + \cos\theta\times a_{iy}\\ \omega\end{pmatrix} \tag{4}$$

where **w** is the system process noise with the covariance **Q**; a_{ix}, a_{iy} ω are the IMU measured accelerations and rotation rate, respectively.

The absolute measurements from GPS module and electronic compass creates the system measurement model in (5).

$$\mathbf{y}(k)=\mathbf{H}\mathbf{x}(k)+\mathbf{v}(k)=\begin{bmatrix}1&0&0&0&0\\0&1&0&0&0\\0&0&0&0&1\end{bmatrix}\mathbf{x}(k)+\mathbf{v}(k) \tag{5}$$

where **v** is the system measurement noise with the covariance **R**;

For a n=5 dimensional variable **x**, the UKF employs the unscented transformation to form a set of 2n+1 weighted points χ using the following equations (6)-(8). [15]

$$\boldsymbol{\chi}_0(k-1)=\mathbf{m}(k-1) \tag{6}$$

$$\boldsymbol{\chi}_i(k-1)=\mathbf{m}(k-1)+\sqrt{n+\lambda}\times[\sqrt{\mathbf{P}_i(k-1)}] \tag{7}$$

$$\boldsymbol{\chi}_{i+n}(k-1)=\mathbf{m}(k-1)-\sqrt{n+\lambda}\times[\sqrt{\mathbf{P}_i(k-1)}],\ i=1,\dots,n \tag{8}$$

The generated sigma points are then propagated through the non-linear dynamic model (4) as following:

$$\hat{\boldsymbol{\chi}}_i(k)=f(\boldsymbol{\chi}_i(k-1)),\ i=0,\dots,2n \tag{9}$$

The predicted mean $\mathbf{m}^-$ and covariance $\mathbf{P}^-$ can be calculated by assign the weight for each sigma points as shown in (10) and (11).

$$\mathbf{m}^-(k)=\sum_{i=0}^{2N} W_i^m \hat{\boldsymbol{\chi}}_i(k) \tag{10}$$

$$\mathbf{P}^-(k)=\sum_{i=0}^{2N} W_i^c (\hat{\boldsymbol{\chi}}_i(k)-\mathbf{m}^-(k))(\hat{\boldsymbol{\chi}}_i(k)-\mathbf{m}^-(k))^{\mathbf{T}}+\mathbf{Q}(k-1) \tag{11}$$

where W_0^m, W_0^c, W_i^m and W_i^c are associated constant weights that can be computed by (12)-(14).

$$W_0^m = \lambda/(n+\lambda) \tag{12}$$

$$W_0^c = \frac{\lambda}{(n+\lambda)}+(1-\alpha^2+\beta) \tag{13}$$

$$W_i^m = W_i^c = 1/2(n+\lambda),\ i=1,\dots,2n \tag{14}$$

where $\lambda=\alpha^2(n+\kappa)-n$ is a scaling parameter. The parameters α and κ determine the spread of the sigma points around the mean. β describes the distributed information, of which the optimal value is 2 for Gaussian distribution.

In order to compare the predicted mean $\mathbf{m}^-$ with the measurement, it has to be propagated through the measurement model (5). Therefore, a new set of 11 Sigma points $\boldsymbol{\chi}^-$ should be computed as (15)-(19);

$$\boldsymbol{\chi}_0^-(k)=\mathbf{m}^-(k) \tag{15}$$

$$\boldsymbol{\chi}_i^-(k)=\mathbf{m}^-(k)+\sqrt{n+\lambda}[\sqrt{\mathbf{P}_i^-(k)}] \tag{16}$$

$$\boldsymbol{\chi}_{i+n}^-(k)=\mathbf{m}^-(k)-\sqrt{n+\lambda}[\sqrt{\mathbf{P}_i^-(k)}],\ i=1,\dots,n \tag{17}$$

$$\hat{\mathbf{Y}}_i(k)=\mathbf{H}\times\boldsymbol{\chi}_i^-(k),\ i=0,\dots,2n \tag{18}$$

$$\boldsymbol{\mu}^-(k)=\sum_{i=0}^{2N} W_i^m \hat{\mathbf{Y}}_i(k) \tag{19}$$

The innovation covariance $\mathbf{S}^-$ and the cross covariance **C** are calculated in (20) and (21).

$$\mathbf{S}^-(k)=\sum_{i=0}^{2N} W_i^c (\hat{\mathbf{Y}}_i(k)-\boldsymbol{\mu}^-(k))(\hat{\mathbf{Y}}_i(k)-\boldsymbol{\mu}^-(k))^T+\mathbf{R}(k) \tag{20}$$

$$\mathbf{C}(k)=\sum_{i=0}^{2N} W_i^c (\hat{\boldsymbol{\chi}}_i(k)-\mathbf{m}^-(k))(\hat{\boldsymbol{\chi}}_i(k)-\mathbf{m}^-(k))^T \tag{21}$$

The Kalman filter gain and the estimated mean and covariance at time step k can be obtained.

$$\mathbf{K}(k)=\mathbf{C}(k)\mathbf{S}^-(k)^{-1} \tag{22}$$

$$\mathbf{m}(k)=\mathbf{m}^-(k)+\mathbf{K}(k)[\mathbf{y}(k)-\boldsymbol{\mu}(k)] \tag{23}$$

$$\mathbf{P}(k)=\mathbf{P}^-(k)-\mathbf{K}(k)\mathbf{S}^-(k)\mathbf{K}(k)^{\mathbf{T}} \tag{24}$$

System iterates at each time step to update the real-time navigational data of the USV and pass the data to the path planning algorithm to generate a safe path in a dynamic environment.

V. Simulation Results

A USV navigating operation is simulated to operate in a practical two-dimensional maritime environment. Two scenarios are considered to validate the performance of the developed UKF based data fusion algorithm.

1) Scenario 1. For most of the maritime missions, vessels are not required to make frequent motion changes by keeping their speeds and headings. Therefore, in the first scenario, the autonomous USV is assumed to come out from Southampton harbour (UK) to east Cowes as shown in Fig. 6 and maintain a constant velocity to follow a straight line trajectory.

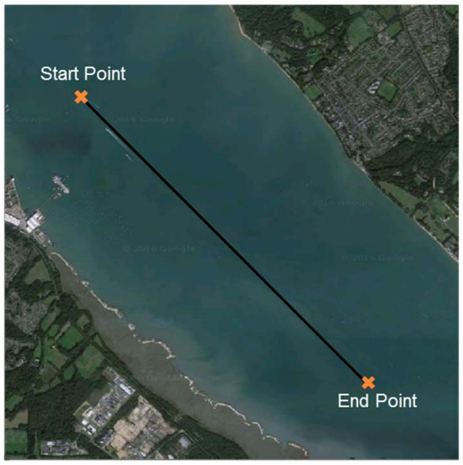

Fig. 6 Scenario 1: testing environment in Southampton east Cowe UK.

Fig. 7-9 have shown how the UKF based algorithm improves raw measurements of the GPS and subsequently provides robust localization capability. Fig. 7 presents the whole simulated USV trajectory, shown as the black line and the GPS raw measurements as the blue dots scattered around the simulated trajectory (black line) subject to the predefined variance. The green trajectory indicates the fused results of UKF based algorithm. It can be seen that the fused trajectory is very close to the true trajectory from the enlarged figure. Fig. 8 illustrates the heading results, compared to the compass measurements (blue line), it can be seen clearly that the fused headings (green line) are closer to the heading that is maintained by USV during the simulation. Therefore, the UKF based algorithm is proved to be efficiently to reduce the errors of standalone sensors. Such a statement can also be validated from the results shown in Fig. 9, which records the rooted mean square error (RMSE) of the navigational data. The green lines that indicate the RMSE of the fused data are all lower than the blue lines, which presents the RMSE of raw sensor measurements.

2) Scenario 2. After showing the improvement by applying the UKF in the simple straight line following mission, the algorithm is then tested in Scenario 2 to deal with more complex motions of the USV. The vehicle is modelled to operate along the coastline of Solent, UK as shown in Fig. 10, where the USV has to make two turnings to accomplish the mission.

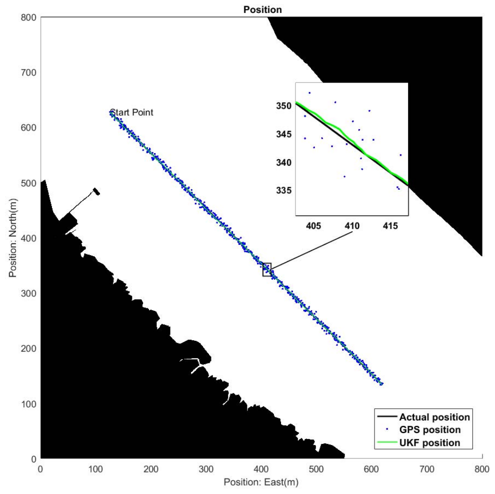

Fig. 7 The whole trajectory of simulation Scenario 1.

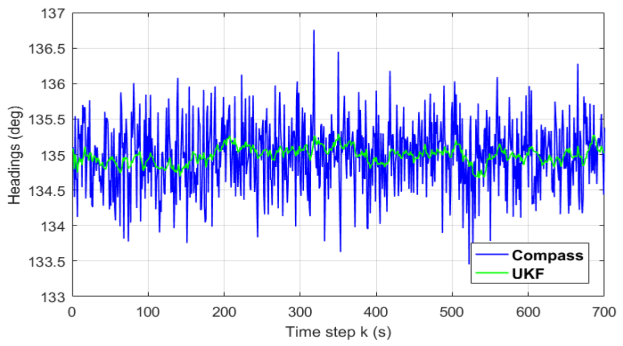

Fig. 8 The compass headings and fused headings of simulation Scenario 1.

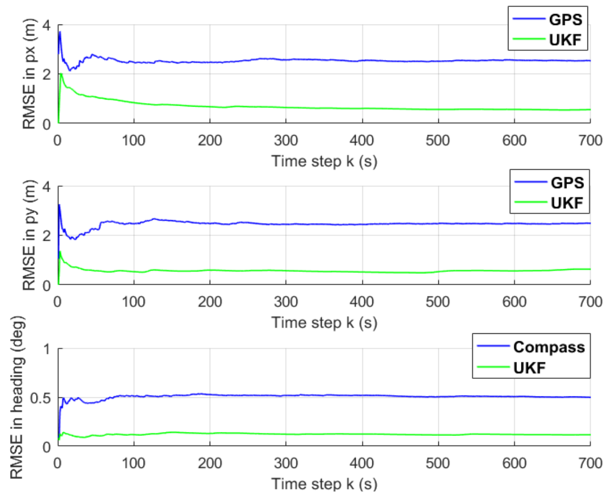

Fig. 9 RMSE of USV's position and heading of simulation Scenario 1.

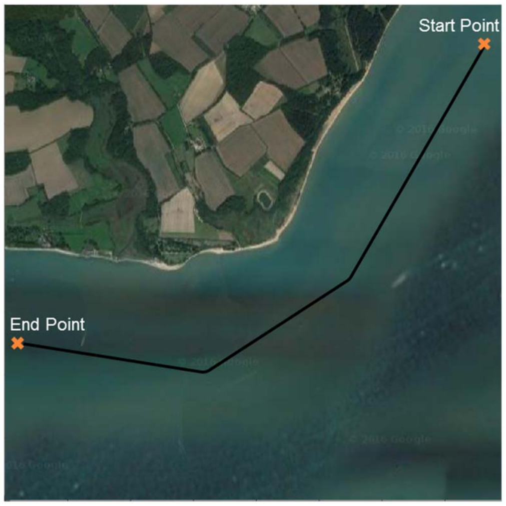

Fig. 10 Scenario 2: testing environment at Solent UK.

The simulation results of Scenario 2 are presented in Fig. 11-13. Fig. 11 presents the whole trajectory of the USV. The fused trajectory (green line) reduces the error of GPS raw measurements and follows the true trajectory closely during the whole operation. It can be seen that even the USV make two turnings, the data fusion algorithm is still capable to achieve the same performance as it does in simulation Scenario 1. Fig. 12 shows clearly that the fused headings have less error and more stable when the USV maintains the same direction. The RMS values of USV's navigational data are demonstrated in Fig. 13. Similar to simulation Scenario 1, the errors are largely reduced by the data fusion algorithm and the reduction ratio is achieved approximately 50%.

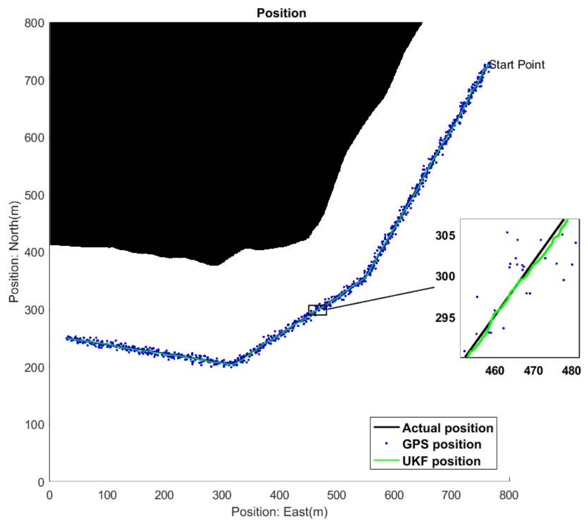

Fig. 11 The whole trajectory of simulation Scenario 2.

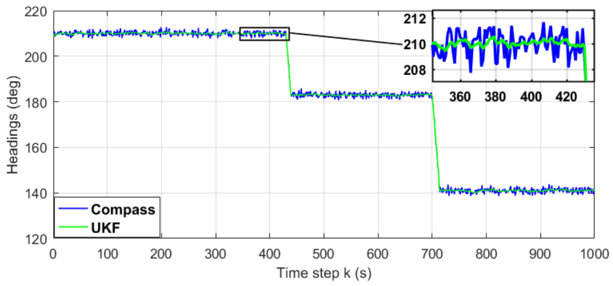

Fig. 12 The compass headings and fused headings of simulation Scenario 2.

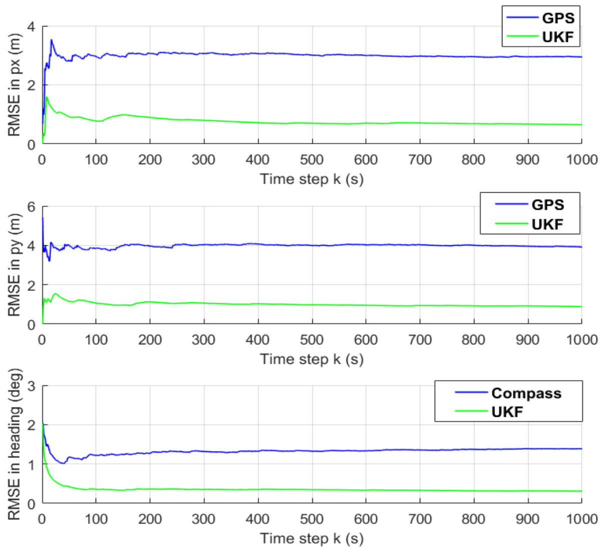

Fig. 13 RMSE of USV's position and heading of simulation Scenario 2.

VI. Conclusion

In this paper, an embedded real time multi-sensor data fusion system has been designed and implemented for USVs' navigation. The design put forward the idea of using Kalman filtering technology to improve the accuracy of raw measurements from navigational sensors, which is proved by two different simulations that are carried out in practical environments. The implemented freeRTOS based sensoring system doesn't show significant improvement of its working performance on extracting raw sensor data and communicating with the control computer. But the potential of such system is foreseen to port the large amount codes of the data fusion algorithm, the path planning algorithm and the control algorithm to the on-board hosting platform in the future.

References

[1] P.D. Groves, Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems, Boston: Artech House, 2008

[2] J. Bijker and W. Steyn, "Kalman filter configurations for a low-cost loosely integrated inertial navigation system on an airship," Control Engineering Practice, vol. 16(12), pp. 1509-1518, 2008.

[3] V. Subramanian, T.F. Burks, and W.E. Dixon, "Sensor Fusion using Fuzzy Logic Enhanced Kalman Filter for Autonomous Vehicle Guidance in Citrus Groves', American Society of Agricultural and Biological Engineers, vol. 52(5), pp. 1411-1422, 2009.

[4] A. Stateczny and W. Kazimierski, "Multisensor Tracking of Marine International Journal of Electronics and Telecommunications, vol. 57(1), pp. 65-70, 2011.

[5] O. Maklouf, A. Ghila, A. Abdulla, and A. Yousef, "Low Cost IMU/GPS Integration using Kalman Filtering for Land Vehicle Navigation Application," International Journal of electrica, Computer, Electronics and Communication Engineering, vol. 7(2), pp. 117-123, 2013.

[6] A. Motwani, S.K. Sharma, R. Sutton, and P. Culverhouse, "Interval Kalman Filtering in Navigation System Design for an Uninhabited Surface Vehicle," Journal of Navigation, vol. 66, pp. 639-652,2013.

[7] A. R. Jose, and E. White, "Fusion Filter Algorithm Enhancements for a MEMS GPS/IMU," in Proceedings of the 14th International Technical Meeting of the Satellite Division of the Institute of Navigation, United States, pp. 1382-1393, 2001.

[8] B. Xu, J. Bai, G. Wang, Z. Zhang, and W. Huang, "Cooperative Navigation and Localization for Unmanned Surface Vessel with Low-cost sensors," in Inertial Sensors and Systems Symposium, Germany, pp.19-33, September 2014.

[9] R. V. D. Merwe, and E. A. Wan, "Sigma-point Kalman Filters for Integrated Navigation," in Proceedings of the 60th Annual Meeting of the Insititute of Navigation, United States, pp.641-654, June 2004.

[10] Tronico, "The NMEA 0183 Protocol," 2015.

[11] Ardu-imu, "Introduction to Arduimu V3," 2014.

[12] Pandaboard.org, "OMAP4460 Pandaboard ES System Reference Manual," September 2011.

[13] R. Goyette, "An Analysis and Description of the Inner Workings of the FreeRTOS Kernel," April 2007.

[14] F. Guan, L. Peng, L. Perneel, and M. Timmerman, "Open source FreeRTOS as a case study in real-time operating system evolution," Journal of Systems and Software, vol.118, pp. 20-35, 2016.

[15] E. A. Wan, and R. V. D. Merwe, "The Unsented Kalman Filter for Nonlinear Estimation"