Fault Diagnosis Challenge in a Flight-Class Autonomous Underwater Vehicle

Juhan Ernits[1], **Richard Dearden**[1], **Miles Pebody**[2]

[1] *School of Computer Science, University of Birmingham, B15 2TT, UK*
{j.ernits,rwd}@cs.bham.ac.uk
[2] *National Oceanography Centre, Southampton, UK*
M.Pebody@noc.ac.uk

Abstract

Advanced AUVs that are capable of long duration missions are becoming increasingly common. However, making the vehicles reliable is a significant challenge, and fault detection has an important role to play in achieving this. To enhance the state of the art we present the data of a selection of Autosub 6000 missions. The data is given in DXC format with known faults injected into the logs.

1 INTRODUCTION

Autonomous Underwater Vehicles (AUVs) have become an increasingly important tool in a number of applications over the recent years. Due to improvements in battery, computing, mechanical and sensing technology the missions they carry out have become longer and take place in less well known environments. Thus improving the reliability of the vehicle becomes more and more important. While the greatest gains will probably be made with more reliable hardware and control systems, and in designing missions to be safer (Griffiths and Brito, 2008), we believe that there is also a place for on-board diagnosis of these vehicles to detect faults when they occur and take appropriate actions to mitigate their effects. The environment these vehicles operate in, including extreme pressure, the corrosive effects of seawater, and the risk of damage due to waves while on the surface and collisions during launch and later, is too harsh to expect even the best-designed components to operate flawlessly over repeated missions.

Autosub 6000 (McPhail, 2009) is an AUV which currently has the capability to operate for up to 27 hours (McPhail *et al.*, 2010b) at a time at depths of up to 6000m to collect science data from the deep ocean. During more than 400 previous scientific missions, Autosub 6000 and its predecessors have suffered both near losses and one actual loss. In two cases the AUV has been recovered using a remotely operated vehicle at significant expense. In one case the Autosub2 AUV was permantly lost 17Km under the 200m thick Fimbul Ice Shelf in the Antarctic. There are numerous cases of missions that have had to be aborted but where recovery was possible by the operations team and the attending support ship. Based on the experience of operating the Autosub AUVs a project to apply automated diagnosis and recovery methods for Autosub 6000 was initiated with the primary focus being on the detection of faults that may result in collisions with the seabed. Collision with the seabed is undesirable because it has been demonstrated to have been one of the primary causes of vehicle loss.

The paper is organised as follows: Section 2 contains a more detailed description of Autosub 6000. Section 3 specifies the variables and commands present in the system. Section 4 summarises the diagnosis challenge for Autosub 6000.

1.1 Related Work

There has been a long history of work on fault detection and fault-tolerant control for AUVs. As with most vehicles, Autosub 6000 and its predecessors have the simplest of these, an emergency abort system (McPhail and Pebody, 1998) that is triggered by a variety of events on the vehicle such as critical subsystems being unresponsive, or the vehicle exceeding its maximum depth limit.

A survey of fault detection and fault tolerance schemes for AUVs and ROVs has been presented in (Antonelli, 2006). Many of these are model-based approaches, but tend to be only for restricted subsystems, for example the actuators in (Alessandri *et al.*, 1999). The models in that case are continuous, and faults consist of changes to the parameters of the system. Such approach can be considered component level. System level diagnosis for Autosub 6000 has been proposed in (Ernits *et al.*, 2010a; Ernits *et al.*, 2010b).

2 AUTOSUB 6000

Autosub 6000 (McPhail, 2009) is a flight class AUV that is used for various research missions as part of the National Marine Equipment Pool of the UK.

The AUV has a network variable based control system (McPhail and Pebody, 1998) where sensor readings, events and commands are output to the network. Each component subscribes to a subset of the variables available in the system.

The AUV has three main actuators: the motor used for propulsion, the rudder for turning left and right, and the stern plane for controlling pitch and vertical movement. In addition there are abort weight release nodes that when triggered release abort weights that result in increasing the positive buoyancy of the AUV and causing it to surface.

The stern plane and rudder actuators have feedback position sensors based on a potentiometre. The motor control node provides access to propeller RPM, motor current, propeller torque, and propeller power variables.

The navigation of the vehicle is based on several sensors: GPS when the vehicle is on surface; IXSEA PHINS Intertial Navigation System (INS) based on fibre optic gyros; Acoustic Doppler Current Profiler (ADCP) that provides ground speed when the sea floor is within the 200m range of the sensor and water speed otherwise. In addition it provides vehicle pitch and orientation information. The depth is measured by a Digiquartz depth sensor. The combination of the INS and ADCP yields a 5m per hour drift accuracy. The pressure sensor has 0.01% accuracy. The vehicle is also equipped with a front-looking collision avoidance sonar (McPhail *et al.*, 2010a).

The vehicle is controlled by setting demands, e.g. a depth demand, to which the appropriate control node responds by sending commands to actuators to achieve the goal. In addition to the master control node that interprets the mission script, there are three nodes: depth control node, position control node and motor control node.

One of the recent application scenarios of Autosub 6000 is described in (McPhail *et al.*, 2010b). It gives an overview of the circumstances the AUV may find itself in and how such problems are overcome.

3 DATA SPECIFICATION

The data is divided up into missions. Missions are grouped into campaigns which correspond to deployments of research vessels carrying Autosub 6000 with some specific purpose. For example, campaign JC027 and D343 were development campaigns for Autosub 6000. Campaign JC044, on the other hand was a full fetched research application where the AUV was used for locating underwater thermal vents (McPhail *et al.*, 2010b).

The core of the vehicle stays the same from campaign to campaign, but there are slight changes. For example, the front looking obstacle avoidance sonar was added and tested during D343. It was in active use in JC044.

The data is provided in DXC scenario format. The variables are listed in Table 1 and the commands in Table 2. The data is available from `http://www.cs.bham.ac.uk/go/afda`.

The enumerations in Table 1 take the following values:

MsnCtrlStatEnum : Stopped (the system is stopped), RunningMain (the system is running), Paused (the system is paused), Aborted (mission has been aborted), Surfacing (in surfacing mode), Homing (reacting to a homing signal), RunningTerm1 (running termination script 1), RunningTerm2 (running termination script 2).

MCLastEventEnum is an enumeration type which is typically used for a variable that is output of the node executing the mission script.

MCLastEventEnum : Null (placeholder), Always (always true), CmdStart (start command received from surface vessel), CmdStop (stop command received from surface vessel), CmdAbort (abort command received), CmdPause (pause command received), EndOfScript (mission script completed), StartScript (start executing mission script), GotGPSFix (GPS fix received), GotPos (got position specified by a position demand), GotDepth (got depth specified by the depth demand), DepthGT (depth greater than current demand), DepthLT (depth less than currend demand), Dived (vehicle has dived), OnSurface (vehicle on surface), TooDeep (vehicle deeper than the set maximum depth), MissionLineTimeout (timeout exceeded for achieving a goal in mission script), PowerLow (low power), HomingSignalDetected (detected homing signal), CmdStartTerm1 (starting termination script 1), CmdStartTerm2 (starting termination script 2), CollisionAvoided (collision successfully avoided resuming previous behaviour), ObstacleAvoided (obstacle successfully avoided resuming previous behaviour), AvoidanceFailed (collision avoidance failed, giving up).

4 THE DIAGNOSIS CHALLENGE

A consistency based fault diagnosis solution for the depth control system is provided in (Ernits *et al.*, 2010b). In addition to diagnosing faults in the depth control system, the approach provides a solution to overcome the unobservability of the commands issued in the control node. This approach is used to reduce the technical complexity of the data described in the previous section. An analysis of faults that have occurred during 400+ missions of Autosub 6000 and its predecessors is provided in (Ernits *et al.*, 2010a). Yet several interesting and important avenues of research remain to be pursued, for instance:

- Automated learning of the nominal behaviour in various control modes;
- Application of stochastic model-based diagnosis;
- Comparison of alternative consistency based and stochastic diagnosis methods;
- Diagnosis of other subsystems of the AUV;

Autosub 6000 provides a unique opportunity for the automated fault diagnosis community: it is an AUV built for research purposes that is well documented and in active use as it is part of the National Marine Equipment Pool of the UK. The latter means that the vehicle is available on demand for research groups. Thus it is potentially possible to add the data of new missions to the repository. On the other hand successful demonstrations of fault diagnosis on the recorded data of real life missions of Autosub 6000 will impact the development of future AUVs where increased autonomy becomes an essential enabling technology.

Table 1: The list of sensor readings in Autosub 6000

Variable name	Type	Unit	Description
MsnCtrlStat	MsnCtrlStatEnum	-	Mission control status indication.
MCLastEvent	MCLastEventEnum	-	Last mission control event.
AbortWeightReleased	Boolean	-	*true* when abort weight is released.
ADCPLog1.R1	float	metre	Range of beam 1 (front left) of the ADCP
ADCPLog1.R2	float	metre	Range of beam 2 (front right) of the ADCP
ADCPLog1.R3	float	metre	Range of beam 3 (rear left) of the ADCP
ADCPLog1.R4	float	metre	Range of beam 4 (rear right) of the ADCP
ADCPLog1.VelNorth	float	mm/sec	Measured velocity towards the North
ADCPLog1.VelEast	float	mm/sec	Measured velocity towards the East
ADCPLog1.Altitude	float	metre	Altitude from the sea floor (1000 if not in range)
ADCPLog1.VelMode	int	-	1 if sea floor in range
ADCPLog2.VwaterNorth	float	m/s	Velocity in water towards the North
ADCPLog2.VwaterEast	float	m/s	Velocity in water towards the East
ADCPLog2.MinAlt	float	metre	Minimum altitude the ADCP can measure
ADCPLog2.Pitch	float	radian	Vehicle pitch
ADCPLog2.Roll	float	radian	Vehicle roll
DepCtlNode.DepthDemand	float	metre	Depth demanded in metres
DepCtlNode.PitchDemand	float	radian	Pitch demanded in radians
DepCtlNode.SplanePosition	float	radian	Stern plane feedback status (within range of -20..20 deg)
DepCtlNode.SplaneDemand	float	radian	Stern plane demand
DepCtlNode.ADCPAltitude	float	metre	Shortest range of the four ADCP beams
GPSPosNode.Latitude	float	deg	Dead reconed latitude
GPSPosNode.Longitude	float	deg	Dead reconed longitude
GPSPosNode.GPSLatitude	float	deg	Last GPS latitude
GPSPosNode.GPSLongitude	float	deg	Last GPS longitude
GPSPosNode.TSLF	float	seconds	Time Since Last Fix
GPSPosNode.NumSat	int	-	Number of satellites
MotorNode.PropRPM	float	rpm	Propeller rotations per minute
MotorNode.PropTorque	float	N/m	Propeller torque calculated from current and RPM
MotorNode.PropPower	float	W	Prop power
MotorNode.EstSpeed	float	m/s	Estimated vehicle speed based on RPM
MotorNode.MotorCurrent	float	A	Motor current
MotorNode.MotorVoltage	float	V	Motor voltage
MotorNode.Fault	Boolean	-	*true* when fault is present
INS.Roll	float	radian	Vehicle roll. Should be stable around 5 deg.
INS.Pitch	float	radian	Vehicle pitch
INS.Heading	float	radian	Vehicle heading
INS.YawRate	float	radian/s	Yaw rate
INS.PitchRate	float	radian/s	Pitch rate
INS.RollRate	float	radian/s	Roll rate
INS.Latitude	float	deg	Dead reconed latitude
INS.Longitude	float	deg	Dead reconed longitude
INS.Depth	float	m	Depth as measured by the depth sensor
INS.Vz	float	m/s	Vertical velocity
INS.VNorth	float	m/s	Velocity towards the North.
INS.VEast	float	m/s	Velocity towards the East.
INS.Serial	long	-	Serial number of the INS reading.
PosCtlNode.RangeToGo	float	m	Range to the next waypoint.
PosCtlNode.RudderDemand	float	radian	Rudder angle demand
PosCtlNode.RudderPosition	float	radian	Reading of the rudder feedback potentiometre.
PosCtlNode.HeadingDemand	float	rad	Heading demand.
Power.RawVoltage	float	V	Voltage of the battery
Power.MotorCurrent	float	A	Current of the motor
Power.CriticalSysCurrent	float	A	Current of the critical systems
Power.NonCritCurrent	float	A	Current of the science sensors
Power.Temperature	float	degC	Temperature in the power pressure vessel.
Power.Leak	int	-	Uncalibrated number. Should not increase.
Power.BatteryFault	Boolean	-	*true* if any of the batteries has failed.

Table 2: The list of commands in Autosub 6000

Name	Description
Depth(demand)	Achieve and maintain demanded depth. Demand in metres in range -10 to 6000.
Altitude(demand)	Achieve and maintain demanded altitude. Demand in metres in range 0 to 500.
Profile(top,bottom)	Achieve profiling between top and bottom depths, both with ranges between -10 to 6000.
Surface()	Command results in the vehicle surfacing.
SPlaneAngle(dem)	Set the stern plane angle to demanded value. Valid range between -20 and +20 degrees
Heading(demand)	Causes heading mode and heading demand message to be sent to the rudder control node. Range +/-360.
PositionHold()	Sets the position control to store the current vehicle location as a waypoint and continuously head towards it.
PosP(d_1,d_2)	Causes position mode. The rudder will attempt to steer the vehicle in a straight line to position (d_1, d_2). d_1 is latitude in decimal minutes of a degree. Range +/-90(North) -/+90(South). d_2 is longitude in decimal minutes of a degree. Range +/-180(East) -/+180(West). If vehicle already at waypoint it will circle or draw figure "8" around the waypoint until next demand.
TrackP(d_1,d_2)	Causes track mode and position demand message to be sent to the rudder control node. The rudder node will attempt to steer the vehicle onto and then to maintain a straight line track between the last used position and the position supplied in the new demand values. d_1 and d_2 are as in PosP()
RudderAngle(d)	Sets the rudder control node into direct rudder mode. Range +/-20 degrees. Negative to port positive to starboard.
MotorOff()	Switches the motor off
MotorCurrent()	Causes the current mode and demand message to be sent to the motor control node. The sign sets the motor direction. Positive is forwards. Units are Amperes and range is +/-16.

ACKNOWLEDGEMENTS

This research has been funded by the UK's Natural Environment Research Council as part of Grant NE/F01256X/1, "Automated Diagnosis for Fault Detection, Identification and Recovery in Autosub6000".

REFERENCES

(Alessandri *et al.*, 1999) A. Alessandri, T. Hawkinson, A. J. Healey, and G. Veruggio. Robust model-based fault diagnosis for unmanned underwater vehicles using sliding mode-observers. In *11th International Symposium on Unmanned Untethered Submersible Technology, Autonomous Undersea Systems Institute*, pages 22–25, 1999.

(Antonelli, 2006) Gianluca Antonelli. 4. Fault detection/tolerance strategies for AUVs and ROVs. In *Underwater Robots - 2nd Edition*, volume 2 of *Springer Tracts in Advanced Robotics*, pages 79–91. Springer Berlin / Heidelberg, 2006.

(Ernits *et al.*, 2010a) Juhan Ernits, Richard Dearden, and Miles Pebody. Automatic fault detection and execution monitoring for AUV missions. In *Autonomous Underwater Vehicles, 2010. AUV 2010. IEEE/OES*, pages 1–9, Sept. 2010.

(Ernits *et al.*, 2010b) Juhan Ernits, Richard Dearden, Miles Pebody, and James Guggenheim. Diagnosis of Autosub 6000 using automatically generated software models. In *Proc. of the Twenty First International Workshop on Principles of Diagnosis (DX 2010)*, pages 1–8, Portland, OR, 2010.

(Griffiths and Brito, 2008) Gwyn Griffiths and Mario Brito. Predicting risk in missions under sea ice with autonomous underwater vehicles. In *Proceedings of the IEEE AUV2008 Workshop on Polar AUVs*, Woods Hole, 2008.

(McPhail and Pebody, 1998) Stephen D. McPhail and Miles Pebody. Navigation and control of an autonomous underwater vehicle using a distributed, networked control architecture. *The International Journal of the Society for Underwater Technology*, 23:19–30, 1998.

(McPhail *et al.*, 2010a) S. McPhail, M. Furlong, and M. Pebody. Low-altitude terrain following and collision avoidance in a flight-class autonomous underwater vehicle. *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, 224:279–292, November 2010.

(McPhail *et al.*, 2010b) S. McPhail, P. Stevenson, M. Pebody, M. Furlong, J. Perrett, and T. Lebas. Challenges of using an AUV to find and map hydrothermal vent sites in deep and rugged terrains. In *Autonomous Underwater Vehicles (AUV), 2010 IEEE/OES*, pages 1 –8, sept. 2010.

(McPhail, 2009) Stephen McPhail. Autosub6000: A deep diving long range AUV. *Journal of Bionic Engineering*, 6(1):55 – 62, 2009.