Mission planner for multiple AUVs: Verification procedures combining simulations and experiments

Stephanie Buadu
Dept of Marine Technology
NTNU
Trondheim, Norway
stephaniebuadu@gmail.com

Ingrid Schjølberg
Dept of Marine Technology
NTNU
Trondheim, Norway
ingrid.schjolberg@ntnu.no

Tore Mo-Bjørkelund
Dept of Marine Technology
NTNU
Trondheim, Norway
tore.mo-bjorkelund@ntnu.no

***Abstract*- This paper presents a mission control system (MCS) for a cooperative system of LAUVs. The proposed method is part of a system for operating several AUVs on a joint mission. The system is run on software and hardware from LSTS in Porto. The MCS can function as the only command and control software in the system or in integrated with Neptus, the command and control software offered in the LSTS-tool chain. The proposed mission planner has been tested through simulations and experiments conducted in the Trondheim Fjord on a cooperative system consisting of two AUV simulators and LAUV Fridtjof. The results are very promising, the MCS performed in accordance with its requirements, and the formation control proved able to control the formation during mission execution.**

Index terms - AUV, mission planning, formation control, cooperative system.

I. INTRODUCTION

Among the current applications for AUVs, both civil and military many require time, space, and functional distribution which is impossible to obtain with a common single vehicle approach. Multi-vehicle application introduces new challenges that must be addressed, such as formation control; this is the task of controlling multiple vehicles to complete mission objectives while keeping a formation. Formation control becomes a central part of the high-level mission plan which is broken down and specified to compute lower-level mission plans for the individual team members.

In 2014 a survey conducted by [1] stated that the literature on formation control of AUVs is lacking compared to those on mobile robots and aircraft. Also, most of the available research is pure theoretical without sufficient experiments and practices. The current situation within the field has not changed significantly. Although, there are some more examples of literature referring to experiments and practice in connection with formation control of AUVs the number fades in comparison to other fields such as ground robots and aerial vehicles. In other words, the applications are ready, and there is a need for well-tested and functioning approaches to formation control for AUVs, but the current research does not cover the need.

[2] defined three ways for mission planning given specific objectives. A fixed set of maneuvers can be generated:

- Directly by a human operator.
- By solving a control problem on a fixed discrete/hybrid graph containing the set of possible environment and vehicle actions.
- By using constraint problem-solving algorithms.

The optimal approach will depend on the mission objective, computational power available, system communication and environmental conditions. The presented work gives an example of mission plan generated by a human operator and directly implemented online. The uniqueness of the presented work is the implementation of the mission planner on a real system.

There are several challenges associated with mission planning for AUVs, and the number of issues grows when several vehicles are added to the mix. A common denominator for many of the challenges is unreliable system communication. On the surface, some AUVs use cellular and satellite communication, but this has high latency and limits the size of the data being transferred. Underwater, not all types of signals can be transmitted, vision and acoustics, limiting the communication abilities even further. Acoustic communication is utilized for passing messages underwater, but this is prone to interference, disruption and unpredictable delays [3]. As a consequence of the unreliable and restricted underwater communication, missions with several AUVs are prone to lacking reliable relative position measurements, spatial restrictions between vehicles to preserve communication links, and challenges in how to handle large volumes of data. Also, mission planners and cooperative algorithms, in these types of missions, must be robust for communication faults. Changing dynamics in mission environments pose a challenge if a vehicle's dynamics are altered as a result. [4] discusses how not considering this possibility in the initial planning phase may render certain tasks infeasible if a vehicle's dynamics are changed. In the case of unknown or uncertain environment dynamics, a solution could be to estimate the next state of the environment based on the current state and use this to conclude on the

impact on the vehicle's dynamics in-situ. However, this approach is not without its challenges [5].

In this paper, we propose a mission planner for multiple AUVs. It is a cooperative AUV system based on the LSTS tool-chain [6] that includes a newly developed application for mission planning, execution, and monitoring of multi-vehicle missions, the MCS (Mission Control System). Within the application, a proposed formation control method designed by adapting existing multi-vehicle mission theory from other fields is implemented. The system has been tested in simulation, and experiments, and in combination. The motivation behind this paper is the need for integrated AUV mission planners and procedures for testing and verification. The procedures are documented through implementation on a real system and experiments and field tests performed in the fjord of Trondheim (Norway).

II. SYSTEM DESCRIPTION

The vehicle applied in the presented mission and formation control is Fridtjof, an LAUV owned by NTNU, and shown in Fig.1.

Fig. 1. LAUV Fridtjof.

The LAUV is developed by the Underwater System and Technology Laboratory (LSTS) at Porto University in cooperation with OceanScan-MST. The LAUV was developed for security and surveillance, oceanography, and hydrography purposes, and is easily launched, operated, and recovered with minimal operational setup [3]. The vehicle is operated with software developed by LSTS; DUNE for on board operations and IMC messages for communication. The vehicle is operated with LAUV Remote close to shore, and a desktop command and control software like Neptus once it is a safe distance from shore. LAUV Fridtjof supports twelve motion primitives, but the system only requires one; the *go to* maneuver. The motion primitive maneuvers the vehicle to a waypoint defined by latitude, longitude, and depth or altitude, at a specified speed. The desired attitude at the destination can also be defined, but this property is not enabled by the system.

III. METHOD

The formation control method designed in this work has properties from existing formation control methods. AUV teams are categorized as either *master* or *slave*, and only one master is assigned per vehicle team. The method is described in three parts; formation, path generation and guidance, and maneuvering, and control algorithm. All equations in this section relating WGS 84 coordinates[1] to a displacement in meters are based on the assumption that the earth is spherical. The formation of the AUV team is inspired by virtual structures, and the vehicles are placed in a coordinate system where the master's position defines origin. The position of each slave i is defined as a displacement δx_i on the x-axis and δy_i on the y-axis, from the master's position. The structure does not rotate, and the attitude of the individual vehicle is not taken into account. Figure 2 visualizes the displacements and constraints for a team consisting of a master and slave. A constraint (3.1) is set onto the distance d between the master and slave to preserve the geometrical relationship between the vehicles. Some deviations are allowed before the slave is considered to violate the constraints and counteractive measures are taken.

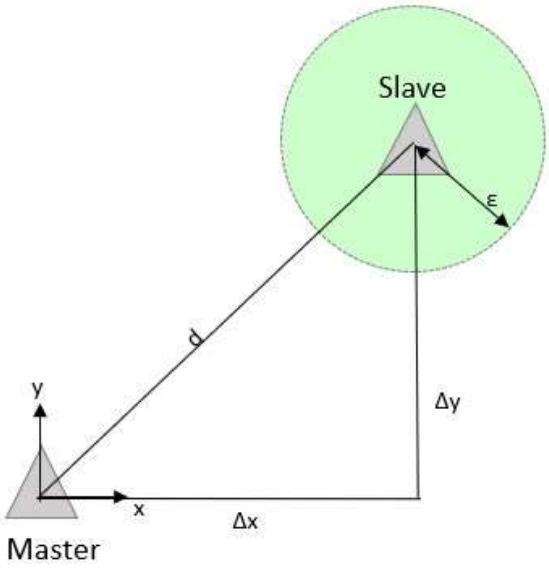

Fig. 2. Displacement, distance and constraints on a two-vehicle team in formation.

The vehicles' positions are defined in WGS 84 coordinates in the MCS and require conversion before the constraints can be checked. The Haversine formula, given in *Calculate distance (3.1), bearing and more between Latitude/Longitude points* (n.d.), is used to relate two latitude/longitude coordinates to a displacement in meters.

[1] The coordinate origin of WGS 84 is meant to be located at the Earth's center of mass and is used as a standard in navigation.

[redacted] (3.1)

[redacted] (3.2)

The master's path is generated by the operator selecting waypoints, and these being connected by straight lines. The waypoints are later displaced to generate paths with the same shape and length for the slaves. The equation for displacing the waypoints is given in *Calculate distance, bearing and more between Latitude/Longitude points* (n.d.) and stated in (3.2)(3.3).

$$\varphi_S = \text{asin}(\sin\varphi_M \cdot \cos\delta + \cos\varphi_M \cdot \sin\delta \cdot \cos\theta),$$

$$\lambda_S = \lambda_M + \text{atan2}(\sin\theta \cdot \sin\delta \cdot \cos\varphi_M, \cos\delta - \sin\varphi_M \cdot \sin\varphi_S)$$

[redacted] (3.3)

where φ is latitude, λ is longitude, Δx and Δy are displacements in meters, and R is the earth's radius. Before the MCS accepts the paths, they are checked for potential collisions. The paths are parameterized, and the distance between points on the parameterized lines are checked against a threshold set by the operator. The piecewise parametrization of the path is given in (3.4), where A, B are two waypoints. The distance d between points on the path is given by (3.3) and checked to satisfy $d > d_{min}$. d_{min} is set by the operator and is the minimum distance allowed between two vehicles in the team.

[redacted] (3.4)

where φ is latitude, λ is longitude, t is the parameterization variable.

The guidance in the formation control method stands out from other leader-follower approaches because there is no explicit synchronization of path variables or direct feedback of the master's position to the slaves during guidance. Instead, all vehicles start execution of their predetermined path at the same time, with the same desired speed set onto the speed controller, and no external actions are taken unless team constraints are violated. A prerequisite for this strategy to work is that the vehicles are in formation before mission execution begins, and this is achieved by commanding the vehicles to the first waypoint before mission execution. To ensure that the vehicles have the right heading before starting the mission their path includes a waypoint X before reaching the first waypoint. X is generated based on the two first waypoints on the path A and B. Starting in A, X extends the line between A and B by 20 meters.

The formation control is placed in the MCS, resulting in a centralized control architecture. A control algorithm that checks the team constraints, and takes action if any are violated, is looped during mission execution. The algorithm falls into the category of behavior-based control, and three behaviors are available for the vehicles; *path following*, *collision avoidance*, and *formation preservation*. However, the vehicle's resulting behavior is not a weighted combination, as the regular practice, but rather a single behavior. Path following is the default behavior, and the cooperative strategies activate the two remaining behaviors.

Two team constraints (TC1 and TC2) have been defined in the formation control method, and they are presented in Table 3.1. TC2 is in place to ensure that the vehicles do not collide, and violation of this constraint should result in one or more vehicle switching behaviors from path following or formation preservation to collision avoidance. Cooperative strategies for handling the violations have not been implemented and is left as further work. To preserve the formation, TC1 is set onto each vehicle, and five cooperative strategies, presented in Table 3.2, are designed to handle violations of this constraint. The cooperative strategies define which vehicle should switch behaviors if different situations. Cooperative strategies CS1 and CS4 state that the master waits, i.e., take on formation preservation behavior, once for each slave if it falls behind, and once if the master catches up to it. A consequence of this is that a master will stop considering a slave as a team member if it has previously taken action not to violate the constraint. The master switches back to path following behavior once the constraint is satisfied or when the waiting period, currently set to 30 seconds, is over, whichever comes first. CS2 and CS3 state that a slave must always wait if the master falls behind or the slave catches up to the master. There is no waiting period, and therefore the slave must wait until the constraint is satisfied before switching to path following behavior. To avoid producing of additional constraint violations, C5 ensures that all slaves satisfying TS1 wait while the master waits.

Table 3.1: Team constraints in the formation control method.

ID	Description
TC1	A slave must always be within a radius of from it's desired position.
TC2	The distance between two vehicles must always be greater than d_{min}.

Table 3.2: Cooperative strategies for handling TC1 violations.

ID	Situation	Action
CS1	A slave falls behind during mission execution.	Master must wait once.
CS2	Master falls behind during mission execution.	A slave must always wait.
CS3	A slave catches up to master during mission execution.	A slave must always wait.
CS4	Master catches up to a slave during mission execution.	Master must wait once.
CS5	Master waits for a slave.	All remaining slaves must always wait.

The control algorithm is looped five times per second for each slave and has conditions in place to catch and identify defined constraint violations.

The requirements for the MCS are specified in Table 3.3, and the priority ranging from ’High’ to ’Low’ was determined by the author.

Table 3.3 Requirements for MCS

Requirement ID	Requirement	Priority
R1	Communication between the system and the LAUV should be enabled by DUNE and IMC messages.	High
R2	All system functionality should be available offline.	Medium
R3	A user should be informed if the communication link to a vehicle is lost.	High
R4	The system should automatically reconnect to a vehicle if the communication link is lost.	High
R5	Functionality that requires communication with a vehicle should not be displayed if the communication link is not established.	Medium
R6	The system should automatically generate paths for the team members based on a path generated by an operator.	High
R7	It should be possible to send a mission plan to the vehicle and start the execution at a later point in time.	Medium
R8	A user should be able to review the mission plan at all times.	Medium
R9	The system should enable the user to monitor an ongoing mission.	High
R10	A user should be able to abort an ongoing mission.	Medium
R11	A user should be to able to replan the most recent mission.	Medium
R12	It should be possible to generate a path by marking the waypoints.	High
R13	The system should resend a message until a callback is received.	Medium

In general, a significant amount of work is needed to design and implement the MCS. To simplify interfacing DUNE with the MCS the backend components of the application were written in C++. Keeping the MCS as a web application was considered, but linking the backend code in JavaScript proved challenging. Several programming languages were unsuccessfully tested for the frontend development before C++ proved successful. Qt Creator (discussed in Section 4.2.4) was used as the Integrated Development Environment (IDE) for building the application.

The development has mainly taken place in two files; gui.cpp and mcslib.cpp. The user interface was designed using drag-and-drop functionality in Qt Creator, and a .ui file was auto-generated. The code for handling the TCP-connections in the application is developed by OceanScanMST and open source, and the code for handling incoming messages is developed by NTNU.

IV. RESULTS AND VERIFICATION

Fig. 2 gives an overview of the proposed system including hardware and software components. Starting at the top left, the Light Autonomous Underwater Vehicle (LAUV) ‘Fridtjof’ is the physical vehicle used during field tests. The Manta Communications Gateway enables communication between LAUV Fridtjof and the remaining software components by routing WiFi-signals. Neptus is the mission command and control software offered in the LSTS toolchain, and in this system, it is utilized first and foremost for commanding LAUV Fridtjof between missions and as a backup control application. There is no explicit functionality for cooperative systems in Neptus, hence the need for the additional functionality that the MCS introduces next to the toolchains offerings. The MCS may be run both standalone and in combination with Neptus. During simulations for instance, Neptus can be left out of the system architecture, as there is no risk of damaging the vehicle, and the extra security and single vehicle control functionality that Neptus offers is not required.

The LAUV simulator used for testing run on the same computer as the MCS, and all communication throughout the system was enabled by Inter-Module Communication (IMC) messages, which are also included in the LSTS toolchain.

Figures 3-5 show the developed MCS’ user interface in which an operator may generate a high-level mission plan which is decomposed and sent to the individual vehicles. The MCS handles formation control and allows the operator to monitor and intervene during mission execution.

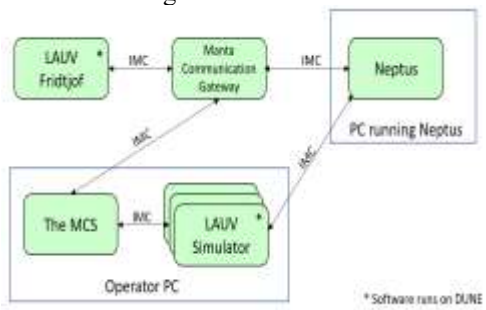

Fig. 2. System overview MCS.

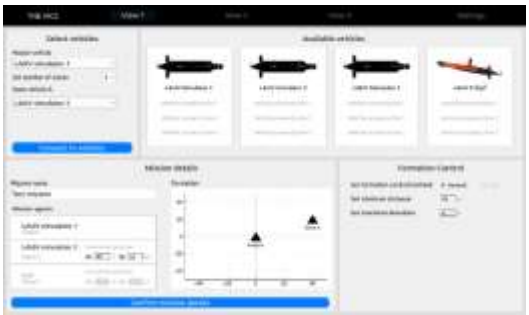

Fig. 3. View 1 of the MCS. Interface for defining the vehicle team, formation, and constraints.

Fig. 4: View 2 of the MCS. Enables defining and reviewing the path for the team and loading the mission plan onto the individual vehicles.

Fig. 5: View 3 of the MCS. The view for starting mission execution and monitoring mission execution.

The designed formation control method includes elements from virtual structures, leader-follower, and behavioral methods. Each vehicle implements three behaviors; path following, obstacle avoidance, and formation preservation. Path following is the default behavior. Five cooperative strategies were implemented to handle constraint violations and set the right behavior onto each vehicle.

The vehicles switching behavior based on what happens in real time makes it difficult to prove the validity of the formation control method analytically. Therefore, simulations and field tests have been conducted to test the method. The logical reasoning behind why the implemented cooperative strategies should yield successful formation control is presented in this section. First, an argument is made for the validity of the formation control method in the case where the vehicle team is exposed to equal current, wind, and wave forces. It has been established that the path generated for each vehicle has the same length and shape and that the desired speed is identical. Further assumptions are:

- The vehicles are homogeneous with similar vehicle dynamics.
- The motion controllers on the vehicles are equal.
- The vehicles are in formation and have the same heading before mission execution.
- The generated mission is feasible.

The conclusion to be drawn from this is; if the vehicle's start mission execution at the same point in time and respond equally to the forces they are exposed to, their mission progress will be identical, and the vehicles will complete the mission while staying in formation.

The second case to consider is the case of the vehicle team being exposed to non-uniform external forces causing a violation of TC1, and the following assumptions are made in addition to the assumptions stated in the previous case:

- The mission progress sent by the vehicle is continuously updated and gives an accurate representation of the vehicle's mission progress.

The master's wait period is sufficient for a slave to get back into the formation in the

Initial testing of the MCS and the formation control method, in particular, was performed through simulations with the LAUV simulators. The simulations were conducted first and foremost to validate the cooperative strategies used in the formation control method in a controlled environment before proceeding with field testing. Two of the tests are shown in the following. In the case that the desired constraint violation did not occur naturally, vehicles were manually stopped and started, i.e., mission execution was aborted or resumed. All missions were conducted on the surface, without restrictions on the minimum distance between the vehicles. A number of case studies were performed.

In Case 1, Slave B falls behind mission was designed to demonstrate CS1 and CS5. The mission entailed a three-vehicle team maneuvering north in parallel vertical lines. Maximum allowed deviation was set to 20%, which translates to ± 5.7m for both slaves. The vehicles' paths and trajectories are depicted in Figure 6, and the vehicles' positions and headings at times of interest are represented by rectangular markers. $t = 0$ marks the beginning of mission execution.

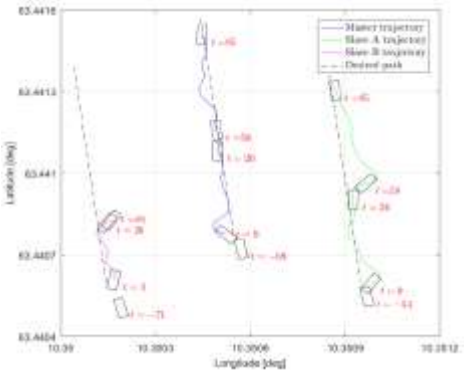

Fig. 6. Case 1. Slave falls behind position.

In Case 2, the master catches up to slave mission was designed to demonstrate CS4, however, during mission execution, CS3 was also activated. The mission objective was for two vehicles to maneuver north in parallel vertical lines. Maximum allowed deviation was set to 15%, which translates to ± 4.2m. None of the constraint violations that occurred during mission execution were induced. Figure 8 illustrates the LAUV in operation during Case 2 in the fjord.

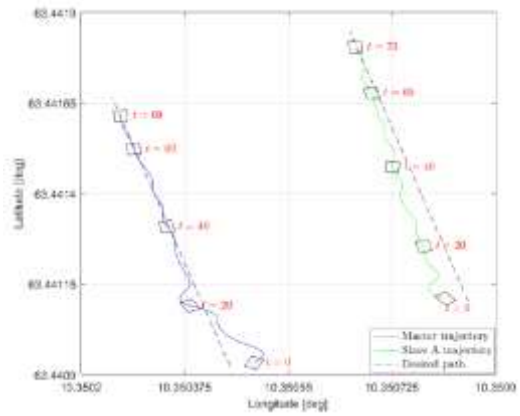

Fig. 7. Case 2. Master catches up on slave mission.


Fig.8. Case 2 Mission.

V. CONCLUSIONS

This paper has presented a MCS application for controlling a cooperative system of LAUVs with mission planning and formation control capabilities. The application is a part of a system for operating cooperative AUV systems that consists of software and hardware from the LSTS. The MCS can function as the only command and control software in the system or in cooperation with Neptus, the command and control software offered in the LSTS-toolchain. The proposed mission planner has been tested through simulations and experiments were conducted in the Trondheim Fjord with a cooperative system consisting of two simulators and LAUV Fridtjof. The results are very promising, the MCS performed in accordance with its requirements, and the formation control proved able to control the formation during mission execution. Verification and testing of the MCS, show that the mission planning capabilities of the application satisfy the application requirements and needs of a cooperative system. Although, the field tests revealed that the path generation should be made more complex to obtain paths that are better suited for physical vehicles. The formation control capabilities in the MCS are enabled by a formation control method with properties from leader-follower systems, virtual structures, and behavior-based formation control. The combination of properties from different types of existing formation control proved successful for cooperative LAUVs, and results from simulations and field tests show that the vehicle team's formation was successfully maintained during mission execution. The experimental verification method allowed for testing before the for the formation method was completed, and collision avoidance should be designed and implemented before experiments with multiple physical vehicles are to be conducted.

Further work will be to further develop the application to include more advanced mission capabilities.

REFERENCES

[1] Li, X., Zhu, D., & Qian, Y. (2014, 10). A Survey on Formation Control Algorithms for Multi-AUV System. *Unmanned Systems*.

[2] Pinto, J., Dias, P. S., Martins, R., Fortuna, J., Marques, E., & Sousa, J. (2013). The LSTS toolchain for networked vehicle systems. *2013 MTS/IEEE OCEANS - Bergen*, (pp. 1-9). Bergen.

[3] Madureira, L., Sousa, A., Braga, J., Calado, P., Dias, P., Martins, R., Pinto, J. and Sousa, J. (2013), The light autonomous underwater vehicle: Evolutions and networking, *OCEANS Bergen, 2013 MTS/IEEE* . pp. 1 – 6. doi: 10.1109/OCEANS-Bergen.2013.6608189.

[4] McMahon, J. and Plaku, E. (2016), Mission and motion planning for autonomous underwater vehicles operating in spatially and temporally complex environments, *IEEE Journal of Oceanic Engineering* 41(4). pp. 893–912. doi: 10.1109/JOE.2015.2503498.

[5] MahmoudZadeh, S., Powers, D. M., Sammut, K. and Yazdani, A. (2016), Toward efficient task assignment and motion planning for large-scale underwater missions, *International Journal of Advanced Robotic Systems* 13(5). pp. 1–13. doi: 10.1177/1729881416657974.

[6] Pinto, J., Calado, P., Braga, J., Dias, P., Martins, R., Marques, E. and Sousa, J. (2012), Implementation of a Control Architecture for Networked Vehicle Systems, *IFAC Proceedings Volumes* 45(5). 3rd IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles, pp. 100 – 105. doi: 10.3182/20120410-3-PT-4028.00018.