

14. **CalculiX** CrunchiX USER'S MANUAL version 2.5 // Convergent Mechanical. 2003–2015, available at: <http://bconverged.com/calculix/doc/ccx/html/ccx.html>
15. **OpenFOAM**. Free Open Source CFD. The OpenFOAM Foundation// OpenFOAM Foundation, available at: <http://www.openfoam.org/>
16. **COMSOL** Multiphysics. The Platform for Physics-Based Modeling and Simulation: available at: <https://www.comsol.com/comsol-multiphysics>.
17. **Octave** and Matlab FEM Toolbox, available at: <http://www.featool.com>
18. **Barulina M. A., Pankratov V. M.** *Modelirovanie dinamicheskikh protsessov v mikromekhanicheskikh datchikakh inertial'noi informatsii i ikh komponentakh s pomoshch'yu spetsializirovannogo programmnogo obespecheniya* (Modeling of dynamic processes in micromechanical inertial sensors and their components using specialized software), *Vestnik SSAU*, 2015, vol. 14, no. 2, pp. 223–233 (in Russian).
19. **Eliseev V. V.** *Mekhanika uprugikh tel* (The mechanics of elastic bodies), St. Petersburg, Publishing house of SPbGTU, 1999, 341 p. (in Russian).
20. **Grigolyuk E. I., Selezov I. T.** *Neklassicheskie teorii kolebaniy sterzhney, plastin i obolochek* (Non-classical theory of vibrations of rods, plates and shells), *Itogi nauki i tekhniki. Ser.: Mekh. tverd, deform. tel*, Moscow, VINITI, 1973, vol. 5, 272 p. (in Russian).
21. **Dzhashitov V. E., Pankratov V. M., Barulina M. A.** *Teoreticheskie osnovy razrabotki i sozdaniya superminiaturnogo mikromekhanicheskogo mnogofunktsional'nogo datchika inertial'noy informatsii* (The Theoretical Bases of Development and Creation of the Superminiature Micromechanica/ Multifunction Sensor of the Inertia/ Information), *Nano- i Mikrosistemnaya Tekhnika*, 2010, no. 5 (118), pp. 46–54 (in Russian).
22. **Barulina M. A.** *Matematicheskoe obespechenie konechno-elementnogo modelirovaniya mikromekhanicheskikh datchikov inertial'noi informatsii v ramkakh neklassicheskoi teorii izgiba* (Finite-Element Modeling of the Micromechanical Inertial Sensors Using Non-Classical Beam Theory), *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2015, vol. 16, no. 11, pp. 764–770 (in Russian).
23. **Barulina M. A.** *Postroenie matritsy mass trekhmernogo konechnogo elementa dlya modelirovaniya dinamiki mikromekhanicheskikh datchikov inertial'noy informatsii i ikh uzlov* (Development of a Mass Matrix of the 3D Finite Element for Modeling of the Dynamics of Micromechanical Inertial Sensor Data and their Components), *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2015, vol. 16, no. 5, pp. 352–360 (in Russian).
24. **Filippov A. P.** *Kolebaniya uprugikh sistem* (Vibrations of elastic systems), Publisher Ukrainian Academy of Sciences, Kyiv, 1956 (in Russian).

УДК 004.942 + 004.5

DOI: 10.17587/mau.18.202-207

Б. Д. Аминеv, мл. науч. сотр., daianovich@mail.ru,
С. К. Данилова, канд. техн. наук, вед. науч. сотр., lab45_1@ipu.ru,
 Институт проблем управления им. В. А. Трапезникова РАН

Автоматизация управления движением морского подводного объекта по заданному маршруту на основе имитационного моделирования*

Описывается автоматизация управления движением морского подводного объекта по заданному маршруту в пространстве средствами разрабатываемого программного обеспечения с применением методов полномасштабного имитационного моделирования и теории управления сложными динамическими объектами. Рассмотрены проблемы, возникшие во время разработки, и пути их решения, приведен анализ перспектив дальнейшего развития разработанной системы. Приведены результаты работы приложения в рамках реального проекта при решении задачи реализации маршрута за допустимый интервал времени движения.

Ключевые слова: автоматизация, управление, полномасштабное имитационное моделирование, морской подводный объект, траекторное движение, малозумное маневрирование

Введение

Модель динамики морского подводного объекта (МПО) представляет собой сложную систему нелинейных дифференциальных уравнений высокого порядка. Управление МПО является по своей структуре сетевым (осуществляется с различных постов управления), многомерным, многоканальным, проводится с ограничениями на фазовые координаты движения с использованием различных средств управления, которые имеют как гидродинамическую, так и гидростатическую природу. К первой группе относятся гидродинамические рули, позволяющие МПО маневрировать в пространстве на ходу. Эффективность управления горизонтальными и вертикальными рулями зависит от скорости дви-

жения МПО, углов перекладки и их допустимых значений. Гидростатические средства управления (цистерны) применяются на малых скоростях и в режиме "без хода" [1]. В исследуемом режиме управления цистерны используются для создания балласта, компенсирующего силы обжатия корпуса МПО и учета гидрологических разрезов. Разработка работоспособных алгоритмов управления движением МПО в различных режимах управления должна проводиться в ходе исследования его маневров с ограничениями на интенсивность управления, состав средств управления, скорость движения с применением полномасштабной имитационной модели объекта [2]. Управление должно основываться на физической и технической интерпретации работы технических средств управления, модели движения объекта и обеспечивать функциональность для выполнения режимов движения.

* Работа выполнена при поддержке РФФИ, грант № 15-08-05133а.

Особенности задачи управления движением МПО

Реализуемый маршрут задается набором пространственных координат на карте, соединенных прямыми. Во время реализации маршрута необходимо обеспечить такой режим движения, чтобы МПО создавал как можно меньше шумов (например, акустических и гидрофизических). Для решения этой задачи применяется малозумный режим маневрирования, предполагающий понижение кавитационных и других шумов за счет введения ограничений на фазовые координаты и скорость движения объекта [3]. Визуальный образ задачи малозумного маневрирования приведен на рис. 1 (см. третью сторону обложки).

Справа видна зеленая область допустимого маневрирования, слева — визуализация МПО. Снизу расположены органы управления, позволяющие задавать параметры маневра, настройки алгоритмов управления, а также ограничения на средства управления МПО.

Для реализации малозумного маневрирования по точкам маршрута на карте необходимо выполнить следующий набор действий:

1) извлечь из внешней базы данных координаты точек, составляющих маршрут (точки заданы в Земной системе координат);

2) преобразовать полученные координаты в связанную систему координат;

3) на основании координат точек маршрута, заданных в связанной системе координат, сгенерировать задания для системы полномасштабного имитационного моделирования и сохранить их во внутренней базе данных;

4) по полученным из внутренней базы данных заданиям на маневрирование вдоль заданного маршрута провести полномасштабное имитационное моделирование в связанной системе координат;

5) результаты полномасштабного имитационного моделирования сохранить во внутренней базе данных, перевести в Земную систему координат и послать во внешнюю базу данных;

6) во время демонстрации синхронно с отображаемым движением МПО по карте (на основе результатов моделирования из внешней базы данных) проводить визуализацию результатов моделирования в связанной системе координат по сохраненным данным из внутренней базы данных.

Для реализации маневрирования МПО по отмеченному маршруту на карте было разработано специализированное программное обеспечение (ПО) *Wanderer*.

Архитектура ПО *Wanderer*

ПО *Wanderer* написано на C++ 11 с использованием библиотек *Boost* [4] (для реализации некоторого функционала отдельных модулей), *Qt* 5 [5], *Qwt* 6 [6] (для реализации графического интерфейса пользователя), *Ogre 3D* [7] (для отображения МПО).

Следует отметить также, что *Wanderer* опирается на разрабатываемую систему полномасштабного имитационного моделирования МПО и способствует ее развитию как в функциональном, так и в качественном плане.

ПО *Wanderer* можно разделить на следующие модули:

- **io** — инкапсулирует работу с внешней базой данных (ВБД). В этом модуле собраны все запросы к ВБД и преобразование полученных данных для работы в инфраструктуре *Wanderer*;
- **program_options** — содержит функционал работы с настройками приложения — запросы на получение значений тех или иных настроек из файла настроек или параметров вызова приложения;
- **coordinate_transformer** — модуль преобразования координат из связанной системы координат в географическую;
- **trajectory_validator** — реализует графический интерфейс пользователя и занимается диспетчеризацией работы двух подмодулей;
- **trajectory_simulator** — проводит моделирование траектории — реализует заданный реперными точками маршрут с использованием полномасштабной имитационной модели движения МПО, работы технических средств управления и алгоритма малозумного маневрирования;
- **trajectory_loader** — проводит фоновую загрузку набора координат из заданной промоделированной с помощью *trajectory_simulator* траектории из локальной базы данных (ЛБД) и отображение фазовых координат МПО на участке и в заданный момент времени.

Управление движением МПО на основе компьютерного имитатора *Wanderer*

Вычислительным ядром для ПО *Wanderer* служит разрабатываемая коллективом лаборатории ИПУ им. В. А. Трапезникова РАН система полномасштабного имитационного моделирования (СИМ) движения МПО, структура которой приведена на рис. 2.

Непосредственно перед началом работы над *Wanderer* СИМ претерпела сильные изменения — в первоначальной версии результаты проведения экспериментов сохранялись в файлах и группировались в виде папок. Но затем этот подход начал представляться не расширяемым и не достаточно гибким. В связи с этим было решено перейти на хранение результатов экспериментов в локальной базе данных (ЛБД). Сейчас для ЛБД используется *SQLite 3* [8], что достаточно для работы с не очень большой и не сильно нагруженной базой данных, которая не требует сложной настройки и запуска сервера системы управления базой данных — вся база данных хранится в одном файле.

Структура спроектированной ЛБД позволяет сохранять результаты экспериментов и собирать их в группы. Каждый эксперимент может включать несколько объектов различных типов (сейчас рас-



Рис. 2. Структура разрабатываемой СИМ (ТСУ — технические средства управления; МШМ — малозумное маневрирование; ФИП — функции информационной поддержки; СИП — система информационной поддержки)

считается только один класс МПО, но структура открыта для расширений новыми видами объектов). В ЛБД можно сохранить коэффициенты алгоритмов управления для различных наборов заданий, ограничений, параметров внешней среды и оценок качества переходного процесса.

Спроектированная база данных еще требует доработок. Многие недочеты были обнаружены и исправлены в ходе реализации *Wanderer*. В качестве очевидного узкого места можно указать желание проектировщика сделать набор обобщенных таблиц для хранения элементов фазовых координат МПО, что требует при записи или чтении состояния объекта обращения к нескольким таблицам. Более перспективным представляется оставление основного набора таблиц, реализующих логику экспериментов, сохранение коэффициентов алгоритмов управления и создание под каждый новый вид экспериментов своей таблицы, содержащей все необходимые в рамках эксперимента фазовые координаты.

Представленная структура базы данных позволяет проводить моделирование нескольких МПО в рамках одного эксперимента, группировку экспериментов, сохранение результатов проведения экспериментов (оценок качества переходного процесса), хранение информации об окружающей среде и пр.

СИМ, которая сама по себе является большой системой, при переходе на работу с ЛБД также расширила свои возможности (в том числе получила возможность моделирования нескольких объектов

в рамках одного эксперимента вдобавок к уже имевшейся возможности проведения нескольких экспериментов одновременно).

Модуль *io*

В этом модуле реализуется получение и запись данных ВБД. Особенностью рассматриваемой задачи является то, что маршрут движения МПО строится в географической системе координат, а управление МПО по заданному маршруту — в связанной, поэтому при получении данных — набора точек в географической системе координат — сначала происходит их перевод в связанную систему координат (модуль *coordinate transformer*). Также проводится фильтрация точек маршрута — этот этап обработки необходим, чтобы преобразовать программно сгенерированные точки маршрута в удовлетворяющие здравому смыслу без потери информации. Например, рассмотрим маршрут, вид которого представлен на рис. 3.

Сгенерированная траектория может содержать точки, находящиеся друг под другом и означающие, что на данном этапе МПО должна погружаться или всплывать (рис. 3). Все эти точки объединяются в единое задание на переход по глубине с помощью кормовых или носовых рулей с заданной скоростью хода. При этом надо начинать всплытие после первой такой точки (самой нижней в серии, см. всплытие на рис. 3), а погружение — до первой, самой верхней в серии (случай раннего погружения представлен на рис. 3), иначе погружение будет проводиться с запозданием и не будут удовлетворены требования к реализации маршрута (случай позднего погружения см. на рис. 3). Для расчета того, насколько раньше надо проводить погружение с учетом ограничений на фазовые координаты движения МПО, потребуется выполнить еще немало исследований и экспериментов в СИМ.

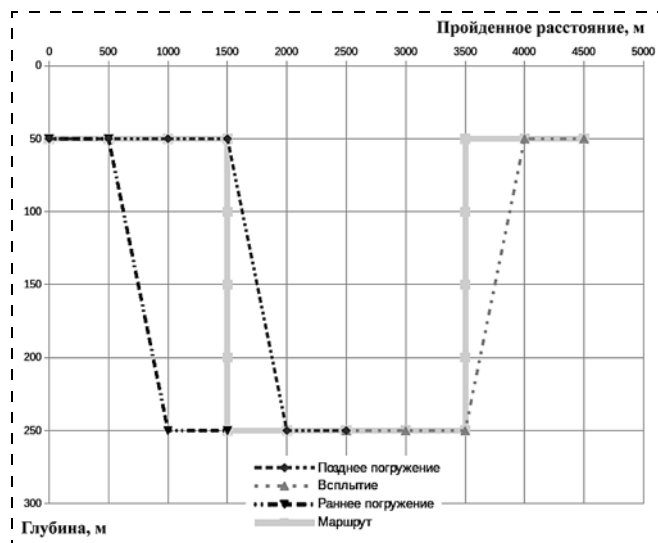


Рис. 3. Пример маршрута и маневров всплытия/погружения

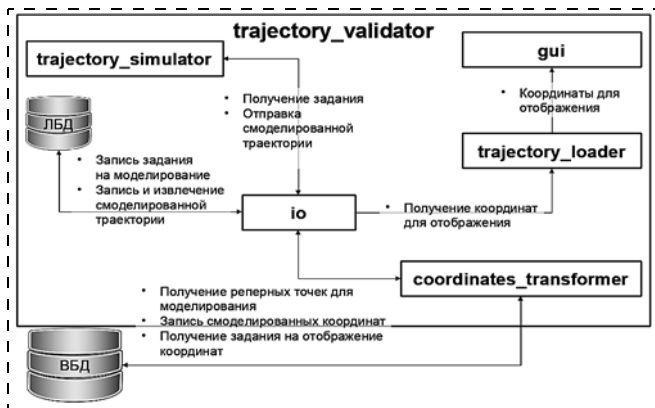


Рис. 4. Структура приложения

На рис. 4 приведена структура приложения *Wanderer*, на которой схематично показано взаимодействие модулей.

При сохранении результатов требуется постобработка: возникает необходимость проанализировать последовательность фазовых координат, чтобы разметить места проведения маневров, это может помочь стороннему программному обеспечению для отображения движения МПО в ускоренном масштабе времени. Чтобы не выполнять слишком медленную демонстрацию многочасового перехода по точкам маршрута, необходимо ускорить демонстрацию, но для того чтобы не пропустить места, заслуживающие более пристального внимания (переходные процессы), демонстрацию можно приостанавливать на соответствующих участках траектории, размеченных заранее оговоренным образом.

Модуль *trajectory_validator*

В этом модуле реализован весь графический интерфейс пользователя и осуществляется запуск двух независимых потоков, которые работают в фоне и выполняют различные задачи, позволяя решать две различные задачи: моделирование и отображение результатов моделирования.

Подмодуль *trajectory_simulator*

Данный модуль работает в отдельном потоке в ожидании сигнала к проведению моделирования. При получении такого сигнала с помощью модуля *io* начинается моделирование.

Сначала получается траектория (модуль *io*), которая затем преобразуется в набор заданий (модуль *io*). Задания представляют пары начального и конечного состояний МПО — переходных процессов в рамках маневров по маршруту. Они объединяют задание на переход по глубине, курсу и изменение скорости.

При этом до работы над *Wanderer* СИМ не обладала возможностью движения в направлении заданной точки, имелась лишь возможность перехода на заданный курс и стояла задача попадания в заданную точку по окончании маневра. Для этого

пришлось дать СИМ возможность задавать курс парой координат в связанной системе координат и пересчитывать это задание каждый такт времени — реализовать следующее управление.

В рамках проекта *Wanderer* было необходимо не только провести полномасштабное имитационное моделирование маневрирования МПО, но и сделать это движение по возможности скрытым, для чего было решено использовать два подхода. Во-первых, режим малозумного маневрирования, который представляет собой алгоритм управления энергетической установкой в зависимости от фазовых координат МПО (заглубление винта, курс, дифферент и т. д.). Этот алгоритм задает особый режим работы энергетической установки, который должен обеспечить низкую шумность объекта. В качестве недостатков этого алгоритма стоит отметить тот факт, что даже "зная" о задании на скоростной режим, он нещадно урезает скорость движения МПО до небольших и нешумных, что приводит к запаздыванию при длительном маневре — МПО требуется больше времени на реализацию маршрута. Второй использованный подход — ограничение фазовых координат движения МПО. Были установлены достаточные для обеспечения нормальной управляемости ограничения на углы отклонения горизонтальных кормовых и вертикальных рулей и значение дифферента. Подобные ограничения призваны еще больше уменьшить шумность объекта.

В ходе проведения экспериментов пришлось установить нижнюю границу задания на скорость для объекта, чтобы не попасть на инверсионные скорости. Также на малых скоростях МПО начинает издавать больше шумов.

С учетом всех этих факторов к полученным из модуля *io* заданиям на маневры добавляются необходимые алгоритмы управления (если происходит переход по глубине, добавляется соответствующий алгоритм, если такового не происходит, алгоритм не включается), ограничения на фазовые координаты движения МПО. После этого модуль сохраняет полученные дополненные задания в ЛБД и запускает СИМ.

СИМ осуществляет моделирование и оповещение остальных модулей об этапе проведения эксперимента. С целью минимизировать воздействие алгоритма малозумного маневрирования на время переходного процесса и до реализации алгоритма управления скоростью МПО в зависимости от времени прибытия в точку назначения было решено "включать" алгоритм малозумного маневрирования только на этапах совершения маневров (по глубине, курсу), а все оставшееся время, когда МПО движется сравнительно прямолинейно в направлении реперной точки, включать особый режим работы энергетической установки и просто набирать необходимую для достижения цели скорость. Это позволяет уменьшить потери во времени до приемлемых значений.

Стоит отметить, что СИМ позволяет проводить моделирование переходного процесса с траектори-

ей длиной порядка сотен километров и проходить в окрестности реперных точек с точностью порядка нескольких десятков метров, что говорит о качестве разработанных алгоритмов управления и программного обеспечения СИМ, прошедшего, таким образом, некоторое боевое крещение при решении данной прикладной задачи.

Также стоит отметить небольшой недостаток рассматриваемого подхода, который состоит в том, что задания формируются один раз и навсегда и не подразумевают дополнения промежуточными заданиями. Есть мнение, что в ЛБД стоит сохранять "целевое задание", а внутри СИМ дать возможность полученное задание дополнять маневрами на усмотрение командира МПО. Подобного рода изменение в СИМ не предполагает сильных переработок, но повышает гибкость решения. Также, возможно, следует добавить генерацию промежуточных состояний в *trajectory_simulator*, поскольку во время маневров по курсу с учетом ограничений на фазовые координаты МПО заметно отплывает от заданной прямой, соединяющей две реперные точки, и движется рядом с ней по направлению к следующей реперной точке, не пытаясь вернуться. При генерации маршрута следует указывать коридор, в котором МПО может безопасно маневрировать и скорректировать алгоритмы управления с учетом условия невыхода из заданного коридора. Для подобной корректировки необходимо будет рассмотреть различные маневры по курсу и начать маневр до подхода к реперной точке начала поворота.

По завершении моделирования *trajectory_simulator* вновь обращается к *io* для сохранения состояний МПО после моделирования. На данный момент сохранение проводится в ЛБД и ВБД по завершении каждого конкретного задания, т. е. по окончании перехода от одной реперной точки к другой. Частоту сохранения можно будет в будущем увеличить (для длительных переходов). После окончания моделирования *trajectory_simulator* ждет следующих указаний на моделирование, работая в фоновом режиме.

Подмодуль *trajectory_loader*

Для отображения смоделированной траектории в *Wanderer* используется модуль *trajectory_loader*. Он, как и *trajectory_simulator*, работает в фоновом режиме и ожидает сигнала к действию — номер траектории и момент времени для отображения. Получив данные, *trajectory_loader* отображает МПО в заданный момент времени, а также графики курса и глубины в часовой окрестности от заданной точки. Это необходимо для демонстрации переходных процессов.

В ходе реализации этого модуля были приняты важные проектные решения относительно ЛБД для ускорения работы, так как изначального быстродействия при текущей реализации ЛБД было недостаточно (сигналы на отображение МПО прихо-

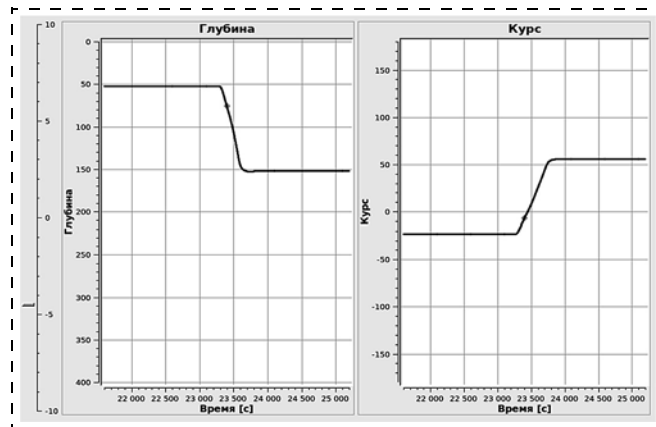


Рис. 5. Визуализация перехода по глубине и курсу

дят каждую секунду, а выполнять поиск состояний МПО в ЛБД в окрестности заданного времени получалось небыстро). За счет некоторых компромиссов (отхода от обобщенности) и введения индексов для полей таблиц удалось добиться приемлемой скорости работы с БД.

На рис. 5 представлена визуализация переходного процесса по глубине и курсу с использованием подмодуля *trajectory_loader*.

Заключение

Работа над задачей реализации движения МПО вдоль заданной траектории привела к созданию системы *Wanderer* как надстройки над СИМ. Эта работа дала толчок к развитию и обогащению возможностей разрабатываемой СИМ и позволила опробовать многие подходы в рамках поставленной задачи, решить интересные проблемы движения МПО по заданному маршруту. Созданная система демонстрирует хорошие результаты и применяется для осуществления маневрирования по заданной траектории на карте как составная часть большого проекта.

Задача, подобная данной, уже решалась ранее В. Г. Борисовым и С. К. Даниловой, но имела свою специфику, а именно: маневрирование проводилось не в маломощном режиме, не накладывались ограничения на технические средства управления (углы отклонения рулей) и фазовые координаты (дифферент). Также отличительной особенностью *Wanderer* можно назвать развитую систему взаимодействия с несколькими базами данных, работу в рамках другой системы и синхронизацию с ней.

В качестве дальнейших перспектив развития хотелось бы отметить желание применить некоторые возможности библиотеки *Boost*:

1. Использовать *Boost.Units* для проверки соответствия типов физических величин в описываемых моделях различных классов МПО во время компиляции программы, что в конечном итоге повысит качество продукта в целом.

2. Применить *Boost.Numeric.Odeint* для внедрения гибкого механизма интегрирования с возможностями аппаратного ускорения и выбора различных вычислительных методов интегрирования.

3. Использовать *Quaternions* из *Boost.Math* для хранения и реализации поворотов МПО вместо углов Эйлера.

Список литературы

1. Данилова С. К., Кузьмин С. В., Кусков И. М. Теоретические и методические основы разработки управления движением морских подводных объектов в режиме "без хода" // Труды XII Всеросс. совещания по проблемам управления (ВСПУ-2014). М.: ИПУ РАН, 2014. С. 3553—3563.

2. Васильев С. Н., Данилова С. К. Имитационное моделирование как метод исследования и проектирования комплексной системы управления классом морских подводных объектов //

Труды 6-й Всеросс. научно-практической конференции "Имитационное моделирование. Теория и практика" (ИММОД-2013). Казань: Изд-во "ФЭН" Академии наук РТ, 2013. № 1. С. 35—45.

3. Аминев Б. Д., Данилова С. К. Использование пакета OpenFOAM для исследования шумовых характеристик морского подводного объекта // Известия ЮФУ. Технические науки. 2015. № 1 (162). С. 41—49.

4. **Boost C++ Libraries** [Электронный ресурс]. URL: <http://www.boost.org/> (дата обращения: 14.09.2016).

5. **Qt Home** [Электронный ресурс]. URL: <https://www.qt.io> (дата обращения: 14.09.2016).

6. **Qwt User's Guide: Qt Widgets for Technical Applications** [Электронный ресурс]. URL: <http://qwt.sourceforge.net/> (дата обращения: 14.09.2016).

7. **OGRE — Open Source 3D Graphics Engine | Home of a marvelous rendering engine** [Электронный ресурс]. URL: <http://www.ogre3d.org/> (дата обращения: 14.09.2016).

8. **SQLite Home Page** [Электронный ресурс]. URL: <https://www.sqlite.org/> (дата обращения: 14.09.2016).

Design Automation for Marine Underwater Object Route Control

B. D. Aminev, daianovich@mail.ru, S. K. Danilova, lab45_1@ipu.ru✉,

V. A. Trapeznikov Institute of Control Sciences of the Russian Academy of Sciences, 117997, Moscow

Corresponding author: **Danilova Svetlana K.**, Ph.D., Leading Researcher, V. A. Trapeznikov Institute of Control Sciences of the Russian Academy of Sciences, Moscow, 117997, Russian Federation, e-mail: lab45_1@ipu.ru

Received on September 14, 2016

Accepted on September 23, 2016

The paper concerns the design automation for the marine underwater object route control using "Wanderer" software. Marine underwater object (MUO) model is presented as a complex system of the non-linear differential equations of a high order. MUO control is a network structure by its nature (performed from several control stations). It is multidimensional, multichannel and carried out with the constrained maneuvering phase coordinates using several control facilities of the hydrodynamic and hydrostatic nature. The former group includes the hydrodynamic planes (rudder and planes), which allow MUO maneuver in space. The hydrodynamic plane's control efficiency depends on MUO velocity and their steering angles, permissive and actual values. The hydrostatic control facilities (balloons) are used, when a vessel is stopped. During the considered control mode we use balloons to make the ballast compensate for the squeezing forces and forces resulting from hydrology. Development of MUO maneuvering control algorithms, which are able to work correctly for several control modes, should be performed during investigation of MUO maneuvers with constraints on the intensity and a set of control facilities using a full-scale model of the object under consideration. In the process of realization of the route we have to ensure a maneuvering mode, in which MUO will make as little noises (acoustic, e. g.) as possible. In order to accomplish this task a muted noise maneuvering mode is used. This mode minimizes the cavitation noises based on an object's phase coordinates monitoring. "Wanderer" allows us to perform an automated maneuvering of MUO along the route in space utilizing the muted noise maneuvering mode algorithm with the additional phase coordinates' and object control facilities' constraints. "Wanderer" project gave an impulse to evolution and improvement of the facilities of the developed full-scale simulation system and allowed us to test many approaches and solve interesting problems. "Wanderer" has already been used as a part of a bigger project for solving of the set tasks and it produced good results.

Keywords: automation, control, full-scale simulation, marine underwater object, trajectory maneuvering, muted noise maneuvering, submarine

Acknowledgements: This work was supported by RFBR grant № 15-08-05133a.

For citation:

Aminev B. D., Danilova S. K. Design Automation for Marine Underwater Object Route Control, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2017, vol. 18, no. 3, pp. 202—207.

DOI: 10.17587/mau.18.202-207

References

1. Danilova S. K., Kuz'min S. V., Kuskov I. M. *Teoreticheskie i metodicheskie osnovy razrabotki upravleniya dvizheniem morskikh podvodnykh ob'ektov v rezhime "bez hoda"* (Theoretical and methodical foundation for development of control for marine underwater objects motion in "zero speed" mode), *Trudy XII Vserossiyskogo soveshchaniya po problemam upravleniya (VSPU-2014)*, 2014, pp. 3553—3563 (in Russian).

2. Vasil'ev S. N., Danilova S. K. *Imitacionnoe modelirovanie kak metod issledovaniya i proektirovaniya kompleksnoy sistemy upravleniya klassom morskikh podvodnykh ob'ektov* (Imitational modeling as re-

search and design method for the complex control system of the marine mobile objects class), *Trudy 6-j Vserossiyskoy nauchno-prakticheskoy konferencii "Imitacionnoe modelirovanie. Teoriya i praktika"* (IMMOD-2013), 2013, no. 1, pp. 35—45 (in Russian).

3. Aminev B. D., Danilova S. K. *Ispol'zovanie paketa openfoam dlya issledovaniya shumovykh harakteristik morskogo podvodnogo ob'ekta* (Application of OpenFOAM package in research of marine underwater object noise characteristics), *Izvestiya Jufu. Tehnicheskie Nauki*, 2015, no. 1 (162), pp. 41—49 (in Russian).

4. **Boost C++ Libraries**, available at: <http://www.boost.org/> (accessing date: 14.09.2016).

5. **Qt Home** available at: <https://www.qt.io> (accessing date: 14.09.2016).

6. **Qwt User's Guide: Qt Widgets for Technical Applications** available at: <http://qwt.sourceforge.net/> (accessing date: 14.09.2016).

7. **OGRE — Open Source 3D Graphics Engine | Home of a marvelous rendering engine**, available at: <http://www.ogre3d.org/> (accessing date: 14.09.2016).

8. **SQLite Home Page**, available at: <https://www.sqlite.org/> (accessing date: 14.09.2016).