Санкт-Петербургский государственный университет

ЯКОВЛЕВ Алексей Александрович

Выпускная квалификационная работа

Синтез алгоритмов управления для динамического позиционирования автономного необитаемого подводного аппарата

Уровень образования: магистратура
Направление 01.04.02 «Прикладная математика и информатика»
Основная образовательная программа ВМ.5504 «Исследование операций и системный анализ»

Научный руководитель: доктор физ.-мат. наук, профессор, заведующий кафедрой компьютерных технологий и систем СПбГУ **Веремей Е. И.**

Рецензент: доктор техн. наук, профессор, заведующий кафедрой теплофизических основ судовой энергетики СПбГМТУ **Дядик А. Н.**

Санкт-Петербург
2020

Содержание

Введение

По мере развития научно-технического прогресса появляются возможности для решения широкого спектра задач, связанных с деятельностью в недоступных или опасных для человека подводных регионах. К таким задачам относятся:

- прокладка и обслуживание подводных кабелей и трубопроводов;
- подводная геологическая разведка;
- исследование морской флоры и фауны;
- подводная береговая охрана

и многие другие. Потребность в их решении порождает рост интереса к созданию и эксплуатации всё более совершенных автономных необитаемых подводных аппаратов, которые, в частности, должны быть способны самостоятельно и с высокой точностью стабилизироваться в некотором желаемом положении.

Такой режим функционирования носит название «динамическое позиционирование», а создаваемая для него система автоматического управления должна, используя информацию о текущем и желаемом состояниях аппарата, формировать задание для его приводов таким образом, чтобы создаваемые ими силы и моменты компенсировали воздействие внешних возмущений и удерживали подводный аппарат в требуемой позиции.

Разработка современных алгоритмов управления сложными динамическими системами опирается на использование их математических моделей различных типов. Если существует техническая и экономическая возможность, то создаются аналитические модели на основе известных законов, согласно которым функционирует система. Особенно актуально это для механических систем (для них подобные модели обычно имеют вид системы дифференциальных уравнений), к которым относятся подводные аппараты.

Однако, несмотря на достаточно высокую точность подобных моделей, в их параметрах, особенно связанных с гидродинамикой аппарата, всегда присутствует неопределённость. Также синтез закона управления на основе сложной нелинейной модели является крайне нетривиальной задачей.

Перечисленные обстоятельства совокупно со существенной нелинейностью режима динамического позиционирования служат причиной создания в данной работе моделей и алгоритмов управления с применением методов нечёткой логики (учитывая аналитическую модель), которые позволяют синтезировать закон управления с помощью хорошо изученных подходов, адаптируя его при этом к текущим условиям функционирования системы.

Постановка задачи

В данной работе в качестве объекта управления рассматривается автономный необитаемый подводный аппарат (АНПА) со следующей конфигурацией исполнительных органов:

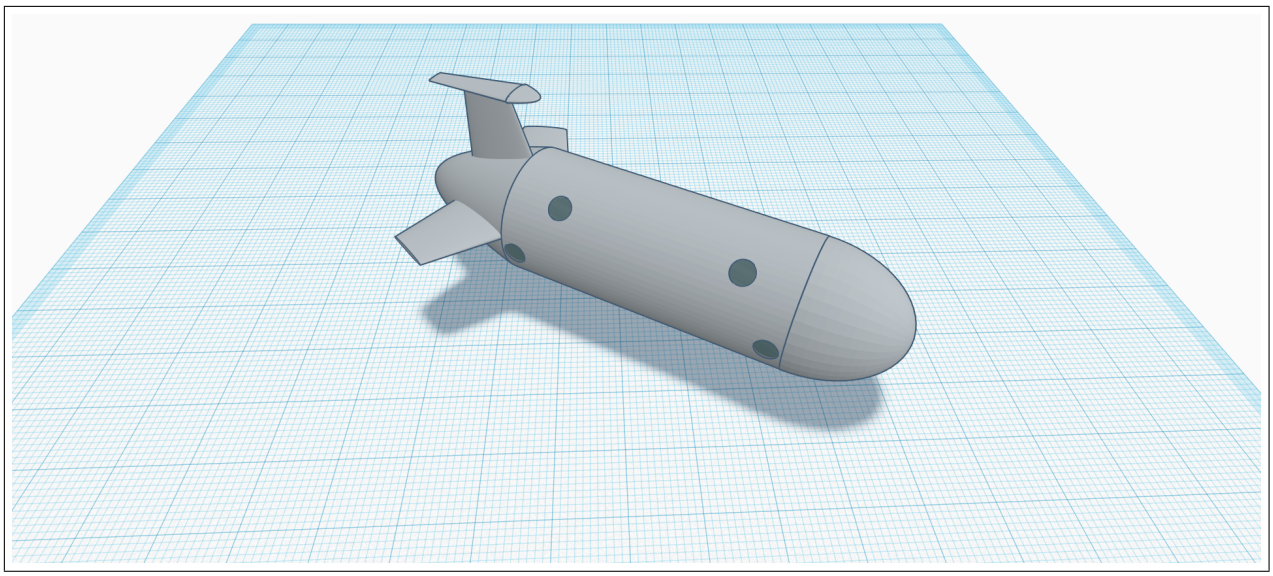

Рис. 1. Приблизительный внешний вид объекта управления

- гребной винт;
- три гидродинамических руля (один вертикальный и два горизонтальных);
- шесть туннельных подруливающих устройств (четыре вертикальных и два горизонтальных).

Цель управления заключается в осуществлении динамического позиционирования (ДП) АНПА, то есть в стабилизации аппарата в окрестности некоторого желаемого положения, где желаемое положение — это вектор, включающий в себя координаты объекта управления в некоторой неподвижной системе координат и углы его поворота. При этом в ходе ДП аппарат может быть подвержен воздействию постоянного (или медленно меняющегося) внешнего возмущения — подводного течения.

Необходимость в ДП подводного аппарата возникает при решении широкого спектра практических задач; в контексте данной работы — при выполнении манёвров по посадке и всплытию АНПА с подводной платформы.

Формально задача ДП АНПА формулируется следующим образом: пусть $x_v(t)$ — вектор проекций линейных и угловых скоростей аппарата на оси связанной системы координат; $x_p(t)$ — вектор линейных перемещений и углов поворота АНПА в некоторой неподвижной системе координат; $\widetilde{x}_p$ — желаемое значение вектора $x_p(t)$. Необходимо обеспечить глобальную асимптотическую устойчивость положения равновесия [47]

$$x_v(t) = 0,\ x_p(t) = \widetilde{x}_p.$$

При посадке на посадочную платформу, например, $\widetilde{x}_p$ — нулевой вектор, так как в ходе позиционирования за начало неподвижной системы координат принимается центр величины платформы, а углы поворота отсчитываются как отклонения от её ориентации.

Так как динамическое позиционирование осуществляется при малых величинах скорости хода АНПА (что существенно снижает эффективность гидродинамических рулей), то для реализации управления используются только подруливающие устройства и гребной винт. При этом на управление накладываются ограничения, обусловленные наличием собственной динамики приводов аппарата, а также физических ограничений на максимальные абсолютные значения управляющих воздействий, ими производимых.

Для достижения поставленной цели необходимо решить следующие задачи:

- в связи с тем, что модель объекта управления представлена нелинейной системой алгебро-дифференциальных уравнений с крайне сложной структурой, необходимо создать её аппроксимацию, подходящую для синтеза на её основе закона управления;
- синтезировать стабилизирующий закон управления, позволяющий решить задачу динамического позиционирования АНПА;
- разработать программный комплекс (реализующий, в числе прочего, математическую модель объекта управления и синтезированный регулятор), с помощью которого осуществить имитационное компьютерное моделирование процессов управления.

Обзор литературы

В ходе работы над решением задачи синтеза алгоритмов управления для ДП АНПА было изучено большое количество разноплановой литературы, как непосредственно связанной с управлением подводными аппаратами, так и посвящённой общим вопросам различных подходов к управлению динамическими системами.

Из наиболее важных для данного исследования трудов можно выделить следующие группы:

- [36] — в книге доступно объясняется концепция нечёткой логики и системы нечёткого вывода, а также демонстрируются различные варианты симбиоза нечёткой логики, нейронных сетей и эволюционных алгоритмов;
- [7, 25, 33, 52] — книги посвящены различным вопросам, связанным с нечёткой логикой и её использованием в задачах моделирования и управления, могут служить основой для ознакомления с данной темой;
- [38, 39, 40, 41, 42, 48, 49] — основополагающие статьи о системе нечёткого вывода типа Такаги–Сугено (T–S FIS), её устойчивости, синтезе на её основе нечётких регуляторов (PDC) и идентификации систем в виде T–S FIS;
- [43] — книга во многом является компиляцией результатов статей из предыдущего пункта, служит основным источником информации о различных методах синтеза PDC на основе T–S FIS;
- [15, 21, 22, 23] — в статьях и книге поднимаются вопросы, связанные с синтезом нечётких наблюдателей, использованием их в составе системы управления и устойчивостью замкнутой системы, а также исследуются пределы возможностей нечёткого наблюдения и регулирования;
- [3, 45] — статьи посвящены использованию PDC в задаче управления квадрокоптером и наглядно демонстрируют эффективность практического применения данного подхода;

- [31, 44, 46] — в статьях рассматривается использование различных вариантов функций Ляпунова в процессе синтеза нечётких законов управления, основной целью является разработка менее консервативных подходов к синтезу PDC;
- [26] — книга посвящена линейным матричным неравенствам (LMI) в контексте теории управления, затрагивает в том числе вопросы, связанные с численным решением различных задач на базе LMI и используемыми при этом алгоритмами;
- [16, 17, 18, 19, 20] — статьи являются основными источниками информации о методологии velocity-based линеаризации, поднимаются, среди прочего, вопросы моделирования нелинейной системы с помощью семейства velocity-based линеаризаций и разработки на их основе регуляторов различного вида;
- [4, 12, 24, 29] — статьи, посвящённые вопросам практической реализации velocity-based регуляторов, в [4, 24] при построении закона управления используется операция численного дифференцирования, в [12] предлагается альтернативный подход с использованием интегральной составляющей и наблюдателя, [29] рассматривает вопросы качества аппроксимации нелинейной системы с помощью семейства velocity-based линеаризаций;
- [51] — обзорная статья, посвящённая навигации и управлению морскими подвижными объектами с использованием нечёткой логики;
- [14, 30, 50] — статьи посвящены решению задачи динамического позиционирования морского судна, используются различные подходы, базирующиеся на использовании нечётких законов управления;
- [5] — статья рассматривает вопрос построения T–S FIS и PDC на основе модели движения АНПА в горизонтальной плоскости;
- [37] — материал статьи позволяет лучше понять принципы, которыми следует руководствоваться при выборе вектора параметров в ходе построения T–S FIS на основе модели движения АНПА.

Глава 1. Теория, методы и алгоритмы

В настоящей главе приводится краткое изложение основных теоретических результатов, на которых базируется данная работа, а также обобщённое описание подхода, выбранного для решения поставленной задачи.

1.1. Система нечёткого вывода Такаги–Сугено

В данной работе синтез закона управления базируется на представлении нелинейного объекта управления в виде системы нечёткого вывода, предложенной Такаги и Сугено [40] (далее T–S FIS — Takagi–Sugeno fuzzy inference system). T–S FIS описывается набором нечётких If–Then правил, которые представляют собой локальные линейные преобразования от входа нелинейной системы к её выходу. Основной особенностью T–S FIS является представление локальной динамики каждой нечёткой импликации (каждого нечёткого правила) в виде линейной модели.

В итоге нечёткая модель рассматриваемой системы образуется взятием взвешенного среднего локальных линейных моделей, что может интерпретироваться как их интерполяция. Доказано, что с помощью T–S FIS возможно аппроксимировать широкий класс нелинейных преобразований с любой заданной степенью точности [43].

1.1.1. Структура T–S FIS

В T–S FIS i-ое правило имеет вид

$$\textbf{If } z_1(t) \textit{ is } M_{i1} \textbf{ and } ... \textbf{ and } z_p(t) \textit{ is } M_{ip},$$
$$\textbf{Then } \begin{cases} \dot{x}(t) = A_i x(t) + B_i u(t), \\ y(t) = C_i x(t), \end{cases} \quad i = 1,\, 2,...,\, r. \tag{1}$$

Здесь $t \geqslant 0$ — время; M_{ij} — нечёткое множество, заданное на диапазоне возможных значений $z_j(t)$ с помощью своей функции принадлежности [33, 36] (далее MF — membership function); r — число нечётких правил; $x(t) \in R^n$ — состояние системы; $u(t) \in R^m$ — вход системы; $y(t) \in R^q$ — выход системы; $A_i \in R^{n \times n}$, $B_i \in R^{n \times m}$, $C_i \in R^{q \times n}$; $z_1(t)$,..., $z_p(t)$ — известные переменные, которые могут быть функциями от компонент $x(t)$, $u(t)$, внешних возмущений и/или времени. Функции $A_i x(t) + B_i u(t)$ и $C_i x(t)$, $i = 1,\, 2,...,\, r$,

являющиеся выходами нечётких правил, далее будут называться «подсистемами».

Для заданной пары $(x(t),\, u(t))$ подсистемы (1) преобразуются следующим образом:

$$\dot{x}(t) = \frac{\sum\limits_{i=1}^{r} w_i(z(t))[A_i x(t) + B_i u(t)]}{\sum\limits_{i=1}^{r} w_i(z(t))} = \sum_{i=1}^{r} h_i(z(t))[A_i x(t) + B_i u(t)], \qquad (2)$$

$$y(t) = \frac{\sum\limits_{i=1}^{r} w_i(z(t)) C_i x(t)}{\sum\limits_{i=1}^{r} w_i(z(t))} = \sum_{i=1}^{r} h_i(z(t)) C_i x(t), \qquad (3)$$

где

$$\begin{aligned} z(t) &= \Big(z_1(t) \quad z_2(t) \ \ldots \ z_p(t)\Big)^T, \\ w_i(z(t)) &= \prod_{j=1}^{p} M_{ij}(z_j(t)), \qquad (4) \\ h_i(z(t)) &= \frac{w_i(z(t))}{\sum\limits_{j=1}^{r} w_j(z(t))} \end{aligned}$$

для всех t. $M_{ij}(z_j(t))$ обозначает степень принадлежности $z_j(t)$ к нечёткому множеству M_{ij}. При этом, так как

$$\begin{cases} \sum\limits_{i=1}^{r} w_i(z(t)) > 0, \\ w_i(z(t)) \geqslant 0, \quad i = 1,\, 2,\ldots,\, r, \end{cases}$$

то для всех t выполняется свойство

$$\begin{cases} \sum\limits_{i=1}^{r} h_i(z(t)) = 1, \\ h_i(z(t)) \geqslant 0, \quad i = 1,\, 2,\ldots,\, r, \end{cases}$$

то есть уравнения (2), (3) являются выпуклой комбинацией подсистем (1), что в дальнейшем позволит использовать в процессе синтеза коэффициентов закона управления современные численные методы выпуклой оптимизации, гарантирующие, в зависимости от поставленной задачи, нахождение допустимого или оптимального решения [26].

1.1.2. Устойчивость T–S FIS

Анализ устойчивости T–S FIS осуществляется с использованием теории устойчивости Ляпунова. Одним из подходов является поиск допустимого решения для системы линейных матричных неравенств Ляпунова. При отсутствии внешнего воздействия условия устойчивости системы (2) формулируются следующей теоремой [43]:

Теорема 1. *Положение равновесия нечёткой системы (2) при* $u(t) \equiv 0$ *является глобально асимптотически устойчивым (GAS — globally asymptotically stable), если существует положительно определённая матрица* P*, такая что*

$$A_i^T P + P A_i < 0, \quad i = 1, 2, ..., r, \tag{5}$$

то есть такая матрица P *существует для всех подсистем.*

В (5) и далее в контексте линейных матричных неравенств (LMI — linear matrix inequalities) знаки $<$, $\leqslant$, $>$, $\geqslant$ обозначают, соответственно, отрицательную определённость, отрицательную полуопределённость, положительную определённость, положительную полуопределённость.

В теореме 1 при исследовании устойчивости системы (2) используется единственная квадратичная функция Ляпунова

$$V(x(t)) = x^T(t) P x(t),\ P > 0, \tag{6}$$

что, особенно при достаточно большом количестве If–Then правил, приводит к консервативным результатам. Существуют менее консервативные подходы с использованием других функций Ляпунова (например, нечётких квадратичных функций Ляпунова [44, 46]), однако их использование приводит к усложнению условий устойчивости (что при большом количестве подсистем затрудняет вычисления) и необходимости в использовании дополнительной информации (например, вычислении или оценке $\dot{h}_i(z(t))$, $i = 1, 2, ..., r$).

Таким образом в данной работе основой для синтеза стабилизирующего закона управления являются условия (5) (с возможными модификациями в рамках использования функции Ляпунова (6)).

1.2. Параллельный распределённый регулятор

Концепция параллельного распределённого регулирования [42, 43, 48, 49] (PDC — parallel distributed compensation/compensator) заключается в синтезе нечёткого закона управления на основе имеющейся модели объекта управления в форме T–S FIS. Каждое нечёткое If–Then правило регулятора соответствует нечёткому If–Then правилу модели. Нечёткие множества PDC совпадают с нечёткими множествами модели (то есть совпадают наборы переменных $z_j(t)$, $j = 1, 2,..., p$ и заданные на диапазонах их возможных значений функции принадлежности).

Одним из преимуществ данного подхода к управлению нелинейными системами является его простота. Большинство других методов требуют значительного объёма специализированных знаний и сложны для понимания.

1.2.1. Структура PDC

Для статического PDC по состоянию, соответствующего T–S FIS (1), i-ое правило имеет вид

$$\begin{aligned} &\textbf{If } z_1(t) \textit{ is } M_{i1} \textbf{ and } ... \textbf{ and } z_p(t) \textit{ is } M_{ip}, \\ &\quad \textbf{Then } u(t) = -F_i x(t), \quad i = 1, 2,..., r. \end{aligned} \tag{7}$$

Подсистемы PDC (7) представляют собой линейные законы управления. В данном случае — статические регуляторы по состоянию $x(t)$, возможны также варианты со статическими или динамическими регуляторами по выходу $y(t)$, однако в любом из упомянутых случаев все подсистемы должны иметь одинаковую структуру.

Для заданного $x(t)$ подсистемы (7) преобразуются следующим образом:

$$u(t) = -\frac{\sum\limits_{i=1}^{r} w_i(z(t)) F_i x(t)}{\sum\limits_{i=1}^{r} w_i(z(t))} = -\sum_{i=1}^{r} h_i(z(t)) F_i x(t). \tag{8}$$

Процесс синтеза регулятора заключается в поиске локальных коэффициентов обратной связи $F_i,\ i = 1, 2,..., r$. При этом, несмотря на построение нечёткого регулятора (8) с использованием локальных законов управле-

ния (7), для гарантий GAS замкнутой системы F_i должны определяться через единую процедуру синтеза. Также возможна апостериорная проверка условий GAS с уже синтезированными некоторым образом локальными регуляторами — такой вариант может быть рассмотрен при достаточно малом количестве подсистем и понимании желаемых динамических характеристик переходных процессов (желаемых спектров матриц замкнутой системы) для каждой из них.

1.2.2. Устойчивость замкнутой системы

Уравнения замкнутой системы получаются подстановкой (8) в качестве управления $u(t)$ в (2):

$$\dot{x}(t) = \sum_{i=1}^{r}\sum_{j=1}^{r} h_i(z(t))h_j(z(t))[A_i - B_iF_j]x(t). \tag{9}$$

Введя обозначение $G_{ij} = A_i - B_iF_j$, (9) можно переписать в виде

$$\begin{aligned}\dot{x}(t) =& \sum_{i=1}^{r} h_i(z(t))h_i(z(t))G_{ii}x(t) + \\ &+ 2\sum_{i=1}^{r}\sum_{j=i+1}^{r} h_i(z(t))h_j(z(t))\left(\frac{G_{ij}+G_{ji}}{2}\right)x(t).\end{aligned} \tag{10}$$

Применением условий GAS теоремы 1 к системе (10) получаются условия GAS для замкнутой системы [43]:

Теорема 2. *Положение равновесия нечёткой управляемой системы (10) является глобально асимптотически устойчивым, если существует матрица $P > 0$, такая что*

$$G_{ii}^T P + PG_{ii} < 0, \quad i = 1,\, 2,...,\, r, \tag{11}$$

$$\left(\frac{G_{ij}+G_{ji}}{2}\right)^T P + P\left(\frac{G_{ij}+G_{ji}}{2}\right) \leqslant 0, \quad i < j \;\; s.t. \;\; h_i \cap h_j \neq \emptyset. \tag{12}$$

При этом, если $B_1 = B_2 = ... = B_r = B$, то достаточно выполнения условий (11).

Обозначение $h_i \cap h_j \neq \emptyset$ в (12) следует понимать как возможность одновременной ненулевой активности правил с индексами i и j (определяется исходя из вида используемых функций принадлежности).

Таким образом, для проверки условий GAS системы (10) необходимо найти матрицу P, удовлетворяющую системе LMI:

$$P > 0,\ (11),\ (12),$$

или определить, что такой матрицы P не существует. Это выпуклая проблема поиска допустимого решения, которая эффективно решается современными численными методами [8, 26, 34, 35].

Для получения менее консервативных результатов при достаточно большом количестве подсистем, возможно использование облегчённых условий GAS [43]:

Теорема 3. *Пусть число правил, одновременно активных при любом t, меньше или равно s, где $1 < s \leqslant r$. Положение равновесия нечёткой управляемой системы (10) является глобально асимптотически устойчивым, если существуют матрицы $P > 0,\ Q \geqslant 0$, такие что*

$$G_{ii}^T P + P G_{ii} + (s-1)Q < 0, \quad i = 1,\, 2,...,\, r, \tag{13}$$

$$\left(\frac{G_{ij}+G_{ji}}{2}\right)^T P + P\left(\frac{G_{ij}+G_{ji}}{2}\right) - Q \leqslant 0, \quad i < j \ s.t.\ h_i \cap h_j \neq \emptyset. \tag{14}$$

При этом, если $B_1 = B_2 = ... = B_r = B$, то достаточно выполнения условий (13).

В теоремах 2, 3 предполагается, что $h_i(z(t))$ в (2) и $h_i(z(t))$ в (8) совпадают для всех t, в противном случае следует использовать условия

$$G_{ij}^T P + P G_{ji} < 0, \quad i,\, j = 1,\, 2,...,\, r. \ \ [43]$$

1.2.3. Синтез статического PDC по состоянию

С помощью преобразования условий теоремы 2 GAS замкнутой системы (10) можно сформулировать задачу синтеза матриц F_i закона управления (8), стабилизирующего систему (2), как задачу поиска допустимого решения системы LMI (без преобразования указанные условия не являются LMI относительно переменных P, F_i) [43].

Для этого неравенства (11), (12) необходимо умножить слева и справа на P^{-1}, а затем подставить в них обозначения $X = P^{-1}$, $M_i = F_i X$. Таким

образом, задача синтеза заключается в поиске матриц $X > 0$, M_i, $i = 1, 2,..., r$, удовлетворяющих условиям

$$-XA_i^T - A_iX + M_i^T B_i^T + B_iM_i > 0, \quad i = 1, 2,..., r, \tag{15}$$

$$-XA_i^T - A_iX - XA_j^T - A_jX + M_j^T B_i^T + B_iM_j + \\ + M_i^T B_j^T + B_jM_i \geqslant 0, \quad i < j \ s.t. \ h_i \cap h_j \neq \emptyset. \tag{16}$$

Коэффициенты F_i стабилизирующего регулятора (8) вычисляются следующим образом:

$$P = X^{-1}, \quad F_i = M_iP, \quad i = 1, 2,..., r.$$

Аналогичным образом ставится задача синтеза с использованием результатов теоремы 3 [43]: необходимо найти матрицы $X > 0$, $Y \geqslant 0$, M_i, $i = 1, 2,..., r$, удовлетворяющие условиям

$$-XA_i^T - A_iX + M_i^T B_i^T + B_iM_i - (s-1)Y > 0, \quad i = 1, 2,..., r, \tag{17}$$

$$2Y - XA_i^T - A_iX - XA_j^T - A_jX + M_j^T B_i^T + B_iM_j + \\ + M_i^T B_j^T + B_jM_i \geqslant 0, \quad i < j \ s.t. \ h_i \cap h_j \neq \emptyset. \tag{18}$$

Описанный процесс синтеза PDC может быть модифицирован путём добавления к условию GAS замкнутой системы дополнительных ограничений, например, ограничения нормы управления $\|u(t)\|_2 \leqslant \mu \ \forall \ t \geqslant 0$ и/или ограничения нормы выхода $\|y(t)\|_2 \leqslant \lambda \ \forall \ t \geqslant 0$.

Также возможно введение различных критериев качества и, как следствие, постановка задач условной оптимизации с ограничениями в виде соответствующих LMI. Например, если вследствие требований к характеристикам замкнутой системы возникает необходимость в ускорении переходных процессов, то рассматривается задача максимизации скорости убывания функции Ляпунова (6) на решениях $x(t)$ замкнутой системы (9) при сохранении свойства GAS (что в итоге сводится к решению GEVP — generalized eigenvalue minimization problem [6, 13, 26, 43]).

Однако ***на данном этапе работы*** задача синтеза ***оптимального*** стабилизирующего PDC не рассматривается.

1.3. Построение T–S FIS

Как уже было сказано ранее, синтез PDC осуществляется на основе имеющейся модели объекта управления в форме T–S FIS. Таким образом, построение нечёткой модели является краеугольным камнем всей процедуры разработки закона управления.

Существует два основных подхода к моделированию системы в виде T–S FIS:

1. идентификация на основе экспериментальных данных от входа объекта управления к его выходу [7, 25, 36, 38, 39, 52];
2. построение T–S FIS с использованием имеющихся нелинейных уравнений (математической модели), описывающих динамику объекта управления;

при этом в каждом из вариантов также можно выделить две основные части:

1. определение структуры нечёткой модели (например, выбор переменных $z(t)$ в (1));
2. определение параметров нечёткой модели (например, поиск A_i, B_i, C_i в (1)).

Идентификационный подход к построению T–S FIS используется в том случае, если для объекта управления невозможно или слишком затруднительно получить аналитическую (в частности, физическую) модель. В противном случае предпочтительным является второй метод. Так как в контексте данной работы известна физическая модель объекта управления, то именно на её основе производится построение T–S FIS.

Далее в текущем параграфе рассматривается нелинейная модель динамики объекта управления в общем виде:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)), \\ y(t) = g(x(t), u(t)), \end{cases} \tag{19}$$

где, как и во введённых ранее обозначениях, $x(t) \in R^n$ — состояние системы; $u(t) \in R^m$ — вход системы; $y(t) \in R^q$ — выход системы; $f(\cdot, \cdot)$, $g(\cdot, \cdot)$ —

непрерывно дифференцируемые векторные функции соответствующих размерностей (также их первые производные удовлетворяют условию Липшица — свойство, необходимое в дальнейшем для использования velocity-based линеаризаций [4]).

Существуют различные методы нечёткого моделирования на основе нелинейной математической модели (19), например, использование локальной ограниченности нелинейностей модели секторами [43] (T–S FIS, полученная таким способом, представляет собой эквивалентное аналитической модели описание объекта управления в рассмотренном локальном регионе). Однако в ситуации, когда уравнения динамики крайне сложны и/или модель содержит интерполяционные таблицы и условные операторы, использование данного подхода не представляется возможным.

Альтернативу можно коротко (и не в самом общем виде) описать следующим образом:

1. Выбрать набор переменных $z(t) \in R^p$, параметризующий расширенное пространство состояний $(x(t),\, u(t))$ системы (19) (или, иначе, параметризующий функции $f(\cdot),\, g(\cdot)$). Наиболее распространённый вариант — часть вектора состояния $x(t)$ и/или различные функции от его компонент — особенно в ситуации, когда правая часть уравнений модели аффинна относительно $u(t)$:

$$\begin{cases} f(x(t), u(t)) \equiv f_1(x(t)) + f_2(x(t))u(t), \\ g(x(t), u(t)) \equiv g_1(x(t)) + g_2(x(t))u(t). \end{cases} \tag{20}$$

2. Для переменных $z_1(t)$,..., $z_p(t)$ определить диапазоны их возможных (в рамках решаемой задачи) значений, то есть найти $d_j,\, D_j$:

$$d_j \leqslant z_j(t) \leqslant D_j, \quad j = 1,\, 2,...,\, p.$$

   При этом объединение этих интервалов образует компактное подпространство в R^p.

3. На интервалах $[d_j,\, D_j]$, $j = 1,\, 2,...,\, p$, задать нечёткие множества M_{jk}, $k = 1,\, 2,...,\, r_j$. Другими словами, задать наборы MF $\mu_{jk}(\cdot) : [d_j,\, D_j] \longrightarrow [0, 1]$, $k = 1,\, 2,...,\, r_j$.

4. Для каждой из $r = \prod_{j=1}^{p} r_j$ комбинаций нечётких множеств M_{jk} (каждая такая комбинация представляет собой условие в If-части правила T–S FIS (1)) определить подсистему (Then-часть), являющуюся линейной аппроксимацией локальной динамики системы (19).

T–S FIS, построенная описанным способом, не эквивалентна модели (19), а представляет собой её аппроксимацию. Таким образом, PDC, синтезированный на основе такой нечёткой модели, не гарантирует устойчивости системы (19). Одним из возможных решений данной проблемы, помимо экспериментальной верификации работы регулятора, является синтез робастного PDC [43].

Реализация пунктов 1 — 3 описывается в параграфе 3.1 данной работы применительно к решению конкретной задачи; далее в текущем параграфе рассматривается пункт 4.

1.3.1. Семейство velocity-based линеаризаций

Одним из возможных подходов к получению семейства линейных локальных моделей, результат нечёткой интерполяции которых является аппроксимацией системы (19), носит название «velocity-based linearization» (далее — V-B линеаризация). Основные теоретические результаты, касающиеся представления нелинейной динамики в V-B форме, а также его преимущества над классическими методами, могут быть найдены в [16, 17, 18, 19, 20]. Главными достоинствами подхода являются получение линейных локальных моделей вне окрестности положений равновесия системы, а также широкие возможности численной реализации.

Разложение правой части системы (19) в ряд Тейлора для функции многих переменных в окрестности некоторой точки (x^0, u^0) расширенного пространства состояний $R^{n\times m}$ с последующим отбрасыванием всех членов ряда с частными производными порядка большего, чем первый (линеаризация), приводит к системе в отклонениях вида

$$\frac{d}{dt}(x(t) - x^0) = f(x(t) - x^0, u(t) - u^0) = \\ = f(x^0, u^0) + \frac{\partial f(x^0, u^0)}{\partial x}(x(t) - x^0) + \frac{\partial f(x^0, u^0)}{\partial u}(u(t) - u^0), \qquad (21)$$

$$y(t) - y^0 = \frac{\partial g(x^0, u^0)}{\partial x}(x(t) - x^0) + \frac{\partial g(x^0, u^0)}{\partial u}(u(t) - u^0), \tag{22}$$

где $y^0 = g(x^0, u^0)$, $\frac{\partial}{\partial x}(\cdot)$, $\frac{\partial}{\partial u}(\cdot)$ — Якобианы.

После подстановки обозначений

$$\begin{aligned}
A(x^0, u^0) &= \frac{\partial f(x^0, u^0)}{\partial x},\ B(x^0, u^0) = \frac{\partial f(x^0, u^0)}{\partial u},\\
C(x^0, u^0) &= \frac{\partial g(x^0, u^0)}{\partial x},\ D(x^0, u^0) = \frac{\partial g(x^0, u^0)}{\partial u},\\
a(x^0, u^0) &= f(x^0, u^0) - A(x^0, u^0)x^0 - B(x^0, u^0)u^0,\\
c(x^0, u^0) &= g(x^0, u^0) - C(x^0, u^0)x^0 - D(x^0, u^0)u^0
\end{aligned}$$

уравнения (21), (22) принимают вид

$$\begin{cases}
\dot{x}(t) = a(x^0, u^0) + A(x^0, u^0)x(t) + B(x^0, u^0)u(t),\\
y(t) = c(x^0, u^0) + C(x^0, u^0)x(t) + D(x^0, u^0)u(t)
\end{cases} \tag{23}$$

— локальная аффинная аппроксимация системы (19) в окрестности $(x^0,\, u^0)$.

Переход к V-B форме осуществляется дифференцированием системы (23) по времени t:

$$\begin{cases}
\dot{x}(t) = w(t),\\
\dot{u}(t) = v(t),\\
\dot{w}(t) = A(x^0, u^0)w(t) + B(x^0, u^0)v(t),\\
\dot{y}(t) = C(x^0, u^0)w(t) + D(x^0, u^0)v(t).
\end{cases} \tag{24}$$

С соответствующими начальными условиями динамика (21) и (24) эквивалентна [18], однако, в отличие от (23), система (24) линейна.

Расширенное пространство состояний $(x(t),\, u(t))$ системы (19) было параметризовано $z(t)$. Таким образом, точке $(x^0,\, u^0) \in R^{n \times m}$ ставится в соответствие $z^0 \in R^p$, а система (24) имеет вид

$$\begin{cases}
\dot{x}(t) = w(t),\\
\dot{u}(t) = v(t),\\
\dot{w}(t) = A(z^0)w(t) + B(z^0)v(t),\\
\dot{y}(t) = C(z^0)w(t) + D(z^0)v(t).
\end{cases} \tag{25}$$

Динамическая эквивалентность моделей (24), (25) зависит от произведённого процесса параметризации. Эквивалентность присутствует, если, например, правые части системы (19) имеют вид (20) и $z(t)$ содержит все компоненты $x(t)$, влияющие на систему. Однако зачастую $z(t)$ представляет собой набор переменных, ***наиболее существенно*** определяющих динамику объекта управления. В таком случае переход от (24) к (25) (от (21) к (25)) является аппроксимацией.

1.3.2. T–S FIS с V-B подсистемами

Модель (25) является локальным линейным приближением динамики системы (19) в окрестности точки $z^0 \in R^p$. Степень принадлежности $z(t)$ к данной окрестности определяется If-частью правила T–S FIS (1) с помощью вычисления $w_i(z(t))$, $i = i^0 \in \{1, 2, ..., r\}$ по формуле (4).

Обычно нечёткие множества M_{jk} задаются так, что на интервале $[d_j, D_j]$ существует единственная точка z_{jk}^0: $M_{jk}(z_{jk}^0) = 1$, $j = 1, 2, ..., p$, $k = 1, 2, ..., r_j$. Таким образом, на всех непрерывных интервалах $[d_j, D_j]$ задаётся дискретная сетка с узлами z_{jk}^0, интерполяция между которыми осуществляется с помощью соответствующих MF.

Всевозможные комбинации узлов одномерных сеток представляют собой узлы z^i, $i = 1, 2, ..., r$, p-мерной дискретной сетки в компактном подпространстве пространства R^p, образованном объединением интервалов $[d_j, D_j]$. При этом z^i — единственная точка в рассматриваемом компакте, в которой $w_i(\cdot) = 1$. Интерполяция между z^i осуществляется с помощью всего механизма T–S FIS.

Таким образом, для построения T–S FIS с V-B подсистемами необходимо синтезировать семейство из r моделей вида (24) в окрестностях точек $(x^i, u^i) \in R^{n \times m}$, соответствующих z^i, $i = 1, 2, ..., r$. Пусть в (24) $D(x^0, u^0) = 0 \ \forall \ (x^0, u^0) \in R^{n \times m}$, тогда нечёткая модель имеет вид

$$\textbf{If } z_1(t) \textit{ is } M_{i1} \textbf{ and } ... \textbf{ and } z_p(t) \textit{ is } M_{ip},$$
$$\textbf{Then} \begin{cases} \dot{w}(t) = A_i w(t) + B_i v(t), \\ \dot{y}(t) = C_i w(t), \end{cases} \quad i = 1, 2, ..., r, \tag{26}$$

$$\begin{cases} \dot{x}(t) = w(t), \\ \dot{u}(t) = v(t), \\ \dot{w}(t) = \sum\limits_{i=1}^{r} h_i(z(t))[A_i w(t) + B_i v(t)], \\ \dot{y}(t) = \sum\limits_{i=1}^{r} h_i(z(t)) C_i w(t), \end{cases} \tag{27}$$

где $A_i = \dfrac{\partial f(x^i, u^i)}{\partial x}$, $B_i = \dfrac{\partial f(x^i, u^i)}{\partial u}$, $C_i = \dfrac{\partial g(x^i, u^i)}{\partial x}$. Остальные обозначения соответствуют параграфу 1.1.

T–S FIS (27), в сущности, является дискретной (в смысле дискретизации параметрического пространства) аппроксимацией непрерывной LPV модели (25) (с аргументом $z(t)$ вместо z^0). Возможные погрешности при переходе от модели (24) к модели (25) заложены в весовых функциях $h_i(z(t))$ (идеальным вариантом были бы функции $h_i(x(t), u(t))$, или, иначе, $z(t) = (x(t),\, u(t))$).

1.4. Закон управления на основе V-B T–S FIS

Существуют различные варианты конфигурации закона управления нелинейной системой (19) на основе параметризованного семейства V-B линеаризаций [4, 12, 16, 18, 24]. Для реализации большинства из них требуется численное дифференцирование входных сигналов регулятора. Эта процедура является крайне нежелательной (даже с использованием дифференцирования с фильтрацией, как, например, в работах [4, 24]), так как добавляет в процесс разработки дополнительный этап с аппроксимацией и может привести к негативным последствиям при наличии достаточно сильного шума в каналах передачи информации.

Необходимость в дифференцировании можно избежать, разумным образом включив в регулятор интегральную составляющую [12, 16]. Для этого T–S FIS (26), (27) модифицируется следующим образом: пусть $\eta(t) \in R^q$ — командный сигнал, представляющий собой желаемое значение выхода $y(t)$ (можно интерпретировать как внешнее возмущение, которое ***не нужно компенсировать***), $w_I(t) = y(t) - \eta(t)$, $\omega(t) = \begin{pmatrix} w_I(t) & w(t) \end{pmatrix}^T$, $C_1 = C_2 = \ldots = C_r = C$ (наиболее распространённый вариант), тогда расширенная T–S FIS имеет вид

If $z_1(t)$ *is* M_{i1} **and** ... **and** $z_p(t)$ *is* M_{ip},

$$\textbf{Then} \begin{cases} \dot{\omega}(t) = A_i^{\omega}\omega(t) + B_i^{v}v(t) + B^{\dot{\eta}}\dot{\eta}(t), \\ \dot{y}(t) = C^{\omega}\omega(t), \end{cases} \quad i = 1,\, 2,...,\, r, \tag{28}$$

$$\begin{cases} \dot{\omega}(t) = \sum\limits_{i=1}^{r} h_i(z(t))[A_i^{\omega}\omega(t) + B_i^{v}v(t)] + B^{\dot{\eta}}\dot{\eta}(t), \\ \dot{y}(t) = C^{\omega}\omega(t), \end{cases} \tag{29}$$

где

$$A_i^{\omega} = \begin{pmatrix} 0 & C \\ 0 & A_i \end{pmatrix},\ B_i^{v} = \begin{pmatrix} 0 \\ B_i \end{pmatrix},\ B^{\dot{\eta}} = \begin{pmatrix} -I \\ 0 \end{pmatrix},\ C^{\omega} = \begin{pmatrix} 0 & C \end{pmatrix}.$$

Управление $v(t)$ для нечёткой системы (29) строится в форме статического PDC по расширенному состоянию $\omega(t)$:

$$\begin{aligned} &\textbf{If } z_1(t) \textit{ is } M_{i1} \textbf{ and } ... \textbf{ and } z_p(t) \textit{ is } M_{ip}, \\ &\quad \textbf{Then } v(t) = -F_i^{w_I} w_I(t) - F_i^{w} w(t) = \\ &\qquad = -F_i^{\omega}\omega(t), \quad i = 1,\, 2,...,\, r, \end{aligned} \tag{30}$$

$$v(t) = -\sum_{i=1}^{r} h_i(z(t)) F_i^{\omega}\omega(t). \tag{31}$$

Замкнутая система имеет вид

$$\begin{aligned} \dot{\omega}(t) &= \sum_{i=1}^{r} h_i(z(t)) \left[A_i^{\omega}\omega(t) - B_i^{v} \sum_{j=1}^{r} h_j(z(t)) F_j^{\omega}\omega(t) \right] + B^{\dot{\eta}}\dot{\eta}(t) = \\ &= \sum_{i=1}^{r} h_i(z(t)) \left[\left(A_i^{\omega} - B_i^{v} \sum_{j=1}^{r} h_j(z(t)) F_j^{\omega} \right) \omega(t) \right] + B^{\dot{\eta}}\dot{\eta}(t) = \\ &= \sum_{i=1}^{r}\sum_{j=1}^{r} h_i(z(t)) h_j(z(t)) [A_i^{\omega} - B_i^{v} F_j^{\omega}]\omega(t) + B^{\dot{\eta}}\dot{\eta}(t). \end{aligned} \tag{32}$$

Система (32) представлена в форме, аналогичной замкнутой системе (9), GAS которой исследовалась в параграфе 1.2 (неоднородность $B^{\dot{\eta}}\dot{\eta}(t)$ не оказывает влияния на устойчивость системы). Таким образом, синтез коэффициентов F_i^{ω} PDC (31), обеспечивающего GAS T–S FIS (29), осуществляется с использованием алгоритмов, описанных в разделе 1.2.3 параграфа 1.2 настоящей главы.

Алгоритм управления нелинейной системой (19) с помощью T–S FIS (27) и PDC (31) представлен на рис. 2 [12] (с несколько иными обозначениями) и использует нечёткую модель (27) в контуре управления в качестве наблюдателя:

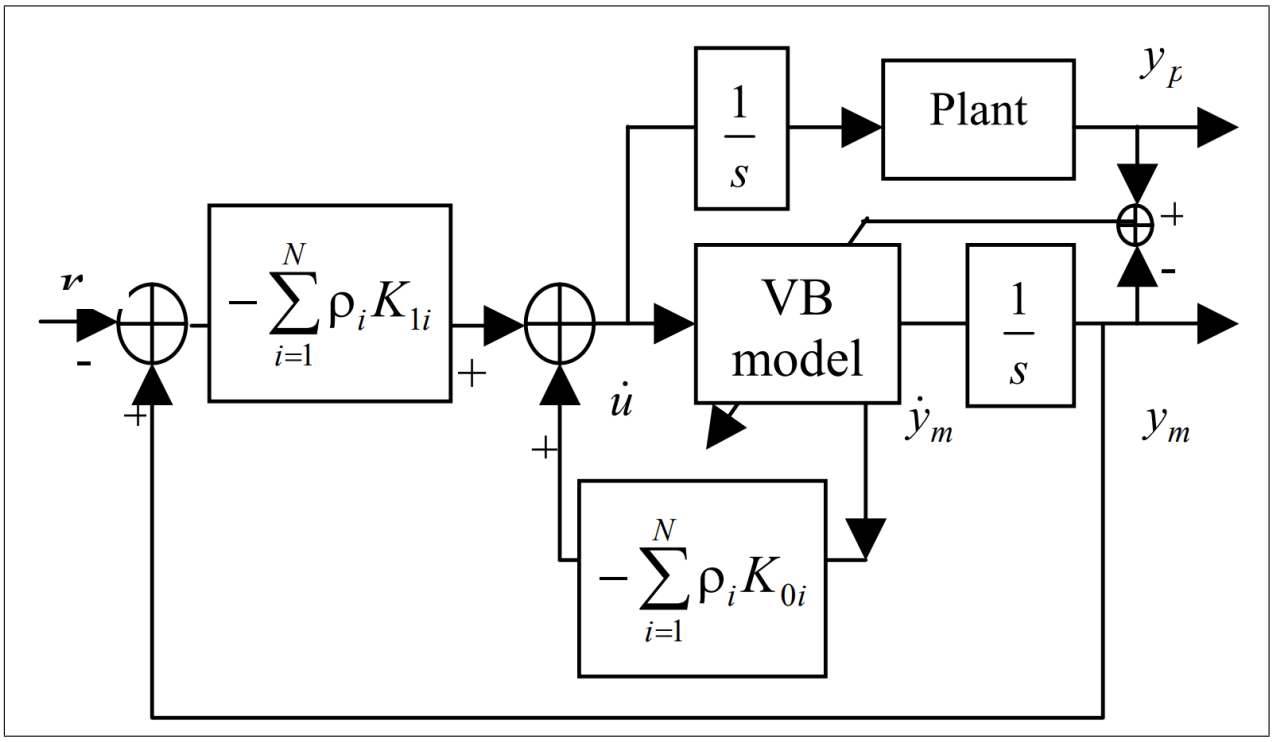

Рис. 2. [12] Блок-схема системы управления

Недостатком V-B T–S FIS является наличие статических ошибок при аппроксимации динамики исходной системы [29], однако такие модели демонстрируют хорошие результаты во время переходных процессов. Чтобы избежать накопления статических ошибок и, как следствие, расхождения решений наблюдателя и объекта управления, следует использовать выход объекта управления для корректировки состояния V-B модели [12].

На этом описание основных элементов математического аппарата, используемого на данном этапе работы для решения поставленной задачи, можно считать выполненным.

Глава 2. Математическая модель объекта управления

В настоящей главе приводится описание основных структурных элементов физической нелинейной модели объекта управления. При этом описание элементов математической модели движения АНПА, полученных для конкретного аппарата с помощью численного компьютерного и/или экспериментального моделирования (например, расчёт гидродинамических коэффициентов), не представляется возможным изложить в рамках данной работы ввиду его крайне сложной структуры и занимаемого объёма (большое количество интерполяционных функций, таблиц и т. п.).

Основой для построения модели движения морского подвижного объекта (в том числе АНПА) может служить материал, изложенный в трудах [1, 2, 10, 11, 32].

Также стоит упомянуть, что в данной главе зависимость переменных от времени t зачастую будет для краткости опускаться, а единицы измерения всех величин соответствуют системе СИ.

2.1. Общий вид математической модели движения АНПА

Для математического описания модели объекта управления вводится жёстко связанная с аппаратом декартова прямоугольная система координат $Oxyz$, а также неподвижная (земная) система координат $O_1\xi\eta\zeta$. Начало связанной системы координат расположено в центре величины объекта, оси сонаправлены с главными осями инерции АНПА.

Обобщённая математическая модель движения объектов рассматриваемого типа представляет собой алгебро-дифференциальную систему уравнений и имеет вид

$$M\dot{x}_v = F_{in}(V, \omega) + F_{hd}(V, \omega, x_p, \delta) + f_w(t), \tag{33}$$

$$\dot{x}_p = R(x_p)x_v, \tag{34}$$

$$\dot{\delta} = F_\delta(\delta, u). \tag{35}$$

Здесь

- M — матрица инерции, в компоненты которой входят присоединённые массы, статические моменты и моменты инерции АНПА, центробежные и осевые моменты АНПА, а также моменты, порождённые смещением центра масс аппарата относительно центра величины;
- $x = \{x_v, x_p\} \in E^{12}$ — вектор состояния АНПА;
- $x_v = \{V, \omega\} \in E^6$;
- $V = \{V_x, V_y, V_z\} \in E^3$ — проекции вектора линейной скорости АНПА на оси подвижной системы координат $Oxyz$;
- $\omega = \{\omega_x, \omega_y, \omega_z\} \in E^3$ — проекции вектора угловой скорости АНПА на оси подвижной системы координат $Oxyz$;
- $x_p = \{\xi, \eta, \zeta, \theta, \varphi, \psi\} \in E^6$ — вектор линейных перемещений и углов поворота АНПА (углов Эйлера) в неподвижной системе координат;
- $\delta \in E^m$ — вектор управляющих воздействий — для рассматриваемого аппарата $m = 10$, а компонентами вектора δ являются частота вращения гребного винта, отклонения гидродинамических рулей и частота вращения винтов подруливающих устройств;
- $F_{in}(\cdot)$, $F_{hd}(\cdot)$, $f_w(t)$ — внутренние силы и моменты инерционной (F_{in}) и гидродинамической (F_{hd}) природы, а также внешние силы и моменты (f_w);
- $R(x_p)$ — матрица вращения (преобразования координат из подвижной системы координат $Oxyz$ в смещённую неподвижную систему координат $O\xi\eta\zeta$);
- $F_\delta(\cdot)$ — модель динамики приводов — при этом, если управление осуществляется в пределах линейных участков соответствующих функций, то приводы можно представить линейной моделью $\dot{\delta} = u$ [2];
- $u \in E^m$ — управление, подаваемое на приводы движительного комплекса АНПА;
- $F_{in}(\cdot)$, $F_{hd}(\cdot)$, $f_w(\cdot)$, $F_\delta(\cdot)$ — нелинейные векторные функции соответствующих размерностей.

2.2. Структура уравнений движения АНПА

В данном разделе приводится описание структуры уравнений движения рассматриваемого аппарата. АНПА является симметричным относительно вертикальной продольной плоскости, длина аппарата ≈ 10 м, масса ≈ 20 т.

Система уравнений (33) может быть представлена в более общем виде:

$$M\dot{x}_v = F(t, V, \omega, x_p, \delta).$$

Здесь

$$F(t, V, \omega, x_p, \delta) \equiv F_{in}(V, \omega) + F_{hd}(V, \omega, x_p, \delta) + f_w(t);$$

$$M = \begin{pmatrix} \lambda_{11}+m & \lambda_{12} & 0 & 0 & 0 & \lambda_{16}-my_g \\ \lambda_{12} & \lambda_{22}+m & 0 & 0 & 0 & \lambda_{26}+mx_g \\ 0 & 0 & \lambda_{33}+m & \lambda_{34}+my_g & \lambda_{35}-mx_g & 0 \\ 0 & 0 & \lambda_{34}+my_g & \lambda_{44}+J_x & \lambda_{45}-J_{xy} & 0 \\ 0 & 0 & \lambda_{35}-mx_g & \lambda_{45}-J_{xy} & \lambda_{55}+J_y & 0 \\ \lambda_{16}-my_g & \lambda_{26}+mx_g & 0 & 0 & 0 & \lambda_{66}+J_z \end{pmatrix},$$

где, в свою очередь, m — масса АНПА с учётом массы воды в проницаемых объёмах, $\{x_g,\ y_g,\ 0\}$ — координаты центра масс АНПА в связанной системе координат, $J_x,\ J_y,\ J_z$ — осевые моменты инерции, J_{xy} — центробежный момент инерции, λ_{ij} — присоединённые массы, статические моменты и моменты инерции АНПА;

$$\begin{aligned}
F_1(\cdot) = &- mV_z\omega_y - \lambda_{33}V_z^{rel}\omega_y - my_g\omega_x\omega_y - \lambda_{34}\omega_x\omega_y - \lambda_{35}\omega_y^2 + mx_g\omega_y^2 + \\
&+ mV_y\omega_z + \lambda_{22}V_y^{rel}\omega_z + \lambda_{12}V_x^{rel}\omega_z + \lambda_{26}\omega_z^2 + mx_g\omega_z^2 + \\
&+ (\varrho W - m)gsin(\psi) + X_{hd} + X_\delta + T_n, \\
F_2(\cdot) = &\ mV_z\omega_x + \lambda_{33}V_z^{rel}\omega_x + (\lambda_{34} + my_g)\omega_x^2 + \lambda_{35}\omega_x\omega_y - mx_g\omega_x\omega_y - \\
&- mV_x\omega_z - \lambda_{11}V_x^{rel}\omega_z - \lambda_{12}V_y^{rel}\omega_z - \lambda_{16}\omega_z^2 + my_g\omega_z^2 + \\
&+ (\varrho W - m)gcos(\psi)cos(\theta) + Y_{hd} + Y_\delta + \\
&+ T_n^{fvl} + T_n^{fvr} + T_n^{bvl} + T_n^{bvr}, \\
F_3(\cdot) = &- mV_y\omega_x - \lambda_{22}V_y^{rel}\omega_x - \lambda_{12}V_x^{rel}\omega_x - \lambda_{26}\omega_x\omega_z - mx_g\omega_x\omega_z + \\
&+ \lambda_{11}V_x^{rel}\omega_y + mV_x\omega_y + \lambda_{12}V_y^{rel}\omega_y + \lambda_{16}\omega_y w_z - my_g\omega_y\omega_z - \\
&- (\varrho W - m)gcos(\psi)sin(\theta) + Z_{hd} + Z_\delta + T_n^{fh} + T_n^{bh},
\end{aligned}$$

$$
\begin{aligned}
F_4(\cdot) = &- J_z\omega_y\omega_z - \lambda_{66}\omega_y\omega_z - \lambda_{16}\omega_y V_x^{rel} + my_g\omega_y V_x - (\lambda_{26} + \lambda_{35})V_y^{rel}\omega_y + \\
&+ J_y\omega_z\omega_y + \lambda_{55}\omega_z\omega_y + (\lambda_{26} + \lambda_{35})V_z^{rel}\omega_z + \lambda_{45}\omega_x\omega_z - J_{xy}\omega_x\omega_z - \\
&- my_g V_y\omega_x - \lambda_{34}V_y^{rel}\omega_x + y_g mgcos(\psi)sin(\theta) + M_X^{hd} + M_X^{\delta} + M_n - \\
&- 2\pi(J_n + \Delta J_n)\dot{n} - z^{fvl}T_n^{fvl} - z^{fvr}T_n^{fvr} - z^{bvl}T_n^{bvl} - \\
&- z^{bvr}T_n^{bvr} + y^{fh}T_n^{fh} + y^{bh}T_n^{bh}, \\
F_5(\cdot) = &- J_x\omega_x\omega_z - \lambda_{44}\omega_x\omega_z - (\lambda_{16} + \lambda_{34})V_z^{rel}\omega_z - \lambda_{45}\omega_y\omega_z + J_{xy}\omega_y\omega_z + \\
&+ J_z\omega_z\omega_x + \lambda_{66}\omega_z\omega_x + \lambda_{26}\omega_x V_y^{rel} + mx_g\omega_x V_y + (\lambda_{16} + \lambda_{34})V_x^{rel}\omega_x + \\
&+ \lambda_{35}V_x^{rel}\omega_y - mx_g V_x\omega_y - x_g mgcos(\psi)sin(\theta) + M_Y^{hd} + M_Y^{\delta} - \\
&- x^{fh}T_n^{fh} - x^{bh}T_n^{bh} - 2\pi(J_n + \Delta J_n)n\omega_z, \\
F_6(\cdot) = &- J_y\omega_x\omega_y - \lambda_{55}\omega_x\omega_y - \lambda_{35}\omega_x V_z^{rel} + mx_g\omega_x V_z - \lambda_{45}\omega_x^2 - J_{xy}\omega_x^2 + \\
&+ J_x\omega_y\omega_x + \lambda_{44}\omega_y\omega_x + \lambda_{34}\omega_y V_z^{rel} + my_g\omega_y V_z + \lambda_{45}\omega_y^2 - J_{xy}\omega_y^2 - \\
&- \lambda_{26}V_x^{rel}\omega_z - mx_g V_x\omega_z + \lambda_{16}V_y^{rel}\omega_z - my_g V_y\omega_z - \\
&- x_g mgcos(\psi)cos(\theta) + y_g mgsin(\psi) + M_Z^{hd} + M_Z^{\delta} + x^{fvl}T_n^{fvl} + \\
&+ x^{fvr}T_n^{fvr} + x^{bvl}T_n^{bvl} + x^{bvr}T_n^{bvr} + 2\pi(J_n + \Delta J_n)n\omega_y.
\end{aligned}
$$

Здесь

- V_x^{rel}, V_y^{rel}, V_z^{rel} — проекции вектора скорости АНПА относительно воды на оси связанной системы координат — то есть разности между проекциями вектора абсолютной скорости аппарата на оси $Oxyz$ и проекциями вектора скорости течения на оси $Oxyz$;
- θ, φ, ψ — углы крена, рыскания и дифферента аппарата;
- n — частота вращения гребного винта (δ_1);
- J_n, ΔJ_n — моменты инерции гребного винта и вала гребного винта и вращающихся с ним частей маршевого электропривода;
- W — общий объём АНПА;
- ϱ — плотность воды;
- g — ускорение свободного падения;

- X_{hd}, Y_{hd}, Z_{hd}, M_X^{hd}, M_Y^{hd}, M_Z^{hd} — гидродинамические силы и моменты, действующие на аппарат,

$$\begin{aligned} X_{hd} &= C_x \frac{\varrho}{2} W^{\frac{2}{3}} V_{rel}^2, \\ Y_{hd} &= C_y \frac{\varrho}{2} W^{\frac{2}{3}} V_{rel}^2, \\ Z_{hd} &= C_z \frac{\varrho}{2} W^{\frac{2}{3}} V_{rel}^2, \\ M_X^{hd} &= m_x \frac{\varrho}{2} W V_{rel}^2 + m_{\omega_x} \frac{\varrho}{2} W^{\frac{5}{3}} \omega_x |\omega_x|, \\ M_Y^{hd} &= m_y \frac{\varrho}{2} W V_{rel}^2 + m_{\omega_y} \frac{\varrho}{2} W^{\frac{5}{3}} \omega_y |\omega_y|, \\ M_Z^{hd} &= m_z \frac{\varrho}{2} W V_{rel}^2 + m_{\omega_z} \frac{\varrho}{2} W^{\frac{5}{3}} \omega_z |\omega_z|; \end{aligned}$$

- V_{rel} — модуль вектора скорости АНПА относительно воды;
- X_δ, Y_δ, Z_δ, M_X^δ, M_Y^δ, M_Z^δ — гидродинамические силы и моменты, создаваемые отклонениями рулей (при этом отклонения рулей от нулевого положения — δ_2, δ_3, δ_4);
- T_n, M_n — тяга и вращательный момент от гребного винта,

$$\begin{aligned} T_n &= K_n \varrho (V_x^2 + (nD_n)^2) D_n^2, \\ M_n &= -M_{magn} + 2\pi \Delta J_n \dot{n}; \end{aligned}$$

- T_n^i — тяги, создаваемые вращением винтов подруливающих устройств,

$$T_n^i = K_n^i \varrho n_i |n_i| (D_n^i)^4 K_{T_n^i};$$

- n_i, $i \in \{fvl,\ fvr,\ bvl,\ bvr,\ fh,\ bh\}$ — частота вращения i-ого подруливающего устройства (δ_5 — δ_{10}).

Вектор управляющих воздействий δ в режиме динамического позиционирования имеет вид

$$\delta = \begin{pmatrix} n & 0 & 0 & 0 & n_{fvl} & n_{fvr} & n_{bvl} & n_{bvr} & n_{fh} & n_{bh} \end{pmatrix}^T, \tag{36}$$

при этом имеются ограничения: $|n| \leqslant 4{,}2$, $|n_i| \leqslant 15$.

Матрица преобразования координат в системе уравнений (34)

$$\dot{x}_p = R(x_p)x_v$$

имеет вид

$$R(x_p) = \begin{pmatrix} cos(\varphi)cos(\psi) & sin(\theta)sin(\varphi) - cos(\theta)cos(\varphi)sin(\psi) & cos(\theta)sin(\varphi) + sin(\theta)cos(\varphi)sin(\psi) & 0 & 0 & 0 \\ sin(\psi) & cos(\theta)cos(\psi) & -sin(\theta)cos(\psi) & 0 & 0 & 0 \\ -sin(\varphi)cos(\psi) & sin(\theta)cos(\varphi) + cos(\theta)sin(\varphi)sin(\psi) & cos(\theta)cos(\varphi) - sin(\theta)sin(\varphi)sin(\psi) & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \frac{-cos(\theta)sin(\psi)}{cos(\psi)} & \frac{sin(\theta)sin(\psi)}{cos(\psi)} \\ 0 & 0 & 0 & 0 & \frac{cos(\theta)}{cos(\psi)} & \frac{-sin(\theta)}{cos(\psi)} \\ 0 & 0 & 0 & 0 & sin(\theta) & cos(\theta) \end{pmatrix}.$$

2.3. Модификация модели движения АНПА

Для дальнейшего удобства математическая модель движения АНПА (33) — (35) приводится к более общему виду:

$$\begin{cases} M\dot{x}_v = F_{in}(V,\omega) + F_{hd}(V,\omega,x_p,\delta) + f_w(t), \\ \dot{x}_p = R(x_p)x_v, \\ \dot{\delta} = F_\delta(\delta,u); \end{cases} \Longrightarrow$$

$$\begin{cases} \dot{x}_v = M^{-1}\widehat{F}(x_v,x_p,\delta) + f_w(t), \\ \dot{x}_p = R(x_p)x_v, \\ \dot{\delta} = F_\delta(\delta,u); \end{cases} \Longrightarrow$$

$$\begin{cases} \dot{x} = F_x(x,\delta) + f_w(t), \\ \dot{\delta} = F_\delta(\delta,u). \end{cases} \tag{37}$$

При использовании подхода, изложенного в главе 1, наиболее удобным видом исходной нелинейной модели является

$$\dot{x}(t) = f(x(t)) + Bu(t), \tag{38}$$

где $x(t)\in R^n$ — состояние системы; $u(t)\in R^m$ — вход системы; $B\in R^{n\times m}$ — постоянная матрица.

Причины для этого следующие:

1. аффинный относительно $u(t)$ вид уравнений (38) позволяет (с большей долей уверенности в удовлетворительном качестве аппроксимации) использовать параметризующую переменную вида

   $$z(t) = z(x(t)),$$

   то есть избежать использования $u(t)$ в If-части T–S FIS (1), что привело бы к сложному процессу реализации нечёткого регулятора [43];

2. $u(t)$ входит в правую часть системы (38) как линейная комбинация своих компонент с постоянными коэффициентами, следовательно, в T–S FIS (2) $B_1 = B_2 = ... = B_r = B$, что, согласно результатам теорем 2, 3, значительно снижает количество LMI в процессе синтеза

PDC (8) — это крайне существенно с вычислительной точки зрения при большом количестве If–Then правил.

Однако математическая модель объекта управления (37) не соответствует виду (38). Для устранения данного препятствия необходимо произвести ряд преобразований и упрощений:

1. В качестве управляющего воздействия δ рассматривать не вектор (36), а вектор соответствующих тяг (гидродинамические рули из рассмотрения исключаются), то есть

$$\delta = \begin{pmatrix} T_n & T_n^{fvl} & T_n^{fvr} & T_n^{bvl} & T_n^{bvr} & T_n^{fh} & T_n^{bh} \end{pmatrix}^T. \tag{39}$$

Для этого предварительно требуется несколько упростить исходную модель, исключив из её уравнений некоторые составляющие, связанные с влиянием гребного винта (n, $\dot{n}$). Существенного влияния данное упрощение на свойства синтезируемых T–S FIS и PDC не оказывает.

В процессе управления АНПА необходимо совершить обратный переход — через желаемые значения тяг, полученные на выходе регулятора, рассчитать желаемые значения оборотов, которые являются входами приводов:

$$n = sign(T_n)\sqrt{|\frac{|T_n|}{|K_n|\varrho D_n^4} - \frac{V_x^2}{D_n^2}|},$$

$$n_i = sign(T_n^i)\sqrt{|\frac{|T_n^i|}{K_n^i K_{T_n^i}\varrho (D_n^i)^4}|}, \quad i{\in}\{fvl,\ fvr,\ bvl,\ bvr,\ fh,\ bh\}.$$

2. В качестве модели приводов использовать линейную модель

$$\dot{\delta} = u,$$

которая, как уже было сказано ранее, достаточно хорошо аппроксимирует динамику (35), если управление осуществляется в пределах линейных участков соответствующих функций [2].

После описанных действий модель (37) преобразуется в модель

$$\frac{d}{dt}\begin{pmatrix} x(t) \\ \delta(t) \end{pmatrix} = \begin{pmatrix} \widetilde{F}_x(x(t)) + B\delta(t) \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ I \end{pmatrix} u(t), \tag{40}$$

где $\delta(t)$ имеет вид (39). Модель внешнего возмущения $f_w(t)$ на данном этапе работы в процессе синтеза не используется.

Модель (40) соответствует виду (38) (более того, правая часть зависит только от $x(t)$) и служит основой для синтеза T–S FIS и PDC. При этом в качестве выхода $y(t)$ в процессе построения PDC используется вектор x_p, то есть

$$y(t) = C \begin{pmatrix} x(t) \\ \delta(t) \end{pmatrix}, \quad C = \begin{pmatrix} 0_{6\times6} & I_{6\times6} & 0_{6\times7} \end{pmatrix}. \tag{41}$$

Глава 3. Реализация алгоритмов управления

В данной главе приводится описание элементов подхода, изложенного в главе 1, являющихся уникальными для рассматриваемой задачи ДП АНПА. Также в общих чертах описывается программный комплекс, разработанный в среде MATLAB & Simulink для реализации решения задачи, и приводятся результаты имитационного компьютерного моделирования процессов управления.

Стоит упомянуть крайне важный аспект конкретной решаемой задачи: в ходе ДП АНПА информацией о состоянии аппарата, поступающей от посадочной платформы, является вектор x_p. Эти данные крайне точны и наилучшим образом подходят для использования в качестве обратных связей, особенно с учётом высоких требований к допустимым погрешностям позиционирования.

Однако далее будет показано, что для формирования вектора $z(t)$ используются линейные и угловые скорости АНПА в связанной системе координат (компоненты вектора x_v). Эта информация также доступна, с разной степенью точности, от бортовой навигационной системы аппарата.

При этом, с учётом худшего качества таких данных (а в большинстве приложений они и вовсе недоступны), в ходе дальнейшего развития работы имеет смысл использовать подходы, основанные на использовании $\widehat{z}(t)$ — оценке вектора $z(t)$, формируемой нечётким наблюдателем [21, 22, 23, 43] (что, однако, приводит к значительному усложнению процесса синтеза).

3.1. Аппроксимация динамики АНПА с помощью T–S FIS

Построение T–S FIS вида (27) осуществляется на основе модифицированной модели движения АНПА (40).

Согласно процедуре, описанной в параграфе 1.3 главы 1, первым этапом данного процесса является выбор набора переменных $z(t)$, параметризующий пространство состояний системы. Для этой цели следует изучить структуру уравнений аппроксимируемой модели, а также определить значимость влияния различных переменных на динамику объекта управления

при функционировании в условиях рассматриваемого режима (в том числе с помощью проведения ряда модельных экспериментов). Итогами данных действий являются следующие выводы:

- Идеальным набором является

$$z(t) = \Big(V_x \;\; V_y \;\; V_z \;\; \omega_x \;\; \omega_y \;\; \omega_z \;\; \theta \;\; \psi\Big)^T,$$

так как именно данные компоненты вектора состояния влияют на динамику скоростей аппарата x_v (входят в соответствующие уравнения движения). Угол рыскания φ влияет только на линейное смещение аппарата в неподвижной системе координат (в составе матрицы поворота $R(x_p)$), которое не входит в правую часть уравнений (40).

- Анализ условий режима ДП АНПА и серия модельных экспериментов свидетельствуют о том, что наиболее активные переходные процессы (следовательно, наиболее сильные отклонения от нулевого положения равновесия) происходят по переменным

$$\{V_x,\, V_y,\, V_z,\, \omega_y\},$$

при этом значительное влияние на динамику аппарата (и на точность T–S FIS) оказывают отклонения углов крена θ и дифферента ψ. Влияние угловых скоростей ω_x и ω_z не столько существенно, и, так как высокая размерность вектора $z(t)$ негативным образом влияет на процесс синтеза нечёткой модели, их следует исключить из рассмотрения.

Таким образом, выбирается

$$z(t) = \Big(V_x \;\; V_y \;\; V_z \;\; \omega_y \;\; \theta \;\; \psi\Big)^T. \tag{42}$$

Далее необходимо для компонент вектора (42) определить диапазоны их возможных (в рамках режима ДП) значений. С этой целью осуществляется анализ спецификаций режима динамического позиционирования, требований к динамике аппарата во время ДП, а также проводится серия

модельных экспериментов. Итогами являются следующие результаты:

$$V_x\in[-1,\ 1],\quad V_y\in[-0{,}42,\ 0{,}42],\quad V_z\in[-0{,}39,\ 0{,}39],\qquad (43)$$
$$\omega_y\in[-0{,}1,\ 0{,}1],\quad \theta\in[-0{,}17,\ 0{,}17],\quad \psi\in[-0{,}24,\ 0{,}24].$$

Следующим этапом является задание на полученных интервалах нечётких множеств. Это осуществляется с помощью параметризованных MF, параметры которых определяют их расположение и форму. Существуют различные типы MF: треугольные MF, «π-shaped» MF, «Gaussian» MF и другие [36] (реализация показана в приложении 4). Наилучшим образом при моделировании системы (40) показывают себя «π-shaped» MF:

$$f(z;a,b,c,d)=\begin{cases}0,\ z\leqslant a\\ 2\left(\dfrac{z-a}{b-a}\right)^2,\ a\leqslant z\leqslant\dfrac{a+b}{2}\\ 1-2\left(\dfrac{z-b}{b-a}\right)^2,\ \dfrac{a+b}{2}\leqslant z\leqslant b\\ 1,\ b\leqslant z\leqslant c\\ 1-2\left(\dfrac{z-c}{d-c}\right)^2,\ c\leqslant z\leqslant\dfrac{c+d}{2}\\ 2\left(\dfrac{z-d}{d-c}\right)^2,\ \dfrac{c+d}{2}\leqslant z\leqslant d\\ 0,\ z\geqslant d\end{cases}\qquad (44)$$

При этом для всех MF (44) параметры выбирались следующим образом:

$$a_{jk}=z^0_{jk-1},\quad b_{jk}=c_{jk}=z^0_{jk},\quad d_{jk}=z^0_{jk+1}.$$

Таким образом, для всех z_j одновременно активны не более двух MF, следовательно, число одновременно активных подсистем T–S FIS не превышает $2^p=2^6=64$. То есть, в обозначениях теоремы 3, $s=64$. Точки z^0_{jk} на интервалах (43) задаются функцией из приложения 1. Графики построенных MF приводятся на рис. 3 — 8:

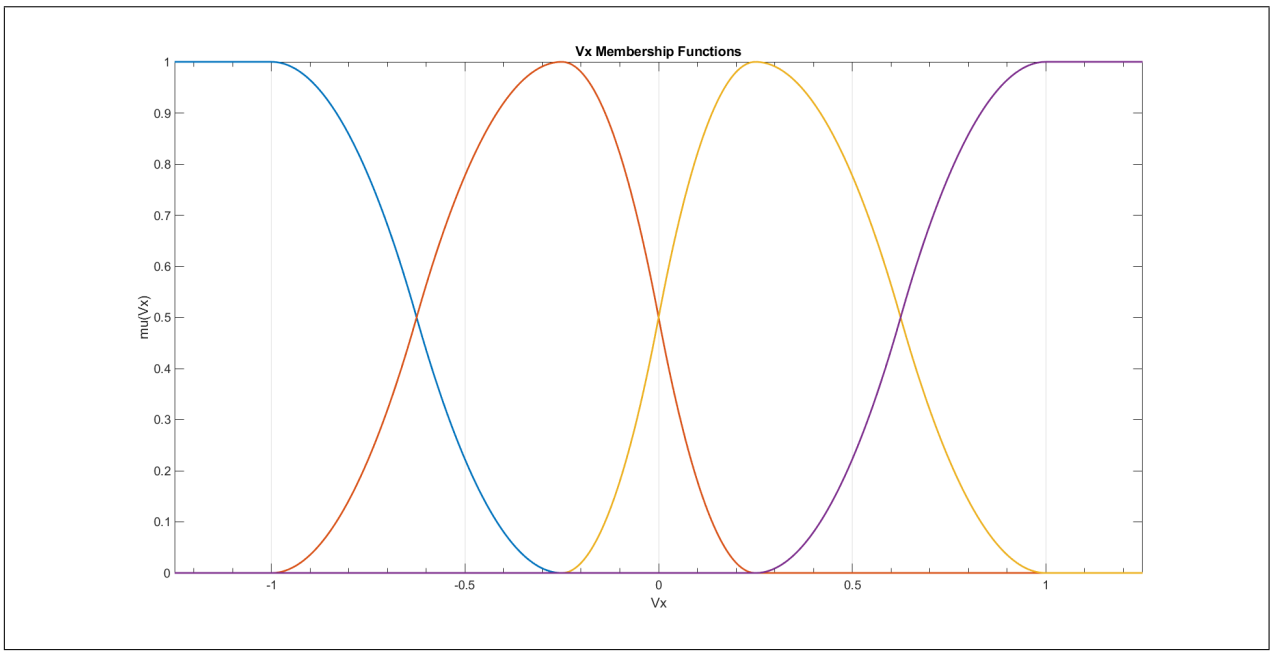

Рис. 3. Функции принадлежности для $V_x(t)$

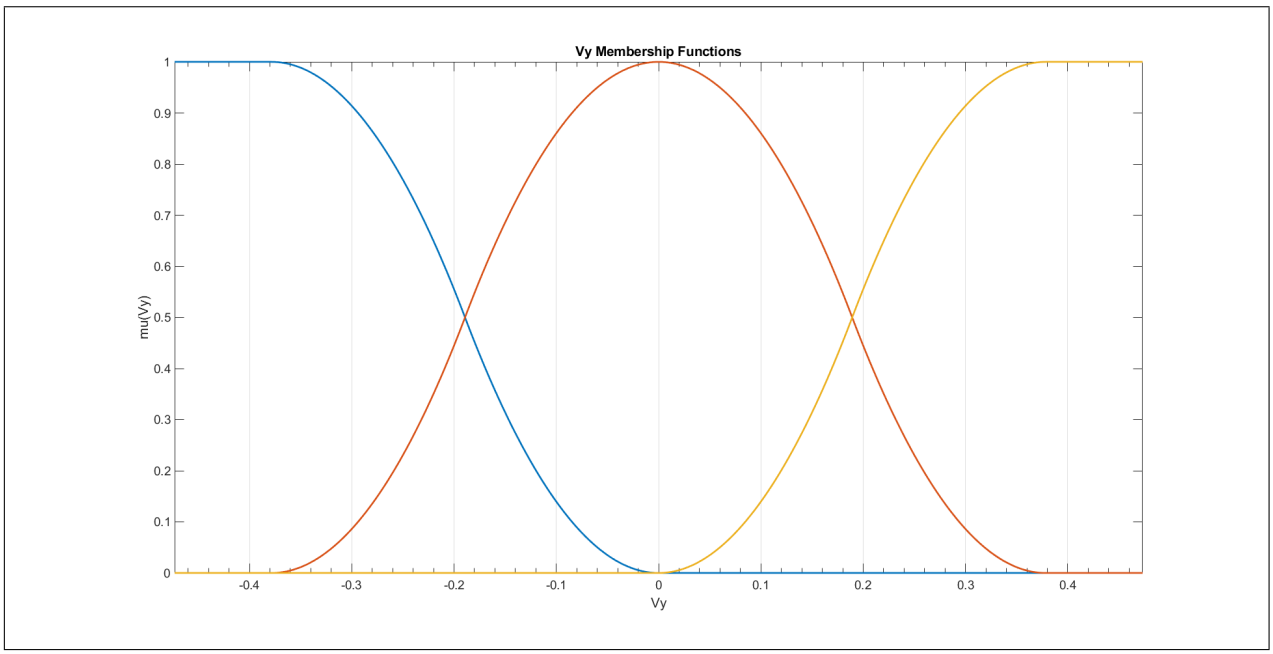

Рис. 4. Функции принадлежности для $V_y(t)$

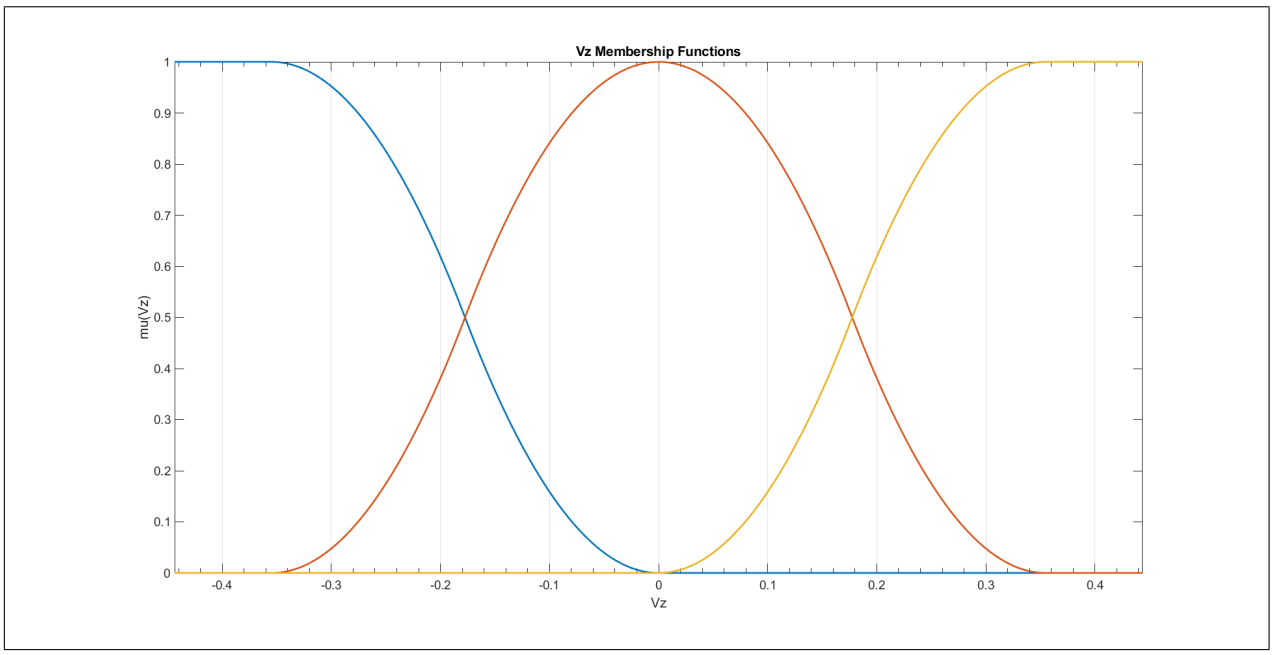

Рис. 5. Функции принадлежности для $V_z(t)$

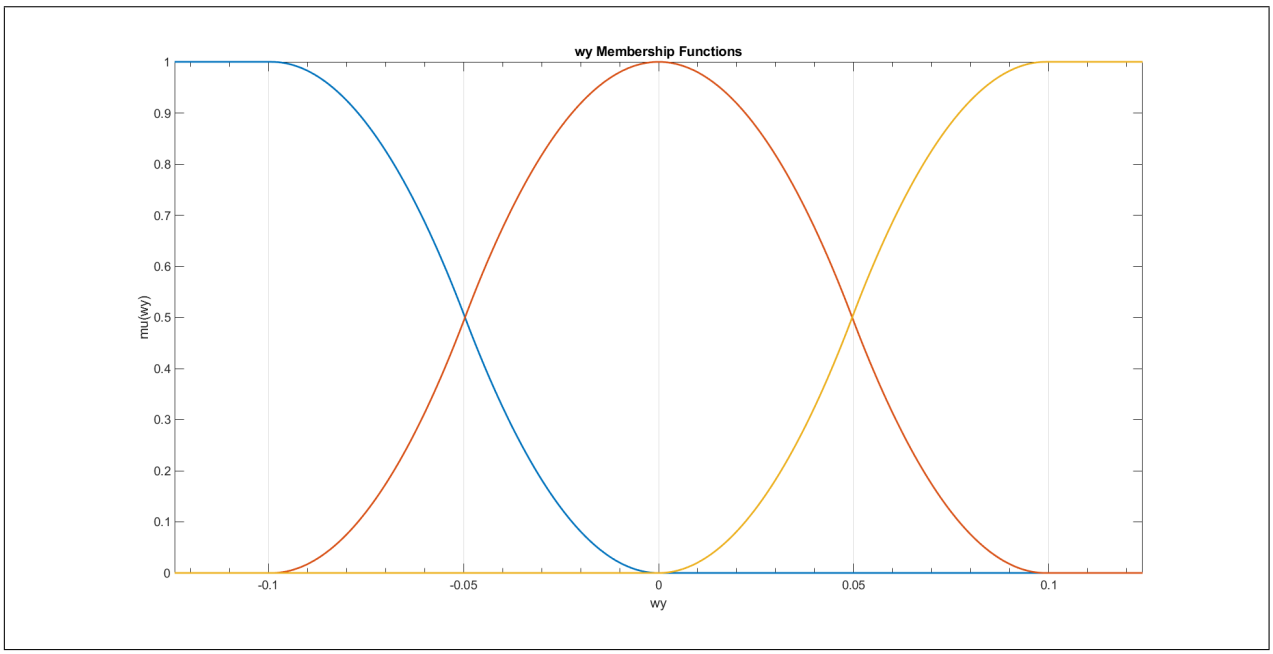

Рис. 6. Функции принадлежности для $\omega_y(t)$

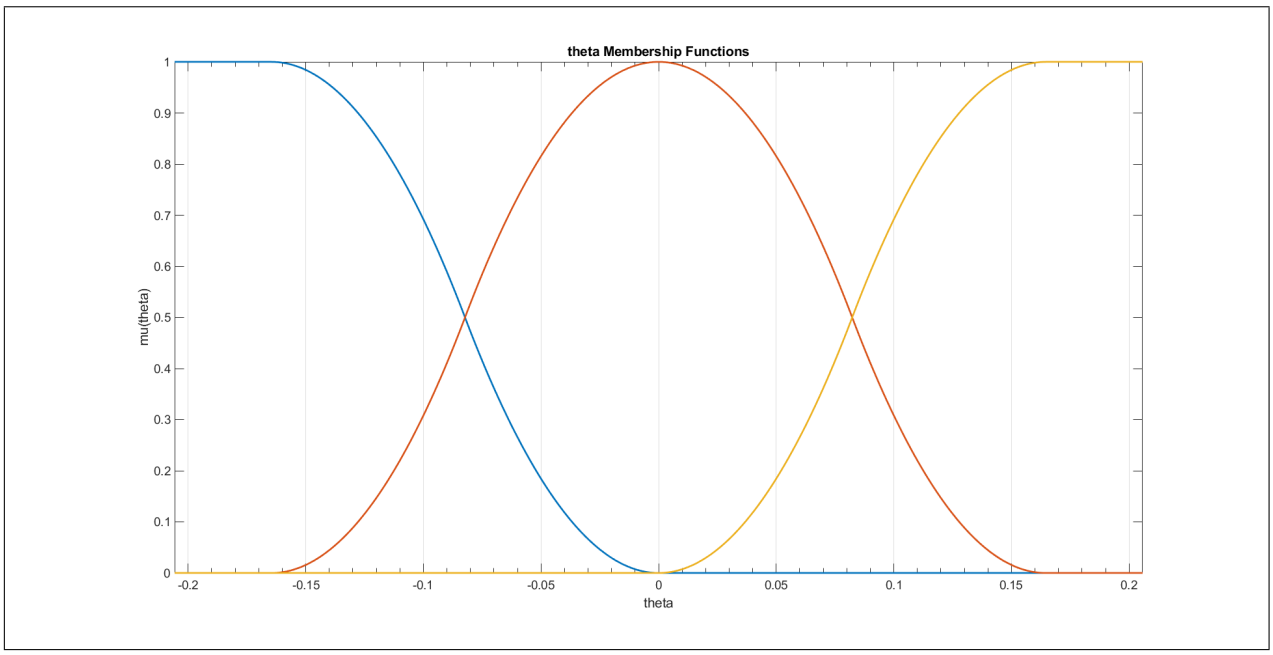

Рис. 7. Функции принадлежности для $\theta(t)$

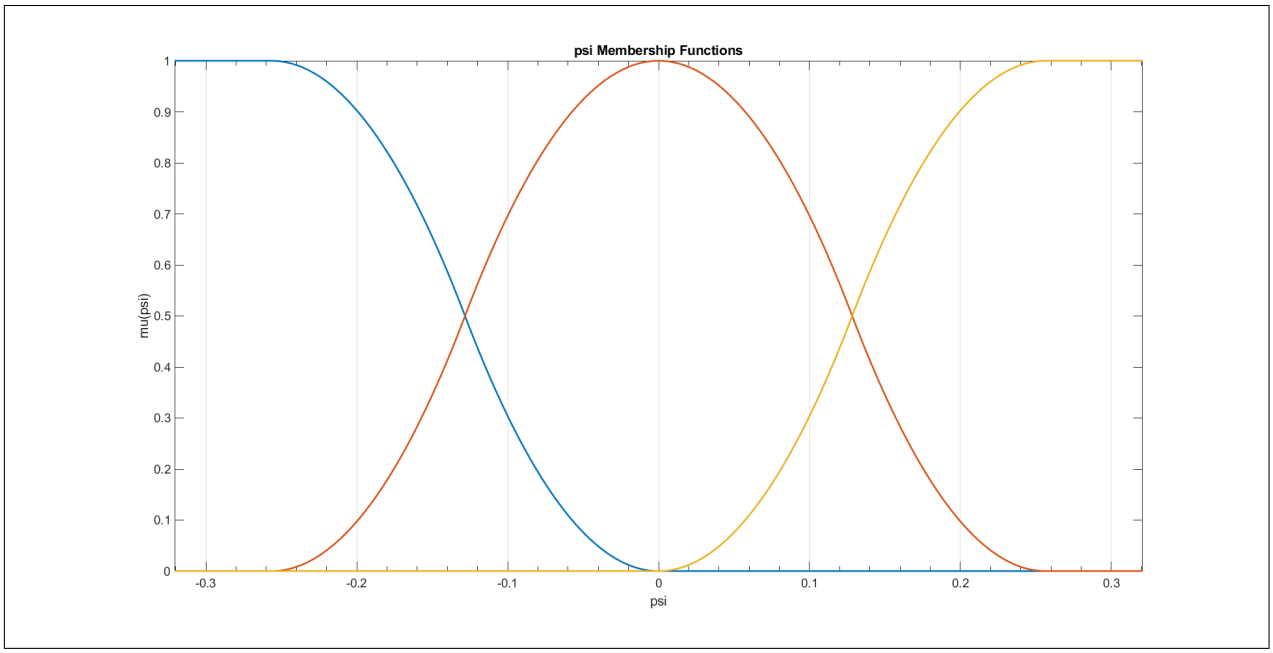

Рис. 8. Функции принадлежности для $\psi(t)$

Таким образом, общее число подсистем T–S FIS (27) (число VB линеаризаций) $r = \prod_{j=1}^{p} r_j = 972$.

Набор точек $(x_i^0,\ u_i^0)$, $i = 1, 2, ..., r$, расширенного пространства состояний $R^{n \times m}$, в окрестностях которых производится линеаризация, формируется с помощью функций из приложения 2. При этом для нахождения u_i^0 используется функция [9]. Линеаризация в узлах сформированной сетки реализуется функциями из приложения 3 с использованием функции

[27]. Основные элементы реализации механизма T–S FIS представлены в приложениях 4 — 7.

Некоторые результаты модельных экспериментов, посвящённых сравнению динамики построенной системы нечёткого вывода и исходной модели АНПА (33) — (35) приводятся на рис. 9 — 14:

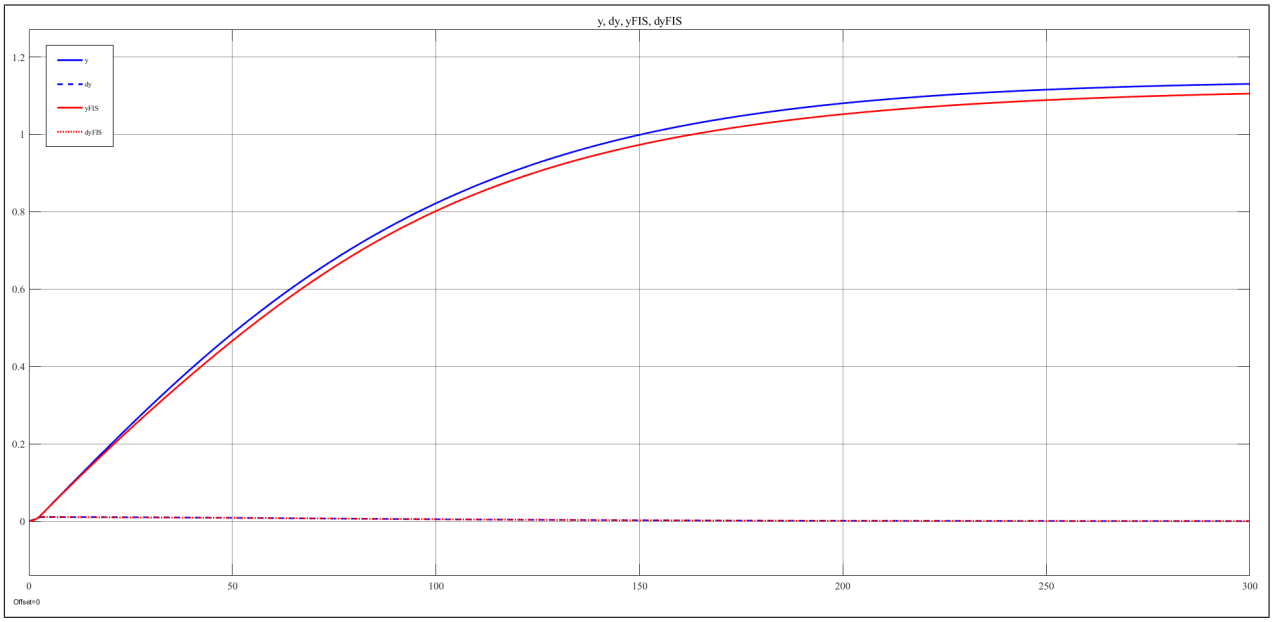

Рис. 9. Сравнение переходного процесса по $V_x(t)$

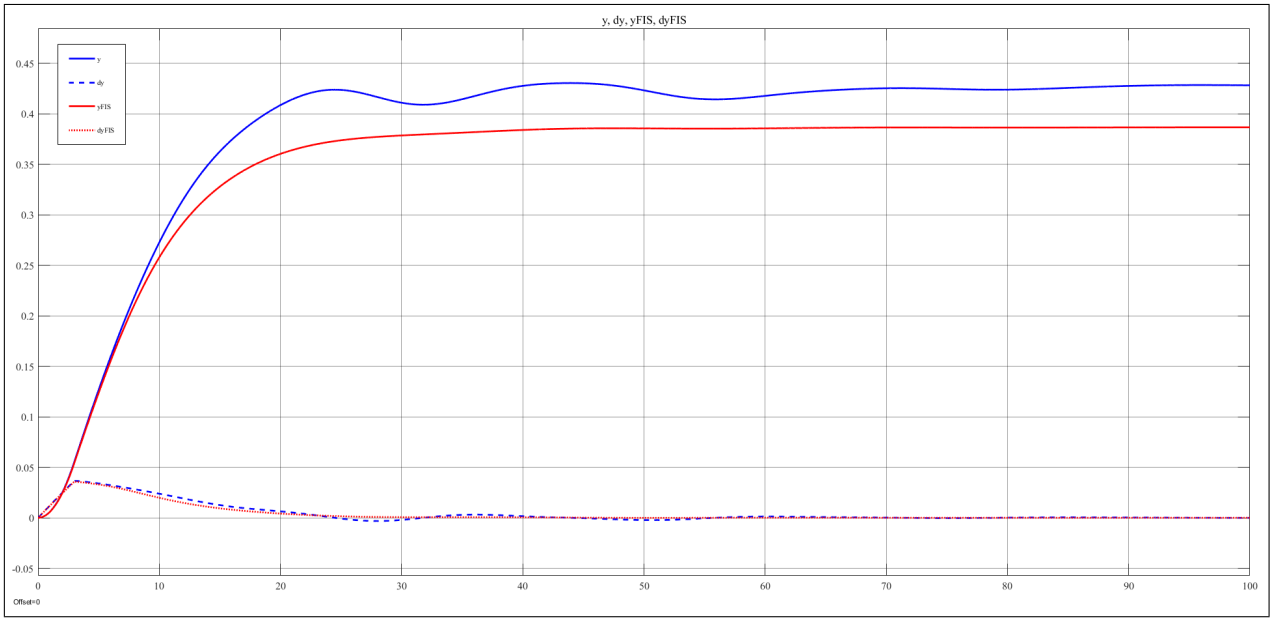

Рис. 10. Сравнение переходного процесса по $V_y(t)$

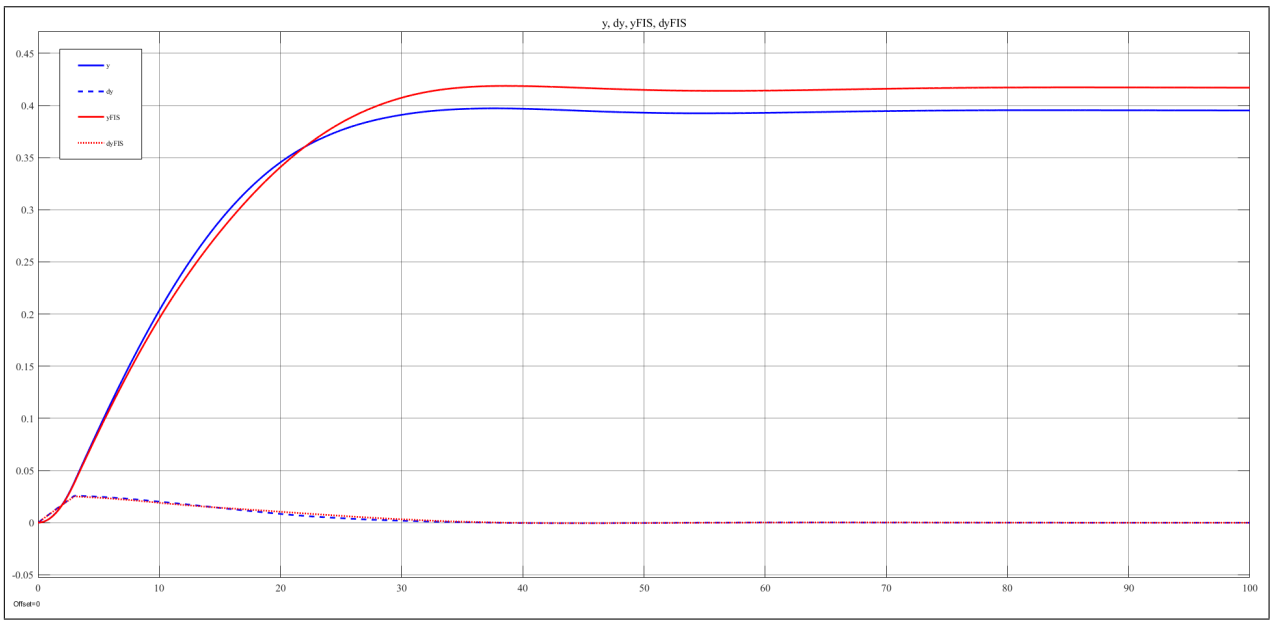

Рис. 11. Сравнение переходного процесса по $V_z(t)$

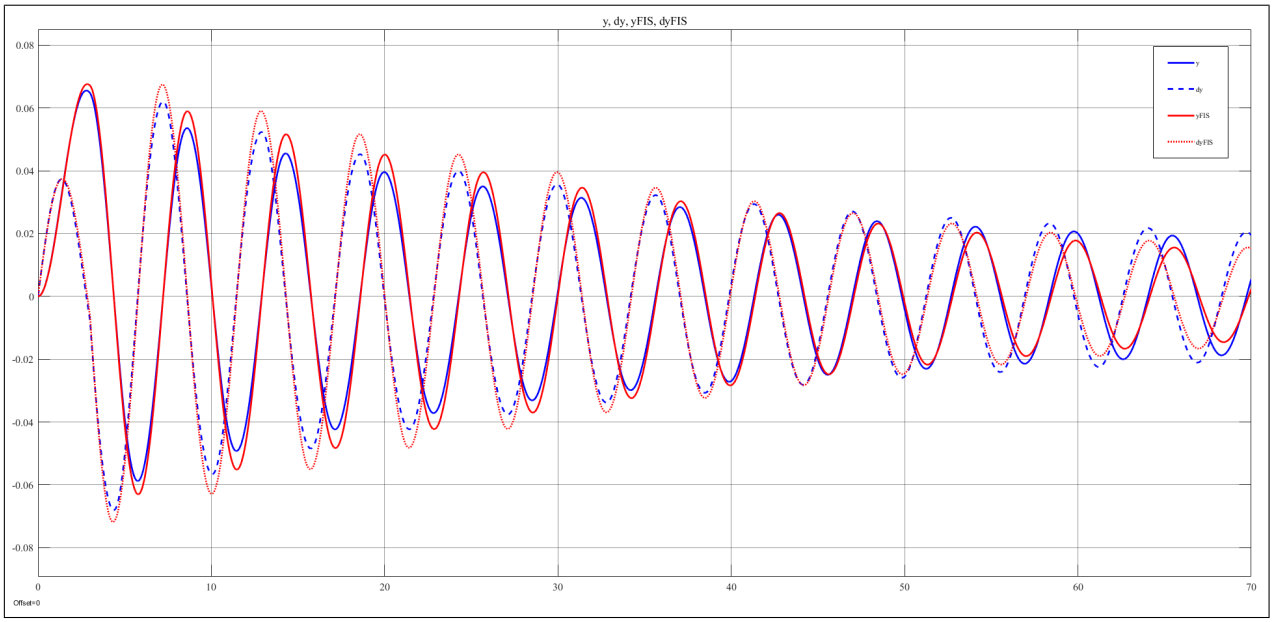

Рис. 12. Сравнение переходного процесса по $\omega_x(t)$

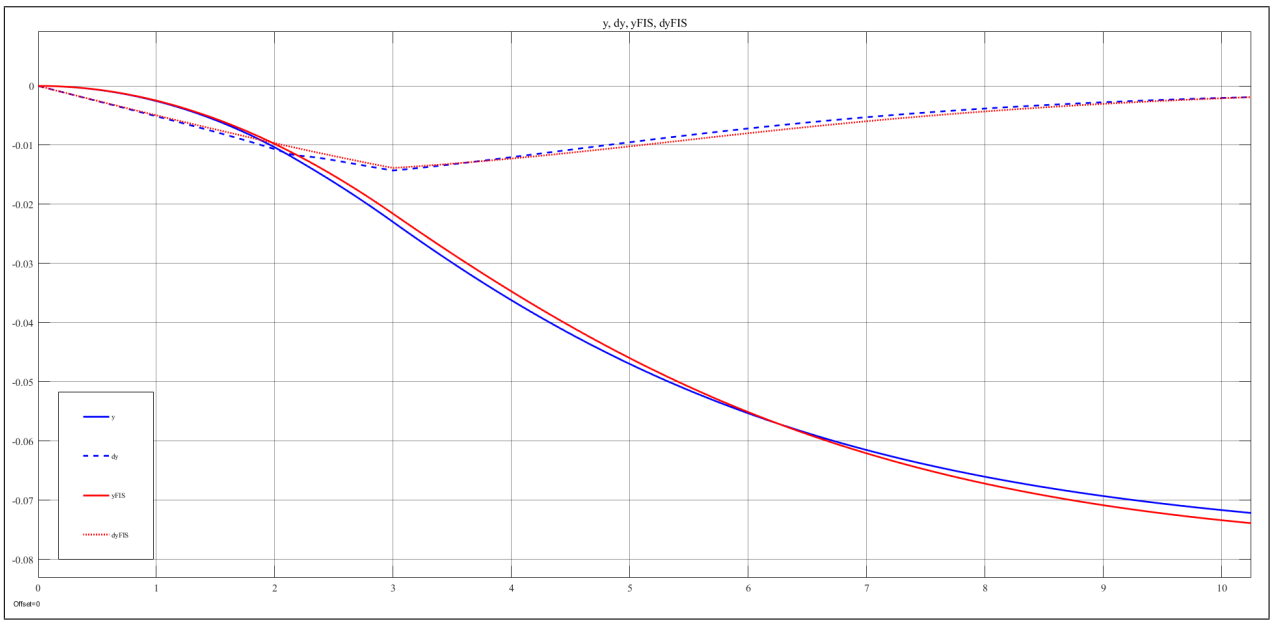

Рис. 13. Сравнение переходного процесса по $\omega_y(t)$

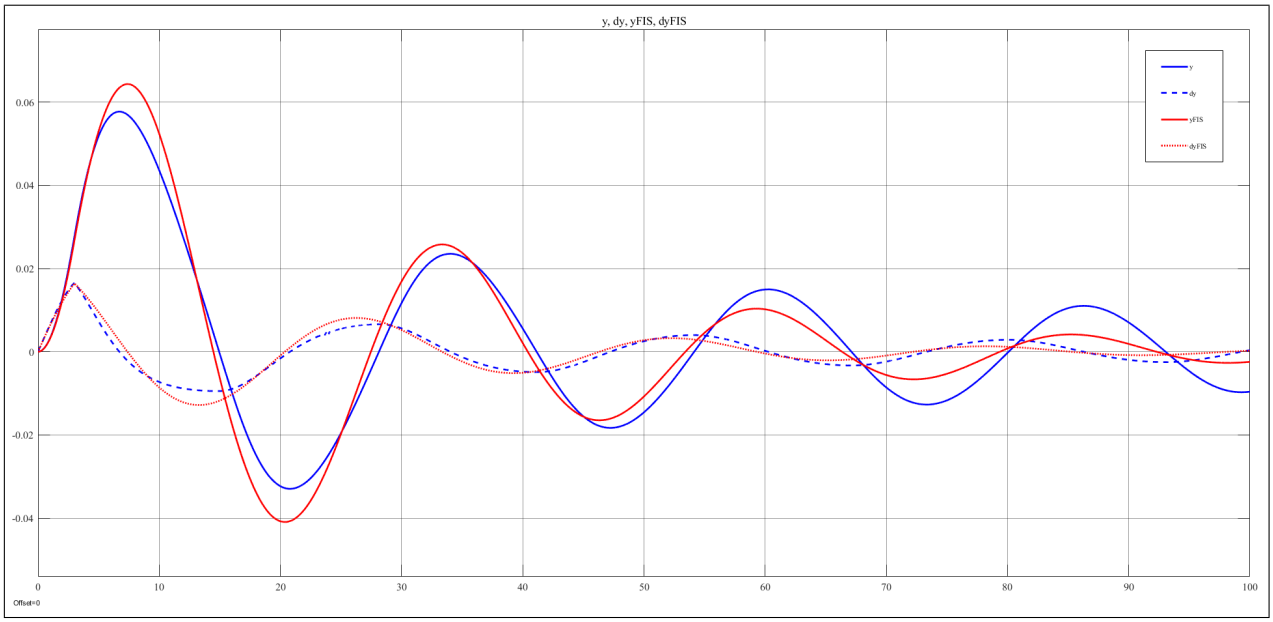

Рис. 14. Сравнение переходного процесса по $\omega_z(t)$

Можно заметить наличие статических ошибок при хорошем качестве аппроксимации переходных процессов. Подобное свойство V-B T–S FIS упоминалось в последнем параграфе главы 1 и было проанализировано в литературе [12, 29]. В целом, несмотря на несомненное наличие возможностей для улучшения, качество построенной модели достаточно высоко.

3.2. Синтез PDC для ДП АНПА

Синтез стабилизирующего PDC (31) на основе T–S FIS (29) — полностью формализованный процесс, заключающийся в решении системы LMI по одному из алгоритмов раздела 1.2.3 главы 1.

Один из вариантов реализации данной процедуры приводится в приложении 8 (используются результаты теоремы 2 и рассматривается случай различных матриц B_i). При этом используются свободно распространяемые toolbox-ы для MATLAB [34, 35]. Также возможен вариант с использованием функции [8]. Формирование управления на выходе PDC реализуется в функции из приложения 9.

Модель системы управления, созданная в Simulink для проведения имитационного компьютерного моделирования процессов управления, структурно идентична схеме, представленной на рис. 2.

PDC (31) гарантирует GAS системы нечёткого вывода (29). По результатам моделирования, представленным в предыдущем параграфе, указанная T–S FIS достаточно хорошо аппроксимирует динамику исходной модели (33) — (35), что с высокой долей уверенности позволяет предположить, что синтезированный регулятор обеспечивает устойчивость объекта управления.

Существуют подходы для осуществления строгого обоснования данного предположения (например, [21, 22]), однако их реализация достаточно сложна, а потому на данном этапе работы они не применяются. Альтернатива заключается в исследовании устойчивости замкнутой системы при помощи проведения разнообразных модельных экспериментов с различными начальными состояниями АНПА $\begin{pmatrix} x(0) & \delta(0) \end{pmatrix}^T$, командными сигналами $r(t)$, скоростями и направлениями подводного течения.

Переходные процессы по $x_p(t)$, получившиеся в результате одного из таких экспериментов, представлены на рис. 15 — 20. На них график синего цвета — желаемое значение регулируемой величины, красного цвета — текущее значение регулируемой величины, являющейся одной из компонент

выхода объекта управления. При этом

$$x_v(0) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^T, \quad x_p(0) = \begin{pmatrix} 0 & -50 & 0 & \frac{\pi}{12} & -\frac{\pi}{2} & \frac{\pi}{10} \end{pmatrix}^T,$$
$$\delta(0) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^T, \quad r(t) = \begin{pmatrix} 20 & -70 & 20 & 0 & 0 & 0 \end{pmatrix}^T,$$

модуль скорости течения равен 0,35 м/с, направление действия течения в неподвижной системе координат (относительно ориентации посадочной платформы) равно $\frac{\pi}{6}$ — один из граничных допустимых вариантов.

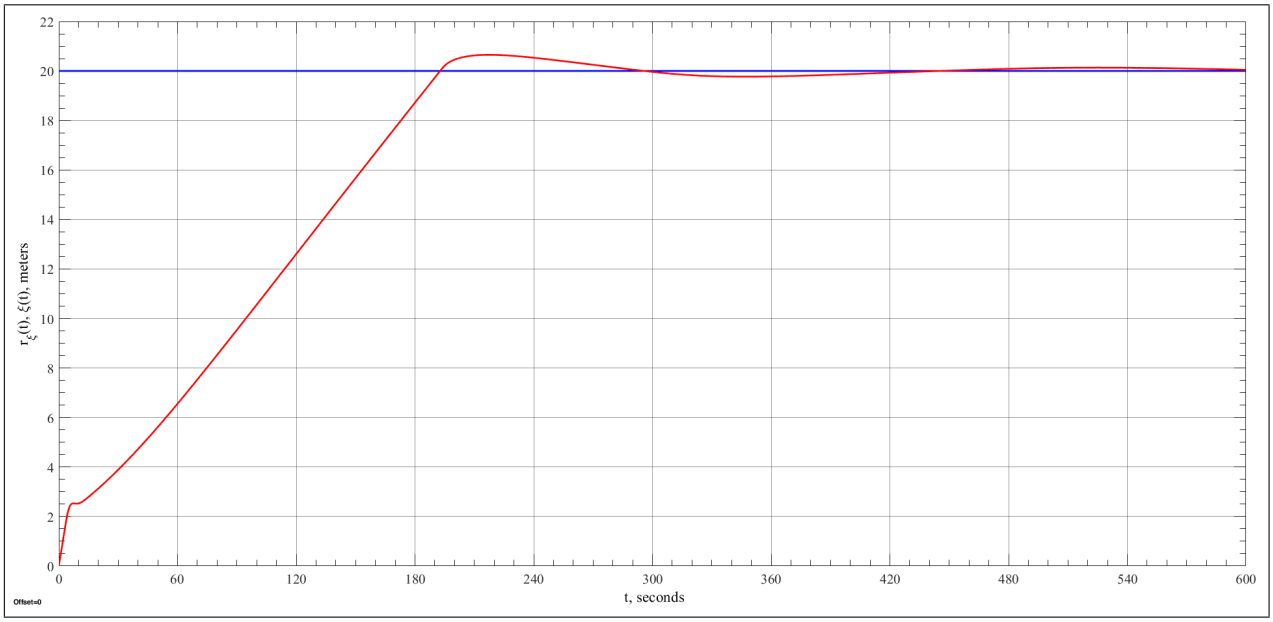

Рис. 15. Переходный процесс по $\xi(t)$

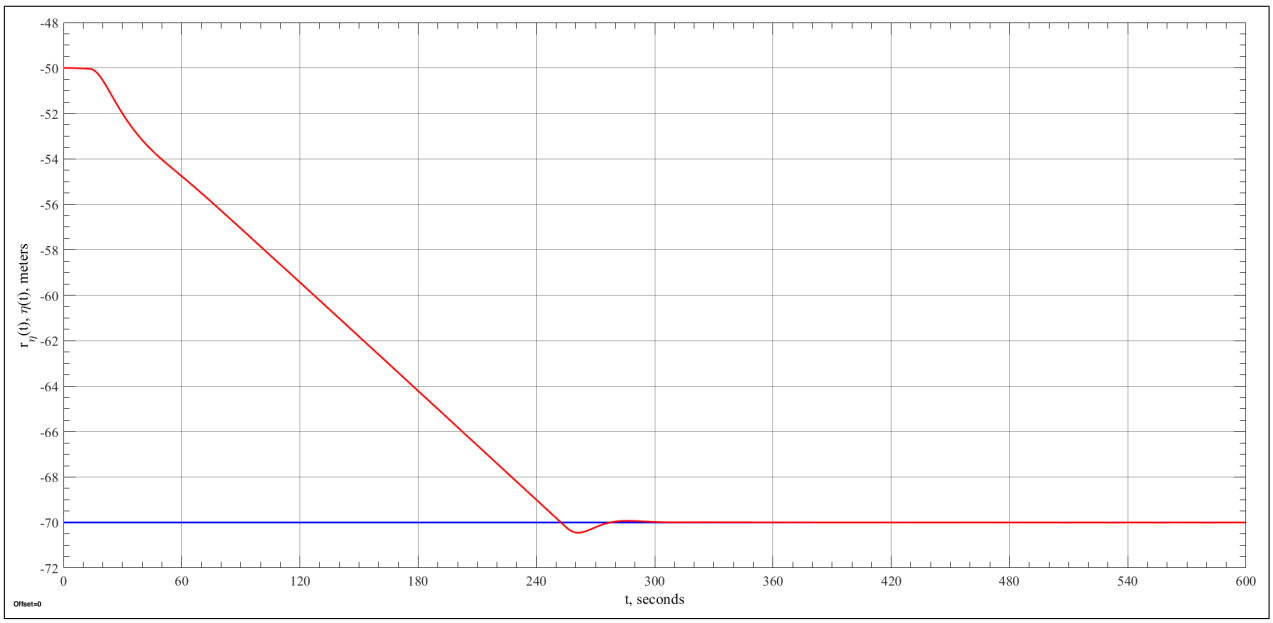

Рис. 16. Переходный процесс по $\eta(t)$

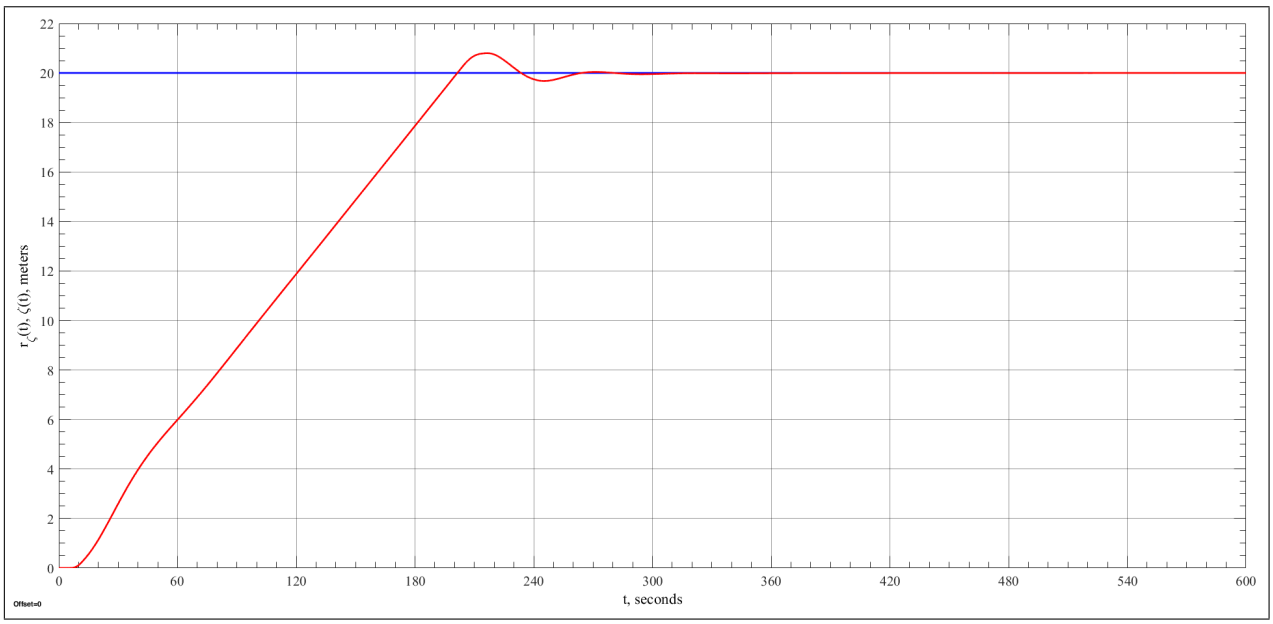

Рис. 17. Переходный процесс по $\zeta(t)$

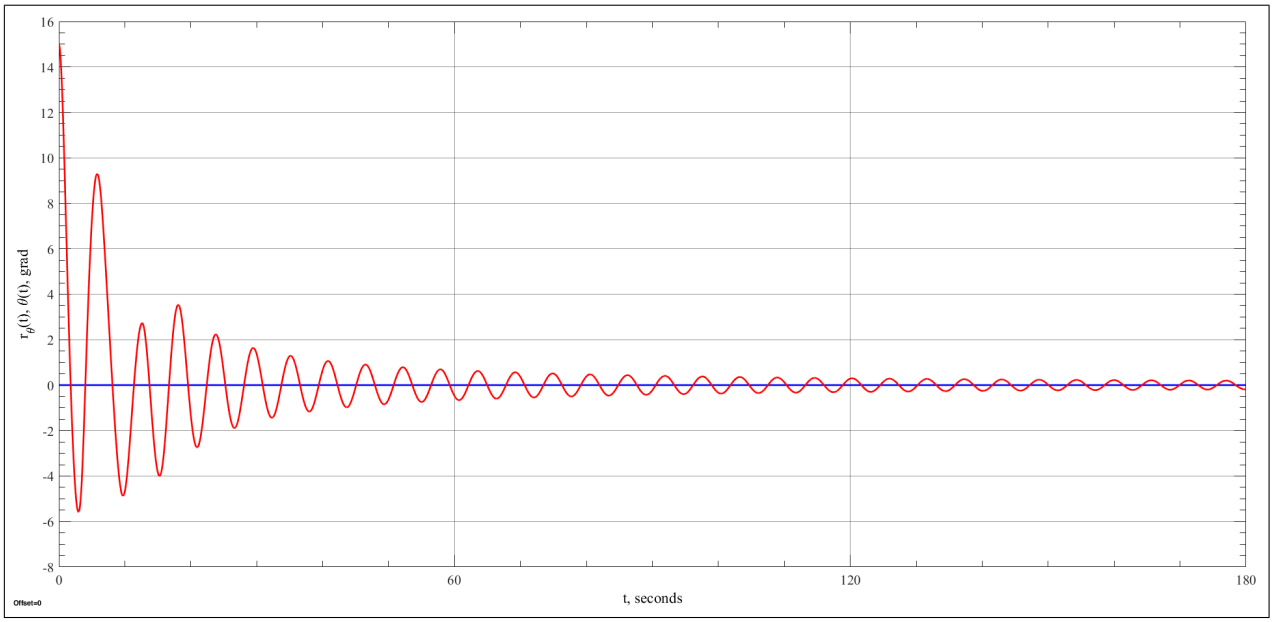

Рис. 18. Переходный процесс по $\theta(t)$

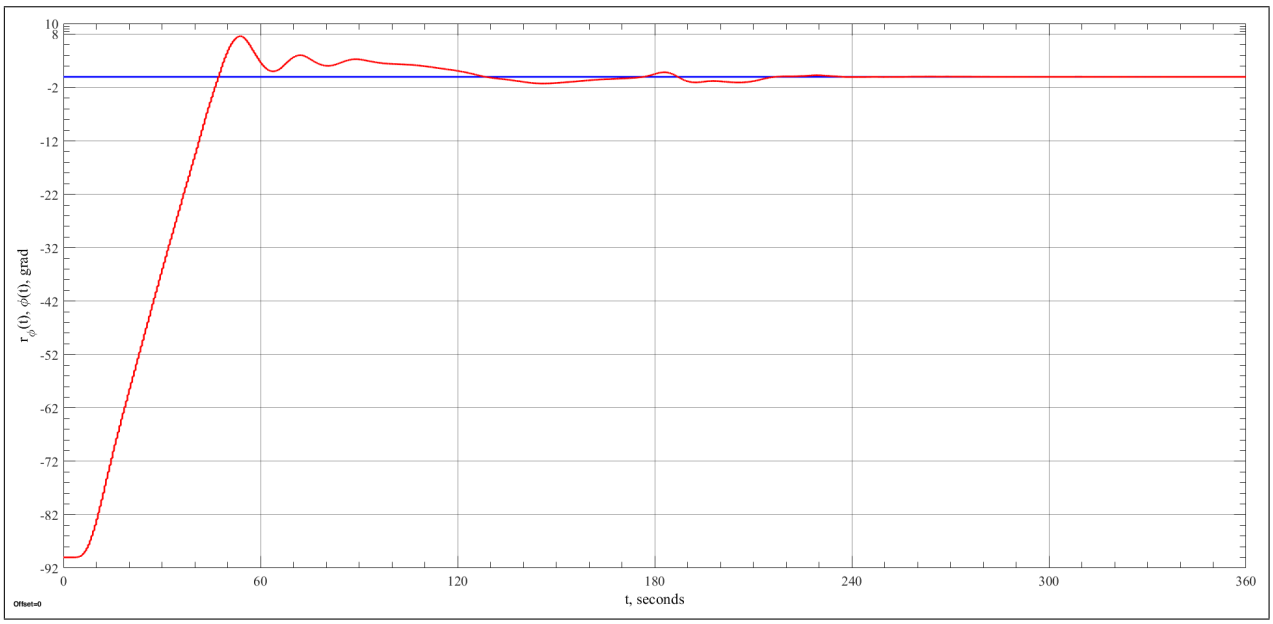

Рис. 19. Переходный процесс по $\varphi(t)$

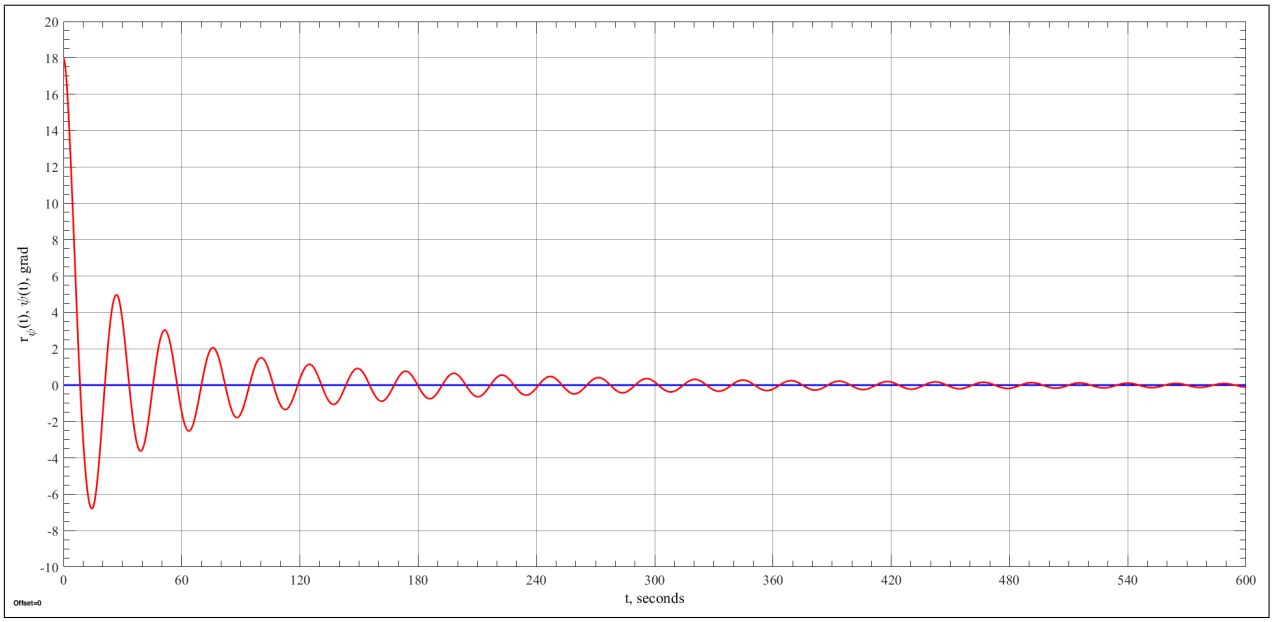

Рис. 20. Переходный процесс по $\psi(t)$

На приведённых графиках видно, что аппарат стабилизируется в окрестности желаемого положения.

Имитационное компьютерное моделирование при иных (реалистичных) начальных условиях и допустимых (в соответствии с имеющимися исходными данными) конфигурациях подводного течения демонстрирует аналогичный результат: разработанный алгоритм управления решает задачу ДП АНПА.

Выводы

Целью работы являлось создание системы автоматического управления, решающей задачу динамического позиционирования автономного необитаемого подводного аппарата.

В качестве исходных данных выступала нелинейная математическая модель движения АНПА. На её основе была построена аппроксимация, сочетающая в себе механизм системы нечёткого вывода типа Такаги–Сугено и подход к приближению нелинейной динамической системы с помощью семейства velocity-based линеаризаций.

Для устранения необходимости в численном дифференцировании при реализации закона управления в построенную T–S FIS была добавлена интегральная составляющая, затем на основе расширенной нечёткой модели был синтезирован стабилизирующий нечёткий закон управления (PDC). Процедура синтеза базировалась на использовании прямого метода Ляпунова, что в итоге свелось к задаче поиска допустимого решения системы линейных матричных неравенств. Затем синтезированный регулятор и построенная нечёткая модель были объединены в едином контуре управления.

В среде MATLAB & Simulink был разработан программный комплекс, в рамках которого были реализованы исходная математическая модель объекта управления, процесс построения системы нечёткого вывода и процедура синтеза стабилизирующего нечёткого закона управления.

Выполнено имитационное компьютерное моделирование процессов управления, в ходе которого была произведена проверка качества аппроксимации исходной математической модели системой нечёткого вывода, а также исследованы переходные процессы при ДП АНПА с использованием синтезированных алгоритмов управления. Результаты моделирования свидетельствуют об эффективности реализованного подхода.

В то же время данная работа требует дальнейшего развития по двум основным направлениям:

1. При практической разработке алгоритмов управления движением всегда существуют требования к эффективности и качеству динамики замкнутой системы, которые формализуются в виде различных опти-

мизационных задач. Таким образом, необходимо перейти от синтеза стабилизирующего нечёткого регулятора к синтезу ***оптимального*** стабилизирующего нечёткого регулятора [43].

2. В большинстве случаев скорости аппарата в ходе ДП недоступны в качестве обратных связей, что не позволяет использовать способ построения T–S FIS, описанный в данной работе. Или, если обобщить, далеко не всегда компоненты вектора состояния объекта управления, которые необходимо использовать в качестве параметров при формировании нечёткой модели и нечёткого закона управления, являются наблюдаемыми. Таким образом, необходимо модифицировать предлагаемый подход, ***оценивая*** (например, с помощью специальным образом синтезируемого наблюдателя [23]) не наблюдаемые параметры.

Заключение

В ходе выполнения данной работы получены следующие основные результаты, выносимые на защиту:

1. На основе исходной физической модели АНПА создана её аппроксимация в форме системы нечёткого вывода Такаги–Сугено с velocity-based подсистемами, которая используется для синтеза закона управления.
2. Синтезирован стабилизирующий закон управления, объединяющий PDC и T–S FIS в единую систему управления и решающий задачу динамического позиционирования АНПА.
3. Разработан программный комплекс, реализующий исходную математическую модель объекта управления, процесс построения T–S FIS и процедуру синтеза PDC. Произведено имитационное компьютерное моделирование процессов управления.

Список литературы

[1] Веремей Е. И. Компьютерное моделирование систем управления движением морских подвижных объектов / Веремей Е. И., Корчанов В. М., Коровкин М. В., Погожев С. В. СПб.: СПбГУ, 2002. 370 с.

[2] Веремей Е. И., Сотникова М. В. Многоцелевая структура законов управления морскими подвижными объектами // XII Всероссийское совещание по проблемам управления ВСПУ-2014. Москва, 16–19 июня 2014 г.: Труды. М.: Ин-т проблем управления им. В. А. Трапезникова РАН, 2014. С. 3289–3300.

[3] Baranyi P., Korondi P., Tanaka K. Parallel Distributed Compensation Based Stabilization of A 3-DOF RC Helicopter: A Tensor Product Transformation Based Approach // Journal of Advanced Computational Intelligence and Intelligent Informatics. 2009. Vol. 13, No. 1. P. 25–34.

[4] Cai G., Duan G., Hu Ch. A velocity-based LPV modeling and control framework for an air-breathing hypersonic vehicle // International Journal of Innovative Computing, Information and Control. 2011. Vol. 7, No. 5 (A). P. 2269–2281.

[5] Chang W. J., Chang W., Liu H. H. Model-Based Fuzzy Modeling and Control for Autonomous Underwater Vehicles in the Horizontal Plane // Journal of Marine Science and Technology. 2003. Vol. 11, No. 3. P. 155–163.

[6] Decay rate computation in LTI system // YALMIP URL: `https://yalmip.github.io/example/decayrate/`

[7] Espinosa J, Vandewalle J., Wertz V. Fuzzy Logic, Identification and Predictive Control. London: Springer-Verlag, 2005. 263 p. (Advances in Industrial Control: series).

[8] Feasp. Compute solution to given system of LMIs // MathWorks URL: `https://www.mathworks.com/help/robust/ref/feasp.html`

[9] Findop. Steady-state operating point from specifications (trimming) or simulation // MathWorks URL: https://www.mathworks.com/help/slcontrol/ug/findop.html

[10] Fossen T. I. Guidance and Control of Ocean Vehicles. Chichester, England: John Wiley & Sons, Ltd, 1999. 480 p.

[11] Fossen T. I. Handbook of Marine Craft Hydrodynamics and Motion Control. 1st ed. Chichester, United Kingdom: John Wiley & Sons, Ltd, 2011. 575 p.

[12] Gao R., O'Dwyer A., McLoone S., Coyle E. State feedback integral control by velocity-based multiple model networks // Proceedings of the American Control Conference. Boston, USA, 2004. P. 2039–2044.

[13] Gevp. Generalized eigenvalue minimization under LMI constraints // MathWorks URL: https://www.mathworks.com/help/robust/ref/gevp.html

[14] Ho W. H., Chen Sh. H., Chou J. H. Optimal control of Takagi-Sugeno fuzzy-model-based systems representing dynamic ship positioning systems // Applied Soft Computing. 2013. No. 13. P. 3197–3210.

[15] Korba P., Babuska R., Verbruggen H. B., Frank P. M. Fuzzy Gain Scheduling: Controller and Observer Design Based on Lyapunov Method and Convex Optimization // IEEE Transactions on Fuzzy Systems. 2003. Vol. 11, No. 3. P. 285–298.

[16] Leith D. J., Leithead W. E. Appropriate realisation of MIMO gain-scheduled controllers // International Journal of Control. 1998. Vol. 70, No. 1. P. 13–50.

[17] Leith D. J., Leithead W. E. Gain-scheduled and nonlinear systems: Dynamic analysis by velocity-based linearization families // International Journal of Control. 1998. Vol. 70, No. 2. P. 289–317.

[18] Leith D. J., Leithead W. E. Gain-scheduled controller design: An analytic framework directly incorporating non-equilibrium plant dynamics // International Journal of Control. 1998. Vol. 70, No. 2. P. 249–269.

[19] Leith D. J., Leithead W. E. Input-output linearization by velocity-based gain-scheduling // International Journal of Control. 1999. Vol. 72, No. 3. P. 229–246.

[20] Leith D. J., Leithead W. E. Survey of gain-scheduling analysis and design // International Journal of Control. 2000. Vol. 73, No. 11. P. 1001–1025.

[21] Lendek Zs., Babuska R., Schutter B. D. Stability bounds for fuzzy estimation and control. Part I: State estimation // 2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), Vol. 1. Cluj-Napoca, Romania, 2010. P. 1–6.

[22] Lendek Zs., Babuska R., Schutter B. D. Stability bounds for fuzzy estimation and control. Part II: Output-feedback control // 2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), Vol. 1. Cluj-Napoca, Romania, 2010. P. 1–6.

[23] Lendek Zs. Stability Analysis and Nonlinear Observer Design Using Takagi-Sugeno Fuzzy Models / Lendek Zs., Guerra T. M., Babuska R., Schutter B. D. Berlin, Heidelberg: Springer-Verlag, 2010. 196 p. (Studies in Fuzziness and Soft Computing: series / Kacprzyk J., editor-in-chief; vol. 262).

[24] Lhachemi H., Saussie D., Zhu G. An enhanced velocity-based algorithm for safe implementations of gain-scheduled controllers // International Journal of Control. 2017. Vol. 90, No. 9. P. 1973–1989.

[25] Lilly J. H. Fuzzy control and identification. Hoboken, New Jersey, USA: John Wiley & Sons, Inc., 2010. 231 p.

[26] Linear Matrix Inequalities in System and Control Theory / Boyd S., Ghaoui L. E., Feron E., Balakrishnan V. Philadelphia, Pennsylvania: SIAM, 1994. 193 p. (SIAM Studies in Applied Mathematics: series; vol. 15).

[27] Linearize. Linear approximation of Simulink model or subsystem // MathWorks URL: `https://www.mathworks.com/help/slcontrol/ug/linearize.html`

[28] Loria A., Fossen T. I., Panteley E. A. Separation Principle for Dynamic Positioning of Ships: Theoretical and Experimental Results // IEEE Transactions of Control Systems Technology. 2000. Vol. 8, No. 2. P. 332—343.

[29] McLoone S. C., Irwin G. W. On nonlinear modelling using velocity-based multiple model networks // American Control Conference. June, 2001. Arlington, USA, 2001. P. 25–27.

[30] Ngongi W. E., Du J., Wang R. Robust Fuzzy Controller Design for Dynamic Positioning System of Ships // International Journal of Control, Automation, and Systems. 2015. Vol. 13, No. 5. P. 1294–1305.

[31] Ohtake H., Tanaka K., Wang H. O. Switching Fuzzy Controller Design Based on Switching Lyapunov Function for a Class of Nonlinear Systems // IEEE Transactions on systems, man, and cybernetics. Part B: Cybernetics. 2006. Vol. 36, No. 1. P. 13–23.

[32] Perez T. Ship Motion Control: Course Keeping and Roll Stabilization using Rudder and Fins. London: Springer, 2005. 300 p.

[33] Ross T. J. Fuzzy Logic with Engineering Applications. 3rd ed. Chichester, United Kingdom: John Wiley & Sons, Ltd, 2010. 585 p.

[34] SEDUMI // SeDuMi. Optimization over symmetric cones URL: `http://sedumi.ie.lehigh.edu/`

[35] Semidefinite programming // YALMIP URL: `https://yalmip.github.io/tutorial/semidefiniteprogramming/`

[36] Siddique N., Adeli H. Computational intelligence : synergies of fuzzy logic, neural networks, and evolutionary computing. Chichester, United Kingdom: John Wiley & Sons, Ltd, 2013. 512 p.

[37] Silvestre C., Pascoal A., Kaminer I. On the design of gain-scheduled trajectory tracking controllers // International Journal of Robust and Nonlinear Control. 2002. No. 12. P. 797–839.

[38] Sugeno M., Kang G. T. Structure Identification of Fuzzy Model // Fuzzy Sets Syst. 1986. Vol. 28. P. 329–346.

[39] Sugeno M. Fuzzy Control. Tokyo, Japan: Nikkan Kougyou Shinbunsha Publisher, 1988.

[40] Takagi T., Sugeno M. Fuzzy Identification of Systems and Its Applications to Modeling and Control // IEEE Trans. Syst. Man. Cyber. 1985. Vol. 15. P. 116–132.

[41] Tanaka K., Sugeno M. Stability Analysis of Fuzzy Systems Using Lyapunov's Direct Method // NAFIPS'90. 1990. P. 133–136.

[42] Tanaka K., Sugeno M. Stability Analysis and Design of Fuzzy Control Systems // Fuzzy Sets Syst. 1992. Vol. 45, No. 2, P. 135–156.

[43] Tanaka K., Wang H. Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach. New York, USA: John Wiley & Sons, Inc., 2001. 305 p.

[44] Tanaka K., Hori Ts., Wang H. O. A Multiple Lyapunov Function Approach to Stabilization of Fuzzy Control Systems // IEEE Transactions on Fuzzy Systems. 2003. Vol. 11, No. 4. P. 582–589.

[45] Tanaka K., Ohtake H., Wang H. O. A Practical Design Approach to Stabilization of a 3-DOF RC Helicopter // IEEE Transactions on Control Systems Technology. 2004. Vol. 12, No. 2. P. 315–325.

[46] Tanaka K., Ohtake H., Wang H. A Descriptor System Approach to Fuzzy Control System Design via Fuzzy Lyapunov Functions // IEEE Transactions on Fuzzy Systems. 2007. Vol. 15, No. 3. P. 333–341.

[47] Veremey E. I. Dynamical Correction of Positioning Control Laws // 9th IFAC Conference on Control Applications in Marine Systems. The International Federation of Automatic Control. September 17–20, 2013. Osaka, Japan, 2013. P. 31–36.

[48] Wang H. O., Tanaka K., Griffin M. F. Parallel Distributed Compensation of Nonlinear Systems by Takagi-Sugeno Fuzzy Model // IEEE International Conference on Fuzzy Systems. March 20–24, 1995. Yokohama, Japan, 1995. P. 531–538.

[49] Wang H. O., Tanaka K., Griffin M. F. An Analytical Framework of Fuzzy Modeling and Control of Nonlinear Systems: Stability and Design Issues // American Control Conference. June 21–23, 1995. Seattle, USA, 1995. P. 2272–2276.

[50] Xia G., Xue J., Jiao J. Dynamic Positioning Control System with Input Time-Delay Using Fuzzy Approximation Approach // International Journal of Fuzzy Systems. 2018. Vol. 20, No. 2. P. 630–639.

[51] Xiang X., Yu C., Lapierre L., Zhang J., Zhang Q. Survey on Fuzzy-Logic-Based Guidance and Control of Marine Surface Vehicles and Underwater Vehicles // International Journal of Fuzzy Systems. 2018. Vol. 20, No. 2. P. 572–586.

[52] Zhang H., Liu D. Fuzzy Modeling and Fuzzy Control. Boston: Birkhauser, 2006. 416 p. (Control Engineering: series / Levine W. S., editor).

Приложения

В приложениях приведены некоторые элементы разработанного в ходе данной работы программного комплекса.

Приложение 1. Формирование значений z_{jk}^0

```
function sgrid = schedulingGrid()

names = ["Vx"; "Vy"; "Vz"; "wx"; "wy"; "wz"; "theta"; "phi"; "psi"];
mult  = [1; 0.9*ones(6, 1); 1; 0.5];
num   = [4; 3; 3; 1; 3; 1; 3; 1; 3];

oprange = cell(size(names));
oprange{1} = [- 1.0;                 1.0               ];          % Vx
oprange{2} = [- 0.420587994122353;   0.415695760785741];          % Vy
oprange{3} = [- 0.394311280947313;   0.384313194744852];          % Vz
oprange{4} = [- 5.735800265037361;   5.734607807671845].*pi./180; % wx
oprange{5} = [- 6.318897114765901;   6.318897114765901].*pi./180; % wy
oprange{6} = [- 3.466915106537084;   3.489978563292468].*pi./180; % wz
oprange{7} = [-10.478253691216160; 10.476216140134220].*pi./180; % theta
oprange{8} = [- 0.8*180;             0.8*180          ].*pi./180; % phi
oprange{9} = [-29.070610150678352; 29.400996789990781].*pi./180; % psi

sgrid = cell(size(oprange));
scale = [0.75*ones(7, 1); 1; 0.75];

for i = 1:length(sgrid)

    oprange{i} = [-max(abs(oprange{i})); max(abs(oprange{i}))];
    oprange{i} = oprange{i}.*mult(i);
    sgrid  {i} = struct;

    sgrid{i}.Name = names(i);

    if num(i) > 1

      sgrid{i}.Value = linspaceMod(linspace(oprange{i}(1), ...
        oprange{i}(2), num(i)), scale(i));

    else

      sgrid{i}.Value = 0;

    end
```

```
end
end

function space = linspaceMod(space, scale)

n = length(space);
k = 1;

while k <= floor(n/2)

    space = [space(1:k), linspace(space(k + 1)*scale, ...
      space(n - k)*scale, n - k*2), space(n - k + 1:n)];
    k = k + 1;

end
end
```

Приложение 2. Формирование сетки в $R^{n\times m}$

```
function [op, report] = OPdesignFamily(mdl, sGrid)

options                                    = findopOptions;
options.OptimizerType                      = 'graddescent-proj';
options.ConstraintType.dx                  = 'hard';
options.ConstraintType.x                   = 'hard';
options.ConstraintType.y                   = 'hard';
options.DisplayReport                      = 'off';
options.OptimizationOptions.Algorithm      = 'interior-point';
options.OptimizationOptions.MaxFunEvals = 3e4;
options.OptimizationOptions.MaxIter        = 1e4;
options.OptimizationOptions.UseParallel = false;

VxGrid    = sGrid{1}.Value;
VyGrid    = sGrid{2}.Value;
VzGrid    = sGrid{3}.Value;
wxGrid    = sGrid{4}.Value;
wyGrid    = sGrid{5}.Value;
wzGrid    = sGrid{6}.Value;
thetaGrid = sGrid{7}.Value;
phiGrid   = sGrid{8}.Value;
psiGrid   = sGrid{9}.Value;

[Vx, Vy, Vz, wx, wy, wz, theta, phi, psi] = ...
  ndgrid(VxGrid, VyGrid, VzGrid, wxGrid, wyGrid, wzGrid, ...
    thetaGrid, phiGrid, psiGrid);

```

```
shape = [];
for i = 1:length(sGrid)

    shape = [shape, length(sGrid{i}.Value)];

end

opNo        = prod(shape);
maxIterOpNo = 840;
if opNo > maxIterOpNo

    op      = [];
    report  = [];
    opLeft  = opNo;
    opDone  = 0;

    if mod(opNo, maxIterOpNo) == 0, iterNo = opNo/maxIterOpNo;
    else,                           iterNo = floor(opNo/maxIterOpNo) + 1;
    end

    disp(['Number of iterations: ', num2str(iterNo)]);

    for i = 1:iterNo

        disp(['Current iteration: ', num2str(i)]);

        opNoCurr  = min(maxIterOpNo, opLeft);
        rangeCurr = opDone + 1 : opDone + opNoCurr;

        VxCurr    = Vx   (rangeCurr);
        VyCurr    = Vy   (rangeCurr);
        VzCurr    = Vz   (rangeCurr);
        wxCurr    = wx   (rangeCurr);
        wyCurr    = wy   (rangeCurr);
        wzCurr    = wz   (rangeCurr);
        thetaCurr = theta(rangeCurr);
        phiCurr   = phi  (rangeCurr);
        psiCurr   = psi  (rangeCurr);

        [opCurr, reportCurr] = ...
          parallelTrimming(mdl, opNoCurr, VxCurr, VyCurr, VzCurr, ...
            wxCurr, wyCurr, wzCurr, thetaCurr, phiCurr, psiCurr, options);

        op     = [op;         opCurr];
        report = [report; reportCurr];
        opLeft = opLeft - opNoCurr;
```

```
74          opDone = opDone + opNoCurr;
75
76      end
77  else
78
79      disp(['Number of iterations: ', num2str(1)]);
80
81      [op, report] = ...
82        parallelTrimming(mdl, opNo, Vx, Vy, Vz, wx, wy, wz, ...
83          theta, phi, psi, options);
84
85  end
86
87  op     = reshape(op,     shape);
88  report = reshape(report, shape);
89
90  end
91
92  function [opArr, reportArr] = ...
93    parallelTrimming(mdl, opNo, Vx, Vy, Vz, wx, wy, wz, ...
94      theta, phi, psi, options)
95
96  workersNo = 6;
97  iterOpNo  = [];
98
99  VxIter    = cell(workersNo, 1);
100 VyIter    = cell(workersNo, 1);
101 VzIter    = cell(workersNo, 1);
102 wxIter    = cell(workersNo, 1);
103 wyIter    = cell(workersNo, 1);
104 wzIter    = cell(workersNo, 1);
105 thetaIter = cell(workersNo, 1);
106 phiIter   = cell(workersNo, 1);
107 psiIter   = cell(workersNo, 1);
108
109 if mod(opNo, workersNo) == 0, maxIterOpNo = opNo/workersNo;
110 else,                         maxIterOpNo = floor(opNo/workersNo) + 1;
111 end
112
113 if maxIterOpNo > 1 && maxIterOpNo*(workersNo - 1) < opNo
114
115     opLeft = opNo;
116
117     for i = 1:workersNo
118
119         iterOpNo = [iterOpNo, min(maxIterOpNo, opLeft)];
```

```
120
121         if i > 1, iterRange = sum(iterOpNo(1:end - 1)) + ...
122           1:sum(iterOpNo(1:end));
123         else, iterRange = 1 : iterOpNo(end);
124         end
125
126         VxIter   {i} = Vx   (iterRange);
127         VyIter   {i} = Vy   (iterRange);
128         VzIter   {i} = Vz   (iterRange);
129         wxIter   {i} = wx   (iterRange);
130         wyIter   {i} = wy   (iterRange);
131         wzIter   {i} = wz   (iterRange);
132         thetaIter{i} = theta(iterRange);
133         phiIter  {i} = phi  (iterRange);
134         psiIter  {i} = psi  (iterRange);
135
136         opLeft = opLeft - iterOpNo(end);
137
138     end
139 else
140
141     workersNo = 1;
142     iterOpNo  = opNo;
143
144     VxIter   {1} = Vx;
145     VyIter   {1} = Vy;
146     VzIter   {1} = Vz;
147     wxIter   {1} = wx;
148     wyIter   {1} = wy;
149     wzIter   {1} = wz;
150     thetaIter{1} = theta;
151     phiIter  {1} = phi;
152     psiIter  {1} = psi;
153
154 end
155
156 op     = cell(workersNo, 1);
157 report = cell(workersNo, 1);
158
159 dtheta = @(wx, wy, wz, theta, psi) ...
160   wx - (wy*cos(theta) - wz*sin(theta))/cos(psi)*sin(psi);
161 dphi   = @(wx, wy, wz, theta, psi) ...
162   (wy*cos(theta) - wz*sin(theta))/cos(psi);
163 dpsi   = @(wx, wy, wz, theta, psi) wy*sin(theta) + wz*cos(theta);
164
165 parfor j = 1:workersNo
```

```
166
167     mdlCurr = [mdl, num2str(j)];
168     opspec  = operspec(mdlCurr, iterOpNo(j));
169
170     fixedStatesInd   = [1:6, 10:12];
171     freeStatesInd    = 7:9;        % [xi; eta; zeta]
172     thetaInd         = 10;
173     phiInd           = 11;
174     psiInd           = 12;
175     activeContrInd   = [1, 5:10]; % [Tn; TnT]
176     inactiveContrInd = 2:4;        % [deltaT; deltaL; deltaR]
177     contrBoundsU     = [ 9.456e3;  30*pi/180*ones(3, 1);  400*ones(6, 1)];
178     contrBoundsL     = [-4.864e3; -30*pi/180*ones(3, 1); -400*ones(6, 1)];
179
180     for i = 1:iterOpNo(j)
181
182         VxCurr    = VxIter   {j}(i);
183         VyCurr    = VyIter   {j}(i);
184         VzCurr    = VzIter   {j}(i);
185         wxCurr    = wxIter   {j}(i);
186         wyCurr    = wyIter   {j}(i);
187         wzCurr    = wzIter   {j}(i);
188         thetaCurr = thetaIter{j}(i);
189         phiCurr   = phiIter  {j}(i);
190         psiCurr   = psiIter  {j}(i);
191
192         zCurr = [VxCurr; VyCurr; VzCurr; wxCurr; wyCurr; wzCurr; ...
193           thetaCurr; phiCurr; psiCurr];
194
195         opspec(i).Outputs(1).Block      = [mdlCurr, '/', 'y'];
196         opspec(i).Outputs(1).y    (end) = zCurr(end);
197         opspec(i).Outputs(1).Known(end) = true;
198
199         opspec(i).States(1).Block = [mdlCurr, '/', mdl, '/', 'X'];
200         opspec(i).States(1).x     (fixedStatesInd) = zCurr;
201         opspec(i).States(1).Known(fixedStatesInd) = ...
202           true (length(fixedStatesInd), 1);
203         opspec(i).States(1).SteadyState(fixedStatesInd) = ...
204           true (length(fixedStatesInd), 1);
205         opspec(i).States(1).SteadyState(freeStatesInd)  = ...
206           false(length(freeStatesInd),  1);
207
208         if wxCurr ~= 0 || ...
209           dtheta(wxCurr, wyCurr, wzCurr, thetaCurr, psiCurr) ~= 0, ...
210             opspec(i).States(1).SteadyState(thetaInd) = false; end
211         if wyCurr ~= 0 || ...
```

```
212           dphi(wxCurr, wyCurr, wzCurr, thetaCurr, psiCurr) ~= 0, ...
213             opspec(i).States(1).SteadyState(phiInd)   = false; end
214         if wzCurr ~= 0 || ...
215           dpsi(wxCurr, wyCurr, wzCurr, thetaCurr, psiCurr) ~= 0, ...
216             opspec(i).States(1).SteadyState(psiInd)   = false; end
217
218         opspec(i).Inputs(1).Block = [mdlCurr, '/', 'u'];
219         opspec(i).Inputs(1).u(inactiveContrInd) = ...
220           zeros(length(inactiveContrInd), 1);
221         opspec(i).Inputs(1).Known(inactiveContrInd) = ...
222           true (length(inactiveContrInd), 1);
223         opspec(i).Inputs(1).Min(activeContrInd) = ...
224           contrBoundsL(activeContrInd);
225         opspec(i).Inputs(1).Max(activeContrInd) = ...
226           contrBoundsU(activeContrInd);
227
228         opspec(i).Inputs(2).Block = [mdlCurr, '/', 'd'];
229         opspec(i).Inputs(2).u = zeros(length(opspec(i).Inputs(2).u), 1);
230         opspec(i).Inputs(2).Known = ...
231           true (length(opspec(i).Inputs(2).u), 1);
232
233     end
234
235     [op{j}, report{j}] = findop(mdlCurr, opspec, options);
236     disp(['Worker ', num2str(j), ' have finished']);
237
238 end
239
240 delete(gcp);
241
242 opArr     = [];
243 reportArr = [];
244
245 for i = 1:workersNo
246
247     opArr     = [opArr;          op{i}];
248     reportArr = [reportArr; report{i}];
249
250 end
251 end
```

Приложение 3. Линеаризация в узлах сформированной сетки

```
1 function [linFam, offsets, shape] = LTIdesignFamily( ...
```

```
  mdl, op, sGrid, useRegular)

shape = properShape(op);
grid  = samplingGrid(op, sGrid, shape, useRegular);

for i = 1:numel(op)

    linOp(i) = operpoint(mdl);

    if ~useRegular

        linOp(i).States(1).x = op(i).States(1).x;

    else

        linOp(i).States(1).x([1:3, 5, 10, 12]) = ...
          [grid.Vx(i); grid.Vy(i); grid.Vz(i); grid.wy(i); ...
            grid.theta(i); grid.psi(i)];
        linOp(i).States(1).x([4, 6, 11]) = zeros(3, 1); % [wx; wz; phi]

    end

    linOp(i).States(1).x(7:9) = [0; -50; 0]; % [xi; eta; zeta]
    linOp(i).Inputs(1).u = op(i).Inputs(1).u;
    linOp(i).Inputs(2).u = op(i).Inputs(2).u;

end

options                          = linearizeOptions;
options.LinearizationAlgorithm = 'blockbyblock';
options.SampleTime               = 0;
options.IgnoreDiscreteStates   = 'on';
options.StoreOffsets             = true;
options.StoreAdvisor             = false;

block = [mdl, '/', mdl];
[linFam, ~, linInfo] = linearize(mdl, block, linOp, options);
offsets = getOffsetsForLPV(linInfo);

linFam     = reshape(linFam,     shape);
offsets.x  = reshape(offsets.x,  [size(linFam.a, 1), 1, shape]);
offsets.dx = reshape(offsets.dx, [size(linFam.a, 1), 1, shape]);
offsets.u  = reshape(offsets.u,  [size(linFam.b, 2), 1, shape]);
offsets.y  = reshape(offsets.y,  [size(linFam.c, 1), 1, shape]);

linFam.u = {'Tn', 'deltaT', 'deltaL', 'deltaR', 'TnFVL', 'TnFVR', ...
```

```
  'TnBVL', 'TnBVR', 'TnFH', 'TnBH', 'Vsxi', 'Vseta', 'Vszeta'};
linFam.y = {'xi', 'eta', 'zeta', 'theta', 'phi', 'psi'};
linFam.SamplingGrid = grid;

end

function grid = samplingGrid(op, sGrid, shape, useRegular)

if ~useRegular
    for i = 1:numel(op)

        Vx    (i) = op(i).States(1).x(1);
        Vy    (i) = op(i).States(1).x(2);
        Vz    (i) = op(i).States(1).x(3);
        wx    (i) = op(i).States(1).x(4);
        wy    (i) = op(i).States(1).x(5);
        wz    (i) = op(i).States(1).x(6);
        theta(i) = op(i).States(1).x(10);
        phi   (i) = op(i).States(1).x(11);
        psi   (i) = op(i).States(1).x(12);

    end
else

    VxGrid     = sGrid{1}.Value;
    VyGrid     = sGrid{2}.Value;
    VzGrid     = sGrid{3}.Value;
    wxGrid     = sGrid{4}.Value;
    wyGrid     = sGrid{5}.Value;
    wzGrid     = sGrid{6}.Value;
    thetaGrid  = sGrid{7}.Value;
    phiGrid    = sGrid{8}.Value;
    psiGrid    = sGrid{9}.Value;

    [Vx, Vy, Vz, wx, wy, wz, theta, phi, psi] = ...
      ndgrid(VxGrid, VyGrid, VzGrid, wxGrid, wyGrid, wzGrid, ...
        thetaGrid, phiGrid, psiGrid);

end

Vx    = reshape(Vx,    shape);
Vy    = reshape(Vy,    shape);
Vz    = reshape(Vz,    shape);
wx    = reshape(wx,    shape);
wy    = reshape(wy,    shape);
wz    = reshape(wz,    shape);
```

```
theta = reshape(theta, shape);
phi   = reshape(phi,   shape);
psi   = reshape(psi,   shape);

grid = struct;

for i = 1:length(sGrid)
    if length(sGrid{i}.Value) > 1

        grid.(sGrid{i}.Name) = eval(char(sGrid{i}.Name));

    end
end
end

function shape = properShape(refArr)

shape = [];
s     = size(refArr);

for i = 1:length(s)
    if s(i) > 1

        shape = [shape, s(i)];

    end
end

if     isempty(shape),     shape = [1, 1];
elseif length(shape) == 1, shape = [shape, 1];
end
end
```

Приложение 4. Функции принадлежности

```
function mu = triangularMF(z, a, b, c, placement)

if                              z == b, mu = 1;
elseif placement == int32(-1) && z  < b, mu = 1; % left
elseif placement == int32( 1) && z  > b, mu = 1; % right
elseif                 z <= a || z >= c, mu = 0;
elseif                 z >= a && z <= b, mu = (z - a)/(b - a);
else,                                    mu = (c - z)/(c - b);
end
end

```

```
function mu = gaussianMF(z, sigma, c, placement)

if                                z == c, mu = 1;
elseif placement == int32(-1) && z  < c, mu = 1; % left
elseif placement == int32( 1) && z  > c, mu = 1; % right
else,                                    mu = exp(-(z - c)^2/(2*sigma^2));
end
end

function mu = piShapedMF(z, a, b, c, d, placement)

if                     z >= b && z <= c, mu = 1;
elseif placement == int32(-1) && z  < b, mu = 1; % left
elseif placement == int32( 1) && z  > c, mu = 1; % right
elseif                 z <= a || z >= d, mu = 0;
elseif z >= a && z <= (a + b)/2, mu =     2*((z - a)/(b - a))^2;
elseif z >= (a + b)/2 && z <= b, mu = 1 - 2*((z - b)/(b - a))^2;
elseif z >= c && z <= (c + d)/2, mu = 1 - 2*((z - c)/(d - c))^2;
else,                            mu =     2*((z - d)/(d - c))^2;
end
end
```

Приложение 5. Вычисление степеней принадлежности для $z_j(t)$

```
function [MU, activeMFs] = fuzzification(z, zNo, zGrid, zLen, zLenMax,
    parForm, mfType)

MU        = zeros(zNo, zLenMax);
activeMFs = zeros(zNo, 2);

if mfType == int32(0) % gaussian MF
    for i = 1:zNo
        for j = 1:zLen(i)

            if j == 1, MU(i, j) = gaussianMF(z(i), ...
                    parForm(i, j, 1), zGrid(i, j), int32(-1));
            elseif j == zLen(i), MU(i, j) = gaussianMF(z(i), ...
                    parForm(i, j, 1), zGrid(i, j), int32( 1));
            else, MU(i, j) = gaussianMF(z(i), ...
                    parForm(i, j, 1), zGrid(i, j), int32( 0));
            end
        end
    end
elseif mfType == int32(1) % pi-shaped MF
```

```
20      for i = 1:zNo
21
22          activeCount = 0;
23
24          for j = 1:zLen(i)
25
26              if j == 1, MU(i, j) = piShapedMF(z(i), ...
27                      zGrid(i, j), zGrid(i, j), zGrid(i, j), ...
28                        zGrid(i, j + 1), int32(-1));
29              elseif j == zLen(i), MU(i, j) = piShapedMF(z(i), ...
30                      zGrid(i, j - 1), zGrid(i, j), zGrid(i, j), ...
31                        zGrid(i, j), int32( 1));
32              else, MU(i, j) = piShapedMF(z(i), ...
33                      zGrid(i, j - 1), zGrid(i, j), zGrid(i, j), ...
34                        zGrid(i, j + 1), int32( 0));
35              end
36
37              if MU(i, j) ~= 0
38
39                  activeCount                = activeCount + 1;
40                  activeMFs(i, activeCount) = j;
41
42                  if activeCount > 1, break; end
43              elseif activeCount > 0, break;
44              end
45          end
46      end
47  else % default - triangular MF
48      for i = 1:zNo
49
50          activeCount = 0;
51
52          for j = 1:zLen(i)
53
54              if j == 1, MU(i, j) = triangularMF(z(i), ...
55                      zGrid(i, j), zGrid(i, j), zGrid(i, j + 1), int32(-1));
56              elseif j == zLen(i), MU(i, j) = triangularMF(z(i), ...
57                      zGrid(i, j - 1), zGrid(i, j), zGrid(i, j), int32( 1));
58              else, MU(i, j) = triangularMF(z(i), ...
59                      zGrid(i, j - 1), zGrid(i, j), ...
60                        zGrid(i, j + 1), int32( 0));
61              end
62
63              if MU(i, j) ~= 0
64
65                  activeCount                = activeCount + 1;
```

```
                activeMFs(i, activeCount) = j;

                if activeCount > 1, break; end
            elseif activeCount > 0, break;
            end
        end
    end
end
end
```

Приложение 6. Вычисление весов нечётких правил $w_i(z(t))$

```
function [ind, w] = rulesFiringDegreeV1( ...
  MU, w, zNo, zData, rulesNo, rulesInd, eps)

muCurr = zeros(zNo, 1);

for i = 1:rulesNo

    active = true;

    for j = 1:zNo % define fuzzified input values for the current rule

        muCurr(j) = MU(j, zData(j, rulesInd(i)));

        if muCurr(j) <= eps

            active = false;
            break

        end
    end
    if active

        w(rulesInd(i)) = prod(muCurr); % product AND operator

    end
end

% rules with firing degree <= 1% are not in consideration
ind = rulesInd(w(rulesInd) > eps);
w   = w(ind);
end
```

Приложение 7. Вычисление итогового выхода T–S FIS

```
% open-loop plant approximation
function [dx, y] = consequent(x, u, ind, w,  ...
    statesNo, inputsNo, outputsNo,           ...
    constB, constC, constD,                  ...
    Adata, Bdata, Cdata, Ddata,              ...
    xoffData, dxoffData, uoffData, yoffData, ...
    ssType)

if ~isempty(ind) % implication, aggregation & defuzzification

    w = w./sum(w); % convex sum property

    A     = zeros(statesNo,  statesNo);
    B     = zeros(statesNo,  inputsNo);
    C     = zeros(outputsNo, statesNo);
    D     = zeros(outputsNo, inputsNo);
    xoff  = zeros(statesNo,  1);
    dxoff = zeros(statesNo,  1);
    uoff  = zeros(inputsNo,  1);
    yoff  = zeros(outputsNo, 1);

    for i = 1:length(ind) % product implication & sum aggregation

        A = A + w(i)*Adata(:, range(ind(i), statesNo));

        if ~constB, B = B + w(i)*Bdata(:, range(ind(i), inputsNo)); end
        if ~constC, C = C + w(i)*Cdata(:, range(ind(i), statesNo)); end
        if ~constD, D = D + w(i)*Ddata(:, range(ind(i), inputsNo)); end

        if ssType ~= int32(0) % not velocity-based

            xoff = xoff + w(i)*xoffData(:, ind(i));
            uoff = uoff + w(i)*uoffData(:, ind(i));
            yoff = yoff + w(i)*yoffData(:, ind(i));

            if ssType ~= int32(1) % not linear

                dxoff = dxoff + w(i)*dxoffData(:, ind(i));

            end
        end
    end

    if constB, B = Bdata(:, range(1, inputsNo)); end
```

```
    if constC, C = Cdata(:, range(1, statesNo)); end
    if constD, D = Ddata(:, range(1, inputsNo)); end

    xL = x - xoff; % local variables
    uL = u - uoff;

    dx = A*xL + B*uL + dxoff; % weighted average defuzzification
    y  = C*xL + D*uL + yoff;

else

    dx = zeros(statesNo,  1);
    y  = zeros(outputsNo, 1);

end
```

Приложение 8. Синтез коэффициентов PDC (общий случай)

```
function [K, F] = stateFeedbackGainsLMI( ...
  linFam, report, controlInd, useEq)

% ui = -Ki*x;

eqInd      = [];
statesNo   = size(linFam.a, 1);
controlsNo = length(controlInd);
zNo        = length(struct2cell(linFam.SamplingGrid));

if useEq
    for i = 1:numel(report)
        if report(i).TerminationString == ...
          "Operating point specifications were successfully met."

            eqInd = [eqInd; i];

        end
    end
else, eqInd = (1:numel(report))';
end

rulesNo = length(eqInd);
disp(['Number of rules: ', num2str(rulesNo)]);

X = sdpvar(statesNo, statesNo);
```

```
27 F = [X >= eye(statesNo)];
28
29 for i = 1:zNo
30
31     grid{i} = [];
32
33 end
34
35 for i = 1:rulesNo
36
37     z{i} = struct2cell(linFam(:, :, eqInd(i)).SamplingGrid);
38     A{i} = linFam(:, :, eqInd(i)).a;
39     B{i} = linFam(:, :, eqInd(i)).b(:, controlInd);
40     M{i} = sdpvar(controlsNo, statesNo);
41
42     F = [F, X*A{i}' + A{i}*X - M{i}'*B{i}' - B{i}*M{i} <= 0];
43
44     for j = 1:zNo
45         if isempty(grid{j}) || isempty(find(grid{j} == z{i}{j}, 1))
46
47             grid{j} = [grid{j}, z{i}{j}];
48
49         end
50     end
51 end
52
53 for i = 1:zNo
54
55     grid{i} = sort(grid{i});
56     disp(grid{i});
57
58 end
59
60 for i = 1:rulesNo
61     for j = i + 1:rulesNo
62
63         add = true;
64
65         for k = 1:zNo
66             if abs(find(grid{k} == z{i}{k}, 1) - ...
67               find(grid{k} == z{j}{k}, 1)) > 1
68
69                 add = false;
70                 break
71
72             end
```

```
        end
        if add

            F = [F, X*A{i}' + A{i}*X + X*A{j}' + A{j}*X - ...
              M{j}'*B{i}' - B{i}*M{j} - M{i}'*B{j}' - B{j}*M{i} <= 0];
        end
    end
end

opts = sdpsettings('solver', 'sedumi', 'showprogress', 1);
optimize(F, [], opts);

X = value(X);
P = inv(X);

for i = 1:rulesNo

    M{i} = value(M{i});
    K{i} = M{i}*P;

end
end
```

Приложение 9. Вычисление итогового выхода PDC

```
% static state feedback regulator
function u = consequent1(x, ind, w, statesNo, ...
  inputsNo, Kdata, xoffData, ssType)

if ~isempty(ind) % implication, aggregation & defuzzification

    w = w./sum(w); % convex sum property

    K     = zeros(inputsNo, statesNo);
    xoff  = zeros(statesNo, 1);

    for i = 1:length(ind) % product implication & sum aggregation

        K = K + w(i)*Kdata(:, range(ind(i), statesNo));

        if ssType ~= int32(0) % not velocity-based

            xoff = xoff + w(i)*xoffData(:, ind(i));

        end
    end
```

```
22
23      xL = x - xoff; % local variables
24      u  = -K*xL;    % weighted average defuzzification
25
26  else
27
28      u = zeros(inputsNo, 1);
29
30  end
31  end
```