УДК 004.942

© 2022 г. **А.Н. Рыбалев,** канд. техн. наук
(Амурский государственный университет, Благовещенск)

ИССЛЕДОВАНИЕ БЫСТРОДЕЙСТВИЯ КОМБИНИРОВАННЫХ МОДЕЛЕЙ СИСТЕМ АВТОМАТИЧЕСКОГО РЕГУЛИРОВАНИЯ

Рассмотрены вопросы определения предельного быстродействия комбинированных моделей систем автоматического регулирования на базе виртуального и реального программируемого логического контроллера и Simulink-модели объекта, взаимодействующих по протоколу OPC.
Ключевые слова: комбинированная имитационная модель, ПЛК, управляющая программа, протокол OPC.

DOI: 10.22250/18142400_2022_74_4_29

Введение

Современные подходы к построению самоорганизующихся систем управления, в частности метод большого коэффициента усиления [1,2], предполагают использование алгоритмов регулирования, процессы в которых протекают существенно быстрее, чем процессы в самом объекте управления. В связи с этим представляется актуальной задача определения требований к быстродействию аппаратуры управления и каналов связи с объектом, удовлетворение которых позволило бы успешно реализовать предлагаемые алгоритмы в реальном времени.

В [3] рассмотрена технология построения «комбинированных» моделей систем автоматического регулирования, функционирующих в реальном времени. Технология предполагает использование виртуального контроллера, программирование которого производится на языках программирования МЭК 61131, и Simulink-модели объекта управления. Обмен данными между «контроллером» и «объектом» производится по протоколу OPC. В дальнейшем эта технология успешно применялась для апробации различных алгоритмов регулирования. В [4] показана контроллерная реализация алгоритма

адаптивного управления с явной эталонной моделью. Рассматривались два варианта систем: с виртуальным и реальным ПЛК, системы показали практически идентичное поведение.

В настоящей работе приведены результаты экспериментальных исследований по определению ограничений, накладываемых технологией на максимально достижимое быстродействие алгоритмов управления. В тех случаях, когда быстродействие зависело от аппаратной и программной платформ персонального компьютера, результаты представлены для двух вариантов:

вариант А: «слабая» конфигурация – Intel® Celeron® CPU N2840 @ 2,16 ГГц, 2 ГБ ОЗУ, Windows 8.1;

вариант В: «средняя» конфигурация – Intel® Core® i5 6200U CPU @ 2,3 ГГц, 8 ГБ ОЗУ, Windows 10.

Исследование производительности ПЛК

Контроллерные системы, программируемые в CoDeSys, допускают запуск отдельных программ в различных режимах, из которых для нас интересны два: «свободный» (по мере возможности) и «циклический» (по таймеру, с заданным интервалом). В обоих случаях для тестирования применялась программа, приведенная в приложении 1. Программа определяет количество вызовов самой себя за заданный интервал времени (1 с), причем интервал контролируется по таймеру самого контроллера. Варьируемым параметром является предельное значение счетчика цикла, в котором выполняется однаединственная операция над числом с плавающей точкой.

Результаты тестирования «свободного вызова» на виртуальном контроллере показали, что вне зависимости от количества операций (до некоторого предела) программа выполняется 50 раз в секунду, таким образом, интервал между вызовами составляет 20 мс. Фактически это означает, что никакого «свободного вызова» не реализуется и программа работает «по расписанию». Согласно теореме Котельникова (Найквиста/Шеннона), теоретически программа может обрабатывать сигналы с частотами не более 25 Гц (157,1 рад/с), а практически – еще меньше. В то же время компьютерный симулятор позволяет выполнять гигантское количество вычислительных операций. На платформе *A* удалось достигнуть показателя 1,75 млн. операций с плавающей точкой за один вызов, на платформе *B* – 2,6 млн. операций. Дальнейшее увеличение предельного значения счетчика цикла на платформе *A* приводило к снижению частоты вызова программы, на платформе *B* – к срабатыванию сторожевого таймера. Разница в поведении связана, вероятно, с особенностями операционных систем. Однако очевидно, что такое боль-

шое число операций при каждом вызове избыточно. Так, например, достаточно сложная программа адаптивного управления с явной эталонной моделью [4] содержала 7 интегрирующих блоков и 30 блоков, выполняющих арифметические операции (сложение, вычитание, умножение двух операндов). Поскольку интегрирование выполняется методом трапеций (о чем говорится в справочной системе CoDesys), общее количество операций составляет всего 7×4+30 = 58. Отсюда следует вывод, что использовать режим свободного вызова виртуального контроллера для реализации высокоскоростных алгоритмов крайне нерационально.

Реальный ПЛК (Овен ПЛК 150), как и следовало ожидать, показал в режиме свободного вызова совершенно другое поведение (рис. 1).

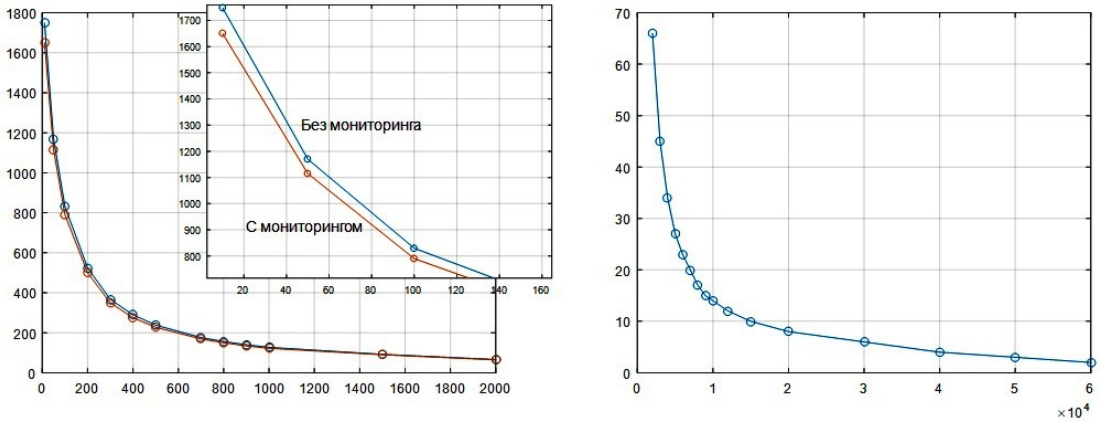

Рис. 1. ПЛК 150. Свободный вызов программы.
Зависимость частоты вызова, Гц, от числа операций с плавающей точкой.

Частота вызовов программы действительно обратно пропорциональна количеству операций. На начальном участке зависимости, где частота большая, имеет некоторое значение статус режима непрерывного мониторинга переменных со стороны системы программирования. При включении режима частота несколько уменьшается, поскольку контроллер теряет относительно большое время на обмен данными с системой. При увеличении числа программных операций доля этого времени в общем объеме активности контроллера снижается и становится несущественной. Из графиков на рис. 1 также видно, если программа состоит из нескольких десятков операций с плавающей точкой, частота ее вызова составляет 1000 и более Гц, что в 20 и более раз больше, чем частота вызова программы виртуального контроллера. Соответственно расширяется и частотный диапазон сигнала, который может быть обработан.

Основная проблема при свободном вызове программы состоит в том, что невозможно обеспечить правильную работу интегрирующих блоков, поскольку они требуют фиксированной настройки интервала между вызовами.

Поэтому были проведены исследования по определению предельной производительности программ, вызываемых в циклическом режиме. Был установлен минимально возможный в системе интервал вызова, равный 1 мс, а сторожевой таймер дезактивирован.

Виртуальный контроллер продемонстрировал следующие результаты. Заявленная частота вызова 1000 Гц оказалась недостижимой (как об этом и сообщалось в [4]). Максимальная частота составила 500 Гц и успешно поддерживалась при числе операций до 160000 на платформе *A* и 230000 – на платформе *B*. Это говорит о способности виртуального контроллера решать практические задачи с интервалом вызова программы 2 мс. При дальнейшем увеличении числа операций частота вызовов снижается обратно пропорционально числу операций и при миллионе операций составляет 85 на платформе *A* и 125 на платформе *B*.

Результаты реального ПЛК оказались более скромными. Небольшие программы (до 40 операций с плавающей точкой) он действительно способен выполнять с частотой 1000 Гц. Уменьшение частоты до 500 Гц позволяет увеличить число операций всего лишь до 170.

Производительность ввода-вывода

Ввод-вывод в комбинированной имитационной модели производится по схеме, представленной на рис. 2.

Рис. 2. Схема организации ввода-вывода в комбинированной имитационной системе.

Эта схема не зависит от того, какой ПЛК используется: реальный или виртуальный. Взаимодействие контроллера с моделью объекта управления проходит через две промежуточные программные сущности: OPC-сервер и программный шлюз CoDeSys. Шлюз обеспечивает диспетчеризацию потоков информации и представляет собой, по сути, расширение «укороченной» версии протокола TCP/IP, реализованной в ПЛК, до полнофункциональной версии. Никаких настроек временных параметров обмена он не имеет. Эти параметры настраиваются OPC-сервером и самой Simulink-моделью. Для сервера при конфигурировании устанавливается параметр Update Rate, интервал

«обновления», минимальное значение которого составляет 1 мс. Параметр определяет частоту обмена данными с контроллером. В Simulink-модели настройке подлежат блоки OPC Read и OPC Write, для которых устанавливается цикличность обмена данными с сервером.

Simulink-модели, осуществляющие обмен по протоколу OPC, функционируют в так называемом *псевдореальном* времени. В промежутках между обменами, которые производятся по таймеру, происходит пересчет состояния внутренних блоков. Если обмены выполняются относительно редко, пересчет выполняется многократно, т.е. модельное время движется быстрее реального, вследствие чего оно должно периодически «останавливаться» в ожидании очередного обмена. Если, наоборот, обмены выполняются часто, то оставшегося времени может не хватить для пересчета внутренних блоков. В таком случае фиксируется событие «Psevdo real time violation» – нарушение псевдореального времени. Реакция на это событие зависит от настройки блока OPC Config Real Time, который автоматически создается в любой модели, участвующей в обмене по OPC. Возможны три варианта: 1) Error – выдается сообщение об ошибке, расчет модели останавливается; 2) Warning – в окне команд Matlab выдается предупреждение, расчет продолжается; 3) None – событие игнорируется.

Первый вариант накладывает чрезвычайно жесткие требования к «качеству» функционирования модели. Время, необходимое для выполнения межпрограммного обмена, является переменным и существенно зависит от текущей загрузки процессора, объема доступной памяти и т.д. Поэтому, если на одном из «тактов» модельное время немного отстало от реального, вполне вероятно, что на следующих тактах оно его может догнать. С другой стороны, возможна и такая ситуация: отставание модельного времени от реального будет только увеличиваться.

Второй вариант подразумевает контроль поступления предупреждений в окне команд Matlab, что не очень удобно. Кроме того, сам вывод предупреждений требует от системы определенного времени.

На наш взгляд, лучшим решением будет выключить генерацию сообщений, а контроль модельного времени проводить по состоянию сигнала на выходном порту Psevdo real-time latency port блока OPC Config Real Time (этот выход нужно предварительно «включить» в настройках блока). Отрицательное значение сигнала означает нарушение псевдореального времени, положительное – отсутствие нарушения. Однако, как было сказано выше, одиночные нарушения могут быть скомпенсированы, поэтому фактически во многих случаях модель ведет себя вполне адекватно, если значение сигнала

на выходе блока колеблется около нуля. Однозначным признаком того, что модельное время не может «угнаться» за реальным, является постоянное уменьшение значения сигнала.

На рис. 3 показана модель для тестирования производительности ввода-вывода по OPC.

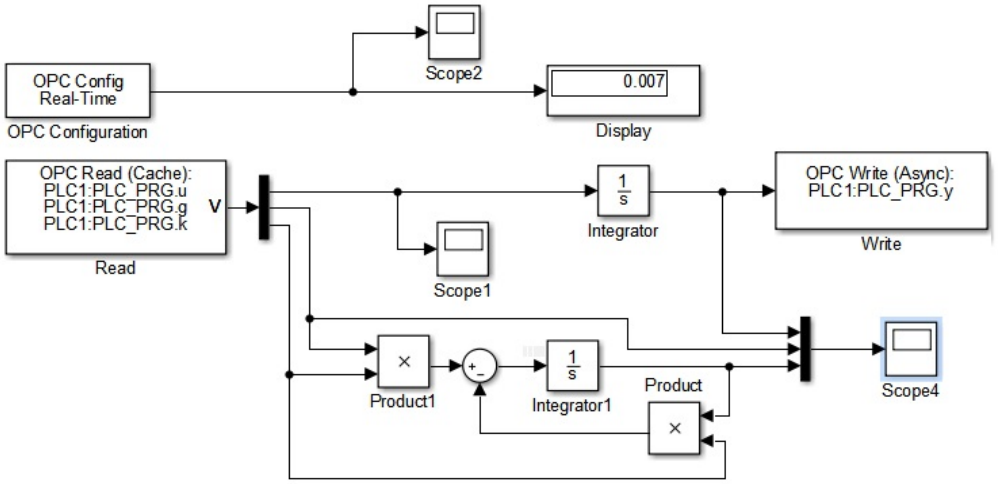

Рис. 3. Тестовая Simulink-модель.

Объектами «регулирования» являются два интегратора. Верхний интегратор находится в контуре, замыкаемом через ПЛК, а нижний – в собственном контуре с настраиваемой постоянной времени (коэффициентом обратной связи k) из программы контроллера. Передаточные функции контуров:

$$W(s)=\frac{k}{s+k}=\frac{1}{(1/k)s+1}.$$

Задание обеим системам формируется в ПЛК.

Таким образом, имеется возможность сравнивать поведение комбинированной системы с поведением системы, реализованной в Simulink.

Программа ПЛК приведена в приложении 2. Программа формирует синусоидальный «сигнал задания» с настраиваемой частотой для нижней подсистемы модели и «сигнал управления» для верхней. Программа содержит также собственную, «контроллерную» модель замкнутого контура (также для сравнения).

Вспомогательный экран визуализации (рис. 4) позволяет наблюдать сигнал задания и реакцию «комбинированного» контура и контура, полностью реализованного в контроллере. Имеются также кнопки для обнуления контроллерного времени, изменения частоты задающего сигнала и настройки коэффициента обратной связи.

К сожалению, в бесплатной версии CoDeSys минимальный интервал дискретизации трендов ограничен величиной 50 мс, поэтому при больших частотах качество графиков оставляет желать лучшего.

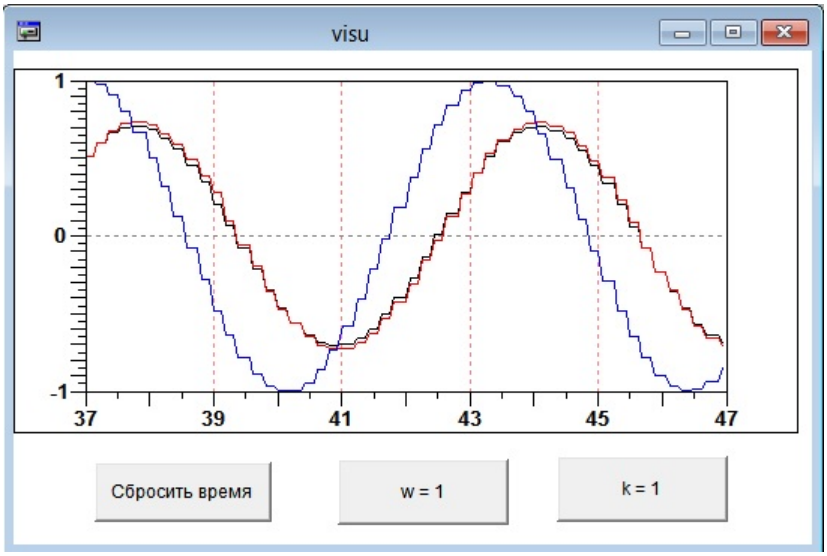

Рис. 4. Экран визуализации.

Ниже кратко изложены результаты экспериментов в системе.

Виртуальный контроллер, вариант А

Интервал обновления Update Rate OPC-сервера был установлен равным 10 мс. Попытки запустить Simulink-модель в режиме, не допускающем нарушения псевдореального времени, успеха не имели: при старте фиксировалась ошибка, и расчет останавливался даже при больших интервалах обмена Simulink-модели с сервером (1–2 сек). Поэтому далее исследования проводились по схеме, описанной выше.

Последовательно уменьшая интервал обмена, установили его минимально возможное значение, которое составило 20 мс. Дальнейшее уменьшение приводит к «катастрофическому» нарушению псевдореального времени. Так, при интервале обмена 10 мс за 100 сек. контроллерного времени отставание модельного времени составило около 30 сек. Это отставание было зафиксировано блоком Display модели и подтверждено непосредственным сравнением конечных значений модельного времени и переменной, отвечающей за текущее время в программе контроллера.

Как и следовало ожидать, отношение частот «модельных» и «контроллерных» колебаний оказалось обратным отношению «скоростей течения времени» в подсистемах. Так, при задании частоты в контроллере, равной 1 рад/с, частота модельных колебаний составила 1/0,7 ≈ 1,43 рад/с (рис. 5).

Негативные последствия «рассогласования» времени в подсистемах комбинированной модели очевидны. Если алгоритм регулирования, реализованный в ПЛК, «работает со временем» (например, выполняет интегрирование или дифференцирование), комбинированная модель будет неадекватна. (Здесь мы по умолчанию предполагаем, что в Simulink-модели динамические блоки есть всегда) .

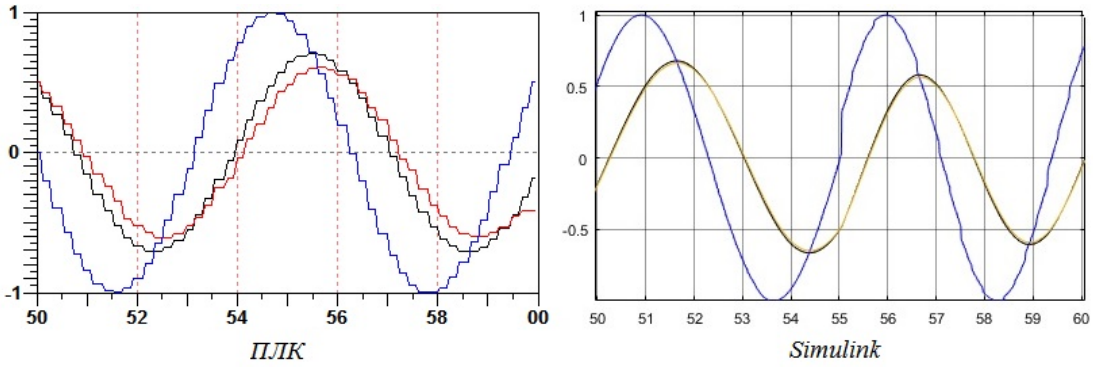

Рис. 5. Нарушение псевдореального времени.

Рассматриваемый пример в этом плане имеет некоторые особенности. ПЛК реализует безынерционный (пропорциональный) закон регулирования, поэтому отклики комбинированной модели и «чистой» Simulink-модели на правом тренде (рис. 5) практически совпадают. С другой стороны, сравнение откликов комбинированной системы и модели, полностью реализованной в ПЛК (левый тренд), показывает существенное их различие, которое объясняется тем, что в этом случае динамическое преобразование (интегрирование) было задействовано в обеих подсистемах.

В результате экспериментов с допустимой дискретностью обмена 20 мс, проведенных в диапазоне частот задающего сигнала 0,1…7,5 рад/с, выявлено предельное значение параметра k объекта, при достижении которого комбинированная модель теряет устойчивость. Как и ожидалось, оно оказалось немного меньшим 25, что соответствует постоянной времени объекта (40 мс), вдвое превышающей интервал дискретизации. На частотах выше 7,5 рад/с имели место существенные искажения синусоидальности сигналов и отличия в поведении моделей.

Виртуальный контроллер, вариант В

Интервал обновления OPC-сервера первоначально был установлен равным 5 мс, позже уменьшен до 2 мс без каких-либо негативных последствий. Запустить Simulink-модель в режиме, не допускающем нарушения псевдореального времени, не удалось. Установлено, что система стабильно работает, если интервал обмена Simulink-модели с сервером составляет 10 мс. При уменьшении этого показателя до 5 мс наблюдались значительные, хотя и самоликвидирующиеся, нарушения псевдореального времени (до нескольких секунд) при любой «внешней» активности, – такой, например, как переключение между окнами программ.

Двойное снижение интервала обмена по сравнению с вариантом А позволило примерно в два раза увеличить допустимый частотный диапазон сигналов (до 15 рад/с) и коэффициент k (до 45, что соответствует постоянной времени 22 мс).

Реальный контроллер

Основной вклад в задержку при передаче информации в данном случае, очевидно, вносит обмен между OPC-сервером и ПЛК, который по-прежнему производится по протоколу TCP/IP, но теперь уже посредством физического канала связи.

Исследования проводились на платформе A. Интервал обновления OPC-сервера был установлен равным 10 мс. Интервал пересчета программы ПЛК первоначально оставлен минимально возможным – 1 мс.

При запуске системы были обнаружены сообщения об ошибках записи, генерируемые блоком Simulink-модели OPC Write при работе в «асинхронном» режиме обмена (который, однако, успешно применялся во всех предыдущих вариантах). Ошибки появлялись нерегулярно, но связь их появления с таким фактором как включение мониторинга ПЛК со стороны CoDeSys была вполне очевидной. Согласно справочной системе Matlab [5], «synchronous writes are generally more reliable than asynchronous, but have slightly more overhead» – «синхронная запись, как правило, более надежна, чем асинхронная, но требует несколько больше накладных расходов». Следуя этой «рекомендации», в дальнейшем все эксперименты проводились в «синхронном» режиме.

Предельное значение интервала обмена между моделью и сервером составило 70 мс – меньшие значения дают необратимое нарушение псевдореального времени. Столь большой интервал, очевидно, и послужил причиной заметных амплитудных и фазовых отклонений отклика комбинированной системы от откликов систем, полностью реализованных в контроллере и Simulink, даже на небольших частотах (рис. 5).

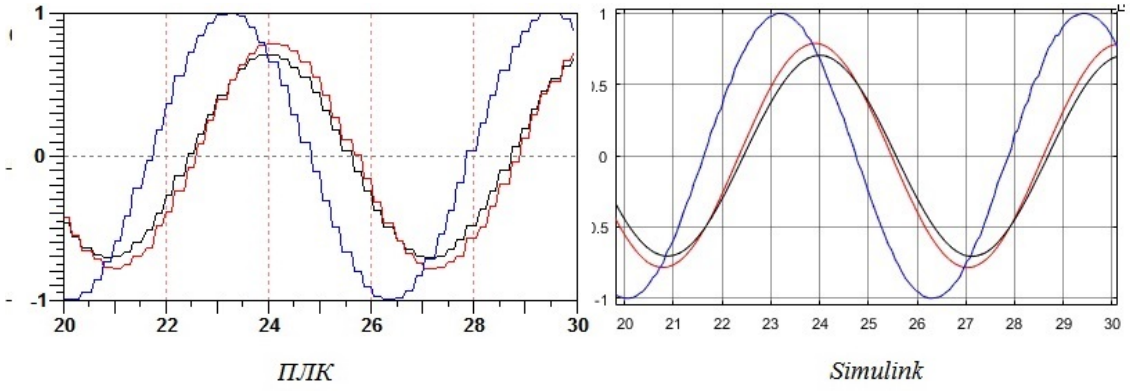

Рис. 6. Осциллограммы при ω = 1 рад/с, k = 1.

Выходные колебания комбинированной системы имеют б***о***льшую амплитуду, отстают по фазе от колебаний контроллерной модели и, наоборот, опережают колебания модели в Simulink. Отклонения возрастают с ростом частоты и уменьшаются с увеличением коэффициента k системы (т.е. сниже-

нием ее постоянной времени). Предельное значение коэффициента k составило около 7,5 (постоянная времени 0,13 с). Предельное значение частоты – 5 рад/с. Несколько улучшить достигнутые показатели можно, увеличивая интервал вызова программы ПЛК. Так, при увеличении интервала вызова до 10 мс интервал обмена можно уменьшить до 50 мс.

Масштабирование линейных динамических моделей во времени

Сопоставляя полученные выше результаты с параметрами вычислительных экспериментов [1,2], можно сделать однозначный вывод, что воспроизвести эти эксперименты на базе комбинированной модели принципиально невозможно.

Чтобы сохранить «контроллерную» реализацию алгоритмов управления в модели, можно пойти тремя путями:

1) перенести реализацию алгоритмов со всеми ее особенностями в Simulink-модель. Фактически это означает выделение в модели отдельной подсистемы, работающей в квазиреальном времени (т.е. пересчитываемую с постоянным и известным шагом и упрощенно выполняющую интегрирование). Это вполне реально, но о языках программирования контроллера придется забыть;

2) перенести модель объекта в реальный (высокопроизводительный) контроллер, который будет ее пересчитывать в реальном времени;

3) масштабировать время, искусственно замедляя одновременно динамику объекта и регулятора. В этом случае можно использовать имеющуюся технологию, а аппаратная реализация потребует лишь несложного пересчета коэффициентов регулятора.

Подробно рассмотрим последний из подходов, представляя любую передаточную функцию n-го порядка следующим образом:

$$W(s)=\frac{b'_n s^n + b'_{n-1}\omega_0 s^{n-1} + b'_{n-2}\omega_0^2 s^{n-2} + \ldots + b'_1\omega_0^{n-1} s + b'_0\omega_0^n}{s^n + a'_{n-1}\omega_0 s^{n-1} + a'_{n-2}\omega_0^2 s + \ldots + a'_1\omega_0^{n-1} s + \omega_0^n},$$

где ω_0 – так называемый среднегеометрический корень – параметр, влияющий исключительно на скорость протекания переходного процесса и ни на что более. Этот параметр определить достаточно просто. Зная ω_0, легко найти коэффициенты $b'_n \ldots b'_0$ и $a'_{n-1} \ldots a'_1$.

Для автоматизации расчетов была написана Matlab-функция, принимающая в качестве входного параметра произвольную передаточную функцию и возвращающая векторы коэффициентов числителя и знаменателя передаточной функции вида

$$W^*(s)=\frac{b'_n s^n + b'_{n-1}s^{n-1} + b'_{n-2}s^{n-2} + \ldots + b'_1 s + b'_0}{s^n + a'_{n-1}s^{n-1} + a'_{n-2}s + \ldots + a'_1 s + 1}$$

и сам параметр ω_0.

Также была написана Matlab-функция, решающая обратную задачу: нахождение передаточной функции по заданным векторам

$[b'_n \ldots b'_0]$, $[1\ a'_{n-1} \ldots a'_1\ 1]$

и среднегеометрическому корню ω_0. Коды функций приведены в приложениях 3, 4.

Разработанные функции позволяют масштабировать динамику моделей по времени таким, например, образом, чтобы получать процессы, протекающие в заданное число раз быстрее или медленнее. Для этого достаточно в это же число раз изменять параметр ω_0. Ниже приведен код тестовой программы, в которой относительно исходной передаточной функции

$W(s)=(s^2+2s+3)/(2s^3+6s^2+6s+8)$

получена передаточная функция

$W_1(s)=(s^2+4s+12)/(s^3+6s^2+12s+32)$,

описывающая объект с удвоенным быстродействием:

```
W = tf([1 2 3],[2 6 6 8])
[num, den, w0] = num_den_w0_of_W(W) %вызов функции из Прил.1
W1 = W_of_num_den_w0(num,den,2*w0)  %вызов функции из Прил.2
step(W,W1), grid
s = stepinfo(W); s1 = stepinfo(W1); s.SettlingTime/s1.SettlingTime
```

Результат выполнения программы показан на рис. 7. Отношение параметров *SettlingTime* (времени переходного процесса) для исходной и полученной передаточных функций составило ровно 2.

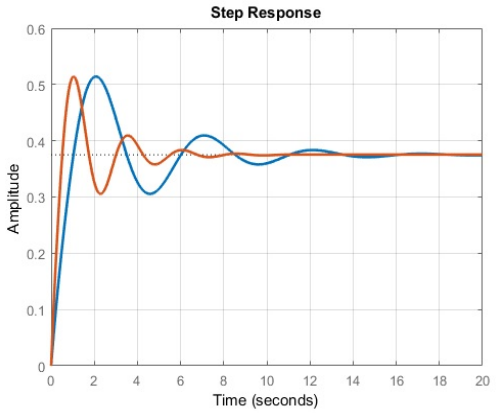

Рис. 7. Результаты тестирования функций.

Заключение

В результате исследований определены ориентировочные значения ограничений на динамику процессов в комбинированных моделях с обменом по протоколу OPC. На имеющихся аппаратных и программных средствах

получены предельное значение постоянной времени простейшего контура регулирования (около 20 мс) и допустимый частотный диапазон сигналов в нем (до 15 рад/с).

Для обхода ограничений, накладываемых технологией, предложен подход, предполагающий масштабирование линейных динамических моделей (как объекта управления, так и управляющего алгоритма) во времени. Разработаны и протестированы инструменты для осуществления такого масштабирования, которые планируется опробовать при разработке как комбинированной, так и контроллерной моделей самоорганизующейся системы, построенной по принципу большого коэффициента.

Приложение 1

Программа ПЛК для определения производительности контроллера, язык ST

```
PROGRAM PLC_PRG
VAR
      timer:TON;
      counter:DWORD:=0;
      count:DWORD;
      a:DWORD;
      b:REAL:=1;
END_VAR

timer(IN:=TRUE, PT:=T#1s);
IF NOT timer.Q THEN
      counter:=counter+1;
ELSE
      count:=counter;
      counter:=0;
      timer(IN:=FALSE);
      b:=1;
END_IF
FOR a:=1 TO 1000000 BY 1 DO
b:=b*1.0000001;
END_FOR
```

Приложение 2

Программа ПЛК для определения производительности ввода-вывода, язык ST

```
PROGRAM PLC_PRG
VAR
      t1: DWORD:=0;
      t: REAL; (*время*)
      g: REAL; (*задание*)
      u,y: REAL; (*управление и выход объекта*)
      w: REAL:=1; (*частота*)
      x: REAL:=0; (*выход внутренней модели*)
      k:REAL:=1; (*коэффициент обратной связи*)
      integ: INTEGRAL;
END_VAR

IF t1=0 THEN
      t1:=TIME_TO_DWORD(TIME());
END_IF
t:=DWORD_TO_REAL(TIME_TO_DWORD(TIME())-t1)/1000;
g:=SIN(w*t);
```

```
u:=k*g-k*y;
integ(IN:=k*g-k*x,TM:=2,out=>x); (*для реального ПЛК TM:=1 или TM:=10*)
```

Приложение 3

Код функции Matlab для определения коэффициентов $W^*(s)$ и среднегеометрического корня ω_0.

```
function [num, den, w0] = num_den_w0_of_W(W)
      [num,den]= tfdata(W);
      num=num{:}; den=den{:};
      d = length(den);
      num=num/den(1); den=den/den(1);
      w0 =  abs(den(d))^(1/(d-1));
      w0_ = w0;
      for i=2:d
            num(i) = num(i)/w0_;
            den(i) = den(i)/w0_;
            w0_=w0_*w0;
      end
end
```

Приложение 4

Код функции Matlab для определения $W(s)$ по коэффициентам $W^*(s)$ и среднегеометрическому корню ω_0

```
function W = W_of_num_den_w0(num, den, w0)
      d = length(den);
      w0_ = w0;
      for i=2:d
            num(i) = num(i)*w0_;
            den(i) = den(i)*w0_;
            w0_= w0_*w0;
      end
      W = tf(num,den);
End
```

ЛИТЕРАТУРА

1. *Еремин Е.Л.* Метод большого коэффициента усиления в задаче самоорганизации систем управления структурно неопределенными линейными объектами с переключениями. I // Информатика и системы управления. – 2021. – №4(70). – С.95-109.
2. *Еремин Е.Л.* Метод большого коэффициента усиления в задаче самоорганизации систем управления структурно неопределенными линейными объектами с переключениями. II // Информатика и системы управления. – 2022. – №2(72). – С.60-73.
3. *Рыбалев А.Н., Николаец Ф.А.* Разработка и эмулирование АСУ ТП с использованием программ разных производителей и типов // Вестник АмГУ. – 2014. – Вып. 65. – С. 73-82.
4. *Рыбалев А.Н.* Реализация алгоритма адаптивного управления с эталонной моделью для программируемых логических контроллеров // Информатика и системы управления. – 2021. – №4(70). – С.30-38.
5. OPC Write. Write data to OPC server // MathWorks ® Help Center [Электронный ресурс]. Режим доступа: https://www.mathworks.com/help/icomm/ug/opcwrite.html.

Статья представлена к публикации членом редколлегии Е.Л. Ереминым.

E-mail:
Рыбалев Андрей Николаевич – amgu_appe@mail.ru