

САМОНАСТРАИВАЮЩИЙСЯ РЕГУЛЯТОР В СРЕДЕ IntelligentPad

*БОДЯНСКИЙ Е.В., РУДЕНКО О.Г., ТАНАКА Ю.,
ШТЕФАН А., КОЛОДЯЖНЫЙ В.В., ПОПОВ С.В.*

Описывается реализация алгоритмов цифрового управления реальным временем на платформе IntelligentPad. На основе ранее разработанных алгоритмов управления и идентификации создан самонастраивающийся регулятор. Тестирование регулятора в специально разработанной среде моделирования выявило высокое качество управления нестационарным динамическим объектом при наличии значительного стохастического возмущения.

Введение

Важнейшим этапом создания современных интеллектуальных цифровых систем управления является проведение имитационного моделирования разработанных алгоритмов и моделей для экспериментальной проверки теоретических результатов.

В настоящее время созданы и успешно используются десятки различных систем имитационного моделирования, как универсальных, так и специализированных. Они, как правило, имеют очень хорошее математическое оснащение, но страдают недостаточной гибкостью интерфейса с внешним миром. Это выражается в ограниченности возможностей визуального представления информации, используемых форматов данных и механизмов управления процессом моделирования только жестко заданными средствами, предусмотренными разработчиками системы. Большинство таких систем также плохо приспособлено для моделирования процессов реального времени и не позволяют вносить изменения в моделируемую систему в ходе ее функционирования.

На наш взгляд, исключить описанные трудности при проведении имитационного моделирования систем, включая системы реального времени, можно, используя в качестве платформы для реализации среду визуального создания приложений IntelligentPad [1–5].

1. Описание платформы IntelligentPad

Приложение в среде IntelligentPad, разработанной специалистами Meme Media Laboratory университета Хоккайдо (Саппоро, Япония), представляет собой набор взаимосвязанных функциональных единиц, называемых компонентами (pads), имеющих визуальное представление в виде блоков прямоугольной формы. Компоненты являются функционально завершенными независимыми модулями. Это могут быть и простейшие элементы ввода, и отображения чисел или строк, и сложные средства доступа к базам данных — степень функциональной нагрузки не является определяющим фактором. Компоненты имеют стандартизированный интерфейс, позволяющий легко соединять их между

собой и объединять в функциональные группы. Такие группы впоследствии легко могут быть перенесены в другое приложение или снова разбиты на составляющие, что является залогом повторного использования ранее разработанных и отлаженных модулей. Среда разработки IntelligentPad содержит большое количество готовых прикладных компонентов.

Связь компонентов между собой происходит посредством так называемых “слотов”, представляющих собой доступные извне элементы данных компонента. Слоты имеют типы, что соответствует типам данных в универсальных языках программирования. В случае недостатка имеющихся типов для представления данных решаемой задачи легко могут создаваться новые, удовлетворяющие требованиям конкретного приложения. Изменение состояния слота вызывает процесс обновления в компоненте, который может привести к обновлению состояния других слотов компонента и других соединенных с ним компонентов. Таким образом, событие, произошедшее в одном из компонентов, вызывает процесс обновления состояния всех прямо или опосредованно связанных с ним компонентов во всей системе. Этот процесс напоминает распространение нервных импульсов в живых организмах и оказывается очень полезным при моделировании искусственных нейронных сетей.

Новые компоненты создаются путем изменения и композиции существующих или специальными средствами IntelligentPad, позволяющими использовать фрагменты кода на C или BASIC. Более сложные компоненты могут быть разработаны внешними средствами с использованием универсальных языков программирования. Таким образом, это позволяет преодолевать барьер исходной функциональной ограниченности при создании специализированных систем имитационного моделирования, а также делает возможным использование собственных способов отображения информации и работы с внешними данными. Кроме того, существует специализированный компонент, позволяющий применять в системе элементы управления ActiveX. Это дает возможность использовать широкие возможности существующих библиотек элементов ActiveX, включая такие мощные средства, как доступ в Internet непосредственно из моделируемой системы.

IntelligentPad — совершенно новый способ создания приложений. Для этого достаточно выбрать необходимые компоненты, разместить их на экране, соединить вместе в соответствии с функциональностью приложения и придать им желаемый внешний вид. Важной отличительной особенностью приложений IntelligentPad является их визуальное представление: разработчик видит на экране то же самое, что и пользователь приложения. Кроме того, в системе IntelligentPad изначально заложен принцип кроссплатформенности: среда разработки существует для различных операционных систем, а сами компоненты являются абсолютно платформо-независимыми.

Еще одной уникальной особенностью IntelligentPad является отсутствие разграничения режимов разработки (design-time) и выполнения (run-time): в процессе создания приложения компоненты уже являются активными, т.е. любое изменение свойств компонента немедленно воспринимается им самим и другими связанными с ним компонентами. Это свойство становится незаменимым при создании систем имитационного моделирования реального времени.

Указанные особенности позволяют эффективно использовать IntelligentPad для решения самых различных задач. Механизмы асинхронного выполнения, простота реализации вычислительных алгоритмов с помощью встроенных языков программирования, высокая производительность и устойчивость работы, низкие требования к объему оперативной памяти обеспечивают более удобные возможности моделирования систем, функционирующих в реальном времени, по сравнению с популярной системой MATLAB / SIMULINK.

Таким образом, представляется перспективным, а иногда и единственно возможным, использование системы IntelligentPad для моделирования интеллектуальных систем управления реального времени и, прежде всего, адаптивных, нейро- и фаззи-процедур.

В настоящей статье синтезирован самонастраивающийся регулятор для нестационарного стохастического динамического объекта. Имитационное моделирование разработанной адаптивной цифровой системы управления выполнено в среде IntelligentPad.

2. Постановка задачи

Пусть объект управления (ОУ) описывается разностным уравнением типа ARX:

$$y(t) + a_1 y(t-1) + \dots + a_n y(t-n) = b_1 u(t-1) + \dots + b_m u(t-m) + w(t), \quad n \geq m, \quad (1)$$

где $y(t)$ – выходная переменная объекта управления в дискретный момент времени t ; $u(t)$ – сигнал управления; $w(t)$ – неизмеряемое возмущение типа белого гауссовского шума с нулевым математическим ожиданием и ограниченной дисперсией.

Качественный смысл любой задачи теории управления состоит в выборе таких управляющих воздействий $u(t)$, чтобы ОУ вел себя так, чтобы выходные сигналы $y(t)$ были близки к внешнему желаемому закону $y^*(t)$, т.е. требуется обеспечить минимум некоторой функции от ошибки управления $v_c = y^*(t) - y(t)$ [6]. Для решения этой задачи в системах управления технологическими процессами наиболее часто применяются пропорционально-интегрально-дифференциальные (ПИД) регуляторы.

Уравнение ПИД-регулятора в непрерывном времени имеет вид

$$u(\tau) = k \left(v_c(\tau) + \frac{1}{T_I} \int_0^\tau v_c(l) dl + T_D \frac{dv_c(\tau)}{d\tau} \right), \quad (2)$$

где k – коэффициент усиления регулятора; T_I – постоянная интегрирования; T_D – постоянная дифференцирования; τ – непрерывное время.

Для реализации закона управления (2) цифровой системой используют его разностное представление с периодом квантования T_0 . Заменяв непрерывное интегрирование интегрированием по методу прямоугольников, можно получить позиционный алгоритм управления в виде

$$u(t) = k \left(v_c(t) + \frac{T_0}{T_I} \sum_{i=0}^t v_c(i-1) + \frac{T_D}{T_0} (v_c(t) - v_c(t-1)) \right). \quad (3)$$

Вычитая из (3) уравнение

$$u(t-1) = k \left(v_c(t) + \frac{T_0}{T_I} \sum_{i=0}^{t-1} v_c(i-1) + \frac{T_D}{T_0} (v_c(t-1) - v_c(t-2)) \right), \quad (4)$$

получаем

$$u(t) - u(t-1) = g_0 v_c(t) + g_1 v_c(t-1) + g_2 v_c(t-2), \quad (5)$$

где

$$\begin{cases} g_0 = k(1 + T_D T_0^{-1}), \\ g_1 = -k(1 + 2T_D T_0^{-1} - T_0 T_I^{-1}), \\ g_2 = k T_D T_0^{-1}. \end{cases} \quad (6)$$

Уравнение (5) может быть также представлено в форме

$$u(t) = u(t-1) + g_0 v_c(t) + g_1 v_c(t-1) + g_2 v_c(t-2) = u(t-1) + (g_0 + g_1 q^{-1} + g_2 q^{-2}) v_c(t) = u(t-1) + G(q^{-1}) v_c(t), \quad (7)$$

(здесь q^{-1} – оператор сдвига назад, $G(q^{-1}) = g_0 + g_1 q^{-1} + g_2 q^{-2}$). При этом в процессе управления вычисляется только текущее приращение управляющей переменной, и поэтому этот алгоритм еще называется скоростным [6].

3. Алгоритм настройки параметров регулятора

Для синтеза алгоритма настройки параметров регулятора представим модель объекта (1) в виде

$$y(t) = -a_1 y(t-1) - \dots - a_{n_A} y(t-n_A) + b_1 u(t-1) + \dots + b_{n_B} u(t-n_B) + w(t) = b_1 u(t-1) + a^T \psi(t-1) + w(t), \quad (8)$$

а регулятор (7) в виде

$$u(t) = u(t-1) + g_0 v_c(t) + g_1 v_c(t-1) + g_2 v_c(t-2) = u(t-1) + g^T V_c(t). \quad (9)$$

Здесь $a = (b_2, \dots, b_{n_B}, -a_1, \dots, -a_{n_A})^T$,

$$\psi(t-1) = (u(t-2), \dots, u(t-n_B), y(t-1), \dots, y(t-n_A))^T,$$

$$g = (g_0, g_1, g_2)^T, \quad V_c(t) = (v_c(t), v_c(t-1), v_c(t-2))^T.$$

Запишем выражение для ошибки управления в $(t+1)$ -й момент времени:

$$v_c(t+1) = y^*(t+1) - b_1 u(t-1) - b_1 g^T V_c(t) - a^T \psi(t) - w(t+1) = \tilde{v}(t+1) - b_1 g^T V_c(t) - w(t+1) \quad (10)$$

(здесь $\tilde{v}(t+1) = y^*(t+1) - a^T \psi(t) - b_1 u(t-1)$) и введем критерий оптимизации

$$J_t^c = M \{ v_c^2(t+1) + r(u(t) - u(t-1))^2 \}. \quad (11)$$

Заметим также, что значение ошибки $\tilde{v}(t+1)$ может вычисляться рекуррентно согласно соотношению

$$\tilde{v}(t+1) = \tilde{v}(t) - a^T \Delta \psi(t) - b_1 \Delta u(t-1) + \Delta y^*(t+1), \quad (12)$$

где $\Delta x(t) = x(t) - x(t-1) = (1 - q^{-1})x(t)$.

Применяя для минимизации критерия (11) процедуру типа Гаусса-Ньютона

$$g(t) = g(t-1) - [J_{gg}^c]^T \nabla_g J_t^c \quad (13)$$

(здесь $\nabla_g J_t^c = -2\tilde{v}(t+1)b_1 V_c(t) + 2(b_1^2 + r)V_c(t)V_c^T(t)g(t-1)$)

— вектор-градиент критерия J_t^c по параметрам g ;

$[J_{gg}^c]^T = 0.5(b_1^2 + r)^{-1} V_c(t)V_c^T(t)\|V_c(t)\|^{-4}$ — матрица, псевдообратная к матрице $[J_{gg}^c]$, образованной вторыми производными критерия по настраиваемым параметрам), приходим к алгоритму настройки параметров:

$$g(t) = g(t-1) + \frac{\tilde{v}(t+1)b_1 - (b_1^2 + r)g^T(t-1)V_c(t)}{(b_1^2 + r)} \cdot \frac{V_c(t)}{\|V_c(t)\|^2}. \quad (14)$$

С использованием полученных оценок регулятор (9) может быть представлен в виде

$$u(t) = u(t-1) + g_0(t)v_c(t) + g_1(t)v_c(t-1) + g_2(t)v_c(t-2) = u(t-1) + g^T(t)V_c(t). \quad (15)$$

4. Идентификация настраиваемой модели

Алгоритм управления (14), (15) подразумевает, что параметры объекта известны. Поскольку в общем случае это не так, для реализации такого алгоритма можно воспользоваться оценками, полученными в процессе идентификации по ходу функционирования объекта управления и заменить в законе управления истинные параметры объекта их текущими оценками. Заметим, что такая замена вполне корректна, поскольку в рамках рассматриваемой задачи справедлив принцип стохастической эквивалентности [7]. Реализацию такого подхода рассмотрим на примере объекта (1) и закона управления (14), (15).

Запишем уравнения объекта в виде

$$y(t) = -a_1 y(t-1) - \dots - a_{n_A} y(t-n_A) + b_1 u(t-1) + \dots + b_{n_B} u(t-n_B) + w(t) = \Theta^T \varphi(t) + w(t), \quad (16)$$

и поставим ему в соответствие уравнение настраиваемой модели

$$\hat{y}(t) = \hat{\Theta}^T(t-1)\varphi(t). \quad (17)$$

В теории и практике адаптивных систем наряду с рекуррентным методом наименьших квадратов (РМНК) и его модификациями широкое распространение получили алгоритмы, основанные на стохастической аппроксимации.

Рассмотрим модифицированный алгоритм типа стохастической аппроксимации [8]:

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + \alpha r^{-1}(t)(y(t) - \hat{\Theta}^T(t-1)\varphi(t))\varphi(t), \quad (18)$$

где $r(t) = \alpha r(t-1) + \|\varphi(t)\|^2$, $0 \leq \alpha \leq 1$, $r(0) = 1$, $0 < \alpha < 2$, совпадающий при $\alpha = 0$ с одношаговым алгоритмом Качмажа, а при $\alpha = 1$ — с адаптивным алгоритмом Гудвина-Рэмеджа-Кэйнса [9].

Принципиальным отличием алгоритма (18) от алгоритма, введенного в [9], является использование в нем параметра дисконтирования α , что обеспечивает процедуре требуемые адаптивные свойства и в то же время исключает возможность “взрыва параметров”.

Сравнительный анализ алгоритмов РМНК и (18) показывает, что процедура РМНК, обладая высокой скоростью сходимости, работает в весьма узком диапазоне изменения параметра дисконтирования, а алгоритм (18), как и все процедуры стохастической аппроксимации, характеризуется низкой скоростью сходимости.

5. Практическая реализация алгоритма и результаты моделирования

Рассмотрим реализацию адаптивного регулятора (14), (15), (18) для объекта (1) в среде IntelligentPad. Для этого создадим следующие компоненты: ARX-модель, самонастраивающийся регулятор, блок идентификации, а также блок генератора нормального распределенных случайных чисел, реализующего возмущение типа гауссовского, которое действует на выход объекта управления. Создание компонентов осуществляется с использованием специального мастера для Microsoft Visual C++, а также библиотек, поставляемых со средой IntelligentPad.

Затем создадим простую систему для тестирования регулятора в различных условиях. После этого выполним моделирование для проверки качества функционирования разработанного адаптивного регулятора.

Для создания нового компонента необходимо определить, какие слоты он должен иметь, а также типы данных этих слотов. Все данные, с которыми будут работать компоненты адаптивного регулятора, являются цифровыми, поэтому соответствующим

щие слоты будут иметь тип **NUMBER** или **DICTIONARY**, состоящий из элементов типа **NUMBER**. **DICTIONARY** — это составной тип данных, который может включать элементы других типов, в том числе и **DICTIONARY**. Таким образом, может быть реализована передача векторных значений между компонентами. Кроме того, некоторые слоты, предназначенные для получения тактовых сигналов от таймера, будут иметь специальный тип **COMMAND**. Имена слотов и их типы приведены в табл. 1–4. Жирным шрифтом выделены выходные слоты.

Таблица 1
Слоты компонента ARX-модели

Имя слота	Тип	Назначение
y	NUMBER	Выход объекта
w	NUMBER	Возмущающее воздействие
u	NUMBER	Сигнал управления
process	COMMAND	Сигнал от таймера
params	DICTIONARY	Параметры объекта
ut	DICTIONARY	Вектор предыстории сигнала управления
yt	DICTIONARY	Вектор предыстории выходного сигнала

Таблица 2
Слоты компонента самонастраивающегося регулятора (STC)

Имя слота	Тип	Назначение
u	NUMBER	Сигнал управления
y	NUMBER	Выход объекта
yref	NUMBER	Задающий сигнал
theta	DICTIONARY	Параметры объекта
r	NUMBER	Параметр критерия управления
gt	DICTIONARY	Вектор параметров цифрового ПИД-регулятора
ut	DICTIONARY	Вектор предыстории сигнала управления
yt	DICTIONARY	Вектор предыстории выходного сигнала
et	DICTIONARY	Вектор предыстории сигнала ошибки

Тестовая система содержит следующие блоки (рис. 1):

- 1) модель объекта управления (ARX);
- 2) самонастраивающийся регулятор (STC);
- 3) блок идентификации (IDENT);
- 4) генераторы задающих сигналов и сигналов возмущений и дрейфа параметров объекта;
- 5) переключатели сигналов;

Таблица 3
Слоты блока идентификации (IDENT)

Имя слота	Тип	Назначение
theta	DICTIONARY	Параметры настраиваемой модели
na	NUMBER	Размерность вектора выходов
nb	NUMBER	Размерность вектора сигналов управления
theta0	NUMBER	Начальное значение параметров настраиваемой модели
alpha	NUMBER	Параметр забывания
y	NUMBER	Выход объекта
u	NUMBER	Сигнал управления
yref	NUMBER	Задающий сигнал
theta	DICTIONARY	Параметры объекта
r	NUMBER	Переменная r в формуле (18)
ut	DICTIONARY	Вектор предыстории сигнала управления
yt	DICTIONARY	Вектор предыстории

Таблица 4
Слоты генератора нормально распределенных случайных чисел (GAUSS)

Имя слота	Тип	Назначение
out	NUMBER	Выход генератора
process	COMMAND	Сигнал от таймера
sx	NUMBER	Среднеквадратичное отклонение
mx	NUMBER	Математическое ожидание

- 6) тактовый генератор — таймер;
- 7) элементы управления типа Slider;
- 8) цифровые индикаторы;
- 9) кнопки запуска и сброса времени;
- 10) компоненты для построения графиков (не показаны).

Генераторы и переключатели сигналов созданы с помощью компонента BASIC Programming. Остальные блоки являются стандартными компонентами системы IntelligentPad.

Моделирование выполнено для объекта второго порядка, описываемого уравнением (1). При этом параметры объекта менялись по синусоидальному закону. Начальные значения параметров объекта: $a_1=0.2$, $a_2=0.3$, $b_1=0.7$, $b_2=0.8$. На объект управления действовало нормально распределенное возмущение со следующими параметрами: $M\{w(t)\} = 0$, $M\{w^2(t)\} = y_w^2$, $y_w = 0.05$.

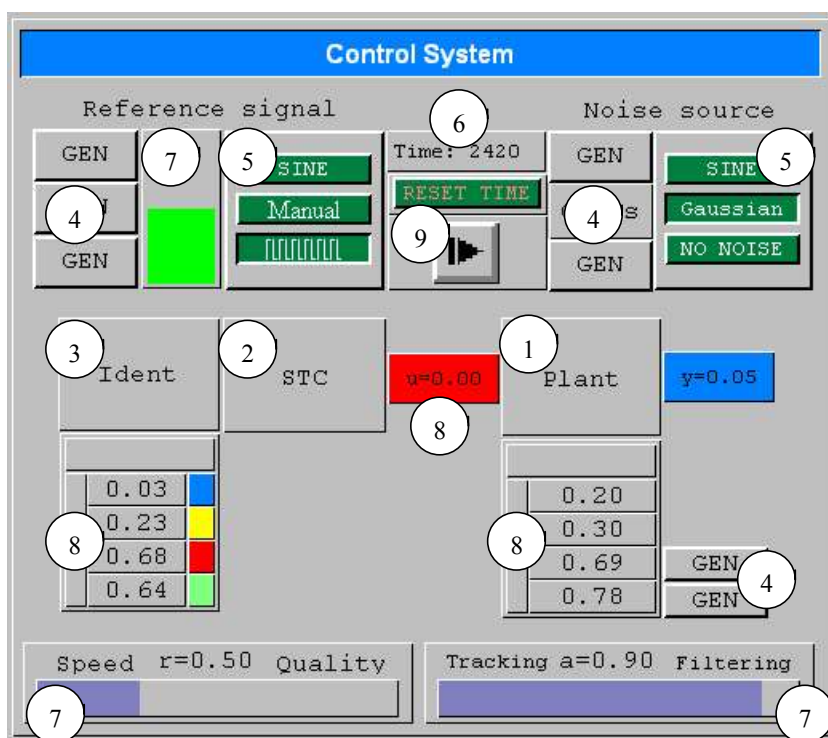


Рис. 1. Тестовая система

Результаты моделирования приведены на рис. 2. По виду графиков можно сделать вывод о том, что самонастраивающийся регулятор обеспечивает высокое качество управления нестационарным объектом.

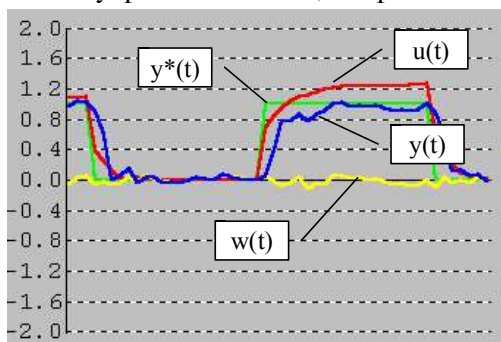


Рис. 2. Графики изменения выходного сигнала $y(t)$, сигнала управления $u(t)$, задающего сигнала $y^*(t)$ и возмущения $w(t)$

Рис.3 иллюстрирует изменение оценок параметров объекта.

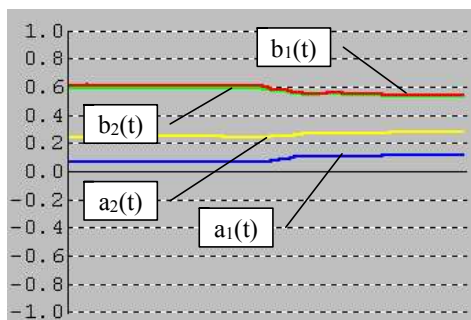


Рис. 3. Графики изменения оценок параметров объекта, вычисляемых в блоке идентификации

Для построения графиков использован стандартный компонент Graph.

Литература: 1. Tanaka Y. Meme Media and Meme Market Architectures for the Reediting and Redistribution of Knowledge Resources // International Conference on Multimedia Modeling'98. Lausanne. 1998. 2. Tanaka Y. Meme Media and a World-Wide Meme Pool // The Fourth ACM International Multimedia Conference. Nov. Boston. 1996. P. 175-186. 3. Tanaka Y. From Augmentation to Meme Media: IntelligentPad and the World-Wide Repository of Pads // Proc. of European Japanese Seminar on Information Modeling and Knowledge Bases. Kista, Sweden. June, 1994. 4. Tanaka Y., Nagasaki A., Akaishi M., Noguchi T. A synthetic Media Architecture for an Object-Oriented Open Platform // Personal Computers and Intelligent Systems, Information Processing (ed. F.H. Vogt). 1992. Vol. III. P. 104-110. 5. Bodyanskiy Ye., Rudenko O., Tanaka Y., Stephan A., Popov S. Investigations Regarding Adaptive PID Controllers and their Implementation in IntelligentPad. Meme Media Laboratory, Hokkaido-University, 1999. 6. Бодянский Е.В., Котляревский С.В., Ачкасов А.Е., Вороновский Г.К. Адаптивные регуляторы пониженного порядка. Харьков: ХГАГХ, 1996. 144 с. 7. Isermann R. Digitale Regelsysteme. Band I: Grundlagen Deterministische Regelungen. Berlin: Springer-Verlag, 1988. 8. Бодянский Е.В., Плисс И.П., Соловьева Т.В. Многошаговые оптимальные упредители многомерных нестационарных стохастических процессов // Докл. АН УССР. 1986. А. № 12. С. 47-49. 9. Goodwin G.C., Ramadge P.J., Caines P.E. A globally convergent adaptive predictor, *Automatica*, Vol. 17, No. 1, 1995. P. 135-140.

Поступила в редколлегию 12.08.2000

Рецензент: д-р техн. наук, проф. Петров Э.Г.

Бодянский Евгений Владимирович, д-р техн. наук, профессор кафедры искусственного интеллекта, научный руководитель ПНИЛ АСУ ХТУРЭ, член IEEE. Научные интересы: адаптивные системы, искусственные нейронные сети. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 40-98-90. E-mail: bodyanskiy@ieee.org

Руденко Олег Григорьевич, д-р техн. наук, профессор, заведующий кафедрой ЭВМ ХТУРЭ, член IEEE. Научные интересы: идентификация систем, искусственные нейронные сети. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел.: 40-93-54. E-mail: roden@kture.kharkov.ua

Танак Юзуру, доктор, профессор, директор Meme Media Laboratory, университет Хоккайдо, Саппоро, Япония. E-mail: tanaka@meme.hokudai.ac.jp

Штефан Андреас, доктор-инженер, почетный профессор ХТУРЭ, глава фирмы Dr.Stephan & Partner, Ильменау, Германия.

E-mail: stephan@dr-stephan-und-partner.de

Колодяжный Виталий Владимирович, аспирант кафедры искусственного интеллекта, младший научный сотрудник ПНИЛ АСУ ХТУРЭ, член IEEE. Научные интересы: адаптивные, нечеткие и нейро-фаззис системы. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 40-98-90. E-mail: kolodyazhnyi@ieee.org

Попов Сергей Витальевич, аспирант кафедры искусственного интеллекта, младший научный сотрудник ПНИЛ АСУ ХТУРЭ, член IEEE. Научные интересы: адаптивная обработка информации в двумерных системах, нейро-технологии. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 40-98-90.

E-mail: Serge.Popov@ieee.org