other publications listed in reference. We propose to apply these pulse forms for ultra-wideband communication systems due to simple generation and high value of mask utilization coefficient.

Keywords: *optimization, spectral mask, ultra-wideband communication, wavelet, Hermite polynomials*

DOI: 10.18469/ikt.2015.13.4.05

Kislinskiy Vladislav Sergeevich, student, Department of Telecommunication systems, Ufa State Aviation Technical University, Ufa, Russian Federation. Tel. +79371695341. E-mail: be.me@bk.ru
Grakhova Elizaveta Pavlovna, PhD-student, Assistant of the Department of Telecommunication systems, Ufa State Aviation Technical University, Ufa, Russian Federation. Tel.: +79173522353. E-mail: eorlingsbest@mail.ru
Abdrakhmanova Guzel Idrisovna, PhD in Technical Science, Senior Lecturer of the Department of Telecommunication systems, Ufa State Aviation Technical University. Tel.: +79874739042. E-mail: tekasesochka@yandex.ru

References

1. *Resheniya Gosudarstvennoi komissii po radiochastotam (GKRCh)* [Decisions of State Radio Frequencies Commission (SRFC)]. Approved on 15 Dec., prot. № 09-05. Available at: http://minsvyaz.ru/uplo aded/files/prot_09-05_1.pdf (accessed 16.08.2010).
2. Abdrakhmanova G.I., Bagmanov V.Kh. Modelirovanie SShP impulsa na osnove proizvodnikh Releya i geneticheskogo algoritma [UWB pulse shaping on the basis of Rayleigh derivatives and genetic algorithm]. *Infokommunikacionnye tehnologii*, 2013, vol. 11, no. 3, pp. 84-88.
3. Kalinin V.O., Nosov V.I. Otsenka parametrov korotkoimpulsnoi sverkhshirokopolosnoi sistemi svyazi [Short pulse ultra wideband communication system parameters assessment]. *Vestnik SibGUTI,* 2011, no. 11, pp. 73-85.
4. Grakhova E.P., Meshkov I.K., Bagmanov V. Kh., Vinogradova I.L. Modelirovanie SShP radioimpulsov na osnove proizvodnikh Gaussa i Releya s uchjetom spektralnoi maski GKRCh [UWB radiopulses modeling on the basis of Gaussian and Rayleigh derivatives taking into account SRFC spectral mask] (In Russ.). *Elektrotelhnocheskie i informatsionniye kompleksy i sistemy,* 2014, no. 3, vol.10, pp.62-69.
5. Michael, L.B., Ghavami, M., Kohno, R. Multiple pulse generator for Ultra-wideband communication using Hermite polynomial based orthogonal pulses. *Proc. IEEE Conf. on Ultra wideband systems and technologies*, 2002, pp. 47-51. doi: 10.1109/UWBST.2002.1006316
6. Hedayati H., Fotowat-Ahmady A. A novel tunable UWB pulse design for narrowband interference suppression implemented in BiCMOS technology. *Proc. IEEE Conf.*, 2009 pp. 405-408. doi: 10.1109/ISCAS.2009.5117771
7. Lazorenko O.V., Chernogor L.Ph. Sverkhshirokopolosniye signaly i phizicheskiye protsessi. 1. Osnovniye ponyatiya, modeli i metody opisaniya [Ultra wideband signals and physical processes. 1. The main concepts, models and description methods]. *Radiophisika i radioastronomiya*, 2008, vol.13, no. 2, pp. 166-194.

Received 19.03.2015

ТЕХНОЛОГИИ КОМПЬЮТЕРНЫХ СИСТЕМ И СЕТЕЙ

УДК 004.41

АЛГОРИТМЫ И ПРОГРАММНЫЙ ИНСТРУМЕНТ ПОСТРОЕНИЯ ЭКВИВАЛЕНТНЫХ ПРЕДСТАВЛЕНИЙ ИСХОДНЫХ ТЕКСТОВ ПРОГРАММ

Ковалевский А.А., Пустыгин А.Н.
Челябинский государственный университет, Челябинск, РФ
E-mail: morskoyzmey@gmail.com

В статье рассмотрен прототип программного инструмента статического анализа программных систем, основанный на специальном наборе данных, полученном из исходного текста программ с помощью компилятора с открытым исходным кодом. Прототип позволяет получать эквивалентные представления путем линейных и нелинейных преобразований этого набора данных. Пользовательский интерфейс построен на языке запросов, синтаксис которого позволяет задавать произвольные сочетания доступных преобразований с дополнительными параметрами, соответствующими целям анализа. Варианты преобразований определяются в конфигурационном файле, который представляет собой XML-документ. Прототип также позволяет выполнять анализы над полученными

эквивалентными представлениями. В статье в качестве примера описан «температурный» анализ синтаксической перегруженности строк и сложности блоков исходного текста в контексте потока управления. Прототип был протестирован на проекте с открытым исходным текстом – библиотеке для разбора XML-документов.

Ключевые слова: промежуточное представление, исходный код, поток данных, поток управления, синтаксическое дерево.

Задача статического анализа программных систем является одним из традиционных способов улучшения программ [1].

В предшествующих разработках [2] было предложено и реализовано универсальное промежуточное представление (УПП) открытого исходного текста, предназначенное для последующего преобразования, анализа и извлечения информации из исходного кода программ. На основе этого набора данных был построен прототип генератора системы эквивалентных представлений, предназначенных для решения некоторой совокупности задач над исходным текстом.

Рассмотренный прототип позволяет получать эквивалентные представления исходного текста путем линейных и нелинейных преобразований его УПП, выполнять анализы над этими представлениями и получать срезы этих представлений [3]. Помимо известных, возможно получение объединенных представлений с заданными общими параметрами среза. Правила построения представлений задаются в конфигурационном файле.

Разработанный прототип не является законченным пользовательским продуктом статического анализа, которыми являются соответствующие пакеты и плагины сред разработки (например, Microsoft Visual Studio и IntelliJ IDEA). Однако функциональные возможности прототипа на данный момент покрывают де-факто или в перспективе большинство широко используемых функций статического анализа (анализы потока управления, потока данных, диаграмма классов в различных аспектах). Например, функция поиска использования переменной (или поля объекта) в IntelliJ IDEA уже входит в базовый комплект фильтров прототипа, а функция иерархии вызовов в Visual Studio легко может быть реализована поверх соответствующего представления. В данном прототипе эти функции усилены за счет использования дополнительных правил и сложных условий параметризации выборки: заданием контекста при поиске использования поля объекта можно исключить 90% ненужной информации и оставить, например, только те вхождения, которые лежат в условиях циклов определенного класса.

Прототип допускает произвольные объединения известных представлений, которые являются не изученными ранее. Возможность задания своих представлений в прототипе практически не ограничена за счет полного доступа к дереву синтаксического разбора (AST) через УПП и достаточной гибкости правил, формирующих выборку. Это позволяет выполнять специфичные анализы, которым в пакетах статического анализа не нашлось места. Например, в различного рода анализах потока управления учитываются неявные вызовы перегруженных операторов и конструкторов копирования.

Пользовательский запрос для описания требуемого анализа (преобразования)

Пользователь вводит запрос в инструмент со стандартного ввода. Запрос представляет собой комбинацию доступных в конфигурационном файле фильтров с дополнительными параметрами и правилами их совмещения, а также анализ, выполняемый над полученным представлением.

Результатом применения любого фильтра является подмножество элементов УПП. Результатом комбинации двух и более фильтров считается объединение соответствующих подмножеств элементов УПП.

Формальный вид запроса:

<имя фильтра> [<имя фильтра>] [<условие> [<логический бинарный оператор> <условие>]].

Синтаксис условий:

<параметр><оператор><значение>;

<параметр> – параметр фильтра, например, ParentClass – имя родительского класса, ContextClass – имя класса, в контексте которого находится фильтруемый узел УПП;

<значение> – значение, с которым сравнивается параметр. Может быть строкой символов или именем другого параметра;

<оператор> – оператор сравнения со значением параметра, например, равенство «=» или «!=» - неравенство;

<логический бинарный оператор> – бинарный оператор совмещения условий. Может быть: «AND» и «OR».

Также условия могут обособляться круглыми скобками. Для проработки возможных вариантов

использования инструмента был разработан тестовый набор фильтров и анализов, а также базовый набор параметров для задания срезов.

Так, фильтр o_modify_f позволяет получить выборку модифицирующих операций для заданного множества полей. Множество можно задать принадлежностью классу параметр SelfClass, а также через имя поля FieldName. Фильтр o_share_f позволяет найти места в исходном тексте, где поля объектов из заданного множества передаются по ссылке или через указатель. Фильтр ucr выдает диаграмму классов со всеми методами и полям. Задать срез такой диаграммы можно с помощью определения родительского класса ParentClass или самого класса через параметр ContextClass.

Множество вызовов в определенном контексте позволяет получить фильтр calls. Контекстом может быть имя вызываемого метода MethodName, контекстный метод ContextMethod, в котором производится вызов, а также класс определения метода SelfClass. Через фильтры cf и df с указанием желаемого контекста можно получить, соответственно, поток управления и поток данных. При этом доступна любая комбинация фильтров.

Конфигурационный файл программного инструмента

Существует в виде текстового XML-файла, формат которого позволяет задавать как линейные, так и нелинейные (со сложными зависимостями) условия фильтрации и трансформации УПП.

В корневом теге <analysises> перечисляются фильтры, каждый из которых задается иерархичным набором тегов из множества тегов УПП с указанием правил фильтрации в их атрибутах, и обособляется тегом <analysis> с атрибутом «name», задающим имя фильтра.

Перечень основных правил:

contexted (bool=true) — если true, узел фильтруется, если в УПП он представлен в контексте такого же родительского узла, как в фильтре; если false, то ведется рекурсивный поиск вниз по иерархии дерева документа;

strict (bool=false) – если true, то узел выводится, только если он и все дочерние узлы удовлетворили условиям фильтра; если false, то узел выводится, если он и хотя бы один дочерний узел (если такой есть) удовлетворили условиям фильтра;

all_in (bool=false) – если true, то, если данный узел не соответствует сложному условию, поиск в данной ветке дерева документа прекращается.

Конфигурационный файл [4] считывается при запуске. Проверка тега на соответствие правилам фильтра производится рекурсивным обходом иерархии узлов фильтра по иерархии узлов дерева XML-документа.

Тестирование

Для контроля правильности функционирования инструмента было произведено функциональное тестирование на УПП проекта с открытым исходным кодом pugixml [5] – библиотеке для разбора XML-документов. Набор тестов был получен выполнением множества пользовательских запросов, которое было сгенерировано путем комбинирования всех возможных различных пар из перечня доступных фильтров. Из тестового перечня фильтров было получено 136 эквивалентных представлений.

В соответствии с поставленными задачами были реализованы некоторые из анализов.

1) Поток управления для конкретного класса и метода. Распространенный вид анализа с дополнительной фильтрацией.

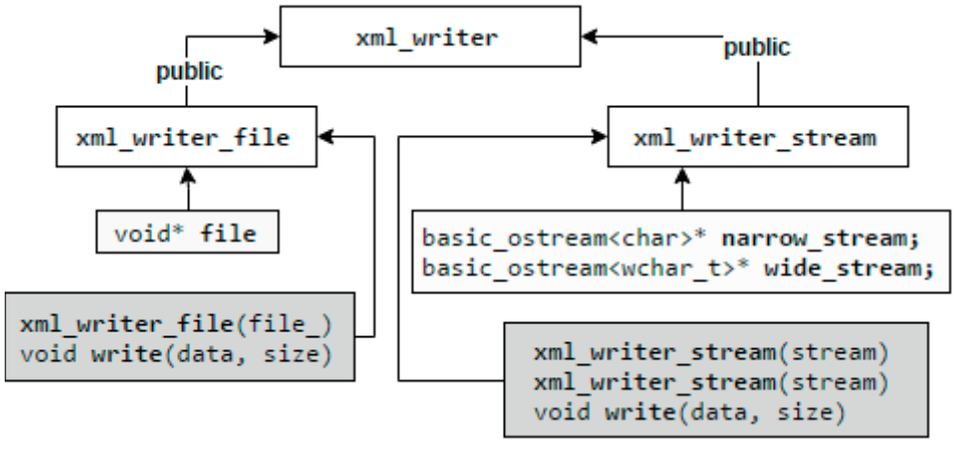

Рис. 1. Диаграмма классов-наследников класса xml_write

Командная строка:

cf ContextClass=xml_text AND ContextMethod =as_string

Результат в файле cf.xml [4].

2) Запрос комбинации потока управления и потока данных при тех же параметрах. Данная комбинация позволяет строить анализ потока данных в контексте потока управления [3].

Командная строка:

cf df ContextClass=xml_text AND ContextMethod=as_string

Результат в файле dcf.xml [4].

3) Диаграмма классов, в которую должны попасть только классы-наследники «xml_writer».

Командная строка:

ucr ParentClass=xml_writer

Результат в файле ucr.xml [4] (рис. 1).

4) Поиск модификаций и передачи по ссылке и указателю полей всех объектов заданного типа «xml_attribute_struct», учитывать только вхождения за пределами данного класса. Позволяет отследить явную и неявную модификацию членов класса. В отличие от обычного текстового поиска исключаются вхождения, не принадлежащие заданному классу, но совпадающие по именам; также исключаются вхождения, в которых модификации полей объекта не производятся (чтение или копирование). Результат фильтрации выдает вхождения для всех членов сразу, чего не позволяет получить обычный текстовый поиск. Данный фильтр полезен для отладки, задачи рефакторинга и выявления проблемных точек взаимодействия с классом через поля объектов.

Командная строка:

modify_f share_f SelfClass=xml_attribute_struct AND ContextClass!=SelfClass

Результат в файле modify.xml [4].

Строка исходного текста 3867 в файле «pugixml.cpp» [5] содержит вызов метода «strcpy_insitu», который определен в строке 1526. Первым и вторым аргументом вызова по ссылке передаются два поля («name» и «header») объекта «_attr» заданного типа «xml_attribute_struct».

Вызов метода:

strcpy_insitu(_attr->name, _attr->header, …);

Сам вызов происходит в контексте метода «set_name», принадлежащего классу «xml_attribute». Данный результат соответствует составленному запросу.

5) Поиск вызовов метода с именем «next_sibling» в контексте классов, не являющихся классом определения данного метода. Позволяет осуществлять контроль и рефакторинг точек вызовов заданного метода. Фильтр исключает вхождения, которые текстовый поиск не способен отфильтровать, когда классы не разбиты по отдельным файлам, а именно – вхождения вызовов метода в классе его определения.

Командная строка:

calls MethodName=next_sibling AND ContextClass!=SelfClass

Результат в файле calls.xml [4].

Строка исходного текста 5122 в файле «pugixml.cpp» [5] содержит вызов метода «next_sibling». Вызов находится внутри определения метода «reset», который принадлежит классу «xml_document».

Сам метод «next_sibling» определен в строке 4055 и принадлежит классу «xml_node» (см. рис. 2). Такой результат соответствует составленному запросу.

Температурная диаграмма исходного текста для любого эквивалентного представления

Температурной диаграммой исходного текста будем считать диаграмму зависимости условного количественного параметра (температуры) от номера строки исходного текста.

Пользователь задает вес для выбранных элементов УПП в соответствующем фильтре конфигурационного файла. В соответствии с весом фильтруемого элемента УПП для каждой строки вычисляется ее суммарная температура, которую формируют:

- температура основания – вес структуры AST, в которую вложен первый взвешенный элемент строки;
- собственная температура строки – суммарный вес всех входящих в нее взвешенных элементов.

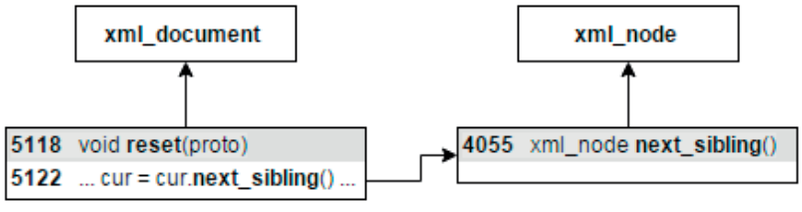

Рис. 2. Иллюстрация вызова метода за пределами его класса определения

Результат анализа записывается в XML-файл, формат которого представляет собой последовательность тегов <Line>, соответствующих строкам исходного текста с заданной в атрибуте «weight» температурой. Тела методов выделяются блоками <Method>, которые атрибутированы максимальной среди строк метода температурой.

Так как имеется возможность произвольного назначения весов каждому фильтруемому элементу УПП, а значит, и каждой синтаксической конструкции исходного текста, можно получить большое число разнообразных метрик исходного текста в виде температурных диаграмм.

«Температуризация» может быть использована для эквивалентных представлений, например для построения температурной карты диаграммы классов.

В соответствии с поставленными задачами был предложен «температурный» анализ синтаксической перегруженности строк и сложности блоков исходного текста в контексте потока управления. Может быть использован для полуавтоматического поиска участков кода, которые, вероятно, имеет смысл подвергнуть процессу рефакторинга. Могут быть выявлены потенциально «медленные» участки кода. Методика ранжирования весов тех или иных синтаксических конструкций лежит на операторе. Благодаря тому, что в анализ попадают и перегруженные операторы, которые по смыслу и весу равнозначны методам, оценка сложности исходного кода в данном случае будет более точно соответствовать коду исполняемому.

Командная строка:

cf_w CT ContextMethod=find_node.

Результат в файлах ct_ir.xml, ct.xml, ct.html [4].

Строка 514 исходного текста в файле «pugixml.hpp» [5] имеет «температуру» 16 (рис. 3), которая складывается из следующих составляющих:

- температура основания – 5, складывается из вложенности в цикл «while» (вес 3, 506 строка), в условный оператор «if» (вес 1, 510 строка) и еще один «if» (вес 1, 511 строка);

```
500(16):   template <typename Predicate> xml_node find_node(Predicate pred) const
502(4):   if (!_root) return xml_node();
504(2):   xml_node cur = first_child();
506(3):   while (cur._root && cur._root != _root)
508(5):    if (pred(cur)) return cur;
510(10):    if (cur.first_child()) cur = cur.first_child();
511(11):    else if (cur.next_sibling()) cur = cur.next_sibling();
514(16):     while (!cur.next_sibling() && cur._root != _root) cur = cur.parent();
516(10):     if (cur._root != _root) cur = cur.next_sibling();
520(3):   return xml_node();
```

Рис. 3. Температурная диаграмма фрагмента исходного текста

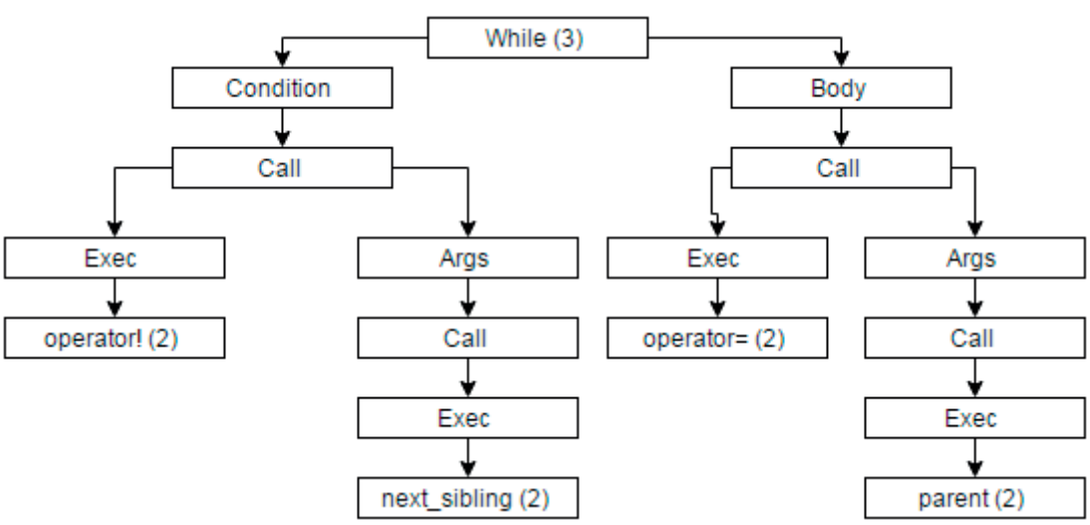

Рис. 4. Дерево разбора 514 строки исходного текста в контексте потока управления

- собственная температура строки – 11, складывается из весов конструкций: цикл «while» (3), вызов «next_sibling» (2), неявный вызов «operator!» (2), вызов «parent» (2), неявный вызов «operator=» (2).

Наличие операторов «operator!» и «operator=» обусловлено тем, что они перегружены для типа «xml_node», которому принадлежит объект «cur» (см. 504 строка).

Выводы

Рассмотренный инструмент позволяет получать линейные и нелинейные эквивалентные представления исходного текста из его УПП, выполнять анализы над этими представлениями и получать срезы этих представлений, а также получать объединения выбранных представлений с заданными общими параметрами среза. Таким образом, появляются поле для исследования ранее не изученных представлений исходных текстов, а также база для сведения к универсальному виду давно известных.

Литература

1. Инструменты статического анализа кода // URL: http://www.viva64.com/ru/t/0074/ (д.о. 20.06.2015).
2. Ошнуров Н.А., Пустыгин А.Н., Ковалевский А.А. Построение универсального промежуточного представления исходных текстов на языках C++ и C# // Доклады ТГУСУР. Т. 33, № 3, 2014. – С. 135-139.
3. Ковалевский А.А., Пустыгин А.Н., Ошнуров Н.А. Построение эквивалентного представления исходных текстов программ в форме, пригодной для выполнения анализов потока данных в потоке управления // Известия ЮЗГУ. Серия управление, вычислительная техника, информатика, медицинское приборостроение. №1 (14), 2015. – С. 28-34.
4. MultiCrossAnalyzer // URL: https://yadi.sk/d/a5GbG-1uhLbG2 (д.о.19.06.2015).
5. Pugixml v1.2 // URL: http://pugixml.org/2012/05/01/pugixml-1.2-release.html (д.о. 22.06.2015).

Получено 28.07.2015

Ковалевский Алексей Анатольевич, аспирант Кафедры компьютерной безопасности и прикладной алгебры (КБиПА) Челябинского государственного университета (ЧелГУ). Тел. 8-351-799-72-92. E-mail: morskoyzmey@gmail.com

Пустыгин Алексей Николаевич, к.т.н., доцент Кафедры КБ и ПА ЧелГУ. Тел. 8-351-799-72-92. E-mail: p2008an@rambler.ru

ALGORITHMS AND SOFTWARE TOOL FOR BUILDING EQUIVALENT REPRESENTATIONS OF SOURCE CODES

Kovalevskiy A.A., Pustygin A.N.
Chelyabinsk State University, Chelyabinsk, Russian Federation
E-mail: morskoyzmey@gmail.com

This article describes prototype of software tool for static analysis of software systems based on a special data set. This data set received from the source code of programs using the open source compiler. The prototype provides to obtain equivalent representation by linear and nonlinear transformations of this data set. The user interface is built by data query language, which syntax provides to set arbitrary combinations of available transformations, and additional parameters correspond to the objectives of the analysis. Conversion options are defined by configuration file being an XML-document. The prototype also provides analysis of the obtained equivalent representations. We descried «temperature» analysis of row syntactic overflow and complexity of source code blocks in the context of control flow by way of example. The prototype was tested on open source library for parsing of XML-documents.

Keywords: intermediate representation, source code, data flow, control flow, syntactic tree

DOI: 10.18469/ikt.2015.13.4.06

Kovalevskiy Aleksey Anatolevich, PhD-student, Department of the Computer Security and Applied Algebra, Chelyabinsk State University, Chelyabinsk, Russian Federation. Tel.: +73517997292. E-mail: morskoyzmey@gmail.com.

Pustygin Aleksey Nikolaevich, PhD in Technical Science, Department of the Computer Security and Applied Algebra, Chelyabinsk State University, Chelyabinsk, Russian Federation. Tel.: +73517997292. E-mail: p2008an@rambler.ru.

References

1. *Instrumenty staticheskogo analiza koda* [List of tools for static code analysis]. Available at: http://www.viva64.com/ru/t/0074/ (accessed 20.06.2015).
2. Oshnurov N.A., Pustygin A.N., Kovalevskiy A.A. Postroenie universal'nogo promezhutochnogo predstavlenija ishodnyh tekstov na jazykah C++ i C# [Universal intermediate representation construction of source code in C++ and C# languages]. *Doklady Tomskogo gosudarstvennogo universiteta sistem upravlenija i radiojelektroniki*, 2014, no. 3, pp. 135–139. Available at: http://www.tusur.ru/filearchive/reports-magazine/2014-33-3/23.pdf (accessed 22.06.2015).
3. Kovalevskiy A.A., Pustygin A.N., Oshnurov N.A. Postroenie jekvivalentnogo predstavlenija ishodnyh tekstov programm v forme, prigodnoj dlja vypolnenija analizov potoka dannyh v potoke upravlenija [Construction of equivalent representations of applications source code in form of data propagation in control flow graph]. Izvestija Jugo-Zapadnogo gosudarstvennogo universiteta. Serija: Upravlenie, Vychislitel'naja Tehnika, Informatika. Medicinskoe Priborostroenie, 2015, no. 1, pp. 28-34.
4. MultiCrossAnalyzer. Available at: https://yadi.sk/d/a5GbG-1uhLbG2 (accessed 19.06.2015).
5. pugixml v1.2. Available at: http://pugixml.org/2012/05/01/pugixml-1.2-release.html (accessed 22.06.2015).

Received 28.07.2015

УДК 004.93

МЕТОД ОБНАРУЖЕНИЯ ДВИЖУЩИХСЯ ОБЪЕКТОВ НА ОСНОВЕ АДАПТИВНОГО МЕТОДА ФИЛЬТРАЦИИ И ПРИМЕНЕНИЯ ТЕОРЕМЫ БАЙЕСА ДЛЯ ОЦЕНКИ ИЗМЕНЕНИЙ

Червяков Н.И., Тупикин А.А.
Северо-Кавказский федеральный университет, Ставрополь, РФ
E-mail: k-fmf-primath@stavsu.ru

Рассматривается новый метод обнаружения движущихся объектов в видеонаблюдении, который сочетает в себе адаптивный метод фильтрации с алгоритмом обнаружения изменений с применением теоремы Байеса. Сначала адаптивный алгоритм обнаруживает края движущихся объектов, затем алгоритм с использованием теоремы Байеса корректирует их форму. Рассматриваемый способ имеет значительную устойчивость к шуму, тени, изменению освещенности и повторным движениям в фоновом режиме по сравнению с другими алгоритмами. В данном алгоритме обнаружение движения выполняется в адаптивной схеме, и предварительная информация о переднем и заднем планах не требуется. В ходе работы показано, что предложенный алгоритм вычислительно эффективен и подходит для работы с интернет-видеонаблюдением и подобными приложениями.

Ключевые слова: обнаружение, видеоинформация, теорема Байеса, адаптивный фильтр.

Введение

В наше время, широко используются системы видеонаблюдения. Например, во многих приложениях, таких как мониторинг дорожного трафика, для обнаружения необычной активности в человеческой деятельности, подсчет количества объектов, и многое другое. Наиболее типичная схема системы видеонаблюдения включает в себя три следующие части.

1. Движущийся объект обнаружения.
2. Слежение за объектом.
3. Анализ его движения

Обнаружение областей, соответствующих подвижным объектам, является первым шагом обработки, поскольку остальные стадии обработки применяются уже локально к областям движущихся объектов. Это говорит об актуальности выявления движущихся объектов из потока видеоинформации для систем наблюдения.

Существует множество алгоритмов обнаружения движения в потоке видеоинформации. Самые простые используют операцию пороговой разности интенсивности, например, соседних видеокадров или текущих и фоновых. Чаще всего результат работы этих алгоритмов не подходит для дальнейшего анализа. Чтобы повысить производительность, другие алгоритмы используют вероятностные модели и статистические тесты, которые используются для моделирования и извлечения фона. Их эффективность в значительной степени зависит от выбора порога. Более высокая производительность может быть получена