УДК 004.9, 519.7

САМОАДАПТИРУЕМОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ АНАЛИЗА ИЗОБРАЖЕНИЙ

А.М. НЕДЗЬВЕДЬ, В.В.СТАРОВОЙТОВ

Объединенный институт проблем информатики НАН Беларуси
ул. Сурганова 6, Минск, Беларусь

Поступила в редакцию 21 декабря 2012

Описывается методика разработки интеллектуального программного обеспечения для анализа изображений путем формирования управляющих скриптов интерпретатора, которые генерируются интеллектуальным агентом в программном обеспечении.

Ключевые слова: программное обеспечение, анализ изображения, интерпретатор.

Введение

Современные требования к программному обеспечению анализа изображений требуют динамической организации функций, пользовательского интерфейса, управления данными для методов обработки и т.д. Исходная программа обрабатывается компилятором или интерпретатором. Иногда программа реализуется комбинированием этих вариантов. Такое программное обеспечение должно иметь ядро, включающее следующие компоненты: GUI (графический пользовательский интерфейс), основные функции и команды, различные типы представления данных. Интерпретатор используется для сохранения истории операций, которые применяются для обработки новых изображений одного класса. Он позволяет создавать дополнительные простые функции на базе возможностей ядра, не меняя программного обеспечения [1]. Мы предлагаем воспользоваться возможностями интерпретатора в качестве ядра для создания программного обеспечения. В этом случае интерпретатор работает как менеджер действий и событий. Он поддерживает выполнение функций и GUI события. Кроме того, можно изменить дизайн программного обеспечения и без этапа компиляции. С другой стороны, сложные функции и расчеты выполняются во внешних статически скомпилированных модулях. Это сохраняет скорость вычислений на том же уровне, что в скомпилированном программном обеспечении. Сегодня подобная организация архитектуры программного обеспечения используется для разработки сетевых систем, а также в играх и называется «открытой архитектурой программного обеспечения» [2].

Разработка любого приложения начинается с изучения объектов с точки зрения их происхождения, структуры, функций и направлений решаемой задачи (например, медицинских задач [3]). Практические решения с конкретным направлением приложения имеют ключевое значение в формировании программы и включают:

1) определение типа объектов и их основных классов;
2) анализ взаимодействий объектов, необходимых в формировании проблемы;
3) обнаружение сцен для создания общей картины задачи;
4) практический контекст сцены в определенном направлении решения задачи.

Действия, выполняемые в процессе анализа изображения, регулярны и зависят от оценки типов изображений и объектов на нем. Соответствие особенностей изображений и функций осуществляется благодаря тезаурусу как списку соответствий процессов обработки и правил анализа изображений.

В данной работе предлагается методика создания программного обеспечения на основе формирования таблиц тезауруса в ядре интерпретатора. Методика основана на комбинации возможностей динамических библиотек и интерпретатора с множеством функций для обработки изображений. В результате программа может быть разделена на две части: первая ориентирована на профессиональных разработчиков программного обеспечения, вторая – на работу пользователей. Интерпретатор может использовать функции динамических библиотек, что позволяет изменять свойства программного обеспечения без стадии компиляции. С другой стороны, пользователи могут изменять графический интерфейс программы [3].

Иерархия дескрипторов для анализа изображений

Набор дескрипторов для анализа изображений определяется современным состоянием теории анализа изображений, опытом решения прикладных задач, функциональными требованиями и спецификой лексического наполнения языка предметной области. Она включает следующие базовые элементы: тематические разделы (поля) дескрипторов, функциональные разряды (категории) дескрипторов, набор связей между дескрипторами, схему словарной статьи. Исходя из специфики предметной области «Обработка, анализ и понимание изображений», в набор дескрипторов для анализа изображений включено шесть категорий терминов. Их можно добавлять и модифицировать.

1. Категория «Объекты», в которую входят: наименования видов изображений («binary image», «raster image» и т.п.); наименования элементов изображений («edge», «pixel» и т.п.).

2. Категория «Задачи», в которую входят: наименования задач обработки изображений («image enhancement», «image restoration» и т.п.); наименования задач анализа изображений («binarisation», «segmentation» и т.п); наименования задач анализа изображений, в том числе наименования задач распознавания изображений («cluster analysis», «error estimation» и т.п.).

3. Категория «Подходы», в которую входят: наименования подходов к обработке изображений («wavelet-based image processing» и т.п.); наименования подходов к анализу изображений («model-based image analysis» и т.п.); наименования подходов к анализу изображений («statistical pattern recognition» / статистическое распознавание образов и т.п.).

4. Категория «Методы», в которую входят: наименования методов (алгоритмов) обработки изображений («fractal image compression», «equal interval quantization» и т.п.); наименования методов анализа изображений («structural texture description» и т.п.); наименования методов анализа изображений («image algebra-based technique» и т.п.).

5. Категория «Инструменты», в которую входят: наименования инструментов (операторов, преобразований, фильтров) обработки изображений («median filter», «discrete Fourier transform» и т.п.); наименования инструментов анализа изображений («Prewitt edge detector», «Sobel edge detector» и т.п.); наименования инструментов анализа изображений, в том числе наименования инструментов распознавания изображений (например, «maximum likelihood decision rule», «cluster assignment function» и т.п.).

6. Категория «Характеристики», в которую входят: наименования характеристик инструментов («threshold», «convolution kernel» и т.п.); наименования характеристик (элементов описания) изображений («brightness», «color model», «contrast difference» и т.п.).

Для процессов обработки изображений необходимо сформировать связи дескрипторов, характерные для рассматриваемой предметной области:

– «задача – метод» (например, «image segmentation – edge-based image segmentation»);

– «метод – инструмент» (например, «gradient-based edge detection – Sobel edge detector»);

– «объект (вид изображения) – метод» (например, «binary image – binary noise reduction»);

– «инструмент – результат» (например, «edge detection operator – edge image»);

– «инструмент – характеристика инструмента» (например, «image opening operation – structuring element»).

Пример анализа изображений гистологических препаратов

Обработку медицинских изображений можно разделить на несколько этапов.

1. Ввод и улучшение изображений.
2. Сегментация на информативные области.
3. Выделение объектов интереса.
4. Измерение объектов, вычисление их характеристик.
5. Анализ объектов.

Каждый этап состоит из выполнения разных последовательностей функций. Их применение зависит от свойств изображения и его оценок: например, оценок контраста, шума или размытия. Можно построить таблицу функций обработки и оценок свойств изображений.

Для гистологического изображения выбор методов сегментации зависит от ряда условий, а сегментация (определение однородных областей) рассматривается как важный шаг для формального описания сцены. Искомые гистологические объекты определяются в соответствии с задачами, которые предстоит решить. Автоматизация анализа гистологических препаратов может быть выполнена на основе топологических особенностей изображения, а результат зависит от оптического увеличения изображения. На разных уровнях увеличения существует определенная группа топологических особенностей ткани и ее компонентов, поэтому на разных уровнях увеличения процедура обработки может состоять из разных функций с разными параметрами.

На рис. 1 представлена общая схема иерархического анализа объектов для гистологических изображений. Фрагменты тканей состоят из групп клеток и волокон. Эти объекты представлены разными текстурами. Для их выделения часто используют алгоритмы наращивания областей.

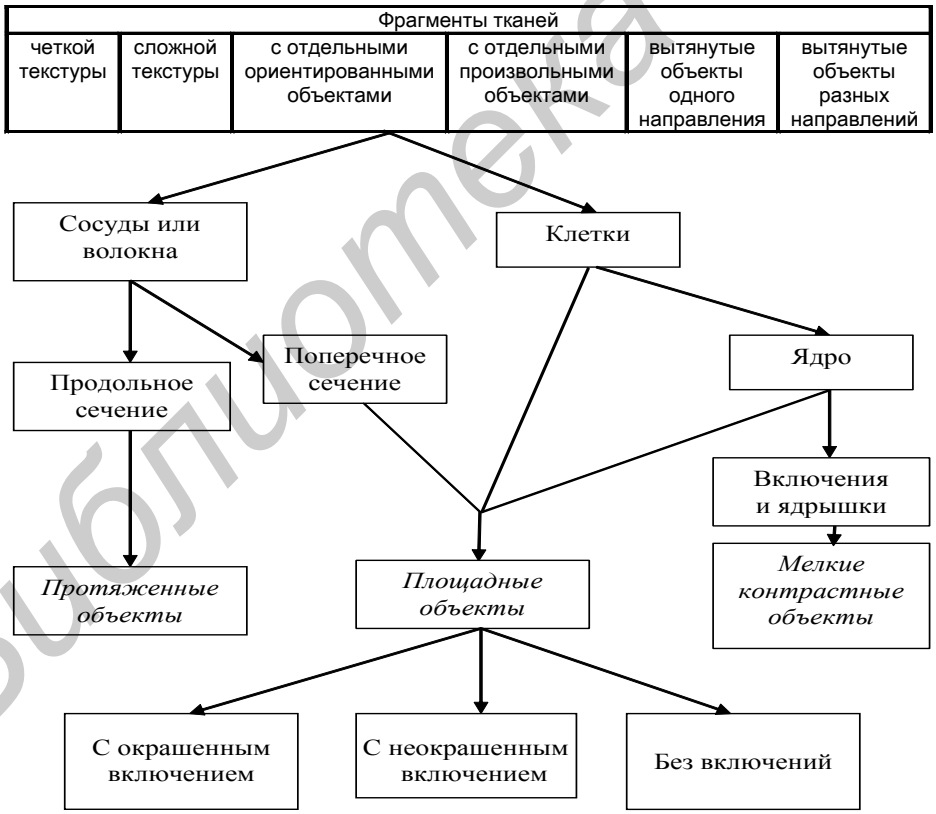

Рис. 1. Иерархическая структура гистологической ткани

Оценка особенностей изображения позволяет определить функции необходимые для его обработки. В результате строится таблица связей функций и оценок характеристик изображения (см. рис. 2).

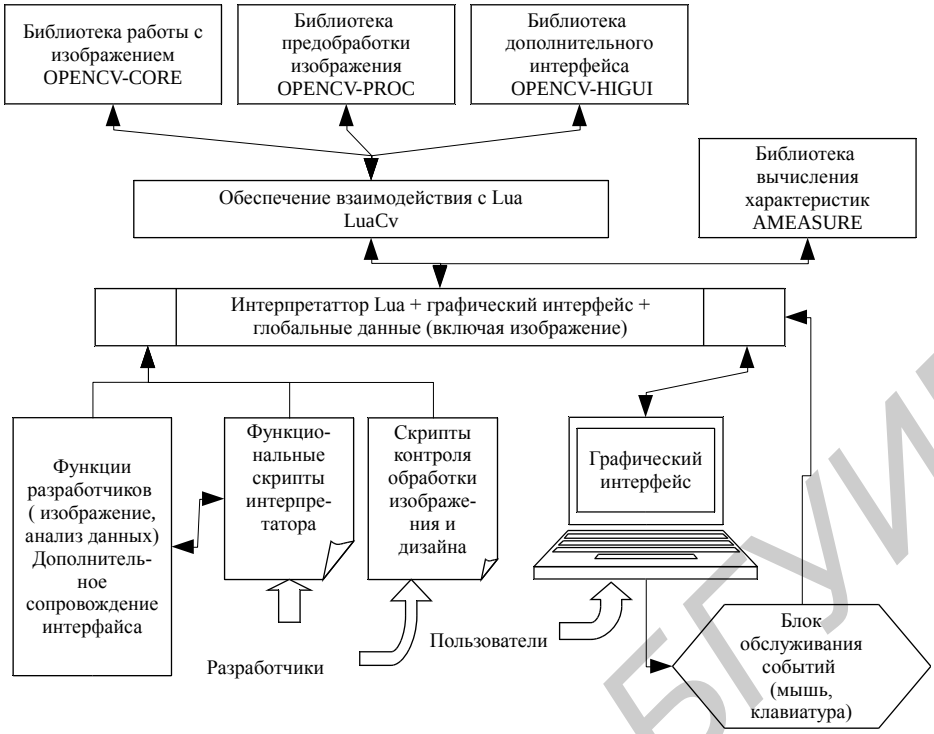

Рис. 2. Общая схема иерархического анализа гистологических изображений

На каждом шаге для функций определены приоритеты их выполнения. Например, для улучшения изображения – удаление импульсных шумов, следующий уровень приоритета будут иметь функции улучшения контраста и коррекции границы объекта. Приоритеты определяют порядок выполнения функций и необходимость дополнительного анализа особенностей изображения.

Адаптация программного обеспечения под практические задачи

Для разработки структурной схемы программного интерфейса и архитектуры была выполнена оценка функциональности и совместимости имеющихся в распоряжении инструментов разработки программного обеспечения.

Функции обработки изображения относятся к основным разделам компьютерного зрения. Это соответствует их применению для изменения изображения. Каждая функция меняет свойства изображения и применяется для конкретных случаев обработки. Свойства и особенности функции должны описываться в глобальной таблице интерпретатора и сопровождаться дополнительной информацией.

На базе материалов, размещенных на сайте Гийома Марсо [1], который исследовал 72 параметра для оценки реализаций языков программирования и сравнил их на основе 19 специальных тестов, в качестве ядра выбран интерпретатор Lua [6].

Наша система основана на интерпретаторе языка Lua. Он используется в качестве основного модуля и обеспечивает взаимодействие сложных компонентов. В ядро интерпретатор дополнительно включены: графический интерфейс, глобальные переменные и структуры изображения. Архитектура графического интерфейса базируется на объединении библиотек Highgui [7] из пакета OpenCV [7] и Qt. Функции обработки и анализа изображений поддерживаются благодаря связи библиотеки OpenCV с Lua, которые реализуются в отдельной библиотеке.

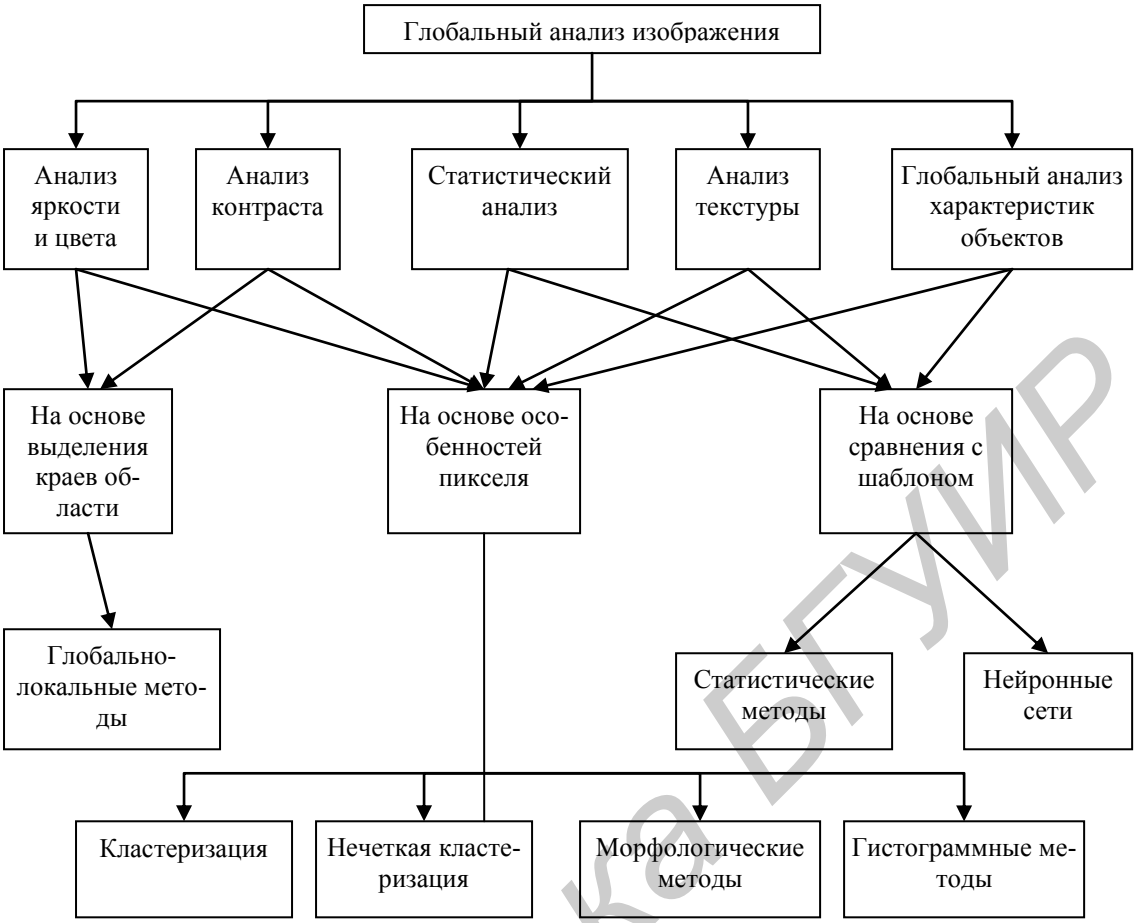

Рис. 3. Схема интеллектуального программного обеспечения с возможностями самомодификации

Структуры изображения определяются в модуле графического интерфейса на основе библиотеки OpenCV, которая отвечает за визуализацию и представления изображений. Заголовки структуры изображения определяются в качестве глобальных указателей интерпретатора Lua и имеют специальный тип – для пользовательских данных. Этот тип (Userdata) соответствует указателям адресного пространства компьютера. Этот модуль также включает в себя функции чтения/записи изображения, простые функции обработки изображений и выделения интерактивного контура. Все интерактивные функции возвращают значения в отдельную таблицу данных интерпретатора, меняя его состояние. Для задач мониторинга различных заболеваний возможно синхронное использование нескольких модулей. В этом случае взаимодействие выполняется с помощью глобальных переменных интерпретатора Lua и свойств пользовательских данных типа интерпретатора.

Во время обработки изображения ядром Lua выполняются базовые наборы последовательностей команд. Однако во время обработки команды в сценарии могут быть изменены, что влияет на результат анализа изображения. Таким образом, весь процесс обработки и анализа изображений формируется несколькими наборами скриптов, связными узловыми точками, обеспечивающими получение результатов, зависимых от частных условий задачи. Связь между узлами в том же сценарии осуществляется через прямые вызовы функций по событиям-слотам, либо с помощью виртуального взрыва метода для его первого входа. Добавление новых сложных функций для обработки и анализа осуществляется через группу функций разработчика. Новые функции определяется через переменные глобальной таблицы интерпретатора Lua.

Все вопросы внутреннего контроля программного обеспечения осуществляются посредством текстовых сценариев Lua, которые делятся на две категории:

1) сценарии для анализа последовательностей изображений;

2) сценарии для обеспечения оперативной функциональности и настройки программного обеспечения на рабочем месте пользователей.

Все сценарии-скрипты хранятся в текстовом формате и легко доступны. Однако они не могут быть изменены пользователями, но могут быть изменены разработчиками. Сценарии управления программным обеспечением имеют возможность создания новых дополнительных функций для анализа и обработки изображений.

Модуль генерации скрипта определяет новые возможности программы в виде набора простых функций из библиотек, описанных выше. Он включает в себя интеллектуальный компонент для связи результатов обработки изображений и характеристик. Конечно, такой вариант может быть реализован только для конкретных задач. В нашем случае задачи были ориентированы на гистологический анализ изображений. Каждый интерпретатор определяет функции по определенной таблице. В нашей схеме она дополнительно используется для определения связи характеристик изображения с функциями программы, включенными в управляющий скрипт. В результате программное обеспечение приобретает интеллектуальные возможности самопрограммирования.

Модуль генерации скрипта состоит из двух частей: оценки-анализа изображений и непосредственно генерации. Определение структуры скрипта начинается с глобального анализа изображений, который включает определение ряда характеристик: описание гистограммы и основные характеристики статистического анализа распределения пикселей, фрактальную оценку и описание текстуры. На основе таких оценок формируются необходимые последовательности функций для обработки изображений. Например, по оценкам, полученным во время анализа шума и размытия на изображении, определяются необходимые наборы фильтров влияющих на общее качество изображения. После сопоставления оценок с расширенной глобальной таблицей интерпретатора формируется сценарий предварительной обработки изображения для повышения качества изображения.

Локальный анализ изображения выполняется с помощью свертки с разными фильтрами и статистического анализа линейных профилей фрагментов изображения. Такой анализ позволяет оценить характерные границы клеток и контрастности. Такая оценка определяет функции для контрастирования изображений и подчеркивание границ. После того как изменится изображение, результат снова оценивается. Если изображение имеет низкое качество после анализа изображений и определения корректирующей функции, эту процедуру следует повторить. В результате формируется устойчивый скрипт, который может быть изменен пользователем или разработчиком в случае необходимости. Этот сценарий является всего лишь предложением для улучшения изображения и определения пользовательских элементов управления в интерфейсе программы. Механизм анализа изображения и функция определения этапов работ по сегментации изображений и пост-обработки те же.

В результате, программа генерирует наборы скриптов, которые соответствуют функции для выделения нужных объектов. Он может быть использован для определения гистологических объектов на изображении. В нашей программе были разработаны сценарии выделения ядер и мембран на гистологических изображениях. Затем рассчитываются характеристики объектов. Это необходимо для выявления типа объектов, которые представлены на изображении. Первоначально объекты делятся на пять основных типов структур: площадные, фронт, иглы, дендриты и сети. Такая процедура обнаружения объектов начинает выполнения каскада операций по выделению объектов. Направление и организация такого каскада зависят от типа решаемых задач. Использование глобальных фрактальных и текстурных характеристик является первым шагом обнаружения геометрического типа и формы описания объекта. Затем в каскаде используются разные характеристики и их комбинации.

Для определения функций обработки изображений в сценарии использовались нейронные сети Кохонена. Оценки глобальных условий изображения используются в качестве весов в нейронной сети. Регулировка входных весов и вектор квантования сигнала сети тесно связаны с алгоритмом кластерного анализа (например, методом динамических ядер или K-средних).

Модуль генерации скриптов использует Lua-метатаблицы функций для обработки изображений дополненной оценками изображения. Такая таблица определения функции реализуются в нейронные сети Кохонена как наборы весов. В результате работы модуля формируются сценарии для различных задач обработки изображений в виде текстовых файлов. Пользователи могут проводить анализ этих сценариев и изменить в них.

Заключение

Использование принципов открытой архитектуры и искусственного интеллекта позволяет облегчить разработку программного обеспечения. Пакеты, разработанные таким образом, проходят две стадии разработки: компилируемая стадия и стадия интерпретатора. В результате цикл жизни и роста программного обеспечения значительно продлевается и значительно улучшается динамичность программы. Используя компоненты интерпретатора, программу можно легко модифицировать и адаптировать к решению новых классов задач. Использование интеллектуального агента снабженного соответствующим механизмом связей дескрипторов и функций в этой схеме позволяет программному обеспечению модифицироваться самостоятельно или с помощью пользователя. Эта методика была успешно опробована для анализа гистологических изображений.

SELF-ADAPTABILITY SOFTWARE FOR IMAGE ANALYSIS

A.M. NEDZVED, V.V. STAROVOYTOV

Abstract

The method for development of intelligent software for image analysis is described. It is based on control shell of scripts generating that are controlled by an intelligent agent in software. The method is based on a combination of features of dynamic libraries and interpreter with a lot of functions for image processing.

Список литературы

1. Square root of x divided by zero. [Электронный ресурс]. – Режим доступа: http://gmarceau.qc.ca/blog/2009/05/speed-size-and-dependability-of.html Saturday. – Дата доступа: 30.05.2009.
2. *Reitmayr G., Schmalstieg D.* // Processing of VRST'01. Alberta, Canada, 15–17 November, 2001. P. 47–54.
3. *Nedzved A., Belotserkovsky A., Lehmann T.M. et. al* // Processing 5th Int. Conference on Biomedical Engineering. Innsbruck, 14–16 February, 2007. P. 379–384.
4. *Kanade T., Yin Z., Bise R. et. al.* // Proc. of WACV11. 2011. P. 374–381.
5. *He L., Long R., Antani S. et. al.* // Sequence and Genome Analysis: Methods and Applications, iConcept Press. 2011. P. 271–287.
6. *Ierusalimschy R.* Programming in Lua. 2006.
7. *Bradski G., Kaehler A.* Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly, 2008.