

УДК 519.6:[004.05.:[004.514:004.93]]

А.В. БОРОВИНСКИЙ, В.О. МИЩЕНКО

Харьковский национальный университет им. В.Н. Каразина, Украина

СЛОЖНОСТЬ РЕАЛИЗАЦИИ ИНТЕЛЛЕКТУАЛЬНЫХ ГРАФИЧЕСКИХ ИНТЕРФЕЙСОВ ДЛЯ ПРИЛОЖЕНИЙ, ОСНОВАННЫХ НА МДО

Рассматриваются итоги разработки приложения, которое обеспечивает для программных систем моделирования дифракционных явлений методами дискретных особенностей ввод данных в форме эскизов с границами областей рассеяния волн и надписями, которые могут готовиться с помощью произвольного графического редактора. В статье предлагается метод, основанный на сочетании содержательного и метрического анализа вопроса: насколько, сложны, рискованны и дороги подобные интерфейсы, причём используются энергетические метрики. Обосновывается тезис о том, что интеллектуальные графические интерфейсы оправданы тогда и только тогда, когда могут быть многократно использованы в системах компьютерного моделирования без сложных модификаций.

Ключевые слова: интеллектуальный графический интерфейс, качество, риски, стоимость, метрики.

Введение

Работа вызвана к жизни актуальностью задач программной реализации вычислительных методов, известных как методы дискретных особенностей (МДО), которые, в частности, эффективны в задачах вычислительной электродинамики [1]. За последнее время с использованием МДО решены новые классы задач рассеяния волн и пучков на акустических и электродинамических устройствах в 2 и 3-мерных постановках задач, причём с приемлемой точностью уже при невысоких порядках дискретизации (см., напр., [2-4]). Во многих случаях эти постановки требуют ввода данных, определяющих двумерные области сложной формы (см., напр., [5-7]). Причём важно иметь возможность производить не только компьютерное моделирование для устройств известной формы, но и поиск новых оригинальных форм путём их подбора. Целесообразно, чтобы интерфейс реализующих программ предусматривал возможность ввода данных о таких формах с рисунка, а не требовал конструирования в специальном редакторе на основе примитивов (как, напр., в [8, 9]). Это требует применения подходов искусственного интеллекта. Краткое описание функциональности, методов и результатов реализации такой системы ввода-вывода, ориентированной на поддержку компьютерного моделирования МДО дано в [10].

Интеллектуализация (в трактовке, близкой к [11]) ввода-вывода графической информации требуется и в других случаях наукоёмкого компьютерного моделирования. Например, в тех же задачах дифракции волн на электродинамических уст-

ройствах бывает очень нетривиален ввод данных об источниках падающего поля, и его нужно упрощать для пользователя, возлагая на систему моделирования как бы «интерполяцию» требований. Однако возникает вопрос о стоимости и рисках создания таких интерфейсов. Поэтому имеющийся опыт требует анализа, объективность которого нужно подкреплять программометрической поддержкой.

Целью работы являлось охарактеризовать сложность программной реализации интеллектуальной подсистемы ввода геометрических и физических данных в системах компьютерного моделирования МДО.

1. Постановка задачи

Для достижения этой цели был предпринят обобщающий анализ выполненной разработки приложения, предназначенного для автоматизации подготовки данных для расчетных модулей МДО на основе графических эскизов. Это приложение воспринимает на входе графический файл чертежа, который удовлетворяет мало ограничительным правилам, а выходом имеет файл, в котором сохранена информация о распознанной на эскизе системе открытых и замкнутых контуров и числовых величин, записанных возле контуров. Этот формат файла согласован со входом расчётных блоков систем 2D и 3D моделирования МДО, таких как в [5 – 7]. Он согласован также со входом простой подсистемы визуализации данных, что позволяет обеспечить контроль постановки задачи человеком перед началом численного моделирования.

Обсуждаемое интеллектуальное приложение, реализующее специальный графический интерфейс МДО, разрабатывалось на языке программирования Ада. Это объясняется тем, что на данном языке ведётся разработка ряда программных систем МДО, которые имеют перспективу длительного сопровождения. Также имелось в виду, что гибкая модульность этого языка существенно упрощает разработку сложных приложений, а также имеются встроенные средства распараллеливания вычислений, что удобно на многоядерных компьютерах и весьма уместно по характеру многих алгоритмов обработки графических файлов. Для поддержки экранной графики использована GtkAda - портация под Аду общеизвестной кроссплатформенной библиотеки Gtk, так что приложение можно собрать и под Windows, и под Linux. В связи с этим подчеркнём, что для Ада программ решены все принципиальные и технические вопросы оценивания энергетических метрик [12].

В то же время разработка метода анализа эскизов сталкивалась с рядом проблем математики распознавания образов в конкретной постановке и проблемой адекватности интерфейса с пользователем. Возникшие вопросы можно сгруппировать так:

- какие требования к подготовленному пользователем эскизу достаточны для правильности автоматической идентификации и интерпретации графических элементов этого эскиза?
- если такие требования логически достаточны, но минимальны, то насколько сложно и нужно ли реализовать гарантированное распознавание любого эскиза, который этим требованиям удовлетворяет?
- на каком языке пользователь может формулировать собственные претензии к качеству автоматического распознавания эскиза?

Далее мы сперва содержательно анализируем опыт разработки приложения Intellectual_I. Затем проводим метрический анализ исходных текстов в качестве формальной системы, отражающей сложность и риски проведенной работы.

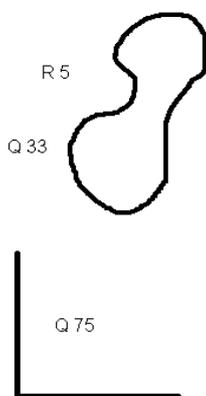


Рис. 1. Пример вводимого эскиза

Отметим, что с момента публикации сведений о первой рабочей версии Intellectual_I [10] удалось усовершенствовать функцию поддержки обработки ошибок, усовершенствован диалог с пользователем, который страхует правильность автоматической интерпретации эскиза, практически сняты ограничения на класс вводимых гладких кривых.

2. Метод работы

В основе суждения о простоте или сложности разработанной и реализованной программной системы должна находиться хотя бы краткая информация о проблемах, найденных или требуемых решениях, о поисках, ошибках и успехах. Поэтому мы делаем обзор таких проблем, связанных с Intellectual_I. На его основе читатель, вероятно, согласится с тем, что данная реализация интеллектуализированного графического интерфейса и любая подобная ей является самостоятельной задачей, которая требует проведения определённых исследований и затрат на кодирование, измеряемых, скорее, не месяцами, а годами. Однако по обзору нельзя ответить, например, на вопрос о том, является ли создание такого приложения более простой или более сложной задачей, чем разработка самих приложений МДО, для которых оно предназначено? Поэтому далее мы обращаемся к результатам метрической оценки кода, которые, принимая определённые гипотезы, можно считать количественным уточнением рисков и затрат.

3. Содержательная сложность

Перечислим факторы, которые определяют повышенную сложности реализации естественной схемы распознавания эскизов (кластеризация элементов матрицы изображения, идентификация символов, кодирование кривых, идентификация надписей [10]).

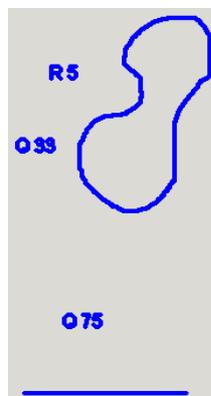


Рис. 2. Результат простого утоньшения линий

Сложность подсистемы скелетизации изображения. Ради удобства пользователей толщина линий изображения кривых на эскизах формально не ограничивается. Поэтому на предварительном этапе распознавания контуров использован алгоритм утоньшения линий Зонга-Суня [13] (реализация в форме Ада пакета *Bmp_File*, операция *Thin_B*), который действительно обеспечивает однопиксельность линий, анализируемых на следующих этапах. Однако этот (а в силу природы задачи и любой другой подобный) алгоритм является эвристическим. В частности, при некоторых обстоятельствах он может «ужать» строго вертикальную или строго горизонтальную линию в точку. В *Intellectual_I* проблема решена за счёт индикации таких эффектов и повторного применения алгоритма Зонга-Суня после поворота эскиза на случайный угол (отредактированное приложение демонстрируется пользователю для контроля после обратного поворота). В программе за это «отвечают» операция *Rotate_Image* и ресурсы пакета *Bmp_File*. Пример см. на рис. 1 - вход, рис. 2 – неудача, рис. 3 – удачная обработка, 4 - выход.

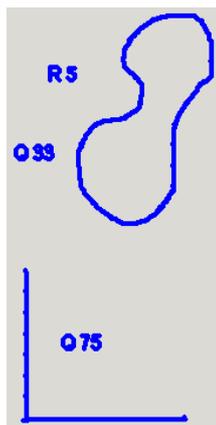


Рис. 3. Результат поворота, утоньшения и обратного поворота

Наконец надписи нужно ассоциировать с кривыми и областям, которые они ограничивают (если ограничивают). Критерием служит близость в смысле расстояния между кривыми и центрами тяжести надписей (метод *Create_Input_Objects*).

Идентификация надписей. Она, ради упрощения, основана на идентификации отдельных графических символов в предположении, что надписи делаются стандартными шрифтами, но от надписи к надписи можно, в принципе, варьировать размер и даже вид шрифта.

На основе анализа таких требований создан не требовательный к ресурсам детерминированный однопроходный алгоритм. Он воплощает известный принцип сравнения с образцами [11]. Для пользова-

Множественность и разнородность задач этапа кластеризация изображения. При группировке элементов изображения выделяются связанные подмножества элементов матрицы пикселей изображения (считая соседями пикселя 8 прилегающих к нему пикселей). Это рекурсивный алгоритм:

- пакет *Coherent_Components*,
- операция *Get_Coherent_Components*.

Каждое выделенное подмножество относится к одному из классов: надписи или кривые. Их различие основано на требовании, чтобы диаметры кривых значительно превышали диаметры всякого элемента-надписи. Это обеспечивает существование скачка диаметров, что позволяет основать критерий различия на сравнении диаметров изображений, определяемых подмножествами пикселей: операция («метод») *Coherent_Components.Define_Curves_Notations*.

Затем требуется выявить надписи по признаку расположения символов рядом друг с другом (метод *Coherent_Components.Create_Legends_Clusters*, используется мера близости, определённая в [10]).



Рис. 4. Демонстрация результатов распознавания для контроля

телей, которым недостаточно шрифта умолчания (*Times New Roman*) и размеров умолчания (12, 14), реализована операция обучения другим шрифтам и размерам (пакет *Font_Recognition*, операция *Get_Symbol*). В надписи различаются буквенные имена и числа (целые или десятичные дроби). Транслятор надписей в атрибуты кривых и ограничиваемых ими областей требует конкретных таблиц, определяемых приложением компьютерного моделирования, но для приложений [5-7] они унифицированы.

Идентификация кривых. Обнаруженные кривые различаются как замкнутые и открытые. При этом принимается во внимание возможность случайных разрывов связности после утоньшения ли-

ний. Многосвязные области и самопересечения кривых формально запрещены (*Coherent_Components*).

Сплайн-интерполяция кривых (тот же пакет). Приложения МДО обычно требуют «передачи» им кривых в форме передачи параметров сплайнов.

Контроль интерпретации распознанных кривых. Демонстрировать пользователю картинки до и после кластеризации легко (пакет *Step_Half*, операции *Run_Preview*, *Run*). Но пришлось разработать язык рекламаций и методы корректировки результата ввода графического ввода (средства пакета *Content*).

Устойчивость при сбоях. Имеется перечень требований к эскизам (монохромность и абсолютная контрастность изображения, запрет на самопересечение кривых и т. п.). Они простые и мало ограничительные, достаточны для корректной работы программы. Однако не все сбои, возможные при нарушениях, легко предупредить, отследить или обработать как исключения. Работа в этом направлении далеко не окончена.

Перейдём к результатам метрической оценки программной системы. Подчеркнем, что разделы 3, 4 готовились разными авторами статьи независимо.

4. Энергетический анализ

Энергетические метрики разрабатывались на основе четверть векового опыта использования в исследованиях и индустрии знаменитых научных метрик М. Холстеда (о которых см., напр., в [14]). Он создавал их, в основном, на базе статистики и наблюдений за разработкой простых, однофайловых программных модулей, чьи информационные связи с остальной частью программы не учитывались. Энергетические метрики, разработанные на основе прототипов из числа научных метрик, подобных ограничений не имеют и используют свои методы для прогноза качества программных систем [12]. Поясним трактовку этих метрик:

B – прогнозируемый уровень риска, связанного с ошибками кодирования (в предельном случае простых программ – норма числа таких ошибок);

D – трудность кодирования (определённый опыт [14] позволяет предполагать трудность порядка 10 незначительной, 100 – существенной, порядка 1000 – угрожающей успеху отладки и модификации);

A – работа программирования, равная $B^2 \cdot D$ с точностью до постоянного множителя (связывается с затратами времени разработки, но только в смысле идеализированного процесса, в котором, в частности, не допускаются неточности и поиск вариантов [12]);

E – спецификационная энергия (определяется архитектурой и интерфейсами программной системы, нормируется на λ – уровень языка по Холстеду).

Величины **A** и **E** в процессе разработки имеют тенденцию к взаимному притяжению, и для готовой продукции часто примерно равны между собой хотя бы по порядку величин.

Однако для модельных реализаций с непроработанной функциональностью энергия **E** значительно превышает **A**, если же ведётся долгая доработка функциональности или возникли трудности, которые приходилось преодолевать за счёт многих исправлений и добавлений, то наверняка работа **A** будет значительно больше **E**.

В табл. 1 приводится оценка в целом по системе и по 5 наиболее крупным пакетам (из 13). Отметим, что в языке Ада пакет представляется компонентами спецификации и реализации. И для всего приложения, и для его компонент работа **A** на 2 порядка превосходит энергию **E**, причём это наблюдалось и в предыдущих версиях. Поэтому мы должны заключить, что реализация *Intellectual_I* оказалась трудной задачей в сравнении с тем, что можно ожидать от приложений с примерно такой же структурой программы.

Это согласуется с тезисом предыдущего раздела о том, что распознавание эскизов по прозрачной на вид схеме на самом деле является содержательно сложной задачей.

Таблица 1

Фрагмент результатов оценивания приложения *Intellectual_I*
(1 КХд = 1 символ·бит; $\lambda=1,5$)

Модули	B	D	A, КХд	E, КХд
все	79,8	52,4	135	0,75
coherent_components	18,4	89,9	35,3	0,05
content	13,2	558	21,5	0,04
create_file_selection	5,0	630	9,2	0,00
show_message	3,6	941	9,2	0,0
step_half	18,1	978	52,2	0,02

Отметим, что в табл. 1 в колонке “D” для системы в целом приводится *среднее значение* трудности D по всем 27 компилируемым модулям, а для отдельных пакетов – *максимальное значение* из значений для компонент спецификации и реализации.

Обратимся к анализу подсистем, в роли которых выступают библиотечные пакеты Ада программы. Проблемы, о которых шла речь в предыдущем разделе, связаны с явно упоминаемыми там 5-ю пакетами: *Bmp_File*, *Coherent_Components*, *Font_Recognition*, *Step_Half*, *Content*. Мы видим, что 3 из них попадают в 5-ку наиболее крупных в составе всей программной системы. Они составляют там первые тройки самых трудозатратных (по величине А) и самых проблемных по риску кодирования (по индексу В) пакетов, а два из них – в четвёрке самых трудных (по метрике D). Поэтому заключаем, что трудность анализируемой задачи программирования действительно в основном связана с проблемами, описанными во 2-м разделе.

Согласие между выводами на основе содержательного анализа и на основе метрического оценивания означает валидность энергетических метрик в данном проекте. Следовательно, разумно использовать числовые энергетические оценки как основу суждений и прогнозов об этом проекте. Как пример, сравним трудоёмкости реализации интеллектуального интерфейса и наиболее сложного на сегодня приложения МДО из тех, которые этот интерфейс используют [7] – *Diffraction_On_RE_Polygon*. По табл. 2 статьи [7] для него А = 26.4 КХд, что почти вчетверо меньше оценки приложения *Intellectual_I*. При этом спецификационная энергия данного приложения МДО в 35 раз больше энергии *Intellectual_I*. Т. е., по постановке программистской задачи приложение МДО в 35 раз сложнее, чем приложение, обеспечивающее ему удобный интерфейс, но в реализации интерфейс оказался в 4 раза сложнее. Поэтому интеллектуальный интерфейс должен быть ориентирован на крупные группы приложений МДО и так, чтобы быть к ним легко адаптируемым.

Выводы

Теоретическое значение данной работы состоит в том, что дан и испытан метод комплексной оценки важных факторов качества программных систем. Его особенность – сочетание не подробного, но содержательного рассмотрения проекта с использованием энергетических метрик (затраты на которые пренебрежимо малы). Практический результат состоит в получении подкреплённых метрическими показателями суждений о трудозатратах разработки интеллектуального графического интерфейса МДО, оценён уровень рисков. В дальнейшем важно изучить возможности энергетических метрик по про-

гнозу сложности и повышению эффективности сопровождения подобных интерфейсных приложений.

Литература

1. Гандель, Ю.В. Парные и гиперсингулярные интегральные уравнения задач дифракции электромагнитных волн на плоских решетках и экранах [Текст] / Ю.В. Гандель // Труды XI Международного симпозиума МДОЗМФ. – 2003. – Харьков-Херсон. – С. 53 – 58.
2. Антонец, А.В. Численный анализ гиперсингулярного интегрального уравнения задач дифракции на плоском экране [Текст] / А.В. Антонец, Ю.В. Гандель // Вісник Харківського національного університету. Серія «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління». Вип. 1. – № 590. – 2003. – С. 9 – 14.
3. Гандель, Ю.В. Гиперсингулярное интегральное уравнение математической модели гиротрона для случая ТМ-волн [Текст] / Ю.В. Гандель, А.С. Кононенко // Вестник Харьковского национального университета. Серія "Математическое моделирование. Информационные технологии. Автоматизированные системы управления". – 2005. – №661. – С. 83 – 88.
4. Гандель, Ю.В. Краевые задачи для уравнений Гельмгольца и Максвелла на многоцелевых цилиндрических структурах и граничные интегральные уравнения на системе отрезков [Текст] / Ю.В. Гандель, С.В. Духопельников // Крайові задачі для диференціальних рівнянь: Зб. наук. праць. – Чернівці: Прут. – Вип. 16. – 2008. – С. 264 – 293.
5. Гахов, А.В. Трёхмерная модель метода дискретных особенностей рассеяния скалярных волн экраном на границе раздела сред [Текст] / А.В. Гахов, В.О. Мищенко // Вестник ХНТУ. – № 2 (25). – Херсон. – 2006. – С. 135 – 140.
6. Мищенко, В.О. Построение программных систем моделирования дифракции на идеально проводящих экранах, лежащих в диэлектрическом полупространстве [Текст] / В.О. Мищенко // Вісник Харк. нац. ун-ту., Сер. «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління», Вип. 10. – 2008. – № 833. – С. 170 – 184.
7. Боровинский, А.В. Компьютерное моделирование 3D дифракции в плоскопараллельной среде: опыт реализации МДО методом расширенных программ [Текст] / А.В. Боровинский, А.В. Гахов, В.О. Мищенко // Вестник. – 2009. – С. 21 – 35.
8. ЭДЭМ. Программа для расчета электромагнитных полей и исследования электродинамических свойств структур из проводящих элементов. – [Электр. ресурс]. – Режим доступа: <http://www.edem3d.ru>. – 22.12.2012 г.
9. intergraph. SmartSketch® Plant... – [Электр. ресурс]. – Режим доступа: <http://www.intergraph.com/products/ppm/smarts sketch/default.aspx>. – 22.12.2012 г.

10. Боровинский, А.В. Ввод данных с эскизов для компьютерного моделирования двумерных физических полей [Текст] / А.В. Боровинский, В.О. Мищенко // Труды XIV Международного симпозиума «Методы дискретных особенностей в задачах математической физики» (МДОЗМФ-2009). – Харьков – Херсон, 2009. – С. 18 – 22.

11. Бондарев, В.Н. Искусственный интеллект [Текст] / В.Н. Бондарев, Ф.Г. Аде. – Севастополь: СевНТУ. – 2002. – 615 с.

12. Мищенко, В.О. Энергетический анализ программного обеспечения с примерами реализации для

Ада-программ [Текст] / В.О. Мищенко. – Х.: ХНУ им. В.Н. Каразина. – 2007. – 119 с.

13. Suen, Z. A fast parallel algorithm for thinning digital patterns [Текст] / Zhang Suen, C.Y. Suen // Communications of the ACM, 27:236-239. – 1984. – С. 33. – 87.

14. IEEE Std 982.2-1988. IEEE Guide for the Use of Standard Dictionary of Measures to Produce Reliable Software– [Электр. ресурс]. – Режим доступа: <http://members.aol.com/geshome/IEEE982/IEEE9822.pdf>. – 22.12.2011 г.

Поступила в редакцию 22.02.2012

Рецензент: д-р техн. наук, проф. О.В. Поморова, Хмельницкий национальный университет, Украина

СКЛАДНІСТЬ РЕАЛІЗАЦІЇ ІНТЕЛЕКТУАЛЬНИХ ГРАФІЧНИХ ІНТЕРФЕЙСІВ ДЛЯ ПРОГРАМ, ЩО ЗАСНОВАНІ НА МДО

О.В. Боровінський, В.О. Міщенко

Розглядаються підсумки розробки програми, що забезпечує комп'ютерну систему моделювання дифракційних явищ методами дискретних особливостей введенням даних у формі ескізів з контурами областей розсіяння хвиль та написами, що готуються у довільному графічному редакторі. У статті запропоновано метод, заснований на сполученні змістовного та метричного аналізів, причому застосування знаходять енергетичні метрики. Обґрунтовано тезу про те, що інтелектуальні графічні інтерфейси виправдовують себе у разі багатократного застосування у складі програмних систем моделювання без складних модифікацій.

Ключові слова: інтелектуальний графічний інтерфейс, якість, вартість, ризику, метрики.

ON THE COMPLEXITY OF THE DEVELOPMENT OF INTELLECTUAL GRAPHIC INTERFACES FOR APPLICATIONS BASED ON DSM

A.V. Borovinskiy, V.O. Mishchenko

We developed an intellectual data input solution for DSM modeling applications. Input data is read as sketches of wave scattering region boundaries and text labels which can be prepared with arbitrary text editor. We discuss the results of developing of such application. We propose a method based on combination of content and metric analyses for determining how complex, risky and expensive such interfaces. We use the energetic metrics. We argue that the development of intellectual graphic interfaces is acceptable only in cases when they can be reused many times without complex modifications.

Key words: intellectual GUI, software quality, cost, risks, metrics.

Боровинский Алексей Викторович – соискатель каф. моделирования систем и технологий Харьковского национального университета им. В.Н. Каразина, Харьков, Украина.

Мищенко Виктор Олегович – канд. физ.-мат. наук, доц., Харьковской национальной университет им. В.Н. Каразина, Харьков, Украина.