

Логинов Владислав Николаевич

Магистрант,
Уральский федеральный университет имени
первого Президента России Б. Н. Ельцина
(Екатеринбург, Россия)
E-mail: logos-13_94@mail.ru

Loginov Vladislav Nikolaevich

Master student,
Ural Federal University named after the first
President of Russia B. N. Yeltsin
(Ekaterinburg, Russia)
E-mail: logos-13_94@mail.ru

Щипанов Кирилл Александрович

Кандидат технических наук, доцент,
Уральский федеральный университет имени
первого Президента России Б. Н. Ельцина
(Екатеринбург, Россия)
E-mail: kirill.shchipanov@urfu.ru

Shchipanov Kirill Aleksandrovich

PhD in Technical Sciences,
Associate Professor,
Ural Federal University named after the first
President of Russia B. N. Yeltsin
(Ekaterinburg, Russia)
E-mail: kirill.shchipanov@urfu.ru

Лавров Владислав Васильевич

Доктор технических наук, профессор,
Уральский федеральный университет имени
первого Президента России Б. Н. Ельцина
(Екатеринбург, Россия)
E-mail: v.v.lavrov@urfu.ru

Lavrov Vladislav Vasilyevich

Doctor of Technical Sciences, Professor,
Ural Federal University named after the first
President of Russia B. N. Yeltsin
(Ekaterinburg, Russia)
E-mail: v.v.lavrov@urfu.ru

**РАЗРАБОТКА WEB-ПРИЛОЖЕНИЯ
СИСТЕМЫ ИДЕНТИФИКАЦИИ
ДЕФЕКТОВ МЕТАЛЛА НА
ПОЛУТОНОВЫХ ИЗОБРАЖЕНИЯХ С
ИСПОЛЬЗОВАНИЕМ КАСКАДНОГО
КЛАССИФИКАТОРА ХААРА НА
ПЛАТФОРМЕ ASP.NET CORE MVC**

**DEVELOPMENT OF WEB
APPLICATION SYSTEMS
IDENTIFICATION OF METAL
DEFECTS ON HALF-TONE IMAGES
USING THE HAAR CASCADE
CLASSIFIER ON ASP.NET CORE
MVC PLATFORM**

Аннотация. В процессе производства металлических изделий и металлоконструкций в материале нередко возникают различные дефекты, влияющие на эксплуатационные характеристики объекта. Для выявления внутренних дефектов используется радиографический метод контроля, позволяющий оценить качество материала контролируемого объекта по рентгенографическому снимку, который далее оцифровывается и передается на анализ. В статье рассмотрены основные методы поиска объектов на полутонном изображении, а также рассмотрен процесс разработки информационной системы, использующей в качестве основного инструмента обработки изображений каскадный классификатор Хаара.

Abstract. In the process of manufacturing metal products and metal structures, various defects often appear in the material that affect the performance characteristics of the object. In order to detect internal defects, a radiographic control method is used, which allows assessing the quality of the material of the test object from an X-ray image, which is further digitized and transmitted for analysis. The article discusses the main methods of searching objects on a halftone image, and also describes the process of developing an information system using the Haar cascade classifier as the main image processing tool.

Ключевые слова: дефекты металла, поиск, радиография, алгоритмы, полутоновые изображения, неразрушающий контроль

Keywords: metal defects, search, radiography, algorithms, halftone images, non-destructive testing

Введение

Учитывая масштабную автоматизацию разнообразных отраслей производства, а в частности тяжелого машиностроения, потребность в разработке качественного программного обеспечения и его внедрения особенно высока. При производстве продукции отрасли тяжелого машиностроения, как правило, проводится процедура неразрушающего контроля с последующим формированием отчетной документации. Так, например, при применении радиографического метода неразрушающего контроля создается снимок исследуемого объекта с присутствующими в нем дефектами либо без них. Далее снимок обрабатывается при помощи программно-аппаратных комплексов, которые обеспечивают обнаружение и регистрацию найденных на изображении дефектов для составления акта или заключения о прохождении / не прохождении объектом неразрушающего контроля [4]–[6].

На текущий момент имеется несколько коммерческих программных средств, позволяющих выполнить корректную обработку полутоновых изображений с последующим анализом графического содержимого для поиска дефектов металла. Однако следует уточнить, что эти программные средства в большинстве своем поставляются на рынок программного обеспечения в составе дорогостоящих программно-аппаратных комплексов. Так, например, программное обеспечение «Видеорен-Про» [16] поставляется в составе программно-аппаратного комплекса «Градиент», а программное обеспечение X-Vizor [13] поставляется только в составе программно-аппаратного комплекса «Карат КР».

Для создания прототипа программного обеспечения, выполняющего поиск дефектов металла на полутоновых изображениях, требуется разработать и обосновать архитектуру прототипа, выбрать средства реализации тех или иных модулей системы, а также определить основной инструмент поиска объектов на изображениях.

В рамках исследовательской части разработки необходимо обеспечить проведение различных тестов, демонстрирующих способности прототипа при обращении к нему множества пользователей.

Основная часть

Для разрабатываемого программного продукта массив входных данных представляет собой изображение рентгенографического снимка контролируемого объекта. Снимок по своему содержанию и структуре является полутоновым изображением объекта контроля с наличием или отсутствием дефектов металла, таких как флокены, поры, деформации, трещины и др.

В роли основного инструмента обработки входных данных используется библиотека обработки изображений и алгоритмов компьютерного зрения OpenCV [3], [9], [17]–[19]. Она является открытым программным обеспечением с лицензией BSD, которое может иметь как коммерческую, так и научную применимость. На начальном этапе разработки продукта главная задача состоит в сравнительном анализе и выборе основного алгоритмического обеспечения. В частности, необходимо выбрать наиболее подходящий механизм обнаружения искомых объектов на изображении.

Библиотека OpenCV обеспечивает поддержку соответствующих программных средств:

- поиск по шаблону с использованием функции `matchTemplate(img, img_tpl, mode)`, где `img` – изображение, на котором требуется провести поиск по шаблону; `img_tpl` – непосредственно сам шаблон; `mode` – параметр, определяющий формулу расчета корреляции между двумя изображениями;
- поиск с использованием цветовых фильтров с применением функции `InRange(img, hsv_min, hsv_max)`, где `img` – изображение, на которое накладывается цветовой фильтр; `hsv_min` – начальное значение цвета диапазона; `hsv_max` – конечное значение цвета диапазона;
- поиск с использованием контуров объектов с применением функции `FindContours(img, mode, method)`, где `img` – изображение, на котором требуется провести поиск контуров; `mode` – параметр, определяющий режим поиска контуров; `method` – метод аппроксимации контуров;
- поиск с использованием характерных для искомого объекта точек с применением функции `cv2.SimpleBlobDetector_create(detector_params)`, которая предоставляет функционал детектора областей, обладающих достаточно четкой границей и одинаковой яркостью. Аргументами функции являются: `detector_params` – объект конфигурации детектора, представляющий собой стек параметров, определяющих режим работы алгоритма;
- поиск с использованием методов машинного обучения при помощи классификатора (каскада) Хаара [8], [12]. Каскад Хаара – это совокупность признаков Хаара, содержащая множество примитивов, характеризующих множество искомых объектов. В основе поиска используется функция `CascadeClassifier(cascade)`, где `cascade` – предназначенный для обнаружения определенных объектов классификатор (каскад) Хаара.

1. Исследование эффективности алгоритмов поиска объектов на изображении

Описанные выше средства поиска объектов на изображении по порядку применены на тестовом изображении (см. рис. 1) для выявления наиболее оптимального метода для достижения цели исследования. Ниже приведены заключения по результатам исследования для каждого метода поиска.

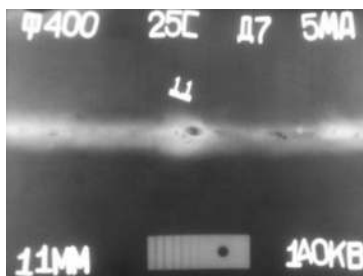


Рис. 1. Тестовое изображение рентгенографического снимка с дефектом в материале контролируемого объекта

1.1. Поиск по шаблону. Для исследования данного типа поиска из частей тестового изображения создано несколько шаблонов (см. рис. 2).

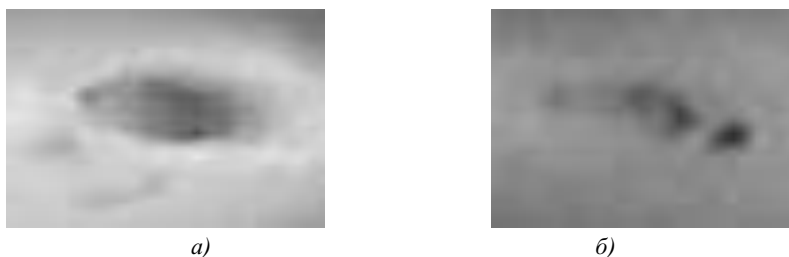


Рис. 2. Шаблоны, по которым производился поиск:

a – шаблон с изображением поры диаметром ~ 3 мм; *б* – шаблон с изображением цепочки пор протяженностью ~ 4 мм

Итоговый результат использования данного типа поиска изображен на рис. 5а. Следует уточнить, что среда для запуска скриптов содержит интерпретатор Python 3.6.1 (модуль Spyder 3.2.1), который входит в состав программного пакета Anaconda и не имеет различий в работе по сравнению с интерпретатором Python 3.6.1, устанавливаемым отдельно.

В процессе анализа полученных результатов установлено, что метод поиска объекта по шаблону обладает сверхчувствительностью к изменению содержания шаблонов поиска и не может быть использован в разрабатываемом продукте. При этом формирование множества шаблонов для корректного поиска требует большой объем трудозатрат ввиду сложности обеспечения его репрезентативности.

1.2. Поиск с использованием цветовых фильтров. В процессе применения данного метода поиска обрабатываемый образец изображения, представляющий для алгоритма входные данные, необходимо конвертировать в черно-белую цветовую схему для более корректной работы механизма поиска. Также для более тонкой настройки поиска разработано дополнительное окно интерфейса с регулируемыми параметрами (см. рис. 3), характеризующими параметры цветовой модели HSV (Hue, Saturation, Value – тон, насыщенность, значение), которая используется в алгоритме по умолчанию. Параметры “h1”, “s1” и “v1” определяют значения цветовых характеристик нижней цветовой границы поиска, “h2”, “s2” и “v2” – верхней цветовой границы поиска. Таким образом, в контексте применения алгоритма предусмотрена возможность гибкой настройки параметров поиска для достижения оптимального результата.

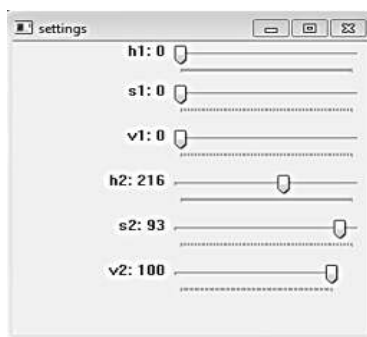


Рис. 3. Интерфейс тонкой настройки параметров поиска

Результаты поиска объектов с использованием цветowych фильтров демонстрируются на рис. 5б. При этом для параметров, обеспечивающих тонкую настройку поиска, установлены оптимальные для целей поиска значения.

В процессе анализа результатов работы алгоритма определена его непригодность в качестве основного алгоритма поиска объектов на изображении в разрабатываемой программе ввиду многообразия оттенков цветов, размеров и форм дефектов металла на изображении. В большей степени этот факт обусловлен отсутствием ярко выраженного относительно фона цвета искомого объекта.

1.3. Поиск с использованием контуров объектов. При использовании данного типа поиска главным критерием обнаружения являлось наличие у геометрической фигуры, аппроксимирующей искомый объект, более восьми вершин. Этот критерий указывает на то, что детектируемый объект должен иметь округлый контур, соответствующий таким типам дефектов металла, как поры (см. рис. 1), кратеры, свищи, а также другим дефектам, имеющим похожую форму контура. Однако не все искомые объекты имеют округлую форму контура, что несколько усложняет процесс поиска из-за отсутствия требуемой гибкости алгоритма. Итог использования поиска с применением аппроксимации контуров объектов представлен на рис. 5в.

На данном рисунке четко видны несколько ложных совпадений, обусловленных, в первую очередь, присутствием на изображении специфичных для обработанных рентгенографических снимков знаков, являющихся источником помех для работы алгоритма. Необходимо отметить, что алгоритм хорошо подходит для поиска объектов, имеющих сплошные, ярко выраженные линии контура, при отсутствии источников помех на изображении. Те изображения, которые содержат разнообразные по форме объекты, данным алгоритмом корректно обрабатываться не могут.

1.4. Поиск с использованием характерных для искомого объекта точек. В процессе работы данного метода используется информация о так называемых ключевых точках blob-объекта. Blob-объект – это совокупность на изображении связанных пикселей, имеющая общие свойства для своего содержимого (например, значение оттенков серого). Для ключевой точки, как правило, формируется так называемый дескриптор – ее характеристика. Существует несколько методик расчета дескрипторов для признаков объекта: ORB, SIFT, SURF и другие, при этом некоторые из этих методик являются запатентованными (например, SIFT) и их коммерческое использование ограничено. В функции SimpleBlobDetector_create(), обеспечивающей

формирование дескриптора и функционирование детектора на его основе, по умолчанию используется метод “SimpleBlob”, который в большей степени соответствует цели исследования.

Предварительно установлены необходимые значения конфигурационных параметров, вычисленные эмпирическим путем для более оптимальной процедуры поиска объектов. На рис. 4 изображена визуализация регулируемых параметров с целью создания наиболее четкого представления об их назначении. Для каждого параметра изменение его численного значения вызывает изменение примитива, которым этот параметр представлен:

- Area – площадь искомого объекта. Определяются ее минимальная и максимальная границы.
- Thresholds – порог значения яркости искомого объекта. Определяются его минимальное и максимальное значения.
- Circularity – параметр, устанавливающий близость контура детектируемого объекта к окружности. Определяется его минимальная величина.
- Inertia – параметр, устанавливающий удлинение формы окружности. Для формы окружности, например, значение этого параметра равно 1, для формы эллипса оно находится между 0 и 1, для линии – равно 0. Определяется его минимальная величина.
- Convexity – параметр, устанавливающий плотность контура детектируемого объекта. Определяется его минимальная величина.

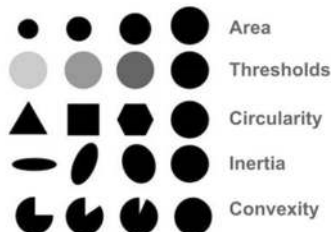


Рис. 4. Визуализация устанавливаемых параметров

Результаты поиска с использованием ключевых точек детектируемого объекта изображены на рис. 5г, на котором были помечены найденные объекты. Для успешного поиска более разнородных по размеру и форме объектов требуется более тонкая настройка конфигурационных параметров.

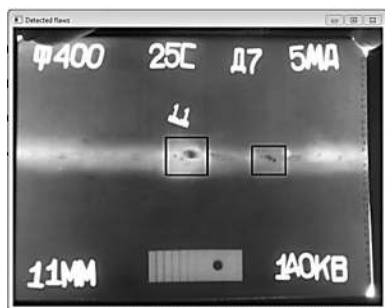
На рис. 5г изображены все найденные объекты, в число которых входят и два искомым. Обнаружение объектов, не являющихся искомыми, объясняется наличием на изображении специальных знаков, применяющихся в процессе обработки рентгенографических снимков. При этом стоит отметить, что метод поиска показал достаточную точность для решения задачи обработки изображений рентгенографических снимков без специальных символов.

1.5. Поиск с использованием методов машинного обучения при помощи классификатора (каскада) Хаара. В основе функционирования каскадного классификатора находится алгоритм Виолы-Джонса, позволяющий в реальном времени

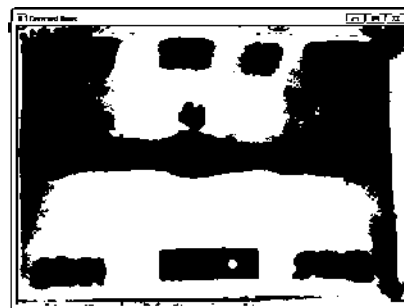
распознавать объекты на изображении и использующий для обеспечения такого функционала признаки Хаара. Признак (примитив) Хаара определяется как разность сумм пикселей двух смежных областей внутри контура некоторого геометрического примитива, который может занимать различные положения и масштабироваться с разными коэффициентами на изображении. В данном методе примитивы Хаара организованы в структуру, именуемую каскадным классификатором.

В процессе разработки классификатора выборка образцов – изображений, содержащих искомый объект, генерируется встроенной в библиотеку утилитой *opencv_createsamples.exe*, а затем подается в роли входных данных в утилиту *opencv_traincascade.exe*, которая отвечает за процесс обучения созданного классификатора распознаванию требуемых объектов. Всего при создании и обучении каскадного классификатора использовано 200 позитивных и 400 негативных образцов. Результаты использования каскадного классификатора Хаара на тестовом изображении продемонстрированы на рис. 5д.

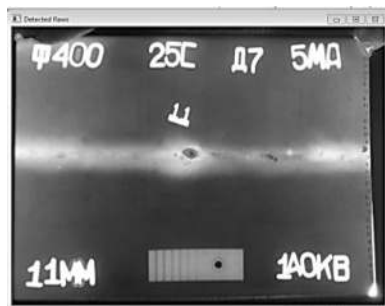
Необходимо уточнить, что объем материалов для обучения и работы классификатора крайне невелик из-за ограниченности доступных данных. Тем не менее данный метод показал достаточную точность обработки при поиске объектов на изображении, корректно обнаружив при этом два дефекта из двух. Вместе с тем был найден особый символ, являющийся частью специального шаблона для масштабирования дефектов. При условии увеличения объема материала для обучения каскадного классификатора описанный метод может успешно применяться в роли основного алгоритма детектирования объектов в разрабатываемой программе.



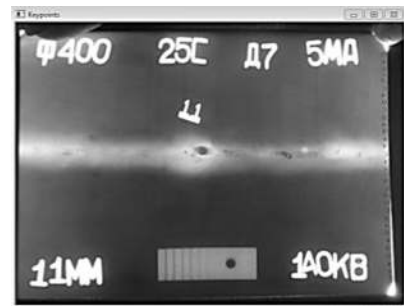
а)



б)



в)



г)



д)

Рис. 5. Результаты применения к тестовому изображению различных видов поиска:

- а – поиск с использованием шаблонов;
- б – поиск с использованием цветowych фильтров;
- в – поиск с использованием контуров объектов;
- г – поиск с использованием характерных для искомого объекта точек;
- д – поиск с использованием классификатора Хаара

2. Разработка архитектуры информационной системы

Архитектура разрабатываемой информационной системы должна обеспечивать простоту внедрения и сопровождения, а также надежность функционирования. При заданных требованиях и особенностях основного метода поиска объектов принято решение вынести часть программы, отвечающую за поиск объектов, за пределы клиентских приложений, что дает преимущество в виде частичной минимизации объема программ. Это решение определяет архитектуру системы как «клиент-сервер». При этом взаимодействие клиентской программы и программы, выполняющей поиск, производится посредством API-интерфейса. В первую очередь это обусловлено необходимостью регулярного обновления классификатора для увеличения точности поиска. В противном случае требуется вносить изменения в каждое клиентское приложение, что увеличивает затраты на сопровождение продукта.

Также принято решение о создании веб-версии системы, позволяющей получить необходимый функционал без установки дополнительного программного обеспечения на компьютер клиента. Данная версия продукта использует тот же механизм взаимодействия с поисковой программой, что и полноценное клиентское приложение. Таким образом, имеется единый функциональный узел, осуществляющий обработку изображений, и тем самым исключается излишнее дублирование программного кода. Данный факт обосновывает выбор архитектуры «клиент-сервер», принятой в начале этапа проектирования информационной системы.

Принципиальная структура системы изображена на рис. 6. Основные компоненты представлены отдельными сущностями, между которыми осуществляется информационное взаимодействие. Следует отметить, что различные сущности реализованы с помощью определенных фреймворков и языков программирования, наиболее подходящих для разработки в соответствующем контексте. Приложения, реализованные сторонними разработчиками, также могут использовать API-сервер, основываясь на идее открытого программного обеспечения.

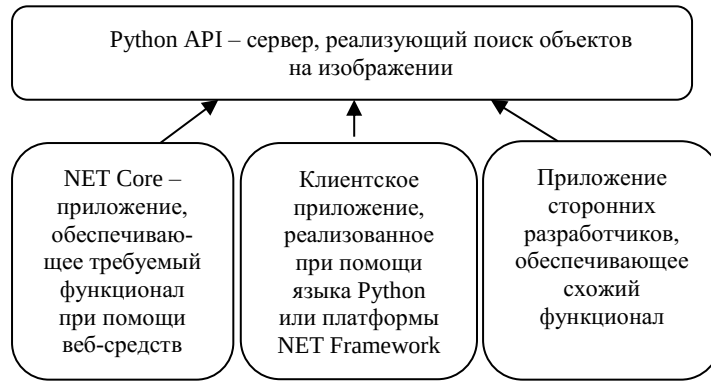


Рис. 6. Архитектура программного обеспечения разрабатываемой системы

3. Реализация информационной системы

В процессе исследований для анализа качества поиска с использованием вновь созданного классификатора возникает потребность в приложении, демонстрирующем базовый функционал прототипа программного продукта. Так как основным назначением прототипа является проверка работоспособности классификатора Хаара, консольного интерфейса программы вполне достаточно для обеспечения ввода / вывода информации. В качестве языка программирования для реализации приложения принят язык Python, обладающий обширной поддержкой библиотек обработки изображений. Для полноценного использования классификатора необходимо установить библиотеку OpenCV, которая содержит основные инструменты поиска объектов на изображениях. Процесс установки библиотек в модульной IDE Anaconda инициализируется при помощи вызова встроенного менеджера пакетов Conda. Сигнатура вызова менеджера и установки библиотеки OpenCV представлена ниже:

```
conda install -c conda-forge opencv
```

После установки необходимых модулей следует процесс работы над программным кодом. В данном случае принято решение использовать функциональный подход, обеспечивающий простоту и быстроту написания программы. Объектно-ориентированный подход, скорее всего, является избыточным для написания небольших по объему и функционалу программ и должен быть применен при реализации полноценной системы.

При выполнении консольной программы тестовое изображение обработано успешно. Все найденные объекты выделены красным прямоугольником. Результат выполнения программы изображен на рис. 7.

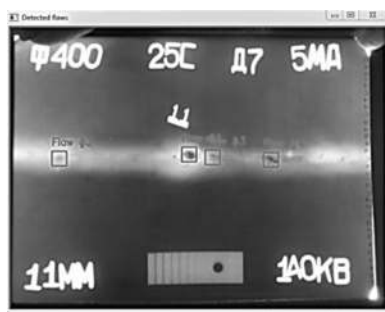


Рис. 7. Результат выполнения консольного приложения

При сравнительном анализе результатов работы двух консольных приложений, использующих классификаторы, основанные на различных по объему исходных данных, последний классификатор показал более полный и точный результат (см. рис. 8).



Рис. 8. Результаты обработки изображений классификаторами Хаара:
а – первоначальным классификатором, основанным на ~ 600 образцах;
б – классификатором, основанным на ~ 16000 образцах

4. Реализация серверного приложения, осуществляющего поиск объектов на изображениях

Для реализации серверного приложения использован веб-фреймворк Django, написанный на языке Python. В процессе разработки требуется также установить некоторые дополнительные модули, обеспечивающие поддержку необходимых функций. Так, например, для качественного формирования и обработки запроса нужно установить модуль Django Rest Framework, сигнатура команды установки которого приведена ниже:

```
conda install -c conda-forge djangoestframework
```

Для установки самого фреймворка Django требуется выполнить команду, листинг которой приведен ниже:

```
conda install -c anaconda django
```

При помощи установленных инструментальных средств разработка серверного приложения заметно упрощается. При написании основного кода программы использован объектно-ориентированный подход, дающий некоторые преимущества, критичные при разработке такого рода систем. Приложение на фреймворке Django организовано по структуре типа Модель – Вид – Контроллер, характерной для большинства веб-проектов. При этом некоторые компоненты данной структуры могут быть упрощены или удалены в контексте серверного приложения, не обладающего классическим HTML-представлением.

Вся требуемая функциональность серверного приложения обеспечена тремя модулями:

- `flaw.py` – модуль, содержащий класс-обработчик изображений `Flaw Detector`. Экземпляр класса `Flaw Detector` возвращает двумерный массив, в котором первый уровень представляет собой индексы найденных объектов, а второй – координаты прямоугольника, содержащего найденный объект.
- `views.py` – модуль, предназначенный для обработки запроса.
- `urls.py` – модуль, определяющий формат веб-запроса.

Корректность работы серверного приложения проверена при помощи программы – генератора запросов `Advanced REST Client`. Переданное тестовое изображение, содержащее 4 искомых объекта, успешно обработано, и в качестве возвращаемых серверной частью данных передан двумерный массив элементов, который описывает 4 найденных объекта.

5. Реализация приложения для обработки изображений на платформе NET Core

Клиентское веб-приложение предназначено для отображения результатов, возвращаемых серверной частью системы, а также для формирования учетных данных пользователей, массивов обработанных изображений и др.

Платформа `NET Core`, используемая для создания переносимых веб-приложений, в наиболее полной мере удовлетворяет требованиям разработки клиентской части системы. В основе разработки на платформе `NET Core` лежит разделение структуры создаваемого приложения по принципу Модель – Вид – Контроллер, также используемому при создании серверной части системы. Данный подход позволяет четко разграничить основные функциональные элементы веб-приложения. В отличие от серверной части, не содержащей HTML-интерфейс, клиентское приложение использует полный набор элементов структуры `MVC` (англ. `Model – View – Controller`) для обеспечения полноценного HTML-интерфейса.

Первичные возможности прототипа подразумевают обработку изображений, хранение обработанных образцов, ведение учетных записей пользователей. Интуитивно понятный интерфейс, не обладающий избыточностью, минимизирует количество ошибок программы из-за неправильной последовательности действий пользователя. Пример типового взаимодействия пользователя и информационной системы приведен на рис. 9.

Для использования приложения необходимо пройти процедуру авторизации при помощи данных, введенных на этапе регистрации пользователя. В приложении использован стандартный механизм валидации и аутентификации, предоставляемый платформой `NET Core` по умолчанию. При заполнении требуемых для обработки

данных пользователю необходимо ввести масштабный коэффициент, определяющий полноту поиска и размер искомых элементов (см. рис. 9а).

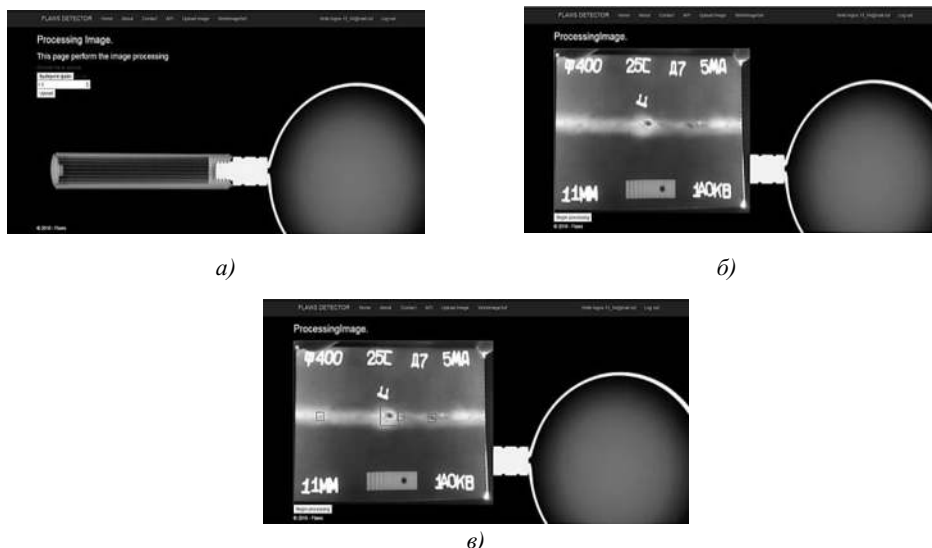


Рис. 9. Возвращаемые серверным приложением данные:
 а – окно ввода данных и загрузки файлов изображений;
 б – окно с загруженным изображением;
 в – окно с изображением и обнаруженными на нем объектами

Выводы

Созданный прототип в большей степени демонстрирует замысел разработки информационной системы, обеспечивающей поиск объектов на полутоновых изображениях. Также отметим, что тестирование на большинстве образцов показало следующее: каскадный классификатор Хаара пригоден в качестве основного инструмента поиска объектов на изображениях. Однако существует определенное множество ложных срабатываний, которое необходимо минимизировать или ликвидировать.

Выбранная модульная архитектура системы обеспечивает максимальную гибкость разрабатываемого продукта. В частных случаях модуль поиска объектов на изображениях может быть использован совместно или подчиненно с другими типами архитектур при условии концептуальной совместимости.

Литература

1. Абрамов Г. В., Медведкова И. Е., Коробова Л. А. Проектирование информационных систем. – Воронеж: Воронежский государственный университет инженерных технологий, 2012. – 172 с.
2. Антонов В. Ф., Москвитин А. А. Методы и средства проектирования информационных систем. – Ставрополь: Северо-Кавказский федеральный университет, 2016. – 342 с.
3. Буэно Глория Гарсия, Суарес Оскар Денис, Эспиноса Хосе Льюис. Обработка изображений с помощью OpenCV. – Москва: ДМК-Пресс, 2016. – 210 с.

4. ГОСТ 15467-79. Управление качеством продукции. Основные понятия. Термины и определения. – Москва: Стандартинформ, 2009. – 21 с.
5. ГОСТ 30242-97. Дефекты соединений при сварке металлов плавлением. Классификация, обозначение и определения. – Минск: Межгосударственный совет по стандартизации, метрологии и сертификации, 1997. – 8 с.
6. ГОСТ Р ИСО 6520-1-2012. Сварка и родственные процессы. Классификация дефектов геометрии и сплошности в металлических материалах. Сварка плавлением. – Москва: Стандартинформ, 2014. – 36 с.
7. Гриценко Ю. Б. Архитектура предприятия. – Томск: Эль Контент, 2011. – 206 с.
8. Классификатор Хаара. – Москва: Тематические Медиа, 2006. – URL: <https://habrahabr.ru/company/recogitor/blog/228195.html> (дата обращения: 15.01.2019).
9. Кэлер А., Брэдки Г. Изучаем OpenCV 3. – Москва: ДМК-Пресс, 2017. – 826 с.
10. Лутц М. Изучаем Python. – Санкт-Петербург: Символ-Плюс, 2011. – 1280 с.
11. Лутц М. Программирование на Python. – Санкт-Петербург: Символ-Плюс, 2011. – 992 с.
12. Признаки Хаара. – URL: https://ru.wikipedia.org/wiki/Признаки_Хаара.html (дата обращения: 15.01.2019).
13. Системы и комплексы компьютерной радиографии DUERR NDT. – URL: <https://newcom-ndt.ru/x-vizor.html> (дата обращения: 15.01.2019).
14. Сорокин А. А., Орлова А. Ю. Реинжиниринг бизнес-процессов. – Ставрополь: Северо-Кавказский федеральный университет, 2014. – 212 с.
15. Стасышин В. М. Проектирование информационных систем и баз данных. – Новосибирск: Новосибирский государственный технический университет, 2012. – 100 с.
16. Цифровая система «Видеорен ПРО». – URL: <http://www.unitest-roentgen.ru/produktsiya/tsifrovaya-radiografiya/kompleksy-tsifrovoj-radiografii-gradient/217-programmnoe-obespechenie-videoren-pro.html> (дата обращения: 15.01.2019).
17. Learn OpenCV. – URL: <http://www.learnopencv.com> (дата обращения: 15.01.2019).
18. OpenCV. – URL: <https://opencv.org> (дата обращения: 15.01.2019).
19. OpenCV. – URL: <https://ru.wikipedia.org/wiki/OpenCV.html> (дата обращения: 15.01.2019).

References

1. Abramov G. V., Medvedkova I. E., Korobova L. A. *Proektirovanie informatsionnykh system* [Designing of information systems]. Voronezh: Voronezhskii gosudarstvennyi universitet inzhenernykh tekhnologii, 2012. 172 p.
2. Antonov V. F., Moskvitin A. A. *Metody i sredstva proektirovaniia informatsionnykh system* [Methods and means of designing information systems]. Stavropol': Severo-Kavkazskii federal'nyi universitet, 2016. 342 p.
3. Bueno Gloria Garsia, Suares Oskar Denis, Espinosa Khose L'uis. *Obrabotka izobrazhenii s pomoshch'iu OpenCV* [Image processing with OpenCV]. Moscow: DМК-Press, 2016. 210 p.
4. GOST 15467-79. *Upravlenie kachestvom produktsii. Osnovnye poniatii. Termy i opredeleniia* [GOST 15467-79. Product quality management. Basic concepts. Terms and Definitions]. Moscow: Standartinform, 2009. 21 p.
5. GOST 30242-97. *Defekty soedinenii pri svarke metallov plavleniem. Klassifikatsiia, oboznamenie i opredeleniia* [GOST 30242-97. Defects of joints in the welding of metals by melting. Classification, designation and definitions]. Minsk: Mezhsudarstvennyi soviet po standartizatsii, metrologii i sertifikatsii, 1997. 8 p.
6. GOST R ISO 6520-1-2012. *Svarka i rodstvennye protsessy. Klassifikatsiia defektov geometrii i sploshnosti v me-tallicheskih materialakh. Svarka plavleniem* [GOST R ISO 6520-1-2012. Wel-

ding and related processes. Classification of defects in geometry and continuity in metallic materials. Welding by melting]. Moscow: Standartinform, 2014. 36 p.

7. Gritsenko Iu. B. *Arkhitektura predpriiatiia* [Architecture of the enterprise]. Tomsk: El' Kontent, 2011. 206 p.

8. *Klassifikator Khaara* [Haar's classifier]. Moscow: Tematicheskie Media, 2006. Available at: <https://habrahabr.ru/company/recognitor/blog/228195.html> (accessed: 15.01.2019).

9. Kaehler A., Bradski G. *Izuchaem OpenCV 3* [Learning OpenCV 3]. Moscow: DMK-Press, 2017. 826 p.

10. Luts M. *Izuchaem Python* [Learning Python]. St Petersburg: Simvol-Plus, 2011. 1280 p.

11. Luts M. *Programmirovaniye na Python* [Programming Python]. St Petersburg: Simvol-Plus, 2011. 992 p.

12. *Priznaki Khaara* [Haar's features]. Available at: https://ru.wikipedia.org/wiki/Признаки_Хаара.html (accessed: 15.01.2019).

13. *Sistemy i komplekсы komp'yuternoi radiografii DUERR NDT* [Systems and complexes of computer radiography DUERR NDT]. Available at: <https://newcom-ndt.ru/x-vizor.html> (accessed: 15.01.2019).

14. Sorokin A. A., Orlova A. Iu. *Reinzhiniring biznes-protsessov* [Reengineering of business processes]. Stavropol': Severo-Kavkazskii federal'nyi universitet, 2014. 212 p.

15. Stasyshin V. M. *Proektirovaniye informatsionnykh sistem i baz dannykh* [Designing information systems and databases]. Novosibirsk: Novosibirskii gosudarstvennyi tekhnicheskii universitet, 2012. 100 p.

16. *Tsifrovaya sistema "Videoren PRO"* [Digital system "Videoren PRO"]. Available at: <http://www.unitest-roentgen.ru/produktsiya/tsifrovaya-radiografiya/komplekсы-tsifrovoj-radiografii-gradient/217-programmnoe-obespechenie-videoren-pro.html> (accessed: 15.01.2019).

17. *Learn OpenCV*. Available at: <http://www.learnopencv.com> (accessed: 15.01.2019).

18. *OpenCV*. Available at: <https://opencv.org> (accessed: 15.01.2019).

19. *OpenCV*. Available at: <https://ru.wikipedia.org/wiki/OpenCV.html> (accessed: 15.01.2019).

Для цитирования: Логинов В. Н., Щипанов К. А., Лавров В. В. Разработка web-приложения системы идентификации дефектов металла на полутоновых изображениях с использованием каскадного классификатора Хаара на платформе ASP.NET CORE MVC // Вестник Череповецкого государственного университета. – 2019. – № 4 (91). – С. 7–20. DOI: 10.23859/1994-0637-2019-4-91-1

For citation: Loginov V. N., Shchipanov K. A., Lavrov V. V. Development of web application systems identification of metal defects on half-tone images using the Haar cascade classifier on ASP.NET Core MVC platform. *Bulletin of the Cherepovets State University*, 2019, no. 4 (91), pp. 7–20. DOI: 10.23859/1994-0637-2019-4-91-1