УДК 004.921
В. Н. Логинов, К. А. Щипанов, В. В. Лавров
ФГАОУ ВО «Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина», г. Екатеринбург, Россия

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

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, который позволяет оценить качество материала контролируемого объекта по рентгенографическому снимку, который далее оцифровывается и передается на анализ. В статье рассмотрены основные методы поиска объектов на полутоновом изображении, а также рассмотрен процесс разработки информационной системы, использующей в качестве основного инструмента обработки изображений каскадный классификатор Хаара.

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

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.

Key words: *metal defects, search, radiography, algorithms, halftone images, non-destructive testing.*

Введение. В связи с масштабной автоматизацией различных отраслей производства, а в частности, тяжелого машиностроения, потребность в создании качественного программного обеспечения и его внедрения особенно высока. В машиностроении при проведении радиографического контроля создается снимок контролируемого объекта с дефектами или без них. Далее снимок обрабатывается при помощи программно-аппаратных комплексов, которые обнаруживают и регистрируют найденные на изображении дефекты для составления акта или заключения о прохождении / непрохождении неразрушающего контроля объектом.

В настоящее время на рынке программного обеспечения присутствуют несколько программных продуктов, осуществляющих обработку полутоновых изображений. Однако эти продукты в большинстве своём поставляются в составе программно-аппаратных комплексов, что добавляет им в стоимости.

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

1. Объект информатизации. Для разрабатываемого продукта входные данные являются изображением рентгенографического снимка объекта, который прошел процедуру неразрушающего контроля радиографическим методом. Снимок представляет собой полутоновое изображение объекта контроля с наличием или отсутствием дефектов металла, таких как трещины, поры, кратеры, включения и др.

В качестве основного инструмента обработки входных данных используется библиотека алгоритмов компьютерного зрения и обработки изображений OpenCV [3, 4]. Библиотека OpenCV является свободно распространяемым продуктом под лицензией BSD с открытым исходным кодом, который может применяться как в коммерческих, так и в научных целях. Основной задачей начального этапа создания продукта является анализ и выбор основного алгоритмического обеспечения. В частности, необходимо выбрать наиболее подходящий механизм поиска определённых объектов на изображении. Библиотека OpenCV предоставляет ряд соответствующих программных средств:

− поиск по шаблону с помощью функции matchTemplate(img, img_tpl, mode), где img – изображение, на котором требуется провести поиск по шаблону; img_tpl – непосредственно, сам шаблон; mode – формула расчета корреляции между двумя изображениями;

− поиск с использованием цветовых фильтров с помощью функции InRange(img, hsv_min, hsv_max), где img – изображение, на которое накладывается цветовой фильтр; hsv_min – начальныйцвет диапазона; hsv_max – кноечный цвет диапазона;

− поиск с использованием контуров объектов с помощью функции FindContours(img, mode, method), где img – изображение, на котором требуется провести поиск контуров; mode – режим поиска контуров; method – метод аппроксимации контуров;

− поиск с использованием характерных для искомого объекта точек с использованием функции cv2.SimpleBlobDetector_create (detector_params), описывающей работу детектора областей с одинаковой яркостью и достаточно чёткой границей, выделяющихся на общем фоне. 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]. Каскад Хаара – это машинно-обучаемая совокупность признаков Хаара, которая содержит примитивы, характеризующие искомый объект. В основе используется функция CascadeClassifier(cascade), где cascade – это созданный для обнаружения определённых объектов классификатор (каскад) Хаара.

2. Исследование эффективности алгоритмов поиска объектов на изображении. Вышеописанные методы поиска объектов на изображении опробованы на тестовом образце (рис. 1) по порядку для определения наилучшего метода для цели исследований.

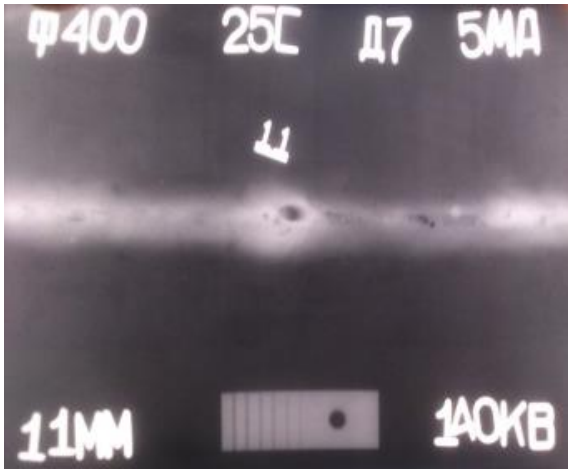

Рис. 1. Тестовый образец изображения рентгенографического снимка с дефектом контролируемого объекта

При исследовании работы методов выявлены некоторые особенности использования того или иного средства, ограничивающие применение его в качестве основного инструмента поиска объектов в разрабатываемом приложении. Так, например, алгоритм поиска с использованием шаблонов показал сильную зависимость результатов поиска от эталонного изображения и тем самым требует полного совпадения искомого объекта и эталонного образца. Прочие классические методы поиска объектов показали более гибкое функционирование, однако при анализе результатов работы обнаружены многочисленные ложные срабатывания. Наиболее точный результат показал метод, использующий в своей основе каскадный классификатор Хаара.

В основе работы классификатора лежит алгоритм Виолы – Джонса, позволяющий распознавать объекты на изображении в реальном времени и использующий для этого признаки Хаара. Признак Хаара – разность сумм пикселей двух смежных областей внутри некоторой геометрической фигуры, которая может занимать различные положения и масштабы на изображении. В данном методе признаки Хаара организованы в каскадный классификатор.

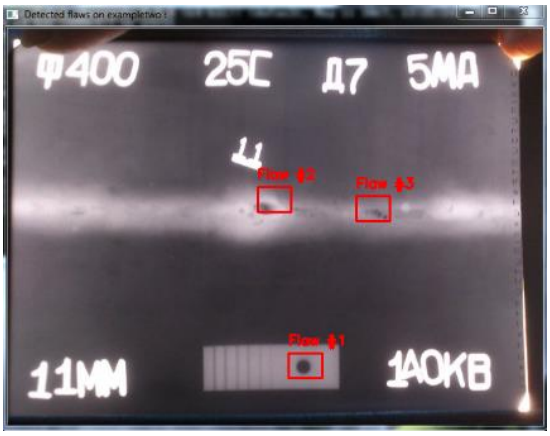

Рис. 2. Результаты поиска с использованием классификатора Хаара

При создании классификатора выборка образцов – изображений, содержащих искомый объект, определяется встроенной в библиотеку утилитой *opencv_createsamples.exe* и затем вводится в качестве входных данных в утилиту *opencv_traincascade.exe*, которая обучает созданный классификатор распознаванию искомых объектов. Всего при создании и обучении классификатора использовано 400 негативных образцов и 200 позитивных. Необходимо отметить, что объём выборок для работы и обучения классификатора крайне невелик из-за ограниченности объема доступного материала. Тем не менее, данный метод успешно показал себя при поиске объектов на изображении, обнаружив два дефекта из двух на изображении (рис. 2). Также обнаружен элемент, являющийся частью специального символа

для масштабирования дефектов. При наличии большего количества материала для обучения классификатора данный метод может успешно использоваться в качестве основного алгоритма обнаружения объектов в создаваемой программе.

3. Разработка 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 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.

Для реализации серверного приложения использован веб-фреймворк Django, написанный на языке Python. В процессе разработки необходимо также установить некоторые дополнительные модули, обеспечивающие поддержку необходимых функций. При помощи установленных инструментальных средств разработка серверного приложения заметно упрощается. 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 FlawDetector. Экземпляр класса FlawDetectorвозвращает 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 AdvancedRESTClient. 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.

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 NETCore 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.

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.

Список использованных источников

1. ГОСТ 30242-97. Дефекты соединений при сварке металлов плавлением. Классификация, обозначение и определения. – Минск: Межгосударственный совет по стандартизации, метрологии и сертификации, 1997. – 8 с.
2. ГОСТ Р ИСО 6520-1-2012. Сварка и родственные процессы. Классификация дефектов геометрии и сплошности в металлических материалах. Часть 1. Сварка плавлением. – М.: Стандартинформ, 2014. – 36 с.
3. OpenCV [Электронный ресурс] // Википедия: свободная энциклопедия. – Электрон. дан. – [Б. м.], 2012. – Режим доступа: https://ru.wikipedia.org/wiki/OpenCV.html, свободный.
4. OpenCV [Электронный ресурс] – Электрон. дан. – [Б. м.], 2018. – Режим доступа: – https://opencv.org, свободный.
5. LearnOpenCV [Электронный ресурс] / *Сатья Маллик* – Электрон. дан. – [Б. м.], 2018. – Режим доступа: http://www.learnopencv.com, свободный.
6. Кэлер А., Брэдски Г. Изучаем OpenCV 3. – М.: ДМК-Пресс,2017. – 826 с.
7. Гарсия Г.Б., Суарес О.Д., Эспиноса Х.Л. Обработка изображений с помощью OpenCV. – М.: ДМК-Пресс, 2016. – 210 с.
8. Признаки Хаара [Электронный ресурс] // Википедия: свободная энциклопедия. – Электрон. дан. – [Б. м.], 2012. – Режим доступа: https://ru.wikipedia.org/wiki/Признаки_Хаара.html, свободный.