XXVII Seminarium

ZASTOSOWANIE KOMPUTERÓW W NAUCE I TECHNICE 2017

Oddział Gdański PTETiS

SYSTEM AUTOMATYZACJI PRZEPROWADZANIA I OCENIANIA SPRAWDZIANÓW STUDENCKICH

Bartłomiej HIRSZ, Sylwia BABICZ

Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki, Katedra Metrologii i Optoelektroniki
tel.: +48 58 347 18 86 e-mail: sylwia.babicz@eti.pg.edu.pl

Streszczenie: Artykuł prezentuje system pozwalający na przeprowadzenie sprawdzianów studenckich w wersji papierowej oraz na komputerach. Kluczowym aspektem systemu jest moduł umożliwiający przeprowadzenie oraz sprawdzenie tradycyjnego testu, którego pytania będą losowane z utworzonej wcześniej bazy (lub kilku baz) pytań. Trzema głównymi zadaniami aplikacji są następująco: generacja i edycja testów oraz zapisywanie ich w bazie pytań, generowanie wersji testu (na podstawie stworzonej uprzednio bazy pytań) wraz z losowaniem kolejności pytań i odpowiedzi, pogląd statystyk indywidualnych oraz grupowych po ewaluacji danego testu. Program umożliwia również przeprowadzenie sprawdzianu na komputerze (poprzez stronę internetową) dzięki możliwości generowania quizów w formie odczytywalnej przez platformę Moodle. Aplikacja do odczytywania i ewaluacji sprawdzianów w wersji papierowej pozwala na odczyt danych studenta (numeru indeksu) oraz wypełnionych odpowiedzi i sprawdzenie ich poprawności na podstawie pobranego z serwera mapowania odpowiedniego dla danej wersji testu.

Słowa kluczowe: apliakcja, moodle, test, OCR, OMR.

1. WPROWADZENIE

Rozpoznawanie znaków i tekstów w plikach graficznych jest obecnie dynamicznie rozwijającą się technologią wykorzystującą wiele technik przetwarzania sygnałów cyfrowych celem uzyskania jak najwierniejszego odtworzenia informacji. Metody te możemy zgrupować w określenia OCR (ang. *Optical Character Recognition*) oraz w OMR (ang. *Optical Mark Recognition*) [1]. Zastosowania tych technik mogą być bardzo szerokie – od rozpoznawania numerów domów na zdjęciach (wykorzystywane przez Google Street View) [2] – aż po digitalizację zbiorów bibliotek [3]. W wielu przypadkach umożliwiają rozpoznanie tekstu znacznie szybciej niż jest to w stanie zrobić człowiek. Oczywiście użyteczność takiego rozpoznania jest wprost proporcjonalna do dokładności użytych algorytmów OCR.

Mimo silnie rozwijających się technik rozpoznawania tekstu, nie istnieje wiele powszechnie dostępnych programów implementujących rozwiązania OCR, a te dostępne na rynku często wymagają wykupienia drogiej licencji. Celem niniejszej pracy – przy wykorzystaniu bezpłatnych bibliotek i ogólnie dostępnych algorytmów – jest implementacja rozwiązania wykorzystującego OCR do praktycznego zadania. Zadaniem tym jest opracowanie i wykonanie systemu umożliwiającego przeprowadzenie i automatyczne sprawdzanie testów zaliczeniowych i egzaminacyjnych. Oprogramowanie napisane w języku C# po skompilowaniu funkcjonuje poprawnie na każdym komputerze osobistym. Jednocześnie aplikacja na telefon komórkowy działa poprawnie na systemie Android i iOS. Co bardzo istotne dla potencjalnego użytkownika – nie wymaga wcześniejszej kalibracji.

2. ISTNIEJĄCE SYSTEMY OPTYCZNEGO ODCZYTU TEKSTU

Oferta programów do generacji i edycji testów wielokrotnego wyboru jest bardzo bogata. Opierają się jednak najczęściej na plikach o formatach dostosowanych tylko do oprogramowania tej samej firmy. Nie oferują zestawiania istniejącego testu z istniejącym oprogramowaniem do automatycznego odczytywania odpowiedzi. Testy wykonywane są albo na platformie online na stronie internetowej producenta albo pisemnie na wygenerowanych stronach. W ostatnim przypadku sprawdzanie wyników wymaga sprawdzania ręcznego.

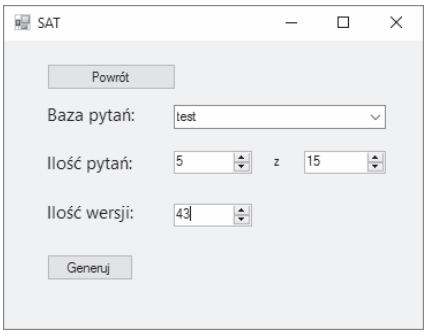
Istnieje wiele rozwiązań do automatycznego odczytu odpowiedzi z formularza, nazywanych ogólnie aplikacjami OMR (ang. *Optical Mark Reader*). Brak znanych polskich producentów tego typu systemów. Na rynku międzynarodowym istnieje za to dużo firm oferujących programy do automatycznego odczytywania testów. Przykładami takich firm są Remark, Omrhome czy Chiron Software. Wszystkie opierają się jednak na odczytywaniu formularzy za pomocą obrazu pozyskanego ze skanera. Nie istnieją praktycznie programy oferujące taką funkcjonalność na platformę Android. Jedynym istniejącym programem jest OMR Evaluator, który potrzebuje specjalnie zaprojektowanych formularzy w celu prawidłowego wykrycia testu.

3. MODUŁ GENERUJĄCY TESTY I KARTY ODPOWIEDZI

Oprogramowanie do odczytywania formularzy z odpowiedziami do poprawnej pracy wymaga ustandaryzowanego formatu formularza. Aplikacja umożliwia wygenerowanie takiego dokumentu automatycznie, na podstawie wskazanej bazy pytań XML i określonych ustawień.

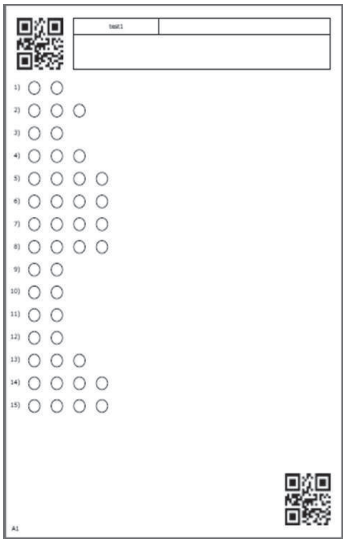
Pierwszym, podstawowym wymaganiem jest możliwość tworzenia oraz edycji baz pytań. Użytkownik programu ma dostęp do przypisanych mu baz pytań, aby z widoku programu edytować pytania i odpowiedzi (łącznie z dodawaniem grafik). Kolejną funkcją programu jest

możliwość generowania losowych testów na podstawie baz pytań. Interfejs graficzny (rys. 1) zawiera pola do określenia liczby pytań oraz wersji testu. Każda wersja testu zawiera pytania i odpowiedzi w kolejności losowej. Po wygenerowaniu i zapisaniu testu w formacie PDF program tworzy pliki z mapowaniem odpowiedzi (indywidualne dla każdej wersji testu) oraz listą poprawnych odpowiedzi (wspólna dla jednego testu).



Rys. 1. Graficzny interfejs użytkownika pozwalający na automatyczne generowanie zadanej liczby testów z losowymi pytaniami i losową kolejnością odpowiedzi

Po wygenerowaniu karty odpowiedzi (rys. 2) i ręcznemu uzupełnieniu jej przez studenta w trakcie sprawdzianu konieczne jest jej poprawne zinterpretowanie przez aplikację.



Rys. 2. Przykładowa karta odpowiedzi

4. MODUŁ ODCZYTUJĄCY KARTY ODPOWIEDZI

W celu umożliwienia odczytania zakreślonych odpowiedzi dokonywana jest analiza obrazu polegająca na rozpoznaniu testu i transformacji obrazu. Przekształcenie obrazu pozwoli na poprawną ewaluację testu nawet przy niskiej jakości obrazie wejściowym. Przykładem takiego obrazu może być wykonanie zdjęcia pod kątem albo aparatem o małej rozdzielczości matrycy kamery. Telefon komórkowy z programem musi mieć zapewnione połączenie internetowe (WiFi lub poprzez sieci komórkowe) w celu połączenia z serwerem i pobraniem listy dostępnych testów. Na jej podstawie dokonuje się weryfikacji, czy wykonane zdjęcie zawiera formularz testowy z istniejącego w bazie testu. Jest to możliwe poprzez odczytanie kodów QR znajdujących się na karcie z odpowiedziami. Po identyfikacji testu program wykorzystuje algorytm wykrywania krawędzi w celu odczytu odpowiedzi. Następnie dokonuje sprawdzenia ich zgodności ze wzorem ewaluacji zgodnie z kluczem.

W celu uniknięcia fałszywych odczytów, pierwszym zadaniem aplikacji jest rozpoznanie, czy na wykonanym zdjęciu faktycznie jest karta z odpowiedziami. W tym celu program sprawdza, czy na zdjęciu znajduje się czworokąt o stosunku boków ok. 30:21 oraz czy w jego lewym dolnym rogu znajduje się kod QR. W pierwszej kolejności obraz zarejestrowany przez kamerę zostaje przekonwertowany do skali szarości, co umożliwia znaczne skrócenie czasu wykonywania algorytmu bez wpływu na ostateczny wynik. Następnie dokonywane jest rozmycie gaussowskie celem usunięcia szumów i zakłóceń oraz filtracja krawędziowa algorytmem Canny [4]. Wykorzystana w bibliotece OpenCV funkcja Canny opiera się na algorytmie opracowanych przez Johna F. Canniego w 1986 roku. Wykorzystuje wielostopniowy algorytm w celu detekcji wielu różnych krawędzi w obrazie. Pierwszym krokiem jest wykorzystanie rozmycia gaussowskiego w celu redukcji szumów. Następnie określane jest natężenie gradientu obrazu z użyciem dowolnej filtracji gradientowej metodą: Sobela, Robertsa lub Prewitta. Każdemu punktowi w obrazie przypisywany jest wektor

$$\vec{\nabla} f_s(x,y) = \begin{pmatrix} \frac{\partial f_s}{\partial x} \\ \frac{\partial f_s}{\partial y} \end{pmatrix} = \begin{pmatrix} g_x \\ g_y \end{pmatrix}, \tag{1}$$

gdzie: x – współrzędne punktu w płaszczyźnie X, y – współrzędne punktu w płaszczyźnie Y, f_s – funkcja obrazu we współrzędnych X i Y.

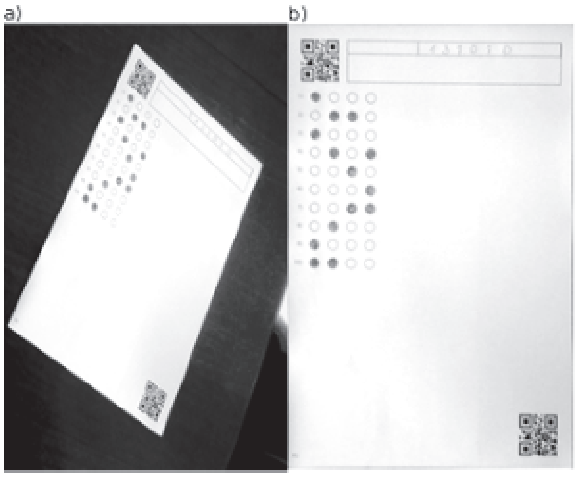
Wyznaczenie wartości gradientu dokonywane jest za pomocą wzoru:

$$M(x,y) = \sqrt{g_x^2 + g_y^2} \qquad \alpha(x,y) = \arctan\frac{g_y}{g_x}, \tag{2}$$

gdzie: M –moduł gradientu obrazu, α – kierunek gradientu obrazu.

Ciągłość krawędzi jest uzyskiwana przez usuwanie niemaksymalnych pikseli. Efektem tego etapu jest ciągła linia złożona z pojedynczych pikseli. Polega na porównaniu gradientu obrazu z jego sąsiadami i wyzerowaniu wartości mniejszych przy zachowaniu lokalnych maksimum w zastosowanej masce. Operację tę powtarza się dla różnych kierunków gradientu. W ostatnim kroku dokonywane jest progowanie z histerezą. Stosowane jest ono do usunięcia nieistotnych krawędzi, żeby nie doszło do przerwania krawędzi o różnym poziomie kontrastu (chwilowo poniżej progu). W takim przypadku zachowuje się krawędzie, dla których w najbliższym sąsiedztwie znajduje się co najmniej jedna krawędź o tzw. silnej krawędzi.

Dalsze przekształcenia mają na celu końcowe wyliczenie i sprawdzenie poprawności stosunku boków prostokąta. Ostatnim etapem jest dokonanie transformacji perspektywy obrazu celem ułatwienia dalszej obróbki. Dopiero efekt tego etapu działania algorytmu (rys. 3) pozwala na analizę odpowiedzi udzielonych przez studenta.



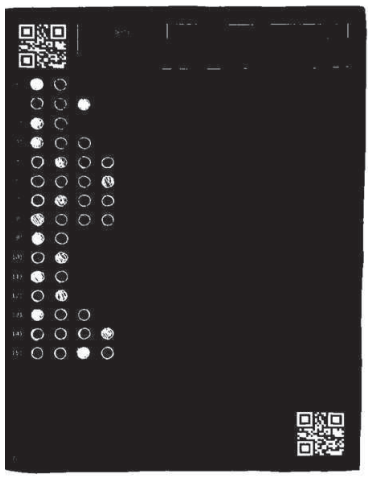
Rys. 3. Zrzut ekranu aplikacji a) przed transformacją płaszczyzny b) po transformacji płaszczyzny

Pola odpowiedzi są okręgami z możliwą różną liczbą odpowiedzi w każdym pytaniu (typowo od dwóch do czterech odpowiedzi). Aby móc je odnaleźć zastosowano binaryzację obrazu testu po transformacji czteropunktowej. Pozwala to oddzielić tło od pierwszego planu i uzyskać dokładniejsze wyniki podczas wyszukiwania krawędzi. Do binaryzacji wykorzystano algorytm Otsu (rys. 4). Zastosowane progowanie pozwala na odczytywanie odpowiedzi zaznaczonych kolorem wyróżniającym się od tła, takimi jak: czarny, niebieski czy czerwony.

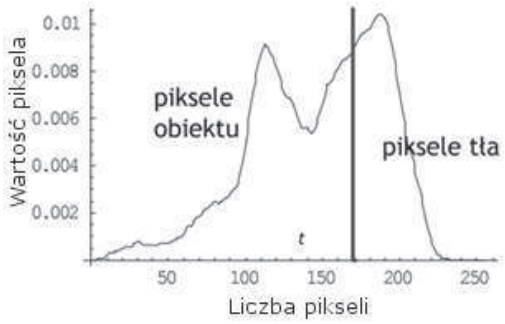
Najbardziej dokładne metody oddzielania obiektu od tła polegają na analizie histogramu obrazu. Taki histogram cechuje się rozkładem dwumodalnym, czyli posiada dwa wyróżniające się wierzchołki (rys. 5). Piksele należące do pierwszej grupy stanowią piksele obiektu, a do drugiej – piksele tła. Trudność algorytmu polega na wyznaczeniu progu, na którego podstawie można podzielić histogram na rozkład pikseli obiektu i tła. W metodzie Otsu wykorzystuje się w tym celu opis statyczny obu klas histogramu przez dwie różne funkcje prawdopodobieństwa [5]. Następnie wyznaczana jest wariancja między klasowa dla każdego możliwego progu histogramu. Punkt, dla którego ta wariancja będzie miała wartość maksymalną będzie optymalnym progiem dla metody Otsu.

Odnalezione kształty są sortowane wedle pozycji (od lewego górnego rogu do prawego dolnego), a następnie usuwane są wszystkie kształty nie spełniające kryteriów. Przyjmuje się, że okrąg z odpowiedzią jest okręgiem o formacie obrazu w zakresie 0,8 – 1,2 do 1 (idealnie 1:1, ale ze względu na błędy transformacji zastosowano dodatkowy margines) oraz średnica okręgu jest większa od 7% wartości szerokości testu. Ostatnia wartość wynika z przyjętych wartości podczas generowania arkuszy testowych – drukowane są one zawsze na kartce A4, więc stosunek wielkości okręgów z odpowiedziami powinien być zachowany. Dalszemu rozwojowi powinna podlegać procedura analizy korekty udzielonej odpowiedzi, która na tym etapie projektu nie została wdrożona.

Drugim istotnym zadaniem modułu, jest poprawne odczytanie numeru indeksu, który jest wpisywany przed studenta ręcznie. Numer ten jest sześciocyfrową liczbą. W celu pełnej automatyzacji sprawdzania testu musi być on również rozpoznany i odczytany przez aplikację mobilną (a następnie wynik testu przypisany określonemu studentowi). W tym celu wykorzystano techniki OCR czyli optycznego rozpoznania znaków. Do realizacji tego zadania użyto biblioteki Tesseract. Działanie algorytmu Tesseract [6], po rozpoznaniu i wyodrębnieniu słów oraz znaków, polega na dwuetapowym procesie. W pierwszym etapie dochodzi do próby rozpoznania każdego słowa, a następnie przesłanie rozpoznanych znaków do przystosowanego klasyfikatora jako dane trenujące. Klasyfikator zwiększa szanse prawidłowego rozpoznania dalszych partii tekstu. Jako, że poziom błędnych odczytów maleje wraz z liczbą wyuczonych słów, stosuje się drugi etap, w którym ponownie rozpoznaje się słowa o niedostatecznie wysokim poziomie rozpoznania.

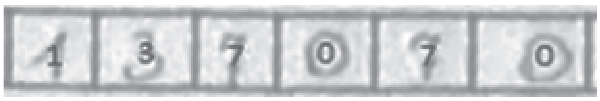


Rys. 4. Karta odpowiedzi po zastosowaniu binaryzacji metodą Otsu



Rys. 5. Karta odpowiedzi po zastosowaniu binaryzacji metodą Otsu

Pełny algorytm Tesseract zawiera także algorytmy do odnajdowania linii, odległości między słowami i literami, rozpoznawania akapitów o różnych wielkości czcionki czy rozpoznawania całych słów. Zostają one pominięte przy odczycie numeru indeksu ze względu na ściśle określony formularz, w którym każda liczba wprowadzona jest w osobnym kwadracie. Jako daną wejściową algorytmu przyjmujemy fragment obrazu zawierający jedną cyfrę, a funkcja powtórzona jest dla wszystkich sześciu cyfr numeru indeksu (rys. 6).



Rys. 6. Efekt pracy algorytmu Tesseract

Przy korzystaniu z biblioteki możemy ustalić białą listę (ang. *white list*), która określa znaki akceptowane podczas

działania algorytmu rozpoznawającego. Pozostałe znaki zostaną zignorowane, zatem przy ustawieniu „0123456789" w białej liście zyskamy pewność, że dane wyjściowe nie będą zawierać liter. Pozwala to także na skrócenie czasu wykonywania algorytmu i zwiększanie jego dokładności rozpoznania. W sytuacji, kiedy aplikacja nie odnajdzie numeru indeksu w bazie, nauczyciel zostanie o tym powiadomiony. Jeżeli taka sytuacja będzie skutkiem niepoprawnego odczytu numeru indeksu, nauczyciel ma możliwość wpisania tego numeru ręcznie w aplikacji.

Ostatnim punktem wymagającym przetwarzania obrazu jest odczyt kodu QR, który umożliwia powiązanie testu z zapisaną na serwerze kartą odpowiedzi. Wartość kodu QR składa się z kilku komponentów: identyfikatora przedmiotu, testu oraz mapowania. Pierwsze dwa umożliwiają szybkie znalezienie określonego zbioru na serwerze. Ostatni określa plik, który został wygenerowany podczas generowania wersji testu, zawierający opis mapowania odpowiedzi. Jest on konieczny ponieważ wybór pytań, ich kolejność oraz porządek odpowiedzi jest inny dla każdej wersji testu. Wymusza to przechowywanie mapowania w celu umożliwienia sprawdzenia pracy.

Dla potrzeb aplikacji generowany jest kod QR przenoszący informację o długości 30 znaków. Ostatnie 10 znaków stanowi kod mapowania (generowana unikalna wartość np. a4fv3858i7h), przedostatnie 5 znaków – identyfikator testu (np. kol1A), a wszystkie znaki poprzedzające stanowią identyfikator przedmiotu. Domyślnie przyjęto jego długość między 4 a 15 znakami (np. BSI_mag_sem3 albo Metro2). Dopuszczalne są znaki kodowania ASCII (ang. *American Standard Code for Information Interchange*).

Aplikacja po rozpoznaniu obszaru testu wydziela fragmenty obrazu, w których znajdują się obydwa kody QR. Są one niezależnie parsowane na łańcuchy tekstowe, a następnie ich wartości są porównywane. Jeżeli są identyczne to wysyłane są do modułu pobierającego mapowanie z serwera. Samo parsowanie kodu odbywa się za wykorzystaniem Barcode API udostępnionym przez Google.

Ewaluacja testu zaczyna się w klasie porównującej odpowiedzi z pobranymi z serwera plikiem mapowania (indywidualny dla każdej wersji testu) oraz plikiem z odpowiedziami (wspólnymi dla wszystkich wersji testu). Po uzyskaniu liczby poprawnie i niepoprawnie zakreślonych odpowiedzi dane zostają przekazane do modułu sumującego punktację. Moduł ten odczytuje z serwera metodę oceniania testu i stosuje ją przy określaniu wyniku.

5. WNIOSKI KOŃCOWE

W ramach prezentowanego projektu powstał kompleksowy program pozwalający nauczycielowi na sprawne przeegzaminowanie dużej liczby studentów. Dzięki zapewnieniu losowości pytań z bazy (lub kilku baz) i zmiennej kolejności odpowiedzi prowadzący może w największym możliwym stopniu zabezpieczyć się przed nieuczciwością zdających. Dodatkową zaletą systemu jest możliwość dodawania obrazów zarówno do pytań, jak i odpowiedzi. Ponadto, aplikacja umożliwia eksport wyników studentów do pliku i ich szybką analizę poprzez zestawienie statystyk dotyczących poszczególnych grup dziekańskich i pytań. Szczególnie ten ostatni aspekt jest istotny, gdyż umożliwia nauczycielowi określenie zagadnień, które są dla studentów najtrudniejsze i które wymagają więcej uwagi podczas omawiania materiału. Stosowanie prezentowanego systemu znacząco ułatwi pracę nauczyciela, przyspieszy proces egzaminowania i oceniania studentów, a także w znaczący sposób przyczyni się do poprawy jakości nauczania.

5. BIBLIOGRAFIA

1. Horst B. P., Shen-pei W.: Handbook of Character Recognition and Document Image Analysis, World Scientific, Singapore 2000.
2. Goodfellow I. J., Bulatov Y., Ibarz J., Arnoud S., Shet V.: Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks, ICLR 2014 International Conference on Learning Representations, 14 – 16 kwietnia, 2014, Banff, Kanada, 6082.
3. Stehno B., Alexander E., Retti G.: Automated Encoding of Digitized Texts, Literary and Linguistic Computing, Vol. 18, Nr 1, New York 2003, s. 77–88
4. Canny J., A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 8, 1986, s. 679-698.
5. Otsu N.: A Threshold Selection Method from Gray-Level Histograms, IEEE transactions on systems, Man, and cybernetics, Vol. 9, No. 1, 1979, s. 62-66.
6. Smith R.: An Overview of the Tesseract OCR Engine, Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), 23 - 26 września, 2007, Curitiba, Brazylia, s. 629-633.

AUTOMATIC SYSTEM FOR STUDENTS' EXAMS

The article presents a system for automatic carrying on students' exams in paper version or on computers. The main aspect of the system is a module that enables performing a traditional paper test, where questions will be randomly chosen from a previously created XML database (or several bases). The three main tasks of the application are: generating and editing tests and saving them into base, generating test versions (based on previously created question base) with random questions and answers, viewing individual and group statistics after evaluation of a test. The test questions are printed on A4 paper. Student set the answers on other piece of paper in special form created by program. The smartphone application (for Android and iOS) enables automatic reading the answers and student index number. The functionality is provided by Otsu thresholding, edge detection and Tesseract algorithm (in case of index number detection). The answers are checked by mapping the test version. Detecting test version is realized by QR code. The program also allows conducting a quiz on computer (via the website) by the Moodle platform. The Moodle platform can use exactly the same questions and answers by importing them from program by special XML file.

Keywords: OCR, OMR, automatic system for students' exams.