Teodora DIMITROVA-GREKOW, Dominik GRODZKI
POLITECHNIKA BIAŁOSTOCKA,
ul. Wiejska 45A, 15-351 Białystok

Rozpoznawanie wzorców cyfrowych za pomocą robota edukacyjnego

Dr inż. Teodora DIMITROVA - GREKOW

Ukończyła studia na Wydziale Elektroniki Politechniki Sofijskiej, Bułgaria w 1991r., obroniła pracę doktorską na Uniwersytecie Technicznym w Wiedniu w 1997r. Jest adiunktem na Wydziale Informatyki przy Politechnice Białostockiej. Jej zainteresowania naukowe to synteza układów programowalnych, robotyka, mechatronika, analiza i przetwarzanie sygnałów.

e-mail: t.grekow@pb.edu.pl

Inż. Dominik GRODZKI

Dominik Grodzki ukończył studia na Wydziale Informatyki Politechniki Białostockiej w 2013r. z tytułem inżyniera na kierunku Informatyka, Politechnika Białostocka, gdzie aktualnie studiuje na drugim stopniu. Od 2012r. pracuje jako programista w technologii Java. Zainteresowania naukowe to tworzenie aplikacji na urządzenia mobilne oraz analiza i przetwarzanie obrazów.

e-mail: d.grodzki@gmail.com

Streszczenie

Artykuł przedstawia system rozpoznający liczby rzymskie przy użyciu edukacyjnego zestawu Mindstorms NXT. Algorytm OCR wybrany do rozpoznania znaków został oparty na klasyfikacji cech. Zaadaptowana wersja algorytmu *Region of Interest ROI* i klasyfikacja cech są głównymi atutami tej pracy. System został skutecznie przetestowany pod wieloma względami. Powstała konstrukcja umożliwiająca skanowanie kartki formatu A4, a obsługujący ją program umożliwia prawidłową interpretację zeskanowanych liczb rzymskich.

Słowa kluczowe: optyczne rozpoznawanie znaków, algorytm ROI, edukacyjny robot.

Digits recognition using an educational robot

Abstract

Pattern recognition is always associated with powerful calculation [1, 2]. A specific branch in this area is *Optical Character Recognition* [3, 4, 5] where one of the most popular techniques is *Feature Extraction*, also known as *Intelligent Character Recognition* [6]. All ICR algorithms are topological [7, 8, 9]. This paper presents an implementation of Roman Number Recognition system realized on LEGO Mindstorms NXT educational robot. The main point is successful minimalistic realization of an on-board pattern recognition system. The NXT platform allows also an easy reconfiguration of the hardware and more building freedom without extra costs (Fig. 1.). An adapted version of the ROI algorithm is implemented [10]. Based on the extracted features (Fig. 2.) a classification of the roman digits is proposed (Fig. 3.). The final stage of the program includes segmentation, end result calculation and visualization of it on the robot screen. The conducted experimental tests proved a 100% efficiency for digit and number recognition having a process in optimal conditions and quite good stability for the optical noises (Fig. 4.) and color chances (Tab. 1). In spite of many drawbacks of the hardware, the implemented system seems very perspective and invokes many ideas toward pattern recognition technics.

Keywords: Optical Character Recognition, Region of Interest algorithm, educational robot.

1. Wstęp

Rozpoznawanie wzorców zawsze jest kojarzone z poważnymi układami obliczeniowymi lub komunikacją z bazą sterująco-obliczeniową [1, 2]. Jako specyficzny obszar w tej dziedzinie Optyczne Rozpoznawanie Znaków (ang. *Optical Character Recognition OCR*) reprezentuje zestaw technik, pozwalających na rozpoznawanie poszczególnych znaków oraz pełnych tekstów w pliku graficznym o postaci rastrowej [3, 4]. Proces ten polega na konwersji zeskanowanego obrazu, zawierającego pismo ręczne lub wydrukowany tekst, do postaci tekstu maszynowego. Jest on powszechnie stosowany jako forma wprowadzania danych z jakiegoś oryginalnego źródła w postaci papierowej, tj. dokumentów, paragonów czy korespondencji [5]. Najbardziej popularne techniki OCR to Dopasowanie Macierzy (ang. *Matrix Matching*) i Ekstrakcja Cech (ang. *Feature Extraction*). Pierwsza metoda porównuje tablicę skanowanego znaku z biblioteką. Technika ta jest prosta i dość rozpowszechniona. Zawiera jednak znacznie więcej ograniczeń dotyczących przede wszystkim formatu znaków. Dla metody drugiej, znaną jeszcze jako *Inteligent Character Recognition* ICR [6] stosowane są algorytmy dotyczące topologii znaków: wyszukanie cyklów [7], obszarów pustych [8], przecięć linii [9] itd.

Artykuł przedstawia implementację systemu rozpoznającego liczby rzymskie z użyciem mobilnego układu edukacyjnego LEGO Mindstorms NXT. Oryginalność niniejszej pracy tkwi w skuteczności minimalistycznej realizacji on-board klasyfikacji znaków w oparciu o ekstrakcję cech, co ma znaczenie z następujących powodów:

- standardowe sensory optyczne zestawów NXT mają bardzo słabe parametry techniczne (zasięg, dokładność, odporność na szumy itd.),
- możliwości obliczeniowe wspomnianych robotów są znaczniej niższe w porównaniu do któregokolwiek smartfona (jednak są wystarczające do prawidłowego rozpoznania cyfr rzymskich, jak praca pokazuje),
- pomimo wymienionych tu wad zostały osiągnięte zadowalające wyniki.

Platforma Mindstorms NXT została wybrana również ze względu na łatwość budowania i modyfikacji konstrukcji. Ostateczne rozwiązanie umożliwia skanowanie kartki formatu A4 z wydrukowanymi liczbami oraz przetwarzanie zeskanowanych danych. Ze względu specyfikę użytego sensora koloru zastosowano pierwotną obróbkę sygnału wejściowego. Zaimplementowana została zaadaptowana wersja algorytmu Obszarów Zainteresowania (ang. *Region of Interest ROI*) [11]. Algorytm wdraża założenia metody rozpoznawania znaków opartej na ekstrakcji cech. Po segmentacji znaków główną częścią projektu jest klasyfikacja pojedynczych cyfr. Metoda oparta jest na cechach wyłonione przez ROI: identyfikacja bazuje na zaznaczonych wolnych obszarach. Końcowy etap eksperymentu zawiera przeliczenia sekwencji pojedynczych cyfr do liczby dziesiętnej i wyświetlenie wyników na ekranie robota.

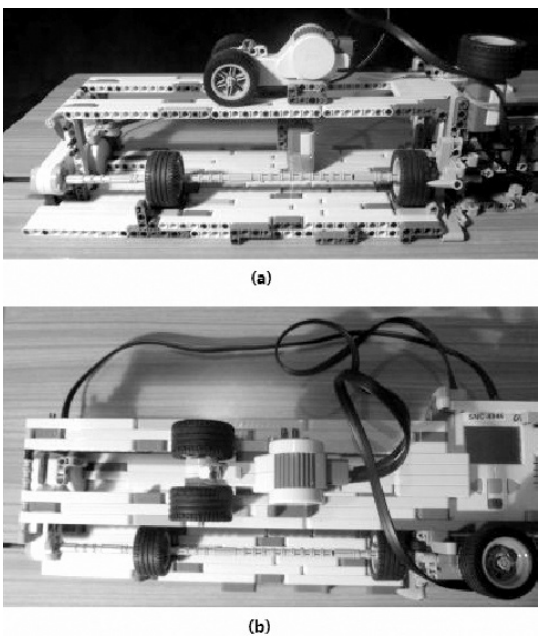
W dalszej części artykułu po kolei będą przedstawione szczegóły konfiguracji sprzętowej i oprogramowania. Zaprezentowane zostaną również testy algorytmu pod względem stabilności oraz wytrzymałości.

2. Struktura systemu

Pierwotne założenia pracy zakładały mobilność robota. Miał on poruszać się po kartce papieru i wyszukiwać na niej cyfry rzymskie oraz je rozpoznawać. Zbudowany w tym celu model nie spełnił jednak oczekiwań eksploatacyjnych. Testy wykazały bowiem duże niedokładności silników NXT, które nakładając się

w miarę trwania jazdy, uniemożliwiały poprawne rozpoznanie cyfr przez robota, zmniejszając jego precyzję.

W związku z tym koncepcja konstrukcyjna została diametralnie zmieniona (rys. 1). Poruszają się dwa elementy budowy: głowica skanująca oraz wałek przesuwający kartkę.



Rys. 1. LEGO skaner (a) widok główny; (b) widok z góry
Fig. 1. LEGO scanner (a) main view; (b) upper view

W ostatecznej, skutecznej wersji robota poza klockami, przewodami i jednostką centralną, do budowy użyto czujnika koloru, czujnika dotyku oraz dwóch silników. Robot posiada cztery istotne elementy struktury: wałek przesuwający kartkę, głowica skanująca, czujnik koloru oraz przycisk uruchamiający i zatrzymujący.

Wałek przesuwający kartkę został zbudowany z dwóch kół, połączonych cienką belką. Ich rozstaw jest dopasowany do rozmiaru A4 skanowanych kartek. Przesuwanie kartki jest możliwe dzięki oponom założonym na koła. Obracanie wałkiem zapewnia jeden silnik, do którego podstawiona jest belka łącząca oba koła.

Głowica skanująca porusza się dzięki jednemu silnikowi oraz dwóm kołom zamontowanym przy nim. Prostą jazdę zapewniają szyny, po których porusza się głowica - są one zbudowane tak, by uniemożliwić ruch głowicy na boki. Wykorzystano je, by zniwelować wpływ niedokładności sprzętowych silników NXT na pracę programu.

Czujnik koloru jest najważniejszym elementem konstrukcji i został umieszczony pod głowicą. Czujnik ten odpowiada za skanowanie kartki. Oklejono go nieprzezroczystą taśmą w celu zminimalizowania wpływu warunków zewnętrznych na odczyt koloru (światło zewnętrzne).

Przycisk uruchamiający i zatrzymujący (START/STOP) składa się z czujnika dotyku oraz koła - w celu zwiększenia powierzchni wciskanej. Użytkownik używa go w celu uruchomienia pracy skanera, a kolejne wciśniecie w dowolnym momencie zakańcza pracę programu.

Ze względu ograniczeń technicznych użytego zestawu, przyjęte zostały **założenia wstępne** dotyczące wchodzących danych. Kartka z danymi musi spełniać następujące wymagania:

- liczby do rozpoznania są koloru ciemnego, najlepiej czarnego - im bardziej kontrastowy kolor do białego, tym lepiej,
- minimalny rozmiar cyfr: 10x20mm, grubość: 1.2mm,
- odstępy miedzy kolejnymi liniami powinny być nie mniejsze niż 25% wysokości liczby.

Wizualizacja skanowanego tekstu ma rozdzielczość podstawową 100 x 64, gdyż taka jest rozdzielczość ekranu kostki NXT. Skanowane dane są na bieżąco wyświetlane na ekranie. Po zeskanowaniu całej kartki lub po zatrzymaniu skanowania przez użytkownika za pomocą przycisku START/STOP, następuje faza rozpoznania liczb, które zostają wypisane również na ekranie, poczynając od lewego górnego rogu.

Kompleksowe przygotowanie danych do etapu rozpoznawania poszczególnych znaków i ostateczna interpretacja liczb składa się z kilku etapów. Podział ten może być odniesiony do klasycznej sekwencji charakterystycznej dla systemów ORC [11], zawiera jednak pewną specyfikę związaną z tym konkretnym przykładem.

3. Przygotowanie danych do przeprowadzenia klasyfikacji cyfr

Pierwszym etapem każdego systemu rozpoznającego wzorce jest zbieranie danych. Wejściowe dane próbkowane są w empirycznie dopasowanym interwale czasowym. Wynikiem końcowym jest wektor danych w postaci numerycznej, które wcześniej zostały przetworzone wstępną binaryzacją surowych wartościach postępujących z czujnika koloru.

Bardzo istotne znaczenie ma obszar działania użytego czujnika. Sensor koloru w zestawach NXT 2.0 pobiera wartość sumaryczną odbitego od powierzchni światła. Pole odbijające może mieć promień od 4mm do 8mm w zależności od odsunięcia sensora od powierzchni.

Skanowanie, czyli zbieranie danych, jest pierwszym etapem całego programu. W tej fazie robot skanuje kartkę papieru i rejestruje kolory. Kartka przesuwana jest przez wałek z mocą silnika 30%. Długość ***l***, o jaką kartka zostaje przesunięta w trakcie jednego obrotu wałka, wyraża wzór (1).

$$l = k * L / W , \tag{1}$$

gdzie: k - współczynnik przesunięcia, wyznaczony empirycznie dla użytej konstrukcji, k = 1,7; L - miara kąta, o jaki może obrócić się silnik poruszający głowicą w trakcie jednego przejazdu; W - szerokość ekranu kostki NXT.

W ciągu jednego przejazdu głowicy dokonywane jest $n*100$ odczytów, równooddalonych od siebie (ekran kostki NXT ma szerokość 100 pikseli), gdzie n to współczynnik skalujący, który domyślnie jest równy 1. Silnik kręcący kołami głowicy jest obracany do momentu, gdy licznik jego kąta obrotu osiągnie wartość ϕ, wyrażoną wzorem (2):

$$\phi = (1/W + 1) * x * L / W , \tag{2}$$

gdzie: W - szerokość kartki/wyświetlacza NXT wyrażona w pikselach; L - miara kąta, o jaki może obrócić się silnik poruszający głowicą w trakcie jednego przejazdu; x - indeks aktualnie analizowanego piksela (od 0 do W-1).

Program przechowuje odczytane dane w tablicy o rozmiarach $n*H$ x $n*W$.

Wstępna obróbka polega na prostym przefiltrowaniu szumów z odczytanych danych. Współczynniki wspomagające ten proces są przede wszystkim zależne od ilości pomiarów na 1 milimetr, czyli od prędkości głowicy oraz od szybkości pomiarów. Wynik obróbki to wygładzony kształt każdej pojedynczej linii skanowanej.

Rozpoznawanie położenia liczb - program ustala ilość liczb oraz współrzędne ich na stronie. Algorytm tutaj zastosowany jest zmodyfikowanym algorytmem ROI [ ]. Ten etap stanowi począt-

kową fazie procesu optycznego rozpoznawania wzorców. Główna idea algorytmu składa się z trzech etapów:

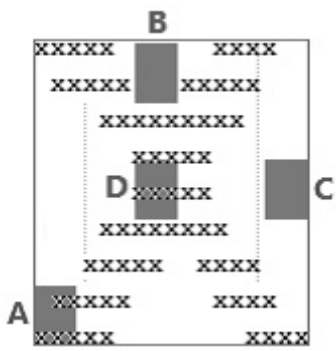
- wskazanie (w oryginalnej wersji - przez użytkownika) czterech współrzędnych ROI,
- wyznaczenie przez algorytm maksymalnych współrzędnych osi x i y, dzięki czemu wyznaczony jest prostokątny obszar do dalszej analizy,
- wyszukanie krawędzi znaku: wykorzystuje się współrzędne górnego lewego oraz prawego dolnego rogu obszaru. Każdy z nich porównuje się z sąsiadującymi pikselami. Jeśli różnica wartości dwóch pikseli przekracza krytyczną wartość *K*, oznacza to, że trafiliśmy na krawędź znaku.

Segmentacja polega na podziale liczb na pojedyncze cyfry.

4. Implementacja algorytmu rozpoznawania cyfr

Implementacja algorytmu rozpoznawania cyfr jest najistotniejszym etapem całej pracy. Następuje tutaj właściwe rozpoznanie najmniejszych składowych danych, czyli cyfr. Za jego realizację odpowiada specjalna funkcja, której parametrami są współrzędne końców badanej cyfry.

Algorytm rozpoznaje cyfrę na podstawie klasyfikacji cech. W tym przypadku specyficznymi cechami cyfr, pozwalającymi odróżnić je od innych, będą kolory niektórych obszarów cyfry. Do rozpoznania każdej cyfry rzymskiej od V do M, wystarczy analiza koloru czterech obszarów cyfry. Obszary te zaznaczono na rys. 2 literami od A do D.



Rys. 2. Obszary cyfry, których analiza pozwala ja rozpoznać oraz oznaczenia
Fig. 2. Regions of interest satisfying the classification algorithm

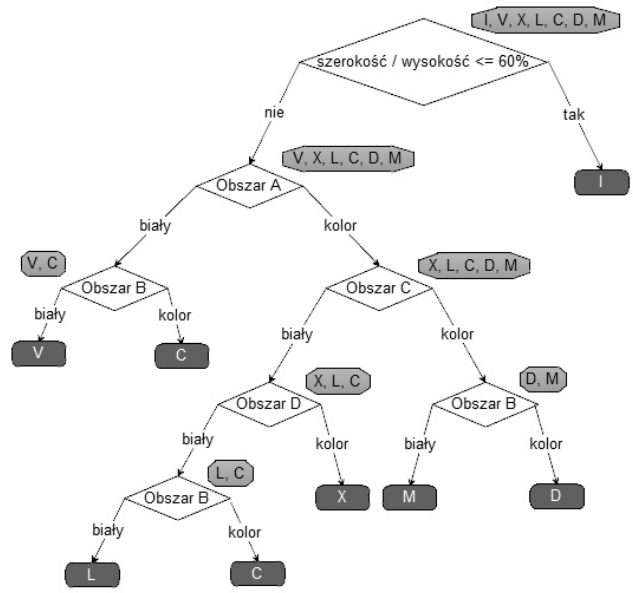
Wyjątkiem jest cyfra I, która zostaje rozpoznana w inny sposób niż reszta cyfr. Wyróżnia się ona tym, że ma najniższy stosunek szerokości *W* do wysokości *H*. Łatwo ją zatem wyłonić już na samym początku: jeśli wyżej wymieniony stosunek (*W/H*) będzie mniejszy lub równy niż 60%, wówczas badana cyfra to I.

Jeżeli pokrycie jest większe lub równe niż 25%, wówczas dany obszar zostaje uznany jako kolorowy. W przeciwnym przypadku zapisuje go do pamięci jako biały.

W momencie, gdy wszystkie cztery obszary są zbadane, następuje faza eliminacji rozwiązań w celu wyłonienia jednego, poprawnego. Odbywa się ona na podstawie drzewa decyzyjnego, znajdującego się na rys. 3.

Przeliczenie liczb końcowych oraz wypisanie wyniku są ostatnimi etapami programu. Należą do fazy przetwarzania końcowego. Przeliczenie liczb rzymskich na arabskie odbywa się według zasady:

Jeżeli w pewnej liczbie rzymskiej Cyfra_2 stoi po Cyfrze_1 i Cyfra_1<Cyfra_2, to liczbę należy pomniejszyć o Cyfre_1. W przeciwnym przypadku do liczby należy dodać Cyfre_1.



Rys. 3. Drzewo decyzyjne stosowane przy rozpoznawaniu cyfry
Fig. 3. Decision tree for digit recognition

Dodatkowo zaimplementowanym elementem systemu jest funkcja zapisująca bieżący odczyt robota do pliku tekstowego, co umożliwia jego późniejsze odczytanie, analizę i końcowe wyświetlenie.

5. Testy praktyczne

Warunki przeprowadzonych testów

System został zaimplementowany oraz przetestowany na następujących warunkach:

1. Liczby są utworzone z cyfr rzymskich ze zbioru {I, V, X, L, C, D, M}
2. Cyfry mają kolor czarny
3. Minimalny rozmiar cyfr: 10x20mm, grubość: 1.2mm
4. Moc silnika głowicy jest równa 50%
5. Współczynnik obrotu wałka przesuwającego kartkę jest równy 1.7
6. Kartka ma kolor biały, jest czysta i jednolita.

Badanie skuteczności rozpoznania pojedynczych cyfr

Na kartkach wydrukowano i napisano zgodnie z warunkami testowymi wszystkie cyfry. Analizę każdej cyfry powtórzono 10 razy. Wyniki testu wykazały 100% skuteczność algorytmu.

Badanie skuteczności rozpoznania liczb

W tym teście została sprawdzona skuteczność systemu w rozpoznawaniu liczb rzymskich. Do testu wybrano następujące liczby:

- MXXII, ze względu na podobieństwo M do X,
- MCDV, ze względu na podobieństwo C do D,
- XXXIV, ze względu na dużą powtarzalność znaku X,
- VII, jako przykład małej liczby,
- CCLIII, ze względu na powtarzalność znaku I oraz podobieństwo I do L,
- DXLIX, ze względu na znajdowanie się mniejszych cyfr przed większymi(I przed X oraz X przed L).

Liczby wybrano tak, by sprawdzić system pod względem różnych kryteriów wymienionych powyżej. Ponadto, do ich stworzenia użyto wszystkich cyfr z zakresu <I, M>.

Wydrukowane lub napisane na kartkach papieru liczby poddano działaniu programu. Analizę każdej liczby powtórzono 10 razy. Stworzony system okazał się niezawodny w rozpoznawaniu testowanych liczb. W 10 próbach nie popełnił ani jednego błędu.

Testy przy zmodyfikowanych warunkach

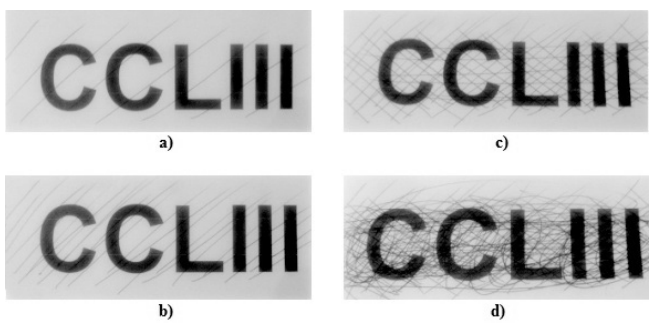
Modyfikacja warunków pozwala szerzej zbadać możliwości systemu. Dzięki niej można sprawdzić różne scenariusze użytkowania aplikacji i ewentualnie zabezpieczyć się przed ich konsekwencjami.

- Zmiana prędkości głowicy:

Analizowany został wpływ wartości mocy silnika głowicy na działanie programu OCR. Liczby poddawane badaniu zostały poprawnie rozpoznane w 100% testów, przy mocy silnika głowicy większej lub równej 20%. Jest to wartość bezpieczna, wykluczająca dodatkowy wpływ poziomu naładowaniu akumulatora systemu.

- Wpływ szumu optycznego:

Zakłócenia w postaci skreśleń zostały losowo naniesione na skanowaną kartkę przy pomocy czarnego długopisu. Ich ilość była stopniowo zwiększana w celu wyznaczenia granicy dawki szumu niewpływającego na pracę programu. Na rys. 4 zaprezentowano kilka etapów przykładowego testu. System bezproblemowo rozpoznawał liczbę w przypadkach widocznych na rys. 4 (a) - (c). Kłopot z rozpoznaniem miał dopiero w sytuacji na rys. 4 (d), gdzie poziom zakłóceń był bardzo wysoki. Zaimplementowane rozwiązanie radzi sobie dobrze z rozpoznawaniem liczb w obecności szumu optycznego. Dopiero bardzo wysoka jego dawka powoduje błędne rozpoznanie.



Rys. 4. Zaszumiona optycznie liczba o poziom szumu: (a) niski, (b) średni, (c) wysoki, (d) bardzo wysoki
Fig. 4. Number with different level of the optical noise: (a) low, (b) middle, (c) high, (d) very high

- Zmiana koloru czcionki:

W teście analizowany jest wpływ poszczególnych składowych kolorów RGB (Red, Green, Blue) czcionki na wynik programu. W tablicy Tab.1 przedstawiono szereg badań dla różnych intensywności poszczególnych składowych koloru.

Tab. 1. Wpływ koloru na rozpoznawalność cyfr
Tab. 1. Impact of color on digit recognition

Cyfra			I	V	X	L	C	D	M
Intenzywność, %	25	R	0	0	0	0	0	0	0
		B	30	40	80	0	10	0	10
		G	20	10	0	0	0	0	20
	50	R	0	0	0	0	0	0	0
		B	80	50	60	80	80	70	70
		G	50	40	60	60	50	40	50
	100	R	100	100	100	100	100	100	100
		B	100	100	100	100	100	100	100
		G	100	100	100	100	100	100	100

Okazało się, że kolory o niskich składowych RGB są słabo rozpoznawane. Jedynym wyjątkiem jest znak X, który w kolorze jasnym niebieskim został rozpoznany poprawnie w 80% prób. Wraz ze wzrostem wartości składowych RGB, robot rozpoznaje cyfry lepiej. Przy najwyższych ich wartościach są one rozpoznawane z taką samą skutecznością jak cyfry czarne: 100%. Najlepiej rozpoznawane są cyfry koloru niebieskiego, najsłabiej - koloru czerwonego.

6. Wnioski

W artykule przedstawiono konstrukcję umożliwiającą skanowanie kartki formatu A4 z liczbami rzymskimi oraz program, który zeskanowane dane przetwarza. Algorytm OCR wybrany do realizacji celu wdraża założenia metody rozpoznawania znaków opartej na klasyfikacji cech. Ponadto, program znajduje liczby według koncepcji wyszukiwania ROI.

W celu weryfikacji poprawności działania stworzonego systemu przeprowadzono szereg testów praktycznych. Wyniki okazały się zaskakująco dobre - robot rozpoznaje bezbłędnie zarówno cyfry, jak i liczby. Badanie wpływu szumu optycznego na wynik programu wykazało, że zaimplementowany system w wysokim stopniu radzi sobie z tego typu zakłóceniami.

Wdrożone rozwiązanie idealnie nadaje się do dalszego rozwoju. Jedną z możliwych dróg jest poszerzenie spektrum rozpoznawanych znaków, na przykład o zbioru liter alfabetu. Naturalnym wydaje się wtedy konieczność zastosowania metod Sztucznej Inteligencji. Do realizacji celu wykorzystano robota edukacyjnego. Ewentualnym kierunkiem rozwoju może być zatem wykorzystanie innego, bardziej zaawansowanego układu. Zwiększyłoby to precyzję pracy.

Publikacja jest finansowana ze środków S/WI/1/13.

7. Literatura

[1] Steven S. S.: The Algorithm Design Manual. Springer, London 2012.
[2] Sesmero M. P., Alonso-Weber J. M., Gutie´rrez G., Ledezma A., Sanchis A.: A new artificial neural network ensemble based on feature selection and class recoding, Springer-Verlag London, Neural Comp&Applic, 21: pp771-783, 2010.
[3] Lei Huang, Zhien Li: Feature-based image registration using the shape context, IJ of Remote Sensing, 31:8, 2169-2177, 2010.
[4] Hiroki Kurashige and Hideyuki Cˆateau: A Method to Construct Visual Recognition Algorithms on the Basis of Neural Activity Data.
[5] Gajer M.: Systemy optycznego rozpoznawania znaków pisma. Pomiary Automatyka Robotyka nr 4, pp 21-25, 2008.
[6] Morita M., Sabourin R., Bortolozzi B., Suen C. Y.: Segmentation and recognition of handwritten dates: an HMM-MLP hybrid approach, IJ of Document Analysis and Recognition, vol. 6, pp. 248-262, 2004.
[7] Dimitrova-Grekow T., Sworowska A.: Rozpoznawanie wzorców cyfrowych pisma ręcznego z użyciem robota edukacyjnego, PAK Pomiary, Automatyka, Kontrola, vol. 59, pp, 611-618, 2013.
[8] Kotani M., Ozawa S.: Feature Extraction Using Independent Components of Each Category, Neural Processing, Vol 22, pp 113-124, 2005.
[9] Rahman1 A. F. R., Fairhurst M. C.: Multiple classifier decision combination strategies for character recognition, IJ of Document Analysis and Recognition, vol 5, pp 166–194, 2003.
[10] Shang Li, Chien Jie, Pin-Gang Su, Yan Zhou: ROI extraction of palmprint images using modified harris corner point detection algorithm, Proc. of the 8th ICIC Theories and Applications, p 479-486, Springer-Verlag Berlin, Heidelberg, 2012.
[11] Cornelius T. Leondes: Image Processing and Pattern Recognition, Academic Press, 1998.

otrzymano / received: 06.02.2014
przyjęto do druku / accepted: 01.04.2014 *artykuł recenzowany / revised paper*