

Mirosław Jabłoński*, Zbigniew Bubliński*

Integracja toru wizyjnego na platformie rekonfigurowalnej

1. Wprowadzenie

1.1. Realizacja algorytmów wizyjnych a rozwój techniki komputerowej

Zagadnienie realizacji algorytmów widzenia maszynowego jest obecne w nauce i technice przynajmniej od połowy ubiegłego stulecia. Ograniczone możliwości maszyn obliczeniowych stanowiły motywację do poszukiwania efektywnych metod implementacji prostych i złożonych algorytmów wizyjnych. Postęp dokonywał się nie tylko poprzez optymalizację metod implementacji algorytmów na komputerach ogólnego przeznaczenia. Poszukiwano również metod akceleracji obliczeń poprzez budowę dedykowanych systemów obliczeniowych i wykorzystanie sprzętowych akceleratorów. Takie prace były również prowadzone w Laboratorium Biocybernetyki Akademii Górniczo-Hutniczej. System Cesaro [20] zbudowany na bazie cyfrowych układów scalonych TTL umożliwiał wykonanie szeregu operacji przetwarzania obrazu w czasie rzeczywistym, co było niemożliwe do osiągnięcia przy wykorzystaniu ówczesnych komputerów osobistych. Dedykowana platforma o nazwie Retina [16] została zbudowana na bazie procesora sygnałowego i cyfrowych układów logicznych. Kolejna heterogeniczna platforma o podobnym przeznaczeniu została wykonana w technologii układów reprogramowalnych PLD i procesora sygnałowego w projekcie [4]. Najnowsze współczesne układy reprogramowalne umożliwiają realizację heterogenicznej platformy obliczeniowej w jednym układzie scalonym.

Wraz z rozwojem technologii wytwarzania układów scalonych i techniki obliczeniowej, moc obliczeniowa komputerów ogólnego przeznaczenia ciągle wzrasta. Jest to efekt zwiększania częstotliwości taktowania oraz zwielokrotnienia jednostek obliczeniowych. Coraz rzadziej napotyka się problem niewystarczającej mocy obliczeniowej podczas realizacji prostych systemów wizyjnych na platformach stacjonarnych.

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie

1.2. Wbudowane systemy wizyjne

Systemy widzenia maszynowego nie ograniczają się jednak tylko do zastosowań stacjonarnych. Zarówno w badaniach, przemyśle czy produktach użytkowych istnieje potrzeba wbudowania algorytmów wizyjnych w pojedyncze urządzenia przenośne lub zorganizowane w sieci sensorów wizyjnych. Dotyczy to między innymi zastosowania w systemach dozoru, sterowania i kontroli jakości na liniach produkcyjnych, diagnostyce medycznej, systemach pomiarowych oraz w szeroko rozumianej wideodetekcji, w tym również wideodetekcji ruchu drogowego. Wybór platformy obliczeniowej podlega różnorodnym kryteriom, zależnym od specyfiki zastosowania. Są to najczęściej: rozmiary, zużycie energii, możliwość przeprogramowania, szybkość działania czy koszt rozwiązania. Najczęściej, wykorzystanie stacjonarnego komputera z uwagi na któreś z wymienionych kryteriów jest kłopotliwe, lub nawet niemożliwe. Najprostsze rozwiązanie polega na zastosowaniu wybudowanego komputera przemysłowego przygotowanego do pracy w określonym reżimie. Rozwiązanie takie cechuje się dobrą przenośnością oprogramowania z platformy stacjonarnej. Znacznie upraszcza to proces prototypowania i konfiguracji. Przeskalowana platforma obliczeniowa cechuje się jednak mocą obliczeniową porównywalną do rozwiązania stacjonarnego lub mniejszą, zależnie od zastosowanej jednostki procesora. Dostosowanie systemu operacyjnego i aplikacji do pracy w czasie rzeczywistym jest możliwe, pod warunkiem odpowiedniego doboru algorytmów i organizacji struktur danych [17].

W stacjonarnych platformach do realizacji algorytmów widzenia maszynowego wykorzystuje się różnorodne techniki zwiększenia wydajności systemów wizyjnych przez akcelerację obliczeń za pomocą specjalizowanych jednostek obliczeniowych (ASIC, procesory DSP, FPGA, GPU), zwielokrotnienie jednostek obliczeniowych lub zastosowanie rozwiązań heterogenicznych [5, 8]. W rozwiązaniach wbudowanych obserwuje się jednak tendencję to wykorzystania homogenicznych elementów obliczeniowych lub zintegrowanych elementów systemów jednoukładowych SoC. Metod przyspieszenia obliczeń upatruje się również w specjalizowanej konstrukcji czujników wizyjnych, optymalnej organizacji danych wizyjnych lub dystrybucji zadań na pracujące współbieżnie węzły sieci czujników wizyjnych [6].

Pomimo dostępnych możliwości obliczeniowych, dla rozwiązań stacjonarnych i opartych na wielordzeniowych architekturach procesorów ogólnego przeznaczenia, problem wydajności platformy obliczeniowej dla zastosowań wbudowanych nadal pozostaje aktualny.

1.3. Ewolucja wbudowanych systemów wizyjnych – *Smart Cameras*

Zarówno w literaturze naukowej, jak i w materiałach informacyjnych producentów kamer i systemów wizyjnych można spotkać się z pojęciem „inteligentna kamera” (*Smart Camera*), które w znacznej części pokrywa się z definicją wbudowanego systemu wizyjnego. Termin ten nie ma jednoznacznej definicji, jednak jest obecnie powszechnie używany w publikacjach naukowych i nomenklaturze produktów dostępnych na rynku. Wartość dodaną idei „inteligentnej kamery” stanowi bezpośrednia integracja czujnika wizyjnego i elementu przetwarzającego w jednym module sprzętowym. Taka architektura poszerza perspektywy

badawcze i daje większe możliwości niż wynikające z prostego złożenia dwóch czynników: wbudowanego elementu obliczeniowego i czujnika wizyjnego. Efekt synergiczny objawia się jednak nie tylko jako skutek połączenia elementu obliczeniowego z czujnikiem wizyjnym. Również zestawienie sieci wielu identycznych [25, 18] lub multimodalnych [6] sensorów wizyjnych daje nowe możliwości i stawia wyzwania nowej natury. Dość wspomnieć o zagadnieniach integracji węzłów sieci (*camera mote*), aspektach energetycznych, finansowych czy choćby etycznych, w przypadku zastosowań w systemach dozoru. Zadaniem *Smart Camer* nie jest tylko akwizycja, przetworzenie i transmisja sygnału wizyjnego, ale przede wszystkim wydobycie użytecznej informacji w czasie rzeczywistym oraz dystrybucja rezultatów analiz czy rozpoznawania. Taki model zakłada znaczą redukcję strumienia danych, co jest zasadniczym atutem zastosowanego podejścia. Architektura *Smart Camer* zawiera w sobie element przetwarzający, który realizuje zarówno akwizycję strumienia wizyjnego przetwarzanie wstępne, jak i pewne etapy analizy obrazu na średnim, a nawet wysokim poziomie. W architekturach inteligentnych kamer wykorzystywane są różnorodne platformy jako elementy przetwarzające: od mikrokontrolerów, poprzez mikroprocesory [6], procesory sygnałowe czy specjalizowane architektury VLIW [13].

2. Sprzętowa realizacja algorytmów wizyjnych

W klasycznym systemie wizyjnym realizowane są trzy etapy przetwarzania danych:

- 1) przetwarzanie wstępne,
- 2) etap pośredni,
- 3) wysoko-poziomowe przetwarzanie obrazu.

Tadeusiewicz i Ogiela [23] opracowali końcowy etap odpowiadający za rozumienie obrazu. Przetwarzanie wstępne sprowadza się do operacji bezkontekstowych na pikselach obrazu i filtracji kontekstowej. Kolejną operacją jest zazwyczaj segmentacja obrazu, która polega na przydzieleniu pikseli do określonej klasy. Choć liczba klas przyporządkowania może być różna, w najprostszym przypadku uzyskujemy wartości binarne: 0 – tło, 1 – obiekt. W bardziej wymagających zastosowaniach do segmentacji, wykorzystuje się metodę przepływu optycznego, adaptacyjne wielomodalne klasyfikatory statystyczne [20, 3] lub elementy metody *Scale Space* [15]. Po segmentacji, obraz poddawany jest najczęściej indeksacji w celu zlokalizowania obiektów w obrazie. Następujące po indeksacji operacje polegają na analizie przestrzennie wydzielonych zbiorów pikseli. Analiza ta obejmuje badanie właściwości kształtów poprzez zastosowanie współczynników kształtów lub wyliczanie momentów dla wydzielonych obiektów [22]. Na podstawie wyodrębnionych masek można wyznaczyć również statystyczne właściwości pikseli należących do obszaru obiektu. Różnorodne struktury torów wizyjnych zawierających indeksację nie ograniczają się do analizy poszczególnych ramek obrazów ale również wykorzystują informację o analizowanych obiektach do śledzenia obiektów w kolejnych ramach poprzez estymację ruchu.

Mocne ograniczenia, jakim podlega wbudowana platforma obliczeniowa i wymagania co do szybkości działania, przekładają się na sposób realizacji algorytmów. Choć są to najczęściej algorytmy niskiego i średniego poziomu, wymagają jednak znacznej mocy obli-

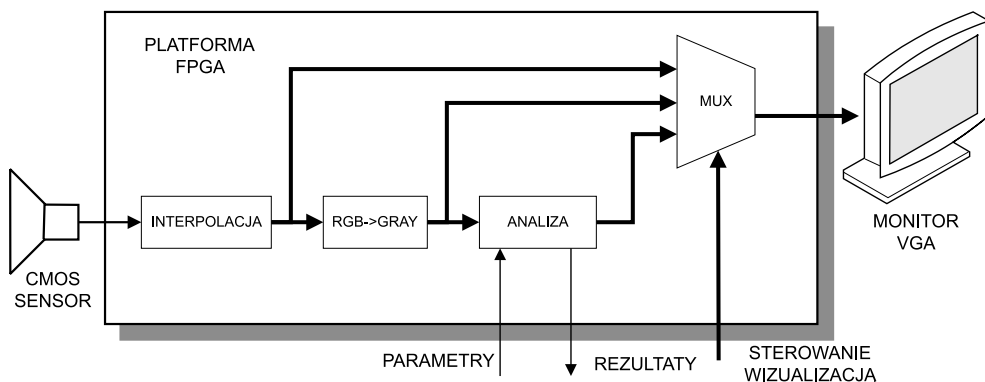
zeniowej z racji na duże ilości przetwarzanych danych (pikseli obrazu). Alternatywny kierunek badań i innowacji polega na odpowiednim doborze algorytmów, struktur i organizacji danych. Implementacja kontekstowych (liniowych i nieliniowych) operacji przetwarzania obrazu (konwolucja, operacje morfologiczne, mediana) w sprzętowych architekturach potokowych [25] jest bardzo efektywna pod warunkiem, że topologiczny (przestrzenny) kontekst (jako lokalne otoczenie aktualnie przetwarzanego piksela obrazu) jest jednoznacznie zdefiniowany. Tor wizyjny zbudowany na bazie takiej idei działa współbieżnie w tempie wyznaczonym przez źródło sygnału wizyjnego. Gwarantuje to płynne przetwarzanie wszystkich pikseli obrazu. Powiększenie liczby operatorów w ścieżce toru nie powoduje spadku wydajności. Zwiększeniu ulega jedynie opóźnienie transportowe które określone jest tutaj sumą długości buforów opóźniających w poszczególnych potokach. Architektura potokowa, o bardzo drobnym ziarnie granulacji (najmniejszym kwantem danych jest piksel obrazu), nie znajduje jednak bezpośredniego zastosowania w realizacji niektórych operacji na obrazie lub przynajmniej niektórych algorytmów wykorzystywanych w ich realizacji:

- Do wyznaczenia nowej wartości piksela niezbędny jest kontekst w zdefiniowany w dziedzinie czasu: np. statystyczna analiza wielomodalna w estymacji tła [20].
- Przestrzenny kontekst potrzebny to wyznaczenia ostatecznej wartości piksela nie jest ograniczony lokalnie: np. segmentacja przez podział obszaru, segmentacja przez rozrost obszaru, segmentacja przez detekcję krawędzi [22].
- Wykorzystany algorytm jest wieloprzebiegowy i wymaga kilkukrotnej analizy każdej ramki (ilość iteracji może nawet zależeć od treści obrazu): indeksacja obrazu metodą „tablicy sklejeń” [22], indeksacja obrazu metodą „pożaru preri” [2].

Podstawowy problem – wspólny dla wyliczonych tutaj przypadków – wiąże się z potrzebą przechowania pośrednich wyników przetwarzania ramki obrazu na kolejnych etapach wykonania algorytmu i jednoczesną obsługą nadchodzących nowych danych z czujnika wizyjnego. Zakładamy przy tym, że wszystkie ramki obrazu, zgodnie z założeniem architektury potokowej, przetwarzane są płynnie bez pominięcia danych. Kluczowe jest więc efektywne zarządzanie ilością niezbędnej pamięci oraz sposób organizacji danych. Zagadnienie realizacji sprzętowej wymienionych klas algorytmów było badane przez autorów pod kątem możliwości strumieniowego przetwarzania danych wizyjnych. W pracy [11], na przykładzie algorytmu estymacji tła, przedstawiono efektywną metodykę zarządzania pamięcią w aplikacjach sprzętowych. Wyniki prac opisanych w [9, 8] potwierdzają możliwość realizacji wieloprzebiegowych algorytmów w połączeniu z algorytmami analiz i rozpoznawania ręcznie pisanymi znakami z wykorzystaniem sieci neuronowych. Wstępne badania przedstawione w pracy [13] pokazują analizę modelu algorytmu wizyjnego o nieznanym *a priori* czasie wykonania pod kątem możliwości strumieniowego przetwarzania danych.

3. Zintegrowany tor wizyjny

W ramach opisywanych prac badawczych zaprojektowano i zbudowano eksperymentalny tor wizyjny składający się z sensora wizyjnego oraz elementu przetwarzającego zawartego w układzie reprogramowalnym FPGA (rys. 1).



Rys. 1. Diagram modułu interpolacji składowych koloru

Dzięki zastosowaniu cyfrowego czujnika wizyjnego, system został w całości zrealizowany w postaci cyfrowej bez stosowania dodatkowego przetwornika analogowo-cyfrowego czy framegrabbera. Dzięki temu znacząco została ograniczona możliwość wystąpienia zakłóceń sygnału wizyjnego. Możliwa stała się też integracja elementu przetwarzającego z sensorem wizyjnym w jednym module.

3.1. Czujnik wizyjny

W zintegrowanym torze wizyjnym wykorzystano barwny czujnik wizyjny MT9V022 (rys. 2) wykonany w technologii CMOS. Wyposażony jest on w wydajny, cyfrowy interfejs szeregowy i równoległy, który umożliwia transmisję sygnału wizyjnego bezpośrednio do FPGA z prędkością 60 fps w formacie Wide-VGA (752×480). Liczba przechwytywanych klatek obrazu może być znacznie większa po zmniejszeniu rozmiarów ramki lub po włączeniu opcji grupowania pikseli. Maksymalny transfer łączy wynosi 26 milionów pikseli na sekundę przy 10-bitowym słowie danych.



Rys. 2. Sensor wizyjny wraz z obiektywem

Czujnik wizyjny posiada również cyfrowy interfejs komunikacyjny IIC, który poprzez konwerter dołączony do magistrali USB umożliwia ustawienie warunków akwizycji oraz parametrów obrazu. Składowe barwnego sygnału wizyjnego kodowane są naprzemiennie w kolejnych pikselach przetwornika analogowo-cyfrowego zgodnie z topologią matrycy Bayera [1].

3.2. Formowanie sygnału wizyjnego

Odtworzenie obrazu barwnego w formacie RGB i monochromatycznego w poziomach szarości z sygnału czujnika, wymaga zastosowania interpolacji brakujących komponentów barwy dla każdego piksela. Programowa realizacja przykładowej formuły interpolacji liniowej (1) została opisana w [10].

$$a) (R,G,B) = [(R1+R2+R3+R4)/4, (G1+G2+G3+G4)/4, B1] \quad (1a)$$

$$b) (R,G,B) = [(R1+R2)/2, (G1+G2+G3+G4+G5)/5, (B1 + B2)/2] \quad (1b)$$

$$c) (R,G,B) = [(R1+R2)/2, (G1+G2+G3+G4+G5)/5, (B1+B2)/2] \quad (1c)$$

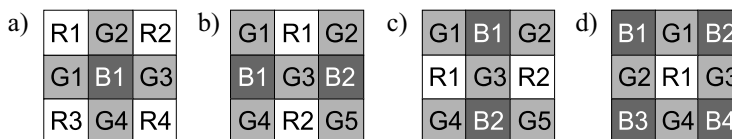
$$d) (R,G,B) = [R1, (G1+G2+G3+G4)/4, (B1+B2+B3+B4)/2] \quad (1d)$$

gdzie:

a), b), c), d) – indeksy konfiguracji matrycy Bayera, dla której należy użyć daną formułę,

R_i, G_i, B_i – wartości pikseli w komórkach matrycy Bayera,

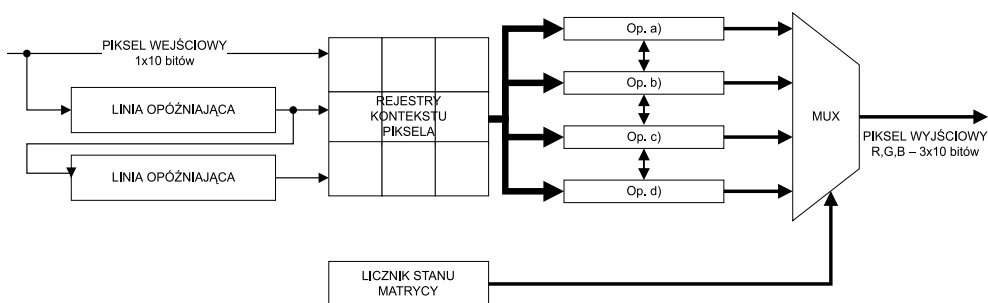
R,G,B – składowe piksela po transformacji z matrycy Bayera.



Rys. 3. Możliwe kombinacje pikseli czujnika aproksymacji oknem 3×3

Zestaw równań (1) odnosi się bezpośrednio do topologii matrycy Bayera przedstawionej na rysunku 3. Kolejne piksele, transmitowane naprzemiennie z czujnika odpowiadają wzorcom a) b) i c) d) w kolejnych liniach obrazu, zależnie od aktualnie ustawionych współrzędnych pierwszego piksela. W celu zaimplementowania interpolacji w strukturze reprogramowalnej dostosowano architekturę potokową wykorzystywaną w strumieniowym przetwarzaniu obrazów (rys. 4). Modyfikacja polega na wykorzystaniu jednego zestawu linii opóźniających oraz czterech operatorów realizujących współbieżnie równania (1a)–(1d).

Wartość składowych R, G i B dla poszczególnych pikseli ustalana jest na podstawie licznika śledzącego topologię matrycy Bayera w trakcie transferu danych. W celu zwiększenia zapasu częstotliwości pracy i zmniejszenia zużycia zasobów, wyniki pośrednie poszczególnych sum równań (1) zostały pogrupowane i są dzielone pomiędzy poszczególnymi operatorami arytmetycznymi.



Rys. 4. Diagram modułu interpolacji składowych koloru

3.3. Przetwarzanie strumienia wizyjnego

Interpolacja obrazu barwnego stanowi pierwszy etap w procesie przetwarzania sygnału wizyjnego przedstawionego na rysunku 1. Tempo pracy wszystkich elementów systemu jest wyznaczone strumieniem danych wizyjnych dostarczanych przez czujnik wizyjny. Na wyjściu systemu pojawia się również sygnał VGA wyświetlający treść strumienia wizyjnego w danym etapie.

W przypadku analizy obrazu, sygnałem wyjściowym może być również skalar lub wektor wyznaczony na podstawie całej ramki obrazu lub jej fragmentu: np. lokalizacja, rozmiar obiektu lub parametry statystyczne ramki wybranego obszaru obrazu. Ponieważ wszystkie moduły działają strumieniowo, wymóg pracy w czasie rzeczywistym jest automatycznie spełniony pod warunkiem, że częstotliwość dostarczania pikseli jest mniejsza od granicznej częstotliwości pracy najwolniejszego z komponentów [5].

3.4. Szacowanie jakości działania systemu wizyjnego

Istotnym parametrem opisującym poszczególne moduły systemu i tym samym jego całość jest opóźnienie transportowe L_p (*latency*). Określono je jako liczbę cykli strumienia wizyjnego, jaka upływa od pojawienia się piksela na wejściu systemu do uzyskania piksela na wyjściu. Dotyczy to sytuacji, gdy rezultatem operacji również jest obraz o topologii zgodnej z obrazem wejściowym.

$$L_p = \left(\frac{(k-1)}{2} X + r \right) f_{pixel}^{-1} \quad (2)$$

gdzie:

- Lp – opóźnienie dla operacji piksel-piksel,
- $k \in \{1, 3, 5, \dots\}$ – pionowy rozmiar okna kontekstu – wartość nieparzysta,
- X – liczba cykli potrzebnych na akwizycję jednej linii obrazu,
- r – opóźnienie dodatkowych rejestrów,
- f_{pixel} – częstotliwość transmisji pikseli w strumieniu wizyjnym $f_g > f_{pixel}$.

Dodatkowy czynnik r odpowiada opóźnieniu wynikającemu z zastosowania dodatkowych rejestrów synchronicznych w elementach obliczeniowych. Mają one na celu takie zbalansowanie czasów propagacji aby spełniony był warunek:

$$f_g > f_{pixel} \quad (3)$$

gdzie f_g – graniczna częstotliwość pracy.

Dla standardowych rozmiarów obrazu, wartość r jest znacznie mniejsza od czynnika związanego z linią opóźniającą. Częstotliwość graniczna f_g (4) wyznaczona jest przez czas propagacji najdłuższej ścieżki kombinatorycznej T_{max} .

$$f_g = \frac{1}{T_{max}} \quad (4)$$

W przypadku gdy rezultatem działania systemu wizyjnego, lub jego części, jest nie obraz, lecz wartość charakterystyczna dla całej ramki obrazu, wówczas:

$$La = \left(\left(\frac{k-1}{2} + Y \right) X + r \right) f_{pixel}^{-1} \quad (5)$$

gdzie:

- La – opóźnienie dla operacji analizy,
- Y – liczba wierszy w analizowanej ramce.

Podobny parametr Ls można zdefiniować dla całego toru wizyjnego jako złożenie elementarnych opóźnień (6).

$$Ls = \sum_i^I L_{p,i} + \sum_m^M L_{a,m} \quad (6)$$

gdzie:

- Ls – sumaryczne opóźnienie toru wizyjnego,
- I – liczba elementów przetwarzania obrazów,
- M – liczba elementów analizy obrazu,
- $L_{p,i}$ – opóźnienie i -tego elementu przetwarzania obrazu,
- $L_{a,i}$ – opóźnienie m -tego elementu analizy obrazu.

4. Rezultaty

Tor wizyjny według opisanego wyżej schematu został zrealizowany w całości na platformie ML501 z układem Virtex5. Czujnik wizyjny podłączono do karty poprzez równoległy interfejs cyfrowy. Do karty podłączono monitor VGA przez interfejs DVI. W implementacji modułów toru wizyjnego wykorzystano oprogramowanie Xilinx Foundation ISE 9.2i. Zestawiono tor wizyjny, który w końcowej części wyznaczał krawędzie na podstawie okna konwolucji 3×3 i współczynniki Sobela.

Tabela 1
Parametry eksperymentalnego toru wizyjnego

Częstotliwość pracy	27 MHz
Szerokość obrazu X	512 pikseli
Wysokość obrazu Y	480 pikseli
k^*	3
r^*	8
Lp^*	20 us
Ls	40 us

* Oznaczone opóźnienia podano dla modułu interpolacji
rozmiary obrazu podano w pikselach.

Do transmisji parametrów poszczególnych modułów w trakcie działania systemu wykorzystano oprogramowanie ChipScope Pro 9.2i oraz interfejs JTAG. W szczególności, konieczne było dopasowanie wartości początkowej licznika wyznaczającego stan matrycy. Wartość progu binaryzacji w algorytmie detekcji krawędzi również była zadawana z komputera PC, aby uzyskać satysfakcjonujące wyniki dla aktualnych warunków oświetleniowych. Tabela 1 przedstawia parametry badanego toru wizyjnego.

5. Wnioski

Przeprowadzone prace projektowe i eksperymenty potwierdzają przydatność platformy rekonfigurowalnej w realizacji całego toru wizyjnego. Co więcej, możliwa jest również bezpośrednia integracja elementu obliczeniowego z czujnikiem wizyjnym. Należy zaznaczyć, że na każdym etapie operacje wykonywane są na pełnym rozmiarze obrazu, bez konieczności redukcji informacji. Zaprezentowany strumieniowy system cechuje się opóźnieniem, które jest znikome w porównaniu z czasem akwizycji pełnej ramki obrazu, który wynosi ok. 16 ms. Przez prostą ekstrapolację można szacować na podstawie równania (6), że opóźnienie będzie rosło liniowo ze wzrostem elementów przetwarzających w torze wizyjnym.

Oprócz zwiększenia stopnia zajętości układu reprogramowalnego i zużycia mocy będzie to jedyny efekt wzrostu stopnia skomplikowania algorytmu wizyjnego. Dalsze prace polegające na integracji algorytmów z klas wymienionych w rozdziale 2 pozwolą ostatecznie zweryfikować przydatność platformy rekonfigurowalnej do realizacji wbudowanego systemu wizyjnego.

Podziękowania

Implementacja w układzie reprogramowalnym została wykonana z wykorzystaniem oprogramowania dostarczonego w ramach donacji z programu Xilinx University Program. Badania zostały wykonane w ramach umowy 11.11.120.612.

Literatura

- [1] Bayer B.E., *Color Imaging Array*. 1975, <http://www.pat2pdf.org/patents/pat3971065.pdf>.
- [2] Bubliński Z., Jabłoński M., Mikrut Z., *Analiza sekwencji filmowych w środowisku VirtualDub w oparciu o platformę FPGA*. Automatyka (półrocznik AGH), t. 10, z. 3, 2006, 323–333.
- [3] Friedman N., Russell S., *Image segmentation in video sequence*. Proc. 13th Conf. Uncertainty in A.I., 1997.
- [4] Gorgoń M., *RETINA – karta dokonująca readresacji i transformacji obrazu w czasie rzeczywistym*. Kraków, AGH, Katedra Automatyki, Laboratorium Biocybernetyki, Nr 94, 2000, 33–40, 49–52 (raport techniczny).
- [5] Gorogon M., *Architektury rekonfigurowalne do przetwarzania i analizy obrazu oraz dekodowania cyfrowego sygnału wideo*. Kraków, UWND 2007.
- [6] Hengstler D., Prashanth S., Fong and H. Aghajan, *MeshEye: a hybrid-resolution smart camera mote for applications*. Proc. 6th International Conference on Information Processing in Sensor Networks (IPSN/SPOTS 2007), Apr. 2007, 360–369.
- [7] Jabłoński M., *Implementacja przekształcenia obrazu do przestrzeni LOG-POLAR w układzie reprogramowalnym FPGA*. Kraków, AGH 2001 (praca dyplomowa).
- [8] Jabłoński M., *Od obrazu do matrycy znaku – implementacja sprzętowa*, *Przetwarzanie i analiza sygnałów w systemach wizji i sterowania*. Katedra Automatyki AGH, Katedra Informatyki Stosowanej Politechniki Łódzkiej, Słok k/Belchatowa, 2002, 38–43.
- [9] Jabłoński M., Gorgoń M., *Handel-C implementation of classical component labelling algorithm*. Proceedings of the EUROMICRO Systems on Digital System Design, 31 August–3 September 2004, Rennes, France, IEEE Computer Society, IEEE, Los Alamitos, 387–393.
- [10] Jabłoński M., *Inteligentna kamera. Podsystem automatycznej kalibracji barwnej*. Automatyka (półrocznik AGH), t. 11, z. 3, 2007, 245–255.
- [11] Jabłoński M., *Reconfigurable fpga-based platform enables real-time video detection*. ITS-ILS'07, 11–12 November 2007, Transportation and Logistics Integrated Systems, Kraków, 2007, 38–47, ISBN 978-83-88309-86-1.
- [12] Jabłoński M., *Hardware architecture for automatic color calibration*. CMS'07, Computer Methods and Systems, 21–23 November 2007, 277–282.
- [13] Kleihorst R., Abbo A., Schueler B., Danilin A., *Camera Mote with a High-Performance Parallel Processor for Real-Time Frame-Based Video Processing*. ICDSC'07 First International Conference on Distributed Smart Cameras, Wiedeń 2007.

- [14] Leeser M., Miller S., Yu H., *Smart Camera Based on Reconfigurable Hardware Enables Diverse Real-Time Applications*. 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'04), 2004, fccm, 147–155.
- [15] Lindeberg T., *Scale-space theory: A basic tool for analysing structures at different scales*. In J. of Applied Statistics, 21(2), 1994, 224–270.
- [16] Mikrut Z., Stanek Z., *Low-cost board for image digitization, remapping and processing*. Automatyka (półrocznik AGH), t. 2, z. 3, 1999, 487–497.
- [17] Mikrut Z., *The Cracovian Videodetector – From Ideas to Embedding*. ItS-ICS, Transportation and Logistic Integrated Systems, Kraków, 2007, 29–37, ISBN 978-83-88309-86-1.
- [18] Morbeel M., Tessens L., Quang Luong H., Prades-Nebot J., Pizurical A., Philips W., *A distributed coding-based content-aware multi-view video system*. ICDSC'07 First International Conference on Distributed Smart Cameras, Wiedeń 2007, 81–86, 355–362.
- [19] Stauffer C., Grimson W., *Adaptive background mixture models for realtime tracking*. Proc. IEEE Conf. Computer Vision and Pattern Recognition, 1999.
- [20] Tadeusiewicz R., Pachwoicz P., *CESARO – system analizy i rozpoznawania obrazów wizualnych*. Informatyka, nr 7/8, 1983, 27–29.
- [21] Tadeusiewicz R., *Systemy wizyjne robotów przemysłowych*. Warszawa, WNT 1992.
- [22] Tadeusiewicz R., Korohoda P., *Komputerowa analiza i przetwarzanie obrazów*. Kraków, Wydawnictwo Fundacji Postępu Telekomunikacji 1997, 243–270.
- [23] Tadeusiewicz R., Ogiela M., *Computers and images: from filtering and other processing procedures, through feature analysis and pattern recognition, up to the automatic understanding of the merit content*. CMS'07 21–23 November 2007, Kraków, Polska, AGH – ONT, 2007, 1–16.
- [24] Zhao F., Liu J., Guibas L., Reich J., *Collaborative signal and information processing: an information directed approach*. Proceedings of the IEEE, vol. 91, no. 8, 2003.
- [25] Wiatr K., Jamro E., *Implementation image data convolutions operations in FPGA reconfigurable structures for real-time vision systems*. Proceedings of International Conference on Information Technology: Coding and Computing, Los Alamitos, USA, 2000, 152–157.