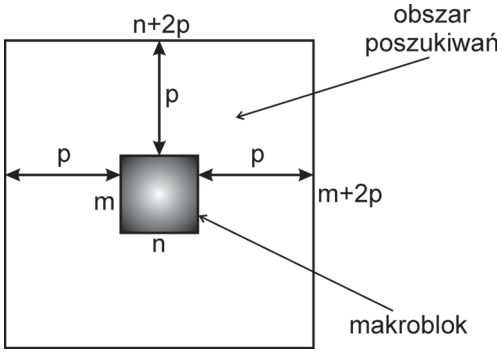
AUTOMATYKA • 2006 • Tom 10 • Zeszyt 3

Agnieszka Dąbrowska*, **, Kazimierz Wiatr*, **

Modyfikacja algorytmu *E3SS* estymacji ruchu na potrzeby implementacji w układach FPGA

1. Wprowadzenie

Estymacja ruchu jest procesem [1, 2], poprzez który znajdowane są odpowiadające sobie bloki pikseli z sąsiednich ramek. W idealnym przypadku zakres poszukiwania takiego bloku jest nieograniczony. Jednak w rzeczywistości byłoby to niepraktyczne, dlatego zakres przeszukiwań (rys. 1) został ograniczony do obszaru wokół bloku w zakresie [–*p*, *p*].



Rys. 1. Obszar przeszukiwań w algorytmach estymacji ruchu

Zatem współrzędne wektora ruchu będą ograniczone warunkami (1) i (2).

$$-p \le u \le p \tag{1}$$

$$-p \le v \le p \tag{2}$$

* Katedra Elektroniki, Akademia Górniczo-Hutnicza w Krakowie
** ACK Cyfronet, Akademia Górniczo-Hutnicza w Krakowie

2. Funkcje celu

Jedną z najczęściej używanych funkcji celu jest różnica średniokwadratowa *MSD* (*Mean-Squared Difference*). Funkcję tę można zdefiniować za pomocą wyrażenia (3).

$$MSD(dx,dy)=\frac{1}{mn}\sum_{i=-n/2}^{n/2}\sum_{j=-m/2}^{m/2}\left[F(i,\,j)-G(i+dx,\,j+dy)\right]^2 \tag{3}$$

gdzie:

$F(i,\,j)$ – kompresowany makroblok,
$G(i,\,j)$ – referencyjny makroblok,
$(dx,\,dy)$ – wektor przesunięcia,
$dx = \{-p, p\}$,
$dy = \{-p, p\}$.

MSD jest funkcją dającą najdokładniejsze wyniki, lecz równocześnie najbardziej złożoną pod względem obliczeniowym. Realizacja funkcji *MSD* wymaga wykorzystania mnożenia każdego piksela makrobloku.

Inną popularną funkcją celu jest średnia różnica bezwzględna *MAD* (*Mean Absolute Diffrence*) reprezentowana przez wyrażenie (4).

$$MAD(dx,dy)=\frac{1}{mn}\sum_{i=-n/2}^{n/2}\sum_{j=-m/2}^{m/2}\left|F(i,\,j)-G(i+dx,\,j+dy)\right| \tag{4}$$

Funkcja *MAD* charakteryzuje się mniejszą złożonością obliczeniową niż *MSD*, lecz również jest mniej dokładna.

Kolejną funkcją celu jest funkcja klasyfikacji różnicy pikseli *PDC* (*Pixel Difference Classification*) [4]. *PDC* charakteryzuje się mniejszą złożonością obliczeniową niż *MSD* oraz *MAD*.

$$PDC(dx,dy)=\sum_i\sum_j T(dx,dy,i,\,j) \tag{5}$$

dla $(dx,dy)=\{-p,p\}$.

Funkcja $T(dx,\,dy,\,i,\,j)$ jest binarną reprezentacją różnicy pomiędzy pikselami definiowaną za pomocą wyrażenia (6)

$$T(dx,dy,i,\,j)=\begin{cases}1\longrightarrow\left|F(i,\,j)-G(i+dx,\,j+dy)\right|\le t\\ 0\longrightarrow \text{w}\quad\text{przeciwnym}\quad\text{przypadku}\end{cases} \tag{6}$$

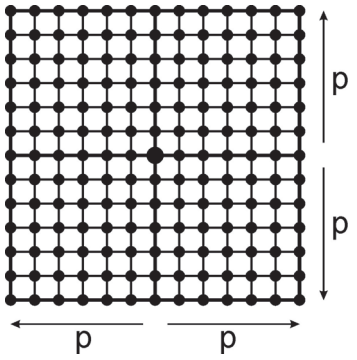
gdzie t – predefiniowana wartość progowa.

Przy tak zdefiniowanej funkcji celu, blok najbardziej zdefiniowany jest to blok z maksymalną wartością funkcji *PDC*.

3. Algorytmy estymacji ruchu

Algorytmy wyszukiwania najbardziej podobnego makrobloku w obrazie referencyjnym można podzielić na dwie grupy. Pierwszą z nich stanowią algorytmy przeszukiwania pełnego. Druga grupa to szybkie algorytmy przeszukiwania.

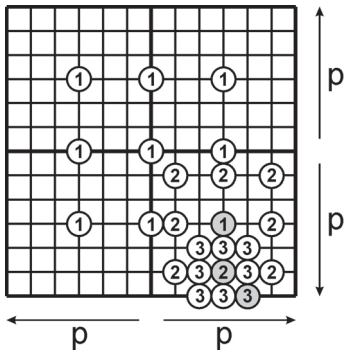
Algorytmem, który najlepiej znajduje najbardziej pasujące bloki, jest algorytm pełnego przeszukiwania *FS* (*Full Search*) [5]. W *FS* (rys. 2) wszystkie możliwe przemieszczenia są używane do obliczenia funkcji celu. Zaletą tego algorytmu jest to, że zostanie znaleziony najlepiej pasujący blok w obszarze przeszukiwania. Wadą algorytmu *FS* jest jego złożoność obliczeniowa wynosząca $O(p^2)$.



Rys. 2. Algorytm przeszukiwania pełnego

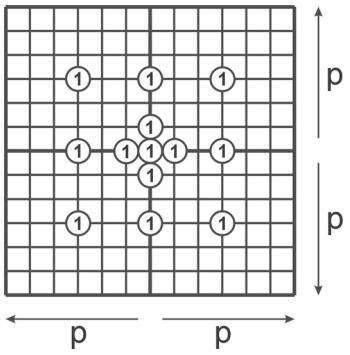
Algorytmami, które cechują się mniejszą złożonością, są szybkie algorytmy estymacji. W przypadku tej grupy algorytmów złożoność obliczeniowa została zredukowana do $O(\log p)$. Jednym z takich algorytmów jest algorytm trójkrokowego przeszukiwania (rys. 3) *TSS* (*Three-Step Search*). W pierwszym etapie obliczana jest funkcja celu dla środka obszaru przeszukiwania, który jest traktowany jako najlepsze dopasowanie. Dalej wyznaczane jest osiem lokacji o następujących przesunięciach względem środka obszaru: $(0, p/2)$, $(p/2, p/2)$, $(p/2, 0)$, $(p/2, -p/2)$, $(0, -p/2)$, $(-p/2, -p/2)$, $(-p/2, 0)$, $(-p/2, p/2)$, dla których wyznaczane są funkcje celu w taki sam sposób, jak to miało miejsce dla środka obszaru przeszukiwania. Spośród obliczonych wartości wybierana jest najlepsza z punktu widzenia zastosowanej funkcji celu. Lokacja, która została uznana za najlepszą, jest również uznana za najlepsze dopasowanie względem makrobloku. W drugim etapie wyznaczane jest kolejne osiem lokacji o przesunięciach względem nowego najlepszego dopasowania wynoszących odpowiednio: $(0, p/2^2)$, $(p/2^2, p/2^2)$, $(p/2^2, 0)$, $(p/2^2, -p/2^2)$, $(0, -p/2^2)$, $(-p/2^2, -p/2^2)$, $(-p/2^2, 0)$, $(-p/2^2, p/2^2)$. Podobnie jak w pierwszym etapie, wyznaczane są funkcje celu dla nowych lokacji. Następnie wybierana jest lokacja z najkorzystniejszą wartością funkcji celu. Wybrana lokacja staje się nowym najlepszym dopasowaniem. Taki proces wyszukiwania najlepszych lokacji trwa, aż krok przesunięcia będzie równy p/p oraz zostanie wyznaczone najlepsze dopasowanie z punktu widzenia zastosowanej funkcji celu przy tym kroku.

W przypadku zastosowania algorytmu *TSS* istnieje możliwość, że lokacja uznana za najlepsze dopasowanie nie będzie równoznaczna z lokacją o absolutnie minimalnej funkcji celu w danym obszarze przeszukiwania. Zaletą *TSS* w porównaniu z algorytmem *FS* jest mniejsza złożoność wynosząca $O(\log p)$.



Rys. 3. Algorytm *TSS*

Algorytmem, który cechuje się większą wykrywalnością małego ruchu niż przy algorytmie *TSS*, jest algorytm efektywnego trójkrokowego przeszukiwania *E3SS* (*Efficient Three-Step Search*) [8, 9].



Rys. 4. Pierwszy etap algorytmu *E3SS*

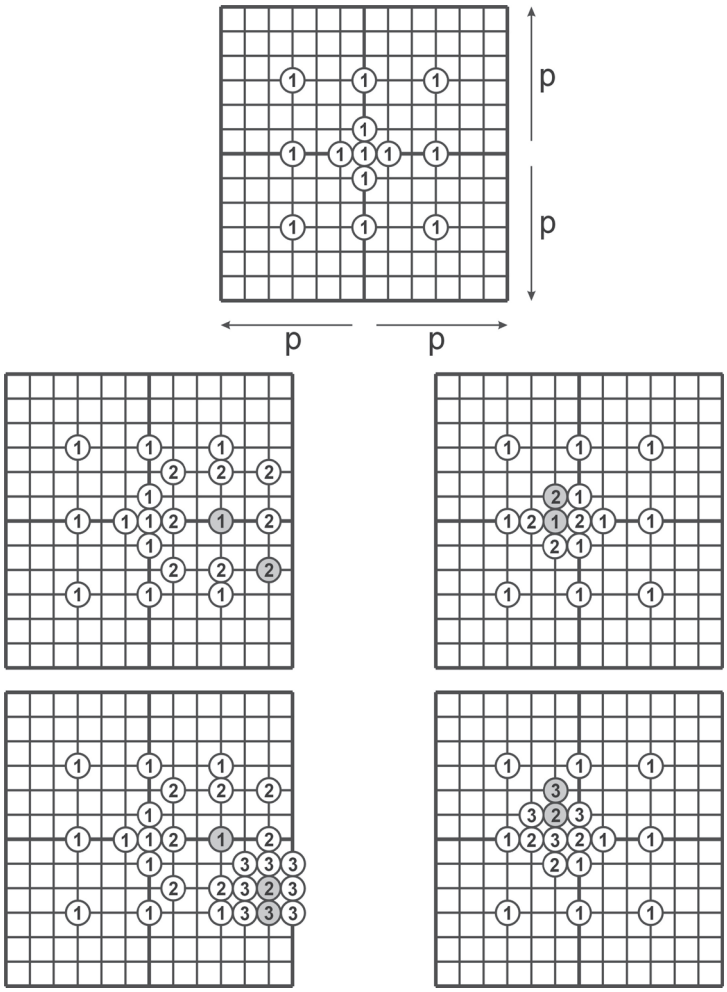
Pierwszym etapem tego algorytmu (rys. 4) jest wyznaczenie podobnie jak w *TSS* ośmiu punktów oraz czterech dodatkowych, leżących w centrum obszaru przeszukiwania, o współrzędnych: (0, 1), (1, 0), (0, –1) oraz (–1, 0). Następnie dla wyżej wymienionych lokacji obliczane są wartości funkcji celu, z których wybierana jest najkorzystniejsza. Dalsze działanie algorytmu jest uwarunkowane lokacją z najkorzystniejszą funkcją celu. Jeżeli wybranym punktem jest środek obszaru przeszukiwania, to zakończone zostaje działanie

algorytmu, a punkt (0, 0) zostaje uznany najlepszym dopasowaniem. Jeżeli najkorzystniejszą funkcją celu charakteryzuje się jeden z ośmiu najbardziej zewnętrznych punktów, to działanie algorytmu jest takie jak w przypadku algorytmu *TSS*. Jeżeli najkorzystniejszą funkcją celu cechuje się jedna z czterech lokacji wokół środka obszaru przeszukiwania, to punkt ten zostaje uznany za chwilowe najlepsze dopasowanie. Następnie zostają wyznaczone wokół niego cztery nowe lokacje o przesunięciach: (0, 1), (1, 0), (0, –1) oraz (–1, 0), dla których obliczane są funkcje celu. Jeżeli najkorzystniejszą funkcją celu charakteryzuje się jedna z nowo wyznaczonych lokacji, to uznawana jest za chwilowe najlepsze dopasowanie i wyznaczane są wokół niej kolejne cztery lokacje na takiej samej zasadzie jak wcześniej. Działanie algorytmu trwa do momentu, gdy nowo wyznaczane lokacje nie będą miały korzystniejszej funkcji celu niż lokacja uznana za chwilowe najlepsze dopasowanie. W takim przypadku lokacja ta zostanie końcowym najlepszym dopasowaniem.

4. Modyfikacje algorytmu *E3SS*

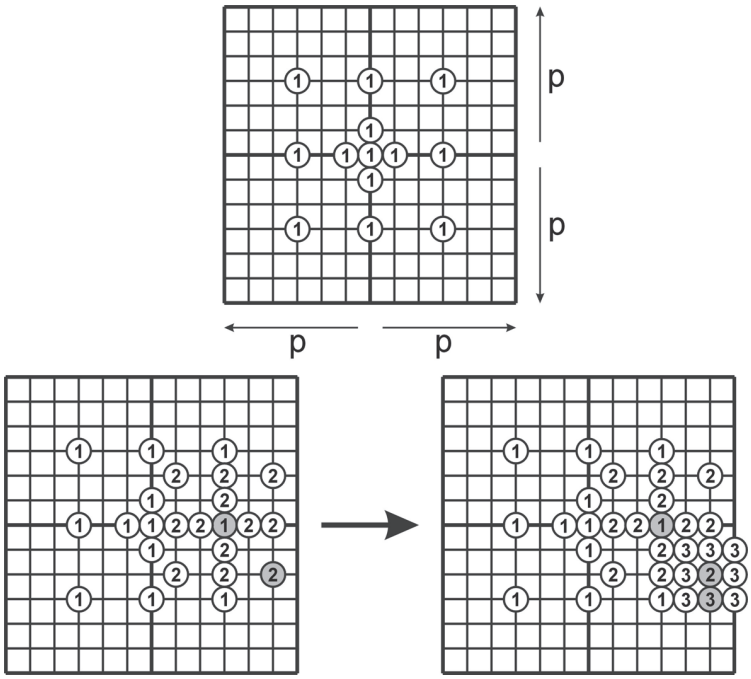
Algorytm *E3SS* nie posiada zdefiniowanej liczby etapów, jakie wystarczają do wyznaczenia lokacji będącej najlepszym dopasowaniem z punktu widzenia zastosowanej funkcji celu. Algorytm efektywnego trójkrokowego przeszukiwania może zakończyć swoje działanie już w pierwszym etapie, jak również, koniec działania algorytmu może nastąpić dopiero po kilku lub nawet kilkunastu etapach. W przypadku sprzętowej implementacji kilku jednostek realizujących równolegle dany algorytm, znaczące są różnice między czasem realizacji przetwarzania danych przez poszczególne jednostki.

Jedną z możliwych modyfikacji algorytmu (rys. 5) *E3SS*, zmniejszającą czas potrzebny do przetworzenia danych przychodzących, jest ograniczenie liczby etapów związanych z lokacjami wokół środka obszaru przeszukiwania lub lokacji uznanej w poprzednim etapie jako najlepsze dopasowanie. Pierwszy etap działania tak zmodyfikowanego algorytmu przebiega w identyczny sposób jak w algorytmie efektywnego trójkrokowego przeszukiwania. Część algorytmu, odpowiadająca za wyznaczanie ośmiu zewnętrznych lokacji na takich samych zasadach jak w przypadku algorytmu trójkrokowego przeszukiwania, pozostała niezmieniona. Jeżeli po pierwszym etapie najkorzystniejszą funkcją charakteryzuje się jedna z czterech lokacji wyznaczonych wokół środka obszaru przeszukiwania, to postępowanie jest takie samo jak dla *E3SS*. Drugi etap tej części algorytmu pozostał bez zmian. Modyfikacja dotyczy natomiast trzeciego etapu związanego z lokacjami wewnętrznymi, który jest jednocześnie ostatnim etapem działania tej części zmodyfikowanego algorytmu. Jeżeli działanie algorytmu nie zostało przerwane w poprzednich etapach, to wyznaczane są cztery nowe lokacje wokół wyznaczonego chwilowego najlepszego dopasowania. Następnie obliczane są funkcje celu dla wyznaczonych lokacji i porównywane z funkcją celu chwilowego najlepszego dopasowania. Wybierana jest lokacja z najkorzystniejszą funkcją celu i staje się ona ostatecznym najlepszym dopasowaniem. Algorytm kończy swoje działanie, w przeciwieństwie do pełnego *E3SS*, nawet jeżeli lokacja z najkorzystniejszą funkcją celu nie jest jednocześnie lokacją uznaną w poprzednim etapie za chwilowe najlepsze dopasowanie.



Rys. 5. Modyfikacja algorytmu *E3SS* z ograniczeniem do trzech etapów (*E3SS* mod.1)

Działanie algorytmu efektywnego trójkrokowego przeszukiwania, oprócz ograniczenia ilości etapów, można również zmodyfikować pod względem wykorzystywania dostępnych zasobów (rys. 6). Z punktu widzenia szybkości odpowiedzi na nadchodzące dane zaimplementowanego sprzętowo algorytmu, pożądane jest zrównoleglenie obliczeń. Dla algorytmu *E3SS* możliwe jest zrównoleglenie obliczania funkcji celu dla lokacji wyznaczanych w poszczególnych etapach działania algorytmu. W pierwszym etapie *E3SS* obliczanych jest równolegle 13 funkcji celu dla 13 różnych lokacji (bez konieczności wprowadzania etapów pośrednich odpowiadających za obliczenie części funkcji celu). W kolejnych etapach, spośród 13 jednostek liczących funkcje celu używane jest osiem lub cztery jednostki, w zależności od decyzji podjętych w pierwszym etapie działania algorytmu.

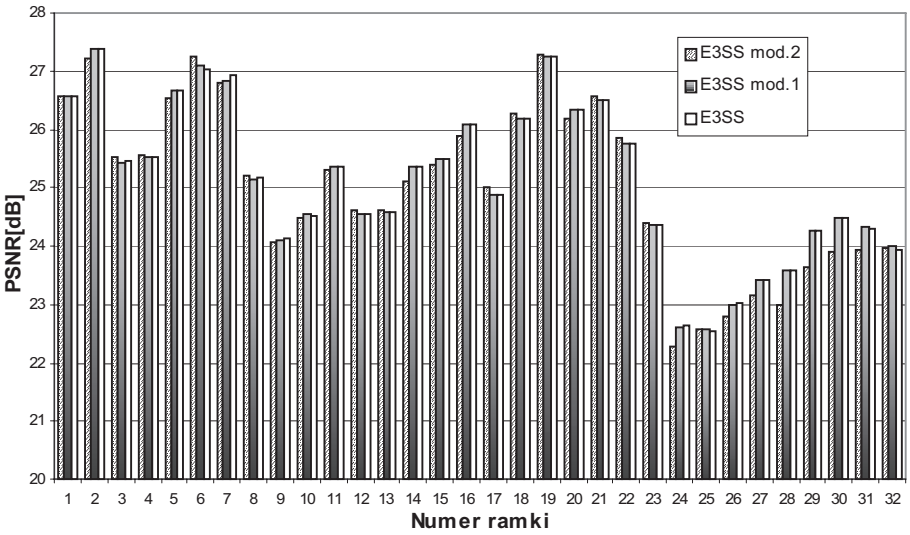


Rys. 6. Modyfikacja algorytmu *E3SS* z ograniczeniem do trzech etapów oraz wykorzystaniem wszystkich jednostek obliczających funkcje celu (*E3SS* mod.2)

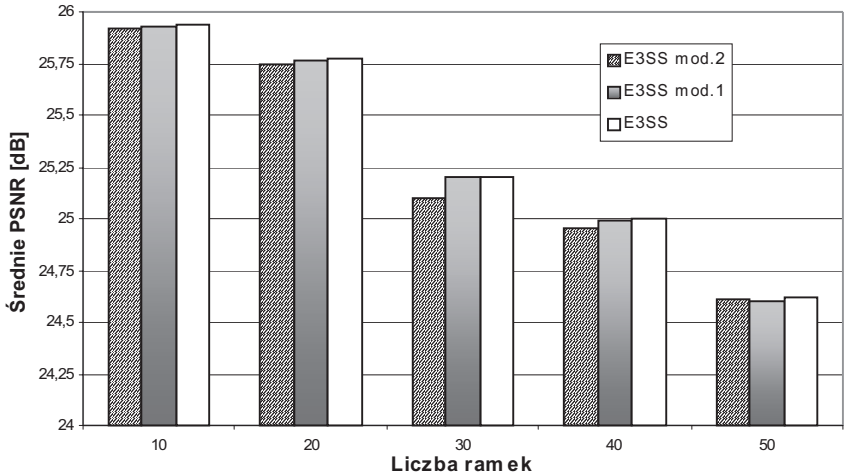
Proponowana modyfikacja polega na rezygnacji z podjęcia decyzji, które lokacje będą wyznaczane w kolejnym etapie: zewnętrzne czy wewnętrzne. Prowadzi to do uproszczenia logiki sterującej działaniem algorytmu. W pierwszym etapie działania algorytmu wyznaczana jest lokacja z najkorzystniejszą funkcją celu. Jeżeli daną lokacją jest środek obszaru przeszukiwania, to algorytm kończy działanie. Jeżeli najkorzystniejszą funkcją celu charakteryzuje się jedna z pozostałych lokacji, to zostaje ona chwilowym najlepszym dopasowaniem i jednocześnie „środkiem" dla kolejnego etapu, w którym wyznaczane jest osiem lokacji zewnętrznych z krokiem $p/4$ oraz 4 lokacje wewnętrzne z krokiem równym 1. Podobnie jak w pierwszym etapie, wyznaczana jest lokacja z najkorzystniejszą funkcją celu z punktu widzenia zastosowanego kryterium. Jeżeli lokacją tą jest chwilowe najlepsze dopasowanie etapu poprzedniego, to algorytm kończy działanie, w przeciwnym razie algorytm przechodzi do następnego etapu, w którym wyznaczane jest osiem lokacji zewnętrznych z krokiem $p/8$ oraz, tak jak poprzednio, cztery lokacje wewnętrzne z krokiem równym 1. Lokacja z najkorzystniejszą funkcją celu zostaje ostatecznym najlepszym dopasowaniem dla makrobloku z obrazu obecnie przetwarzanego.

Zmodyfikowane algorytmy zostały przetestowanie przy użyciu podstawowej sekwencji testowej „tennis" [11]. Uzyskane parametry zmodyfikowanych algorytmów w porównaniu z oryginalnym algorytmem efektywnego trójkrokowego przeszukiwania przedstawiają wykresy na rysunkach 7 i 8. Algorytm z ograniczeniem etapów przetwarzania cechuje się

zbliżoną wartością szczytowego stosunku sygnału do szumu *PSNR* do wartości *PSNR* oryginalnego algorytmu *E3SS*. Algorytm „*E3SS* mod.2" lepiej radzi sobie z wyznaczaniem najlepszego dopasowania w przypadku scen z małym ruchem, natomiast gorsze wartości *PSNR* otrzymywane są dla scen w większym ruchem. Jednak średni szczytowy stosunek sygnału do szumu dla tak zmodyfikowanego algorytmu zrównuje się ze średnim *PSNR* algorytmu *E3SS* przy sekwencjach zawierających większą liczbę ramek.



Rys. 7. Wykres zależności PSNR dla poszczególnych ramek zmodyfikowanych algorytmów *E3SS*



Rys. 8. Wykres zależności średniego szczytowego stosunku sygnału do szumu w zależności od liczby ramek zawartych w sekwencji testowej

5. Podsumowanie

Wyniki otrzymane dla algorytmu *E3SS* oraz algorytmu „*E3SS* mod.1" są zbliżone. Zatem z punktu widzenia implementacji sprzętowej, lepszym algorytmem jest zmodyfikowany algorytm *E3SS* z ograniczoną liczbą etapów wymaganych do wyznaczenia najlepszego dopasowania. Maksymalny czas potrzebny do przetworzenia danych jest równy czasowi trzech etapów działania zmodyfikowanego algorytmu.

Zaletą zaproponowanej modyfikacji „*E3SS* mod.2" jest taki sam maksymalny czas wymagany do przetworzenia danych wejściowych jak w przypadku poprzedniej modyfikacji. Dodatkowym atutem tego algorytmu jest prostsza sprzętowa implementacja logiki sterującej działaniem algorytmu oraz wykorzystanie wszystkich dostępnych jednostek odpowiadających za obliczanie funkcji celu dla poszczególnych lokacji. Jednak algorytm „*E3SS* mod.2" w porównaniu z *E3SS* uzyskuje korzystniejszy szczytowy stosunek sygnału do szumu w przypadku sekwencji cechujących się ruchem na małym obszarze. Dla sekwencji o większym ruchu lepszymi algorytmami są *E3SS* oraz „*E3SS* mod.1".

Literatura

[1] Furht B., Greenberg J., Westwater R.: *Motion estimation algorithms for video compression.* Kluwer Academic Publishers, London 1997

[2] Symes P.D.: *Video compression: Fundamental compression techniques and an overview of the JPEG and MPEG compression systems.* McGraw-Hill, New York 1998

[3] Agha S., Dwayer V.M.: *Algorithms and VLSI Architectures for MPEG-4 Motion Estimation*. Electronic Systems and Control Division Research 2003, 24–27

[4] Lee S., Kim J., Chae S.: *New Motion Estimation Algorithm Using Adaptively Quantized Low Bit-Resolution Image and Its VLSI Architecture for MPEG-2 Video Encoding.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 8, no. 6, 1998, 734–744

[5] Ferguson T.: *CSE5303 – Digital Video Coding and Compression.* www.csse.monash.edu.au/~timf/cse5302/motion.pdf

[6] Lam T.: *Motion compensated prediction for video coding.* http://hitchcock.dlt.asu.edu/media3/a_spanias/mmedia/pdf/PDFmotion-compensation.PDF

[7] Sorwar G., Mushed M., Dooley L.: *Fast Block-Based True Motion Estimation Using Distance Dependent Thresholds.* Journal of Research and Practice in Information Technology, vol. 36, no. 3, 2004, 157–169

[8] Lap-Pui Chau, Xuan Jing: *Efficient three-step search algorithm for block motion estimation in video coding*. ICASSP '03, vol. 3, 421–424

[9] Yen-Chieh Ouyang, Guang-Tzuo Lu: *A Fast Objected-Base Efficient Three-Step Search Algorithm for Block Motion Estimation*. http://www.icss2005.isu.edu.tw/icss2005/PDF/CT512871.pdf

[10] www.xilinx.com

[11] http://bmrc.berkeley.edu/ftp/pub/multimedia/mpeg/EncodeData/tennis.tar.gz