Barometr Regionalny
Nr 3(21) 2010

Symulacja sieci DiffServ z różnicowaniem priorytetu pakietów na podstawie parametru Hursta strumienia obrazu

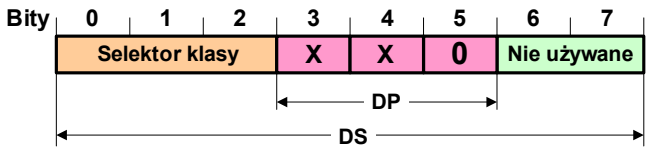
Zbigniew Omiotek
Wyższa Szkoła Zarządzania i Administracji w Zamościu

Streszczenie: *W pracy przedstawiono wyniki symulacji strumieniowej transmisji obrazu ruchomego przez sieć DiffServ. Zaprezentowano także metodykę badania sekwencji wizyjnych wykorzystującą pakiet EvalVid oraz przedstawiono możliwość symulacji sieci z różnicowaniem jakości usług za pomocą symulatora sieciowego NS-2. Otrzymane wyniki pozwoliły obliczyć podstawowe parametry jakości obrazu ruchomego odebranego poprzez sieć, takie jak PSNR, MOS, straty, opóźnienie oraz fluktuację opóźnienia pakietów. Efekty końcowe badań pokazują, iż zastosowanie proponowanej koncepcji polegającej na ustawianiu priorytetu pakietów w sposób dynamiczny prowadzi do poprawy jakości przesyłanego obrazu w porównaniu z tradycyjnym podejściem statycznym.*

Wstęp

Dotychczasowe metody zapewnienia jakości usług w architekturze usług zróżnicowanych (*Differentiated Services*, DiffServ) nie uwzględniają samopodobieństwa i procesów długoterminowych zachodzących w sieci [Willinger i inni 1994, Crovella i Bestavros 1997, Park i inni 1997]. Obecność tych procesów powoduje degradację wydajności sieci, dlatego powinna być ona brana pod uwagę przy projektowaniu nowych mechanizmów poprawy gwarancji QoS (*Quality of Service*). W klasycznym modelu DiffServ przypisanie strumienia do klasy usług oraz ustawienie preferencji w ramach danej klasy odbywa się w sposób ręczny i ma charakter statyczny [Grossman 2002, Heinanen i inni 1999]. W efekcie pakiety oznaczone niższym prawdopodobieństwem odrzucenia mogą być zawsze traktowane preferencyjnie niezależnie od rzeczywistych potrzeb. Nowe podejścia wprowadzają wprawdzie koncepcję dynamicznej klasyfikacji strumieni wejściowych, lecz wykorzystują do tego celu parametry krótkoterminowe jako zmienne sterujące dla procesu klasyfikacji[1]. Obie drogi prowadzą do nieefektywnego wykorzystania ograniczonych zasobów sieci.

Rozwiązaniem tego problemu może być metoda dynamicznego dostosowania poziomu usług oferowanych przez sieć do długoterminowych cech przesyłanego strumienia obrazu[2]. Do tego celu można wykorzystać pole *Drop Precedence* (DP) w nagłówku pakietu IP (rys. 1).



Rys. 1. Struktura pola DP

[1] Por. pozycje literaturowe: Bay 2005, Boucadair i inni 2007, Ahmed i inni 2005, McAllister i inni 2010, Chen i inni 2007.
[2] Metodę tę wraz z zastosowanym algorytmem pomiarowym dokładnie scharakteryzowano w niepublikowanej pracy Z. Omiotka *Dynamiczne zarządzanie QoS podczas przesyłania obrazu ruchomego w sieci DiffServ*.

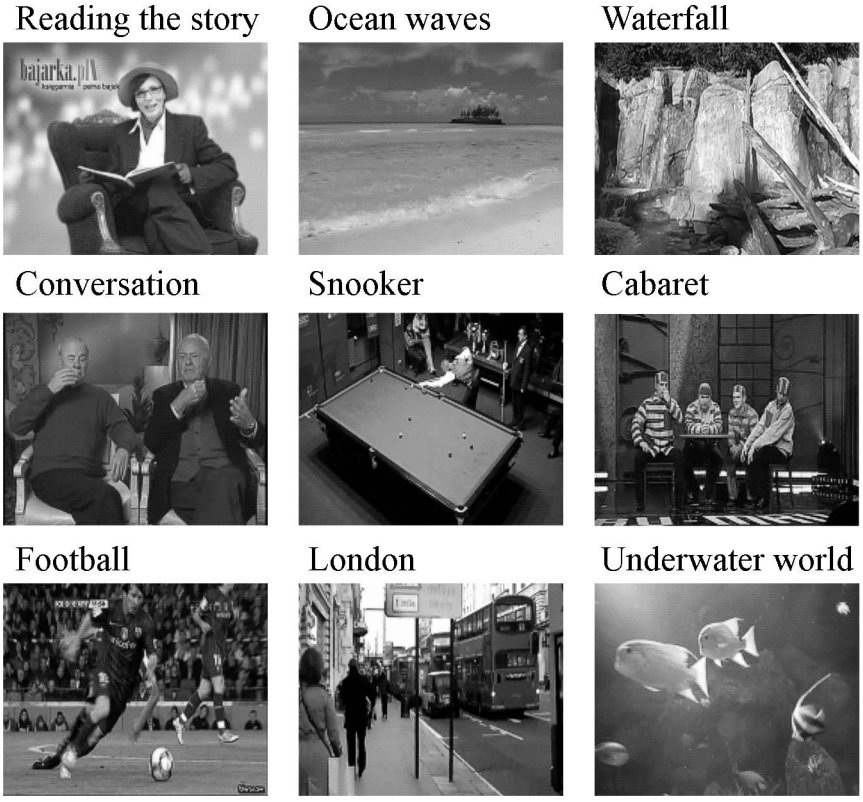
Pole DP może być ustawiane dynamicznie na podstawie pomiaru własności długoterminowych źródłowego strumienia obrazu wprowadzanego do sieci. Poziom zaburzeń obecnych w strumieniu można wyrazić za pomocą parametru Hursta, a następnie odwzorować na wielkość priorytetu zakodowanego w polu DP. Dzięki temu można zrealizować adaptacyjne różnicowanie preferencji poszczególnych strumieni w ramach tej samej klasy usług AF PHB w zależności od wielkości zaburzeń obecnych w danym przepływie. Zastosowanie długoterminowego parametru Hursta jako kryterium klasyfikacji sprawi, że sposób traktowania pakietów oznaczonych daną wartością priorytetu będzie spełniał swoje zadanie również w większej skali czasu.

W niniejszym artykule zaprezentowano wyniki badań symulacyjnych, które pozwoliły porównać podstawowe parametry określające jakość przesyłanego obrazu. Należą do nich: PSNR (*Peak Signal to Noise Ratio*), MOS (*Mean Opinion Score*), straty, opóźnienie oraz fluktuacja opóźnienia pakietów. Porównania dokonano dla dwóch przypadków. W pierwszym z nich priorytet pakietów w ramach danej klasy usług AF PHB był ustawiany statycznie. W przypadku drugim miało miejsce dynamiczne ustawianie priorytetu w zależności od aktualnej wartości parametru Hursta strumienia ramek. W badaniach zostały wykorzystane różne sekwencje wizyjne oraz moduły symulatora pozwalające zaimplementować rzeczywistą funkcjonalność sieci DiffServ.

W rozdziale pierwszym niniejszego artykułu scharakteryzowano zastosowaną metodologię badań wykorzystującą pakiet EvalVid. Przedstawiono w nim również sposób symulacji sieci z różnicowaniem jakości usług za pomocą symulatora sieciowego NS-2. W rozdziale drugim wyjaśniono, w jaki sposób została zasymulowana funkcjonalność modułu zarządzania priorytetem pełniącego kluczową rolę w całym systemie. Rozdział trzeci opisuje konfigurację zastosowanej sieci pomiarowej. Wyniki badań symulacyjnych przedstawiono w rozdziale czwartym, a całość pracy kończy podsumowanie.

1. Metodyka badań zastosowana w pracy

W ramach przygotowań do przeprowadzenia badań zgromadzono zestaw klipów wideo o różnej dynamice zmian sceny i akcji. Etap ten był dość żmudny i pracochłonny, ponieważ istotne było tutaj skompletowanie sekwencji charakteryzujących się jak najszerszym zakresem złożoności ruchu. Przykładowe ramki należące do tych sekwencji, odzwierciedlające ich charakter zostały w sposób zbiorczy przedstawione na rys. 2.



Rys. 2. Przykładowe ramki klipów wideo wykorzystanych podczas badań

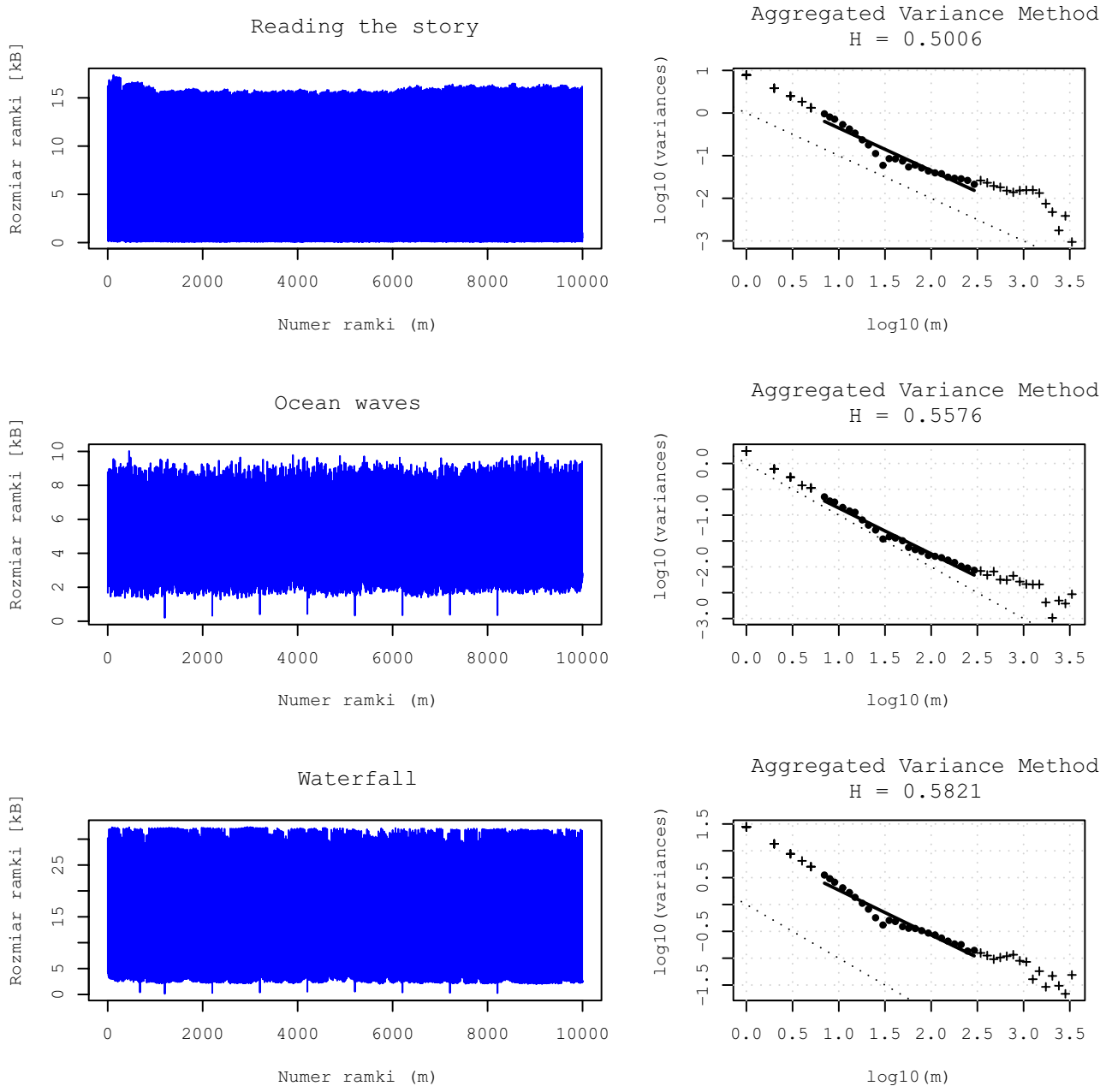
Wszystkie klipy zostały zdekodowane i zapisane w postaci sekwencji nieskompresowanych ramek YUV. Każdy plik zawierał 10 000 ramek w formacie CIF zapisanych z szybkością 30 ramek na sekundę, co odpowiadało materiałowi filmowemu o długości 5 minut 33,3 sekundy. Zestawienie wspomnianych wyżej klipów wraz z krótkim opisem oraz wartością parametru Hursta zamieszczono w tab. 1.

Tab. 1. Zestawienie klipów wideo wykorzystanych podczas badań

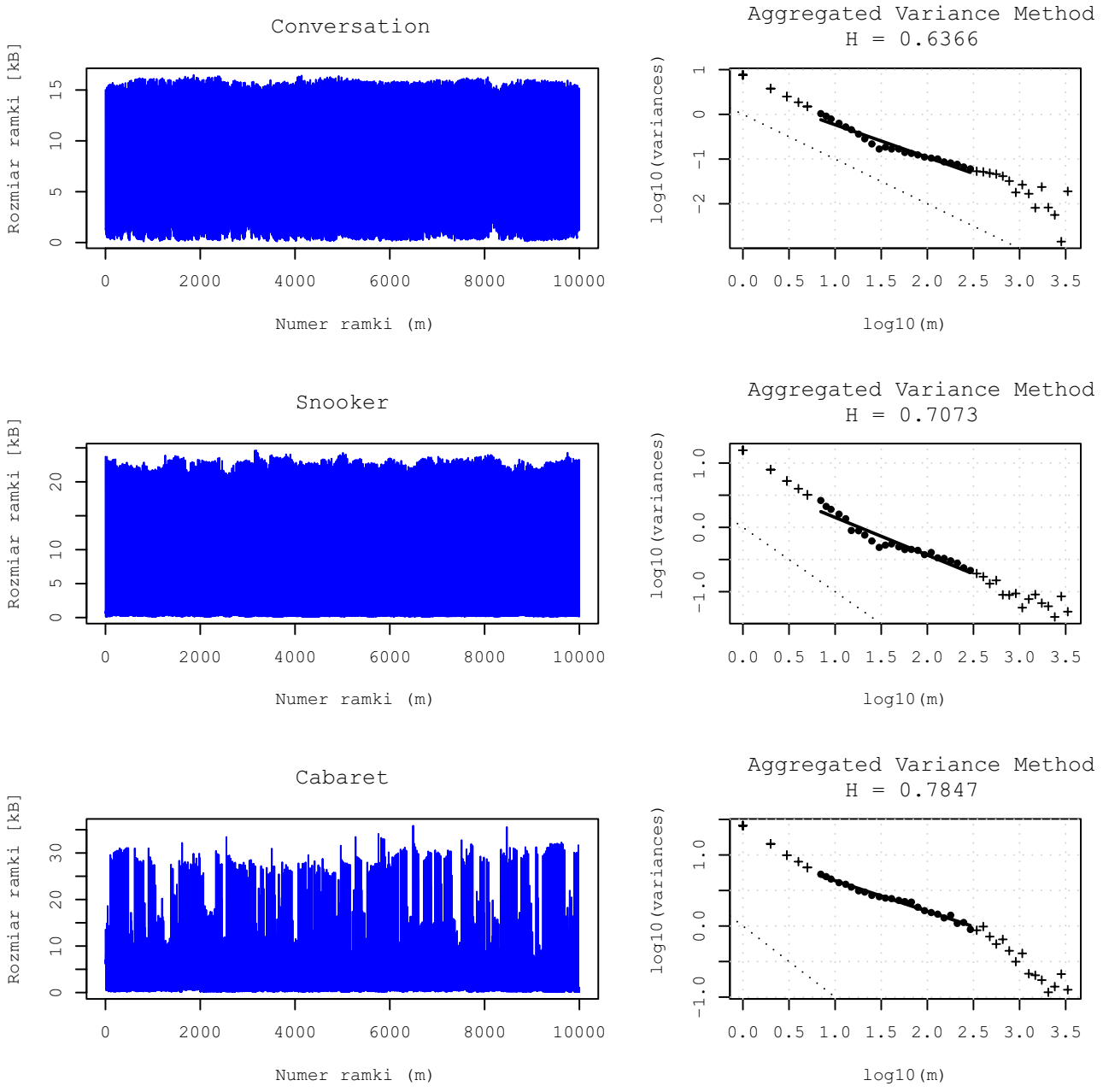
Lp.	Nazwa sekwencji	Opis	Złożoność ruchu	Parametr Hursta
1.	Reading the story	Czytanie bajki	Niska	0,50
2.	Ocean waves	Plaża nad oceanem	Niska	0,56
3.	Waterfall	Mały wodospad	Niska	0,58
4.	Conversation	Rozmowa dwóch osób	Średnia	0,64
5.	Snooker	Gra w snookera	Średnia	0,71
6.	Cabaret	Spektakl kabaretowy	Średnia	0,78
7.	Football	Mecz piłkarski	Wysoka	0,86
8.	London	Ruch uliczny w Londynie	Wysoka	0,91
9.	Underwater world	Podwodna fauna i flora	Wysoka	0,96

Do estymacji poziomu samopodobieństwa klipów wideo wymienionych w tab. 1 zastosowano pakiet R, który jest językiem programowania, a także popularnym środowiskiem do obliczeń statystycznych oraz wizualizacji wyników[3]. W procesie tym (jako szeregi czasowe) wykorzystano wielkości kolejnych ramek obrazu kompresowanego do standardu MPEG-4 za pomocą kodera *xvid_encraw*. Biblioteka *fArma* należąca do pakietu R zawiera 10 funkcji służących do estymacji parametru Hursta. Należą do nich: *aggregated variance method*, *differenced aggregated variance method*, *aggregated absolute value (moment) method*, *Higuchi's or fractal dimension method*, *Peng's or variance of residuals method*, *R/S Rescaled Range Statistic method*, *periodogram method*, *boxed (modified) periodogram method*, *Whittle estimator*, *wavelet estimator*. Podczas badań najbardziej zbliżone wyniki otrzymano jednak za pomocą trzech metod. Były to: *aggregated variance method*, *differenced aggregated variance method* oraz *wavelet estimator*. Ostatecznie do estymacji samopodobieństwa zdecydowano się zastosować *aggregated variance metod*. Wyniki otrzymane za pomocą tej metody dla poszczególnych sekwencji wideo zostały przedstawione na rys. 3, 4 i 5.

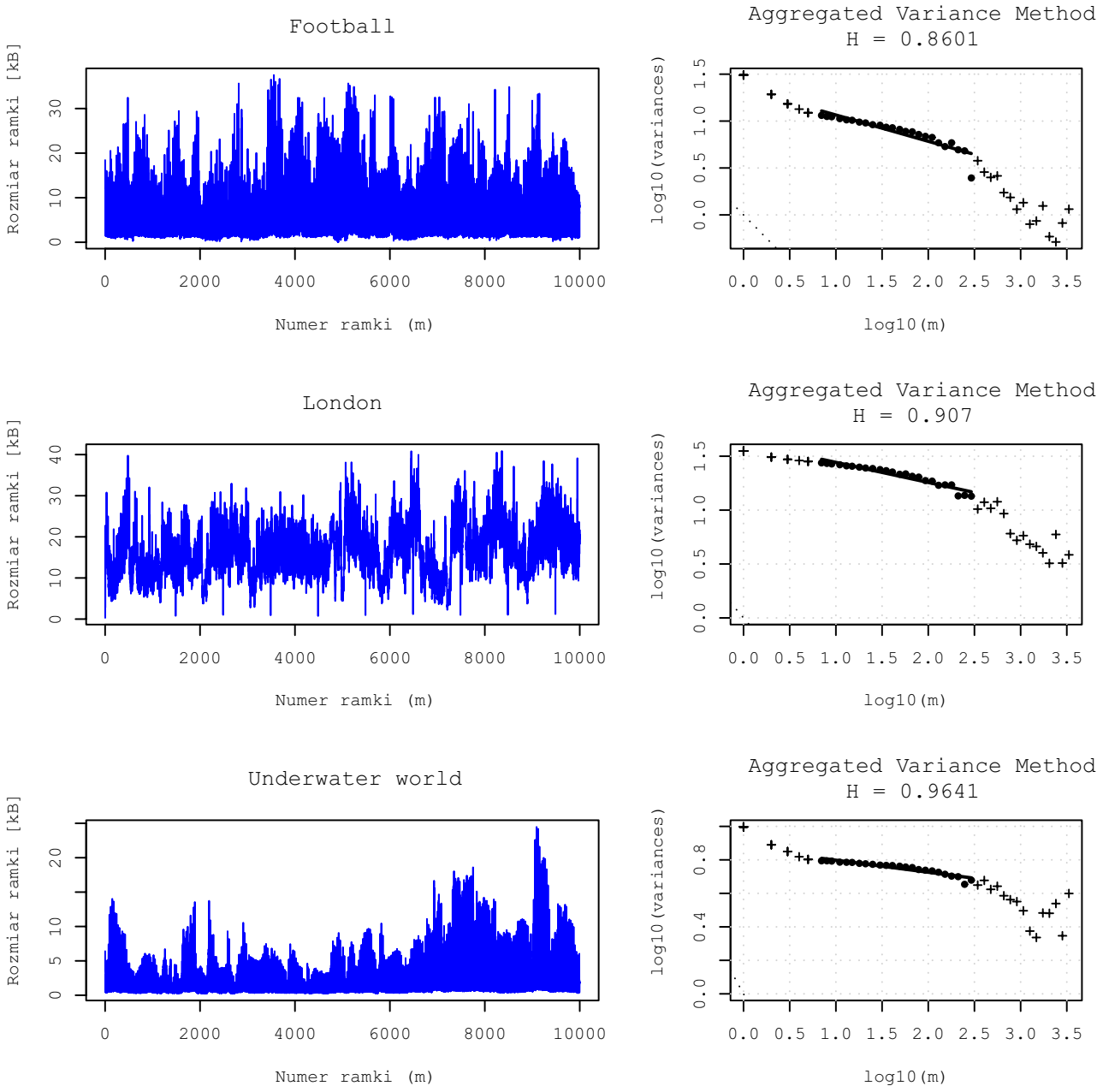
[3] Strona domowa projektu R: http://www.r-project.org/.



Rys. 3. Wielkość kolejnych ramek skompresowanego obrazu oraz parametr Hursta dla sekwencji z niską złożonością ruchu

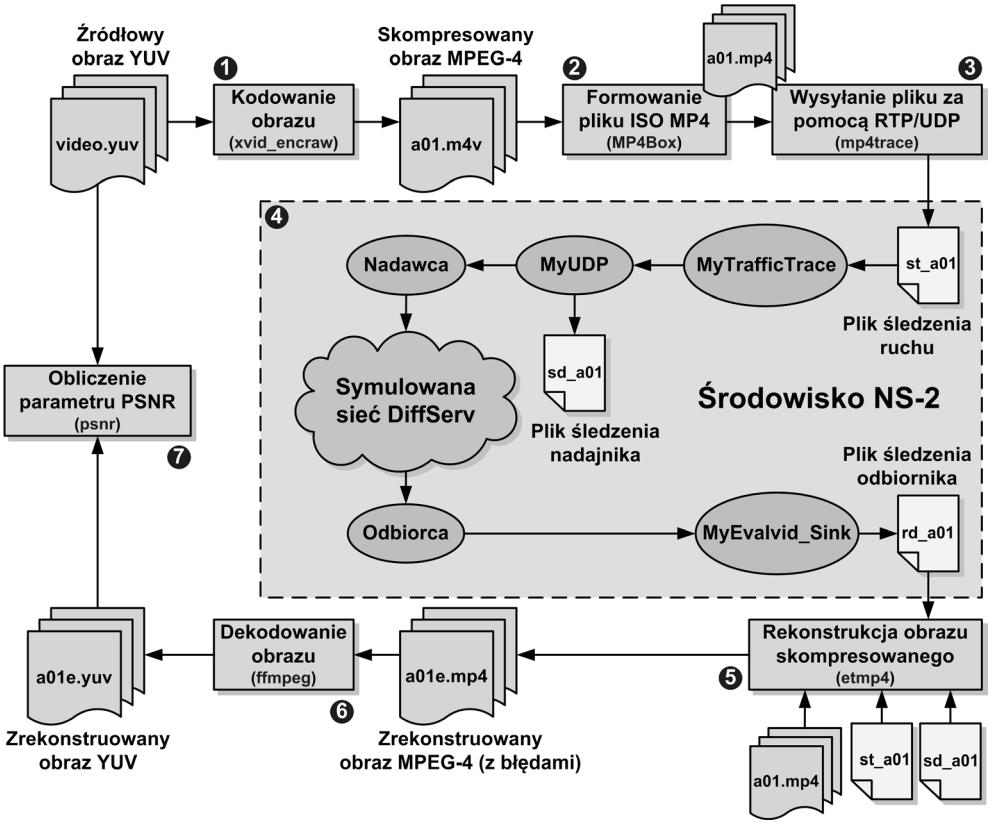


Rys. 4. Wielkość kolejnych ramek skompresowanego obrazu oraz parametr Hursta dla sekwencji ze średnią złożonością ruchu



Rys. 5. Wielkość kolejnych ramek skompresowanego obrazu oraz parametr Hursta dla sekwencji z wysoką złożonością ruchu

Każda źródłowa sekwencja YUV została następnie kolejno wykorzystana do badań. Do przeprowadzenia eksperymentu wykorzystano system oparty na popularnym pakiecie EvalVid, często stosowanym do badania ruchomych obrazów [Klaue i inni 2003].



Rys. 6. Architektura rozszerzonej wersji pakietu EvalVid i jego współpraca z symulatorem sieci NS-2

Symulację działania sieci DiffServ, przez którą przesyłano strumieniowo badane obrazy, wykonano w środowisku znanego symulatora sieciowego NS-2[4]. Standardowa dystrybucja NS-2 nie zapewnia współpracy z pakietem EvalVid, dlatego na tym etapie wystąpiła konieczność modyfikacji kilku klas symulatora sieci oraz instalacji dodatkowego modułu rozszerzającego. Wykorzystano do tego celu pakiet, który rozszerza funkcjonalność wspomnianego wcześniej EvalVid oraz pozwala na jego integrację z symulatorem NS-2 [Ke i inni 2008]. W wyniku tych działań w symulatorze pojawiły się nowe klasy (*MyTrafficTrace*, *MyUDP*, *MyEvalvid_Sink*) zapewniające pełną integrację EvalVid i NS-2 (rys. 6).

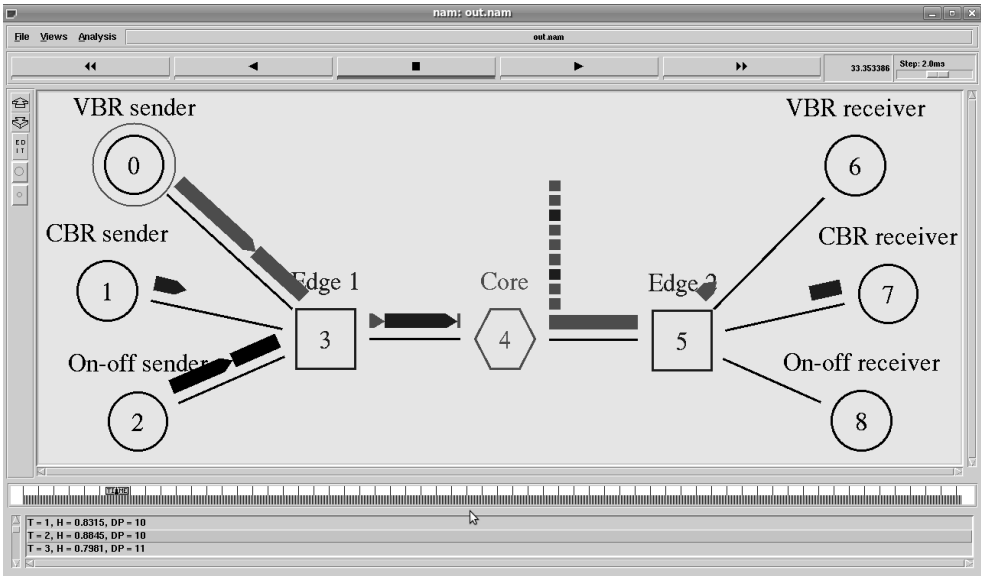
Na etapie przygotowania środowiska do badań pojawił się jeszcze jeden problem. Otóż okazało się, że NS-2 pozwalał symulować działanie sieci DiffServ, ale wyłącznie w trybie statycznym. Oznaczało to, że punkt kodowy DP przypisany danemu strumieniowi na początku symulacji nie zmieniał się w czasie jej trwania. Aby rozwiązać ten problem i dostosować symulator do potrzeb badań, dokonano modyfikacji metody *addPolicyEntry()* należącej do klasy *PolicyClassifier*. W tym celu do pliku *dsPolicy.cc* znajdującego się w podkatalogu DiffServ dodano kod w języku C++, który na poniższym listingu został zaznaczony pogrubioną czcionką.

```
void PolicyClassifier::addPolicyEntry(int argc, const char*const* argv) {
  if (policyTableSize <= 2) {
  // Oryginalny kod
  } else {
  policyTableEntry *policy;
  policy = getPolicyTableEntry(atoi(argv[2]), atoi(argv[3]));
  if (policy == NULL)
  printf("ERROR: cannot update Entry; no existing policy found for Source %d-Desination
      %d.\n", atoi(argv[2]), atoi(argv[3]));
  else {
  policy->codePt = (int) atoi(argv[5]);
  }
  }
}
```

[4] The Network Simulator – NS-2. Strona domowa projektu: http://www.isi.edu/nsnam/ns/index.html.

Po kompilacji NS-2 otrzymano możliwość dynamicznej aktualizacji wartości punktu kodowego DP w dowolnej chwili czasu. Fakt pojawienia się powyższego problemu świadczy o tym, iż podejście wykorzystujące dynamiczną zmianę priorytetu strumieni DiffServ nie jest jeszcze obecnie zbyt popularne. W przeciwnym wypadku w symulatorze NS-2 bardzo popularnym na całym świecie w środowiskach naukowych występowałaby już opisana wyżej funkcjonalność.

W skład symulatora NS-2 wchodzi m.in. program NAM (*Network Animator*) służący do wizualizacji procesu symulacji. Przykładowe okno programu NAM podczas symulacji działania sieci DiffServ przedstawiono na rys. 7.



Rys. 7. Przykładowe okno programu NAM podczas symulacji działania sieci DiffServ

Poniżej scharakteryzowano czynności wykonywane w kolejnych etapach wykorzystania pakietu EvalVid, a także podano przykładowe polecenia. Czynności te zostały również zaznaczone na rys. 6 w postaci kółek z cyframi w środku.

1. Kodowanie sekwencji źródłowej YUV do standardu MPEG-4 za pomocą kodera *xvid_encraw*. Wynikiem kodowania był obraz skompresowany o szybkości zmiany ramek równej 30 ramek na sekundę oraz długości GOP równej 30 ramek (bez ramek typu B).

   ```
   xvid_encraw -i video.yuv -w 352 -h 288 -framerate 30 -max_key_interval 30
   -o a01.m4v
   ```

2. Tworzenie za pomocą programu *MP4Box* pliku ISO MP4. Zawierał on skompresowane ramki obrazu oraz ścieżkę opisującą, jak dzielić ramki na pakiety w celu ich transportu za pomocą protokołu RTP.

   ```
   MP4Box -hint -mtu 1024 -fps 30 -add a01.m4v a01.mp4
   ```

3. Program *mp4trace* z pakietu EvalVid jest odpowiedzialny za wysyłanie pliku MP4 za pomocą protokołu RTP/UDP do określonego hosta przeznaczenia. W poniższym poleceniu wykorzystany został host o adresie IP 192.168.1.1 i porcie UDP 12346. Wynik działania programu *mp4trace* był wykorzystywany później, dlatego na tym etapie został on przekierowany do pliku tekstowego `st_a01`. Plik ten zawierał informację o typach ramek, segmentacji pakietów itp.

   ```
   mp4trace -f -s 192.168.1.1 12346 a01.mp4 > st_a01
   ```

4. Symulacja w środowisku NS-2 polegająca na przesłaniu obrazu MPEG-4 przez sieć. Host źródłowy wykorzystywał plik `st_a01` do transmisji pakietów do hosta przeznaczenia. Plik `be_a01.tcl` zawierał skrypt symulacji.

   ```
   $ns2 be_a01.tcl
   ```

   W wyniku symulacji zostały utworzone pliki `sd_a01` i `rd_a01`. W pliku `sd_a01` był zapisany czas wysłania każdego pakietu IP, natomiast w pliku `rd_a01` – czas odebrania pakietu.

5. Rekonstrukcja obrazu przesłanego przez sieć w taki sposób, w jaki jest to dokonywane przez odbiornik. W tym celu źródłowy plik wideo oraz pliki śledzenia były przetwarzane przez program *etmp4* (*Evaluate Traces of MP4-file transmission*). W efekcie został wygenerowany zrekonstruowany (zawierający uszkodzenia) plik wideo. Podczas tego procesu z oryginalnej ścieżki obrazu zostały usunięte wszystkie ramki, które zostały utracone lub uszkodzone.

   ```
   etmp4 sd_a01 rd_a01 st_a01 a01.mp4 a01e
   ```

   Program *etmp4* utworzył również następujące pliki:
   `loss_a01e.txt` – zawierał straty (wyrażone w %) dotyczące ramek typu I, P, B oraz straty ogólne,
   `delay_a01e.txt` – zawierał numer ramki, flagę strat, opóźnienie od końca do końca, odstęp między ramkami nadajnika, odstęp między ramkami odbiornika oraz łączną fluktuację opóźnienia w sekundach,
   `rate_s_a01e.txt` – zawierał czas, liczbę bajtów na sekundę (w bieżącym przedziale czasu) oraz liczbę bajtów na sekundę (łącznie) zmierzoną w nadajniku,
   `rate_r_a01e.txt` – zawierał czas, liczbę bajtów na sekundę (w bieżącym przedziale czasu) oraz liczbę bajtów na sekundę (łącznie) zmierzoną w odbiorniku.
6. Dekodowanie odebranego, skompresowanego obrazu do formatu YUV.

   ```
   ffmpeg -i a01e.mp4 a01e.yuv
   ```

7. Obliczenie parametru PSNR.

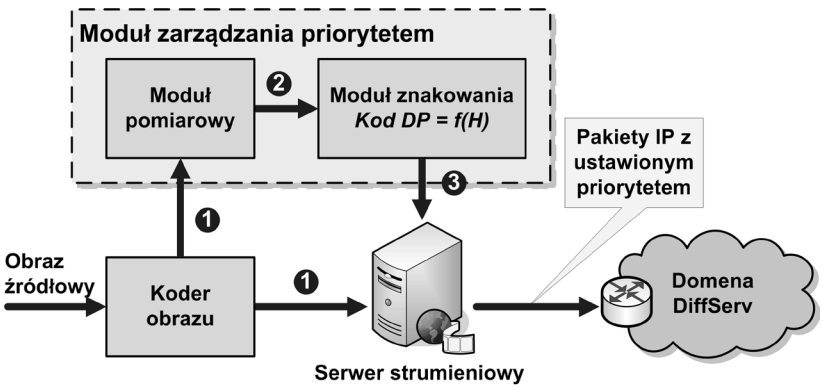
   ```
   psnr.exe 352 288 420 video.yuv a01e.yuv > work\psnr_a01e.txt
   ```

8. Obliczenie parametru MOS. Często oczekujemy większej ilości informacji o różnicy pomiędzy obrazem zakodowanym a odebranym, niż ta dostarczana przez sam parametr PSNR. W takiej sytuacji pakiet EvalVid dostarcza narzędzia MOS. Program `mos.exe` oblicza parametr MOS (*Mean Opinion Score*) każdej pojedynczej ramki odebranego obrazu i porównuje go z parametrem MOS każdej pojedynczej ramki obrazu oryginalnego. Po zapisaniu wyników pomiarów PSNR do plików (np. `ref_psnr.txt, psnr_a01e.txt, psnr_a02e.txt`, itd.) i umieszczeniu plików odpowiadających strumieniom odebranym w katalogu, np. `work`, poniższe polecenie pozwala obliczyć średnią wartość MOS dla każdego pliku z PSNR (ostatnia kolumna pliku `mos.txt`).

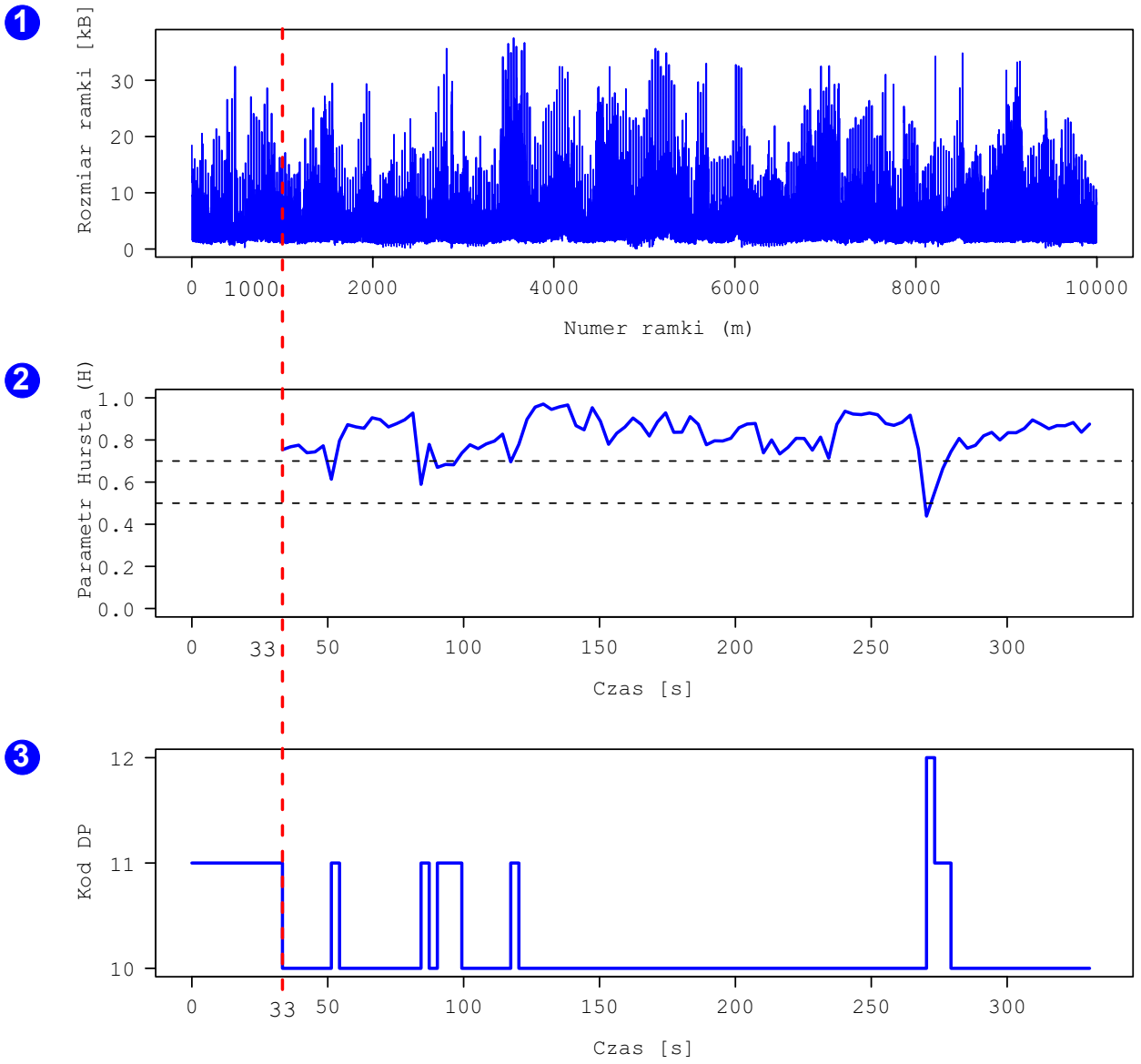
   ```
   mos work ref_psnr.txt 25 > mos.txt
   ```

2. Symulacja działania modułu zarządzania priorytetem

Implementacja modułu zarządzania priorytetem (rys. 8) jest zaplanowana do realizacji w toku przyszłych badań, dlatego na etapie symulacji prezentowanej w niniejszym artykule wystąpiła konieczność wygenerowania sygnałów, które oddawałyby istotę działania wspomnianego wcześniej modułu (rys. 9). Symulując działanie modułu pomiarowego, dla każdego badanego obrazu dokonano pomiaru samopodobieństwa sygnału wyjściowego kodera obrazu (oznaczonego cyfrą 1 na rys. 8 i 9). Wykorzystano w tym celu metodę zagregowanej wariancji (*aggregated variance metod*) z pakietu R.



Rys. 8. Moduł zarządzania priorytetem



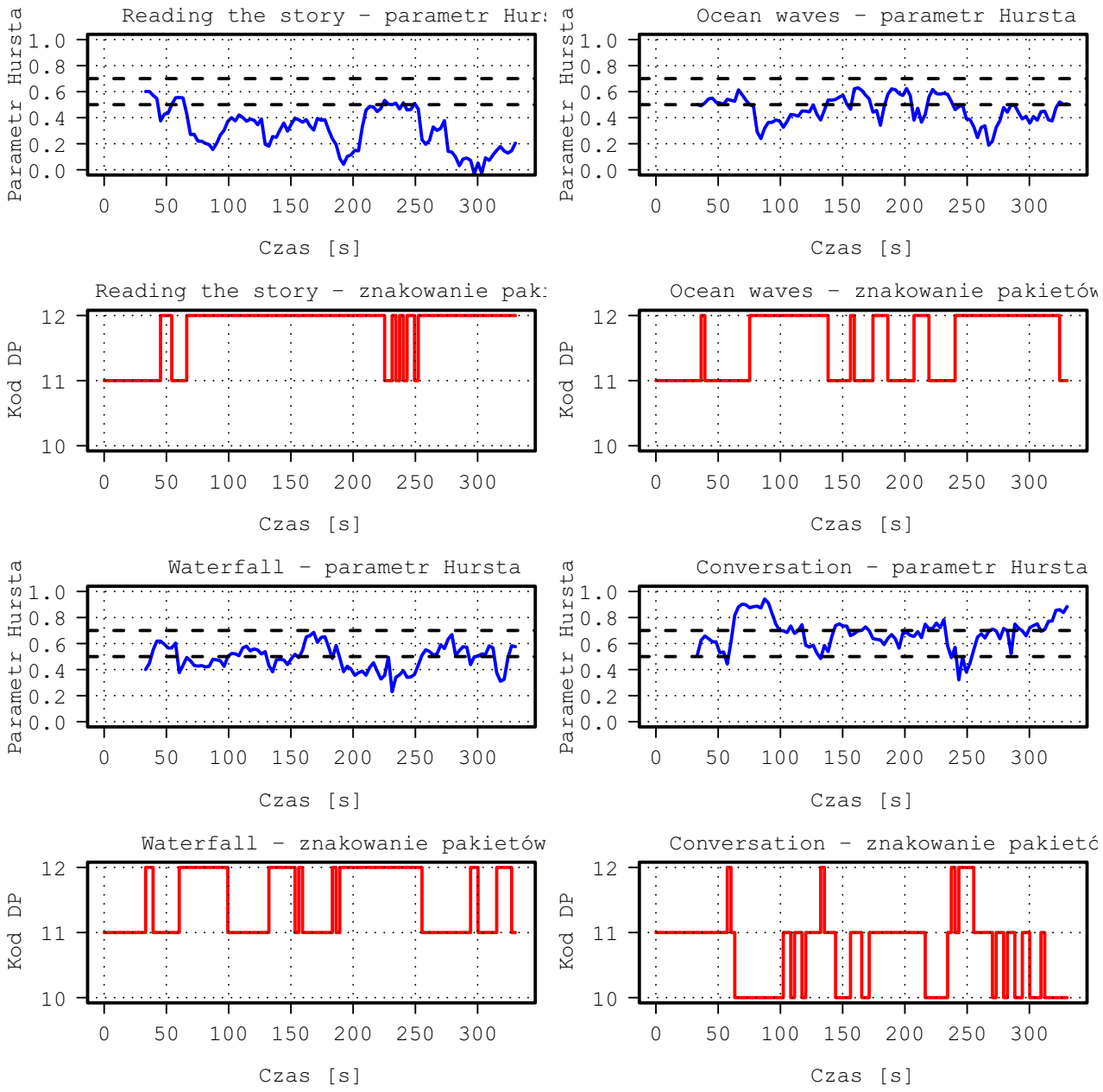
Rys. 9. Przebiegi sygnałów modułu zarządzania priorytetem dla sekwencji *Football*

Każda sekwencja składała się w tym przypadku z 10 000 skompresowanych ramek. Do pomiaru parametru Hursta zastosowano metodę wykorzystującą okno przesuwne. Przyjęto szerokość okna czasowego n = 1000 ramek (33,333 sekundy) oraz przesunięcie okna Δn = 90 ramek (3 sekundy). W rezultacie otrzymano plik tekstowy zawierający szereg czasowy reprezentujący zmiany poziomu samopodobieństwa danego strumienia w czasie (oznaczony cyfrą 2 na rys. 8 i 9). Szereg czasowy został następnie wykorzystany jako dane wejściowe dla modułu znakowania realizującego mapowanie wartości parametru Hursta na 3 poziomy priorytetu pakietów wysyłanych do sieci DiffServ. Sygnał wyjściowy modułu znakowania został na rys. 8 i 9 oznaczony cyfrą 3. Funkcjonalność modułu znakowania została zasymulowana przez wykonanie skryptu w języku Tcl przedstawionego na poniższym listingu.

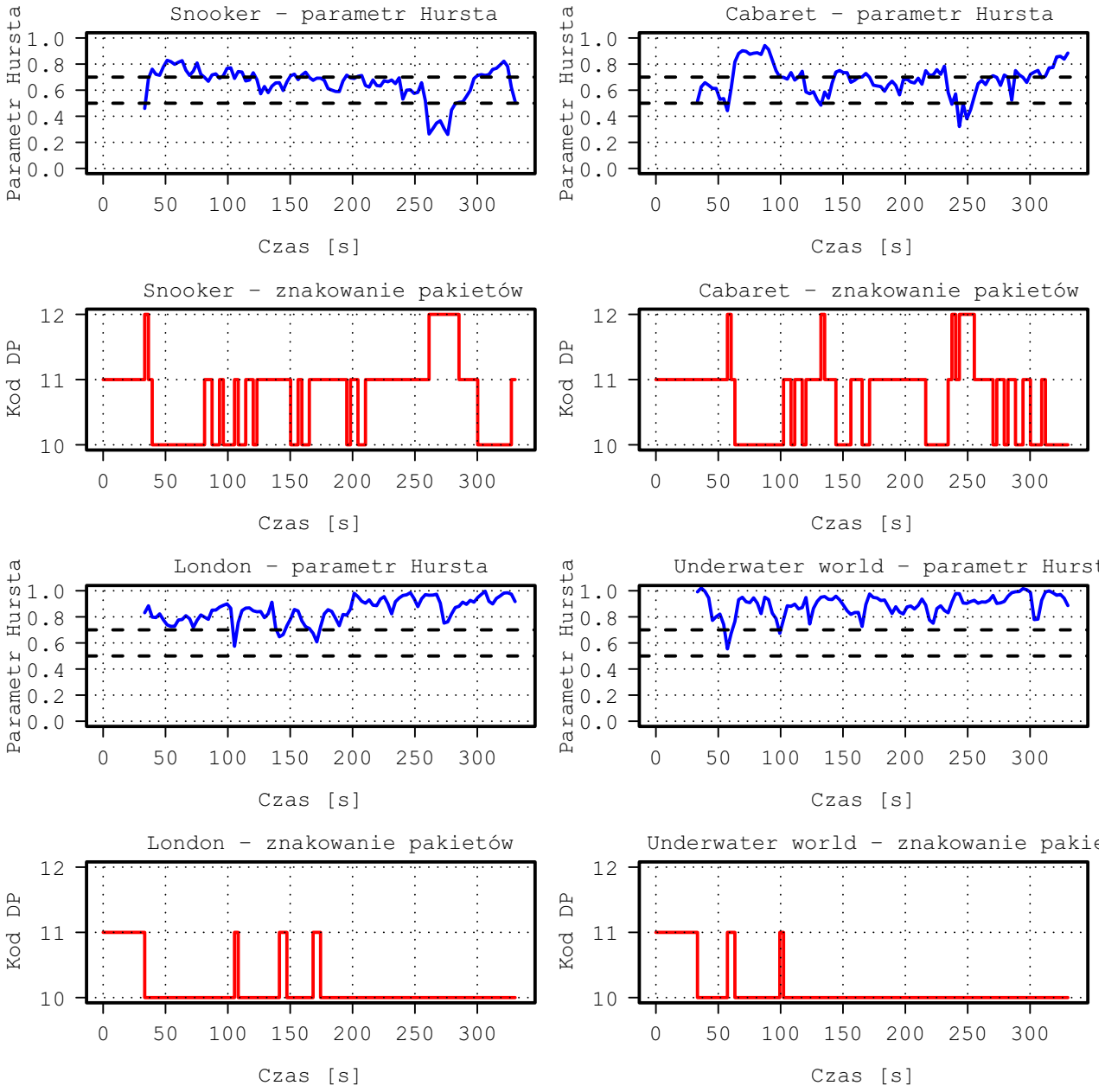
```
…
# Loading the Hurst parameter for the successive measurement periods.
set measurementPeriod 0
set i 0
set textFile [open hurstTimeSeries.txt r]
while {![eof $textFile]} {
 incr i
 set hurstTable($i) [gets $textFile]
}
# Marking module simulation.
# Update the policy entry table every 3 sec.
proc packetMarking {} {
 global ns s1 dest1 qE1C qE2C cir1 pir1 hurstTable measurementPeriod
 set checkTime 3
 set timeNow [$ns now]
 incr measurementPeriod
 set hurst $hurstTable($measurementPeriod)
 if {$hurst >= 0.7} {
set DP 10
 } elseif {$hurst >= 0.5 && $hurst < 0.7} {
 set DP 11
 } else {
 set DP 12
 }
 $qE1C addPolicyEntry [$s1 id] [$dest1 id] TSW3CM $DP $cir1 $pir1
 $qE2C addPolicyEntry [$dest1 id] [$s1 id] TSW3CM $DP $cir1 $pir1
 if {$measurementPeriod < 99} {
 $ns at [expr $timeNow + $checkTime ] "packetMarking"
 }
}
$ns at 33.333 "packetMarking"
…
```

Należy w tym miejscu wyjaśnić, iż symulator NS-2 stosuje swój własny system kodowania priorytetu pakietów. I tak, dla pakietów, które mają być oznaczone najwyższym priorytetem (kolor zielony), stosowany jest w NS-2 kod 10, dla pakietów ze średnim priorytetem (kolor żółty) – kod 11, a dla pakietów z najniższym priorytetem (kolor czerwony) – kod 12. Przypomnijmy również, że im wyższy jest priorytet pakietów, tym mniejsze prawdopodobieństwo ich odrzucenia (*Drop Precedence*) w przypadku przepełnienia kolejki.

Sygnały modułu pomiarowego i modułu znakowania wygenerowane dla sekwencji wideo wykorzystanych podczas eksperymentu zostały przedstawione na rys. 10 i 11.



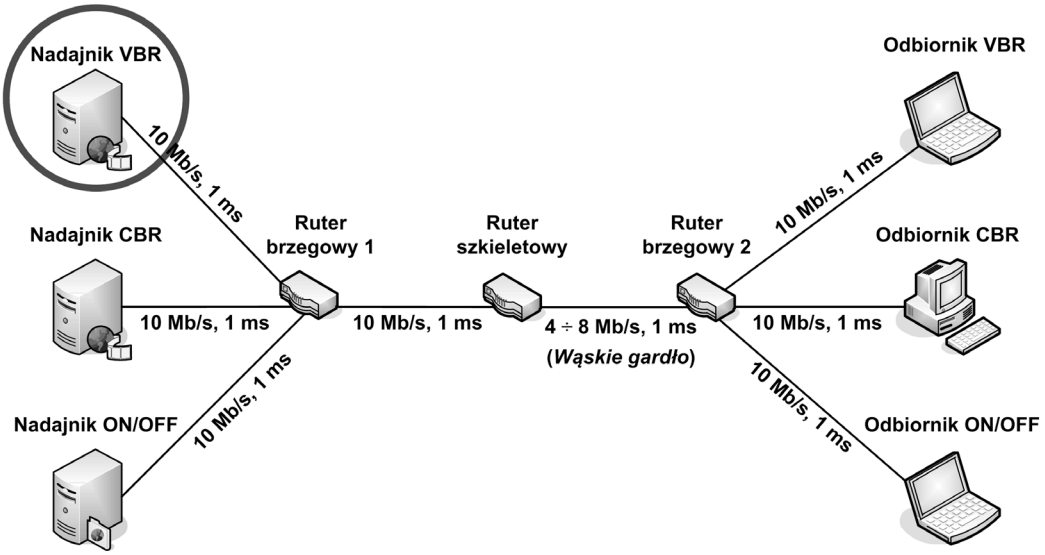
Rys. 10. Przebiegi sygnałów modułu zarządzania priorytetem dla sekwencji *Reading the story*, *Ocean waves*, *Waterfall*, *Conversation*



Rys. 11. Przebiegi sygnałów modułu zarządzania priorytetem dla sekwencji *Snooker*, *Cabaret*, *London*, *Underwater world*

3. Konfiguracja sieci wykorzystanej podczas symulacji

Badania sieci DiffServ zrealizowano w środowisku symulatora NS-2. Na rys. 12 przedstawiono topologię zastosowanej sieci. W jej skład wchodziły 3 rutery należące do domeny DiffServ (2 brzegowe i 1 szkieletowy), 3 nadajniki emitujące strumienie o różnych charakterystykach ruchu oraz 3 odbiorniki. Wszystkie łącza charakteryzowały się szybkością 10 Mb/s oraz opóźnieniem transmisji 1 ms. Wyjątkiem było łącze pomiędzy ruterem szkieletowym a ruterem brzegowym nr 2 (wąskie gardło), którego przepustowość zmieniała się zależnie od średniej szybkości bitowej przesyłanego obrazu.



Rys. 12. Topologia sieci zastosowanej podczas symulacji

Podczas symulacji wykorzystano klipy wideo, które zostały dokładnie scharakteryzowane w rozdziale 1. Każdy z tych klipów był w sposób strumieniowy wysyłany przez nadajnik VBR (*Variable Bit Rate*), przechodził przez rutery domeny DiffServ i docierał do odbiornika VBR. Przed transmisją każda ramka obrazu była dzielona na pakiety UDP o wielkości 1052 bajty. W tab. 2 zestawiono charakterystyki ruchu poszczególnych strumieni pakietów przesyłanego obrazu.

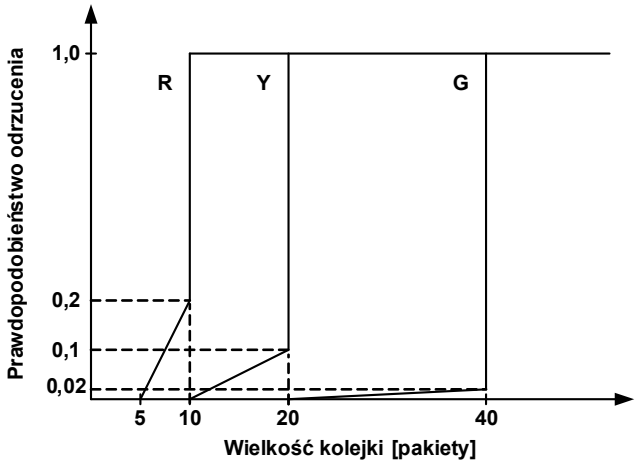
Tab. 2. Charakterystyki ruchu strumieni pakietów przesyłanego obrazu

Lp.	Nazwa sekwencji	Średnia szybkość bitowa [kb/s]	Maksymalna szybkość bitowa [kb/s]
1.	Reading the story	281	670
2.	Ocean waves	595	769
3.	Waterfall	1 034	1 839
4.	Conversation	377	827
5.	Snooker	567	1 214
6.	Cabaret	1 081	2 791
7.	Football	2 317	4 620
8.	London	3 916	7 292
9.	Underwater world	911	3 194

Pakiety przesyłanego obrazu rywalizowały o dostęp do zasobów węzłów domeny DiffServ z pakietami należącymi do dwóch innych strumieni. Pierwszym z nich był strumień CBR (*Constant Bit Rate*), który tworzyły pakiety UDP o wielkości 1052 bajty i szybkości 3,5 Mb/s. Drugi strumień tworzyły pakiety TCP o średniej wielkości 1052 bajty generowane przez źródło ruchu ON/OFF (Pareto). Strumień ruchu ON/OFF posiadał rozkład eksponencjalny i charakteryzował się następującymi parametrami: szybkość – 0,5 Mb/s, czas emisji pakietów – 500 ms, czas przerwy – 0,1 ms. Współczynnik Pareto generowanego rozkładu wynosił 1,5, czemu odpowiadał parametr Hursta równy 0,75.

Jak zaznaczono wcześniej, celem eksperymentu było zbadanie możliwości poprawy jakości usług oferowanych przez sieć DiffServ strumieniom obrazu rywalizującym o dostęp do zasobów sieci z innymi strumieniami, ale w ramach tej samej klasy usług. Preferencyjne traktowanie pakietów obrazu ruchomego ma polegać na odpowiednim ustawieniu priorytetu pakietów, czyli pierwszeństwa odrzucania w warunkach przepełnienia kolejek, stosownie do wielkości zaburzeń ruchu obecnych aktualnie w strumieniu. Dlatego też w ramach

konfiguracji ruterów eksperymentalnej sieci DiffServ przyjęto jedną kolejkę fizyczną odpowiadającą jednej klasie ruchu oraz 3 kolejki wirtualne – przypisane trzem priorytetom obsługi pakietów w ramach danej klasy. Dla ruterów brzegowych zaimplementowano politykę TSW3CM (*Time Sliding Window Three Color Marker*) wykorzystującą parametry CIR (*Committed Information Rate*), PIR (*Peak Information Rate*) oraz 3 priorytety pakietów. Dla badanych przepływów CIR i PIR były równe odpowiednio średniej i maksymalnej szybkości bitowej strumieni pakietów obrazu. Z kolei dla strumienia CBR oba parametry wynosiły 3,5 Mb/s, natomiast dla strumienia ruchu ON/OFF – 0,5 Mb/s. Parametry konfiguracyjne wirtualnych kolejek RED (*Random Early Detection*) dla pakietów zielonych (najwyższy priorytet), żółtych (średni priorytet) i czerwonych (najniższy priorytet) ustawione były zgodnie z danymi przedstawionymi na rys. 13.



Rys. 13. Parametry konfiguracyjne wirtualnych kolejek RED

4. Wyniki badań

W ramach przeprowadzonych badań porównano podstawowe parametry określające jakość obrazu ruchomego po stronie odbiorczej przy statycznym i dynamicznym ustawianiu priorytetu pakietów. W pierwszym przypadku wszystkie pakiety miały stały priorytet, który nie zmieniał się podczas emisji. Pakiety źródeł VBR i CBR oznaczono kolorem czerwonym, a źródła ruchu ON/OFF – żółtym. W przypadku drugim priorytet pakietów źródła VBR zmieniał się zależnie od wielkości parametru Hursta zmierzonego w bieżącym okresie pomiarowym. Ustawianie priorytetu dokonywane było zgodnie z przyjętym algorytmem mapowania modułu znakowania.

Otrzymane wyniki pozwoliły na obliczenie parametrów PSNR, MOS, strat, opóźnienia oraz fluktuacji opóźnienia pakietów. Parametr PSNR wyrażający stosunek wartości szczytowej sygnału do szumu jest jedną z najpopularniejszych metod oceny jakości obrazu [Huynh-Thu i Ghanbari 2008]. Wyraża on poziom podobieństwa między dwoma obrazami (np. pierwotnym i skompresowanym lub emitowanym i odebranym poprzez sieć komputerową). Współczynnik PSNR przyjmuje wartości w skali logarytmicznej (jednostką są decybele). Im jest on większy, tym większe podobieństwo występuje pomiędzy porównywanymi obrazami.

W celu obliczenia współczynnika PSNR należy najpierw obliczyć współczynnik *MSE* (*Mean Square Error*) porównywanych obrazów ze wzoru:

[1] $$\mathbf{MSE} = \frac{1}{N \cdot M} \sum_{i=1}^{N} \sum_{j=1}^{M} \Big(\big[f(i,j) - f'(i,j) \big]^2 \Big),$$

gdzie: N, M – wymiary obrazu,
$f(i,j)$ – obraz oryginalny,
$f'(i,j)$ – obraz skompresowany.

Następnie wartość *MSE* należy podstawić do wzoru:

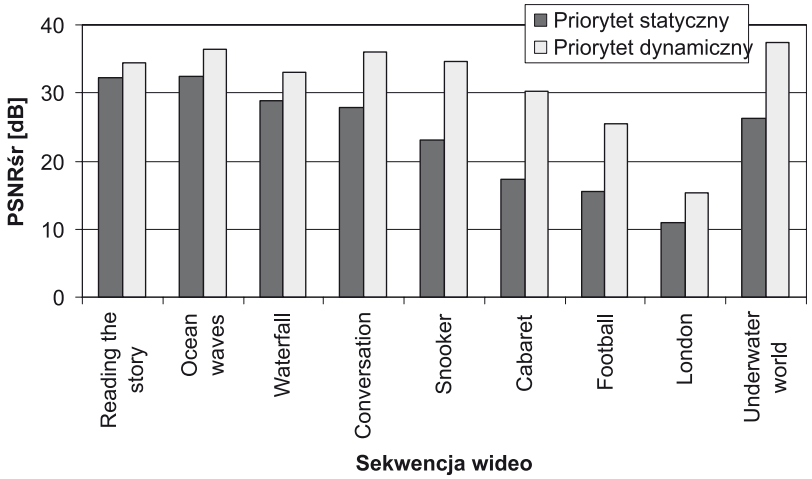
[2] $$\mathbf{PSNR} = 10 \cdot \log_{10} \frac{k^2}{\mathbf{MSE}},$$

gdzie: k – liczba kolorów obrazu minus 1 (dla 8-bitowej głębi kolorów jest to 255, dla 16-bitowej – 65535 itd.).

W tab. 3 oraz na rys. 14 zamieszczono zestawienie wartości średniej parametru PSNR zrekonstruowanych sekwencji obrazu przy ustawianiu priorytetu pakietów w sposób statyczny i dynamiczny.

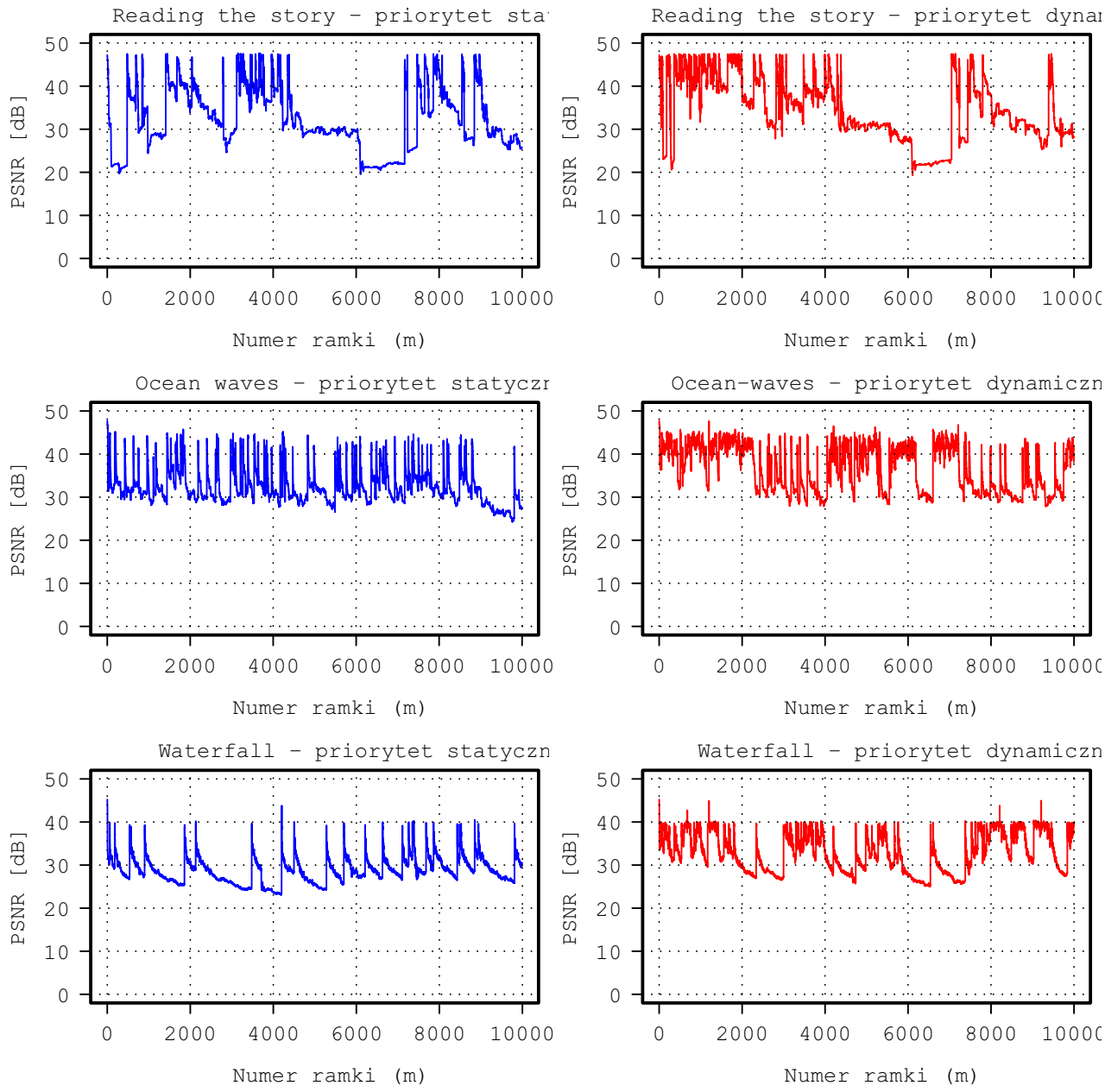
Tab. 3. Zestawienie wartości średniej parametru PSNR zrekonstruowanych sekwencji obrazu

Nazwa sekwencji	PSNRśr [dB]	
	Priorytet statyczny	Priorytet dynamiczny
Reading the story	32,27	34,49
Ocean waves	32,40	36,41
Waterfall	28,88	32,94
Conversation	27,92	35,98
Snooker	23,08	34,67
Cabaret	17,38	30,15
Football	15,57	25,40
London	11,03	15,28
Underwater world	26,30	37,37

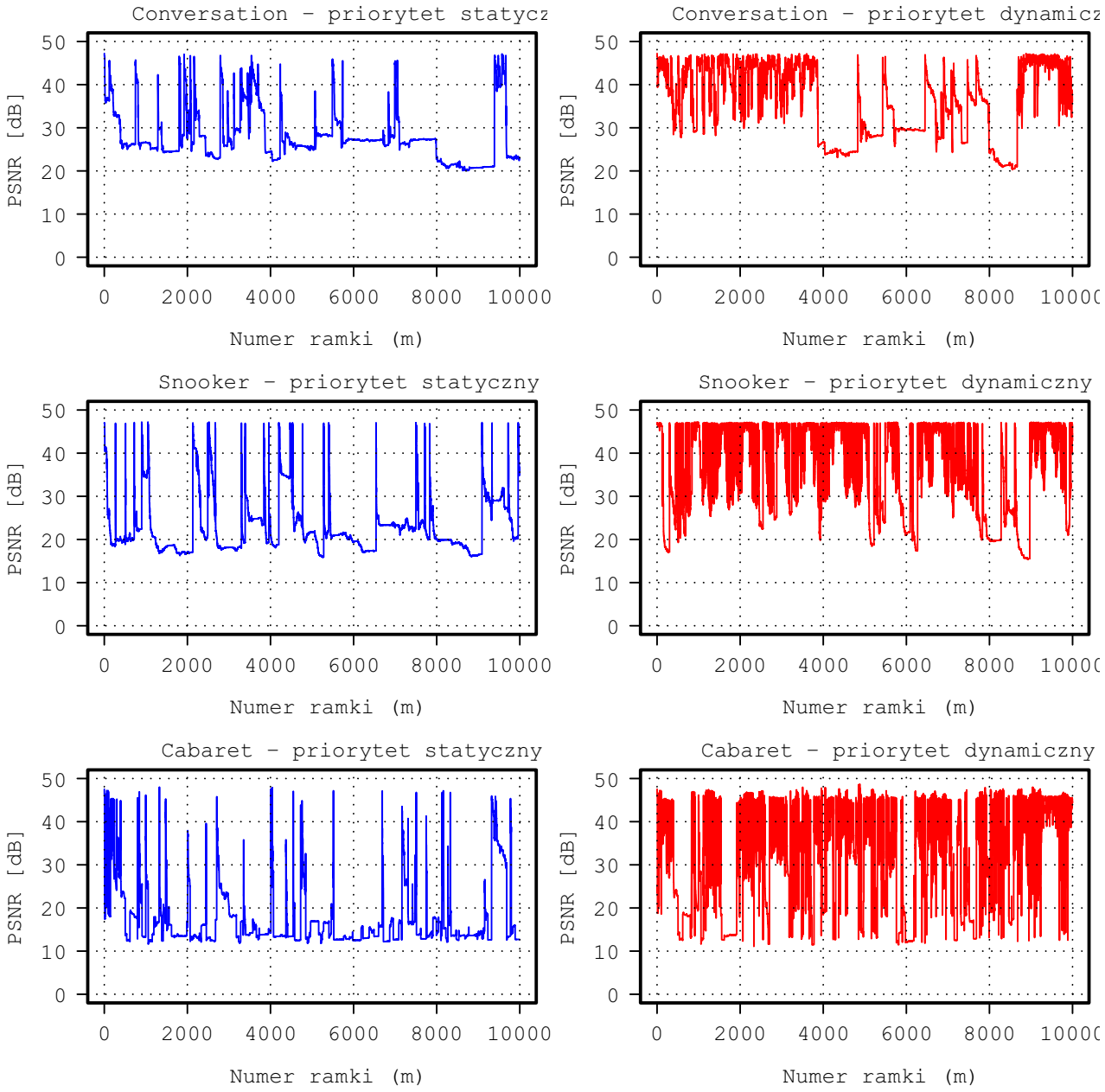


Rys. 14. Porównanie wartości średniej parametru PSNR zrekonstruowanych sekwencji obrazu

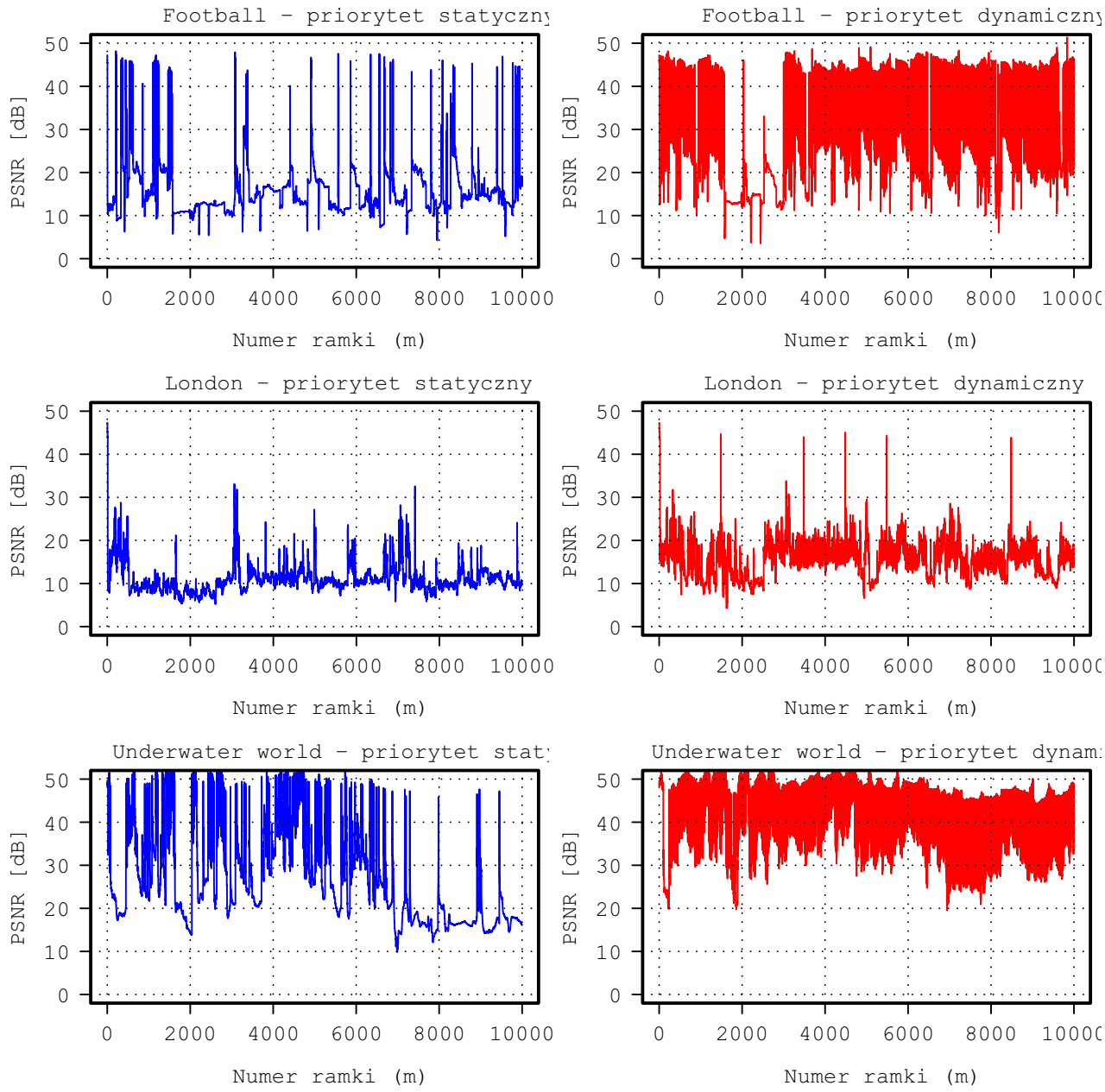
Zastosowanie priorytetu dynamicznego spowodowało zwiększenie średniej wartości PSNR. Miało to miejsce dla wszystkich badanych sekwencji, przy czym w przypadku klipów charakteryzujących się średnią i wysoką złożonością ruchu poprawa jakości była stosunkowo największa. Wynika to z faktu, iż przy większej złożoności ruchu w przesyłanym strumieniu częściej występują okresy charakteryzujące się wyższymi zaburzeniami (większy parametr Hursta). Stawiają one sieci wyższe wymagania odnośnie do jakości obsługi (przepustowość, straty, opóźnienie, fluktuacja opóźnienia). Zwiększenie priorytetu pakietów prowadzi w takich przypadkach do istotnej poprawy jakości odbieranego obrazu. Na rys. 15, 16 i 17 przedstawiono porównanie zmian parametru PSNR przy statycznym i dynamicznym ustawianiu priorytetu pakietów dla każdej sekwencji wideo.



Rys. 15. Parametr PSNR przy statycznym i dynamicznym ustawianiu priorytetu pakietów dla sekwencji z niską złożonością ruchu



Rys. 16. Parametr PSNR przy statycznym i dynamicznym ustawianiu priorytetu pakietów dla sekwencji ze średnią złożonością ruchu



Rys. 17. Parametr PSNR przy statycznym i dynamicznym ustawianiu priorytetu pakietów dla sekwencji z wysoką złożonością ruchu

Sam parametr PSNR dostarcza często zbyt małej ilości informacji. Warto wówczas skorzystać z subiektywnej miary oceny jakości obrazu, jaką jest MOS. Podaje ona w skali od 1 do 5 różnicę między jakością obrazu pierwotnego a skompresowanego lub przesłanego przez sieć (tab. 4). Ogólne metody określania parametru MOS zostały unormowane przez Międzynarodowy Związek Telekomunikacyjny (ITU) w zaleceniu ITU-T P.800 [ITU-T 1996].

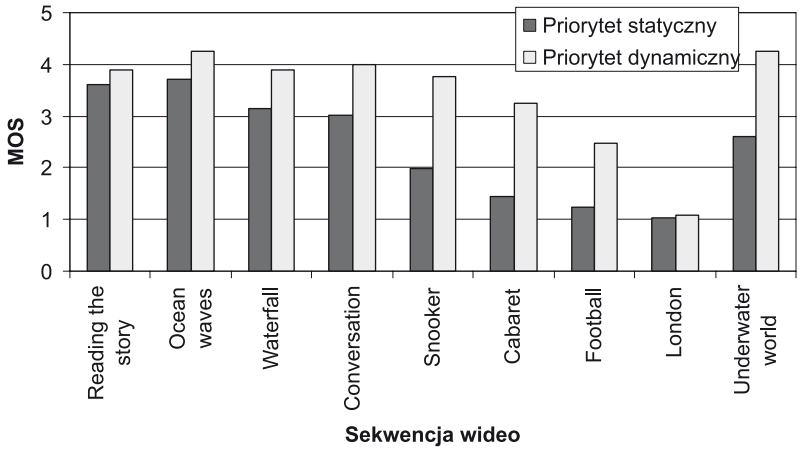
Tab. 4. Skala jakości obrazu, skala pogorszenia obrazu oraz mapowanie parametru PSNR na MOS według zalecenia ITU-T P.800 [ITU-T 1996]

PSNRśr [dB]	**MOS**	**Jakość postrzegana**	**Pogorszenie jakości**
> 37	5	Znakomita	Niedostrzegalne
31 – 37	4	Dobra	Dostrzegalne, lecz nieirytujące
25 – 31	3	Średnia	Nieznacznie irytujące
20 – 25	2	Słaba	Irytujące
< 20	1	Zła	Bardzo irytujące

Zestawienie wartości parametru MOS badanych sekwencji obrazu zamieszczono w tab. 5 oraz na rys. 18.

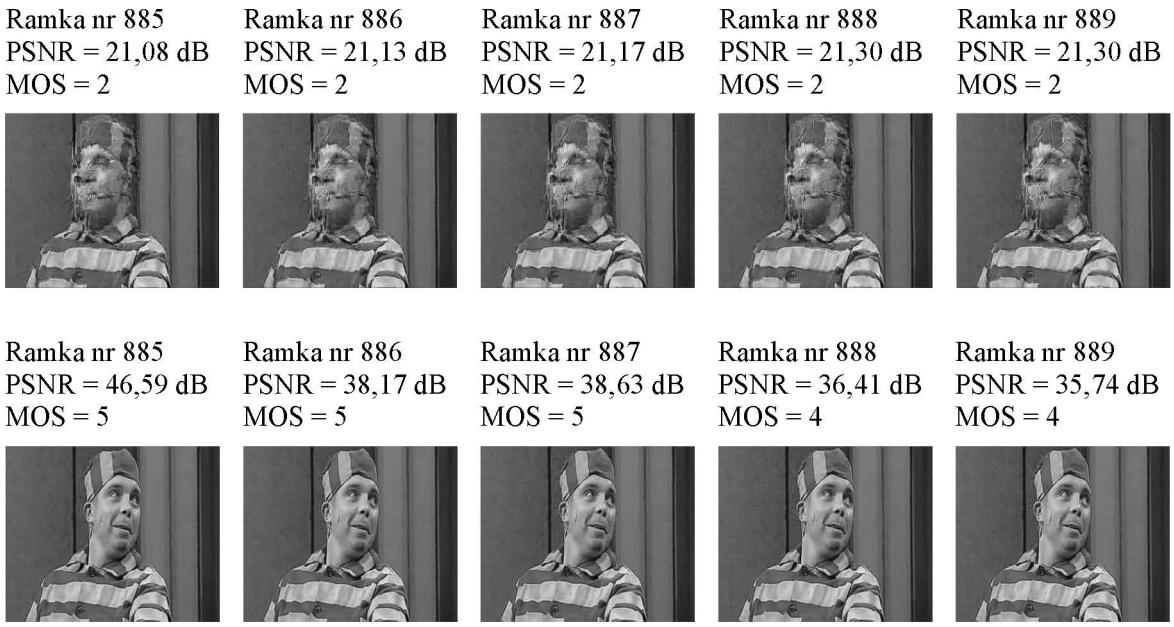
Tab. 5. Zestawienie wartości parametru MOS zrekonstruowanych sekwencji obrazu

Nazwa sekwencji	MOS	
	Priorytet statyczny	Priorytet dynamiczny
Reading the story	4	4
Ocean waves	4	4
Waterfall	3	4
Conversation	3	4
Snooker	2	4
Cabaret	1	3
Football	1	3
London	1	1
Underwater world	3	5



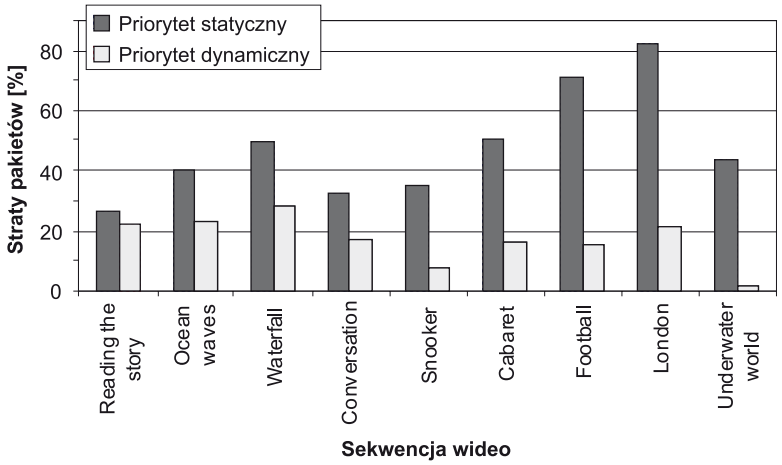
Rys. 18. Porównanie wartości parametru MOS zrekonstruowanych sekwencji obrazu

W przypadku sekwencji charakteryzujących się małą złożonością ruchu nie wystąpiła istotna różnica jakości obrazu wyrażanej parametrem MOS przy ustawianiu priorytetu pakietów w sposób statyczny i dynamiczny. Natomiast znaczna poprawa jakości miała miejsce dla sekwencji ze średnią i wysoką złożonością ruchu. Niemalże dla wszystkich klipów należących do tej grupy zastosowanie priorytetu dynamicznego spowodowało zwiększenie parametru MOS o 2 poziomy jakości w porównaniu z priorytetem statycznym. Wyjątkiem okazała się tutaj sekwencja *London*, dla której zarówno w przypadku priorytetu statycznego, jak i dynamicznego parametr MOS miał wartość równą 1. W porównaniu z innymi klipami (tab. 2) charakteryzowała się ona bardzo dużą maksymalną szybkością bitową, równą 7 292 kb/s. W warunkach rywalizacji o dostęp do zasobów z innymi konkurencyjnymi strumieniami przyjęta przepustowość sieci pomiarowej równa 10 Mb/s okazała się w tym przypadku zbyt mała dla zapewnienia odpowiedniej jakości obsługi. Pewien pogląd na związek pomiędzy wartością parametru MOS a postrzeganą jakością obrazu daje rys. 19, na którym przedstawiono porównanie wizualne wybranych zrekonstruowanych ramek sekwencji *Cabaret* dla przypadków ustawiania priorytetu pakietów w sposób statyczny oraz dynamiczny.



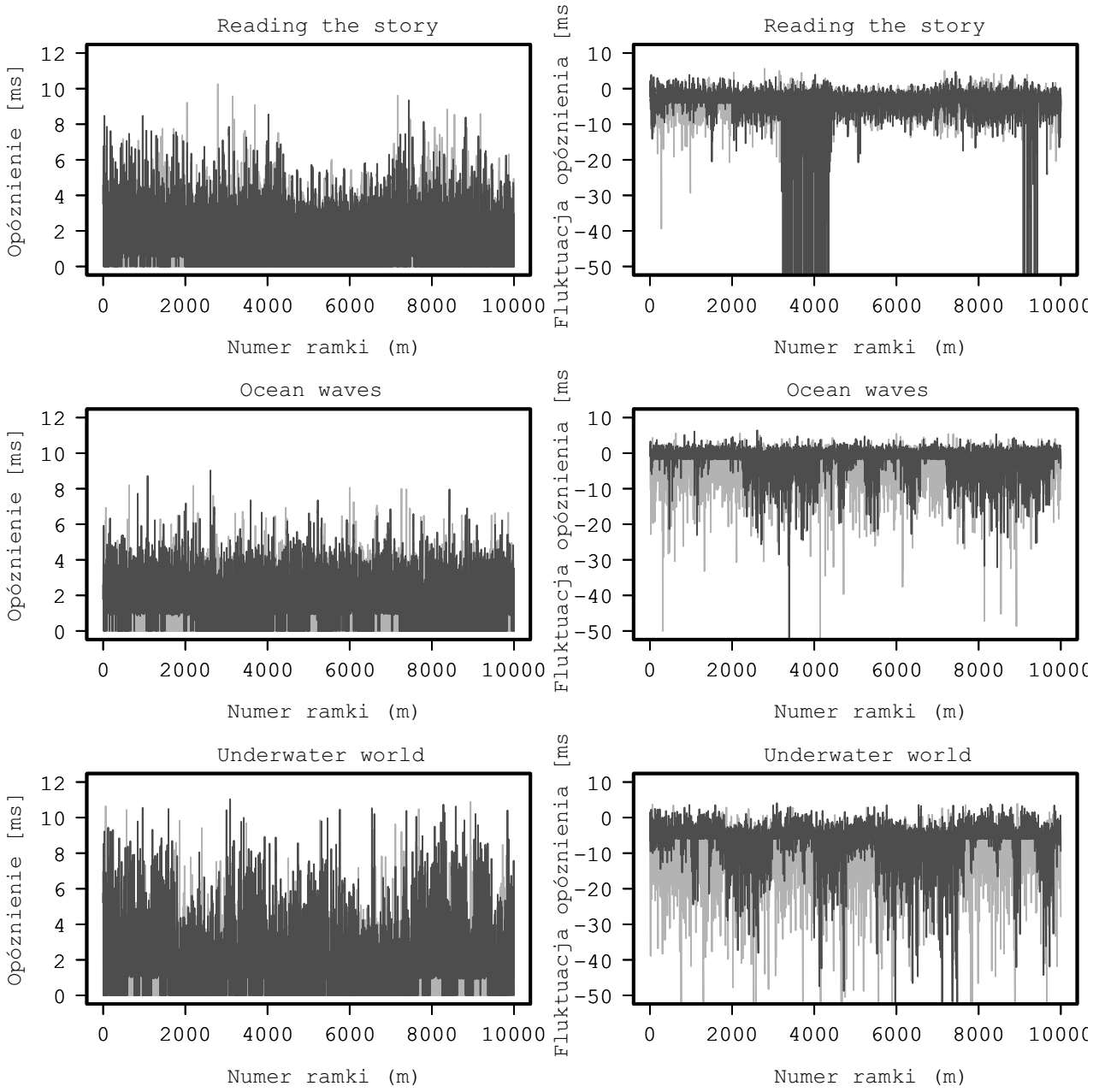
Rys. 19. Porównanie wizualne wybranych zrekonstruowanych ramek sekwencji *Cabaret* dla przypadków ustawiania priorytetu pakietów w sposób statyczny (górny wiersz) oraz dynamiczny (dolny wiersz)

Na rys. 20 dokonano porównania wielkości strat pakietów podczas transmisji badanych sekwencji obrazu. Dla każdej z nich zastosowanie priorytetu dynamicznego powoduje zmniejszenie wielkości strat. Dla sekwencji ze średnią i wysoką złożonością ruchu straty zmniejszają się najbardziej i osiągają poziom poniżej 20%. Zmniejszenie strat przekłada się bezpośrednio na poprawę jakości obrazu.

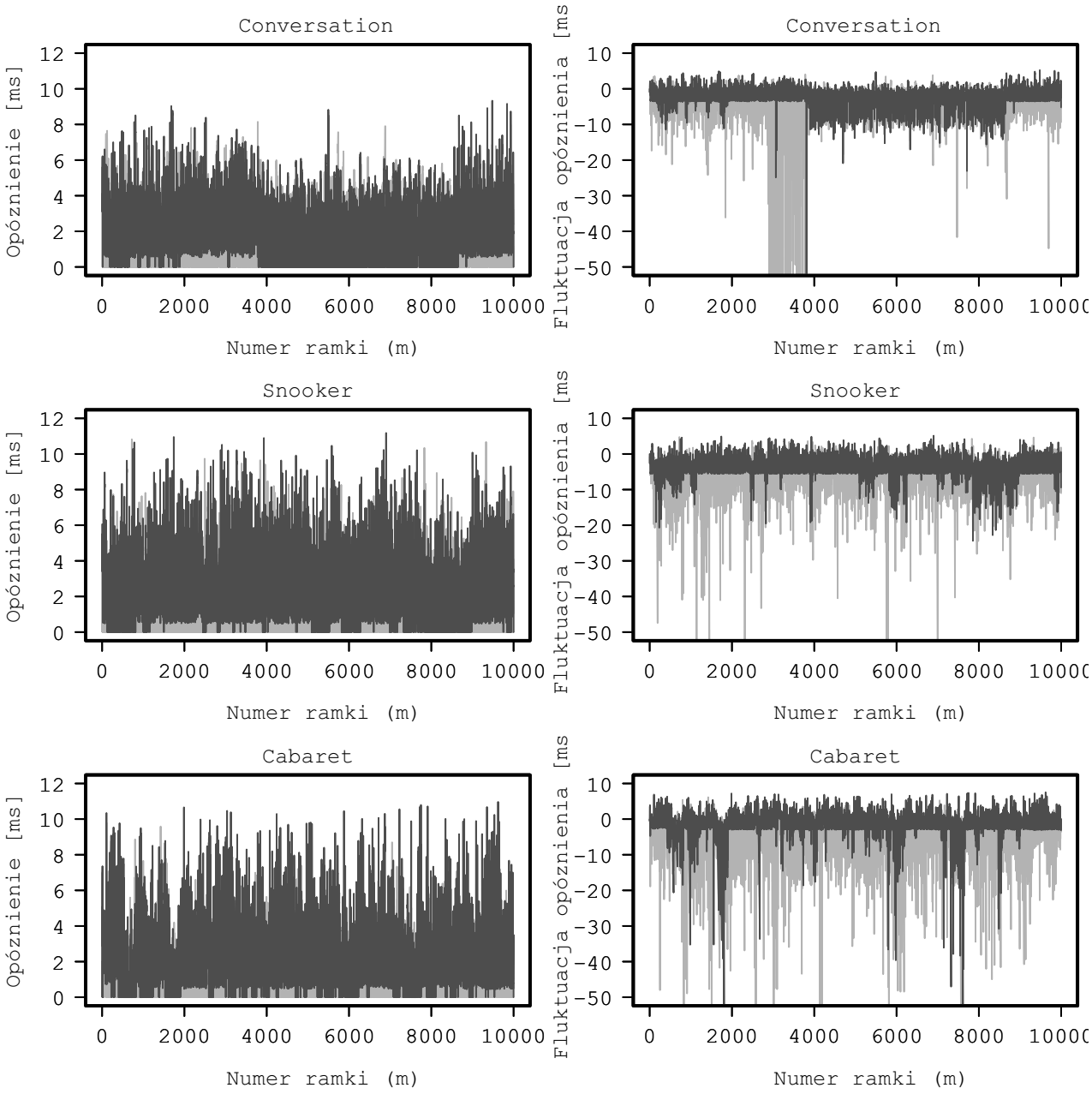


Rys. 20. Porównanie wielkości strat pakietów podczas transmisji badanych sekwencji obrazu

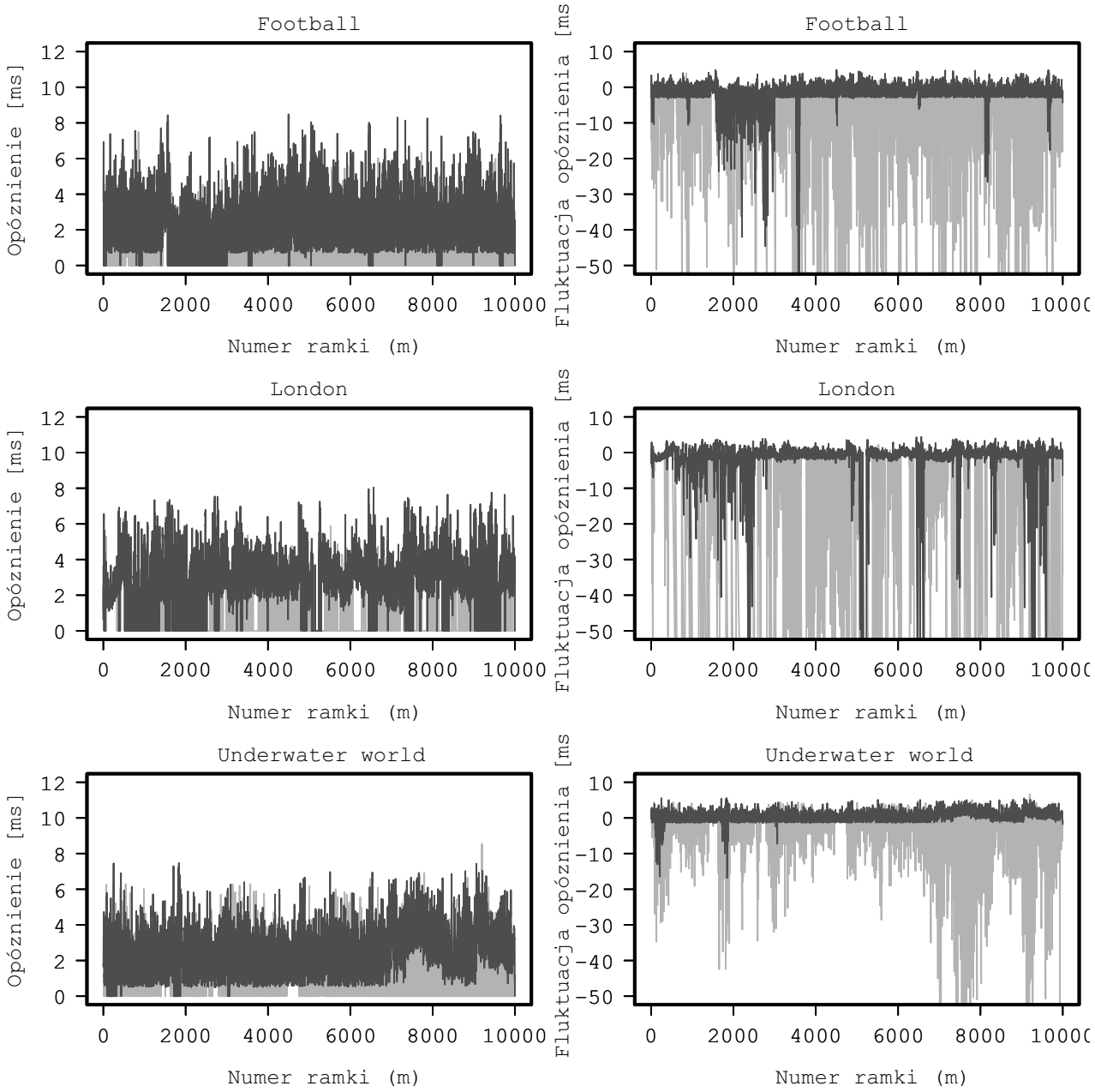
Kolejne 2 parametry, które zostały obliczone w efekcie przeprowadzonych badań, to opóźnienie i fluktuacja opóźnienia pakietów. Zmiany tych parametrów w funkcji numeru ramki obrazu dla każdej badanej sekwencji zostały zaprezentowane na rys. 21, 22 i 23. Wyniki przedstawione na powyższych rysunkach świadczą o tym, iż zmiana priorytetu ze statycznego na dynamiczny miała niewielki wpływ na wielkość opóźnienia pakietów. Jednocześnie można zaobserwować bardzo dużą poprawę fluktuacji opóźnienia pakietów po zastosowaniu priorytetu dynamicznego. Ma to miejsce szczególnie w przypadku sekwencji ze średnią i wysoką złożonością ruchu.



Rys. 21. Opóźnienie i fluktuacja opóźnienia pakietów przy statycznym (kolor jaśniejszy) i dynamicznym (kolor ciemniejszy) ustawianiu priorytetu dla sekwencji z niską złożonością ruchu



Rys. 22. Opóźnienie i fluktuacja opóźnienia pakietów przy statycznym (kolor jaśniejszy) i dynamicznym (kolor ciemniejszy) ustawianiu priorytetu dla sekwencji ze średnią złożonością ruchu



Rys. 23. Opóźnienie i fluktuacja opóźnienia pakietów przy statycznym (kolor jaśniejszy) i dynamicznym (kolor ciemniejszy) ustawianiu priorytetu dla sekwencji z wysoką złożonością ruchu

Podsumowanie

W pracy zaprezentowano możliwość wykorzystania symulatora sieci NS-2 oraz pakietu EvalVid do badania obrazu ruchomego przesyłanego strumieniowo przez sieć z różnicowaniem jakości usług. W efekcie przeprowadzonych badań dokonano porównania jakości obrazu przesyłanego przez sieć dla dwóch przypadków. W pierwszym z nich priorytet pakietów w ramach danej klasy usług AF PHB był ustawiany statycznie. W przypadku drugim miało miejsce dynamiczne ustawianie priorytetu w zależności od aktualnej wartości parametru Hursta strumienia ramek. Wyniki symulacji pokazały, iż zastosowanie proponowanej koncepcji polegającej na ustawianiu priorytetu pakietów w sposób dynamiczny prowadzi do polepszenia jakości obrazu przesyłanego przez sieć w porównaniu z tradycyjnym podejściem statycznym. Poprawie uległy takie parametry jakości obrazu, jak PSNR, MOS, straty czy fluktuacja opóźnienia pakietów. Zaproponowane w artykule adaptacyjne przydzielanie wartości priorytetu może doprowadzić do lepszego wykorzystania zasobów sieci oraz poprawy wydajności Internetu w skali globalnej. Ze względu na długoterminowy charakter parametru Hursta zastosowanego jako kryterium klasyfikacji sposób traktowania pakietów oznaczonych daną wartością priorytetu spełniałby swoje zadanie również w większej skali czasu.

Literatura

Ahmed T., Mehaoua A., Boutaba R., Iraqi Y. (2005): *Adaptive Packet Video Streaming Over IP Networks: A Cross-Layer Approach*, „IEEE Journal on Selected Areas in Communications", February, Vol. 23, No. 2, s. 385–401.

Ahmed T., Boutaba R., Mehaoua A. (2005): *A Measurement-Based Approach for Dynamic QoS Adaptation in DiffServ Networks*, „Computer Communications", Vol. 28, s. 2020–2033.

Bay Y. (2005): *Class-Based Packet Scheduling to Improve QoS for IP Video*, „Telecommunication Systems", Vol. 29, No. 1, s. 47–60.

Boucadair M., Levis P., Griffin D., Wang N., Howarth M., Pavlou G., Mykoniati E., Georgatsos P., Quoitin B., Quoitin B., Rodriguez Sánchez J., Garcia-Osma M. L. (2007): *A Framework for End-to-End Service Differentiation: Network Planes and Parallel Internets*, „Communications Magazine, IEEE In Communications Magazine, IEEE", Vol. 45, No. 9, s. 134–143.

Chen L., Liu G., Zhao F. (2007): *An Improved Marking Mechanism for Real-Time Video over DiffServ Networks*, „Lecture Notes in Computer Science", Vol. 4810, s. 510–519.

Crovella M.E., Bestavros A. (1997): *Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes*, „IEEE/ACM Transactions on Networking", December, Vol. 5, No. 6, s. 835–846.

Grossman D. (2002): *New Terminology and Clarifications for Diffserv*, „IETF RFC 3260", April.

Heinanen J., Baker F., Weiss W., Wroclawski J. (1999): *Assured Forwarding PHB Group*, „IETF RFC 2597", June.

Huynh-Thu Q., Ghanbari M. (2008): *Scope of Validity of PSNR in Image/Video Quality Assessment*, „Electronics Letters", Vol. 44, s. 800–801.

ITU-T Recommendation P.800 (1996): *Methods for Subjective Determination of Transmission Quality*, August.

Ke C-H., Shieh C-K., Hwang W-S., Ziviani A. (2008): *An Evaluation Framework for More Realistic Simulations of MPEG Video Transmission*, „Journal of Information Science and Engineering", Vol. 24, s. 425–440, http://140.116.72.80/~smallko/ns2/myevalvid2.htm.

Klaue J., Rathke B. Wolisz A. (2003): *EvalVid – A Framework for Video Transmission and Quality Evaluation*, Proceedings of the 13th International Conference on Modeling, Techniques and Tools for Computer Performance Evaluation, Urbana, Illinois, September, http://www.tkn.tu-berlin.de/research/evalvid.

McAllister B., Marshall A.J., Woods R.F. (2010): *A Programmable Architecture for Layered Multimedia Streams in IPv6 Networks*, „Journal of Networks", January, Vol. 5, No. 1, s. 65–74.

Park K., Kim G., Crovella M. (1997): *On the Effect of Traffic Self-similarity on Network Performance*, Technical Report CSD-TR-97-024, Purdue University, Dept. of Computer Sciences, April.

Willinger W., Leland W.E., Taqqu M.S., Wilson D. V. (1994): *On the Self-Similar Nature of Ethernet Traffic (extended version)*, „IEEE/ACM Transactions on Networking", February, s. 1–15.