석사학위논문

동영상 재생기를 위한 윈도우 기반 동적 전압조절 알고리즘

- A Window-Based DVS Algorithm for MPEG Player -

지도교수 : 조진성

경희대학교 대학원
컴퓨터공학과

박 경 환

2007년 2월

동영상 재생기를 위한 윈도우 기반 동적 전압조절 알고리즘

- A Window-Based DVS Algorithm for MPEG Player -

지도교수 : 조진성

이 논문을 석사 학위논문으로 제출함

경희대학교 대학원
컴퓨터공학과

박 경 환

2007년 2월

박경환의 공학 석사학위 논문을 인준함

주심교수 : _____ ㊞

부심교수 : _____ ㊞

부심교수 : _____ ㊞

경희대학교 대학원

2007년 2월

국문요약

동영상 재생기를 위한 윈도우 기반 동적 전압조절 알고리즘

경희대학교 대학원
컴 퓨 터 공 학 과
박 경 환

최근 휴대폰, PDA, PMP와 같은 다양한 휴대용 단말기들이 보급이 되고 새로운 모바일 기기들이 증가하면서 많은 사람들이 이러한 휴대용 단말기들을 생활 곳곳에서 유용하게 사용하고 있다. 또한 사용자들은 고화질의 동영상, 게임 등 점차 더 높은 사양의 성능을 요구하게 되면서 배터리로 동작하는 휴대용 단말기들의 전력 소비는 상대적으로 많이 커지게 되었다. 따라서 휴대용 임베디드 시스템의 전력 소모를 줄이기 위해서는 효율적인 전력 관리 기법이 필요하다. 이미 전력 관리 기법으로 많은 기존 연구들이 있지만 대부분은 제한적인 시스템 모델과 가정들로 이루어져 있고 많은 사람들이 사용하는 휴대용 단말기의 특징상 엄격한 실시간성 처리보다는 작업의 처리량(멀티미디어 동영상 등)에 기반한 전력 관리 기법이 요구되나 대부분 경성 실시간 처리 위주이기 때문에 현실적으로 적용

가능한 알고리즘이나 기법은 거의 없다.

따라서 본 논문에서는 기존의 대표적인 전력 관리 기법들에 대해 살펴보고, 현실적으로 적용 가능하고 실용성이 높은 동영상 재생기에서의 동적 전압조절 알고리즘을 제안하고자 한다. 제안하는 알고리즘은 최근 프레임의 정보와 실행 시간 등을 적절한 크기의 윈도우로 유지하며 이를 기반으로 프로세서가 지원하는 (주파수, 전압) 레벨을 조절하여 전력 소비를 낮추게 된다. 이 알고리즘은 간단한 모듈 형태로 구현이 되었으며 일반적인 동영상 재생기에 손쉽게 추가시킬 수 있다. 성능 측정은 실제 환경에서 많이 사용되고 있는 MPlayer를 사용하였으며 수행 결과, 최대 56% 정도의 전력소비 감소 효과를 얻을 수 있었다.

키워드 : 휴대용 단말기, 전력 관리, DVS, 준연성 실시간 시스템

<목 차>

<그 림 목 차>

<표 목 차>

1. 서론

근래에 들어 이동통신의 급격한 발달에 따라 휴대폰(Cellular Phone), PDA, PMP(Portable Multimedia Player), 스마트폰과 같은 다양한 모바일 단말기들의 사용이 빠르게 증가하고 있다. 또한 사용자들은 휴대용 단말기의 성능이 향상될수록 더 높은 품질의 멀티미디어 동영상, 게임 등을 요구하고 있다. 이러한 휴대용 단말기들은 주로 제한된 용량의 에너지를 담고 있는 배터리로 동작하게 되므로 사용자의 요구가 증가할수록 사용 시간은 상대적으로 많이 감소할 수밖에 없다. 물론 휴대용 단말기들의 사용 시간 증대를 위해 대용량의 배터리를 장착하는 방법이 있을 수 있으나 전력 요구량의 증가에 비해 배터리의 기술 발전이 상대적으로 느리기 때문에 용이한 휴대성을 장점으로 하는 휴대용 단말기에 부피가 큰 배터리를 장착할 수가 없는 실정이다. 따라서 배터리의 사용 시간을 증가시키기 위해서는 시스템의 전력 소모를 효과적으로 줄이는 방법이 필요하다.

시스템의 전력 소모를 줄이기 위한 방법으로 많은 연구가 이루어져 왔다. 대표적인 방법으로는 동적 전력관리 기법(Dynamic Power Management: DPM)과 동적 전압조절 기법(Dynamic Voltage Scaling: DVS)이 있다. 동적 전력관리 기법(DPM)은 시스템을 구성하는 컴포넌트들의 사용 패턴에 기반으로 하여 특정 모듈이 장시간 혹은 일정 임계값 이상 사용될 계획이 없다면 전력 소비 모드를 비활성화 상태로 전이함으로써 많은 전력 소모를 줄이는 방법이다. 예를 들어 StrongARM(SA-1100) 프로세서는 활성화 상태(Active State)에서 약 400mW의 전력을 소모하나 비활성화 상태(Sleep State)에서는 겨우 0.16mW 정도의 전력을 소모하듯이 많은 전력 소비를 절감할 수 있다. 하지만 시스템을 구성하는 다양한 모듈은 각기 다른 특성을 가지게 되고 이

들 장치들의 사용 패턴을 정확히 예측하지 못한다면 오히려 상태 전이에 따른 전력 소모가 더 커지게 될 수 있으므로 시스템의 정확한 분석 및 모델링이 요구되며 시간 및 에너지의 부대비용을 적절히 고려해야 한다.

동적 전압조절 기법(DVS)은 태스크가 수행이 될 때 프로세서에 인가되는 클록(Clock Frequency)과 전압(Voltage)을 동적으로 조절하여 전력 소모를 줄이는 방법이다. 일반적으로 CMOS 회로로 구성된 프로세서의 전력 소모는 공급 전압의 제곱에 비례하고 인가되는 클록 또한 공급 전압에 비례하므로 효율적인 전력 소모가 가능하다. 이 기법은 많은 연구가 이루어져 왔으나 대부분은 경성 실시간 시스템(Hard Real-Time System)을 기반으로 하고 있고 또한 알고리즘 위주의 시뮬레이션이 대다수이므로 본 논문에서 고려하고자 하는 준연성 실시간 시스템(Firm Real-Time System)에는 맞지 않다. 본 논문에서는 현재와 같이 휴대용 단말기에서 멀티미디어 응용과 같은 준연성 실시간 시스템을 대상으로 효율적인 전력 소모를 할 수 있는 동적 전압조절 알고리즘을 개발하고 구현하고자 한다.

제안하는 윈도우 기반 동적 전압조절 알고리즘은 최근 프레임의 정보와 실행 시간 등을 적절한 크기의 윈도우로 유지하며 이를 기반으로 CPU의 주파수와 전압 레벨을 조절하여 전력 소비를 낮추게 된다. 이 알고리즘은 간단한 모듈 형태로 구현이 되었으며 일반적인 동영상 재생기에 손쉽게 추가시킬 수 있다. 제안된 알고리즘의 성능 측정 결과 최대 56% 정도의 전력소비 감소 효과를 얻을 수 있었다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대한 내용을 기술한다. 3장에서는 본 논문에서 제안하는 윈도우 기반 알고리즘을 기술하고 4장에서는 구현 및 실험을 통해 성능을 평가하고 분석한다. 마지막으로 5장에서는 결론을 내리고 향후 과제에 대해 기술한다.

2. 관련 연구

2.1 전력관리 개요

휴대용 임베디드 시스템에서 배터리는 그 성능에 따라 차이가 있지만 일반적으로 일정한 양의 에너지를 저장하고 있다. 에너지는 시간의 개념이 포함되기 때문에 큰 전력으로 짧은 시간을 수행하는 것 보다 적은 전력으로 오랜 시간 수행하는 것이 더 많은 에너지를 소모할 수 있으므로 에너지 관리(Energy Management)와 전력 관리(Power Management)가 같은 것이라고 볼 수는 없다. 하지만 멀티미디어 동영상 재생과 같이 일정한 구간동안 주기적으로 처리되는 작업은 전력의 소모만큼 에너지가 비례하여 소비되기 때문에 같은 의미로 사용될 수 있다. 본 논문에서는 동영상 재생기를 위한 동적 전압조절 알고리즘을 제안하고자 하기 때문에 이 두 가지를 같은 의미로 사용하기로 한다.

2.2 전력관리 기법

기존의 전력 소모를 줄이기 위한 방법으로는 회로 설계 수준인 하드웨어 레벨과 시스템, 소프트웨어 레벨로 나뉠 수 있다. 그러나 점차 마이크로프로세서의 성능이 향상되고 네트워크의 대역폭이 커짐에 따라 휴대용 단말기에서 동영상, 게임 등 소프트웨어의 전력 소모가 급격히 증가하게 되면서 하드웨어 레벨보다는 시스템, 소프트웨어 레벨에서의 전력 관리가 더 절실하게 필요하다. 따라서 본 장에서는 시스템, 소프트웨어 레벨에서 대표적인 전력 관리 기법인 동적 전력관리 기법(DPM)과 동적 전압조절 기법(DVS)에 대해 기술하고 본 논문에서 제안하고자 하는 알고리즘의 동기를 살펴보도록 한다.

2.2.1 동적 전력관리 기법(DPM)

동적 전력관리 기법(Dynamic Power Management: DPM)은 시스템의 구성 요소별로 사용된 패턴에 기반 하여 특정 모듈이 장시간 혹은 일정 임계값 이상 사용될 계획이 없다면 비활성화 상태로 전이하게 됨으로써 전력 소모를 줄이는 방법이다[1, 5, 9, 12, 13, 16, 17, 21].
[그림 1]은 이러한 동적 전력관리 기법에 대해 간략히 도식화한 것이다.

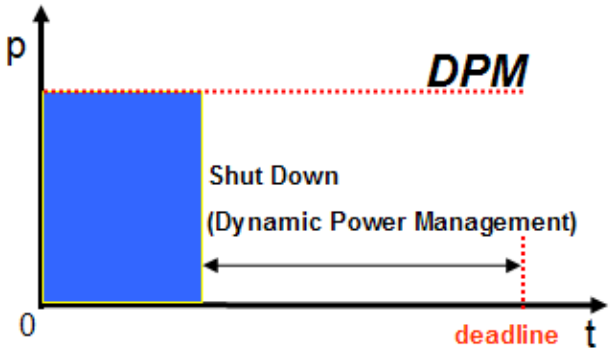


그림 1. 동적 전력관리 기법

동적 전력관리 기법이 적용되는 대표적인 시스템 모듈로는 프로세서, 메모리, 통신 인터페이스 등이 있다. 이들은 일반적으로 시스템 전체의 전력 소모 중 가장 큰 비율을 차지하고 있는 구성요소들이다. 실례로 프로세서에 적용되는 동적 전력관리 기법을 살펴본다면 아래 [그림 2]는 StrongARM(SA-1100)에서의 프로세서 상태 전이를 나타내는 그림이다[1].

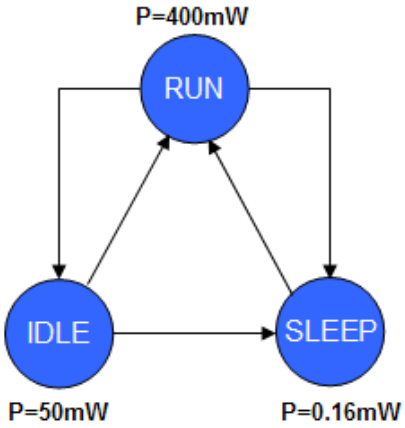


그림 2. StrongARM(SA-1100)에서의 프로세서 상태 전이도

위 그림에서 알 수 있듯이 Run 상태와 Idle 상태는 약 8배정도의 전력 소모의 차이가 나고 Sleep 상태는 극히 적은 양의 전력 소모가 요구되는 것을 볼 수 있다. 즉 적절히 상태 전이를 한다면 많은 에너지 효율을 얻을 수 있게 된다. 하지만 이러한 시스템을 구성하는 다양한 모듈은 각기 다른 특성을 가지게 되고 이들 장치들의 사용 패턴을 정확히 예측할 수 없다면 상태 전이에 따른 전력 소모가 더 커지게 될 수 있으므로 어려움이 있다. 예를 든다면 기기 A가 Sleep Mode로 전환하기 위해 기기 B로부터 정보를 받아야 하는 경우, 만약 B가 이미 Sleep Mode로 들어갔다면 A를 위해 다시 Wake-Up Mode로 돌아와야 하는 경우가 발생할 것이다. 이와 같이 시스템의 동작을 고려한 상태에서 상태 전이에 따르는 시간 및 에너지의 부대비용을 적절히 고려한 동적 전력관리 기법이 이루어져야 한다.

2.2.2 동적 전압조절 기법(DVS)

동적 전압조절 기법(Dynamic Voltage Scaling: DVS)은 프로세서에 인가되는 전압을 동적으로 조절하여 전력 소모량을 줄이는 방법이다. 보통 전력 소모는 공급 전압의 제곱에 비례하고 또한 전압은 주파수와 비례하는 관계에 있다[2-4, 6-11, 14-16, 18-20].

$$E \propto V^2 \rightarrow F_{\max} \propto V \qquad \text{(식 2.1)}$$

그러므로 주어진 작업의 요구 조건을 만족하면서 전압과 주파수의 관계를 적절히 조절하면 전력 소모를 효율적으로 줄일 수 있다. [그림 3]은 동적 전압조절 기법에 대해 간략히 도식화한 것이다.

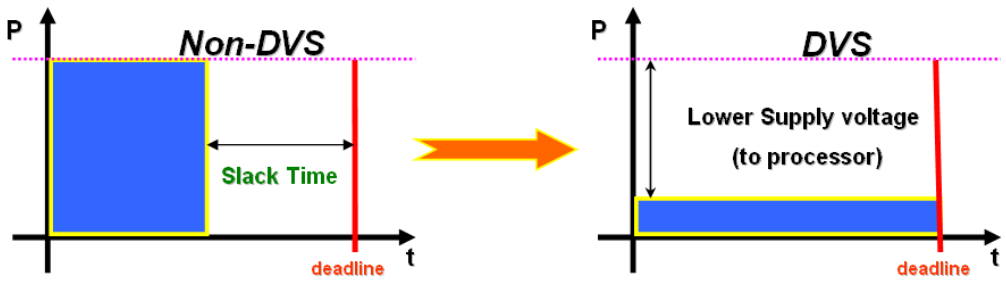


그림 3. 동적 전압조절 기법

일반적으로 동적 전압조절 기법은 [그림 3]과 같이 실시간 작업들이 가지는 임계점 내에서의 여유 시간을 잘 분배하여 공급되는 전압과 주파수를 동적으로 변화하게 된다. DVS 기법은 크게 실시간 시스템과 비실시간 시

스템 두 가지로 분류되어 질 수 있다. 실시간 시스템에서의 DVS 기법은 태스크들이 가지는 여유 시간(idle time, slack time)을 정확히 예측하여 공급 전압과 주파수를 바꾸는 방법이 일반적이며 크게 Intra-Task DVS, Inter-Task DVS로 나뉠 수 있다[3, 4, 6, 7]. 비실시간 시스템에서의 DVS 기법은 시스템에 대한 작업 부하량을 정확히 예측, 공급 전압과 주파수를 바꾸는 방법이 사용되며 이때는 프로세서의 작업량을 주기적으로 모니터링하는 Interval-based DVS 기법이 있다[2, 8, 10, 11, 14, 16, 20]. 이 밖에도 프로세서 하나만을 고려하는 것이 아니라 (CPU, Memory), (CPU, WNIC), (CPU, LCD), (CPU, Battery) 등 프로세서와 다른 시스템 모듈 사이의 관계를 고려한 다양한 기법들이 존재한다[8, 19].

2.2.3 Video-specific DVS

최근에는 멀티미디어 동영상을 위한 DVS 기법이 제안되고 있다[2, 10, 14]. Burchard와 Altenbernd는 두 단계에 걸쳐 프레임의 디코딩을 나누어 공급 전압과 주파수를 낮추는 방법으로, 프레임을 디코딩하기 위한 여러 정보들을 알고 있어야 하며 일반적인 동영상 재생기에 많은 수정을 가해야 하는 단점이 있다. Poiwelse는 프레임의 타입과 프레임의 크기로부터 실행시간을 측정하고 측정된 값을 기반으로 공급 전압과 주파수를 낮추는 방법이다. 이 방법 역시 실행 시간을 측정하기 위해서는 디코더에 수정을 가해야 하는 단점이 있다. Richard와 Gilles는 Kiosk(공공장소에 설치된 터치스크린 방식의 정보전달 시스템)과 같이 일정한 크기와 동일한 구간으로만 되어 있는 멀티미디어 동영상에서의 DVS 기법을 제안하였다[14]. 이 알고리즘은 동영상을 재생할 때 프레임의 실행 시간을 보고 공급 전압과 주파수를 변경하게 되며 변경되는 정보를 History에 저장하게 된다.

kiosk의 특성상 일정한 구간의 정해진 동영상을 재생하는 것이기 때문에 한번 수행이 되고 나면 History에 DVS가 수행된 값이 저장이 되므로 다음번부터 실행할 때는 해당 값을 참조하여 그대로 수행하면 되므로 많은 전력소모 감소 효과를 얻을 수 있다. 다만 일반 사용자들이 이용하는 휴대용 단말기에서는 일정하게 정해진 동영상이 아니기 때문에 최적의 성능을 낼 수 없는 단점이 있다. 여기까지 관련 연구를 살펴보았다. 이를 바탕으로 본 논문에서는 실제 많은 사람들이 사용하는 휴대용 단말기를 대상으로 하며 멀티미디어 동영상에 특화된 DVS 알고리즘을 제안하고자 한다.

3. 윈도우 기반 DVS 알고리즘

본 논문은 군사무기, 공장의 작업 제어 시스템과 같이 주어진 마감시간(Deadline)내에 반드시 작업을 처리해야 하는 경성 실시간 시스템(Hard Real-Time System)을 대상으로 하는 것이 아니라 멀티미디어 응용 프로그램과 같이 일정한 주기로 동작하면서 마감시간에 비교적 덜 제약적인 준연성 실시간 시스템(Firm Real-Time System)에서의 동적 전압조절 알고리즘을 제안하고자 한다. 준연성 실시간 시스템은 마감시간을 준수하지 못할 경우 작업의 결과의 가치가 일시적으로는 상실되나 시스템에 지속적으로 문제를 발생시키지 않는 특징을 가지고 있으므로 멀티미디어 응용 프로그램이 대표적인 예라 할 수 있다. 따라서 본 장에서는 수행되고 있는 멀티미디어 응용 프로그램의 품질을 최대한 보장하는 범위 내에서 전력 소모를 최소화할 수 있는 윈도우 기반 DVS 알고리즘을 제안한다.

3.1 문제 정의

일반적으로 시스템에서 제공되는 DVS 가능한 프로세서는 정해진 범위의 주파수와 전압 레벨이 있다. 따라서 클록 주파수는 f_i , 인가되는 전압을 v_j 라고 한다면 (f_i, v_j) 집합으로 나타낼 수 있다. 이 때 수행되는 태스크(동영상 재생 프로그램)를 T 라고 한다면 하나의 프레임이 재생되는데 걸리는 시간을 T_{Exe} , 수행되고 있는 동영상의 프레임 당 주기는 T_{Period} 라고 하며, 여유 시간 T_{Slack} 은

$$T_{Slack} = T_{Period} - T_{Exe} \quad \text{(식 3.1)}$$

[식 3.1]과 같이 표현할 수 있다. T_{Slack} 은 프로세서가 수행되는 (f_i, v_j) 레벨이 낮을수록 0 에 가까워지거나 음수(프레임의 주기를 넘어서게 됨)가 된다. 따라서 사용자의 QoS를 만족하는 범위 내에서 전력 소모를 최소한으로 줄이는 방법은 T_{Slack} 이 음수가 되지 않으면서 제로에 가깝도록 하는 최적의 (f_i, v_j) 레벨을 찾아야 한다. 멀티미디어 동영상 프로그램의 특성상 매 프레임마다 크기가 달라지기 때문에 일정하게 (f_i, v_j) 레벨이 유지되지 않을 수 있으므로 동영상 재생동안 동적으로 지속적인 (f_i, v_j) 레벨을 구하여 적용하도록 한다.

따라서 본 논문에서는 다음과 같이 문제를 정의한다.

· 동영상이 재생되는 동안 사용자의 QoS를 만족하는 범위 내에서, $\sum_{i=1}^{k} T_{Slack[i]} (k = Total\ frame)$ 이 최소화 될 수 있도록 하는 최적의 (f_i, v_j) 레벨을 구한다.

3.2 제안하는 WB-DVS 알고리즘

제안하는 WB(Window-Based)-DVS 알고리즘은 최근까지 수행된 프레임의 정보(프레임이 출력되는 시간과 다음 프레임의 종류)를 일정한 크기의 윈도우 큐($W = Window\,Queue\,Size$)에 유지하여 그 값을 바탕으로 여유시간 평균값(T_{Slack}^{Avg})을 구한다.

$$T_{Slack}^{Avg} = \sum_{i=1}^{W} (a_i \cdot T_{Slack[i]}) \quad (\ a_i < a_{i+1}, \ \sum_{i=1}^{W} a_i = 1 \ ) \qquad \text{(식 3.2)}$$

$$a_i = i \cdot \frac{2}{W(W+1)} \quad (W = Window\,Queue\,Size, W > 0 \ ) \qquad \text{(식 3.3)}$$

a_i 는 T_{Slack}^{Avg} 을 구할 때 W 중 가장 최근의 값에 더 많은 가중치를 두어 계산 값의 정확도를 높이기 위한 변수이며 W 의 값은 가변적으로 변할 수 있기 때문에 실험 결과를 통해 적당한 값을 얻을 수 있다. 이 값을 통해 구해진 T_{Slack}^{Avg} 값은 초기에 정해진 임계값(Th_{Down} , Th_{Up}) 과 비교하여 (f_i, v_j) 레벨을 올릴지 낮출지를 결정하게 되며 이에 따라 임계값도 조절하게 된다. 또한 시스템에 무리가 가지 않도록 최소한의 프레임 Drop(F_{Drop})도 결정한다. [그림 4]와과 [표 1]은 WB-DVS 알고리즘의 전체 동작에 대한 개요와 윈도우 큐를 나타낸 것이고 [그림 5]는 윈도우 큐 크기가 3일때 동작하는 모습을 간략하게 나타낸 그림이다.

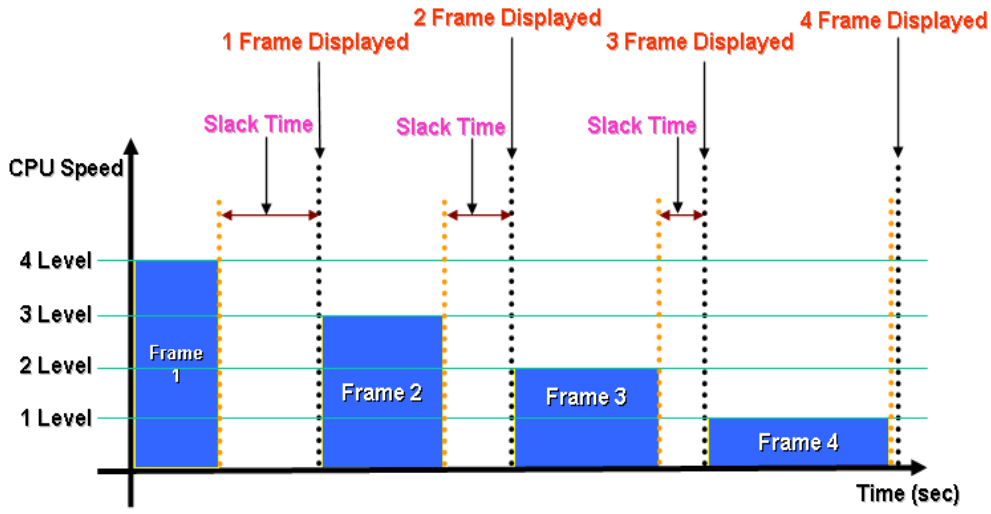


그림 4. WB-DVS 알고리즘 개요도

1 Frame Displayed	2 Frame Displayed	3 Frame Displayed	4 Frame Displayed	5 Frame Displayed	
$N_{frame\ Type}$	$N_{frame\ Type}$	$N_{frame\ Type}$	$N_{frame\ Type}$	$N_{frame\ Type}$	$\Rightarrow$
T_{Exe}	T_{Exe}	T_{Exe}	T_{Exe}	T_{Exe}	
1	2	3	4	5	

표 1. WB-DVS 알고리즘 윈도우 큐

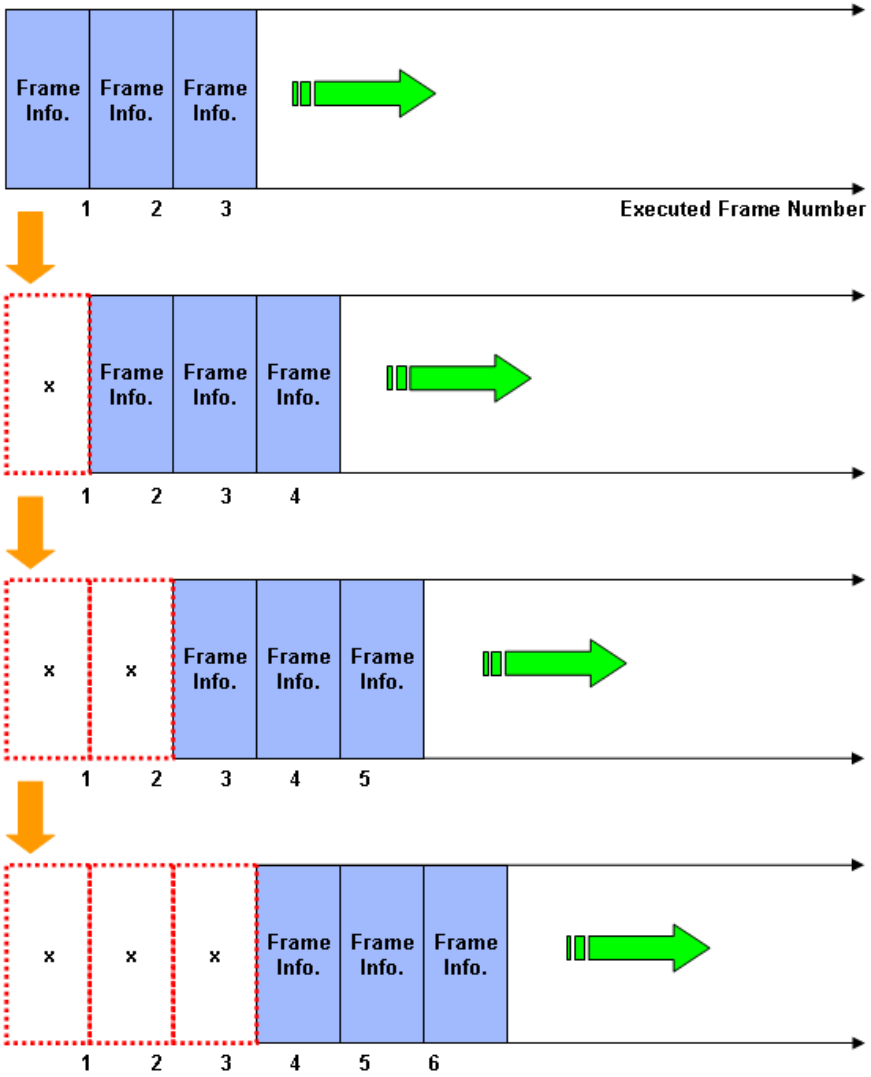


그림 5. 윈도우 큐의 동작 예제 ($W=3$)

[그림 4]를 보면 프레임이 수행될 때마다 프로세서의 속도(f_i, v_j) 가 낮아져 T_{Slack} 이 줄어드는 모습이 보이고 4 프레임에서는 거의 제로에 가깝게 동작한다. 이에 따라 수행되는 프로세서의 속도는 가장 낮은 1레벨에서 동작하므로 많은 전력 감소 효과를 가져 오게 된다. 물론 4 프레임 이후로 지속적으로 1 레벨에서 동작하리라는 보장은 없다. 일반적으로 동영상은 무수히 많은 GOP(Group of Pictures)로 구성되어 있고 GOP의 가장 앞 쪽부터 프레임의 비트량이 가장 많은 I frame으로 구성되어 있으므로 새로운 GOP가 시작되면 다시 프로세서의 속도를 증가시켜야 하며, 하나의 GOP안에서도 프레임의 비트량에 따라 프로세서의 속도를 적절하게 변화시켜야 QoS를 보장할 수 있다. 이때 비트량이 클수록 디코더의 수행 시간이 길어지므로 결과적으로 프레임이 화면에 출력되는 실행 시간과 비례한다고 할 수 있다. [그림 6]은 이러한 멀티미디어 동영상의 특성으로 인해 (f_i, v_j) 레벨이 바뀌게 되는 모습을 예로 나타낸 것이다. [그림 6]과 같이 하나의 GOP가 끝나고 새로운 GOP가 시작될 때 (f_i, v_j) 레벨이 올라간 모습을 볼 수 있다.

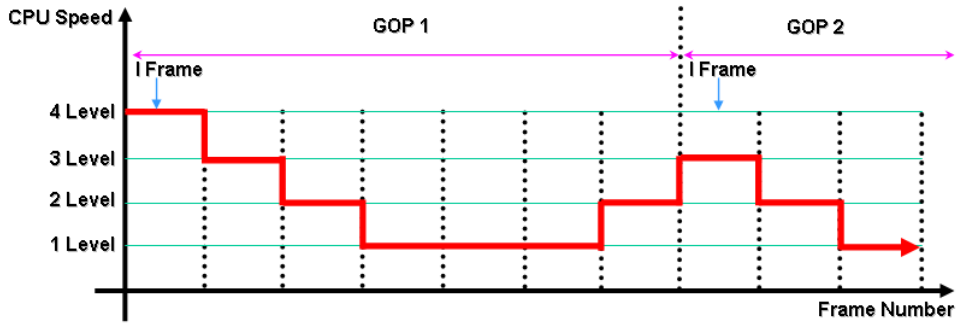


그림 6. WB-DVS 알고리즘 개요도

그러나 위 그림과 같이 동작할 때 발생될 수 있는 문제점은 너무 빈번

한 프로세서의 속도 변화는 오히려 시스템의 성능을 악화시켜 사용자가 원하는 QoS를 보장할 수 없다는 것이다. 따라서 일정한 임계값(Th_{Down} , Th_{Up})을 두어 해당 범위 내에서 동작한다면 프로세서의 속도를 변화시키지 않으며 이것이 누적되어 동영상 재생에 무리를 일으킬 수 있다면 적절한 프레임 Dropping(F_{Drop})을 통해 성능을 유지하도록 한다.

본 논문에서 제안하는 WB-DVS 알고리즘은 간단한 모듈 형태로 구현이 되기 때문에 손쉽게 일반적인 동영상 재생기에 추가할 수 있다. 하지만 알고리즘이 동작하기 위해 필요한 정보는 동영상 재생기로부터 넘겨받아야 한다. 이 값들은 동영상 재생기에서 꼭 필요한 정보이므로 기존 소스의 수정이 없이 넘겨주기만 하면 된다. [표 2]는 이와 같이 넘겨 받아야 하는 Input Value를 나타낸 것이고 [표 3]은 알고리즘이 동작한 후에 발생하는 출력 값을 나타낸 것이다.

Input Value

① T_{Exe} (Frame Execution Time), $N_{frame\,Type}$ (Next Frame Type)

② $T_{Period} = \frac{1}{fps}$

③ Current (f, v) Level

표 2. WB-DVS 알고리즘 입력 값

Output Value
① Adaptation of (f, v) Level ② Drop Frame Number

표 3. WB-DVS 알고리즘 출력 값

제안하는 알고리즘의 동작은 동영상이 재생될 때 매 프레임이 화면에 출력된 직후에 호출이 된다. 따라서 하나의 동영상이 수행될 때 알고리즘의 동작 횟수는 해당 동영상의 전체 프레임 수와 동일하다. 지금까지 기술한 본 논문의 알고리즘의 전체 동작은 다음 장에 나타낸 [표 4]와 같이 나타낼 수 있다.

Window-Based DVS Algorithm	
Ⅰ. Average Slack Time Computation Phase Ⅱ. Frame Drop Decision Phase Ⅲ. (f, v) Level Adaptation Phase	
Ⅰ	$T_{Slack[i]} = T_{Period[i]} - T_{Exe[i]} \ (0 \le i \le W)$ $T_{Slack}^{Avg} = \sum_{i=1}^{W} (a_i \cdot T_{Slack[i]}) \ \ (\ a_i < a_{i+1}, \ \sum_{i=1}^{W} a_i = 1 \ )$
Ⅱ	$If \ (\ T_{Slack}^{Avg} < 0 \ \ \text{and} \ \ Freq = Highest)$ $F_{Drop} = F_{Drop} + 1$ $End \ If$
Ⅲ	$If \ (\ T_{Slack}^{Avg} > 0 \ \ \text{and} \ \ T_{Slack}^{Avg} > Th_{Down} \ )$ $If \ (\ N_{frame\,Type} \neq I_{frame} \ )$ $Th_{Down} = (\ Th_{Down} + T_{Slack}^{Avg} \ ) \ / \ 2$ $(f_i, v_j) = (f_i, v_j) - 1$ $End \ If$ $Else \ If \ (\ T_{Slack}^{Avg} < 0 \ \ \text{and} \ \ T_{Slack}^{Avg} < Th_{Up} \ )$ $(f_i, v_j) = (f_i, v_j) + 1$ $End \ If$

표 4. WB-DVS 알고리즘 의사 코드

4. 구현 및 실험

4.1 구현 및 실험 환경

본 논문에서 제안하는 알고리즘을 구현한 시스템 사양은 아래 [표 5]와 같다. 저전력을 지원하는 인텔사의 PXA255 프로세서가 탑재되어 있고 특별히 전력 소모를 측정하기 위한 핀이 주요 모듈에 장착되어 있어 DAQ (Data Acquisition)장비를 사용해 전력 소모 측정이 가능한 Embedded Board를 사용하였다. 프로세서가 지원하는 실제 (f, v) 레벨은 4 단계이며, [표 6]에 각 레벨에 해당하는 소비전력을 나타내었다. 또한 제안한 알고리즘의 구현 모듈을 추가할 동영상 재생기는 신뢰성을 높이기 위해 많은 사용자들에게 사용되는 MPlayer-0.90 버전을 사용하였으며 소스코드의 수정이 거의 없이 모듈을 손쉽게 추가하였다.

Board	Tynux-Box xe Board (Palm Palm Tech.)
Processor	PXA255
Supported (f, v) Level	(99.5MHz, 1.0V), (199.1MHz, 1.0V) (298.6MHz, 1.1V), (398.1MHz, 1.3V)
OS	Embedded Linux 2.4.19
MPEG Player	MPlayer 0.90

표 5. 구현 환경

398.1MHz Active Mode	411mW
298.6MHz Active Mode	283mW
199.1MHz Active Mode	178mW

표 6. PXA255 프로세서의 소비 전력

4.2 실험 방법 및 결과 분석

제안하는 알고리즘의 성능 측정을 위해 National Instruments사의 NI DAQPad-6015 라는 데이터 수집 하드웨어 장비를 사용하였다. 이 장비는 Host PC에 USB 타입으로 연결이 되며, 프로세서가 사용하는 전력 값을 직접 측정하는 것이 아니라 Tynux Board에 전류가 공급되면 프로세서를 통과한 후에 측정이 되는 전압(전압 강하)을 실시간으로 저장하게 된다. 따라서 아래의 전력 측정 공식을 사용하여 프로세서가 사용한 전력 소모량을 알아낼 수 있다.

$$P = I \times V = \frac{V_1 \times V_2}{R} \quad (\ V_1\text{:공급 전압},\ V_2\text{:출력 전압}) \qquad \textbf{(식 4.1)}$$

R 은 프로세서에 달려있는 저항의 값이며 V_1의 값은 프로세서에 인가되는 공급 전압이므로 저항을 거쳐 강하된 V_2(출력 전압)을 알 수 있다면 P 의 값을 측정할 수 있다. 성능 측정을 위해 사용된 동영상 클립은 [표 7]에 나타내었듯이 두 종류의 다른 동영상을 4개의 fps(Frame Rate : frame per second)로 만들었으며 전체 8개의 동영상으로 실험을 수행하였다.

실험 방법은 우선 알고리즘이 적용되지 않은 일반 MPlayer-0.90의 소비 전력을 구하였고 제안한 알고리즘에서 T_{Slack}^{Avg} 값을 계산할 때 가중치를 적용하는 [식 3.3]의 비교를 위해 $a_i = \frac{1}{W}$ 라는 평균값을 적용해 따로 측정하도록 하였다. 또한 알고리즘이 가장 우수한 성능을 발휘할 수 있는 윈도우 큐($W = Window\,Queue\,Size$) 크기를 알기 위해 1 부터 13 까지 차례로 변화시키면서 실험을 하였고, 또한 알고리즘의 성능의 검증을 위해

- Power Consumption of each MPEG-Clip (10, 15, 20, 25 fps)
- Number of (f, v) Level change
- Number of Frame's Deadline Miss
- Number of dropped Frames

이와 같이 4가지의 분류로 측정하였다. 실험의 신뢰도를 위해 모든 실험은 각각 3번씩 수행하였다.

Name	Superman-Returns.avi				XMEN-3.avi			
Frame Rate (fps)	10	15	20	25	10	15	20	25
Total Frames	611	916	1221	1525	667	998	1330	1662
Data Rate (Kbps)	336	376	424	464	312	352	392	440
Encoder	Divx 4(MPEG-4)							
Screen Size	320 x 240							
Play Time	60 seconds							

표 7. 실험에 사용된 동영상 기본 정보

4.2.1 소비전력 측정 결과

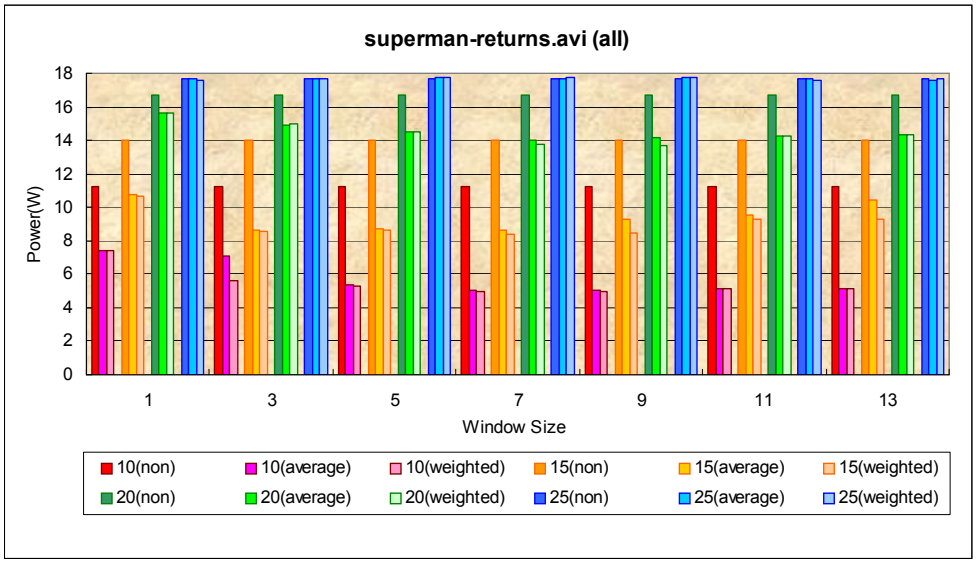


그림 7. Superman-returns 동영상의 소비 전력 측정값

[그림 7]은 superman-returns 동영상을, 알고리즘을 적용하지 않은 일반 MPlayer에서의 소비전력 측정값, 그리고 제안한 알고리즘에서 $a_i = \frac{1}{W}$ 일 때와 $a_i = i \cdot \frac{2}{W(W+1)}$ 일 때의 각각의 소비전력 값을 계산한 그래프이다. 측정된 값에 의하면 모든 경우에서 W 가 약 5에서 7정도 크기일 때 가장 소비전력이 낮음을 알 수 있다. 10 fps의 경우 W 가 7이고 가중치를 적용한 a_i를 사용했을 경우 일반 MPlayer보다 최대 56% 정도의 전력소모 절감효과를 보인다. 주목할 만한 점은 일반 MPlayer에서는 25 fps 동영상을 수행할 때 비디오와 오디오의 싱크가 맞지 않아 종료시 약 5초 정도의 차이를 보인다는 것이다. 이는 하드웨어의 성능이 25 fps 동영상을 재생시키는데 무리가 있다는 것을 나타내며, 본 논문에서 제안한 알고리즘을 적용시키면 25 fps 동영상도 무리 없이 재생이 가능하다. 그 이유는 알고리즘 내부에서 여유 시간이 지속적으로 초과할 때에는 프레임 Dropping이 일어나기 때

문이다. [그림 8]은 측정된 결과 값을 fps별로 구분한 것이다.

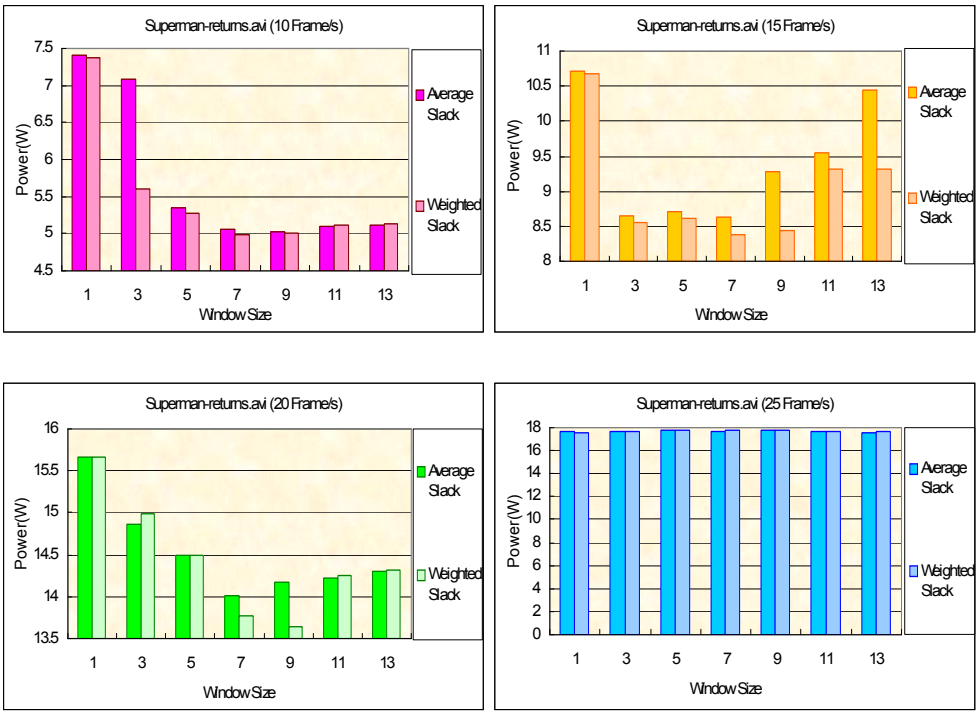


그림 8. Superman-returns fps별 소비 전력 측정값

위 그림을 살펴보면 25 fps를 제외하고 소비전력 값이 W가 약 7을 기점으로 감소하다가 다시 증가하게 된다. 이 결과 값으로 알 수 있는 것은 W의 크기가 무조건 큰 것이 성능 향상에 도움이 되지 않으며, 소비전력이 증가한다는 것은 W의 크기가 증가함으로 인해 최적의 프로세서 속도 레벨을 찾지 못한다는 얘기와 동일하다. 위 동영상의 결과는 [그림 9]에서 측정한 X-MEN3 동영상에서도 거의 비슷하게 나타남을 알 수 있다.

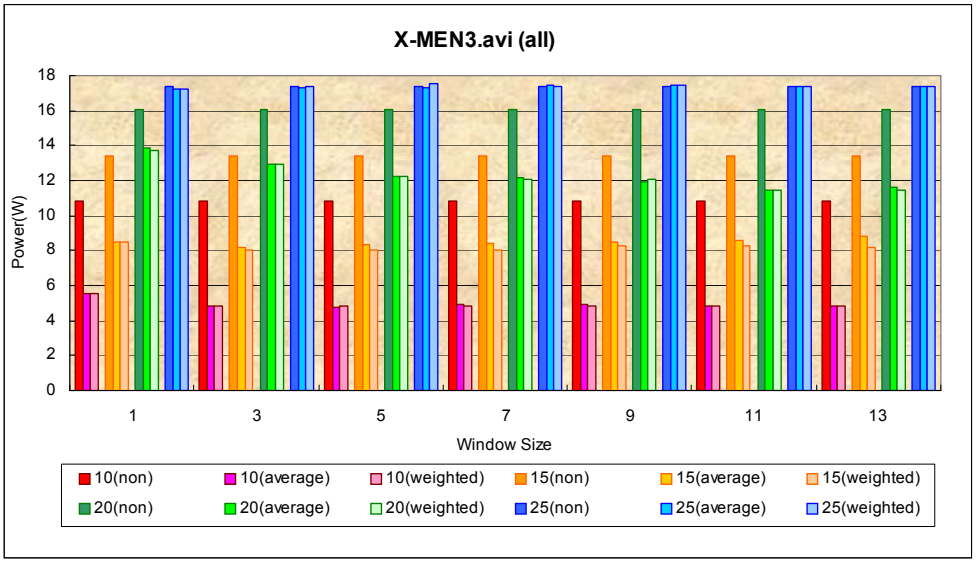


그림 9. X-MEN3 동영상의 소비 전력 측정값

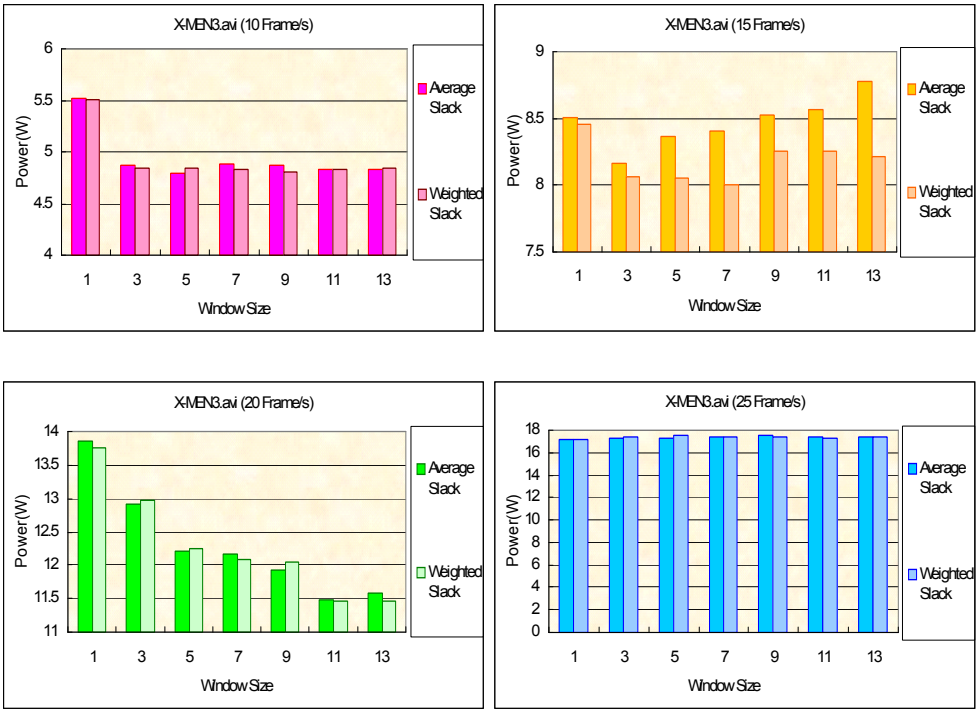


그림 10. X-MEN3 fps별 소비 전력 측정값

[그림 10]의 20 fps 동영상의 경우 W 가 7 이후에도 전력 소모가 더 낮아진 것으로 측정되었으나, W 가 9 일 때부터는 오디오와 비디오의 동기화가 제대로 이루어지지 않고 평균적으로 약 1초 정도의 지연이 발생하였다. 그 외 다른 10, 15, 25 fps의 경우 이전 superman-returns 동영상에 서와 같이 비슷한 형태의 결과로 나왔음을 알 수 있다.

4.2.2 프로세서 (f, v)레벨 변경 측정 결과

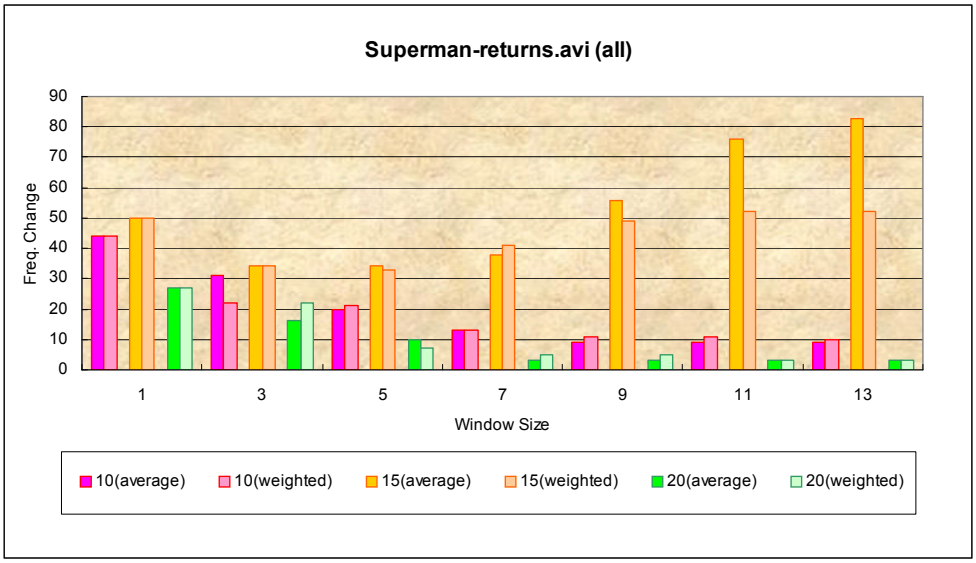


그림 11. Superman-returns (f, v)레벨 변경 측정값

이번 절에서는 제안한 알고리즘이 동작할 때 프로세서의 (f, v) 레벨 변경이 얼마나 이루어졌는지를 측정하였다. 25 fps의 경우 (f, v) 레벨의 변경은 한번도 이루어지지 않았다. 위에서 언급했듯이 25 fps 자체가 일반 MPlayer로 수행시켰을때도 비디오와 오디오의 동기화가 맞지 않기 때문에 프로세서의 속도를 변경하는 대신 프레임을 일정 구간마다 지속적으로 Dropping하여 무리 없이 재생이 가능하도록 하였기 때문이다. 10, 20 fps의 경우 W 가 증가할수록 변경 횟수가 줄어들며 7 이후에는 거의 비슷한 형태를 보이게 된다. 20 fps의 경우 낮아지다가 다시 변경 횟수가 증가하게 되는데, 이와는 반대로 4.2.1.절에서의 소비전력 결과를 보면 알 수 있듯이 W 가 7 이후에는 전력 소모량이 증가하게 된다. 그 이유는 W 의 증가로 부정확

하게 계산된 여유 시간 때문에 낮은 단계로 감소하였다가 다음과 그 다음 프레임에 연속적으로 (f, v) 레벨을 올리는 등 오히려 불필요한 레벨 변경으로 인해 전력 소모를 증가시키기 때문이다.

[그림 12]는 fps 별로 세분화된 그래프를 나타낸 것이다.

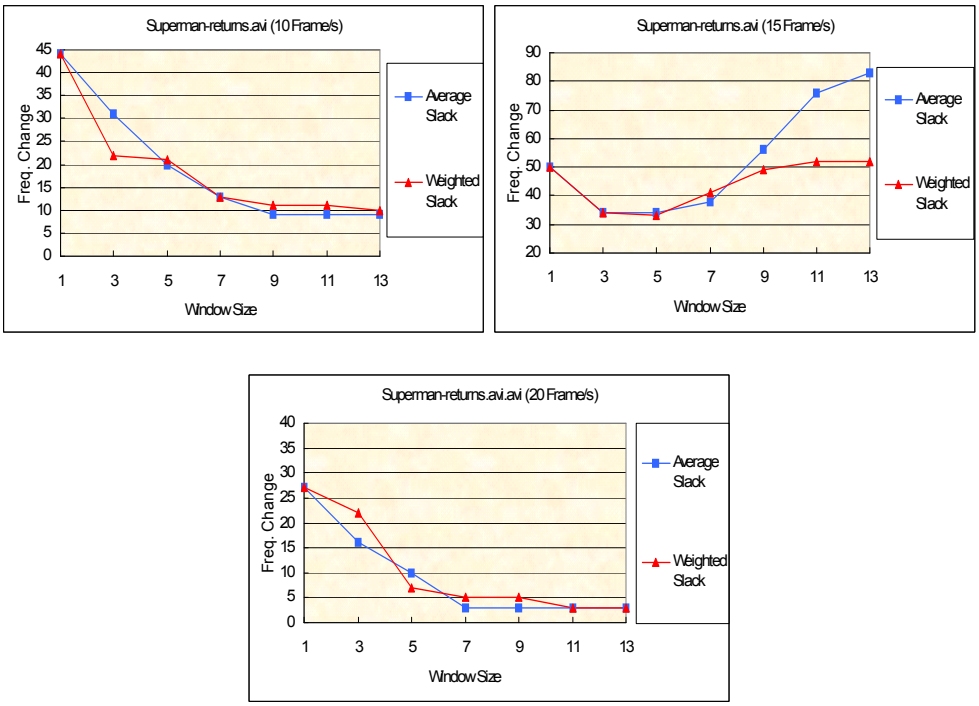


그림 12. Superman-returns fps별 (f, v)레벨 변경 측정값

[그림 13]과 [그림 14]는 Superman-returns 동영상과 동일한 방법으로 X-MEN3 동영상을 수행시켰을 때 (f, v) 레벨 변경을 나타낸 것이다. 그래프를 보면 알 수 있듯이 Superman-returns의 동영상과 거의 비슷한 결과값이 측정된 걸 알 수 있다.

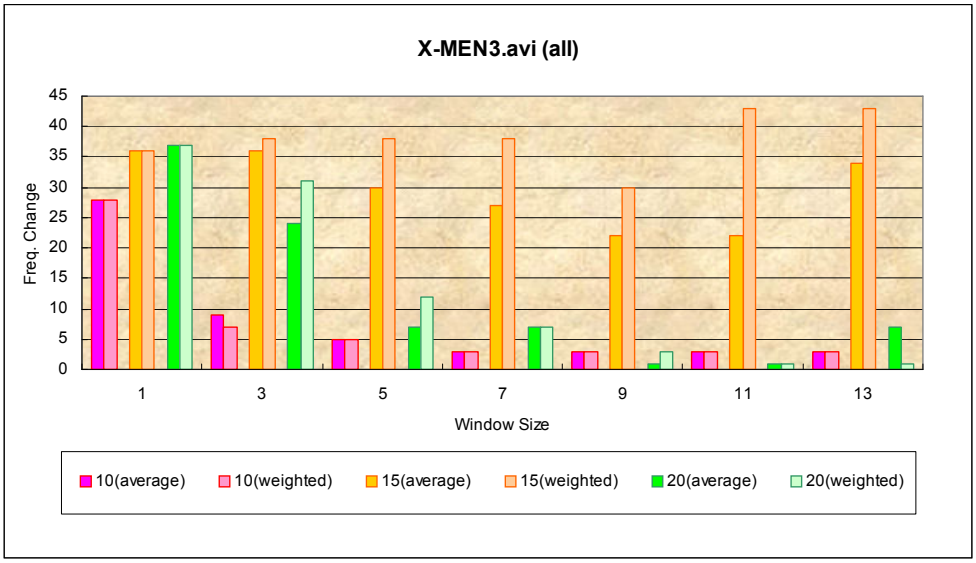
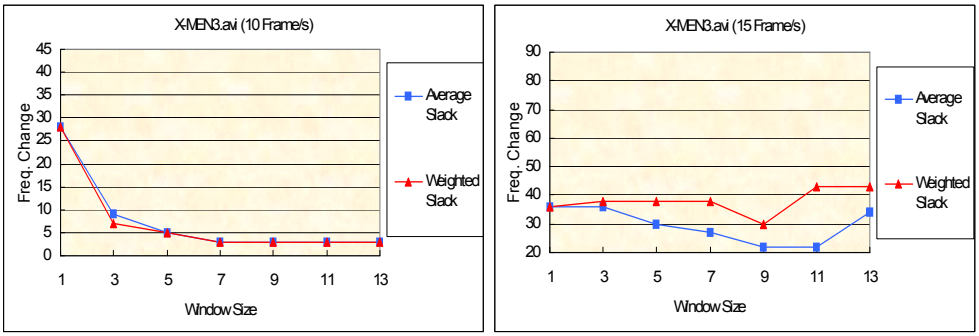


그림 13. X-MEN3 (f, v)레벨 변경 측정값



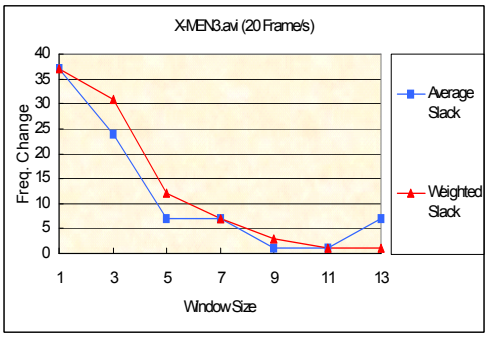


그림 14. X-MEN3 fps별 (f, v)레벨 변경 측정값

4.2.3 Deadline Miss 측정 결과

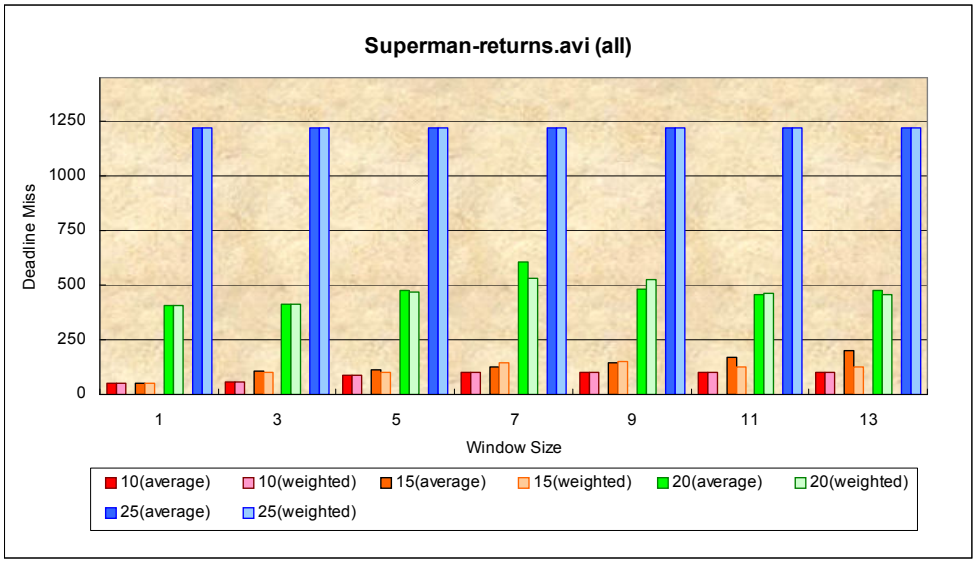


그림 15. Superman-returns Deadline Miss 측정값

이번 절은 제안한 알고리즘이 동작할 때 프레임의 주기를 얼마나 넘었는지에 대한 Deadline Miss 횟수를 측정하였다. 이때, Deadline은 프레임의 주기, 즉 fps가 되며 프레임의 실행시간이 fps를 넘을 경우 Deadline Miss가 된 것으로 간주한다. 일반적으로 동영상을 재생할 때 fps가 높을 경우 fps에 거의 근접하게 약간 넘거나 넘지 않으면서 동영상을 수행시킨다. 따라서 Deadline Miss 의 횟수가 많다고 해서 사용자의 QoS를 떨어트리는 것은 아니다. 이런 사실을 감안할 경우 먼저 10 fps를 살펴본다면 W 가 약 7일 경우를 기점으로 증가하다가 일정해짐을 볼 수 있다. 위에서 10 fps의 (f, v) 레벨은 7을 기점으로 감소하다가 일정해지는 것으로 살펴보면, Deadline을 약간 넘어서고 넘지 않으면서 적정선의 (f, v)레벨을 찾았기 때문에 나타나는 결과이다. 이것으로 다른 fps 들의 그래프 값의 결과

의 타당성을 알 수가 있다. [그림 16]은 fps별로 세부적인 측정 값을 나타내는 그림이다. 가중치를 적용하는 a_i 를 사용할 때가 평균치를 적용한 것에 비해 더 안정적인 모습을 나타내고 있다.

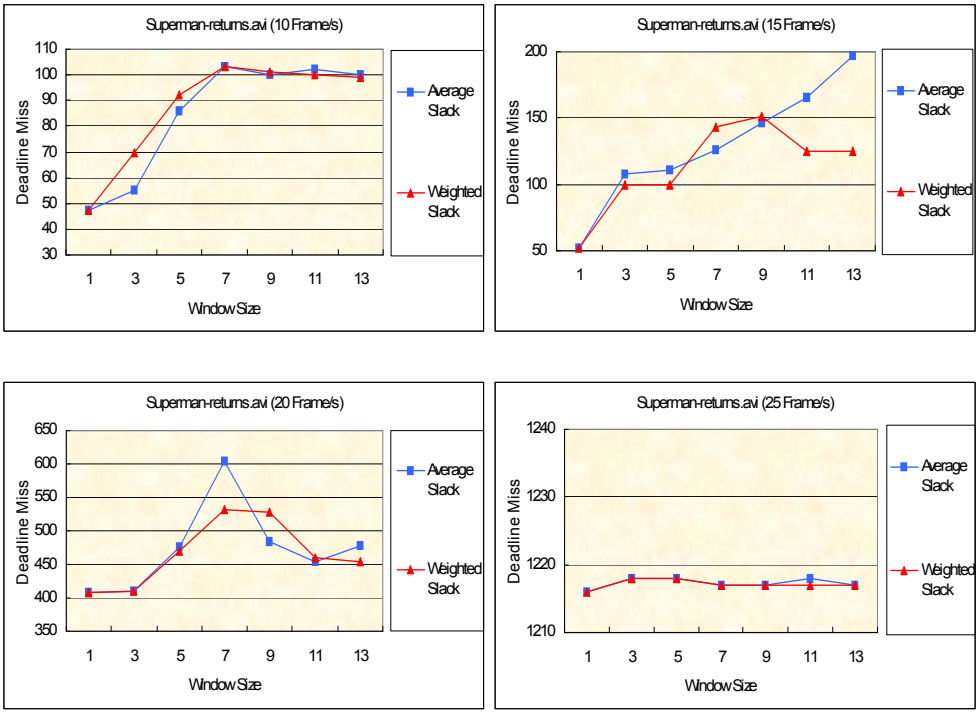


그림 16. Superman-returns fps별 Deadline Miss 측정값

다음 장에 나타낸 [그림 17]과 [그림 18] 역시 동일한 방법으로 X-MEN3 동영상의 Deadline Miss 횟수를 측정한 결과 그래프이며, 마찬가지로 가중치를 적용했을 때 더 안정적인 결과를 나타냄을 알 수 있다.

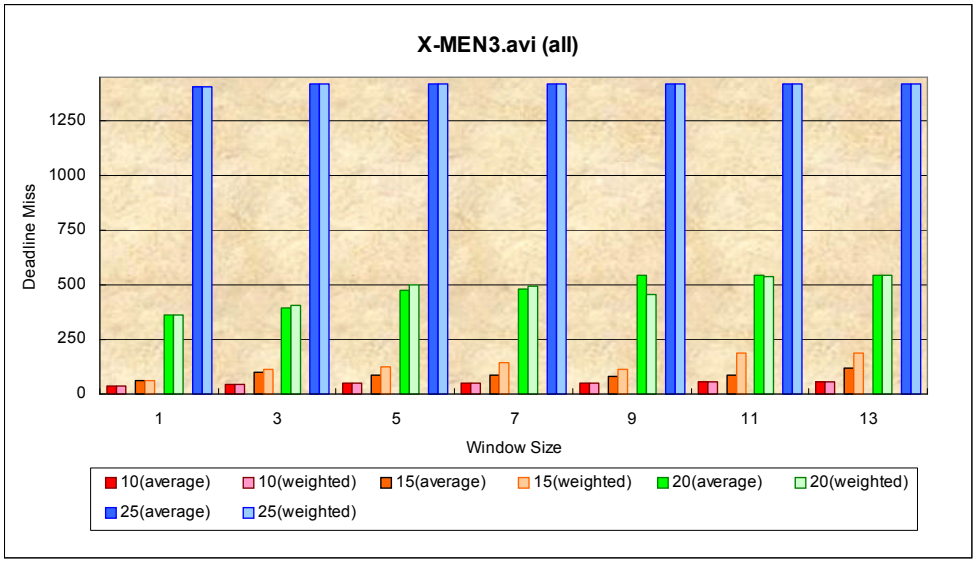


그림 17. X-MEN3 Deadline Miss 측정값

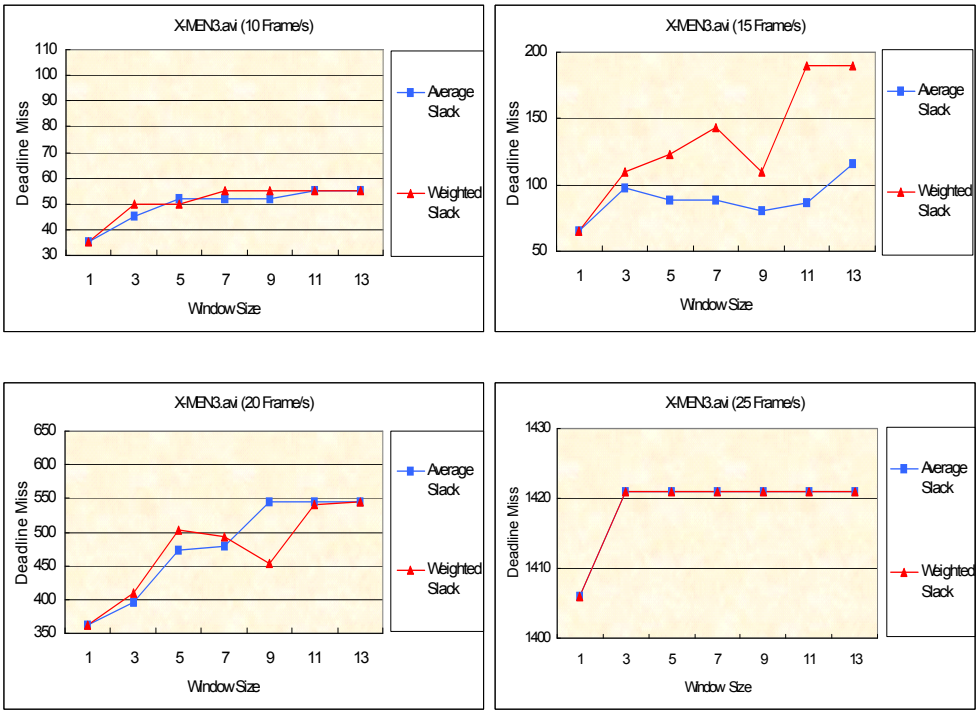


그림 18. X-MEN3 fps별 Deadline Miss 측정값

4.2.4 Dropped Frame 측정 결과

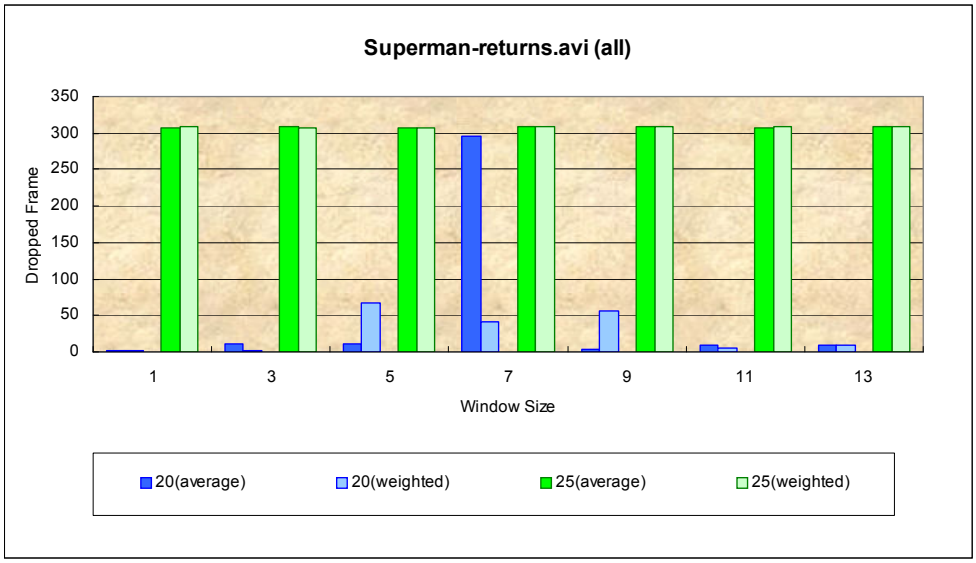


그림 19. Superman-returns Dropped Frame 측정값

이번 절은 제안한 알고리즘이 동작할 때 얼마나 많은 프레임이 Dropping 되는지에 대한 횟수를 측정하였다. 10, 15 fps는 프레임의 Drop이 없이 주기 내에 모두 처리가 되어 그래프로 나타내지 않았다. 반면에 20, 25 fps는 프레임의 주기가 10, 15 fps보다 짧은 만큼 낮은 속도로 동작하다가 비트량이 많은 구간에서는 프레임 Dropping을 하지 않으면 비디오와 오디오의 싱크가 맞지 않게 될 가능성이 크다. 측정 결과처럼 W가 7일때 가중치를 적용하는 a_i 는 적정 수준의 프레임이 Drop되었다. 평균치를 적용하는 경우 비정상적으로 많은 프레임이 Drop되었기 때문에 가중치를 적용한 방법이 더 효율적임을 알 수 있다. 25 fps의 경우 일반 MPlayer로는 정상적인 재생이 되지 않는다. 그러나 제안한 알고리즘으로 수행하였을 때는 비교적 많은 프레임이 Dropping 되긴 하였지만 정상적으로 재생이 될 뿐 아

니라 화질의 감소 또한 사용자가 느끼지 못한다는 측면에서 더 효율적이라고 말할 수 있다. 아래 [그림 20]은 측정 결과를 세부적으로 나타낸 그림이고, [그림 21]과 [그림 22] 또한 XMEN3 동영상으로 수행시켰을 때 Superman-returns와 거의 비슷한 결과를 보이고 있다.

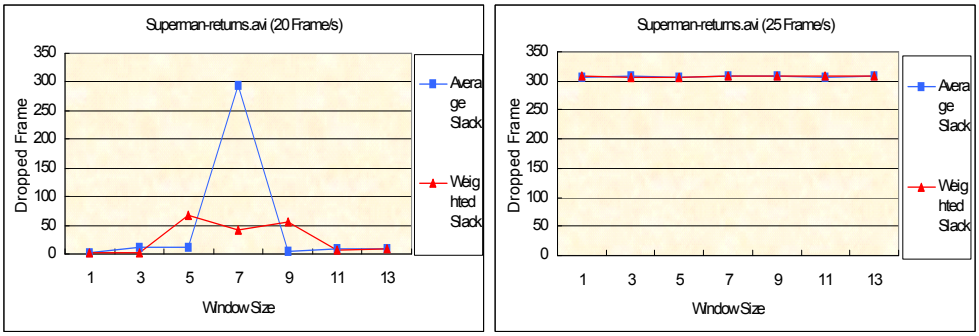


그림 20. Superman-returns 20,25 fps Dropped Frame 측정값

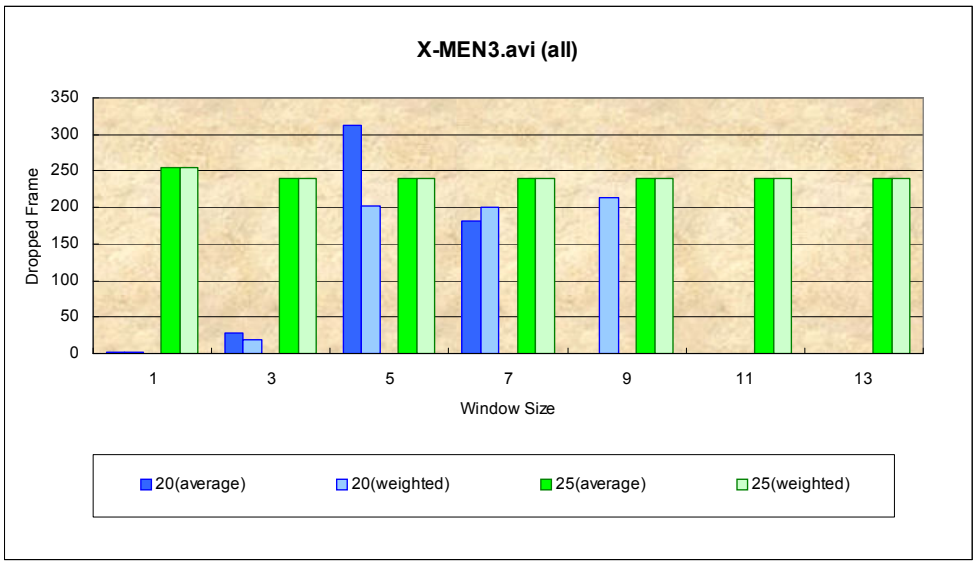


그림 21. X-MEN3 Dropped Frame 측정값

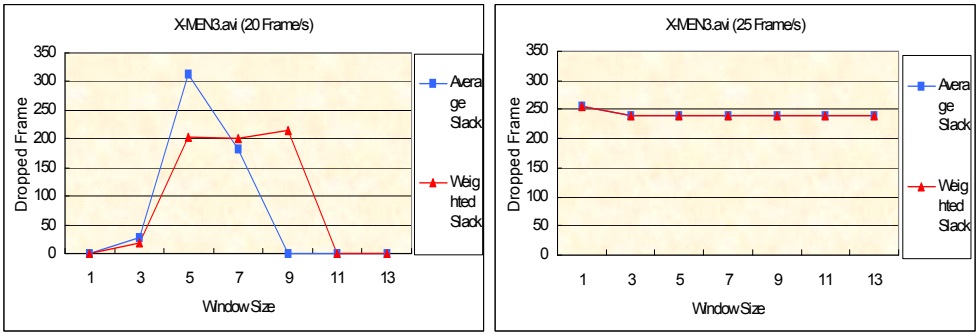


그림 22. X-MEN3 20,25 fps Dropped Frame 측정값

이번 장에서는 제안한 알고리즘을 구현하고 여러 가지 방법으로 실험한 결과에 대해 기술하였다. 측정값 들에 대한 분석에서 알 수 있듯이 W 가 약 5에서 7사이이며 가중치를 사용하는 $a_i = i \cdot \frac{2}{W(W+1)}$ 를 적용하였을 때 더 높은 성능을 발휘하였다. 또한 10 fps의 경우 약 56%, 15 fps의 경우 약 40%, 20 fps의 경우 약 25%의 전력소모 절감효과를 가져왔으며 25 fps의 경우 일반 MPlayer에서는 정상적으로 수행되지 않으나 본 알고리즘을 적용한 수정된 MPlayer는 무리 없이 재생됨을 알 수 있었다.

5. 결론 및 향후 연구과제

최근에 휴대폰, PDA와 같은 다양한 기능을 가진 휴대용 단말기 등이 널리 사용됨에 따라 배터리 기반으로 동작하는 임베디드 시스템 상에서의 전력 관리는 최근 중요한 문제로 대두되고 있다. 이에 따라 본 논문에서는 경성 실시간 시스템(Hard Real-Time System)을 위주로 하는 다소 비현실적인 기존 연구를 바탕으로 일정한 주기로 동작하면서 마감 시간에 비교적 덜 제약적인 준연성 실시간 시스템(Firm Real-Time System)에 해당하는 멀티미디어 동영상 재생기에서의 동적 전압조절 알고리즘을 제안하였다.

제안한 WB(Window-Based)-DVS 알고리즘은 프레임의 타입과 실행 시간이라는 정보를 일정한 크기의 윈도우 큐(Window Queue)로 유지하면서 프레임이 가지는 주기에 최대한 근접하게 동작, 여유 시간을 제로에 가깝도록 최적의 프로세서 레벨을 찾아 적용한다. 이 알고리즘은 간단한 모듈 형태로 구현되어 있기 때문에 일반적인 동영상 재생기에 손쉽게 추가시킬 수 있다. 실험을 통해 본 알고리즘의 성능을 측정한 결과, 25 ~ 56% 정도의 전력소모 감소 효과를 얻을 수 있었다.

본 논문의 향후과제로는 프로세서뿐만 아니라 LCD, 메모리, 통신 인터페이스 등 시스템에서 주요한 모듈들의 전력 소모를 줄이기 위해 동적 전력관리 기법과의 효과적인 연계 방안을 모색하여 운영체제 레벨에서의 전력관리 프레임워크를 구현하고자 한다.

6. 참고문헌

[1] L. Benini, A. Bogliolo, and G.D. Micheli, "A Survey of Design Techniques for System-Level Dynamic Power Management", IEEE Trans. on Very Large Scale Integration Systems, vol.10(2), pp.299-316, 2000.6.

[2] C.S. Im, S.H. Ha, and H.S. Kim, "Dynamic Voltage Scheduling with Buffers in Low-Power Multimedia Applications", ACM Trans. on Embedded Computing Systems, pp.686-705, 2004

[3] D. Shin, J. Kim, "Intra-task voltage scheduling on DVS-enabled hard real-time systems", IEEE Trans, Computer-Aided Design, vol.24, no.10, pp.1530-1549, 2005. 10.

[4] R. Xu, D. Mosse, "Minimizing Expected Energy in Real-Time Embedded Systems", EMSOFT, pp.19-22, 2005

[5] B. Brock and K. Rajamani, "Dynamic Power Management for Embedded Systems", Proceedings of the IEEE SoC Conference, 2003

[6] V. Rao, G. Singhal, A. Kumar, "Real Time Dynamic Scaling for Embedded Systems", International Conference on VLSI Design 17th, pp.650-653, 2004

[7] R. Jejurikar, R. Gupta, "Dynamic Voltage Scaling for Systemwide Energy Miminization in Real-Time Embedded System", International Symposium on Low Power Electronics and Design, pp.78-81, 2004

[8] Y.J. Kim, J.H. Kim, "Exploration of Memory-Aware Dynamic Voltage Scheduling for Soft Real-Time Applications", IEEE International Conference on RTCSA, pp.170-180, 2005

[9] T. Simunic, L. Benini, A. Acquaviva, P. Glynn, G. Micheli, "Dynamic Voltage Scaling and Power Management for Portable Systems", Proceedings of the 38th Conference on Design Automation, pp.524-529, 2001

[10] B. Lee, E. Nurvitadhi, R. Dixit, C.S. Yu and M.C. Kim, "Dynamic Voltage Scaling Techniques for Power Efficient Video Decoding", JSA: the EUROMICRO Journal, pp.633-652, 2005

[11] W. Yuan, K. Nahrstedt, "Practical Voltage Scaling for Mobile Multimedia Devices", Proceedings of the ACM international Conference on Multimedia, pp.924-931, 2004

[12] D. Bertozzi, L. Benini, B. Ricco, "Power Aware Network Interface Management for Streaming Multimedia", Proceedings of Wireless Communications and Networking Conference, Vol.2, pp.926-930, 2003

[13] L. Benini, G. Castelli, A. Macii, R. Scarsi, "Battery-Driven Dynamic Power Management of Portable Systems", International Symposium on System Synthesis, pp.25-33, 2000. 9.

[14] R. Urunuela, G. Muller, J. Lawall, "Energy Adaptation for Multimedia Information Kiosks", EMSOFT'06, pp.22-25, 2006.10.

[15] J. Yu, W. Wu, X. Chen, H. Hsieh, J. Yang, "Assertion-Based Design Exploration of DVS in Network Processor Architectures", Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, 2005

[16] 임성수, 신동윤, “리눅스 기반 이동통신 단말의 설계 및 구현”, SK Telecommunications Review, 15권4호, pp.573-584, 2005.8.

[17] 민정희, 차호정, “WiFi기반 모바일 임베디드 시스템을 위한 통합 전력 제어 기법”, 정보과학회논문지, 컴퓨터시스템, 2006.9.

[18] 연세대학교, “커널 모니터링에 기반한 저전력 운영체제 기법 개발”, 정보통신기초기술연구과제 04-기초-065

[19] 최진욱, 차호정, “이동 단말기를 위한 전력 소모 관리 기법”, SK Telecommunications Review, 15권4호, pp.626-637, 2005.8.

[20] 김웅걸, “준연성 실시간 시스템을 위한 동적 전압 조절 기법”, 서울대:공학석사학위논문, 2005.2.

[21] 이원규, 황선영, “실시간 시스템에서 효율적인 동적 전력 관리를 위한 태스크 스케줄링 알고리듬에 관한 연구”, 한국통신학회 논문지 31권4A호, pp.393-401, 2006.4.

Abstract

A Window-Based DVS Algorithm for MPEG Player

Kyung-Hwan, Park
Dept. of Computer Engineering
Graduate School of Kyung Hee University

As the functionality of portable devices are being enhanced and the performance is being greatly improved, power dissipation of battery-driven portable devices are being increased. So efficient power management for reducing their power consumption is needed. Even though many earlier works exist, they are mostly based on the non-realistic environment that consider limited system model and hard real time system like a military weapon. In this paper, we propose an window-based DVS algorithm for MPEG Player. Proposed algorithm maintains the recently frame information and execution time received from MPEG Player in window queue and it dynamically adjusts (frequency, voltage) level based on window queue information.

the experimental result shows the proposed algorithm reduces energy consumption by 56% on maximal performance.

Key words: Portable Devices, Power Management, DVS

감사의 글

연구실이 처음 만들어질 때부터 생활해온 지금까지 벌써 3년 반이 넘었습니다. 그동안 참 많은 일들이 있었지만 그 모든 기억들이 이제는 아련하고 소중한 추억으로 남을 만큼 저에게 큰 의미로 다가오고 있습니다. 결코 적지 않은 시간동안 저의 우둔한 머리를 항상 자극하고 깨우쳐 주신 조진성 교수님께 진심으로 감사의 말씀을 드리고 싶습니다. 또한 바쁘신 와중에도 논문의 부족한 부분을 정성껏 심사해주신 홍충선 교수님과 허의남 교수님께 깊은 감사의 말씀을 드리며 아울러 저희들을 위해 항상 열정적으로 가르침을 주시는 컴퓨터공학과 모든 교수님들께 감사드립니다.

이미 졸업을 했지만 연구실의 시작부터 함께 고생했던 건백, 두경, 준하, 그리고 박사과정에 진학하여 연구실의 모든 인원에 모범을 보여주는 성실한 대영, 석사과정 내내 훌륭한 룸메이트이자 석사과정 동기인 충용, 연구실에 있기에 든든한 권택, 힘든 일, 궂은 일도 내색하지 않고 열심히 하는 재호, 부족한 나를 잘 따라주던 성실한 상하, 막내이기에 고생을 많이 하지만 귀여운 구석이 있는 용규, 어딜가도 잘해낼 정현, 이제는 입사동기가 된 후배 착한 헌준, 뒤늦게 연구실에 들어왔지만 항상 열심인 윤성, 짧은 시간이었지만 항상 남을 먼저 배려했던 창현, 긴 시간동안 기쁜 일, 슬픈 일 모든 것을 함께한 연구실 인원에게 감사드립니다.

학교 동기이며 나의 소중한 친구들인 종후, 희호, 병현, 재훈, 정곤, 현기, 고향친구이자 고등학교 동기인 정희, 늦은 인연임에도 많이 가까워 질 수 있었던 착한 해광, 지면으로나마 감사의 마음을 전하고 싶습니다.

마지막으로 계시는 것만으로도 저에게 가장 큰 힘이 되고 한결같이 믿음으로 사랑해주시는 아버지와 어머니, 항상 도움만 주는 나의 하나밖에 없는 착한 동생 수진, 우리 가족에게 한없는 감사를 드리며 부족하지만 이 논문을 바칩니다.

2007년 겨울 박경환 드림