

# 무선 센서 네트워크에서 전송 효율과 에너지 소비에 대한 블록 FEC 심볼 크기 영향 분석

안 종 석<sup>†</sup> · 윤 중 혁<sup>\*\*</sup> · 이 영 수<sup>\*\*\*</sup>

## 요 약

본 논문에서는 저속의 무선 센서 네트워크(WSN: Wireless Sensor Network)에서 블록 FEC(Forward Error Correction) 알고리즘의 심볼 크기의 변화에 따른 802.11 MAC 프로토콜의 전송 효율과 전송 에너지를 해석적으로 분석한다. 블록 FEC 알고리즘은 심볼(symbol) 단위로 오류를 복원하므로, 주어진 무선 센서 채널에서 같은 FEC 체크 비트(check bit) 량을 사용하더라도 블록 FEC 알고리즘의 오류 패킷 복원률은 블록 FEC 심볼 크기에 의존적이다. 즉, 같은 양의 FEC 체크 비트를 사용하는 경우에, 연속된 군집 오류 길이는 작으면서 군집 오류가 자주 발생하는 채널에서는 작은 FEC 심볼이, 이에 반해 군집 오류의 길이는 크고 군집 오류 개수가 작은 군집적 분포를 보이는 채널에서 큰 FEC 심볼이 효율적이다. 심볼 크기의 영향을 평가하기 위해서 본 논문에서는 센서 노드 TIP50CM을 사용하는 WSN에서 수집한 패킷 트레이스를 기반으로 WSN 채널을 Gilbert 모델로 모델링하고, 심볼 크기가 다른 RS(Reed-Solomon) 코드를 생성하고 해석하기 위한 에너지를 측정하였다. 이러한 모델링된 채널과 각 RS 코드 생성과 해석 에너지를 이용하여 FEC 심볼 크기에 따른 RS FEC 코드를 채택한 802.11 MAC 프로토콜의 전송 효율과 전송 에너지를 계산하였다. 실제 측정 데이터와 해석적으로 계산한 데이터를 결합한 계산에 의하면 비슷한 FEC 체크 비트 량을 사용하더라도 FEC 심볼 크기에 따라 전송 효율은 최대 4.2%, 그리고 소요 에너지는 최대 35%의 차이가 발생한다.

키워드 : 링크계층 FEC, 무선 이동 네트워크, 무선 센서 네트워크

## Analysis of Block FEC Symbol Size's Effect On Transmission Efficiency and Energy Consumption over Wireless Sensor Networks

Jong-Suk Ahn<sup>†</sup> · Jong-Hyuk Yoon<sup>\*\*</sup> · Young-Su Lee<sup>\*\*\*</sup>

### ABSTRACT

This paper analytically evaluates the FEC(Forward Error Correction) symbol size's effect on the performance and energy consumption of 802.11 protocol with the block FEC algorithm over WSN(Wireless Sensor Network). Since the basic recovery unit of block FEC algorithms is symbols not bits, the FEC symbol size affects the packet correction rate even with the same amount of FEC check bits over a given WSN channel. Precisely, when the same amount of FEC check bits are allocated, the small-size symbols are effective over channels with frequent short bursts of propagation errors while the large ones are good at remedying the long rare bursts. To estimate the effect of the FEC symbol size, the paper at first models the WSN channel with Gilbert model based on real packet traces collected over TIP50CM sensor nodes and measures the energy consumed for encoding and decoding the RS (Reed-Solomon) code with various symbol sizes. Based on the WSN channel model and each RS code's energy expenditure, it analytically calculates the transmission efficiency and power consumption of 802.11 equipped with RS code. The computational analysis combined with real experimental data shows that the RS symbol size makes a difference of up to 4.2% in the transmission efficiency and 35% in energy consumption even with the same amount of FEC check bits.

Key Words : Link Layer FEC, Wireless Mobile Network, Wireless Sensor Network

### 1. 서 론

전원을 교체하기 어려운 무선 센서 네트워크의 사용 지속

시간을 연장하기 위해 전원 사용을 최적화하는 알고리즘에 대한 다양한 연구가 진행되었다. 특히 전송 오류가 빈번히 발생하는 센서 채널에서 전송 에너지를 절약하기 위해 전송 효율을 극대화하기 위한 안정적 전송 알고리즘들을 활발히 개발 구현하고 있다. 참고로 무작위로 센서 노드들을 배치해야 하고 송신 출력에 대한 제한 조건으로 인해서 무선 센서 채널에서는 군집적 오류가 빈번히 발생한다. 즉 이러한 높은 BER 환경에서도 전송 효율을 높이기 위해서 무선 센

\* 본 연구는 한국과학기술연구원 특장기초연구과제(과제번호: 5NK0503502)와 서울시 전략산업 혁신 클러스터 육성 지원사업(과제번호: 2005-0211-0)의 지원을 받아 수행하였습니다.

† 종신회원: 동국대학교 컴퓨터공학과 교수

\*\* 정 회 원: 동국대학교 컴퓨터공학과 석사

\*\*\* 준 회 원: 동국대학교 컴퓨터공학과 석사

논문접수: 2006년 6월 20일, 심사완료: 2006년 8월 28일

서 네트워크에서는 오류 방지(proactive) 방식과 오류 복원(retroactive) 방식들을 채택한다. 일례로 전파 오류를 최소화하기 위한 번복조 방식[1], interleaving 방식, 그리고 FEC 코드[2]들을 적용한다.

이러한 다양한 방식 중에서도 고 전파 오류 특성을 나타내는 무선 채널에서 전송 효율을 향상하고 전원을 절약하기 위해서는 FEC 알고리즘의 적용이 특히 효율적이다. 즉 무선 오류는 오랜 기간 지속적으로 발생하기 때문에 하나의 패킷에 오류가 발생하면 연속적으로 재전송 패킷에도 오류가 발생하여 FEC를 적용하지 않으면 전송 효율이 급격히 저하된다.

FEC 알고리즘은 크게 convolution code 알고리즘, block code 알고리즘, 그리고 turbo code 알고리즘으로 나누어지는데, 무선 채널에서는 군집적 오류에 강한 특성을 나타내는 선형 블록(linear block) 알고리즘의 한 종류인 RS(Reed-Solomon) 코드[3]를 사용한다. RS 코드는 심볼 단위로 오류를 정정하기 때문에 비트 단위로 오류 정정을 하는 FEC 알고리즘에 비해 군집적 오류 특성이 있는 무선 채널에서는 적은 계산 량으로 높은 효율의 오류 정정이 가능하다. 특히 저속의 프로세서를 사용하는 센서 노드와 군집 특성을 가지는 무선 채널에서는 RS 코드가 효율적이다.

이러한 블록 FEC 알고리즘의 에너지 소모는 FEC 심볼 크기에 의해 좌우되는데 심볼 크기에 따라 FEC 코드를 생성·해석하는 계산 시간과 이에 비례하여 소요 에너지가 달라진다. 즉 같은 크기의 사용자 데이터와 FEC 체크 비트량을 전송할 때도 FEC 심볼 크기가 클수록 FEC 코드를 생성하고 해석하는 계산 량이 증가하며 이에 따라 소요 에너지가 증가한다.

또한 FEC 심볼 크기와 채널의 군집 오류 분포도의 상관관계에 따라 패킷 복원 확률이 달라지므로 재전송 횟수와 전송 에너지가 변하게 된다. 참고로, RS 알고리즘에서는 두 개의 FEC 심볼을 이용하여 손상된 사용자 데이터 심볼 하나를 복원할 수 있다. 작은 FEC 심볼은 패킷 당 군집 오류 개수가 많으나 군집 오류의 길이가 작은 무선 채널에서, 큰 FEC 심볼은 패킷 당 군집 오류개수는 적으나 연속된 군집 오류의 길이가 긴 무선 채널에서 효율적이다.

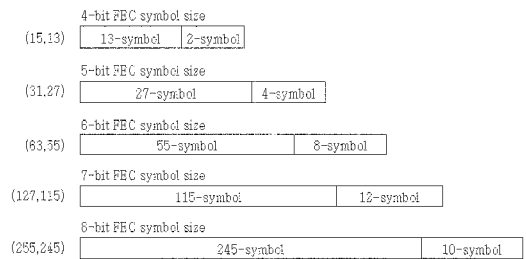
본 논문에서는 802.11 MAC을 채택하는 무선 센서 채널에서 블록 RS FEC 알고리즘을 채택할 때 FEC 심볼 크기와 군집 오류 분포도에 따른 전송 효율과 그에 따른 전송 에너지의 소모를 해석적으로 분석한다. 이러한 분석은 기존의 802.11 알고리즘 효율의 해석적 분석[4]을 기반으로 하여 802.11에 FEC 코드를 적용할 때의 전송 효율 계산식으로 확장한 것이다. 참고로 FEC 코드를 생성·해석하기 위한 에너지 소모는 센서 노드 TIP50CM[5] 상에서 측정하였으며, 무선 채널은 2 상태 마코프(2-state Markov) 방식으로 모델링하였다. 해석적 분석에 의하면 평균 BER이  $1.0 \times 10^{-3}$  인 오류 채널에서는 FEC 알고리즘을 적용한 802.11 알고리즘이 FEC 심볼크기에 따라 전송 효율이 4%이상 그리고 에너지는 35%이상 절약된다.

본 논문의 구성은 다음과 같다. 2절에서 블록 FEC 코드를 패킷 트레이스에 적용하는 방식에 대해서 기술하고 3절에서는 FEC 심볼 크기와 전송 효율의 상관관계에 대해 분석하며 4절에서는 실제 센서 네트워크의 트레이스를 기반으로 FEC 심볼 크기와 평균 BER에 따른 전송 효율에 대해 해석적으로 분석한다. 5절에서는 FEC 심볼 크기에 따른 소요에너지의 상관관계를 해석적으로 분석하고 6절에서는 결론 및 향후 연구 과제를 기술한다.

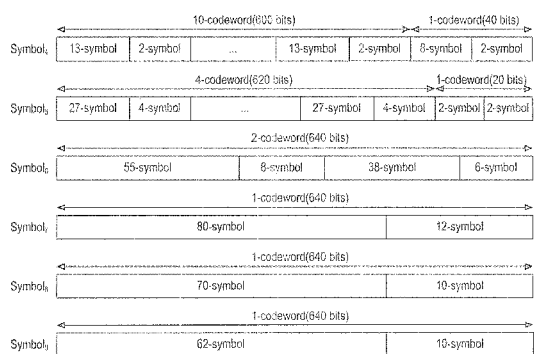
## 2. 심볼 크기에 따른 RS FEC 코드 적용 방식

본 절에서는 같은 크기의 사용자 데이터를 다양한 FEC 심볼 크기를 이용하여 패키징(packaging)하는 방식을 설명한다. 먼저, RS 코드에서 하나의 코드워드는 RS( $n, u$ )로 표현되는데,  $n$ 은 심볼 단위의 전체 코드 워드의 길이를 그리고  $u$ 는 사용자 데이터의 크기를 표현하며, FEC 체크 비트 길이  $t$ 는  $(n - u)$ 이다. RS( $n, u$ )는 오류의 위치와 심볼 당 오류 정도에 상관없이  $n$ 심볼에서 최대  $t/2$ 의 오류 심볼을 복원할 수 있다. RS 코드의 심볼 크기를  $m$ 비트라고 할 때,  $n$ 은  $2^m - 1$  심볼 또는  $(2^m - 1) * m$ 비트이다. (그림 1)은 FEC 심볼 크기가 4, 5, 6, 7, 8 비트일 때 코드 워드의 모습을 보여준다.

(그림 2)는 (그림 1)에서 보여주는 다양한 크기의 FEC 코드 워드를 적용하여 비슷한 크기의 체크 비트와 사용자 데이터를 갖는 가상의 패킷 구조의 일례를 보여준다. 즉 FEC 심볼 크기에 따라서 최대 코드워드의 크기가 결정되므로 일정 크기의 패킷 데이터를 전송하기 위해서는 하나의



(그림 1) RS 심볼 크기에 따른 코드워드 구성



(그림 2) FEC 심볼 크기에 따른 패킷 구성

패킷에 여러 개의 코드워드 또는 코드워드의 일부분만을 포함해야 한다.

(그림 2)에서는 패킷 크기를 대략 80바이트로, 그리고 FEC 체크 비트는 11바이트로 동일하도록 코드워드들을 구성하였다. (그림 2)에서 하얀색 블록은 데이터를, 회색 블록은 FEC 체크 비트를, 또한 실선 화살표 위에는 한 패킷 내에서 배치 가능한, 특정 크기의 FEC 코드워드 블록의 개수를 표기하였다. 일례로  $Symbol_4$ 에서 하나의 코드워드는 15개의 심볼, 즉, 13개의 사용자 데이터 심볼과 2개의 FEC 체크 비트 심볼로 구성된다. 11번째 마지막 코드워드는 최대 패킷 크기와 사용하는 FEC 체크 비트 크기를 다른 FEC 심볼 크기일 때와 똑같이 맞추기 위해서 8개의 데이터 심볼들과 2개의 FEC 체크 비트 심볼로 배열하였다. 그러나 6개의 심볼 크기가 서로 배수가 아니기 때문에 패킷 크기와 FEC 크기를 똑 같이 맞출 수가 없으며 이들의 크기는 최대 8비트까지 차이가 난다.

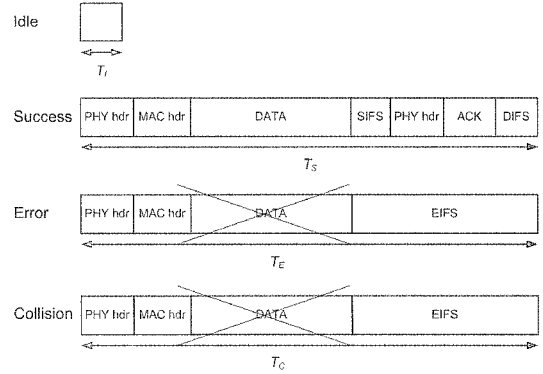
### 3. RS FEC 심볼 크기와 802.11 MAC 전송 효율 상관관계

본 절에서는 RS FEC 코드의 심볼 크기를 다양하게 적용했을 때 균일 오류 모델과 Gilbert 모델로 모델링한 센서 무선 채널에서의 802.11의 전송 효율을 수식적으로 계산한다.

식 (1)은 전송 효율 식으로, 효율 S는 예상 평균 슬롯(slot) 시간  $E[T]$ 와 이 평균 시간동안에 사용자 데이터(payload)  $L_{pld}$ 를 성공적으로 전송하는 시간  $E[L_{pld}]$ 와의 비이다.

$$S = \frac{E[L_{pld}]}{E[T]} \quad (1)$$

평균 예상 시간  $E[T]$ 는 (그림 3)과 같은 802.11에서 4가지 경우, 무 전송(idle) 상태, 성공적인 전송, 오류 상태, 그리고 충돌 상태의 지속 시간  $T_I$ ,  $T_S$ ,  $T_E$ ,  $T_C$ 의 평균 시간을 의미한다. (그림 3)에서 무 전송 상태는 하나의 경쟁 슬롯 단위 시간동안, 성공적인 전송은 하나의 프레임과 ack가 전송되는 시간동안, 오류 상태와 충돌 상태는 하나의 프레임만을 전송하는 시간동안 지속된다. 참고로 본 논문에서는 MACA 방식이 없는 802.11 성능만을 분석한다. 식 (2)에서  $T_{DIFS}$ ,  $T_{SIFS}$ 는 각각 DIFS(Distributed Inter Frame Space), SIFS(Short Inter Frame Space) 시간 간격을,  $T_{PHYhdr}$ 는 물리계층의 헤더 전송시간을,  $T_{MAChdr}$ 는 MAC 계층의 헤더 전송시간을,  $T_{DATA}$ 는 사용자 데이터의 전송시간을,  $T_{ACK}$ 는 응답 패킷 전송 시간을,  $\delta$ 는 전파 지연(propagation delay)을 의미한다. 802.11에서는 충돌 시에 EIFS(Extended Inter Frame Space)시간 간격인  $T_{EIFS}$ 만큼을 기다려야 하는데  $T_{EIFS}$ 는  $T_{SIFS} + T_{PHYhdr} + T_{ACK} + \delta + T_{DIFS}$ 와 같다. 또한 프레임 오류 시에는 수신측에서 확인응답을 하지 않으므로, 송신측은 데이터 전송 후 ack를 기다릴 타임아웃 시간  $T_{ACK-Timeout}$ 과  $T_{DIFS}$ 시간을 기다려야 한다.  $T_{ACK-Timeout}$ 은  $T_{SIFS} + T_{PHYhdr} +$



(그림 3) 802.11의 전송 시간

$T_{ACK} + \delta$ 와 같고  $T_{EIFS} = T_{ACK-Timeout} + T_{DIFS}$  이므로 프레임 오류 시에도  $T_{EIFS}$ 시간을 기다려야 한다.

$$\begin{aligned} T_I &= \sigma \\ T_S &= 2 * T_{PHYhdr} + T_{MAChdr} + T_{DATA} + T_{SIFS} + T_{ACK} + 2 * \delta + T_{DIFS} \\ T_E &= T_{PHYhdr} + T_{MAChdr} + T_{DATA} + \delta + T_{EIFS} \\ T_C &= T_{PHYhdr} + T_{MAChdr} + T_{DATA} + \delta + T_{EIFS} \end{aligned} \quad (2)$$

식 (3) - (6)은 (그림 3)의 네 가지 경우가 발생할 확률을 계산하는데,  $\tau$ 는 하나의 슬롯 시간에서 하나의 노드가 전송할 확률을,  $n$ 은 전체 노드의 수를 나타낸다. 참고로 802.11에서는 채널을  $9\mu s$ 의 값을 갖는 슬롯단위로 분할하여 사용한다.  $\tau$ 는 전송 실패 후 경쟁 윈도우(contention window) 사이즈가 2배로 되는 확률  $p_r$ 로 기술되는데, 이 둘 간의 관계식은 [4]에서 설명하고 있다. Idle 슬롯에서 한 슬롯이 전송을 시도하지 않을 확률은  $(1 - \tau)$ 이므로  $n$ 개 노드가 전송을 하지 않을 확률  $P_I$ 는 식 (3)과 같다. 성공적으로 전체 프레임을 전송할 확률은  $n$ 개 노드에서 1개만이 전송하는 확률이므로 식 (4)와 같다. 즉, 하나의 노드가 전송을 시도하여 전송 패킷에 에러가 발생하지 않아야 한다. 식 (4)에서  $P_e$ 는 패킷 오류 확률이며 식 (11)과 같이 계산된다. 오류 확률은 식 (5)와 같이 계산되며, 충돌 확률은  $P_I$ ,  $P_S$ ,  $P_E$  3가지의 확률을 1에서 뺀 식 (6)과 같다.

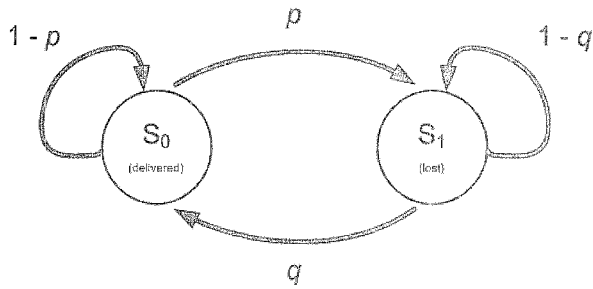
$$P_I = (1 - \tau)^n \quad (3)$$

$$P_S = n\tau(1 - \tau)^{n-1}(1 - P_e) \quad (4)$$

$$P_E = n\tau(1 - \tau)^{n-1}P_e \quad (5)$$

$$P_C = 1 - P_I - P_S - P_E \quad (6)$$

다음의 식 (7) - (10)은 802.11 프레임에 RS FEC 코드를 포함할 때의  $P_e$ 를 계산한다. 먼저 식 (7)은 무선 채널에서 데이터를 변조하여 전송할 때 하나의 전송 심볼 오류 확률  $P_t$ 와 FEC 심볼 오류 확률  $P_f$ 와의 관계를 보여준다. 참고로, 무선 채널에서 변·복조 방식을 통해서 데이터를 전송할 때 한 비트 이상의 연속적인 비트 열을 하나의 전송 심볼로 변조하여 병렬적으로 전송하며 수신자는 심볼 단위로 복조하



(그림 4) 길버트 모델

여 수신한다. 이때 전송 심볼 크기를  $h$ 비트, FEC 심볼 크기를  $m$ 비트, 그리고  $c = m / h$ 라고 가정한다. FEC 심볼 오류 확률을 계산할 때 실제 네트워크의 특성을 반영하기 위해 길버트 모델(Gilbert Model)[6]과 균일 분포(uniform distribution) 모델 2가지를 사용한다.

(그림 4)는 길버트 모델을 보여주는데, 길버트 모델에서는 전송 심볼 오류 여부를 2-state 마코프 체인으로 설명한다. 즉 임의의 전송 심볼이 수신측에 성공적으로 도착하는 경우를  $S_0$ 상태로, 손실되어 도착하는 경우를  $S_1$ 상태로 나타낸다. 이때 전송 심볼 오류는 각 상태간의 전이 확률  $p, q$ 로 표현한다. 일례로,  $n$ 번째 전송 심볼이 도착한 경우에  $(n + 1)$ 번째 전송 심볼이 오류가 나는 확률은  $S_0$ 에서  $S_1$ 으로의 전이 확률  $p$ 로 나타낸다. 길버트 모델에 따라  $c$ 개의 전송 심볼이 모두 성공적으로 도착할 확률은 최초  $S_0$  상태에서  $(c - 1)$ 개의 전송 심볼이 모두  $S_0$ 상태일 확률이며 이에 따라 FEC 심볼 오류 확률은 식 (7)과 같이 계산된다. 식 (8)은  $S_0$ 상태에 있을 확률  $P(S_0)$ 를 보여준다. 각 상태간의 전이 확률은  $p, q$ 이므로 전체 확률은  $p + q$ 이고  $S_0$ 상태에 있을 확률  $P(S_0)$ 은  $q / (p + q)$ 와 같다.

균일 분포를 사용하는 경우, 전송 심볼의 오류 확률이  $P_f$ 일 때 하나의 FEC 심볼에 오류가 발생할 확률  $P_f$ 는  $c$ 개의 전송 심볼 중에 어느 하나의 전송 심볼에 오류가 발생할 확률이다. 참고로, FEC 심볼 크기는 프로그램에 의해서, 전송 심볼 크기는 하드웨어에 의해서 결정된다. offset-DQPSK (Differential Quadrature Phase Shift Keying) 변조 방식을 사용하는 TIP50CM 센서 노드에서  $h$ 는 2비트,  $m$ 은 2비트 이상이며  $c$ 는 1이상의 실수가 된다.

$$P_f = \begin{cases} 1 - P(S_0)(1-p)^{c-1} & \text{in Gilbert Model} \\ 1 - (1 - P_f)^c & \text{in Uniform Distribution} \end{cases} \quad (7)$$

$$P(S_0) = \frac{q}{p+q} \quad (8)$$

식 (9)[7]는 M-ary PSK (Phase Shift Keying) 변조 방식을 사용할 때 전송 심볼 오류 확률  $P_t$ 를 계산하고  $P_t$ 는 전송 심볼의 크기  $h$ 에 따라 달라진다. 이때  $h = \log_2 M$ 이며,  $E_b$ 는 한 비트 전송 에너지,  $N_0$ 는 화이트 잡음 에너지이다.

식 (9)는 같은 전송 에너지를 사용하는 경우에 변조 정도를 나타내는  $M$ 이 증가하게 되면 심볼 오류 확률이 증가하는 것을 보여준다.

$$P_t = 2Q \left( \sqrt{\frac{2hE_b}{N_0} \sin\left(\frac{\pi}{M}\right)} \right) \quad (9)$$

블록 FEC 알고리즘을 이용하여 데이터를 전송할 때 데이터는 사용자 데이터와 FEC 체크 비트로 구분되는데, FEC 심볼 크기가  $m$ 비트일 때 사용자 데이터와 FEC 체크 비트를 합한 전체 데이터 크기  $N$ 비트는  $(2^m - 1) * m$ 으로 제한된다. 이때 FEC 체크 비트 크기  $T$ 비트는 가변적이며 복원하려는 최대 오류 비트 수에 의해 결정된다. 즉,  $n = N / m, t = T / m$ 일 때 이 FEC 체크 비트로 복원할 수 있는 오류 심볼 개수는  $t / 2$ 개이며 전송 패킷의 크기와 FEC 심볼 크기에 의해 보호받는 최대 데이터의 크기는 다를 수 있다. 일례로, 패킷 심볼 개수  $n$ 이  $n \leq (2^m - 1)$ 인 경우에는  $(2^m - 1) - n$ 개의 심볼을 0이라고 가정하여 FEC 코드를 생성한다. 그리고 전송할 때는  $n$ 개 심볼만을 전송하나 수신하여 복원할 때는 FEC 코드를 생성할 때와 마찬가지로 같은 위치의 데이터를 0으로 가정하여 복원한다. 반대로  $n > (2^m - 1)$ 인 경우에는  $n$ 개의 심볼로 구성된 패킷은  $(2^m - 1)$  크기의  $d (= n / (2^m - 1))$ 개의 블록들로 구성되며  $d$ 개의 코드워드 블록마다  $t / d$ 개의 FEC 심볼을 덧붙이게 된다. 이때 정정할 수 있는 최대 오류 수는  $d$ 개의 블록마다  $t / 2d$ 개 이하로 제한적이다. 물론, 마지막 블록이  $2^m - 1$ 개의 심볼보다 작은 경우에는 첫 번째 방식처럼 0을 패딩 하여 FEC 코드를 생성하고 수신자에서는 송신자와 마찬가지로 0을 패딩 하여 복원하게 된다.

식 (10)은 RS FEC 알고리즘을 사용할 때  $t$ 개의 FEC 체크 비트 심볼을 포함한  $n$ 개 심볼로 구성된 패킷을 성공적으로 수신할 확률  $P_a$ 를 나타낸다.  $n \leq (2^m - 1)$ 일 때는  $n$ 개의 전체 심볼 중에서 손상된 심볼 개수가  $t / 2$ 보다 작은 경우의 확률을 계산한다. 반대로  $n > (2^m - 1)$ 일 경우에는  $d$ 개의 코드워드 블록 모두에서 손상된 심볼 개수가  $t / 2d$ 보다 작은 확률을 계산한다. 이때 식 (10)에서 계산 편의상  $t / 2d$ 와  $n / d$ 가 정수라고 가정한다.

$$P_a = \begin{cases} \sum_{k=0}^{t/2} \binom{n}{k} P_f^k (1 - P_f)^{n-k} & \text{when } n \leq 2^m - 1 \\ \left( \sum_{k=0}^{t/2d} \binom{n/d}{k} P_f^k (1 - P_f)^{n/d-k} \right)^d & \text{when } n > 2^m - 1 \end{cases} \quad (10)$$

식 (11)은 전체  $n$ 개의 심볼을 갖으며  $t$ 개의 FEC 체크 비트 심볼을 포함한 패킷 오류 확률  $P_e$ 를 계산하는데,  $P_e$ 는 1에서 패킷을 잘 수신할 확률  $P_a$ 를 뺀 값이다.  $P_a$ 는 패킷을 잘 수신할 확률을 나타내며 식 (10)에 의해 계산된다.

$$P_e = 1 - P_a \quad (11)$$

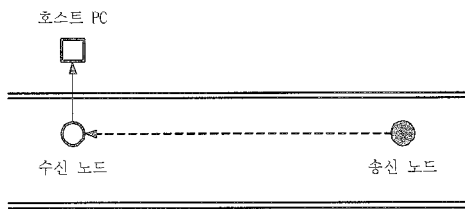
식 (12)는 패킷의 사용자 데이터의 크기가  $L_{pld}$ 일 때 성능  $S$ 를 나타내며 식 (2) - (10)를 이용하여 유도된다.

$$S = \frac{P_S I_{pld}}{P_I T_I + P_S T_S + P_E T_E + P_C T_C} \quad (12)$$

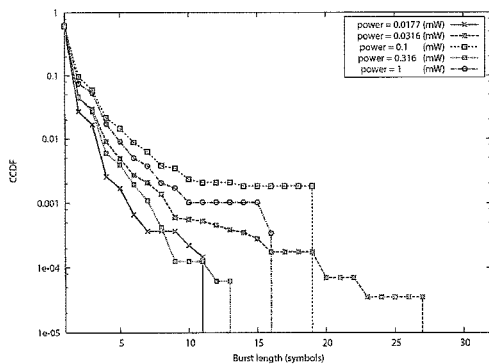
#### 4. 센서 네트워크에서 802.11 MAC 전송 효율

본 절에서는 TIP50CM 센서 네트워크에서 수집한 트레이스를 기반으로 센서 채널을 길버트 오류 모델로 모델링할 때 FEC 심볼 크기에 따른 802.11 MAC의 전송 효율을 계산한다.

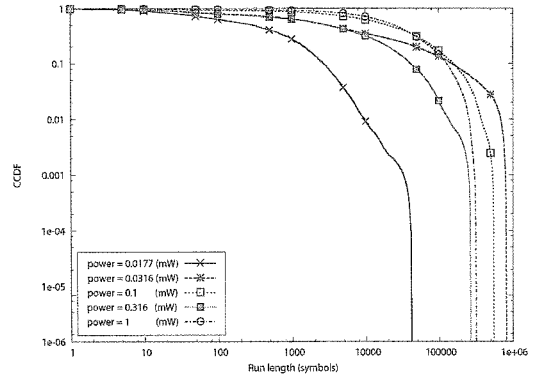
(그림 6)과 (그림 7)은 전송 신호 세기에 따른 비트 트레이스의 특성을 보여주는데, (그림 6)과 (그림 7)은 각각 채널 심볼 오류의 burst length와 무오류 비트의 run length CCDF(Complement Cumulative Density Function)를 보여준다. (그림 5)는 실험을 위한 센서 네트워크 환경을 보여주는데, 10 m 거리의 LOS(Line-Of-Sight)로 정렬된 송·수신 노드로 구성되며, 송신 노드는 계속적으로 80바이트 패킷을 160bytes/sec의 속도로 전송한다. 트레이스는 평일 오후 1시 ~ 4시까지 세 시간동안 사람들의 움직임이 빈번한 복도에서 수집되었으며, 하나의 전송 신호 세기 당 3번씩, 총 15번 수집되었고, 전송 신호 세기는  $1.779 \times 10^{-2}$ ,  $3.16 \times 10^{-2}$ ,  $1 \times 10^{-1}$ ,  $3.16 \times 10^{-1}$ , 1mW로 지정하였다. 참고로, (그림 5)에서 두 개의 이중 실선 사이의 공간은 송·수신 노드가 위치하는 복도를, 점선 화살표는 전송 노드에서 수신 노드로 패킷이 전송됨을, 실선 화살표는 수신된 패킷을 시리얼 포트를 이용하여 호스트 PC로 전송함을 나타낸다. 참고로, 복도의 폭과 높이는 각각 3m, 2.5m이다.



(그림 5) 센서 네트워크 실험 환경



(그림 6) 전송 신호 크기에 따른 burst length



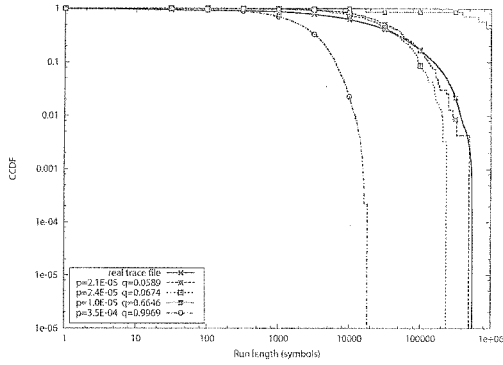
(그림 7) 전송 신호 크기에 따른 run length

(그림 6)에서 군집 오류의 최대 길이는 최대 50비트이며 평균 길이는 약 4비트이다. (그림 6)의 5개의 그래프를 비교해 보면 고정된 10m 거리에서 전송 신호의 크기는 군집 오류 비트의 길이에 큰 영향을 주지 않음을 알 수 있다. 일례로, 전송 신호 크기가 1mW인 그래프를 살펴보면 상대적으로 더 작은 전송 신호 크기를 갖는 그래프보다 더 큰 길이의 군집 오류 비트가 존재하고 신호 크기가 최소인  $1.779 \times 10^{-2}$ mW의 그래프에서는 평균적으로 가장 작은 군집 오류 비트 길이를 보인다.

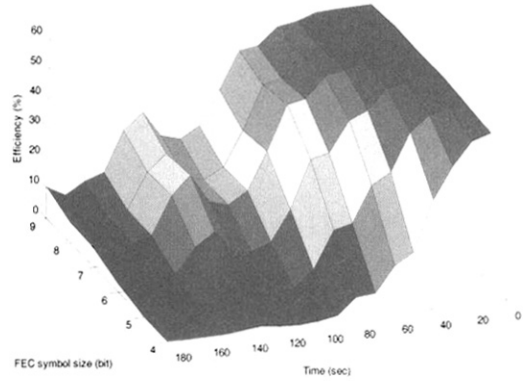
(그림 7)에서 평균 군집 무오류 비트는  $1.5 \times 10^5$ 비트이고 일정 값 이상의 전송 신호 크기에서는 신호 세기에 상관없이 비슷한 평균 군집 무오류 비트 길이를 갖는 것을 알 수 있다. 즉,  $1.77 \times 10^{-2}$ mW의 가장 작은 신호 크기를 제외한 나머지 그래프들은 비슷한 평균 군집 무오류 비트 길이를 보이고 있다. 또한, 신호 크기가  $3.16 \times 10^{-2}$ mW인 그래프의 경우에는 더 큰 신호 크기인 1mW의 그래프보다 평균 군집 무오류 비트 길이가 더 긴 것을 확인할 수 있다.

이러한 센서 채널을 길버트 모델로 모델링하기 위해서 실제 트레이스 평균 BER과 run length 분포도와 가장 유사하도록 길버트 모델의 두 개의 상태 전이 확률  $p$ 와  $q$ 값을 계산한다. 즉 실제 평균 BER과 같은  $S_0$ 에 있을 확률  $p / (p + q)$ 에서 주어진 run length CCDF와 같은 분포도를 생성하는  $(p, q)$  조합을 구한다. 두 개의 CCDF 분포도의 유사도는 LMS(Least Mean Square) 방식에 의해 측정한다. 일례로, (그림 8)은 실제 BER이  $1.17 \times 10^{-3}$ 일 때 실제 run length CCDF와 평균 BER은 같지만 군집도가 다른 네 개의 서로 다른 길버트 모델을 보여준다. (그림 8)에서  $p$ 는  $2.1E-05$ ,  $q$ 는 0.0589일 때 가장 적은 LMS 오차를 보이는 것을 알 수 있다.

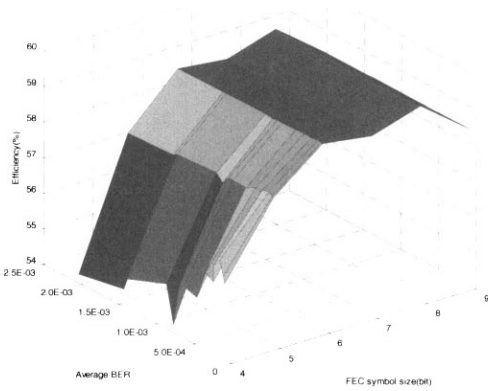
(그림 9)는 15개의 실제 트레이스와 가장 유사한 CCDF 분포도를 보이는 길버트 모델을 적용했을 때의 FEC 심볼 크기와 평균 BER에 따른 802.11의 전송 효율을 보여준다. (그림 9)에서 FEC 심볼 크기가 4 비트일 때 가장 낮은 성능을 나타내며, 8 비트일 때 가장 높은 성능을 나타내는데, 최대 약 4.2%의 효율 차이를 보인다. 또한 5비트 이상의 FEC 심볼 크기를 사용 할 때 평균 BER에 상관없이 전송 효율은 일정하다는 것을 보여준다.



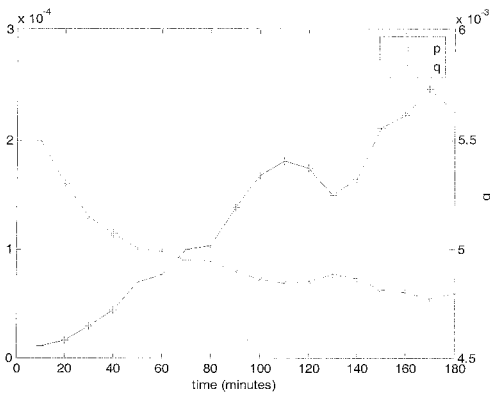
(그림 8) 상태 전이 확률  $p$ 와  $q$ 에 따른 run length



(그림 11) FEC 심볼 크기와 시간에 따른 전송 효율



(그림 9) FEC 심볼 사이즈와 평균 BER에 따른 전송 효율



(그림 10) 10분 단위의 센서 채널 상태 변화

(그림 10)은 센서 채널을 길버트 모델로 모델링할 때 임의의 하나의 트레이스 파일에서 시간에 따른 10분단위의 채널 상태 변화, 즉  $(p, q)$  값의 변화를 보여준다. (그림 10)에서 시간이 흐름에 따라  $p$ 와  $q$ 의 크기의 비는 최대 500배, 최소 20배의 변화를 한다. 참고로,  $p$ 와  $q$ 의 크기가 다를수록 균집적인 특성을, 같을수록 균일 분포를 의미한다. 즉, 순간적으로 센서 채널 오류는 균일 분포를 한다. (그림 10)은 시간이 흐름에 따라 채널 오류가 균집적 특성에서 균일 분포 상태로 변화함을 알 수 있다. 또한 균일 분포 일수록 평균

BER이 현저히 높아짐을 알 수 있는데, 일례로, (그림 10)의 50분부터 180분까지의 평균 BER은  $1.3 \times 10^{-2} \sim 4.8 \times 10^{-2}$ 로 일반적인 무선채널에 비해 평균 BER이 높음을 알 수 있다.

(그림 11)은 (그림 10)의 트레이스를 기반으로 시간에 따라 채널 오류가 균집적인 특성에서 균일 분포를 보일 때 802.11의 전송 효율 변화를 보여준다. 균집적인 특성을 보이는 30분까지의 FEC 심볼 크기별 전송 효율의 차이는 10% 내외이며 FEC 심볼 크기가 8비트 일 때 가장 높은 성능을 보인다. 균일 분포를 보이는 60분에서 80분까지는 FEC 심볼 크기가 9비트 일 때 가장 높은 성능을 보이며, 4비트 일 때 가장 작은 성능을 보인다. 이 때 전송 효율은 40%정도의 차이를 보이는 것을 알 수 있다. 또한 균일 분포에 가까워질수록 평균 BER이 높아지기 때문에 80분 이후부터는 전송 효율이 급격히 감소하며, FEC 심볼 크기별 전송효율의 차이 또한 점점 줄어든다. 가장 균일한 분포를 보이는 160분 이후는 전송효율이 10% 이하임을 알 수 있다.

### 5. RS FEC 심볼 크기와 에너지의 상관관계

본 절에서는 실제 센서 노드에서 블록 FEC 코드를 생성하고 해석하기 위한 에너지, 그리고 전송 에너지를 측정·분석한다. 식 (13)[8]은 RS 코드를 디코딩하기 위한 계산 에너지를 해석적으로 표현하는데, RS 코드 디코딩에 필요한 계산 에너지는 코드워드 길이인  $n$ , FEC 체크 비트 길이인  $t$ , 그리고 합과 곱의 계산에 필요한 에너지  $E_{add}$ 와  $E_{mult}$ 에 비례한다. 즉 선형 블록 코드를 해석하기 위한 에너지는 FEC 체크 비트 길이의 제곱과 전체 코드워드 길이의 합인  $O(t^2) + O(n)$ 에 비례한다.

$$E_{dec} = (2nt + 2t^2) \cdot (E_{add} + E_{mult}) \quad (13)$$

<표 2>는 <표 1>에 성능이 요약된 TIP50CM 센서 노드를 이용하여 (그림 1)에 표현되어 있는 FEC 코드 워드를 생성 해석하기 위한 에너지를 측정된 결과이다. TIP50CM은 Moteiv사의 Telos와 같은 성능을 발휘하는 센서 노드로 3V

〈표 1〉 TIP50의 하드웨어 구성

Processor	16bit RISC, 8MHz
Memory	256KB Program Flash
Operating System	TinyOS
Multi-channel Radio	2.4GHz
Data Rate	250Kbps
Sensor	Temperature, Humidity, Light
Power	3.0~3.3V, 240μA
Range	70m in lab

전원과 240μA 전류를 사용하는 저 전력 프로세서인 TI(Texas Instruments) 사의 MSP430f149를 채택하였다. 참고로 RS 인코딩과 디코딩 코드는 [9]을 사용하였다.

(그림 1)에서 해석에 사용되는 에너지를 측정하기 위해서는 식 (14)를 사용하였다. 즉 인코딩과 디코딩에 필요한 시간  $t_{enc}$ ,  $t_{dec}$ 와 수행할 때의 사용하는 전원  $V$ 와 전류  $I$ 를 측정하여 인코딩 그리고 디코딩에 사용하는 에너지를 측정할 수 있다. 참고로 명령어에 따른 전류와 전원의 변화는 거의 없는데, 이는 일반적인 프로세서에서 고속 실행을 위해 파이프라인 기법을 사용하기 때문이다.

$$E = V \cdot I \cdot t_{dec} \tag{14}$$

〈표 2〉에서 인코딩 에너지는 일정하나 디코딩 시간은 오류 심볼 개수에 따라 소모되는 에너지 량이 변화하게 되는데, 이는 오류 심볼 개수에 따라 수행하는 코드가 달라지기 때문이다. 일반적으로 오류 심볼의 개수가 증가하면 수행 시간과 소요 에너지가 증가한다.

(그림 12)은 (그림 2)의 패킷 구조를 사용할 때 각 FEC 심볼 크기에 따라서 세 가지 경우, 즉, 무오류 상태, 정정 오류 상태, 그리고 정정 불가 상태에서 소요되는 평균 전력을 보여준다. 하나의 패킷에 여러 개의 코드 워드가 포함되는  $Symbol_4$ ,  $Symbol_5$ ,  $Symbol_6$ 의 정정 오류 상태와 정정 불가 상태의 평균 프로세싱 시간은 식 (15)과 식 (16)에 의해서 계산하였다. 식 (15)와 (16)은 각 상태의 프로세싱 시간과 확률의 곱으로 구한다.

식 (15)는 정정 오류 상태의 평균 프로세싱 시간을 보여준다. 하나의 패킷이  $n$ 개의 코드워드를 갖고며 각 코드워드가 최대  $t$ 개의 오류 심볼을 정정할 수 있고  $k_i$  코드워드가  $i(0 \leq i \leq t)$ 개의 오류 심볼을 갖을 때, 정정 오류 상태의 경우에 전체 프로세싱 시간은  $k_0 t_0 + k_1 t_1 + \dots + k_t t_t$ 이다. 이때  $t_i$ 는  $i$ 개의 오류 심볼을 가진 코드워드의 복구 시간을 의미한다. 정정 오류 상태의 모든 경우의 수는  $(t+1)^n$ 이고 가능한 조합은  ${}^n C_{k_0} * {}^{n-k_0} C_{k_1} * \dots * {}^{n-k_0-k_1-\dots-k_{t-1}} C_{k_t}$ 와 같다. 즉,  $n$ 개의 코드워드에서  $k_0, k_1, \dots, k_t$ 을 순서대로 뽑는 조합의 수이며, 이에 따라 정정 오류 상태의 확률은  ${}^n C_{k_0} * {}^{n-k_0} C_{k_1} * \dots * {}^{n-k_0-k_1-\dots-k_{t-1}} C_{k_t} / (t+1)^n$ 이다.

〈표 2〉 RS 코드의 심볼 크기와 오류 심볼의 개수에 따른 계산 에너지 소모량

RS code	오류 심볼 개수	인코딩시간 (sec)	디코딩시간 (sec)	인코딩 에너지소모 (mW)	디코딩 에너지소모 (mW)
(15,13)	0	0.7917	1.0096	0.57	0.7269
	1	0.7917	1.3888	0.57	0.9999
	정정불가	0.7917	1.7463	0.57	1.2573
(31,27)	0	1.2636	1.5461	0.9098	1.1132
	1	1.2636	1.8404	0.9098	1.3251
	2	1.2636	2.1293	0.9098	1.5331
	정정불가	1.2636	1.6354	0.9098	1.1775
(63,55)	0	1.9551	1.7179	1.4077	1.2369
	1	1.9551	1.8921	1.4077	1.3623
	2	1.9551	2.0539	1.4077	1.4788
	3	1.9551	2.2208	1.4077	1.599
	4	1.9551	2.3945	1.4077	1.724
	정정불가	1.9551	2.0642	1.4077	1.4862
(127,115)	0	2.4438	2.0709	1.7595	1.491
	1	2.4438	2.2345	1.7595	1.6088
	2	2.4438	2.3817	1.7595	1.7148
	3	2.4438	2.5323	1.7595	1.8233
	4	2.4438	2.687	1.7595	1.9347
	5	2.4438	2.8566	1.7595	2.0567
	6	2.4438	3.0273	1.7595	2.1797
	정정불가	2.4438	2.7746	1.7595	1.9977
(255,245)	0	2.1638	2.435	1.5579	1.7532
	1	2.1638	3.4301	1.5579	2.4697
	2	2.1638	3.4220	1.5579	2.4639
	3	2.1638	3.5982	1.5579	2.5907
	4	2.1638	3.5980	1.5579	2.5906
	5	2.1638	3.5981	1.5579	2.5906
	정정불가	2.1638	3.3896	1.5579	2.4405

식 (16)은 정정 불가 상태의 평균 프로세싱 시간을 보여준다. RS 코드는 특정 코드워드에서 정정불가 시에는 즉각 디코딩을 중단한다. 따라서 정정 불가 상태의 평균 프로세싱 시간을 계산할 경우에는 정정 불가 상태의 코드워드의 위치에 따라 계산을 해야 하며 두 부분으로 구성된다. 첫째는 정정 가능한 오류 심볼을 갖는  $j(0 \leq j \leq n-1)$ 개의 코드워드를 수정하기 위한 시간이며,  $j$ 개의 정정 가능한 코드워드의 위한 평균 정정 시간은  $j$ 개의 모든 코드워드의 정정시간의 합을  $n$ 으로 나눔으로써 계산된다. 둘째는 정정 불가능한  $j+1$ 번째 코드워드를 프로세싱 하는데 필요한 시간이며  $t_{t+1}$ 과 같다.

$$t_{correctable} = \frac{\sum_{k_t=0}^{n-(k_0+k_1+\dots+k_t)} \dots \sum_{k_2=0}^{n-(k_0+k_1)} \sum_{k_1=0}^{n-k_0} \sum_{k_0=0}^{n-1} {}^n C_{k_0} * {}^{n-k_0} C_{k_1} * \dots * {}^{n-k_0-k_1-\dots-k_{t-1}} C_{k_t}}{(t+1)^n} (k_0 t_0 + k_1 t_1 + \dots + k_t t_t)$$

where  $k_0 + k_1 + \dots + k_t = n$

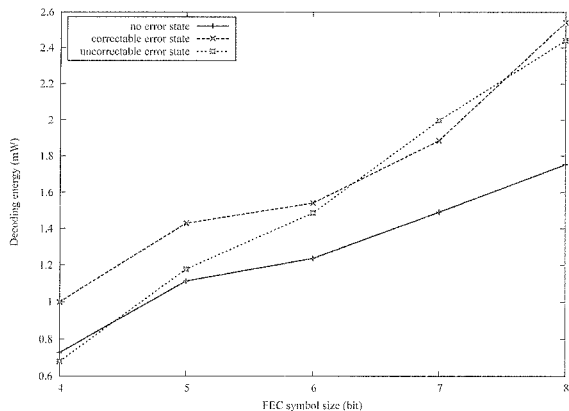
$$t_{uncorrectable} = \frac{1}{n} \sum_{j=0}^{n-1} \left( \sum_{k_1=0}^{j-(k_0+k_1+\dots+k_t)} \dots \sum_{k_{t-1}=0}^{j-(k_0+k_1)} \sum_{k_t=0}^{j-k_0} \sum_{k_0=0}^{j-1} \frac{C_{k_0}^* C_{j-k_0}^* \dots C_{j-k_0-k_1-\dots-k_{t-1}}^* C_{k_t}^*}{(t+1)^j} (k_0 t_0 + k_1 t_1 + \dots + k_t t_t) \right) + t_{t+1}$$

where  $k_0 + k_1 + \dots + k_t = j$

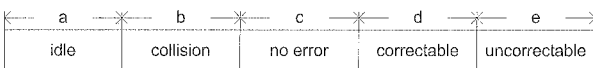
(16)

(그림 12)는 FEC 심볼 크기를 4, 5, 6, 7, 8비트를 사용하며 소요되는 에너지의 단위는 mJ이다. (그림 12)에서 무오류, 정정오류, 정정불가 상태의 순서로 소요되는 에너지가 감소하며 정정불가 상태일 때에 비해 무오류 상태일 때 대략 두 배 정도의 전력을 절약할 수 있다. 또한 각 상태별로 4비트일 때 가장 적은 전력을, 8비트일 때 가장 많은 전력을 소요한다. RS 코드 심볼 크기에 따라, 무오류, 정정오류, 정정불가 상태일 때 각각 최대 25, 28, 33%의 전력을 절약할 수 있다.

(그림 13)은 단위 시간당 각 상태별 소요 시간 비율을 나타낸다. 패킷을 전송하는데 소요되는 전체 시간, 즉 1000초 동안 다섯 가지 상태의 시간 비율을 각각 의미하며 (그림 13)에서는 각각 a, b, c, d, e로 나타내었다. 다섯 가지 상태는 idle, collision, no error, correctable, uncorrectable 상태로 구성되며, 이러한 상태는 크게 두 가지, 즉, 패킷이 도착하지 않는 상태와 패킷이 도착한 상태로 분류할 수 있다. 패킷이 도착하지 않는 상태에는 idle과 collision이 포함되고 패킷이 도착한 상태에는 no error, correctable, uncorrectable이 포함된다. 구체적으로, idle은 패킷을 전송하지 않는 상태의 시간비율을, collision은 전송한 패킷이 충돌되어 수신되지 못한 상태의 시간비율을 나타낸다. 또한 no error는 패킷이 도착하였고 에러가 없는 상태의 시간비율을, correctable



(그림 12) RS 코드 심볼 크기와 오류 심볼 개수에 따른 계산 에너지 소모량



(그림 13) 단위 시간당 각 상태별 소요 시간 비율

은 패킷이 도착하였고 에러가 발생하였으나, FEC 코드로 정정가능한 상태의 시간비율을, uncorrectable은 패킷이 도착하였고 에러가 발생하였으나 FEC 코드로 정정불가능한 상태의 시간비율을 나타낸다.

전체 소요 에너지는 다음과 같은 과정으로 계산한다. 첫째, 단위 시간당 각 상태의 시간 비율을 계산한다. 둘째, 단위 시간 당 전체 패킷 개수를 각 상태의 시간 비율로 나누어 각 상태별 패킷 전송 개수를 도출한다. 셋째, 각 상태별 평균 소요 에너지와 각 상태별 패킷 전송 개수를 곱하여 단위 시간당 각 상태별 소요 에너지를 계산한 후에 다시 초당 각 상태별 소요 에너지를 계산한다. 넷째, BER과 심볼 크기에 따라 위의 과정을 반복적으로 각 상태별 소요 에너지를 계산한 후에 합하여 초당 전체 소요 에너지를 계산하도록 한다.

BER과 FEC 심볼 크기에 따른 평균 소요 에너지를 계산하기 위해 전송 에너지, 수신 에너지, idle 에너지, 인코딩 에너지, 디코딩 에너지 값을 각 상태에 적용하여 계산한다. 참고로, TIP50CM 데이터 스위트의 전송 에너지, 수신 에너지, idle 에너지 값을 사용하였으며 패킷 당 소요되는 각각의 에너지 값을 다시 계산하여 적용한다. 또한 collision, no error, correctable, uncorrectable 상태에서 소요 되는 계산 에너지 값을 위해 <표 2>의 평균 인코딩 에너지와 평균 디코딩 에너지를 적용한다.

식 (17) - (21)은 각 상태별 소요 에너지 계산식을 나타낸다. 패킷이 목적지에 도착하지 않는 idle과 collision 상태의 계산식을 보면, idle 상태는 idle 에너지만을, collision 상태는 인코딩 에너지와 전송 에너지를 각각 소모한다. 반면에, 패킷이 목적지에 도착하는 no error와 correctable, 그리고 uncorrectable 상태의 계산식은 좀 더 많은 에너지를 소모한다. 이 세 가지 상태 모두 동일한 인코딩, 전송, 수신 에너지를 소모하고 디코딩 과정에서 에러가 없는 패킷의 디코딩 에너지, 복구 가능한 패킷의 디코딩 에너지, 복구 불가능한 에러 패킷의 디코딩 에너지를 각각 소모한다.

$$Idle = P_{idle} \tag{17}$$

$$Collision = P_{enc} + P_{trans} \tag{18}$$

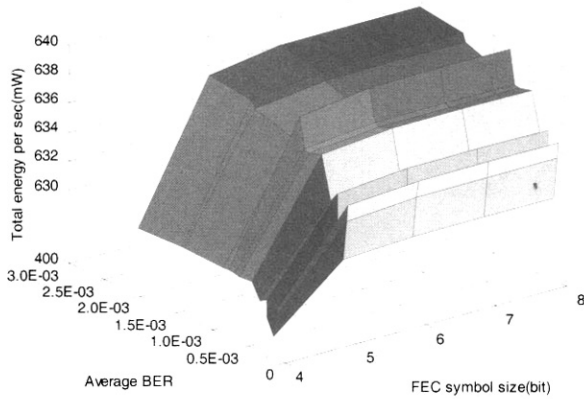
$$No\ error = P_{enc} + P_{trans} + P_{recv} + P_{decnoerr} \tag{19}$$

$$Correctable = P_{enc} + P_{trans} + P_{recv} + P_{deccorr} \tag{20}$$

$$Uncorrectable = P_{enc} + P_{trans} + P_{recv} + P_{decuncorr} \tag{21}$$

실험을 위해, 패킷 전송의 단위 시간을 1000초로 가정하고, 식 (2)의  $T_s$ , 즉 성공적인 전송의 평균 시간 값을 이용하여 285,735개의 전체 패킷 개수를 계산하였다. 또한, 각 상태별 시간 비율을 패킷 개수로 변환한 후에 식 (17) - (21)을 이용하여 초당 소요 에너지를 계산하였으며 BER과 FEC 심볼 크기에 따른 소요 에너지 계산을 반복적으로 수행하였다.





(그림 14) BER과 FEC 심볼 크기에 따른 소요 에너지 비교

(그림 14)는 BER과 FEC 심볼 크기에 따른 소요 에너지를 나타내는 그래프이다. 실험 결과, FEC 심볼 크기가 4비트이고 BER이  $1.66 \times 10^{-4}$ 일 때 최소값인 약 418mW의 에너지를, 그리고, FEC 심볼 크기가 7비트이고 BER이  $7.72 \times 10^{-4}$ 일 때 최대값인 약 639mW의 에너지를 소모한다. BER과 FEC 심볼 크기에 따라 최대 35%의 에너지를 절약할 수 있으며 평균적으로 FEC 심볼 크기가 4비트일 때 가장 작은 에너지 소모를 보였고, 반면에 FEC 심볼 크기가 7비트일 때 가장 높은 에너지 소모를 보였다. 이것은 FEC 심볼 크기가 4비트일 때의 correctable 상태의 평균 소모 에너지가 다른 심볼 크기들의 평균 소모 에너지보다 약 37% 정도 작기 때문이며, 또한 전체적으로 다른 상태들에 비해 correctable 상태일 때의 에너지 소모 비율이 가장 크기 때문에 이러한 차이가 발생한다.

### 6. 결론 및 향후 연구 과제

본 논문에서는 저속의 무선 센서 네트워크에서 사용되는 RS 블록 FEC 알고리즘에서 FEC 심볼 크기에 따른 802.11 MAC의 전송 효율과 소요 에너지의 상관관계를 분석하였다. small-scale fading과 slow fading의 특성을 갖는 저속의 센서 채널에서 FEC 심볼 크기에 따른 전송 효율과 소요 에너지를 해석적으로 분석하였으며, 측정된 채널 오류 상태를 2 상태 마코프 체인으로 모델링하여, FEC 심볼 크기에 따라 실제 TIP50CM 센서 노드의 전송 효율 및 소모되는 전체 에너지를 계산하였다.

실험에 의하면, 무선 센서 채널 오류 분포도에 따라 비슷한 FEC 체크 비트 량을 사용하더라도 FEC 심볼 크기가 8비트일 때 가장 높은 전송 효율을, 4비트일 때 가장 낮은 전송 효율을 보였으며, 최대 4.2%의 차이를 보였다. 그리고 FEC 심볼 크기가 4비트일 때 가장 작은 에너지 소모를 7비트일 때 가장 높은 에너지 소모를 보였으며, 최대 35%의 소요 에너지를 절약할 수 있었다. 또한 무선 채널이 순간적으로 균일 분포를 하는 특성을 확인하였고, 이러한 균일 분포

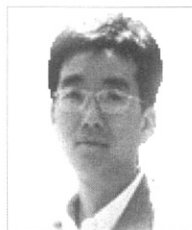
시에 FEC 심볼 크기에 따른 전송 효율은 최대 40%의 차이를 보이는 것을 알 수 있었다.

향후에는 급격히 변화하는 무선 센서 채널의 오류 특성에 따라 FEC 심볼 크기를 동적으로 조정하여 전송 효율을 향상시키는 알고리즘을 제안하고 트레이스 기반 시뮬레이션을 통하여 전송 효율과 소요 에너지를 분석할 것이다.

### 참고 문헌

- [1] G. Holland, N. Vaidya, and P. Bahl, "A Rate-Adaptive MAC Protocol for Multi-Hop Wireless Networks", ACM SigMobile, pp.236-250, July, 2001.
- [2] W. Peterson and E. Weldon, Jr., "Error-Correcting Codes", 2nd Edition, The Massachusetts Institute of Technology.
- [3] I. S. Reed and G. Solomon, "Polynomial codes over Certain Fields," J. Soc. Ind. Appl. Math., 8: 300-304, June, 1960.
- [4] Haitao Wu, Yong Peng, Keping Long, shiduan Cheng, Jian Ma, "Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement," INFOCOM, pp.599-607, Vol.2, 2002.
- [5] <http://www.maxfor.co.kr/maxfor/gnu3/files/TIP50C.alz>.
- [6] E. N. Gilbert. Capacity of a burst-noise channel. Bell Syst. Tech. J., 39:1253 - 1265, Sept., 1960.
- [7] T. S. Rappaport, "Wireless communications principles and practice", Prentice Hall, Upper Saddle River, NJ, 2002.
- [8] [http://www.chipcon.com/files/CC2420\\_Data\\_Sheet\\_1\\_3.pdf](http://www.chipcon.com/files/CC2420_Data_Sheet_1_3.pdf).
- [9] S. Rockliff, "Reed-Solomon (RS) codes" program, 1989, <http://www.eccpage.com>.

### 안 종 석



e-mail : jahn@dgu.edu

1983년 서울대학교 전자공학과(학사)

1985년 KAIST 전기 및 전자공학과 (석사)

1985년~1989년 삼성전자 주임연구원

1995년 USC(University of Southern California) 컴퓨터공학과(박사)

1995년~1996년 삼성전자 선임연구원

2001년~2002년 USC/ISI 교환연구원

1996년~현재 동국대학교 컴퓨터공학과 부교수

관심분야: 네트워크 시뮬레이션, 무선 통신, 라우팅 알고리즘, 센서 네트워크



**윤 종 혁**

e-mail : ronaldo@dgu.edu

2006년 동국대학교 컴퓨터공학과(학사)

2006년~현재 동국대학교 컴퓨터공학과  
재학(석사)

관심분야: 무선 MAC, 센서 네트워크,  
FEC 알고리즘



**이 영 수**

e-mail : diwkd@dgu.edu

2005년 동국대학교 정보통신공학과(학사)

2006년~현재 동국대학교 컴퓨터공학과  
재학(석사)

관심분야: 네트워크 시뮬레이션, 센서  
네트워크, 무선 MAC