PDF417 이차원 바코드의 디코딩 알고리즘 구현

정 정구°, 한 희일
한국외국어대학교 공과대학 정보통신 공학과

Implementation of PDF417 Decoding Algorithm

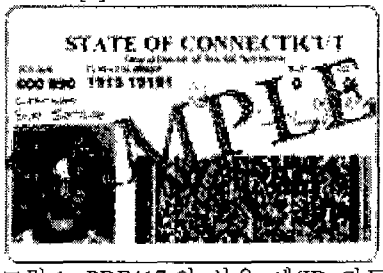
Joung-Koo Joung°, Hee-il Hahn
Department of Information and Communications Engineering, Hankuk University of Foreign Studies
jgchong@hanmail.net

요 약

물류 정보를 표현하는데 있어서 종래의 1 차원 바코드의 데이터 키가 가지는 기능만으로는 해결될 수 없는 상황이 발생 함에 따라, 1980 년대 중반에 이르러 많은 데이터를 독립적으로 표현할 수 있는 2 차원 바코드가 등장하게 되어 그 사용이 점점 확산되어가고 있다. 본 논문에서는 가장 널리 사용되고 있는 PDF417 바코드 심볼로지가 어떻게 구성되는지 알아본다. 또한, 최대 1700 개의 데이터를 저장할 수 있는 이 심볼로지로 인코딩된 정보의 코드워드로부터 원래의 데이터를 디코딩하는 방법에 대하여 알아본다. 데이터를 효율적으로 인코딩하기위해 정보가 숫자인지, ASCII 코드인지, 혹은 바이트 정보인지에 따라 다른 인코딩 방식을 사용하게 되는데, 그에 따른 디코딩 알고리즘을 제안한다.

1. 서론

PDF 417 은 1989 년 미국 Symbol Technologies 사에 의해 개발된 가변적인 심볼 길이와 가변적인 심볼 높이를 가진 다층형 이차원 심볼로지로서 현재 AIM 에 의해 표준화 되었다. PDF417 의 한 심볼은 데이터의 컴팩션(compaction) 모드에 따라 최대 ASCII 1850 문자(character)나 1108 바이트, 또는 2710 수치(digit)를 표현할 수 있다. PDF417 은 많은 데이터를 포함할 수 있고 데이터 오류의 검출 및 수정기능이 있으므로 휴대형 데이터 파일로서 적합하며, 종래의 선형 레이저 스캐너, 라스터 레이저 스캐너, 선형 CCD 스캐너, 2D CCD 스캐너로 판독이 가능하다. 하나의 심볼 문자는 4 개의 바와 4 개의 스페이스의 조합으로 구성되는데 그 길이가 17 모듈(17X)이기 때문에 PDF417 이라는 이름으로 부르게 되었다. PDF417 심볼로지는 다양한 스캐너로 판독이 가능하고 개방형 체계(open system)이므로 어느 사용자라도 용이하고 편리하게 필요한 응용 분야에 적용할 수 있는 장점이 있다. <그림 1>은 한 응용의 예로서 ID 카드에서의 PDF417 의 사용을 보여준다 [5].



<그림 1> PDF417 의 사용 예(ID 카드)

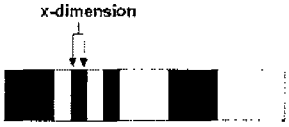
PDF417 은 데이터를 심볼로 표현하는데 있어 2 단계의 절차를 필요로 한다. 첫번째 단계는 표현하려고 하는 데이터를 코드워드로 변환시키는 것이고, 두 번째 단계는 코드워드를 바와 스페이스의 패턴으로 변화시키는 것이다. 사용자는 데이터의 중요도에 따라 오류수정 레벨(error correction level)을 선택하여 데이터의 오류를 검출 및 수정할 수 있도록 한다. PDF417 에는 3 종류의 컴팩션 모드가 있어서 데이터의 종류에 따라 가장 효율적인 방식으로 인코딩할 수 있도록 해준다.

본 논문에서는 바와 스페이스의 이미지 패턴으로부터 인코딩된 코드워드를 구했다고 가정하고 데이터 영역의 코드워드로부터 원래의 데이터로 복원하는 부분을 다루기로 한다.

2. PDF417 바코드의 구조

2.1 심볼 구조

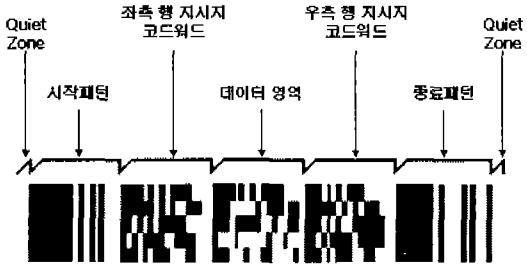
<그림 2>와 같이 x 디멘션(dimension)은 최소의 바의 너비로서 모듈이라고도 부른다.



<그림 2> x 디멘션

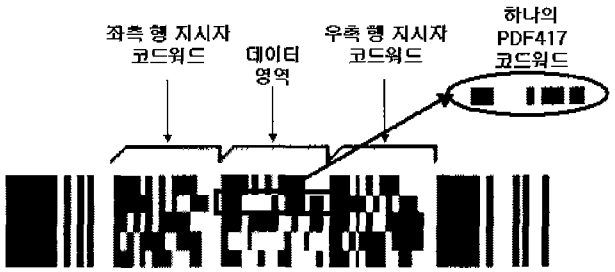
<그림 3>과 같이 PDF417 심볼은 여러 개의 열을 수직적으로 누적하여 배열하며 열의 좌우측에는 빈 여백(quiet zone)이 존재한다. 시작패턴과 종료패

턴은 리더기가 바코드의 방향과 위치를 인식할 수 있도록 해준다. 시작 패턴은 바(bar)와 스페이스(space) 순서로 81111113 이고 종료 패턴은 711311121 이다.



<그림 3> PDF417 심볼 구조

<그림 4>에서 보듯이 좌측, 우측 행 지시자와 데이터 영역을 구성하는 각 심볼 문자는 4 개의 바와 4 개의 스페이스의 조합으로 구성되며, 17 모듈의 너비를 갖는다.



<그림 4> PDF417 심볼 문자(Codeword)

각 심볼 문자는 0-928 까지의 값을 가지는데 이 심볼 문자의 값을 코드워드(codeword)라고 한다. 좌측, 우측 행지시자의 코드워드는 데이터를 저장하지 않으며, 단지 행번호, 행의 수, 데이터 영역의 열의 수, 오류수정 레벨의 정보를 가진다. 데이터 영역의 열의 수는 1-30 까지 가능하며, 행의 수는 3-90 까지 가능하다. 데이터 영역의 첫번째(좌측 상단)의 코드워드는 심볼 길이 지시자(symbol length descriptor)로서 데이터 코드워드의 총 개수를 나타내는데 그 값은 자신의 코드워드를 포함하나, 오류검출용 코드워드는 제외한다. 한 심볼 당 코드워드의 개수는 928 을 초과할 수 없다. 좌측에서 우측으로 위에서 아래로 심볼 길이 지시자, 데이터 코드워드, 패드(pad) 코드워드, 오류 검출용 코드워드 순서로 배열한다. PDF417 은 독특한 심볼 시작, 종료 패턴을 가지고 있다. 심볼의 가독 문자(human readable character)는 심볼의 상하, 좌우 어는 부분에 나타나도 괜찮으며 좌우 여백을 방해하지 않으면 된다 [1].

2.2 컴팩션 모드

컴팩션(compaction) 모드는 많은 양의 데이터를 최소의 코드워드로 만들기 위해 고안된 것으로, 텍스트 컴팩션(TC) 모드, 바이트 컴팩션(BC) 모드, 뉴메릭 컴팩션(NC) 모드가 있다. 각 모드간에는 모드랫치(latch) 와 모드쉬프트(shift) 지시자를 사용하여 이동할 수 있다.

2.2.1 코드워드 집합

PDF417 은 0-928 사이의 값을 가지는 929 개의 코드워드값을 정의한다. 0-899 까지는 데이터를 나타내는 코드워드이고, 나머지는 제어를 위한 코드워드이다.

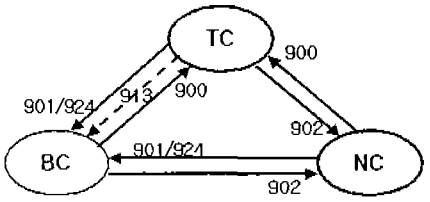
2.2.2 모드 변환

모드 변환을 위해 코드워드 900, 901, 902, 913, 924 가 정의되어 있다. 모드 변환에는 모드랫치와 모드쉬프트가 있는데, 모드쉬프트는 순간적인 모드 변환으로서 모드변환코드워드 다음의 한 코드워드에만 모드변환이 적용되고 원래의 모드로 돌아오는 것이고, 모드랫치는 영구적으로 모드를 변환한다. <표 1>은 각 모드에 할당된 코드워드를 보여준다.

<표 1> 모드 정의

모드	모드랫치	모드쉬프트
텍스트 컴팩션 (TC)	900	
바이트 컴팩션 (BC)	901/ 924*	913
뉴메릭 컴팩션 (NC)	902	

모드변환에서 모드랫치는 어떤 모드에서라도 가능하지만 모드쉬프트는 텍스트 컴팩션에서만 가능하다. <그림 5>는 가능한 모드변환을 나타낸다. 선(—)으로된 화살표는 모드랫치를 나타내고, 점선(---)으로된 화살표는 모드쉬프트를 나타낸다. 바이트 컴팩션의 모드랫치에서 코드워드 924 는 바이트의 수가 6 의 배수로 나누어질 때 사용되며, 코드워드 901 은 그렇지 않을 때 사용된다 [1].



<그림 5> 가능한 모드변환

이런 모드변환은 가능한 적은 수의 코드워드로 많은 양의 데이터를 표현하기위해 고안된 것이다.

2.2.3 텍스트 컴팩션 모드

텍스트 컴팩션(TC) 모드는 ASCII 문자들을 최소한의 코드워드로 표현하며, Alpha, Lower Case, Mixed, Punctuatioin 의 4 개의 부 모드로 구성된다. <표 2>에 정의된 것과 같이, 각 부 모드마다 0 에서 29 사이의 값이 할당되어 있으며, 다음의 식에 의해 코

드워드 하나에 두개의 값이 저장된다.

코드워드 값 = 30 * H + L

여기에서 H 와 L 은 각각 <표 2>에 정의된 값을 나타낸다.

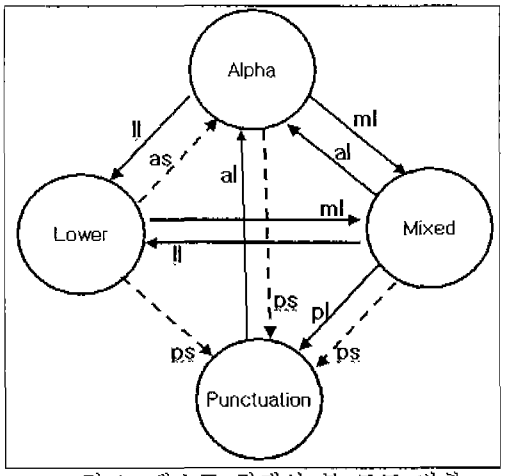
<표 2> 텍스트 컴팩션 부 모드에 사상된 값의 정의와 GLI 0 에서의 아스키 해석

value	Alpha		Lower		Mixed		Punctuation	
	ASCII Value	GLI0 Char	ASCII Value	GLI0 Char	ASCII Value	GLI0 Char	ASCII Value	GLI0 Char
0	65	A	97	a	48	0	59	,
1	66	B	98	b	49	1	60	<
2	67	C	99	c	50	2	62	>
3	68	D	100	d	51	3	64	@
4	69	E	101	e	52	4	91	[
5	70	F	102	f	53	5	92	₩
---------- 중략 ----------								
23	88	X	120	x	61	=	40	(
24	89	Y	121	y	94	^	41	)
25	90	Z	122	z		pl	63	?
26	32	SP	32	SP	32	SP	123	{
27		ll		as		ll	125	}
28		ml		ml		al	39	'
29		ps		ps		ps		al

위의 <표 2>의 문자 중에는 모드변환을 위해 사용되는 것들이 있는데 의미는 다음과 같다.

ll = Lower Case 부 모드로 랫치(latch)한다.
ps = Punctuation 부 모드로 쉬프트(shift)한다.
ml = Mixed 부 모드로 랫치한다.
as = Alpha 부 모드로 쉬프트한다.
al = Alpha 부 모드로 랫치한다.
pl = Latch to Punctuation Sub-Mode

아래의 <그림 6>는 TC 모드의 부 모드간의 가능한 변환을 나타낸다.



<그림 6> 텍스트 컴팩션 부 모드 변환

텍스트 컴팩션 모드에서는 코드워드당 거의 2 개의 ASCII 문자를 표현할 수 있다.

2.2.4 바이트 컴팩션 모드

바이트 컴팩션(BC) 모드는 다국어를 표현하거나, ASCII 에 없는 문자를 저장하는데 사용될 수 있다. 또한 그림이나 특수한 파일을 바이트 자체로 저장할 수 있다. 각 바이트의 값은 GLI 와 매핑되어 값을 표현한다. BC 모드에서는 아래의 식과 같이 900 진법을 256 진법으로 변환함으로써 6 바이트를 5 개의 코드워드로 표현할 수 있다. b 가 바이트이고 c 가 코드워드라고 할 때, 진법 변환식은 다음과 같다 [1].

$b1*256^5 + b2*256^4 + b3*256^3 + b4*256^2 + b5*256 + b6 = c1*900^4 + c2*900^3 + c3*900^2 + c4*900 + c5$

그러나 BC 모드의 코드워드수가 5 바이트 미만일 경우, 각 코드워드는 바이트 값과 일대일로 매핑된다.

2.2.5 뉴메릭 컴팩션 모드

뉴메릭 컴팩션(NC) 모드는 숫자가 연속적으로 13 개 이상일 때 사용되며, 인코딩 알고리즘은 다음과 같다. 45 개의 숫자로 구성된 그룹으로 나눈 후, 숫자들의 앞에 1 을 삽입한다. 10 진법을 900 진법으로 변환 함으로써 각 45 개의 숫자 그룹을 15 코드워드로 인코딩 한다. 마지막 그룹의 숫자의 개수는 45 보다 작을 수도 있다. 다음은 디코딩을 하는 예를 보여주고 있다.[1]

코드워드가 1, 624, 434, 632, 232, 200 일 때, 다음의 수식 $1*900^5+624*900^4+ 434*900^3+ 632*900^2+ 232*900+200$ 은 십진수로 1000213298174000 이다. 여기에서 앞의 1 을 빼주어 000213298174000 인 원래의 데이터를 복원할 수 있다. NC 모드에서 하나의 코드워드는 거의 3 개의 숫자를 표현할 수 있다

2.3 오류 검출 및 수정

코드워드를 원래의 데이터로 복원하기 전에 코드워드에 오류가 있는지를 검사하고 수정하기 위해서 PDF417 심볼의 데이터 영역의 가장 뒷부분에는 오류수정 코드워드를 추가한다. 0-8 사이의 오류수정 레벨에 따라서 추가되는 오류수정 코드워드의 수도 달라진다. 레벨이 높을수록 데이터의 안정도는 높아지지만 데이터의 밀도는 낮아진다. 오류수정 레벨이 L 단계 라고 할 때, 각 레벨당 필요한 오류수정 코드워드의 수는 2^{L+1} 개이다 [1].

오류수정 코드워드는 잘 알려진 Reed Solomon 오류제어 코드 알고리즘으로 생성된다.

2.4 GLI(Global Label Identifier)

2.4.1 GLI의 정의

GLI 는 인코딩된 코드워드를 원래의 데이터로 해석할 수 있도록 하기 위해 정의된 코드워드의 집합이다. 각 GLI 에 따라서 코드워드의 해석이 달라진다. 코드워드 927, 926, 925 에 의해 다음과 같은 세가지 다른 GLI 코드워드 집합이 선택된다.[1]

G1, G2, G3, G4 가 0-899 범위의 값일 때,
927, G1 혹은
926, G2, G3 혹은
925, G4

GLI 값은 다음과 같이 정의된다.

GLI = G1 혹은
GLI = (G2 + 1)*900 + G3 혹은
GLI = 810900 + G4

즉, GLI 값으로 811,800 개가 가능하므로, 어느 나라의 언어라도 PDF417 은 표현할 수 있다.

2.4.2 GLI의 사용

3 종류의 GLI 코드워드 집합의 용도는 다음과 같다.

국제 문자 집합: 코드워드 927 에 의해 선택되며, 0-899 의 값을 가진다. PDF417 에서 디폴트 GLI 는 GLI 0 이다.

일반적으로 제안된 GLI: 코드워드 926 에 의해 선택되며, 두개의 코드워드에 의해 900-810,899 범위의 GLI 를 선택할 수 있다.

사용자 정의 GLI: 코드워드 925 에 의해 선택되며, 810,900-811,799 범위의 GLI 를 선택할 수 있다. 이는 closed-system 어플리케이션에 사용된다.

각 GLI 는 그것에 할당된 문자표를 가지고 있으며, 바이트와 문자 사이에 항상 일대일 매핑을 하지는 않는다. 예를 들면 중국어를 표현하기 위해서는 각 문자마다 2 바이트가 필요할 것이다.

디폴트 GLI 값은 0 이므로 GLI0 을 사용한다면, 바코드에 별도로 GLI0 을 명시할 필요는 없다.

2.4.3 한글을 위한 GLI

PDF417 에서 한글을 표현하기 위해서는 코드워드를 한글로 해석할 수 있도록 하는 GLI 문자표가 필요하다. 하지만 이 작업은 PDF417 을 표준화한 AIM 에서 현재 진행 중에 있다. 독자적으로 코드워드에 매핑할수 있는 한글 문자표를 만들어 close-system 어플리케이션으로서 한글을 표현할 수는 있을 것이다.

3. 디코딩 알고리즘

본 논문에서는 바코드 이미지에서 코드워드를 구했다는 가정하에, 얻어진 코드워드에서 원래의 데이터로 해석하는 알고리즘을 설명하기로 한다. PDF417 의 디코딩은 다음과 같은 과정으로 진행된다.

단계 1: <그림 7>과 같은 바코드 이미지를 <그림 8>과 같은 바와 스페이스의 패턴으로 변환한다.

단계 2: <그림 8>의 패턴을 각 클러스터에 따라 코드워드로 변환시키면 <그림 9>의 (a)를 얻을 수 있다. 여기에 2.3 절에서 설명한 RS 알고리즘을 적용하여 3 개의 코드워드가 복구된 <그림 9>의 (b)를 얻게 된다..

단계 3: 복원된 코드워드의 처음 코드워드는 심볼 길이 지시자이고 다음의 코드워드부터 실제 모드 구조가 적용된다. 심볼의 시작은 항상 GLI0 과 TC 모드의 Alpha 부 모드가 적용된다. 모드는 모드랫치와 모드쉬프트에 의해 변경되며, GLI 는 바뀔 수 있다.

단계 4: BC 모드로 변경되는 경우, 2.2.4 절에 설명한 알고리즘에 의해 5 개의 코드워드가 6 개의 바이트 값으로 해석된다.

단계 5: NC 모드인 경우, 2.2.5 절에 설명한 알고리즘에 의해 15 바이트가 44 숫자로 해석된다.

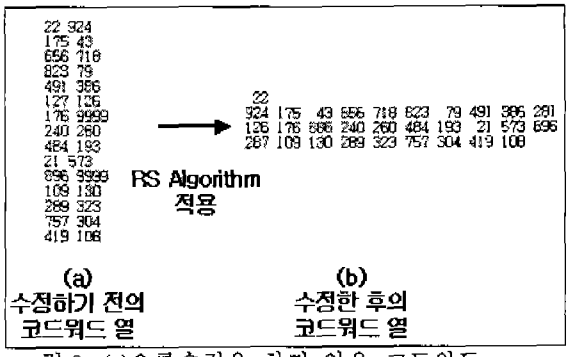
<그림 7> PDF417 이미지

```
00000000 00000000 11113135 14232112 00000000 00000000 0
81111113 41111243 21113144 23332112 41111144 71131111 0
81111113 41111234 21113144 23332111 41111154 71131112 0
81111113 41111414 00000000 32114141 61111142 71131112 0
                      중략
81111113 00000000 13132141 00000000 00000000 71131112 0
81111113 31145111 12232241 21136112 11161241 71131112 0
81111113 31145111 12232241 21136112 11161241 71131112 0
15022000 48100000 00000000 00000000 00000000 00000000 0

|-------||-------||----------------||-------||---------|
    a        b            c            d         e

a, e: 시작, 종료 패턴
b, d: 좌측, 추측 행 지시자
c   : 데이터 코드워드 영역
```

<그림 8> PDF417 의 바, 스페이스 패턴



<그림 9> (a)오류수정을 하지 않은 코드워드
(b) RS-Algorithm 에 의해 오류수정을 한 코드워드

<그림 9> (b)의 코드워드를 디코딩 하는 것을 예로서 단계별로 설명하기로 한다.

단계 1: 심볼 길이 지시자인 첫 코드워드는 22 이므로, 실제 디코딩에 사용되는 코드워드의 개수는 21 개이다.

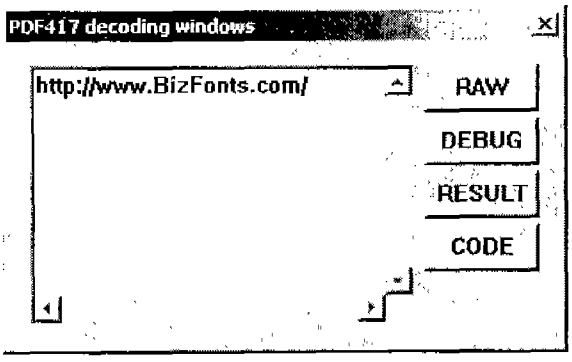
단계 2: 2 번째 코드워드가 924 이므로 BC 모드이며, 인코딩된 바이트의 개수가 6 으로 나누어진다.

단계 3: 3 번째부터 7 번째 코드워드까지 5 개의 코드워드를 2.2.4 절에서 설명한 900 to 256 연산을 하면, 아래의 결과를 얻을 수 있다.

$175*900^4+ 43*900^3+ 656*900^2+ 718*900+823 =$
$104*256^5+116*256^4+116*256^3 +112*256^2 +58*256 +47$

5 코드워드의 열(175, 43, 656, 718, 823)은 6 바이트의 열 (104, 116, 116, 112, 58, 47)로 변환 된다. 구한 6 바이트와 현재의 GLI(디폴트 0)를 비교하여 문자를 얻어낸다. 비교해 보면, " http:/ " 를 얻을 수 있다.

단계 4: 8 번째 코드워드 값이 900 보다 작으므로, 다음에 오는 5 바이트도 동일한 모드로 지속되며, 22 번째 코드워드까지 지속된다. 심볼 길이 지시자의 값이 22 이므로 더 이상 해석할 코드워드가 없으므로 <그림 10>과 같이 문자를 출력하고 디코딩을 종료한다. 이 예에서는 총 24 문자를 구하였다.



<그림 10> 코드워드를 해석하여 나온 문자 정보

4. 결론

본 논문에서는 PDF417 심볼로지의 구조와 디코딩 알고리즘에 대해서 알아 보았다. PDF417 은 최소의 코드워드를 사용하여 최대의 문자를 바코드에 저장하기위해 다양한 컴팩션 모드를 지원하며, 다양한 언어를 지원할 수 있도록 많은 GLI 를 가지고 있고, 강력한 오류 수정기법에 의해 데이터의 안정성을 보장하고 있다. 바이트 컴팩션 모드와 뉴메릭 컴팩션 모드의 구현에 있어서 계산의 중간값의 크기가 C 언어의 32 비트 정수변수의 범위를 넘어서므로 진법변환을 위해서는 특별한 함수를 만들 필요가 있었다. 오류수정 레벨이 높아지면 계산시간이 오래 걸리므로, 알고리즘을 개선하거나, 혹은 오류수정 알고리즘을 하드웨어로 구현하면 계산속도를 상당히 증가시킬 수 있을 것이다. 앞으로의 과제인 한글에 대한 디코딩은 AIM 에서 GLI 를 정의하는 작업이 끝나면 구현할 것이다.

참고 문헌

[1] Uniform Symbology Specification PDF417, AIM USA.
[2] AIM-USA 홈페이지, http://www.aimusa.org
[3] 심볼 테크놀로지 홈페이지, http://www.symbol.com
[4] 2 차원 바코드 페이지, http://www.adams1.com
[5] 오호근, "최신 바코드 기술 및 응용 ", 성안당, 1997