The Asian Journal of TEX, Volume 1, No. 1, April 2007
Article revision 2007/04/01

TEX: 조판, 그 이상의 가능성*

TEX, Beyond the World of Typesetting

조진환 Jin-Hwan Cho
수원대학교 자연과학대학 수학과 chofchof@ktug.or.kr

ABSTRACT 타이포그래피는 넓은 의미에서 출판에 관한 모든 예술 및 기술들을 아우른다. 조판은 타이포그래피 안에서 글자와 그림들로 이루어진 조각 또는 부품들을 판면에 배치하는 과정을 의미한다. 이러한 배치가 얼마나 아름답고 읽기 쉬운가 하는 디자인 측면과 얼마나 간편하고 편리한가 하는 기술적인 측면에서 조판은 발전해 왔다. 이 글에서는 조판의 이러한 측면들과 관련해 탁상출판 소프트웨어 및 워드프로세서, 그리고 TEX 시스템이 어떠한 특징을 가지고 있고, 어떻게 발전해 왔는지 살펴본다. 이러한 논의를 통해 TEX 시스템의 과거를 돌아보고, 현재 위치를 가늠하고, 그리고 미래를 조망한다.

1 타이포그래피: 활자, 조판, 그리고 인쇄

타이포그래피를 논하기에 앞서 이 단어의 의미를 백과사전에서 찾아보자. [15]는 다음과 같이 설명하고 있다.

> – 타이포그래피(typography): 활자를 사용한 인쇄술. 본래 활자로 조판하는 것, 활판(活版)인쇄, 인쇄된 것의 체재(體裁) 등을 뜻하였다. 근대인쇄술 가운데 활판이 가장 일찍 발달, 보급된 것으로, 인쇄술 전반 및 그 표현을 가리키는 말이었다. 그러나 뒤에 인쇄술이 차츰 다양해지고, 그 용도도 넓어져 현대에 와서는 활자뿐 아니라 사진식자(寫眞植字)·전자기술에 의한 인자(印字) 등도 포함해 문자조성의 시각적인 처리를 말하는 경우도 많아지고 있으며, 문자를 살린 레이아웃이나 그래픽디자인과 동의어로 여겨지고 있다. ··· (중략) ···

위의 설명에서 타이포그래피의 의미를 세 가지 구성 요소, 활자(活字; type), 조판(組版; typesetting), 그리고 인쇄(印刷; printing)에서 찾을 수 있다. 타이포그래피는 이 세 가지 구성 요소들이 결합되어 최종 결과물을 만드는 과정 모두를 아우른다. 이 절에서는 컴퓨터에 의한 탁상출판(DTP, DeskTop Publishing)이 출현하기 전까지 타이포그래피의 역사를 세 가지 구성 요소를 중심으로 살펴본다.

*이 글은 2007년 1월 27일 KTUG 5주년 학술발표회에서 강연한 내용을 토대로 작성되었다.

1.1 활자

먼저 활자를 살펴보자. 두 한자 '活(살활)'과 '字(글자자)'가 가지는 뜻 그대로 '살아있는 글자', 즉 '움직이는 글자(movable type)'를 의미한다. 다시 [15]에서 활자에 대한 설명을 찾아보면 다음과 같다.

> – 활자(活字; type): 활판인쇄에 쓰이는 금속의 네모기둥 꼭지면에 글자를 볼록하게 새긴 것. 찰흙에 문자를 새겨 구워낸 활자와 목제활자도 있으나, 오늘날에는 활자 합금을 사용한 금속제의 것이 널리 쓰인다. ··· (중략) ···

활자의 시초는 1041년에서 1048년 사이 중국 송(宋)나라에서 필승(畢昇; Bi Sheng, 990~1051)이 '구운 찰흙(baked clay)'으로 활자를 만들어 사용했다는 기록에서 찾는다.[1] 원(元)나라의 왕정(王禎; Wang Zhen, 1290~1333)은 구운 찰흙으로 만든 세라믹활자의 단점을 극복하기 위해 목제활자를 만들어, 1313년 《농서(農書; Nong Shu)》를 출판했다.[2]

활자를 이야기할 때 중국의 필승이 기록 상 시초로 인정받지만 한국은 중국을 능가하는 타이포그래피 유산들을 많이 가지고 있다.[3] 먼저 1234년 고려 인종 때 활자를 이용한 기록이 있다.[4] 중국의 왕정이 농서를 출간한 1313년과 비교해 보라. 고려시대에 이미 활자를 사용했음을 알 수 있는 명백한 증거는 1377년 고려 우왕 때 청주 흥덕사에서 금속활자로 찍어낸 《직지심체요절(直指心體要節)》이다.[5] 세계에서 금속활자로 만든 가장 오래된 책으로 2001년 유네스코 세계기록유산으로 등록되었다.

이에 비해 서양의 활자는 1440년경 구텐베르크(Johannes Gutenberg, 139?~1468)에 의해 만들어졌다.[6] 1455년에 출판된 《42행 성서》(또는 《구텐베르크 성서》)는 높은 인쇄품질과 비교적 저렴한 가격 때문에 금속활자가 전 유럽으로 급속히 퍼지는 계기가 되었다 [36, Movable type]. 구텐베르크는 비록 세계 최초로 발명한 것은 아니지만 고품질의 금속활자를 비롯해 잘 마르지 않는 유성잉크, 그리고 당시 포도주 생산에 사용되던 프레스를 이용한 인쇄기 등을 함께 만들었다. 무엇보다 이 모든 것들을 이용한 그의 활판인쇄술은 화약, 나침반과 더불어 르네상스 삼대 발명품 중 하나로 취급되며 이후 오백년간 이어지는 근대 활판인쇄술의 기초가 되었다.

---

1. 심괄(沈括; Shen Kuo, 1031~1095)의 《몽계필담(夢溪筆談; Mengxi Bitan)》. 이 기록은 필승이 어떻게 활자를 이용해 인쇄를 했는지 자세하게 설명하고 있다. 영문 설명은 [36, Bi Sheng]을 참고하라. 하지만 깨지기 쉬운 세라믹활자의 특성과 활판 조성에 걸리는 시간 등을 생각하면 널리 사용되지 못했으리라 짐작된다.
2. 왕정은 특히 활판 조성에 걸리는 시간을 획기적으로 단축하는 구체적인 방법을 개발했다. 이 방법에 대한 영문 설명은 [36, Wang Zhen (official)]을 참고하라.
3. 활자를 이용한 것은 아니지만 《무구정광대다라니경(無垢淨光大陀羅尼經, 국보 제126호)》은 세계에서 가장 오래된 목판 권자본(卷子本)이다. 자세한 설명은 [15]를 참고하라.
4. 고려 중기 문인 이규보(李奎報, 1168~1241)의 문집 《동국이상국집(東國李相國集)》은 고려 인종 1234년에 최윤의(崔允儀) 등 17명이 왕명으로 고금의 예의를 수집, 고증하여 50권으로 엮은 전례서 《상정고금예문(詳定古今禮文)》을 활자로 찍었다고 기록하고 있다. 이를 토대로 《상정고금예문》을 한국 최초의 금속활자본으로 추정한다 [15].
5. 정식 명칭은 《백운화상초록불조직지심체요절(白雲和尙抄錄佛祖直指心體要節)》. 하지만 현재 프랑스 국립도서관에 소장되어 있으며 반환을 위해 여러 사람들이 노력하고 있다 [16].
6. 구텐베르크의 금속활자는 납, 주석, 안티몬 합금으로 만들었는데 이 구성은 오늘날까지 이어져 오고 있다 [36, Movable type].

활자가 서양이 아닌 동양에서 먼저 만들어졌지만 왜 더 이상 발전하지 못했을까? 그것은 다름아닌 한자의 영향이라 보아도 무리는 아니다. 알파벳의 수와 한자의 수를 비교해 보라. 그 수많은 한자들을 활자로 만드느니 차라리 목판인쇄술을 이용하는 것이 더 경제적이지 않았을까?[7] 이 문제는 컴퓨터를 이용하는 현대 타이포그래피에서도 마찬가지로 이어지고 있다.[8]

1.2 조판

이제 조판을 살펴보자. [15]에서는 조판을 '원고에 따라 문선한 활자로 인쇄판을 짜는 것'이라 설명한다. 근대 활판인쇄술 초기에는 식자공(compositor)이라 부르는 사람들이 수작업으로 활자를 활판에 배치해서 하나의 판면을 완성했다.[9] 식자공들이 한 자씩 활자들을 조합하는 방식은 당연히 많은 시간을 요구했고, 이로 인한 낮은 생산성은 당시의 많은 신문사들이 여덟 쪽 내외의 신문밖에 만들 수 없었던 큰 요인 중 하나였다.

독일 태생의 미국인 머건탈러(Ottmar Mergenthaler, 1854~1899)는 1886년 하나의 줄을 자동으로 주조(鑄造)하는 기계 '라이노타이프(Linotype)'를 발명했다. 이 자동주식기(自動鑄植機)는 글자의 모형(母型)들을 한 곳에 보관하고 있다가 키보드를 통해 글자가 입력되면 그 글자에 해당하는 모형을 가져와 정렬한 후 하나의 줄이 완성되면 주조하는 역할을 했다.[10]

이에 반해 1885년 미국의 란스턴(Tolbert Lanston, 1844~1914)이 발명한 '모노타이프(Monotype)'는 이름 그대로 한 자씩 주조하고 식자해서 판면을 만드는 자동주식기였다.[11] 기사들의 배치가 자주 바뀌는 신문의 경우 한 줄씩 주조하는 라이노타이프가 더 유용했고, 배치가 고정된 책의 경우 한 자씩 교정이 가능한 모노타이프가 훨씬 유리했다. 신문의 경우 라이노타이프가 가져다 준 생산성의 증대로 쪽수가 네 배 내지 다섯 배가 늘어날 수 있었다 [14].

근대 활판인쇄술에서 조판은 활자나 인쇄에 비해 발전이 늦었다. 자동주식기가 조판에 드는 시간과 비용을 절약했지만 활자를 원하는 위치에 손쉽게 놓을 자유는 계속 제약을 받을 수 밖에 없었다. 차라리 목판인쇄나 수도사들이 책을 필사하던 때가 훨씬 자유롭지 않았을까? 조판이 비로소 꽃을 피우는 때는 컴퓨터 탁상출판의 출현 이후이다.

1.3 인쇄

마지막으로 인쇄를 살펴보자. [15]는 인쇄를 다음과 같이 설명한다.

7. 《직지심체요절》은 흥덕사에서 금속활자본이 간행된 바로 이듬해 취암사에서 목판본이 간행되었다 [16]. 아마 목판인쇄가 시간과 경제적인 면에서 유리하지 않았을까 추측된다.
8. 한자를 빼고는 문서를 작성할 수 없는 중국과 일본에 비해 한자를 많이 사용하지 않는 한국은 다양한 글꼴들을 가지고 있을 뿐만 아니라 새로운 글꼴 개발도 빠르다.
9. 개개의 활자들을 가로로 배치해서 하나의 줄을 얻고 다시 이 줄들을 세로로 나열해서 판면을 얻는 이 방법은 TEX이 판면을 구성하는 그것과 매우 유사하다.
10. 자세한 설명은 [15] 또는 [36, Linotype machine]을 참조하라.
11. 자세한 설명은 [15] 또는 [36, Monotype machine]을 참조하라.

– 인쇄(印刷; printing, press): 판(版)에 잉크를 묻히고 판면의 문자·그림 등을 종이·천 등에 찍어내는 것. 여러 벌의 복제물을 만들어 낼 수 있다. ··· (중략) ···

문자와 그림으로 이루어진 원문은 조판을 통해 활판의 형태로 바뀌고 인쇄를 통해 최종 결과물로 탄생한다. 원문을 보다 좋은 품질로 값싸고 신속하게 대량생산하기 때문에 인쇄는 타이포그래피의 모든 과정에서 매우 중요한 역할을 한다. 구텐베르크가 만든 금속활자가 오백년이 지난 후에도 큰 변화없이 사용되고 자동주식기가 19세기 말에야 비로소 모습을 드러내었던 것에 비해, 인쇄는 여러 면에서 많은 발전을 해 왔다.

판 위에 용지를 올려놓고 부드러운 솜뭉치 등으로 두드리거나 문지르는 인쇄 방식은 구텐베르크가 올리브나 포도를 짜는 데 사용하던 기계를 인쇄에 도입하면서 급속도로 바뀌어 갔다. 강하게 눌러주는 역할을 하는 이 인쇄기로 인해 현재 프레스(press)라는 단어는 인쇄기는 물론, 출판물, 신문, 잡지, 기자 등을 일컫는 용어로 쓰인다.

특히 신속함과 대량생산을 요구하는 신문 제작은 인쇄기의 발달에 많은 영향을 주었다. 1812년 독일의 쾨니히(Friedrich König, 1774~1833)가 만든, 증기를 동력으로 사용하는 원압인쇄기(圓壓印刷機)는 1814년 영국으로 건너가 당시 세계 최대 신문사였던《타임스(The Times)》발행에 쓰였다. 그가 바우어(Andreas Friedrich Bauer, 1783~1860)와 함께 만든 Koenig & Bauer는 지금도 인쇄기를 생산하는 세계 유수의 회사이다.

1833년 미국의 호(Richard March Hoe, 1812~1886)는 '인쇄판을 판통(版筒)에 감고 그것과 종이를 압통(壓筒)으로 가압하여 인쇄하는' 윤전기(輪轉機; rotary press)를 발명하고 [15], 1847년 윤전기로 만든 최초의 신문을 발간했다. 하루에 수백만 장을 인쇄할 수 있는 인쇄기의 등장으로 마침내 출판에서 대량생산은 꿈이 아닌 현실이 되었다.[12]

2 조판의 네 가지 관점

타이포그래피에서 TEX은 조판의 영역을 다루는 시스템이다. 조판은 컴퓨터를 이용한 탁상출판이 출현하면서 더 이상 극소수 전문가들이 소유하는 전유물이 아닌 일반인의 영역이 되었다. 비로소 조판이 타이포그래피의 한 축을 담당하게 됨을 의미한다. 이 절에서는 조판 소프트웨어를 대표하는 탁상출판 소프트웨어와 워드프로세서, 그리고 이 글의 주된 소재인 TEX 시스템을 아름다움, 읽기 쉬움, 편리함, 그리고 자동화라는 네 가지 관점에서 살펴본다.

2.1 아름다움

먼저 '아름다움'이다. 디자인의 측면에서 조판은 레이아웃을 의미한다. [15]는 다음과 같이 레이아웃을 설명한다.

– 레이아웃(layout): 설정된 공간 속에 합목적적·효과적으로 사물을 배치하는 일, 또는 그 기술. ··· (중략) ··· 인쇄용어로서는 신문·잡지 등의 페이지 할당과 활자·

12. 인쇄기의 발달에 대한 자세한 설명은 [15] 또는 [36, Printing press]을 참조하라.

> 기호·괘선의 배치, 배당 지면의 종합을 가리킨다. 특히 그래픽디자인에서는 전달을 위한 표현수단으로서, 사진·삽화·문자·기호·괘선 등의 여러 조형요소를 한정된 공간 안에서 시선의 옮김을 쉽게 하고, 미적으로 효과 있게 배치하는 일을 말한다.
> ··· (중략) ···

탁상출판 소프트웨어들을 흔히 '레이아웃 소프트웨어' 또는 '레이아웃 프로세서'라 부르는 이유가 여기에 있다. 탁상출판의 시작은 1985년 레이아웃 소프트웨어 Aldus PageMaker를 탑재한 애플의 매킨토시 컴퓨터와 그래픽 프로그래밍 언어인 포스트스크립트(PostScript)를 탑재한 레이저프린터 LaserWriter에서 찾는다[36, Desktop publishing].

1987년 처음 출시된 Quark의 탁상출판 소프트웨어 QuarkXPress[33]는 1990년대 전세계 시장의 거의 90퍼센트를 점유하는 명실공히 레이아웃 프로세서의 절대 강자가 되었다. 특히, 레이아웃 디자이너들을 비롯하여 출판계의 조판 전문가들을 탁상출판으로 끌어들이는 역할을 했다. 주위에서 쉽게 접할 수 있는 화려한 색상의 책, 잡지, 브로셔, 그리고 광고 전단들이 모두 이 소프트웨어 또는 비슷한 기능의 소프트웨어들로 제작된다고 해도 과언이 아니다.

매킨토시 클래식 운영체제에서 동작하는 QuarkXPress에 대항해 2002년 어도비는 애플에서 새로 개발한 운영체제 Mac OS X[27]에서 동작하는 InDesign[24]을 출시했다. 현재 이 두 소프트웨어가 탁상출판 소프트웨어 시장을 주도하고 있다. InDesign은 여러 가지 새로운 조판 기능들을 선보였다. 예를 들어, 유니코드를 이용한 다국어 조판이 가능하고 오픈타입[36, OpenType] 글꼴이 가지고 있는 발전된 타이포그래피 기능을 쓸 수 있다. 이러한 조판 기능들 중 많은 것들이 TEX 세계에서 이미 사용되고 있다는 점에 주목하자.[13] 아직 TEX에서 구현되지 않은 것 중 하나로 '옵티컬 커닝'을 들 수 있다.

어떤 글자 다음에 특정한 글자가 따라올 때 글자 사이의 간격을 자동으로 조정하는 것을 '커닝(kerning)'이라 한다. 글꼴(font)을 제작할 때 이 정보를 글꼴 내부에 저장하는 것이 일반적이다.[14] 그림 1은 커닝이 들어간 경우와 그렇지 않은 경우의 차이를 보여준다.

그림 1. 'W'와 'o' 사이에 커닝이 들어간 예(왼쪽)와 그렇지 않은 예(오른쪽)

커닝이 자동으로 들어가지 않는 두 가지 예는 그림 2의 왼쪽과 가운데에서 볼 수 있다. 옵티컬 커닝이란 이런 경우에도 그림 2의 오른쪽과 같이 조판 소프트웨어가 적당한 커닝을 찾아 자동으로 넣어주는 것이다. 이 기능은 무엇이 가장 적당한 커닝인지 판단하기 어렵기 때문에 구현하기 쉽지 않다.

13. 유니코드 다국어 조판을 위해 1990년대 중반에 TEX의 16비트 확장판 Omega[29]가 나왔고, 오픈타입 글꼴의 발전된 타이포그래피 기능들은 '유니코드 TEX'이라 할 수 있는 XETEX[37]이 지원한다.
14. 한글 글꼴에 들어있는 알파벳 글자를 그냥 사용할 때 무언가 허전한 느낌이 들었다면 대부분의 경우 한글 글꼴이 커닝을 저장하고 있지 않기 때문이다.

그림 2. 'W'와 'orld'에 서로 다른 글꼴을 적용했을 때(왼쪽), 그리고 같은 글꼴이지만 서로 다른 크기를 사용했을 때(가운데) 커닝은 자동으로 들어가지 않는다. 오른쪽은 'W'와 'o' 사이에 `-0.15em` 정도 커닝을 주어 옵티컬 커닝을 흉내낸 것이다.

2.2 읽기 쉬움

그렇다면 왜 커닝이나 옵티컬 커닝을 주는 것일까? 이에 대한 대답이 바로 조판을 바라보는 두 번째 관점 '읽기 쉬움'이다. 읽기 쉽다는 것은 아름답다는 것과는 다르다. 읽기 쉬운 레이아웃이 항상 아름다운 것은 아닌 것처럼 아름다운 디자인이 항상 읽기 쉬운 것은 아니다.

여기에서 말하는 읽기 쉬움은 'readability'를 뜻하는 것이 아니라 'legibility'를 뜻한다. 영어 단어 'readability'가 언어 또는 내용의 측면에서 글이 얼마나 읽기 쉽게 쓰여있는가를 의미하는데 반해 'legibility'는 디자인 측면에서 글자 또는 글자들을 배치한 모양이 얼마나 읽기 쉬운가를 의미한다.[15] 읽기 쉬움은 앞에서 언급한 커닝과 그림 3의 합자(合字; ligature)[16]에서 보여주듯 서체 디자인과 밀접한 관련이 있다.

그림 3. Zapfino 글꼴을 사용한 합자(ligature)의 예; 마지막 'o'가 붙는 순간 모든 글자들의 모양이 변한다. 이것은 극단적인 예이고 'fi'가 'fi'로 변하는 것과 같이 주로 두 글자 또는 세 글자가 합자된다.

조판 시스템 TEX의 탄생에도 아름다움과 읽기 쉬움이라는 두 특성을 찾을 수 있다. TEX은 탁상출판 소프트웨어가 등장하기 전인 1978년 스탠퍼드 대학의 크누스(Donald E. Knuth, 1938~)가 만들었다. 크누스가 자신의 저서 [7]에서 밝혔듯이 TEX은 '아름다운' 책을 제작하기 위해 고안되었다.[17] 읽기 쉬움에 대해 TEX은 독특한 '줄나눔 알고리즘

15. 'Readability'와 'legibility'의 차이에 관한 보다 자세한 설명은 [36, Typography]을 참조하라.
16. Zapfino 글꼴은 독일의 저명한 활자 및 서체 디자이너 자프(Hermann Zapf, 1918~)가 1998년 디자인한 것으로 광범위한 합자와 글자의 변화로 유명하다. 예를 들어 알파벳 소문자 'd'는 아홉 개의 모양으로 변한다. 그의 대표작은 1948년과 1952년에 각각 디자인한 'Palatino'와 'Optima' 서체로 좋아하는 사람과 싫어하는 사람이 극명하게 갈리는 것으로 유명하다 [36, Hermann Zapf]. 이 글에서 사용하고 있는 알파벳 서체가 바로 Palatino이다.
17. 탁상출판 소프트웨어나 워드프로세서에 비해 TEX은 안정성(stability), 지속성(consistency), 이식성(portability) 및 공개성에서 뛰어나다. 자세한 내용은 [1, pp. 828–830]을 참고하라.

(line breaking algorithm)'과[18] 다른 조판 소프트웨어에 비해 월등한 수식 조판 능력을 가지고 있다.

TEX은 글자나 그림과 같은 판면의 모든 구성 요소들을 네모 박스(box)로 처리하고, 단어 사이의 공백(space)은 길이가 늘거나 줄어들 수 있는 글루(glue)로 처리한다. 글자 박스들이 모여 하나의 긴 줄을 만드는데 이 줄을 잘라 여러 개의 줄로 이루어진 문단을 만든다. 이 때 TEX의 줄나눔 알고리즘이 적용된다. TEX은 보다 체계적인 줄나눔을 위해 박스와 글루, 그리고 페널티(penalty)라는 세 가지 요소를 이용해 최적의 줄나눔 위치를 계산한다. 하나의 단어를 수정할 때 그 효과가 다음 줄들에만 미치는 것이 아니라 문단 전체에 영향을 주는 이유는 바로 여기에 있다. 마찬가지 이유로 문단의 모양이 바뀌면 다음 쪽들에만 영향을 주는 것이 아니라 이전 쪽에도 영향을 끼친다.[19] 이와 같은 줄나눔 알고리즘의 특징으로 인해 다른 탁상출판 소프트웨어나 워드프로세서와 달리 대다수의 TEX 시스템들은 WYSIWYG[20]을 지원하지 않고 있다.

TEX은 수식 조판에 있어 다른 조판 소프트웨어들에 비해 매우 뛰어나다. 특히 TEX의 수식 입력 방법은 많은 소프트웨어들이 따르는 사실상 표준이 되었다. 수식에 사용되는 여러가지 기호들은 거의 대부분 글꼴에 들어있고, 다른 소프트웨어들도 이 글꼴을 사용할 수 있기 때문에 이것으로 TEX 수식의 장점을 이야기할 수는 없다. 그렇다면 TEX 수식이 읽기 쉬운 이유는 무엇일까? TEX 수식의 특징은 상황에 따라 서로 다른 크기의 공백을 사용한다는 것이다.

먼저 수식 안에 전혀 공백을 주지 않은 상태에서 출발하자.

$$(\sin x+\cos x)^2=(\sin x)^2+2\sin x\cos x+(\cos x)^2=1+2\sin x\cos x$$

이번에는 공백이 필요하다고 생각되는 모든 부분에 균일하게 공백을 주었다. 첫째 줄에서 마지막 줄까지 공백의 크기를 조금씩 늘렸다. 수식을 전문적으로 다루지 않는 소프트웨어들의 경우 다음 세 줄과 같이 균일한 크기로 공백을 처리하곤 한다.

$$(\sin x + \cos x)^2 = (\sin x)^2 + 2 \sin x \cos x + (\cos x)^2 = 1 + 2 \sin x \cos x$$

$$(\sin x + \cos x)^2 = (\sin x)^2 + 2 \sin x \cos x + (\cos x)^2 = 1 + 2 \sin x \cos x$$

$$(\sin x + \cos x)^2 = (\sin x)^2 + 2 \sin x \cos x + (\cos x)^2 = 1 + 2 \sin x \cos x$$

어떤 줄이 가장 아름답고 읽기 쉬워 보이는가? 공백의 크기를 달리 주는 것이 더 좋아 보일 것이라는 생각이 자연스레 들지 않는가?

TEX은 위의 식을 처리할 때 세 가지 크기의 공백을 사용한다. 먼저 sin과 cos 다음에 오는 공백에는 (첫째 줄과 같이) 작은 크기의 `\thinmuskip`을 준다. 또한 '$2 \sin x \cos x$'

18. 크누스와 플라스가 개발한 TEX의 줄나눔 알고리즘은 [9] 또는 [8, Chapter 3]을 참고하라. 일반인도 이해할 수 있는 수학을 사용했기 때문에 이해하기는 어렵지 않지만 한번에 읽기에는 분량이 너무 많다.

19. 한 단어의 수정이 문단 전체는 물론 이전 쪽에도 영향을 주는 이 특징으로 인해 처음 TEX을 경험하는 많은 사람들이 생소함을 느낀다.

20. What You See Is What You Get. 입력과 동시에 최종 출력물과 거의 동일한 결과를 화면에 표시해 주는 사용자 환경을 일컫는다. WYSIWYG을 지원하는 TEX 시스템은 이미 1990년대 초 매킨토시 클래식 환경에서 동작하는 Textures [35]가 있었다.

처럼 2와 $\sin x$ 다음에도 같은 크기의 공백을 준다. 그런데 위의 식을 어떤 방법으로 읽는가? 연산 순서를 생각하자. 덧셈이나 뺄셈보다 곱셈을 먼저 하므로 $\sin x$와 $\cos x$, 그리고 이들의 곱을 하나의 개체로 인식한 다음, 덧셈을 하는 것이 일반적이다. 또한 이렇게 더한 것들이 서로 같다는 것이므로, 등호 기호가 마지막으로 읽힌다. TEX은 이 순서대로, 덧셈과 뺄셈 같은 연산자의 앞뒤 공백에 (둘째 줄에서 사용한) 중간 크기의 `\medmuskip`을, 그리고 등호 같은 관계기호의 앞뒤 공백에 (마지막 줄에서 사용한) 큰 크기의 `\thickmuskip`을 준다. 이렇게 함으로써 보다 읽기 쉬운 수식을 만드는 것이다. TEX이 만든 수식은 다음과 같다.

$$(\sin x + \cos x)^2 = (\sin x)^2 + 2\sin x \cos x + (\cos x)^2 = 1 + 2\sin x \cos x$$

아름답고 읽기 쉬운 최종 결과물을 만드는 것은 모든 조판 소프트웨어들의 공통된 목적이다. 탁상출판 소프트웨어들이나 TEX 모두 이러한 목적을 가지고 출발했다. 탁상출판 소프트웨어의 경우 레이아웃 디자이너의 생각을 얼마나 잘 표현할 수 있느냐에 초점을 두고 있다. 따라서 글자와 그림, 사진 같은 구성물들을 손쉽게 판면에 배치할 수 있는 WYSIWYG 기능은 필수이다. 또한 화려한 색상을 사용하기 위한 도구를 비롯해 글자와 그림에 여러가지 그래픽 효과를 줄 수 있는 기능들을 많이 가지고 있다.

이에 반해 TEX은 판면 조성에 필요한 가장 기본적인 기능 삼백여 개를 명령 형식으로[21] 모은 프로그래밍 언어로, 프로그래밍 언어의 특성상 무한한 확장성을 가진다. 다른 조판 소프트웨어들의 경우 제작사가 제공하는 모듈에만 의존할 수 밖에 없지만 TEX에서는 사용자들이 직접 프로그래밍을 통해 자신에게 맞는 환경을 만들 수 있다. TEX의 장점은 근 삼십년에 이르는 세월동안 전세계의 TEX 사용자들이 만든 수많은 스타일 파일에 있다. 사용하고자 하는 조판 기능에 대한 스타일 파일이 거의 마련되어 있다고 해도 무리는 아니다.

2.3 편리함

이제 세번 째 관점 '편리함'이다. 누구나 간단하고 편리하게 문서의 작성에서 조판, 그리고 출력까지 모든 것을 한꺼번에 할 수 있는 조판 소프트웨어가 바로 워드프로세서(word processor)이다. 워드프로세서는 GUI[22]나 WYSIWYG 같이 탁상출판 소프트웨어와 유사한 기능을 많이 가지고 있지만, 저자가 직접 조판 소프트웨어를 다룬다는 점에서 큰 차이가 있다. 전세계 워드프로세서 시장은 마이크로소프트 워드(Word)가 거의 장악하고 있다고

21. 원시명령(primitive). 원시명령들을 사용하기 편하게 조합한 것을 '매크로(macro)'라 부른다. 특정한 조판 기능을 지원하도록 매크로들을 구성한 것을 '스타일(style)'이라 부르는데 주로 확장자 `.sty`를 가진다. LATEX에서는 문서 전체의 조판 방법에 대한 여러가지 사항들을 모은 '클래스(class)'를 사용한다. 클래스는 확장자 `.cls`를 가지는데 조판에 필요한 스타일들을 부르고 이들을 상황에 맞게 수정하는 역할을 내부에서 수행한다.
22. Graphical User Interface. 컴퓨터의 그래픽 기능을 활용한 사용자 인터페이스를 일컫는 말이다. 반면 UNIX/Linux 운영체제에서 키보드로 명령을 입력받아 수행하는 터미널 환경은 CLI(Command Line Interface)라 부른다.

볼 수 있으며,[23] 최근에는 스프레드시트 및 프리젠테이션 소프트웨어들과 함께 사무용 소프트웨어의 한 축에서 문서 작성이라는 기존 영역을 넘어 발전하고 있다.

워드프로세서와 탁상출판 소프트웨어의 차이는 컴퓨터 및 소프트웨어 기술의 발달로 인해 시간이 지남에 따라 계속 줄어들고 있다. 요즘에는 신속한 제작이 필요한 전산 입문서 및 매뉴얼을 중심으로 많은 책들을 워드프로세서로 제작하곤 한다. 탁상출판 소프트웨어로 제작한 책의 품질이 뛰어남에도 불구하고 이러한 경향이 나타나는 이유는 무엇일까?

경제적 이유는 차치하고 워드프로세서가 신속한 제작을 담보하는 큰 이유 중 하나는 저자가 조판의 거의 대부분을 담당하기 때문이다. 소설이나 일반 교양서의 경우 전형적인 레이아웃이 주를 이루므로 굳이 저자가 원고를 작성할 때 레이아웃을 신경쓸 필요가 없다. 또한 탁상출판 소프트웨어를 다루는 사람이 원고를 입력하고 교정하는 것이 저자에 비해 훨씬 빠르다. 이에 반해 전문서적의 경우 특별한 레이아웃을 많이 요구할 뿐만 아니라 저자가 아닌 비전문가가 원고를 입력함으로써 발생하는 실수도 많고 또한 교정을 할 수도 없다. 차라리 레이아웃을 도와주는 수준에서 저자가 직접 조판을 담당하는 것이 출판 시간을 절약한다.

그러나 양질의 출판을 위해서는 레이아웃과 (글의) 내용을 분리하는 것이 필요하다. 즉, 디자이너의 영역인 레이아웃과 저자의 영역인 내용을 분리함으로써 저자가 레이아웃에 신경쓰는 시간을 줄이고 글 자체에 집중해야 한다는 뜻이다. 탁상출판 소프트웨어의 경우 저자가 이러한 소프트웨어를 다루는 경우는 거의 없으므로 레이아웃과 내용이 '완전히' 분리되었다고 할 수 있다. 하지만 전문서적, 논문 등의 경우 위에서 언급한 문제들로 인해 신속한 출판이 어렵다는 단점이 있다.

TEX은 다른 조판 소프트웨어들에 비해 레이아웃과 내용을 '적절하게' 분리한다.[24] 웹(World Wide Web)에서 CSS(Cascading Style Sheets)[22]를 바꾸면 레이아웃 (또는 테마)가 변경되는 것을 연상하라.[25] 디자이너가 만든 레이아웃은 스타일이나 클래스의 형태로 저자에게 제공되고, 저자는 자신의 문서에 한두 줄만 추가함으로써 레이아웃을 바꿀 수 있다. 주어진 레이아웃에 맞게 저자가 직접 내용을 입력하기 때문에 워드프로세서와 마찬가지로 실수를 줄일 수 있고 교정도 가능하다. 물론 레이아웃을 스타일이나 클래스의 형태로 만드는 것은 다른 조판 소프트웨어에 비해 쉽지 않지만 이미 주어진 스타일이나 클래스를 수정하는 것은 크게 어렵지 않다. 수학과 물리학의 경우 대다수의 논문집을 TEX 시스템을 이용해 출판하는 것은 바로 이러한 이유 때문이다.

2.4 자동화

앞에서 언급한 조판의 세 가지 관점과 함께 인터넷의 급속한 발달은 조판의 '자동화(automation)'라는 새로운 관점을 부각시키고 있다. 비슷한 레이아웃을 가지는 많은 수의

23. 한국은 마이크로소프트 워드에 비해 시장 점유율이 더 높은 워드프로세서, 한글과 컴퓨터의 '아래아한글'을 가진 세계에서 유일한 나라일 것이다.
24. 탁상출판 소프트웨어는 높은 가격과 쉽지 않은 사용법으로 인해 여전히 '완전한' 분리가 대세이다. 이러한 제약이 상대적으로 약한 워드프로세서의 경우 '적절한' 분리를 위한 여러가지 시도가 이루어지고 있다.
25. TEX은 웹에 비해 십년, 그리고 CSS에 비해 거의 이십년 전에 만들어졌다는 점을 기억하라.

문서들을 자동으로 처리하는 경우와 데이터베이스에 들어있는 정보들을 가공해 새로운 출판물을 제작하는 경우가 대표적인 예다. 다른 조판 소프트웨어들과 달리 TEX은 이러한 목적을 위해 자동화를 위한 '基地엔진(base engine)'으로 사용될 수 있다.

비슷한 레이아웃을 가지는 문서들을 자동처리하는 대표적인 예로 수학, 물리, 전산학 및 생물학 분야의 프리프린트(preprint)들을 사십만 개 이상 가지고 있는 'arXiv'[20]를 들 수 있다. 이 곳에 저장된 모든 논문들은 사용자들의 요청이 있을 때 TEX으로 자동처리되어 DVI, 포스트스크립트 또는 PDF 등의 형태로 제공된다. 대규모의 국제 학회를 준비하는 데 TEX이 사용된 대표적인 예는 2003년 호주에서 열린 'ICIAM 2003'[23]이다. 1700명 이상의 연구자들이 참석해 발표한 이 국제 학회의 프로그램을 담은 책은 202쪽, 그리고 발표를 모은 초록집은 무려 447쪽에 달한다.[26] 또한, 수시로 변하는 상품 정보를 데이터베이스에 저장해 백화점 카탈로그를 신속하게 만드는 데 TEX을 사용한 사례가 있고[10], 인도에서는 1200쪽의 트리반드룸 지역 전화번호부를 제작하는 데 TEX을 사용했다[34].

문서 자동화 또는 조판 자동화를 위한 기지엔진으로 TEX을 고려하는 주된 이유는 TEX이 지금껏 보여준 양질의 조판 능력과 강력한 매크로 프로그래밍 기능을 바탕으로 계속 진화하는 시스템이기 때문이다. 크누스가 만든 TEX 자체는 현재 사용되지 않는다. 대부분의 TEX 시스템이 채택하고 있는 가장 일반적인 TEX 엔진은 ε-TEX과 pdfTEX이 결합한 pdfε-TEX으로, 새로운 조판 기능을 위한 원시명령들이 계속 추가되고 있다. 또한 하나의 TEX 엔진에만 국한되어 진화하는 것이 아니라 XƎTEX과 같은 새로운 TEX 엔진도 출현하고 있으며, 이들의 상호작용을 통해 보다 나은 형태로 진화를 거듭하고 있다.

3 조판의 네 가지 관점에 따른 TEX 시스템의 발달

TEX 세계에서도 앞 절에서 살펴본 조판의 네 가지 관점에 따라 여러가지 새로운 포맷과 엔진들이 개발되어 왔다. 개발 순으로 살펴보자.

3.1 $\mathcal{AMS}$-TEX과 $\mathcal{AMS}$-LATEX

미국수학회(AMS)의 요청으로 1983년에서 1985년 사이 Michael Spivak은 $\mathcal{AMS}$-TEX[19]을 개발했다. Plain-TEX에 기반을 둔 이 포맷은 수학자들이 바라는 수준의 '읽기 쉬운' 혹은 '전문적'인 수식을 조판하는 목적에서 출발했으며, 전세계 수학자들이 표준으로 사용하는 TEX 포맷이 되었다. 이후 LATEX의 광범위한 보급으로 1995년 LATEX 2.09 기반에서 동작하는 $\mathcal{AMS}$-LATEX[18]이 발표되었고, 현재 사용하는 $\mathcal{AMS}$-LATEX 2는 LATEX2ε의 표준 패키지로 들어있다. 이 패키지의 대표적인 특징은 그림 4에 주어진 다양한 수학기호를 쓸 수 있을 뿐만 아니라 여러가지 형태의 수식 정렬이 가능하다는 것이다[13, pp. 2–8].

26. Ross Moore가 TEX으로 제작한 이 프로그램과 초록집은 각각 http://www.iciam.org/ProgramBook.pdf와 http://www.iciam.org/BookofAbstracts.pdf에서 볼 수 있다. 그는 같은 해 하와이에서 열린 'TUG 2003' 컨퍼런스에서 'ICIAM 2003' 준비에 관한 상세한 내용을 사례 발표했다[11].

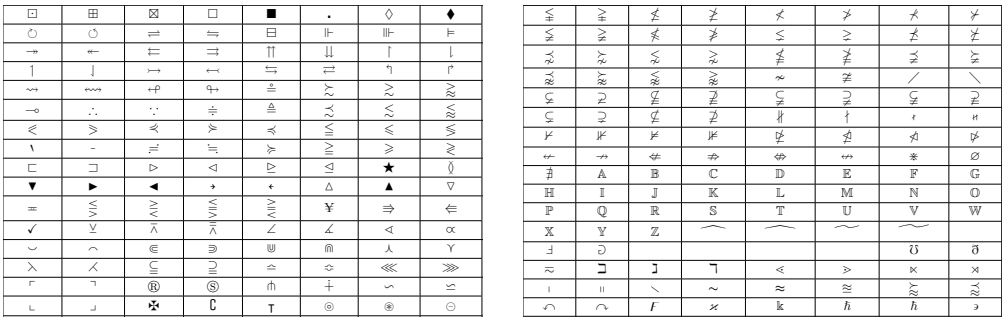

그림 4. AMS 글꼴[17]에 들어있는 다양한 수학기호들. `msam10`(왼쪽)과 `msbm10`(오른쪽)

3.2 LaTeX 2.09와 LaTeX2ε

'편리함'의 관점에서 보면 단연코 LaTeX 포맷을 들 수 있다. 1985년 Leslie Lamport가 발표한 LaTeX 포맷은 '저자는 글이 보여지는 레이아웃보다 글 자체에 집중해야 한다'는 철학을 담고 있다. 이를 위해 TeX에 논리적 구조를 주는 문서 마크업(markup) 언어를 도입했고,[27] 사용자들의 편의를 위해 자동 번호 매기기, 상호 참조, 참고문헌 및 색인 등, 여러가지 자동화된 조판 기능들을 지원했다. 1992년 마지막 버전 LaTeX 2.09가 나왔고, 이후 1994년부터 이를 발전시킨 LaTeX2ε[25]이 LaTeX3 프로젝트 팀에 의해 개발되고 있다. 현재 LaTeX이라고 하는 것은 LaTeX 2.09가 아니라 LaTeX2ε을 의미한다. LaTeX2ε의 마지막 버전은 2005년 12월에 발표되었다.

3.3 ConTeXt와 MetaFun

TeX 세계의 아이디어 뱅크, Hans Hagen이 1990년 만든 ConTeXt[21]는 사용자들이 편하게 레이아웃을 구성할 수 있도록 도와주는 ε-TeX 기반의 포맷이다. TeX 세계에서 LaTeX이 워드프로세서의 역할을 한다면 ConTeXt는 탁상출판 소프트웨어의 역할을 한다. LaTeX처럼 '`\start`환경`...\stop`환경' 형태의 논리적 구조를 제공하지만, LaTeX이 수많은 사용자들이 만든 외부 스타일에 의해 유지되는 것에 반해 ConTeXt는 모든 매크로들을 내부에 포함하고 있다는 점이 큰 차이다. ConTeXt는 탁상출판 소프트웨어가 가지는 여러가지 세밀하고 화려한 그래픽 기능들을 TeX과 METAPOST를 통해서도 구현할 수 있다는 것을 보여준다. 그림 5는 ConTeXt와 MetaFun을 이용한 여러가지 그래픽 효과들을 담고 있다.

3.4 PyTeX, PerlTeX, 그리고 LuaTeX

TeX과 다른 프로그래밍 언어를 연결하려는 '자동화' 시도는 꾸준히 있어 왔다. 대표적인 세 가지 프로그래밍 언어는 Python[32], Perl[30], 그리고 Lua[26] 언어이다. 먼저 Python을 이용한 PyTeX[31]부터 살펴보자.

27. HTML 마크업 언어와 비교해 보라. LaTeX은 '`\begin{`환경`}...\end{`환경`}`' 구조를 가지는 데 비해 HTML은 '`<`환경`>...</`환경`>`' 구조를 가진다.

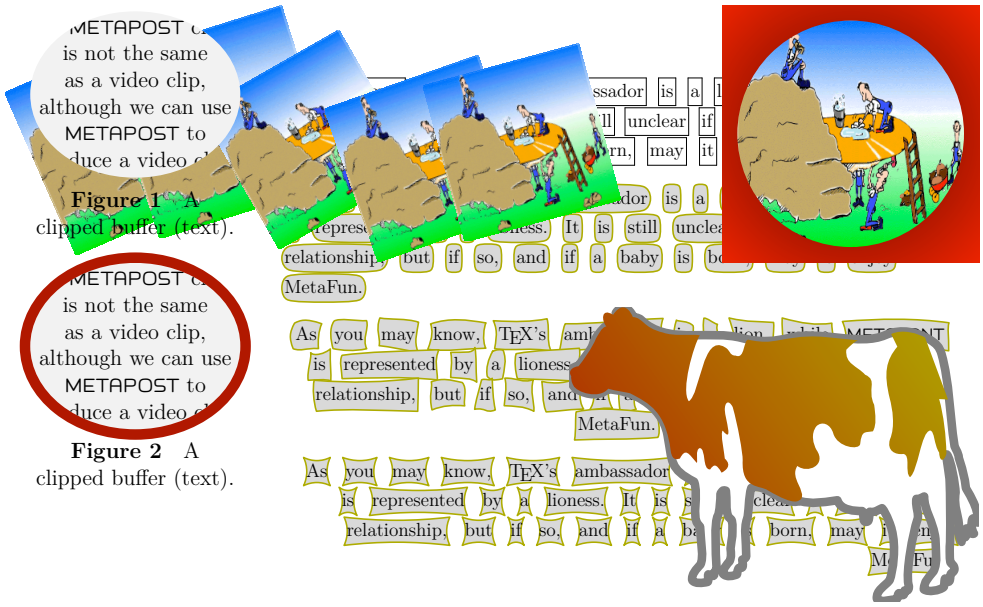

Figure 1 A clipped buffer (text).

Figure 2 A clipped buffer (text).

그림 5. ConTEXt와 MetaFun을 이용한 다양한 그래픽 효과들 [28]

Jonathan Fine은 서버에서 TEX을 데몬(daemon)으로 실행하는 방법 [2]을 기반으로, 2005년 TEX을 Python 클래스로 만든 PyTEX을 발표했다 [3]. 다음 코드는 Python TEX 클래스를 이용해 주어진 문자열을 컴파일하는 예로 결과값은 DVI와 로그 정보이다 [31].

```
from tex import tex, plain
document = 'My beautiful \\TeX\ document.\n'
(dvi, log) = tex(plain, document)
```

PyTEX이 Python에서 TEX을 실행하는 단방향 소통만 가능한 것에 반해 Scott Pakin의 PerlTEX은 Perl과 TEX의 양방향 소통이 가능하도록 고안되었다 [12]. 즉, Perl 코드가 들어간 TEX 문서를 Perl을 통해 처리한 후 이 결과를 다시 TEX으로 처리하는 과정을 반복한다는 것이다. 아래의 PerlTEX 예제는 주어진 수만큼 '*'를 가지는 매크로 \asts를 정의하는 예다 [12, p. 150].

```
\newcommand{\asts}{}
\perlnewcommand{\astsperl}[1]
  {'\renewcommand{\asts}{' . '*' x $_[0] . '}'}
```

위의 모델들이 TEX과 외부 프로그래밍 언어가 서로 소통하는 것에 반해 Hans Hagen은 2005년 [4]에서 TEX 자체에 Lua 프로그래밍 언어를 임베드(embed)하는 새로운 엔진 LuaTEX을 제시했다. 또한 유니코드와 오픈타입 글꼴을 지원하는 pdfTEX 개발을 목적으로 그와 Taco Hoekwater는 2006년 콜로라도 주립대학의 지원을 받아 pdfTEX, Aleph,[28] 그리고 LuaTEX을 붙인 새로운 엔진 개발을 시작했다 [6, 5].

28. ε-TEX의 기능을 Omega에 추가한 16비트 TEX 엔진으로 Giuseppe Bilotta가 만들었다.

4 맺는 말

활자(또는 글꼴), 조판, 그리고 인쇄라는 세 가지 구성 요소들이 결합되어 최종 결과물을 만드는 과정을 모두 아우르는 타이포그래피는 출판에 관한 예술 및 기술들이 총괄된 매력적인 분야다. TEX은 활자를 판면에 배열하는 활판인쇄술의 여러가지 조판 기능들을 매크로 프로그래밍 언어로 구현한 독특한 조판 소프트웨어로, 현대 디지털 타이포그래피에서 근 삼십년 동안 아름다움과 읽기 쉬움이라는 디자인 측면과 편리함과 자동화라는 기술적 측면을 두루 갖춘 조판 시스템으로 자리매김해 왔다. 앞 절에서 조판의 이러한 네 가지 측면들에 대해 TEX 시스템이 어떠한 특징을 가지고 있고, 또 어떻게 발전해 왔는지 살펴보았다. TEX은 지금껏 보여준 양질의 조판 능력과 강력한 매크로 프로그래밍 기능을 바탕으로 계속 진화하고 있으며, 이러한 이유로 문서 또는 조판의 자동화를 위한 기지엔진으로 새로운 역할을 부여받고 있다.

우리나라는 타이포그래피의 찬란한 역사를 지니고 있으며, 세계에서 가장 선진적인 한글을 가졌지만 한글 조판에 대한 연구는 한글 서체 개발 등에 비해 미미한 것이 사실이다. 한글 타이포그래피, 특히 한글 조판에 대한 충분한 연구를 바탕으로 현재 매크로 수준에 머물고 있는 한글 TEX을 뛰어넘어 한글의 묘미를 잘 살린 새로운 한글 TEX 엔진의 탄생을 고대한다.

참고 문헌

1. 조진환, TEX과 타이포그래피에 관한 소고, 『韓國數學教育學會誌 시리즈 E』 19권 4호 (2005), 823–837.
2. Jonathan Fine, *TEX as a callable function*, EuroTEX 2002 Proceedings, 2002, pp. 26–35. http://www.pytex.org/doc/euro2002.pdf
3. ———, *TEX forever!*, EuroTEX 2005 Proceedings (Volker RW Schaa, ed.), 2005, pp. 140–149. http://www.pytex.org/doc/eurotex2005.pdf
4. Hans Hagen, *LuaTEX: Howling to the moon*, TUGboat **26** (2005), no. 2, 152–157.
5. ———, *MKII–MKIV*, TUGboat **27** (2006), no. 2, 219–227.
6. Taco Hoekwater, *Opening up the type*, TUGboat **27** (2006), no. 1, 16–17.
7. Donald E. Knuth, *The TEX book*, Addison-Wesley, 1986.
8. ———, *Digital Typography*, CSLI Lecture Notes, no. 78, CSLI Publications, 1998.
9. Donald E. Knuth and Michael F. Plass, *Breaking paragraphs into lines*, Practice and Experience **11** (1981), 1119–1184.
10. Stephan Lehmke, *ERCOTEX: Yet another database publishing application of LATEX*, TUGboat **24** (2003), no. 1, p. 63, abstract.
11. Ross Moore, *Using TEX to manage IT for a mathematics congress*, TUGboat **24** (2003), no. 1, p. 24, abstract.
12. Scott Pakin, *PerlTEX: Defining LATEX maros using Perl*, TUGboat **25** (2004), no. 2, 150–159.
13. American Mathematical Society, *User's Guide for the amsmath Package*, ver. 2.0. CTAN: macros/latex/required/amslatex/math/amsldoc.pdf

14. Michael Spear, *The Linotype Machine: Thomas Edison called it the "Eighth Wonder of the World"*. http://oncampus.richmond.edu/academics/journalism/lino.html
15. 야후! 백과사전. http://kr.dic.yahoo.com
16. 직지찾기운동. http://korea.emc.com/local/ko/KR/news/events/jikji/
17. AMSFonts 2.2d. http://www.ams.org/tex/amsfonts.html
18. $\mathcal{AMS}$-LaTeX2. http://www.ams.org/tex/amslatex.html
19. $\mathcal{AMS}$-TeX 2.2. http://www.ams.org/tex/amstex.html
20. arXiv.org, e-Print archive. http://arxiv.org
21. ConTeXt. http://www.pragma-ade.com/showcase.pdf
22. Cascading Style Sheets. http://www.w3.org/Style/CSS/
23. 5th International Congress on Industrial and Applied Mathematics, Sydney, Australia, 7–11 July 2003. http://www.iciam.org/iciamHome/
24. Adobe InDesign. http://www.adobe.com/kr/products/indesign/
25. LaTeX project: LaTeX — A document preparation system. http://www.latex-project.org
26. The Programming Language Lua. http://www.lua.org
27. Max OS X. http://www.apple.com/macosx/
28. MetaFun. http://www.pragma-ade.com/general/manuals/metafun-p.pdf
29. The Omega Typesetting and Document Processing System. http://omega.enstb.org
30. The Perl Directory at Perl.org. http://www.perl.org
31. PyTeX— Python programming plus TeX typesetting. http://www.pytex.org
32. Python Programming Language — Official Website. http://www.python.org
33. QuarkXPress. http://www.quark.com/products/xpress/
34. TeX Helps in Timely Publication of BSNL's Trivandrum Telephone Directory. http://www.tug.org.in/bsnl.html
35. Textures. http://www.bluesky.com/products/textures.html
36. Wikipedia, the free encyclopedia. http://en.wikipedia.org
37. The XeTeX typesetting system. http://scripts.sil.org/xetex