



Hangul-ucs를 이용한 옛한글 조판

Typesetting Old Hangul with Hangul-ucs

이기황 Kihwang Lee

연세대학교 대학원 언어정보학과정 kihwang.lee@yonsei.ac.kr

ABSTRACT This article describes a method of typesetting old Hangul with `hangul-ucs`, a L^AT_EX package for handling UTF-8 encoded Korean texts. Typesetting old Hangul is not a trivial task, which involves a number of tough challenges such as text input method, character encoding, and font-related matters, etc. `Hangul-ucs` offers a powerful old Hangul typesetting facility which excels conventional document preparation tools in many ways. To serve as a practical guide, this article shows the old Hangul typesetting process using `hangul-ucs` based on real-world examples. A few technical and theoretical issues regarding the principles of character composition of Hangul and character code system are also presented hoping that it will ultimately provide better understanding of the wide range of technical problems underneath diverse Hangul processing tasks including typesetting old Hangul with L^AT_EX.

1 머리말

우리 민족이 사용하는 고유의 언어는 ‘한국어’이며, 이 한국어의 표기에 사용되는 고유의 문자 체계는 ‘한글’이다.¹ 익히 알려진 대로 한글은 15세기에 세종과 그에 뜻을 따르는 학자들이 당시 한국어의 음운적 특성을 면밀히 분석한 결과를 반영하여 발명한 표음 문자 체계이다.²

전통적으로 문자 생활은 종이로 대표되는 매체와 붓 등의 필기 도구를 사람의 손으로 조작하여 이루어졌으나 활자의 발명 이후 문자 생활의 기계화 노력이 끊임없이 시도되었다. 그 결과 오늘날 우리는 비교적 큰 어려움 없이 기계화된 문자 생활을 영위하고 있다. 그러나 컴퓨터에서의 자유로운 한글의 사용에 있어서는 아직도 해결되어야 할 문제들이 있다. 많은 사람들의 관심 밖에 놓인 옛한글과 관련된 문제는 더욱 그러하다.

이 글에서는 최근 큰 발전을 이룩한 L^AT_EX을 이용한 옛한글 조판 방법에 대하여 소개하고자 한다. L^AT_EX을 이용한 옛한글 조판을 시도해 보려는 이들을 위한 실질적인 지침을 제공하는 것을 목표로 하는 이 글은 L^AT_EX에서의 옛한글 조판에 관한 상세한 기술적 사양과 발달의 역사에 대해서는 다루지 않는다. 또한 모든 설명은 마이크로소프트 윈도우 XP

1. 흔히 혼동되지만 언어 체계인 ‘한국어’와 문자 체계인 ‘한글’은 구별되는 개념이다.
2. ‘한글’은 주시경이 붙인 이름이며, 창제 당시 붙여진 이름은 ‘훈민정음(訓民正音)’이다. ‘훈민정음’이라는 문자 체계를 설명한 책의 이름 역시 ‘훈민정음’이다. 이 글에서 책으로서의 ‘훈민정음’은 『훈민정음』으로 적는다.

운영 체제 상에서 KTUG Collection 2006(이하 KC2006)[22]을 설치하여 사용하는 것을 기준으로 한다.³

이 글의 구성은 다음과 같다. 2절에서는 조판을 비롯한 한글 처리에 관련된 여러 가지 기술적 문제의 이해를 돕기 위해 한글 문자 운용 원리와 문자 코드 체계를 살펴본다. 이어서 3절에서는 L^AT_EX을 이용한 옛한글 조판의 과정을 실제 예를 통하여 보인다. 마지막으로 4절은 이 글의 내용을 요약하면서 L^AT_EX을 이용한 한글 텍스트 조판을 디지털 인문학이라는 문맥에서 짚어 보면서 미래를 전망한다.

2 배경

2.1 옛한글의 개념

‘옛한글’이란 ‘현대한글’에 대비되는 개념이다.⁴ 현대한글은 1933년 한글학회가 ‘한글맞춤법통일안’을 제정한 이후에 한국어의 표기에 사용된 한글을 말한다. 그러나 옛한글 혹은 현대한글이라는 개념은 단순히 시기적으로만 정의할 수는 없다. 1933년 이후에도 그 이전에 사용되었던 표기법을 특별한 목적으로, 예를 들어 상품명 등의 표기를 위해 ‘훈글’이라는 표기를 사용한 경우는 ‘옛한글’이라고 보아야 한다.⁵ 또한 표기 대상 언어를 한국어만으로 못 박는 것도 힘들다. 1933년 이전에도 외래어와 외국어를 표기해야 할 경우가 있었기 때문이다. 그러므로 옛한글의 개념을 다시 정리하면 옛한글은 “훈민정음이 창제된 1443년으로부터 한글맞춤법통일안이 제정된 1933년까지의 시기에 한국어와 외래어, 그리고 외국어를 표기하던 문자”라고 할 수 있다.

우리에게 있어서 옛한글은 보통 현대국어의 정서법으로는 표기할 수 없는 문자를 지칭한다. 그러나 옛문헌에 그러한 문자들만 쓰인 것은 아니다. 옛문헌에는 현대한글에서도 사용되는 문자들, 이를테면 ‘ㄱ’이라는 자소 문자나 ‘나’라는 음절 문자도 사용되었다. 그렇다면 이들 문자는 옛한글이기도 하고 현대한글이기도 하다. 그러나 옛한글의 범위를 이렇게 넓히면 옛한글에 대한 일반적 인식과는 큰 거리가 생긴다. 그러므로 옛한글에서 현대한글에서 사용되고 있는 한글은 제외하여야 한다.

2.2 한글의 문자 조합 원리

이 소절에서는 한글의 코드 체계와 조판 등의 바른 이해에 도움을 줄 수 있는 한글의 문자 조합 원리에 대하여 알아본다.⁶ 한글은 음소 문자로서의 특성과 음절 문자로서의 특성을

3. 이 글에서 소개하는 내용의 대부분은 한글텍사용자그룹(KTUG)의 위키 사이트[23]에서 가져온 것이다. 위키 문서의 작성에 헌신하는 모든 분들께 깊은 감사를 드린다.

4. 이 소절의 내용은 [8]을 참조하여 작성하였다.

5. 옛한글과 비슷한 용어로 ‘고어’라는 용어가 있다. 그러나 앞서 설명한 바와 같이 고어는 문자 체계가 아닌 언어 체계를 뜻하는 용어이므로 적합하지 않다. 또한 ‘옛글자’라는 용어가 쓰이기도 하였는데, 이 용어는 예전에 쓰였던 문자인 구결이나 이두까지도 포함하게 되므로 역시 적합하지 않다.

6. 이 소절의 작성에는 [3]을 참조하였다.

2.3 한글 코드 체계

정보화 시대의 문자 생활에 있어서 새로 등장한 개념은 ‘문자 코드 체계’이다. 문자 코드 체계는 시각적 기호인 문자를 비시각적이고 수치적인 코드로 표상하는 방법이다. 이 때, 각각의 자소를 하나의 문자 코드로 표상할 수도 있으며, 조합된 음절 글자를 하나의 문자 코드로 표상할 수도 있다. 보통 앞의 것을 ‘조합형’이라 부르며 뒤의 것을 ‘완성형’이라고 부른다.⁸

한글 코드 체계 표준이 마련되기 전에는 각기 장단점을 가진 조합형 코드와 완성형 코드가 경쟁하며 사용되다가 1987년 정부에 의해 표준 한글 코드 체계가 제정되었는데 그것이 KS C 5601-1987 규격이다.⁹ 이 규격은 문자 인코딩 방식 명칭을 따라 EUC-KR 완성형 한글 코드로도 널리 알려져 있다. 이 코드 체계는 현대 한글 고빈도 음절 글자 2,350자만을 포함하고 있다는 치명적인 약점이 있었다. 따라서 옛한글 처리는 원천적으로 불가능하였다.

1990년대에 이르러 전세계의 모든 문자를 하나의 문자 코드 체계로 일관되게 다룰 수 있는 산업 표준인 유니코드가 주창되었다. 한글이 반영되기 시작한 것은 1996년 발표된 유니코드 2.0부터이다. 한글은 크게 두 영역에 배정되어 있다. 첫 번째는 음절 영역으로 11,172자의 현대 음절이 들어가 있으며, 두 번째는 자소 영역¹⁰으로 240개의 현대 및 옛 한글 자소가 들어가 있다.¹¹

한글이 유니코드에 포함됨으로 해서 그 동안 해결되지 못했던 한글 입출력의 많은 문제들이 해법을 찾은 것은 사실이나 모든 문제가 해결된 것은 아니다. 첫 번째 문제는 음절 영역에 국어 교육과 연구에 필요한 불완전 음절형 문자들이 포함되어 있지 않다는 것이며, 두 번째 문제는 현재 등록된 자소의 선정이 적절치 못하여 조합하지 못하는 옛한글 음절 글자가 있는 것이다.¹²

유니코드는 1995년 국내 표준으로도 채택되었고 마이크로소프트, 한글과컴퓨터, 삼성 전자 등의 소프트웨어 개발 업체들은 유니코드를 자사의 제품에 채용하기 위한 연구를 시작하였다. 이들 업체들이 따로따로 제품 개발에 박차를 가하던 1999년, 21세기 세종계획의 지원을 받아 설립된 비표준문자등록센터(현 문자코드연구센터)[14]의 주관으로 “문서 편집기에서의 옛한글 및 확장한자 처리에 관한 산학 협동 협의회”가 개최되었다. 이 협의회를 통하여 이들 3개 업체는 이후 개발하는 한국어 워드프로세서의 옛한글 및 확장한자 처리를 위한 공통의 문자 코드를 사용하기로 합의하였다 [5].

이들이 공통으로 사용하기로 한 한글 코드 체계는 마이크로소프트에서 ‘워드 2000’을 개발하며 작성한 것으로 옛문헌 조사를 통해 확인된 5,299자의 옛한글 음절 글자를 사용

8. 병서, 연서, 부서 등에 의해 생성된 자소들을 하나의 코드로 표상하지 않고 기본 자소들의 조합으로 처리하는 미세 조합형도 이론적으로 가능하다.

9. 이 규격은 후에 KS X 1001로 개칭되었다.

10. 이 글에서는 이 영역의 코드를 ‘첫가끝 코드’라고 한다.

11. 유니코드 문자표는 [28]에서 볼 수 있다.

12. [8]에서는 옛한글 코드 체계에 반영할 옛한글 자소를 고르는 자세한 기준을 제시하고 있다.

자 영역에 할당하고, 또 다른 사용자 영역에 초성 자소 124개, 중성 자소 94개, 종성 자소 141개를 할당한 코드 체계였다. 이 자소 집합은 유니코드에 등록된 한글 자소 240개에 [7]에서 보고한 추가되어야 할 옛한글 자소 121개를 통합한 것이다. 물론 이 코드 체계는 비표준 코드 체계이다. 이 코드 체계를 그대로 반영한 글꼴은 한양시스템에서 제작하였고, 그에 따라 이 코드는 ‘한양PUA 코드’로 알려지게 되었다.¹³

한양PUA 코드는 바람직한 옛한글의 처리와는 거리가 먼, 단지 옛한글 텍스트를 입력하고 이를 보기 좋게 출력하는 데에만 관심을 두고 만들어진 코드 체계로 심각한 문제를 일으켰다. 가장 큰 문제는 한양PUA 코드를 이용하여 입력된 옛한글 텍스트를 유니코드의 표준인 첫가끝 코드로 변환하는 일이 불가능하여 옛한글 전자텍스트의 활용 가치가 크게 떨어진 것이다.¹⁴ 한국어의 정보화를 목표로 한 정부 주도의 대규모 프로젝트인 ‘21세기 세종계획’에서 구축한 옛문헌 말뭉치가 그 대표적인 사례이다.

결국 마이크로소프트에서는 ‘워드 2002’부터는 위의 방식을 수정하여 현대한글은 완성형 음절 글꼴을 사용하되 옛한글은 자소 조합에 의한 조합형만을 사용하기로 하였다. 한글 텍스트를 저장할 때에는 현대한글은 유니코드의 음절 영역 코드를, 옛한글은 자소 영역의 첫가끝 코드를 사용하는 것을 원칙으로 하되, 유니코드에 포함되지 않은 자모는 유니코드에 포함된 자모들의 조합으로 표현한다는 원칙을 세웠다. 이 원칙은 문자코드연구센터에서 2004년에 수립한 “옛한글 자료 코드 표현의 기본 원칙 및 지침”에 반영되었다 [6].

2.4 정보화 시대의 『훈민정음해례』의 재해석

글쓰이는 앞에서 보인 문자 코드 체계의 혼란이 발생한 것은 『훈민정음해례』에 제시된 문자 조합 원리에 대한 정보화 시대에 걸맞는 적극적인 재해석이 불충분하게 이루어진 데에 기인한다고 본다.

『훈민정음해례』에 제시된 내용은 크게 28자의 기본 자모 글자를 만드는 제자 원리와 이들 기본 자모 글자들을 실제 문자 생활에 이용할 때에 적용되는 문자 조합 원리의 두 부분으로 구성된다. 문제는 이 두 부분의 정보를 선언적으로 처리할 것인가, 아니면 절차적으로 처리할 것인가 하는 것이다. 제자 원리와 문자 조합 원리까지 모두 선언적으로 처리하는 방법이 곧 완성형 처리 방법에 해당한다. 반대로 절차적으로 처리한다면 이는 곧 조합형 처리 방법에 해당한다.

선언적 처리와 절차적 처리는 각기 장단점이 있다. 선언적 처리를 할 경우에는 제자 및 문자 조합 과정의 최종 출력물인 음절 글자를 조합하는 데에 필요한 복잡한 규칙에 대해서 신경 쓸 필요가 없다. 그 대신 전체 음절 글자가 차지해야 하는 공간은 매우 크다. 요구되는 공간의 크기에 대해서는 여러 가지 주장이 있다. 자소들의 거의 무한한 조합을 인정하는 ‘열린 조합’[4]을 인정할 경우 생성 가능한 음절 글자의 개수는 이론적으로 약 18억 개, 혹은 약 370억 개에 이른다는 주장도 있다. 이러한 주장은 너무 극단적인 것이

13. 이 코드 체계는 MS 확장 유니코드로 불리기도 하였다.

14. 2004년 KTUG에 의해 비표준 한양PUA 코드와 표준 첫가끝 코드 대응표가 제작되어 3.2에서 논하는 hypua2jamo 및 jamo2hypua 유틸리티가 탄생하였다.

라고 할지라도 생성 가능한 음절 글자 전체를 미리 생성하여 선언적으로 제시하는 것은 현실적으로 불가능한 일일 수도 있다. 특히 옛한글의 경우 현재까지 옛문헌에서 조사된 음절 글자는 약 5,300여자라고 하지만 앞으로 새로운 옛한글 음절 글자가 발견되지 않으리라는 확고한 보장은 할 수 없다. 더욱이 문제가 되는 것은 한글의 특성상 음절 글자 구성 자소로 분리하는 처리를 해야 할 때가 있는데 이를 위해서는 매우 복잡한 처리를 해야만 한다.¹⁵

절차적 처리의 경우에는 선언적 처리의 경우와 반대의 현상을 보일 것이다. 그러므로 현재 채택된 옛한글을 포함한 한글 처리에서는 선언적 처리와 절차적 처리를 혼합하는 것이 보통이다. 그러나 그러한 혼합식 처리는 근본적인 문제의 해결을 가져오지는 못하였다.

위의 선언적 처리와 절차적 처리는 밀접한 관련을 지녔지만 서로 다른 층위에 존재하는 한글 음소, 문자, 코드, 글꼴에 걸쳐 모두 적용되는 것이다. 여기서 우리가 짚고 넘어가야 할 것은 이 네 체계가 그 내용에 있어서 모두 일대일로 대응하지는 않는다는 것이다. 예를 들어, 현대 한국어에서 하나의 독립된 음소인 /ㅈ/의 경우 문자로는 두 개의 기본 자소 ㅈ와 ㅉ의 결합인 ㅈ로 표상된다. 문자 코드와 글꼴의 경우도 하나의 요소인 코드 ㅈ와 글꼴 ㅈ로 표상된다.

이와 반대로, 현재 유니코드에 등록되지 않은 자소의 경우에는 기존에 유니코드에 등록된 자소의 코드들을 결합하여 나타내야 한다. 물론 이는 애초에 유니코드에 한글 자소들을 잘못 등록된 결과이다.¹⁶ 그러나 앞서 언급한 바와 같이 새로운 옛한글 음절 글자와 이를 구성하는 새로운 한글 자소가 발견되지 말라는 법은 없기에 처음부터 기초 자소들만 등록하는 것이 옳았으리라는 주장도 할 수 있다. 그러나 이러한 절차적 처리는 한글 음절 글자를 초성, 중성, 종성의 고정된 세 요소로만 보려는 우리의 고정적인 시각으로는 쉽게 받아들이기 어려운 것일 수도 있다.¹⁷

한글 글꼴의 경우에도 결국에는 선언적 처리와 절차적 처리가 혼재된 방식을 이용하게 되었음은 앞 소절에서 확인하였다. 그런데 역시 코드 체계와 일대일 대응을 하고 있지는 않다. 하나의 음절 코드는 하나의 음절 글꼴에 대응하지만 하나의 자소 코드는 여러 개의 자소 글꼴에 대응한다. 즉, 조합된 음절 글자의 품질 보장을 위해 소위 ‘별’을 두어 목표로 하는 음절 글자의 모양에 따라 적절한 자소 글꼴을 골라서 사용하는 것이다.¹⁸

결국 우리의 과제는 유니코드의 문제점을 타하는 것을 넘어서서 한글을 유연하게 처리할 수 있는 이론적, 기술적 토대를 마련하는 것이다.

15. 제한된 자소들로 조합 가능한 모든 음절이 배열된 현대한글 음절의 경우는 비교적 단순한 산술 계산에 의해 자소 분리가 가능하다.

16. 옛한글이 아니더라도 유니코드에 등록된 자소만으로는 ‘뀌었’을 합쳐서 발음할 때 나는 소리를 적는 짚을 조합할 수 없다. [11] 참조.

17. 최근 휴대 전화 자판의 한글 입력 체계는 절차적 처리의 한 극단을 보여준다. [2]는 한글과 같이 모아쓰기가 이루어지는 다른 문자 체계의 처리에 있어서도 절차적 처리가 주조를 이루고 있음을 소개하고 있다. 이 문제에 관해서 ISO에서 작성된 문서 [10]도 참조하라.

18. 이에 대해서는 뒤에서 다시 설명한다.

3 L^AT_EX을 이용한 옛한글 조판

L^AT_EX을 이용한 옛한글의 조판은 `hangul-ucs` 패키지 [21]의 `dhucs-midkor.sys`를 통하여 이루어진다. `Hangul-ucs` 패키지는 UTF-8 방식으로 인코딩된 유니코드 한글 텍스트의 조판을 지원하는 L^AT_EX 패키지로서 2,350자로 제한된 EUC-KR 완성형 한글 음절의 조판만이 가능했던 종래의 방식에서 탈피하여 유니코드에 포함된 11,172자의 모든 현대 한글 음절의 조판을 가능하게 하는 획기적인 발전을 이룩하였으며,¹⁹ 결국에는 `dhucs-midkor`라는 옛한글 조판을 위한 부가 도구가 개발되기에 이르른 것이다.

`Hangul-ucs/dhucs-midkor`를 이용하여 옛한글 조판하기 위해 필요한 것들은 다음과 같다.

(4) `Hangul-ucs/dhucs-midkor`를 이용한 옛한글 조판의 필요 요소

- ㄱ. `Hangul-ucs` 패키지 및 이 패키지를 사용할 수 있는 T_EX 환경
- ㄴ. 옛한글 글꼴
- ㄷ. 옛한글 텍스트 또는 옛한글 입력 도구

글머리에서 밝힌 바와 같이, 이 글에서는 사용자가 KC2006을 마이크로소프트 윈도 XP에서 사용하는 것으로 간주하므로 첫 번째 요소는 갖춰진 것으로 보고, 두 번째와 세 번째 필요 요소에 대하여 먼저 알아본 후에 연이어 `hangul-ucs/dhucs-midkor`를 이용한 실례를 통하여 옛한글 조판 과정을 살펴보자.

3.1 옛한글 글꼴

L^AT_EX을 이용한 옛한글 조판에 있어서 가장 큰 장애는 은글꼴에 견줄만한 적절한 옛한글 글꼴이 없다는 것이다. 따라서 옛한글 조판을 위해서는 상용 옛한글 글꼴을 이용해야 한다.

현재 `hangul-ucs`를 이용한 옛한글 조판에 이용할 수 있는 글꼴은 한글 오피스 2000과 함께 보급된 오피스 플러스팩에 포함된 바탕 옛한글 자모(`obatang.ttf`)와 굴림 옛한글 자모(`ogulim.ttf`)이다.²⁰ 이들 글꼴 파일을 구하는 것은 쉽지 않은 일일 수도 있는데, 굴림 옛한글 자모의 경우는 몇몇 사이트에서 구할 수 있는 것으로 알려져 있다.²¹

19. 이는 물론 11,172자의 음절 글자에 해당하는 글꼴이 존재하였기에 가능한 일이다. 특히 은글꼴 [27]의 개발은 L^AT_EX에서의 한글 조판의 발전에 있어서 지대한 공헌을 하였다. `Hangul-ucs`는 조판 가능한 한글 음절 개수의 한계를 없앴을 뿐만 아니라 한글 문서 고유의 타이포그래피 구현에 있어서 많은 발전을 하였다. 본격적인 한글 문서 조판에 이용하기는 어려우나 CJK 패키지 [19]에서도 UTF-8 환경을 지원한다.

20. 같은 옛한글 자모 계열의 돋움 옛한글 자모(`odotum.ttf`)와 궁서 옛한글 자모(`ogungseh.ttf`)도 `tfm`를 추출하면 사용할 수 있다.

21. 옛한글 조판을 위한 실질적인 지침을 제공하려는 이 글에서 필수 요소인 옛한글 글꼴에 관한 사항에 대하여 소극적일 수밖에 없는 것이 안타깝다. 글쓴이는 [25]에서 굴림 옛한글 자모 글꼴이 포함된 `ohfont.exe`를 얻을 수 있었다.

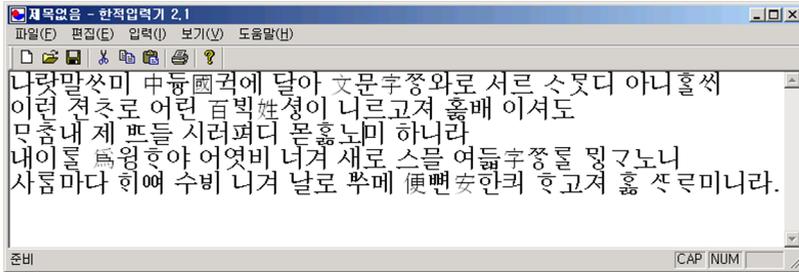


그림 2. 한적입력기의 옛한글 입력 화면

또 한 가지 방법은 한국역사정보통합시스템에서 구할 수 있는 ‘한적입력기’를 사용하는 것이다 [16]. 그림 2에서 볼 수 있는 것처럼 이 프로그램은 입력기라기보다는 작은 편집기에 가깝다. 이 입력기로 입력된 옛한글도 한양PUA 코드로 되어 있으므로 ohi2cb의 경우와 같이 코드 변환을 해야 hangul-ucs에서 조판할 수 있다.

이외에도 마이크로소프트 윈도 환경에서 옛글을 입력할 수 있는 방법이 몇 가지 더 있으므로 관심 있는 독자는 한글텍사용자그룹의 관련 위키 페이지 [15]를 참조하기 바란다.

만일 옛한글 입력을 위해 워드프로세서를 사용했다면 텍스트를 유니코드 형식으로 저장한 뒤에 앞서 보인대로 코드 변환을 해야 한다.²⁵

3.3 옛한글 L^AT_EX 소스 문서 작성 및 컴파일

옛한글 조판을 위한 L^AT_EX 소스 문서를 작성할 때에는 조판하려는 문서의 내용에 따라 몇 가지 고려할 점이 있다. 그것은 입력과 유지가 상대적으로 번거로운 옛한글 텍스트를 어떻게 소스 문서에 효과적으로 포함시킬 것인가에 관련된 것이다.

먼저 조판 대상 문서가 전체적으로 옛한글이 포함된 문서이거나 옛한글이 포함된 텍스트가 문서의 중간에 한 번 정도만 인용문 형식으로 삽입될 경우 소스 파일을 되도록이면 옛한글이 포함되지 않은 상태로 작성하고 본문은 별도의 텍스트 파일로 유지하는 것이 좋다. 다음에서 그림 3에서 보인 『연행별곡』을 조판하는 과정을 살펴보자.

먼저 ‘21세기 세종계획’ 웹사이트 [13] 자료실에서 구한 한글로 입력된 원본 파일에서 필요한 부분만 복사해서 새 파일에 붙인다. Memoir 클래스의 운문 조판 환경을 이용하기 위해 약간의 편집 작업과 찾기 바꾸기를 이용하여 필요한 L^AT_EX 명령들을 단다. 이어서 이 파일을 적당한 이름의 UTF-8 텍스트 파일로 저장한 뒤 앞 절에서 설명한 코드 변환을 하여 최종 파일을 만든다.²⁶ 이렇게 준비된 파일을 EmEditor에서 불러온 모습은 그림 4와 같다.

다음 단계는 위에서 만든 파일을 포함할 L^AT_EX 소스 파일을 아래와 같이 작성하는 것이다. 평이한 oblivoir [24] 클래스 문서이고 dhucs-midkor와, 앞서 작성한 본문 파일을

25. ‘21세기 세종계획’ [13]에서 구축한 역사 자료 말뭉치도 이에 해당함은 앞서 설명했다.

26. 편집 작업을 한글에서 할 수도 있지만 글씨는 화면 글꼴이 옛한글을 표시할 수 있는 글꼴로 설정된 EmEditor에서 작업을 하였다.

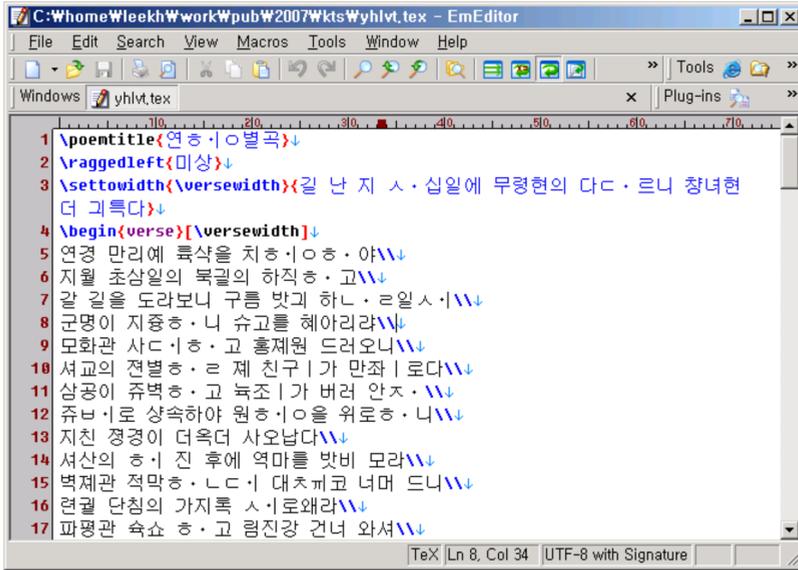


그림 4. EmEditor 상의 『연행별곡』

Hangul-ucs/dhucs-midkor의 옛한글 기본 글꼴은 바탕 옛한글 자모이다. 또 하나의 옛한글 자모 글꼴인 굴림 옛한글 자모를 사용하려면 다음과 같은 줄을 글꼴을 바꾸기 전에 두면 된다.

```
\def\mymidkorfont{ogul}
```

4 맺음말

이상에서 살펴본 hangul-ucs/dhucs-midkor를 이용한 옛한글 조판은 유니코드의 자모 영역, 이른바 첫가끝 영역으로 입력된 옛한글 텍스트를 현재 이용 가능한 옛한글 자모 글꼴을 이용하여 조판하는 것을 가능하게 했다는 데에 큰 의미가 있다. 첫가끝 코드의 사용은 유니코드의 표준에도 부합하며 한국어 텍스트 처리에도 여러 모로 유리하다.

이 기술이 XML 문서 조판 기술과 결합된다면 그 활용의 폭은 매우 넓어질 것이다. 따라서 전자 텍스트의 이용이 주된 관심사인, 최근 대두되고 있는 ‘디지털 인문학(digital humanities)’[18]의 관점에서 이 기술의 중요성은 매우 크다고 하겠다.

앞서 언급되었지만, 현재 hangul-ucs/dhucs-midkor를 이용한 옛한글 조판에 있어서 가장 중요한 과제는 적절한 글꼴의 개발이다. 특히 복잡한 자모 조합 규칙인 mslvt 등이 없어도 음절 조합이 가능한 GSUB(glyph substitution table)²⁷ 기술이 적용된 글꼴의 개발이 요청된다.²⁸

27. [9] 참조. [26]에서 GSUB가 실험적으로 적용된 은바탕Odal 글꼴을 얻을 수 있다.

28. 그러한 글꼴이 개발된다 해도 다양한 현대 한글 글꼴과의 어울림 등의 문제는 쉽게 해결되지 않을 것이다.

전자 텍스트 처리의 핵심 기술 가운데 하나인 조판 기술, 특히 옛글의 조판 기술에 요구되는 글꼴 등의 언어 자원의 개발은 쉽게 성취하기 어려운 일이며, 대부분의 사람들에게는 관심 밖의 일일 것이다. 그렇다고 정부에 이러한 일을 떠맡기는 것도 그리 내키는 일은 아니다. 가까운 시일 안에 좋은 결과를 얻으리라는 보장이 없기 때문이다. 그러므로 이제껏 그래왔듯이, 결국 때가 이르면 이 일은 몇몇 희생적인 선구자들에 의해 성취되리라 믿는다.²⁹

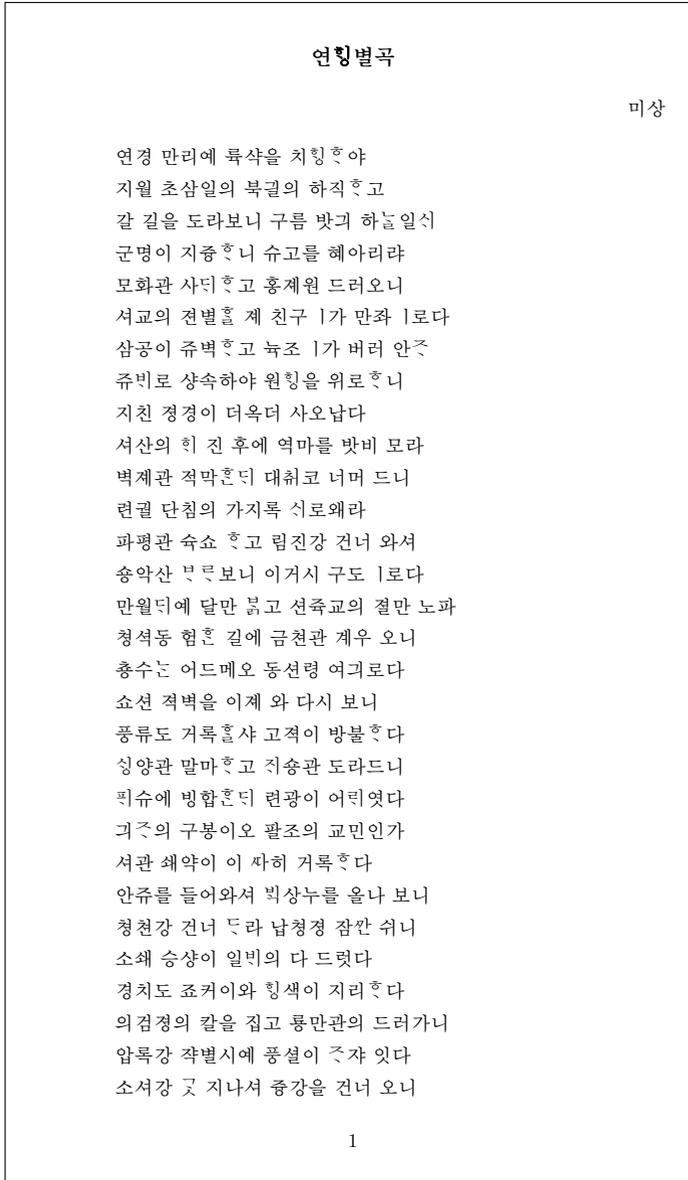


그림 5. 『연행별곡』 조판 결과

29. 물론 관심이 있다 하더라도 넘어야 할 기술적 장벽이 상당히 높은 것은 사실이다.

참고 문헌

1. 고영근, 『중세표준국어문법론』, 탑출판사, 1987.
2. 김경석, 모아쓰기 글자계(script)의 국제 표준화 흐름, 『2002년도 비표준 문자등록센터 옛한글 정보 처리 표준화 워크숍 자료집』, 2002, presentation.
3. 김석득, 훈민정음: 훈민정음의 이해와 말글살이의 역사, 『제10회 국외 한국어 교사 연수회 자료집』, 2006.
4. 이균하, 한글의 열린 조합과 이를 위한 오토마타, 『제3회 한글 및 한국어 정보 처리 학술 발표 논문집』, 1991.
5. 정우봉, 『비표준문자등록센터 사업 보고서』, 비표준문자등록센터, 1999.
6. ———, 『21세기 세종계획 문자코드 표준화 연구 보고서』, 국립국어원, 2004.
7. 홍윤표, 『한글 코드의 연구』, 국립국어원, 1995.
8. ———, 『옛한글의 개념에 대하여』, 『유니코드 정보 환경과 옛한글』, 『옛한글 국제표준화 산학 협동 심포지움』, 비표준문자등록센터, 2002.
9. Adobe Systems Inc., *The Glyph Substitution Table*. http://partners.adobe.com/public/developer/opentype/index_table_formats1.html
10. Kent Karisson, *New Ordering Rules for Hangul, including Rules for Historic Characters, and a DPRK Delta*, Working Group Document ISO/IEC JTC 1/SC2/WG2 N2405, International Organization for Standardization, 2001. <http://std.dkuug.dk/JTC1/SC2/WG2/docs/n2405.pdf>
11. Sang-Oak Lee, *Graphical Ingenuity in the Korean Writing System: With New Reference to Calligraphy*, *The Korean Alphabet: Its History and Structure* (Young-Key Kim-Renaud, ed.), Univ. of Hawaii Press, 1997.
12. Microsoft Typography, *Developing OpenType Fonts for Korean Hangul Script*, 4 2003. <http://www.microsoft.com/typography/OpenType%20Dev/hangul/intro.msp>
13. 21세기 세종계획. <http://www.sejong.or.kr>
14. 문자코드연구센터. <http://www.code.re.kr>
15. 옛한글입력. <http://faq.ktug.or.kr/faq/%BF%BE%C7%D1%B1%DB%C0%D4%B7%C2>
16. 한국역사정보시스템. <http://www.koreanhistory.or.kr/intro/listDataDown.jsp>
17. 森のソフトアトリエ (OHI2CB). <http://homepage3.nifty.com/sohutoatorie/win/o2c.htm>
18. The Alliance of Digital Humanities Organizations. <http://www.digitalhumanities.org>
19. The CJK package for L^AT_EX. <http://cjk.ffii.org>
20. EmEditor Text Editor — Unicode text editor for Windows. <http://www.emeditor.com>
21. KTUGFaq: Hangul-ucs. <http://faq.ktug.or.kr/faq/Hangul-ucs>
22. KTUGFaq: KTUG Collection 2006. <http://faq.ktug.or.kr/faq/KTUGCollection2006>
23. KTUGFaq: Front Page. <http://faq.ktug.or.kr>
24. KTUGFaq: Karnes/Oblivoir. <http://faq.ktug.or.kr/faq/Karnes/Oblivoir>
25. OHFonts, weboldhg. <http://cobra.ee.ntu.edu.tw/~mainlander/yethangul.zip>
26. UnbatangOdal. <http://chem.skku.ac.kr/~wkpark/project/font/GSUB/UnbatangOdal>
27. KLDP.net: Un Fonts. <http://kldp.net/projects/unfonts/>
28. The Unicode Character Code Charts By Script. <http://www.unicode.org/charts/>