CONTENTS PLUS

Journal of Korean Society of Media & Arts Vol.14, No.2

DOI : 10.14728/KCP.2016.14.02.005

한글 아스키 아트의 자동 생성

Automatic Generation of Han-gul ASCII Art

주저자
유미 (You, Mi)
단국대학교 영화콘텐츠전문대학원
Graduate School of Cinematic Content, Dankook University
anubodhih@gmail.com

Abstract

This paper suggests a method that automatically generates Hangul ASCII art that is composed of Hangul letters chosen by the user or KS X 1001 Hangul code sourced from input images. First, we analyze 'TrueType' fonts (.ttf) and 'TrueType' collection files (.ttc), after which we automatically extract the characteristics of individual fonts. Second, we compute the brightness of each pixel in the input images. Third, we map the calculated data to letters. Our method creates stable and reliable Hangul ASCII art from various input images and inputted Hangul letters easily.

Keywords

Hangul ASCII art, automatic generation, KS X 1001 Hangul code

국문초록

이 논문은 KS X 1001 완성형 한글 코드에 선별된 한글 자모음 및 사용자가 임의로 입력한 한글을 기반으로, 이미지 및 영상을 사용자가 원하는 글꼴로 이루어진 한글 아스키 아트로 손쉽게 자동으로 변환해 주는 기법에 대해 다루고자 한다. 이를 위하여 먼저 사용자가 원하는 트루 타입 글꼴(.ttf) 및 트루 타입 글꼴 묶음(.ttc) 파일 분석을 통해 각 글꼴별 특성을 자동으로 추출하고, 입력된 이미지의 명도를 분석하여 이를 사용자가 입력한 글자에 매핑시킨다. 우리의 알고리즘으로 생성된 결과물은 다양한 이미지와 다양한 입력 한글에 대하여 신뢰도 높고 안정적으로 아스키 이미지를 생성할 수 있음을 보여주고 있다.

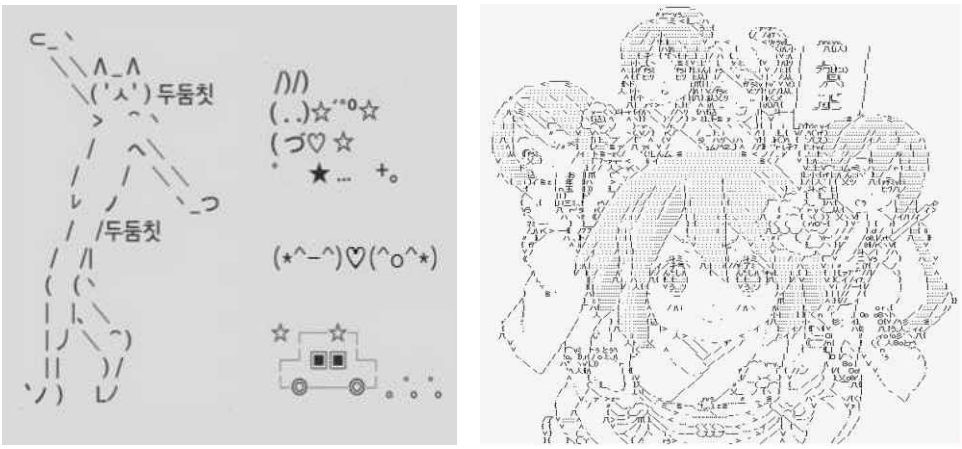
중심어

한글 아스키 아트, 자동 생성, KS X 1001 한글 코드

1. 서론

전통적으로 아스키 아트(ASCII Art) 란 아스키 코드 0x20~0x7e에 포함되는 문자, 기호를 사용한 그림을 말한다. 텍스트 아트, 문자그림으로도 불리며, 영문 이름 첫 자를 줄인 AA로 지칭되기도 한다. 아스키 아트는 이모티콘에서도 찾아 볼 수 있는데, 특히 이모티콘의 1세대인 텍스티콘은 가장 대표적인 아스키 아트로서, 〈figure 1〉의 (a) (Kakaotalk Emoticon, n.d.)에서 보이듯이 몇 개의 글자로 표정이나 캐릭터 등을 이미지처럼 표현한다. 이후 이모티콘은 최근 스마트폰과 카카오톡 등의 보급으로 3D 멀티미디어 이모티콘들까지 등장하고 있어 여기서 다루고 있는 아스키 아트와는 다소 거리가 있는 방향으로 발전하고 있지만, 아직까지도 텍스티콘은 많은 사람들에게 사용되어지고 있다. 하지만 텍스티콘은 주로 몇 줄 안에 표현되는 단순한 이미지를 말하는 반면, 아스키 아트는 주로 여러 줄에 걸쳐 기호를 배열하여 사진이나 풍경 등을 묘사한 작품으로 〈figure 1〉의 (b) (*ASCII Art* Image, n.d.)와 같은 이미지를 지칭한다.

전통적인 1바이트(8bit) 문자 이외에, 일본의 히라가나나, 한국의 한글과 같이 2바이트(16bit) 문자를 가지고 아스키 아트를 사용할 수도 있다. 엄밀히 이들 문자는 아스키 코드에 포함되어 있지는 않으므로 JS 아트나 KS 아트라는 이름이 정식 명칭이지만, 오늘날에는 흔히 구분 없이 아스키 아트라는 용어로 통칭된다.



(a) (b)

〈figure 1〉 Example of ASCII art

초창기의 아스키 아트는 텔레타이프(Teletype) 를 이용하여 글자의 배열을 통해 이미지를 전송할 수 있다는 사실에 착안하여 시작되었다. 이때부터 뛰어난 아스키 아티스트들은 직접 다양한 글자들을 배열하여 아스키 아트를 생성하였으며, 현재도 이러한 수작업 방식으로 제작된 아스키 아트가 여러 용도로 활용되고 있다. 그러나 이렇게 수작업으로 아스키 아트를 생성하는 방식은 아티스트의 감각과 노하우에 크게 의존하며, 제작 시간도 오래 걸리는 고된 작업이다. 이 때문에 이미지를 입력하면 자동으로 아스키 아트를 생성해주는 소프트웨어가 제공되기도 한다.

그러나 기존 아스키 아트 제작 연구 및 소프트웨어에서 한글을 다루고 있는 사례를

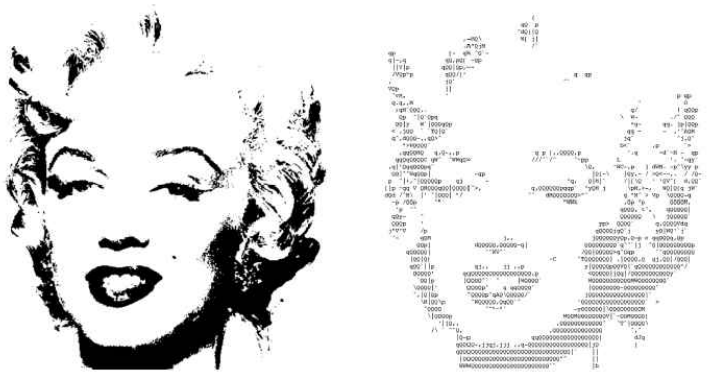
찾기는 힘들다. 이는, 먼저 컴퓨터에서 한글을 처리하는 방식의 차이 때문인데, 영문이나 전통적인 아스키 코드는 1바이트 문자인데 반해 한글은 2바이트 문자이므로 데이터 처리 및 조판 방식에 차이가 발생하기 때문이다. 또 다른 문제는 한글이 가지고 있는 문자적인 독특성 때문인데, 알파벳처럼 풀어쓰는 형태가 아닌 한자와 같이 받침이 있는 모아쓰기 형태이므로 글자의 조합이 방대하고 상당히 많은 글자를 다루어야 한다.

따라서 본 논문은 보다 쉽게 한글에 맞는 아스키 아트를 자동으로 생성하는 방법을 제안한다. 이를 위하여 먼저 사용자가 아스키 아트를 제작하기 원하는 한글 글꼴 파일을 분석하여 글꼴마다 다양한 글자별 메타데이터를 추출한다. 그 이후, 입력받은 이미지를 이미지 처리 과정을 통하여 아스키 아트 제작에 적합하도록 보정한 다음 분석한 글꼴 정보와 비교하여 자동으로 아스키 아트 이미지를 생성해낸다. 본 연구의 결과는 다양한 사용자의 글자 및 이미지 입력에 대해서 아스키 아트 이미지를 쉽고 안정적으로 생성해 낼 수 있음을 보여주고 있다.

2. 관련 연구

인터넷의 채팅과 같이 텍스트를 기반으로 하는 시스템에서 아스키 아트는 이미지를 표현하는 효율적인 방법으로 대중의 관심을 많이 받고 있으며, 이를 반영하듯 아스키 아트는 논문보다는 툴로 더 많이 개발이 되어 온라인에서 쉽게 찾아볼 수 있다. 그러나 아스키 아트의 용도를 확장하거나 효율적인 생성 방법에 대한 학술적인 연구도 몇몇은 찾아볼 수 있었다.

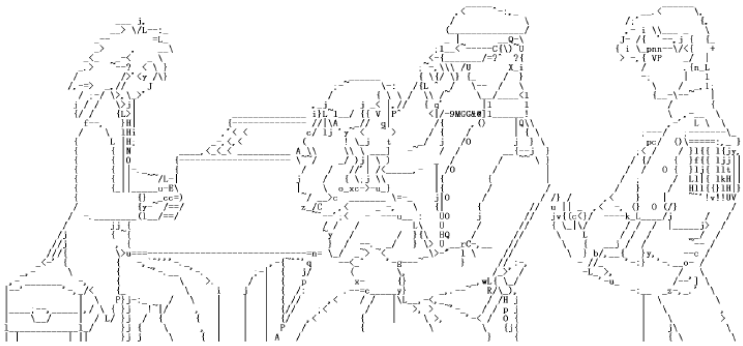
2.1 아스키 아트의 연구 사례



<figure 2> ASCII art image generated by *Winner-takes-all* algorithm

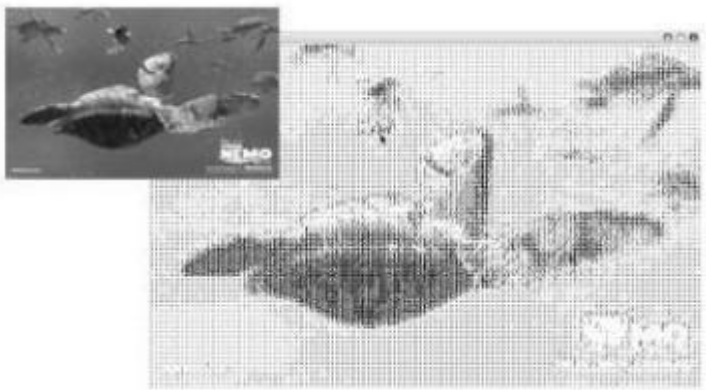
아스키 아트를 이미지에서 자동으로 생성하는 것에 그치지 않고 이를 보안에 사용하는 방법이 제안되었다(Kalpana, 2013). 이 논문은 NNF(Non-negative Matrix Factorization)방법을 사용하여 자동으로 이미지에서 아스키 아트 이미지로 변환하였다. 이미지에서 문자를 매핑할 때는 Winner-takes-all 정렬 방식을 사용하여 아스키 아트를 생성하였다. 이 방법은 톤을 기반으로 하는 아스키 아트(Tone-based ASCII)

제작 방식으로 높은 해상도일수록 좋은 결과물을 발생시킨다. 저자는 여기에 그치지 않고, 이를 신호 처리나 암호 기술에 접목하여 정보 보안에 사용하는 방법까지 확장하여 제안하였다.



〈figure 3〉 Structure-based ASCII art

톤 기반 아스키 아트(Tone-based ASCII) 제작 방식은 하프톤(Halftone)과 같은 결과를 발생시켜 해상도에 의해 제약을 많이 받는다. 이를 해결하기 위해 Xu는 구조 기반 아스키 아트(Structure-based ASCII) 제작 방식을 제안하였다(Xu & Wong, 2010). 이 논문은 입력된 이미지의 형태를 분석하여 주된 선들을 추출하여 비슷한 형태를 그리는 방식을 선보였다. 위의 방식은 입력된 이미지의 외각선을 벡터화 시켜 라인을 추출한 후 외각선을 글자의 간격에 따라 생성한 그리드 셀에 맞게 변형을 시킨 후 이 외각선과 가장 적합한 글자를 대체하는 과정을 통해 아스키 아트 결과물을 생성시켰다. 저자는 그리드 셀을 이용하여 위치나 회전, 크기에 대한 차이를 계산하여 모양을 카테고리화 하고 정렬하는 AISS(Alignment-Insensitive Shape Similarity)라는 새로운 방법을 제안하였다. 이 방법을 통해 해상도와 같은 제약 없이 어떠한 크기의 이미지에서도 인풋 이미지 모양을 유지할 수 있는 아스키 아트를 생성시킬 수 있었다.



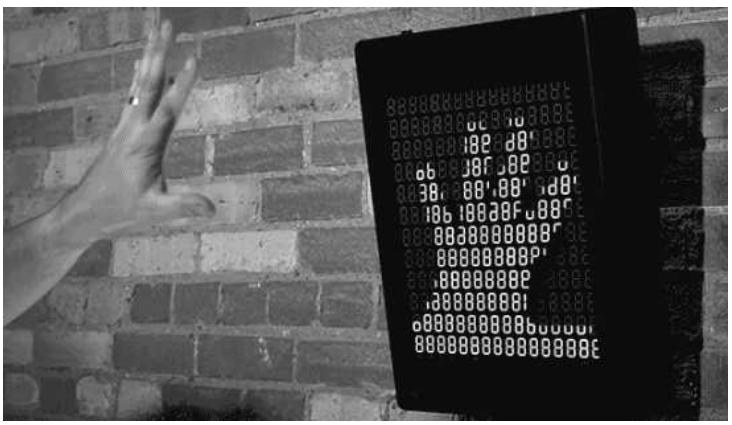
〈figure 4〉 ASCII art using *Processing* language

국내 논문으로는 디지털 아트 작가를 위해 개발된 프로세싱(Processing) 언어를 이용한 아스키 아트 연구(Lee & Park, 2007)가 있다. 위의 연구는 색을 채워 나가는 솔리드(solid) 방식의 아스키 아트 표현 기법을 사용하였으며, 명도별로 분류한 11개의

아스키 코드를 컴퓨터의 명도256단계에 적용하여 각각의 픽셀의 명도에 맞는 코드를 생성하였다. 특히 실시간 아스키 아트 영상을 구현하기 위해서 웹캠(webcam)을 통해 PC로 받은 320*240 사이즈 영상을 인풋으로 사용하였으며, 이를 작품 제작 및 전시로까지 확대하고자 기획한 연구이다.
하지만 이들 모두 사용한 아스키 코드는 영어와 특수문자의 조합으로 이루어진 코드 배합이었으며, 한글의 조합을 연구한 사례는 찾아볼 수 없었다.

2.2 아스키 아트의 제작 사례

아스키 아트가 사람들의 시선을 끄는 이유는, 텍스트라는 정보 전달 방식이 시각 예술 작품으로 변모하는 의외성의 매력에 있다. 사실 한자나 이집트의 상형 문자에서도 알 수 있듯이, 텍스트는 본래 이미지에 그 뿌리를 두고 있다. 하지만 텍스트는 보다 간결하게 많은 정보를 담기 위하여 이미지와 분리되어 독자적인 플랫폼으로 자리 잡았고, 사람들의 인식에서도 텍스트와 이미지는 서로 역할이 다른 독자적인 영역으로 분리되었다. 그러나 정보화 시대에 접어들어 새로이 등장하게 된 아스키 아트는, 텍스트와 이미지의 경계를 허물고 고착화된 정보 전달 매개채로서의 텍스트라는 인식을 무너뜨림으로써, 문자 언어가 고유하게 가지고 있는 이미지로서의 예술성을 찾아 회귀하는 움직임이라고 볼 수 있다. 아스키 아트는 가볍게 주고받을 수 있는 인터넷 상에서의 단순한 놀음에서부터, 전문적인 예술가들의 진지한 예술적 고찰까지 담아낼 수 있는 매우 폭넓고 생명력 넘치는 예술 분야라 할 수 있다.



<figure 5> D.I.G.I.T. digital mirror display by teehan+lax labs

국외에서 이루어진 아스키 아트 사례는 T+L랩스(Teehan + Lax Labs)에서 찾아볼 수 있다. T+L랩스는 아스키 아트를 전자시계에서 사용되는 형식과 같이 7개의 획으로 이루어진 디스플레이에 표현하는 디지트(D.I.G.I.T.)프로젝트가 진행 중이다(D.I.G.I.T., n.d.). 아두이노와 라즈베리파이를 이용해 만들어진 디스플레이에는 카메라가 장착되어 있어 움직이고 있는 물체를 인식하여 이를 아스키 아트로 변환시켜 디지털 문자를 디스플레이 한다. 현재까지는 공책 정도의 크기를 가지고 있으나 이후 몸 전체가 들어가는 크기까지 만드는 것을 목표로 삼고 있으며, 좀 더 빠르게 인식하고 표현해 실시간으로 구동될 수 있도록 개발하고 있는 중이다.
아스키 아트의 독특한 사례 중의 하나는 프로그램을 짜는 소스코드로 아스키 아트를

제작할 수도 있다. 창의적인 C 프로그래밍 기법을 경쟁하는 IOCCC(International Obfuscated C Code Contest) 에서 2000년도 우승을 차지한 사람은, 아스키 아트 형태의 C 언어 소스코드를 제작한 'dhyang' 이라는 유저였다. 이 소스코드는 실제로 동작하는 코드이며, 컴파일해서 실행해 보면 또 다른 아스키 아트로 이루어진 소스코드를 만들어 내는 매우 매력적이면서도 창의적인 결과물을 보여주고 있다.

〈figure 6〉 ASCII Art of IOCCC 2000

아스키 아트에 대한 사랑은 일본에서 특히 두드러지게 확인할 수 있는데, 일본 최대의 커뮤니티 사이트인 '2ch' 의 '디지털 전시관' 에서는 지금 이 순간에도 수많은 아스키 아트가 활발하게 제작, 전시되고 있다(Ru, 2008). 또한, 텍스트만으로도 이미지를 표현할 수 있다는 장점 때문에 텍스트만 입력할 수 있는 게시판이나 댓글 등에서 아스키 아트를 활용하여 감정을 표현하거나 메시지를 전달하는 것이 일본 젊은이들 사이에서는 이미 하나의 문화(Katayama, 2006)로 자리 잡았다.

3. 한글의 아스키 아트

한글을 단순한 문자에서 탈피한 하나의 이미지 언어로서 다루고자 하는 움직임은 그동안 다양한 방식으로 이루어졌다. 가독성을 높이기 위한 활자 조판과 서체 디자인은 물론, 시각적으로 미려함을 주는 핸드 라이팅(hand writing), 캘리그래피(calligraphy)와 같은 글쓰기 작품에서부터, 아예 한글 자모음이나 한글 글자를 이미지처럼 활용하는 타이포그래피(typography)까지 실로 다양한 방식으로 한글의 이미지화가 시도되었다.

한글 낱자는 발음기관과 천지인의 형상을 따서 기본자를 만들고 단순한 기하학적 도형을 기본으로 수많은 글자를 표현하도록 만들어졌다(Kim, 2011). 그렇기 때문에 한글의 기하학적 이미지는 다양한 시각적 표현이 가능해 이미지를 글자로 표현해야 하는 아스키 아트에 매우 적합한 글자이다. 한글이 가지고 있는 기하학적인 특성은 점을 비롯한 기본 도형인 원, 사각형, 삼각형 구조를 바탕으로 하고 있는데, '·', 'ㅇ', 'ㅁ', 'ㅅ'이 그 대표적인 예이다. 단지 이 글자뿐만이 아니라 한글이 가지고 있는 기하학적 특징은 어떠한 이미지가 주어지더라도 쉽게 그 형태를 표현해 낼 수 있다.

아스키 아트는 각 글자들의 조합으로 전체 이미지를 만들어 낸다. 따라서 한 글자씩 사용되는 낱자 쓰기부터 각각의 글자를 모아쓰는 겹쳐 쓰기까지 지원하는 한글의 특

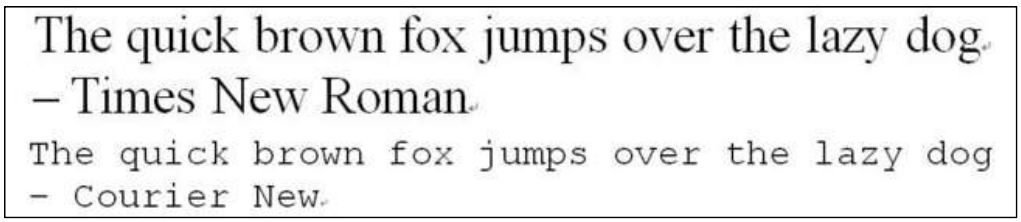
징은 다양한 명도차를 표현해 낼 수 있어 아스키 아트를 만들기에 좋은 특징을 가지고 있다. 일례로 한글은 가장 단순한 글자인 ‘ㆍ’ 글자부터 매우 복잡한 형태인 ‘뗇’과 같은 형태까지 다양한 글자를 지원하는데, 흔히 아스키 아트에서 사용되는 영문에 비해 한 단어 당 검정색 픽셀의 비율로 표현될 수 있는 명암 스펙트럼이 매우 넓다. 그렇기 때문에 한글을 아스키 아트에 사용 할 경우 풍부한 명암 표현이 가능하다.
그러나 아스키 아트로서의 한글은 여러 가지 난제를 가지고 있는데, 가장 큰 어려움 중의 하나는 한글과 기존 아스키 아트를 제작하던 문자들과의 인코딩 방식 차이점이다. 아스키 아트 제작에는 개별 글자의 조판과 글자와 글자사이의 공백 등이 규약 된 텍스트 인코딩 방식이 정확한 이미지를 제작하기 위한 핵심적인 요소이다. 아스키 아트가 많이 제작되는 영어권 텍스트 인코딩 방식은 1바이트 문자를 사용하나, 한글은 2바이트 문자 체계를 사용함으로써 기존에 제작된 아스키 아트 제작 방식이 제대로 동작하지 않거나 잘못된 아스키 아트를 제작하는 문제점이 있다. 일본어의 경우 1바이트와 2바이트가 혼용되는 독특한 체계를 가지고 있어 역시 한글 아스키 아트 제작에 동일한 방식을 적용하기 어렵다. 또한, 한글은 영어의 알파벳과 같이 적은 수의 자모음을 가지고 있으면서도 받침이 있는 한자문화권의 겹쳐 쓰기를 사용하는 문자 형태이므로, 자모음의 조합에 따라 매우 많은 숫자의 글자가 생성되며 그 개별 문자들이 정의된 글꼴(font) 파일을 아스키 아트 제작 시에 일일이 분석해야 하는 어려움이 따른다. 이로 인하여, 한글을 활용한 아스키 아트는 그 사례를 찾아보기 힘들고 관련 기술도 전무하다시피 한 상태이다.
그러나 앞서 살펴본 바와 같이 이미지 언어로서 한글 자체가 가진 예술성은 이미 다양한 방식으로 연구 및 창작이 이루어지고 있으며, 따라서 한글 아스키 아트가 가진 잠재력 또한 무궁무진하다 할 수 있다. 국가적으로도 한글의 우수성과 아름다움에 대한 홍보는 지속적으로 이루어지고 있으며, 이러한 흐름과 요구에 발맞춘 한글 아스키 아트의 제작은 젊은 감각과 함께하는 창의성의 발현, 한글로 이루어진 아름다운 예술작품으로 인한 국가 이미지 재고, 관련 기술 발전으로 인한 미래사업 창출 등 다양한 부가가치를 가진 유망한 기술 분야라 할 수 있다.

4. 한글 분석

4.1 한글 글꼴 파일 분석

한글 아스키 아트를 제작하기 위하여 우선적으로 수행해야 할 작업은, 아스키 아트를 제작할 글꼴을 선택하고, 그 글꼴 파일을 분석하는 일이다. 글꼴이란 폰트, 타입페이스(typeface) 혹은 서체(書體) 라고도 불리며, 일관성 있게 설계된 글자 모양을 이루는 하나의 집합(Kim, 2008)이다. 글꼴은 전통적인 활판인쇄 시대에서부터 심미적이고 가독성이 높은 문서를 제작하기 위하여 여러 종류로 개발되었으며, 컴퓨터를 이용한 문서작업이 보편화된 현대에 이르러서는 수많은 디지털 글꼴들이 제작되어 웹, 출판, 디자인 등 다양한 작업에 활용되고 있다(Lee, Hong & Sohn, 2003).
한글 아스키 아트 제작을 위하여 글꼴의 선택 및 분석은 매우 중요하다. 먼저 글꼴의 선택에 있어서 중요한 사항은 글꼴의 폭에 따른 분류이다. 글꼴은 글자간 폭을 다루

는 방식에 따라 크게 고정폭 글꼴과 가변폭 글꼴과 나뉘게 된다. 고정폭 글꼴은 글자의 종류나 조합에 관계없이 하나의 글자가 차지하는 공간이 항상 일정한 글꼴이고, 가변폭 글꼴은 글자의 종류 및 다음 글자와의 조합에 따라서 차지하는 공간이 달라지는 글꼴이다(Theodore, 2009). 고정폭 글꼴은 19세기경 타자기가 등장하였을 때 일정한 품질의 인쇄물을 보장하기 위한 손쉬운 방법으로 고안되었으며, 디지털 문서작업이 보편화된 현대에는 가독성 및 심미성이 더 높은 가변폭 글꼴이 더욱 각광받고 있다. 그러나 아스키 아트 제작을 위해서는 하나의 글자가 동일한 공간을 차지하는 고정폭 글꼴이 훨씬 유리하다. 따라서 우리는 한글 아스키 아트 제작을 위하여 고정폭 글꼴을 사용하였다.



〈figure 7〉 Example of proportional typeface, 'Times New Roman', and monospaced (non-proportional or fixed-width) typeface, 'Courier New'.

당연하게도, 동일한 글자를 표현함에 있어서도 글꼴이 달라지면 글자의 모양이 달라진다. 즉, 우리가 원하는 형태의 이미지를 제작하기 위해서는 글꼴이 달라지면 사용하는 글자가 달라진다. 이는 이미지의 어느 특정 부분을 표현하기 위하여 A라는 서체에서는 '가' 라는 글자가 더 적합한 글자일수 있지만, B 라는 서체에서는 '나' 라는 글자가 더 적합한 글자일 수 있는 것이다. 이것을 정확하게 구분하여 제작하기 위하여 글꼴 파일의 분석이 필요하다.

마이크로소프트에서는 폰트 파일(.ttf) 의 포맷 형식에 대해서 설명하는 문서를 무료로 배포[1]하고 있는데, 이를 통해 구체적인 폰트 파일의 역할 및 구조에 대해서 알아볼 수 있다. 위 문서에 따르면, 각 폰트가 글자를 표현하는 데에는 기본적으로 2차 또는 3차 베지어 곡선(Bézier Curve) 이 사용되는데, 폰트 파일에는 각각의 글자마다 이 베지어 곡선을 그리기 위한 조절점들의 위치가 정의되어 있다. 또한 각 폰트 파일마다 그 글꼴이 완성형 글꼴이냐, 조합형 글꼴이냐에 따라서 해당 폰트 파일의 인덱스(index) 에 서로 다른 글자가 배정되어 있다.

이렇게 서로 다른 폰트 파일의 폰트 분석을 위하여, 우리는 'Freetype[2]' 이라는 폰트 분석 오픈 소스 라이브러리를 사용하였다. 이 라이브러리를 통하여, 우리는 폰트 파일에서 특정 글자의 모양, 크기, 폰트 파일 안에서의 인덱스 등을 알아낼 수 있다. 따라서 특정 한글 폰트 파일 중에서 우리가 원하는 한글 자모음 및 한글 글자의 위치 및 모양을 분석함으로써, 쉽고 빠르게 자동으로 아스키 아트 이미지를 생성할 수 있었다.

4.2 현대 한글의 구성

한글 아스키 아트를 제작하기 위해서는 먼저 한글에 대한 이해 또한 선행되어야 한

1) http://www.microsoft.com/typography/otspec/default.htm
2) http://www.freetype.org/

다. 현대 한글 낱자는 닿소리(자음) 14자, 홀소리(모음) 15자, 겹닿소리(쌍자음) 5자, 겹홀소리(쌍모음) 11자, 겹받침 13자로 구성되어 있다(Hong, 2015). 〈table 1〉은 각 낱자들을 보여주고 있다.

〈table 1〉 The clusters of letters in modern Han-gul

현대 한글 낱자의 구성	
닿소리	ㄱ, ㄴ, ㄷ, ㄹ, ㅁ, ㅂ, ㅅ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ
홀소리	ㅏ, ㅑ, ㅓ, ㅕ, ㅗ, ㅛ, ㅜ, ㅠ, ㅡ, ㅣ
겹닿소리	ㄲ, ㄸ, ㅃ, ㅆ, ㅉ
겹홀소리	ㅐ, ㅒ, ㅔ, ㅖ, ㅘ, ㅙ, ㅚ, ㅝ, ㅞ, ㅟ, ㅢ
겹받침	ㄳ, ㅄ, ㄵ, ㄶ, ㄺ, ㄻ, ㄼ, ㄽ, ㄾ, ㄿ, ㅀ, ㅄ

현대 한글은 위의 낱자를 엮어 11,172(첫소리 19 × 가운뎃소리 21 × (끝소리 27 + 끝소리 없음 1)) 글자 마디를 쓸 수 있다. 11,172자 중 399자는 받침이 없는 글자이며 10,773자는 받침 글자이다. 사용 빈도는 KS X 1001 완성형 한글 코드에 선별된 2,350글자가 상위 99.9%로 알려져 있다.

4.3 한글의 전산화

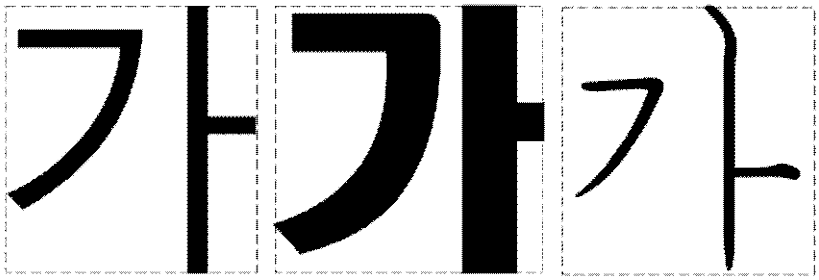
최초로 컴퓨터가 발명되고 발전되어 온 지역이 미국이기 때문에 컴퓨터에서 사용되는 언어는 알파벳을 기반으로 먼저 개발되었다. 이후 비슷한 문자 체계를 지닌 유럽의 언어들이 컴퓨터에서 처리될 수 있도록 개발되었으며, 한글의 전산화는 80년대부터 연구되기 시작하였다.

한글을 컴퓨터에 표시하는 방식으로의 한글 인코딩은 조합형과 완성형으로 나뉜다. 3.2장에서 언급한 것처럼 한글은 낱자들로 구성되기 때문에 초성, 중성, 종성을 독립된 문자로 보고 이들의 조합으로 글자를 표현하는 것이 조합형이며, '가'나 '각'처럼 한글의 한 글자를 독립된 문자로 인식하고 각 글자에 코드를 부여하는 것이 완성형이다. 초창기에는 조합형과 완성형 사이에 어느 방식을 사용할 것인지에 대한 논쟁이 있었지만, 1987년에 2바이트 완성형인 KS 완성형이 개발되고 보급화 되면서 2,350글자를 지원할 수 있는 현재의 KS X 1001 완성형이 자리 잡게 된다. Microsoft Windows 95에서 본격적으로 확장 완성형을 기본으로 지원하면서 11,172자를 처리할 수 있게 되었고 이후 조합형은 사라지게 되었다. 본 연구에서는 KS X 1001 완성형인 상위 99.9%로 선별된 2,350 자를 기반으로, 사용자가 원하는 한글 입력을 통하여 한글 아스키 아트를 제작할 수 있도록 하였다.

5. 한글 아스키 이미지 생성

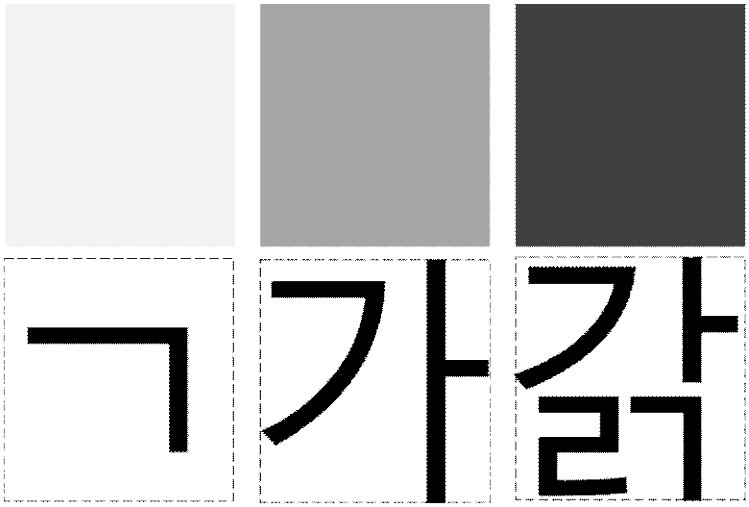
본 연구에서 제안된 한글 아스키 아트의 이미지 생성은 사용할 한글 글자 설정, 폰트 설정 및 분석, 입력된 한글의 글자별 밝기 계산, 입력된 이미지의 밝기 분석, 입력 이미지의 한글 아스키 아트 이미지로 변환의 단계를 거친다.

첫 번째 단계로, 먼저 어떠한 글자로 아스키 아트 이미지를 생성할 것인지를 결정해야 한다. 3.3 에서 언급했다시피, 우리는 사용 빈도가 높은 KS X 1001 완성형 한글 코드에 선별된 2,350 자의 한글 글자를 기반으로, 사용자가 원하는 글자로 아스키 아트를 제작할 수 있도록 하였다. 계산 과정의 설명을 위하여 우리는 사용자로부터 n 개의 한글을 입력받고, 이 n 개의 한글을 통하여 아스키 아트 이미지를 제작한다고 가정을 하도록 한다. 사용자로부터 n 개의 한글 입력을 받으면, 우리가 원하는 지정된 특정 폰트에서 n 개의 글자 리스트 [c_1, c_2, c_3, …, c_n] 가 가지는 각 글자별 글자 모양을 분석할 수 있다. 〈figure 8〉 은 동일한 글자임에도 일정 영역 안에서 글자의 검은 영역이 차지하는 비율이 달라짐을 보여주고 있다. 글꼴 파일(.ttc, ttf) 에는 서로 다른 글꼴에서 개별 글자마다 달라지는 이러한 글자의 모양 정보를 가지고 있으므로, 개별 글자의 밝고 어두운 정보를 분석한 뒤 그 명도차에 따라 입력 글자들을 정렬하는 과정을 수행하기 위하여 글꼴 파일의 분석이 필수적임을 보여주고 있다. 글꼴 파일의 분석은 4.1에서 구체적으로 다룬 바와 같이 Freetype 라이브러리를 활용한 폰트 분석 알고리즘을 구현하여 수행하였다.



〈figure 8〉 Various black proportion caused by font

이 분석을 통하여, 우리는 고정폭 글꼴의 글자 하나에 할당된 영역에서 글자 c_i 가 차지하는 검은색 영역과 흰색 영역을 구분할 수 있고, 이를 통하여 일정 영역에서 하나의 글자가 차지하는 검은색 영역의 비율을 통하여 각 글자별 명도를 분석할 수 있다. 쉬운 예로, ‘ㄱ’ 이라는 글자는 ‘갉’ 이라는 글자보다 더 밝은 글자가 될 것이다. 이 과정을 통하여 우리는 n 개의 글자를 각자의 명도 순서에 따라서 새롭게 정렬할 수 있게 된다. 이렇게 입력받은 글자가 명도 순서대로 재배열되면, 입력 이미지의 명도값에 따라 해당하는 글자 c_i 를 매핑시킴으로써, 최종적으로 전체 아스키 이미지를 생성할 수 있게 된다. 〈figure 9〉 는 명도에 따라서 입력된 글자가 어떻게 매핑될 수 있는지 간단한 예제를 보여주고 있다.



〈figure 9〉 Input images possessing different brightness and the individual result letters

본 연구는 구현을 위하여 스크립트 언어인 Python을 사용하였으며, 글꼴 분석을 위해 freeType 라이브러리와 이미지 처리를 위한 openCV라이브러리를 사용하였다. 〈table 2〉는 한글 아스키 아트의 이미지를 생성 방법의 전체적인 프로세스에 대한 의사 코드이며, 〈table 3〉은 한글의 글자별 밝기 계산 및 리스트 구성 알고리즘에 대한 의사코드이다.

〈table 2〉 Pseudocode of Han-gul ascii art

한글 아스키 아트의 의사코드	
def HangulAsciiArt(uKoreanChar, fontName, imgName):	uKoreanChar : 입력된 유니코드 한글 글자 리스트 fontName : 폰트 imgName : 입력 이미지 이름
fa = FontAnalyzer()	클래스 객체 선언
fa.init_setFont(fontName, 100*100)	폰트 지정 및 크기 설정
fa.init_setuKoreanCharacterList(uKoreanChar)	한글 글자 리스트 설정
fa.createCharIntensityValDicByChar()	글자별 밝기 계산
print AsciiArtImage	출력

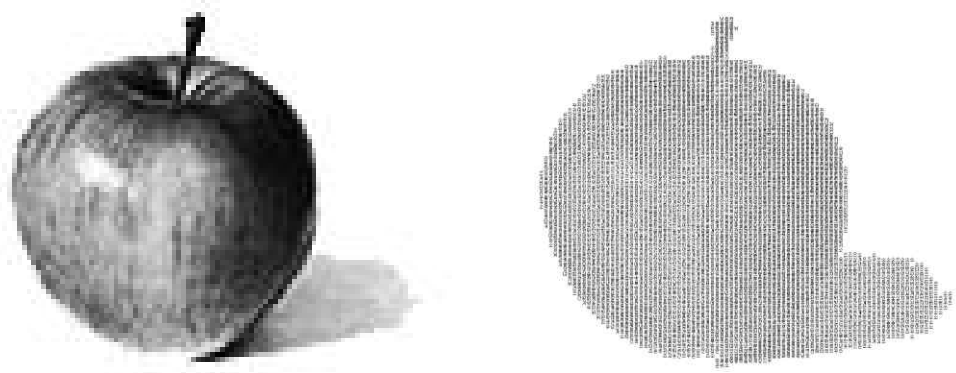
〈table 3〉 Pseudocode for computation of character intensity

한글의 글자별 밝기 계산 및 리스트 구성	
def createCharIntensityValDicByChar(self):	

for char in self.uKoreanCharList :
 	
bitmap = self.face.glyph.bitmap.buffer
 	
nonZeroBitMap = filter(lambda a: a!=0, bitmap)
 	
intensity = len(nonZeroBitMap)
 	
if len(self.orderedIdxBasedOnIntensity) == 0:
self.orderedIdxBasedOnIntensity.append([char, intensity])
 	
else:
tmpSize = len(self.orderedIdxBasedOnIntensity)
 	
for i, current_elem in enumerate(self.orderedIdxBasedOnIntensity):
current_intensity = current_elem[1]
 	
if intensity 〉 current_intensity :
 	
self.orderedIdxBasedOnIntensity.insert(self.orderedIdxBasedOnIntensity.index(current_elem), [char, intensity])
break
 	
if tmpSize == len(self.orderedIdxBasedOnIntensity):
self.orderedIdxBasedOnIntensity.append([char, intensity])
	
	font를 그리기 위한 정보를 buffer에서 추출
	
	nonZero인 bitmap 들만 nonZeroBitmap에 입력
	
	글자의 밝기 계산
	
	밝기가 0인 경우에 리스트에 저장
	
	밝기가 0이 아닌 경우에 임시 저장
	
	밝기 비교
	
	글자별 밝기에 따라 순차적으로 저장

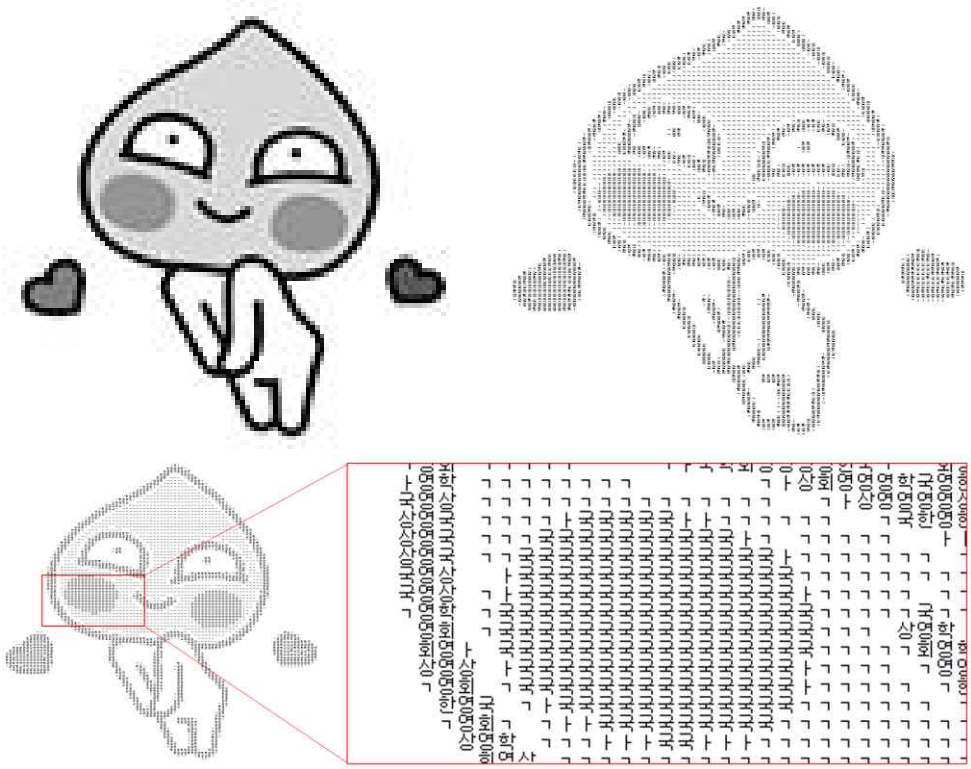
6. 결과 및 고찰

제안한 방법이 한글 아스키 아트 이미지를 성공적으로 생성함을 입증하기 위하여 본 연구에서는 3가지의 이미지를 제작하였다. 〈figure 10〉는 연필 소묘로 그린 사과 그림을 KS X 1001의 2,350 자를 모두 사용한 한글 아스키 아트로 변환한 결과 이미지이다. 해당 결과물은 소묘 작품에서 보여 지는 명암 단계가 아스키 이미지에서도 잘 표현되고 있음을 알 수 있다.



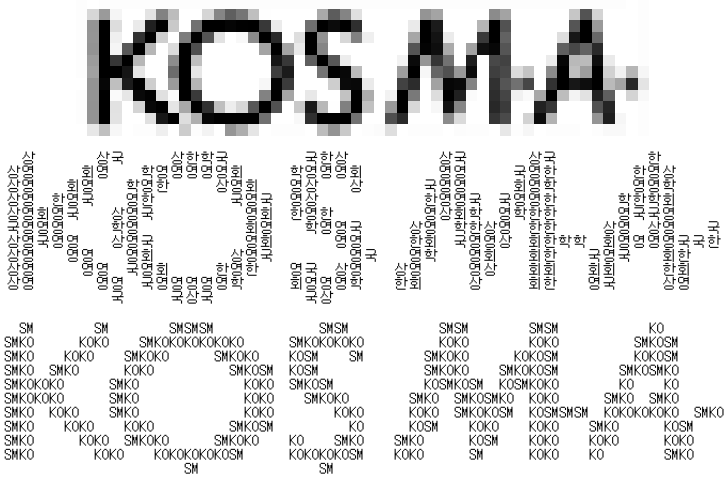
〈figure 10〉 Han-gul ascii art image of an apple drawing

〈figure 11〉은 사용자가 원하는 한글 입력을 통한 한글 아스키 아트 제작의 예제를 보여준다. 변환에 사용된 캐릭터는 모바일 어플리케이션인 카카오톡의 캐릭터 '어피치' 이고, 변환을 위해 입력된 글자는 '한국영상학회' 와 한글 자음 'ㄱ', 한글 모음 'ㅏ' 를 하나씩 사용하였다. 확대된 하단의 이미지를 통해 '한국영상학회' 및 한글 자음 'ㄱ', 한글 모음 'ㅏ'를 확인할 수 있으며, 이를 통하여 임의의 한글 입력을 통해서도 한글 아스키 아트가 효율적으로 생성됨을 볼 수 있었다. 또한, 이미지 픽셀의 명도에 따라서 입력된 한글이 적절하게 배분되고 있음을 확인할 수 있다.



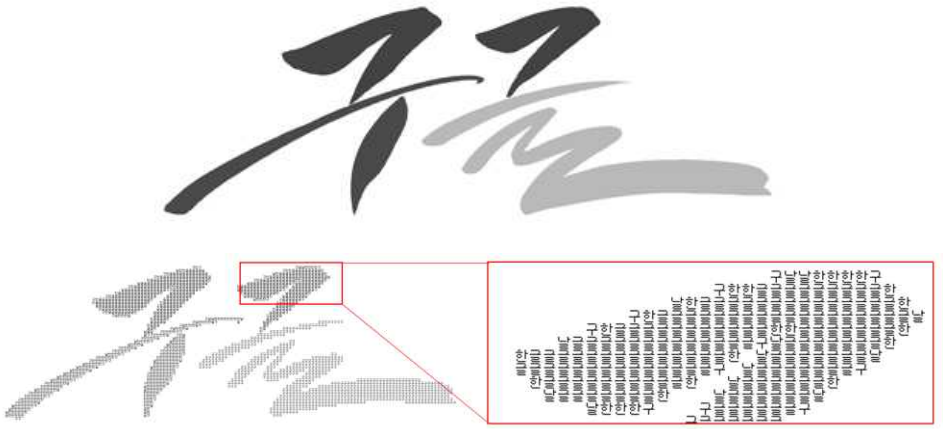
〈figure 11〉 Han-gul ascii art image of 'Kakaotalk' character

〈figure 12〉은 한국영상학회의 영문 로고에 우리의 아스키 아트 제작 알고리즘을 '한국영상학회' 라는 한글 글자와, 'KOSMA' 라는 영어 알파벳을 입력으로 받아 각각 적용해 본 결과물이다. 이를 통하여, 우리의 방법은 2바이트로 처리되는 한글 문자열뿐만 아니라, 기존의 1바이트로 처리되던 영문 문자열도 포함하는 통합된 솔루션을 제공하고 있음을 확인할 수 있다. 그림에서 확인할 수 있는 것은, 한글은 2바이트를 사용하고 영어 알파벳은 1바이트를 사용하므로, 한글 한 글자에 해당하는 공간을 영어 알파벳은 2글자를 사용하고 있음을 볼 수 있다. 기존의 영문 알파벳 기반 아스키 아트 이미지 제작 알고리즘은 한글 인코딩 방식 자체를 지원하지 않아 이미지가 생성이 되지 않거나, 이미지가 생성된다고 해도 2바이트 문자 체계로 인하여 이미지가 찌그러지거나 깨져 나오는 등 제대로 된 아스키 이미지를 생성할 수 없지만, 우리의 결과물은 정상적으로 한글 아스키 이미지를 생성할 수 있음을 보여준다.



〈figure 12〉 Han-gul and English alphabet ascii art images of 'KOSMA' logo

〈Figure 13〉 은 2015년 한글날의 구글 로고에 '한글날구글' 이라는 한글을 입력으로 받아 우리의 한글 아스키 이미지 제작 알고리즘을 적용한 결과물이다. 이렇게 우리의 방식은 기존의 영문 아스키 아트로는 그 가치와 의미를 충분히 전달하기 어려웠던 한국이나 한글의 홍보와 같은 다양한 콘텐츠에 대하여 한글 아스키 이미지 제작을 가능하게 함으로써, 한글의 브랜드화에 이바지하고 그 우수성과 예술적 가치를 드러내는 더욱 다양하고 풍부한 방식을 제공하는 데 그 의의가 있다.



〈figure 13〉 Han-gul and English alphabet ascii art images of 2015 Hangul Day 'Google' logo

다만 아쉬운 점은, 입력 이미지는 100x100px의 작은 사이즈 이미지라고 하더라도, 각 픽셀 당 하나의 한글 글자로 매핑 시킨다면 1만 글자로 나타내야 한다. 홈 비디오 규격인 640x480px 이미지만 하더라도 글자수가 30만개가 넘어가므로, 이를 그대로 아스키 아트로 변환하여 화면에 출력하거나 인쇄한다면 매우 거대한 크기가 될 것이다. 뿐만 아니라 연산에도 상당한 시간이 소모되게 된다. 따라서 차후에는 이 점을 해결할 수 있는 방법으로 연구를 진행할 계획이다.

7. 결론

한글은 글자 자체가 가진 기하학적 조형미가 뛰어나 아스키 아트에 매우 적합한 글자임에도 불구하고 그동안 아스키 아트 제작에 사용되지 않아 왔다. 또한 한글은 낱자를 사용하기도 하고 겹쳐 쓰기까지 지원하는 특성 때문에 다양한 명도 스펙트럼을 가지고 있어 깊이감 있는 아스키 아트를 생성해 낼 수 있으나 이 역시 활용되고 있지 않았다. 반면, 영문 아스키 아트는 지속적으로 연구 및 활용 되고 있으며 일본어 역시 많은 사람들의 사랑을 받으며 현재까지 활발하게 제작되고 있다.

본 연구에서는 사용자가 임의로 입력한 한글 글자를 기반으로 하여 다양한 이미지를 한글 아스키 아트로 자동 변환하는 방법을 제시하여 손쉽게 한글 아스키 아트를 제작할 수 있는 방법을 다루었다. 1바이트 문자로 이루어진 영문이나 전통적인 아스키 코드에 반해 한글은 2바이트 문자이므로 데이터 처리 및 조판 방식에 차이가 발생하기 때문에 기존에 사용되어 왔던 아스키 아트 생성 방법은 한글에 적용하기 힘든 한계를 가지고 있었다. 또한 한글은 한자와 같이 받침이 있는 모아쓰기 형태이므로 글자의 조합이 방대하고 상당히 많은 글자로 다루어야하기 때문에 이를 처리할 수 있는 한글만의 독특한 생성방식이 고안되어야 한다.

이를 위해 본 연구에서는 사용자가 아스키 아트를 제작하기 원하는 한글 글꼴 파일을 분석하여 글꼴마다 다양한 글자별 메타데이터를 추출한 후, 입력받은 이미지를 이미지 처리 과정을 통하여 아스키 아트 제작에 적합하도록 보정한 다음, 분석한 글꼴 정보와 비교하여 자동으로 아스키 아트 이미지를 생성해 내는 방법을 제안한다. 이 일련의 과정들로 이루어진 우리의 알고리즘은 다양한 사용자의 글자 및 이미지 입력에 대해서 아스키 아트 이미지를 쉽고 안정적으로 생성해 낼 수 있게 해 준다.

본 논문은 그동안 영문이나 전통적인 아스키 코드를 이용한 아스키 아트 생성 방법을 넘어서 우리의 우수한 한글에 적합한 아스키 아트 자동 생성 알고리즘을 제시했다는 데 의의가 있다. 여기서 제시된 알고리즘은 한글날과 같은 행사를 기념하는 전시 디자인에 적용하거나 웹이나 스마트 폰 등을 기반으로 배포하여 한글의 우수성을 입증할 수 있을 것이다. 본 연구자는 이 기술을 통하여 그동안 찾아보기 어려웠던 한글 아스키 아트 작품들이 더 많이 제작되고 생성되기를 기대해본다.

그러나 5장에서 언급했듯이 현 한글의 아스키 아트 제작 기술은 한 픽셀 당 명암 크기를 계산하여 한글을 대입하는 방식으로 입력 이미지의 사이즈가 크면 출력 아스키 아트 이미지도 너무 커지는데 한계가 있다. 따라서 각 픽셀에 하나의 글자를 사용하는 것은 바람직하지 못하다. 이를 위하여 본 연구자는 다음 연구로 입력 이미지를 몇 개의 pixel 을 하나로 Super-pixel 단위로 처리함으로써 보다 적은 수의 글자로도 아스키 이미지를 생성할 수 있도록 할 예정이다. 또한 비디오 시퀀스와 같은 애니메이션에도 본 연구의 방법을 적용함으로써 이미지뿐만 아니라 확장된 형태의 영상에도 적용할 수 있는 알고리즘을 개발하고자 한다. 이는 이후 더 나은 한글 아스키 아트의 결과물을 제시할 수 있을 것이다.

참고문헌

• *ASCII Art* (n.d.). Retrieved January 30, 2016, from Wikipedia website, https://en.wikipedia.org/wiki/ASCII_art.

• *ASCII Art* Image (n.d.). Retrieved January 30, 2016, from blog, http://alonestar.egloos.com/4664965

• D.I.G.I.T. (n.d.). Retrieved January 30, 2016, from D.I.G.I.T. website, http://labs.teehanlax.com/project/digit.

• *Emoticon* (n.d.). Retrieved January 30, 2016, from Wikipedia website, https://en.wikipedia.org/wiki/Emoticon.

• Hong, W. P., (2015). Hong, An Analysis on the Korean Language for Optimum Transmission of Hangul Code. *Proceedings of the Korea Contents Association Conference*, 10(7), 888-889.

• Kakaotalk Emoticon, (n.d.). Retrieved January 30, 2016, from apk-dl website, http://apk-dl.com/%EC%B9%B4%ED%86%A1-%EC%9D%B4%EB%AA%A8%ED%8B%B0%EC%BD%98/jacob.kakao.emoticon.

• Kalpana, C.. (2013). Automatic Ascii Art Conversion of Binary Images using NNF and Steganography. *International Journal of Scientific & Engineering Research*, 4(3), 22-26.

• Katayama, L., (2006). Art and ASCII: The Stories Behind All Those Brackets, Slashes, and Carets. Wired, Retrieved from http://www.wired.com/2008/06/mf-hiroyuki-ss/.

• Kim, H.. (2008). Korean Digital Typography. Sungshin Women's University Press, Seoul: Seoungbuk-Gu.

• Kim, Y. K.. (2011). Typography Media(Emoticon) for the Public Service and Communication. Journal of the Korea Contents Association, 11(6), 197-204.

• Lee, H. J., Hong, Y. M., & Sohn, E. M., (2003). A Study on Diversification of Hangul font classification system in digital environment. *Journal of Korean Society of Design Science,* 16(1), 5-14.

• Lee, J. J. & Park, J. W., (2007). A study on the realtime ascii art works using Processing language. *Proceedings of the Korea Contents Association Conference*, 24-28.

• Ru, L. S., (2008). The world of ASCII art, Aliceonnet, Retrieved from http://aliceon.tistory.com/758.

• Theodore, R.. (2009). The Typographic Desk Reference. Delaware: Oak Knoll Press.

• Xu, X., Zhang, L. & Wong, T.T., (2010). Structure-based ASCII Art. *ACM Transactions on Graphics (SIGGRAPH 2010 issue)*, 29(4), 52:1-52:9. doi:10.1145/1833349.1778789.

Submitted: 8 Jan, 2016
Sent for revision: 6 Feb, 2016
Accepted: 15 Feb, 2016