# 로봇을 이용한 벽화 시스템
## Wall Painting System using Robot

장호연*, 류승택**, 박진완*
중앙대학교 첨단영상대학원 영상학과*, 한신대학교 컴퓨터공학부**

Ho-Yeon Jang(hoyoun1227@hanmail.net)*, Seung-Taek Ryoo(stryoo@hs.ac.kr)**,
Jin-Wan Park(jinpark@cau.ac.kr)*

**요약**

본 논문은 사용자와 로봇의 통신을 통해 인간의 손을 대신하여 드로잉(Drawing)을 하는 시스템 설계 및 구현을 하였다. 컴퓨터 안에서 작업을 볼 수 있는 가상 환경과 실제 로봇(Robot)이 드로잉을 하는 현실 환경을 연동하기 위한 UI(User Interface)를 제공하여 사용자의 편의를 돕는 시스템을 제작하였다. 시스템의 UI는 넷빈(NetBean) 툴 안에서 AWT(Abstract Window Toolkit)를 이용하였고, 드로잉을 하기 위해서 벡터(Vector) 방식의 SVG(Scalable Vector Graphics) 파일을 기반으로 하여 이미지 정보를 얻어 표현하였다. 이 시스템은 실시간 통신으로 사용자 요구에 따라 원하는 이미지를 만들어 낼 수 있으며, 이미지 작업의 결과에만 그치는 것이 아닌 드로잉을 하는 과정에서 하나의 퍼포먼스(Performance)로 작용할 수가 있다. 본 연구는 사용자의 선택에 따라 실시간으로 작동하는 모습을 통하여 드로잉 하는 작업 과정 자체를 하나의 퍼포먼스로 볼 수 있도록 한다.

■ 중심어 : | NXT | 로봇 | 벽화 | SVG |

**Abstract**

In this paper, I explain the design and implementation of an automated drawing system by communicating with a user. I made a system providing an UI for virtual environmental actual robot to interwork with environment to make a drawing that I was able to read work, and to help convenience of a user in computers. The UI of a system was used the AWT(Abstract Window Toolkit) in NetBean tools. To draw images, I got image information from SVG vector files. This system can make images according to the demand of users by using real-time communication. Also this kind of drawing process can be an artists performance, not only the final image as a result of the work. So I suggest that this real-time drawing process which is operated by user's command can be considered as an artists performance.

■ keyword : | NXT | Robot | Wall Drawing | SVG |

## Ⅰ. 서론

급격한 산업화, 도시화 속에서 경제는 크게 발전되었고 편리한 생활을 누릴 수 있었다. 하지만 추구하고자 한 바람과는 달리 옛 생활 모습을 잃어버렸고 기존의 공간적 문화유산들을 사라지게 했다. 또한 자연 환경

* 본 연구는 2008년 정부(교육과학기술부)의 재원으로 한국학술진흥재단(KRF-2008-321-H00001) 및 2008년도 2단계 두뇌한국 (BK)21 사업에 의하여 지원되었음.

접수번호 : #090824-015
접수일자 : 2009년 08월 24일
심사완료일 : 2009년 09월 21일
교신저자 : 박진완, e-mail : jinpark@cau.ac.kr

파괴와 교통, 사회 등 수많은 문제를 야기하였다. 그로 인해 도시를 재조성하고 사회 문제를 해결하려는 도시 속 환경과 삶의 질을 높이려는 노력이 제기되었다. 그 중의 하나로 대형화 벽화를 예로 들 수가 있다. 벽화는 차가운 도시 이미지에서 따뜻한 예술적 모티브(motive)를 수용하는 의지로도 나타나고 있다. 이는 환경 조형물이 가진 기본 개념을 바탕으로 기존의 미술영역 안에서 관심 범위를 환경 보존과 자연 보호에 맞추고 자연 시각적, 경험적 체험을 통해 생활의 질적 향상을 도모하려는 시도라고 할 수 있다.

벽화는 현대 시대에서 단순한 그림이 아닌 미의 효과로서 환경 개선의 활력소가 되고 있다. 높은 빌딩, 넓은 벽의 그림을 로봇이 대신하여 그릴 수 있다면 사람이 하는 것보다 제작 시간과 비용을 절감할 수 있으며 사고의 위험을 줄일 수 있다. 또한 벽화라는 것이 하나의 예술 작품으로서 사람만이 할 수 있는 감정 표현을 사람이 직접하는 것이 아닌 로봇이 대신 하는 대에서 로봇도 사람과 같은 감정을 담아 표현을 할 수가 있다.

사용자의 입력에 의하여 상호 소통하는 자동 생성 드로잉이 가능한 시스템을 구축하기 위해서 사용자와 통신할 수 있는 자동 생성 드로잉 시스템 설계와 개발 과정에 대해 언급하고자 한다.

본 논문에서는 로봇을 이용하여 벽에 드로잉을 할 수 있는 시스템을 제안한다. 본 논문의 구성은 Ⅱ장 관련 연구에서 로봇을 이용한 드로잉 사례와 드로잉 및 벽화에 대하여 설명한다. Ⅲ장에서 드로잉을 하기 위한 전반적인 시스템을 설명하고, Ⅳ장에서는 로봇 드로잉의 전처리 부분으로, 웹캠(Web Cam)으로 영상을 받아 드로잉의 스타일을 결정하고 벡터 이미지인 SVG 파일로 변환하는 과정까지를 다루고자 한다. Ⅴ장에서는 로봇 드로잉의 구현 환경과 구현된 알고리즘을 설명하고, Ⅵ장에서 구현 결과에 대한 환경 시스템과 드로잉 환경을 설명하도록 하겠다. 마지막으로 Ⅶ장에서 연구 결과를 요약한다.

## Ⅱ. 관련연구

본 장에서는 드로잉 및 벽화에 대하여 간단하게 설명하고, 로봇을 이용한 드로잉 사례를 살펴보도록 한다.

### 1. 드로잉 및 벽화

일반적인 개념으로서의 드로잉은 선을 위주로 하여 그리는 행위로서 밑그림, 초벌그림의 의미를 지니며 구석기 시대의 동굴 벽화나 암각화에서부터 발견된다. 회화의 탄생 이전부터 존재한 드로잉은 아이디어를 머릿속에서 끄집어내는 과정에서부터 현상적으로 일어나는 상황 모든 것을 그려 내는 내면의 구도와도 같은 것이다.

벽화는 벽에 드로잉을 한 것으로 드로잉의 특성을 가지고 궁전, 사원, 교회, 동굴, 고분, 자연 또는 인공이 가해진 건조물의 내외 벽면, 천장, 기둥 등에 그린 그림이다. 벽화의 제작은 그 내외 공간의 장엄 및 장식을 위한 필수 요건이었다. 단순히 건물의 장식 미화에 그치는 것이 아니라 교리(敎理)상의 체계를 배경으로 성소를 장엄하고 신성한 분위기를 고조시키며 동시에 교화(敎化)의 역할도 겸하게 되는 것이다. 천정, 벽면, 기둥, 문 등에는 그에 알맞은 불교적인 소재의 그림을 그리게 된다. 즉 궁전이나 고급 건축물, 불전이나 신전 같은 종교 건축물, 무덤의 내외 벽면에 주로 벽화가 그려졌다. 이러한 벽화는 일찍부터 사람들의 존경과 사랑을 받는 현대 사회에서 빠질 수 없는 어디에서나 쉽게 찾아 볼 수 있는 당대(當代)의 최고급 미술로 자리를 굳히게 되었다[10].

### 2. 로봇을 이용한 드로잉 사례

Jurg Lehni와 Uli Franke는 헥터(Hektor)라는 직접 제작한 로봇을 이용하여 드로잉을 한 사례이다[11]. 이 작품은 일러스트 환경에서 추출한 벡터 값을 이용하여 경로를 미리 구하는 스크립트(Script) 알고리즘을 이용한 사례이다. 미리 지정된 경로를 따라 정확한 경로로 드로잉이 가능하지만 인터랙티브(Interactive) 환경을 목표로 하지 않는다. 드로잉의 재료로 스프레이를 이용하여 굵기 표현이 가능하고 선명할 수 있으나, 양 조절이 미흡하여 흘러내린다는 단점이 있다.

이미지를 정교하게 만들어 로봇을 이용하여 초상화를 그리는 사례이다[12]. 벽이 아닌 바닥에 드로잉을 하는 방식으로 되어 있다는 점에서 다른 사례들과 다르

다. PC에 붙어 있는 소프트웨어는 Claude Baumann와 그의 제자들의 아이디어를 가지고 MATLAB에서 개발되었다. 하드웨어는 레고(Lego)사(社) 마인드스톰(Mindstorm) NXT 로봇과 소프트 펜(Soft Pen)을 이용하였고, 2008년 Daniele Benedettelli에 의해 만들어졌다. 드로잉 전처리 과정을 하나의 시스템으로 만들어 간단하고 사용자에게 편리한 구조로 되어 있으나 결과 이미지는 정교하지 못하다.

위의 사례와 같은 레고 사 마인드스톰 NXT 로봇을 이용한 드로잉 사례가 있다[13]. 양쪽 모터에 줄을 연결하여 흔들림 방지로 추를 달아 모터를 감으며 드로잉을 하는 방식이다. 드로잉 재료로 펜을 사용하였는데 헥터기기의 스프레이를 이용한 사례에 비해 선명하지 못하고 깔끔하지 못하다.

[14]사례는 NVIDIA의 후원으로 진행된 GPU의 성능을 테스트하기 위한 데모이다. 다양한 색으로 1100개의 페인트 볼을 픽셀(Pixel) 단위로 데이터를 병렬 처리하여 벽에 모나리자를 찍어낸다. 다양한 색으로 한꺼번에 처리 된다는 점에서는 빠르고 간단한 시스템이지만 상세한 그림이라기보다 점묘화 느낌이 강하다.

본 논문에서는 위의 단점들을 개선하기 위한 시스템을 구현하여 굵기 표현과 보다 정교한 결과 이미지를 나타내고, 상호작용하는 환경에서 사용자가 원하는 이미지를 표현할 수 있는 시연 시스템을 구현하였다.

## Ⅲ. 로봇을 이용한 드로잉 시스템

본 장에서는 기존에 연구한 로봇을 이용한 드로잉의 문제점을 보완하여 개선된 시스템을 설계하고 이를 구현한다.

[그림 1]은 로봇을 이용한 드로잉 시스템으로써, UI는 넷빈 툴 안에서 AWT를 이용하였고, 가상 환경과 실제 드로잉 환경을 연동하였다[21]. 드로잉을 하기 위해서 이미지를 벡터 방식 SVG 파일을 기반으로 이미지 정보를 얻어 표현하였다[16]. 이 시스템은 블루투스(Bluetooth) 연동으로 실시간 통신이 가능하여 사용자의 요구에 따라 원하는 이미지를 만들어 낼 수 있다.

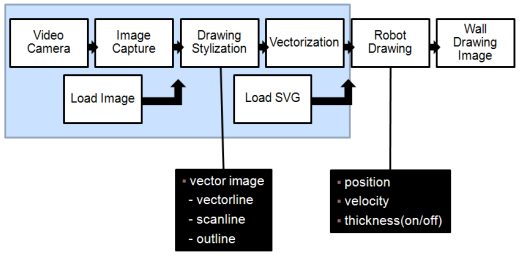


**그림 1. 로봇을 이용한 드로잉 시스템**

원하는 이미지를 불러오거나, 웹캠(Web Cam)을 통하여 이미지 캡쳐(Capture)를 받아 얻어올 수가 있다. 이미지를 받아오면 드로잉 스타일을 선택하게 되는데, 드로잉 스타일은 벡터라인(Vectorline), 스캔라인(Scanline), 아웃라인(Outline) 알고리즘에 따라 다양한 형태로의 드로잉이 가능하다. 선택된 스타일에 따라 벡터 이미지 SVG 형식으로 변환하게 되고, 변환된 파일 정보에 따라 위치, 빠르기, 굵기 등을 계산하여 로봇 드로잉을 하게 된다. 알고리즘에 따라 드로잉이 진행되어 처음 입력된 이미지를 전처리한 벡터 이미지와 일치하는 최종 이미지가 결과물로 생성된다. 드로잉의 재료로 힘 조절의 제약에 따르지 않고 선명하게 나타내 줄 수 있는 싸인펜를 이용하였다. 또한 붓펜을 이용하여 SVG 파일의 두께 정보에 따라 모터를 제어하여 선의 굵기를 표현할 수가 있다.

## Ⅳ. 로봇을 이용한 드로잉 전처리 과정

본 장에서는 로봇을 이용한 드로잉 시스템의 전처리 과정으로, 웹캠으로 이미지를 받아들여 드로잉 스타일을 결정하고 벡터 이미지로 변환하는 과정을 설명한다. 또 벡터 이미지 SVG 파일에 대해 간략하게 설명한다.

[그림 2]는 전처리 과정으로, 먼저 웹캠(Web Cam)을 이용하여 이미지를 받아오게 된다. 실행을 하면 캡쳐 시스템 프레임이 띄어지고 원하는 화면에서 캡쳐 버튼을 누르면 캡쳐된 이미지가 띄어져 창을 닫을 때 저장 여부에 따라 저장을 할 수가 있다. 웹캠을 이용하지 않고 원하는 이미지를 찾아 클릭하면 시스템 안에서 자동으로 이미지를 복사하여 저장하도록 구현되었다.

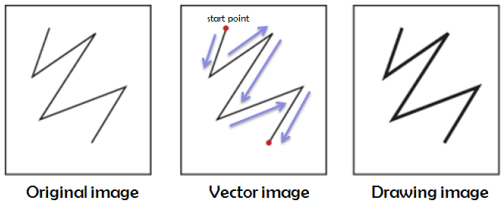


그림 2. 전처리 과정

얻어온 이미지를 이용하여 드로잉 스타일을 고르게 된다. 이 시스템에서는 다양한 스타일 중에서 벡터라인과 스캔라인, 아웃라인 방식으로 실험해 보았다. [그림 3]은 실험한 드로잉 스타일의 예이다.

이 시스템에서의 벡터라인 드로잉은 방향을 가지고 한 라인으로 드로잉을 하는 시스템이고 스캔라인 드로잉은 벡터라인 드로잉처럼 한 방향을 가지고 선으로 연결시켜 드로잉을 하는 시스템이다. 아웃라인 드로잉은 이미지의 밝기를 분석하여 두드러진 부분을 추출하여 드로잉을 하는 시스템이다.

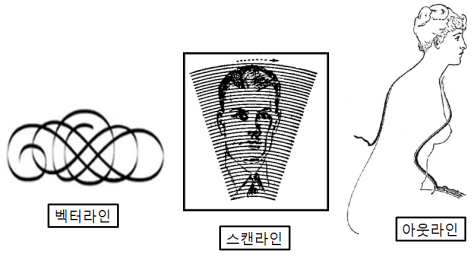


그림 3. 드로잉 스타일 예

선택된 이미지는 픽셀 단위로 저장되어 있기 때문에 이를 드로잉하기 위해서는 벡터 이미지로 변환시켜 주는 벡터화가 필요하다. 이를 위해 본 연구에서는 SVG를 사용하였다.

SVG란 2차원 벡터 그래픽을 표현하기 위한 XML 기반의 파일 포맷으로, 검색화, 목록화, 스크립트(Script)화가 가능하고 압축도 가능하다. 또한 XML 파일로 되어 있기 때문에 SVG 이미지는 어떤 문서 편집기로도 편집이 가능하다. 웹 브라우저(Web Browser) 상에서 열람할 수가 있고 하이퍼링크(Hyperlink)나 자바스크립트 등과도 연동시킬 수가 있고, 벡터 그래픽스로 확대나 축소를 해도 화질에 변화가 없는 언어이다[18].

## V. 로봇 드로잉

파일 정보에 따른 위치, 빠르기 굵기 등을 계산하여 로봇 드로잉을 하면 최종적으로 드로잉 된 아웃풋(Output)이 나오게 된다. 실제 벽화를 물리적으로 그리는 드로잉 피지컬 머신(Physical Machine)과 연산을 통해 복잡한 알고리즘을 수행하는 PC를 블루투스 방식의 통신으로 연결하여 드로잉을 수행하기 위한 시스템을 구축한다.

### 1. 로봇 드로잉 구현 환경

[그림 4]는 구현 시스템과 프로그램 간의 블루투스 연동으로 명령에 따라 드로잉을 하는 환경이다. 실제 환경에서는 로봇의 양쪽 모터가 방향에 따라 줄을 감으면서 좌표로 이동을 하게 구현 되었다.

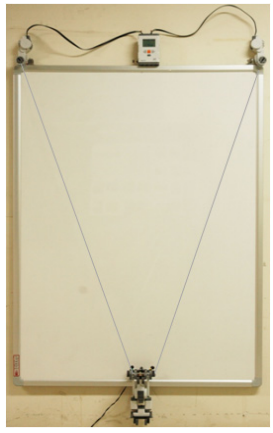

그림 4. 실제 환경

블루투스가 초기화 되고 PC와 로봇 시스템을 연결하게 된다. 프로그래밍을 한 시스템이 실행이 되며 저장된 SVG 파일을 불러들인 후 변환 알고리즘을 통해 문자열로 저장되어 있는 파일 정보가 변환되어 사이즈, 좌표 값 등의 정보를 추출하게 된다. 정보 값을 가지고 길이, 방향, 속도 알고리즘을 거쳐 NXT 시스템에게 명

령을 하게 되고 드로잉 작업을 마치면 NXT 시스템과 연결이 끊어지고 프로그램은 종료한다.

## 2. 드로잉 알고리즘

[표 1]은 값을 추출하기 위해 이미지를 SVG 파일로 변환하여 문자열로 저장되어 있는 정보를 스캔(Scan)하여 처음부터 끝까지 데이터 검색을 하게 된다. 데이터에서 이미지 크기 값, 색상 값, 좌표 값 등을 따로 추출하여 각각의 데이터를 하나의 수치 값으로 변환시키는 알고리즘이다.

**표 1. Algorithm SVG_Convert()알고리즘**

```
Algorithm SVG_Convert()
/* 문자열로 되어 있는 SVG 파일의 데이터를 수치 값으로 변환 */
{
    문자열로 저장되어 있는 정보 스캔;
    데이터 검색;
    이미지 크기, 색상, 좌표 값 등을 추출;
    각각의 값을 수치로 변환;
}
```

[표 2]는 SVG 변환 알고리즘에서 구한 좌표 값을 이용하여 거리를 계산해 주는 알고리즘으로, 기준으로부터 현재의 좌표 값에 따른 거리를 측정하여 현재 상태의 줄 길이와 다음으로 이동되는 줄 길이를 비교히여 각각의 길이를 구한다. 각각의 길이에 따른 카운트(Count) 값을 계산하게 된다.


**그림 5. 모터 이미지**

사용되는 NXT 시스템의 모터는 [그림 5]와 같은 1도 모터로써, 실제 드로잉을 하는 줄의 길이에서 모터가 1바퀴, 즉 360도를 회전했을 때의 줄의 길이가 86.4㎜가 나오게 된다. 1도에 따른 줄의 길이는 약 0.261㎜로써 이에 따라 정규화를 한 알고리즘 시스템이다.

**표 2. Algorithm Motor_Length()알고리즘**

```
Algorithm Motor_Length()
/* x, y 값으로부터 현재 값에 대한 거리와 다음 목표의 거리를 비교하
여 카운트 값을 측정 */
{
    현재 줄의 거리 – 다음 목표 줄의 길이;
    abs; // 절대 값
    차이 거리 값 * 0.261; // 1도에 0.261mm
}
```

[표 3]은 양쪽 각각의 모터 A와 C가 동시에 도착할 수 있도록 하는 알고리즘으로, 양쪽 모터에 최대 속도 값을 주고 회전수를 비교하여 회전이 많은 쪽은 최대 값을 유지하고 적은 쪽은 수정하여 속도 값을 지정해주는 알고리즘이다. 만약 회전수가 적은 모터의 속도가 최소 속도보다 작을 경우에는 최소 속도를 유지하고 회전수가 많은 모터의 속도를 두 모터의 비율에 따라 속도 값을 지정해 준다. 이는 최소 속도보다 적은 속도나 최대 속도보다 많은 속도로는 모터를 제어할 수 없기 때문이다.

**표 3. Algorithm Motor_Speed()알고리즘**

```
Algorithm Motor_Speed()
/* 양쪽 모터 A와 C가 동시에 도착할 수 있도록 각각의 속도 측정 */
{
    양쪽 모터에 최대 속도 적용;
    if 왼쪽 모터의 감을 정도 〈 오른쪽 모터의 감을 정도
        왼쪽 모터에 두 각의 비율을 곱한 속도 적용
    else
        오른쪽 모터에 두 각의 비율을 곱한 속도 적용
}
```

[표 4]는 모터의 방향을 측정하는 알고리즘이다. 최대 줄 길이에서 모터가 감긴 줄 길이를 뺀 값을 다음으로 이동해야 하는 줄 길이에서 빼면 양수나 음수로 값이 나오게 된다. 많이 감겨져 있을수록 줄의 길이는 짧고 조금 감겨져 있을수록 줄의 길이는 길다. 계산된 방향 측정 값이 음수일 경우 다음으로 이동할 줄 길이보다 전에 감은 줄이 더 길다는 것이고 양수일 경우에는 다음으로 이동할 줄 길이보다 전에 감은 줄이 더 짧다는 것이다. 따라서 음수일 때는 정방향인 상위 방향으로 줄을 감게 되고, 양수 일 때는 역방향인 아래 방향으로 줄을 풀어주는 방향 측정 알고리즘이다.

**표 4. Algorithm Motor_Direction()알고리즘**

```
Algorithm Motor_ Direction()
/* 모터의 방향을 측정 */
{
   if 현재 줄 길이와 다음 줄 길이 비교 = 양수;
      모터를 정방향으로 감는다;
   else if 현재 줄 길이와 다음 줄 길이 비교 = 음수;
      모터를 역방향으로 감는다;
}
```

[표 5]는 SVG 파일에서 추출한 펜의 두께 값에 따라 모터를 제어하는 알고리즘이다. 두께의 초기 값은 1로 표현되며, 그 이상의 값으로 제어가 가능하다. 측정된 현재 펜의 두께와 비교하여 다음 펜의 두께가 굵을 경우, 즉 두께 값이 클 경우에는 펜을 제어하는 중앙 모터를 정방향으로 감는다. 또, 현재 펜의 두께와 비교하여 다음 펜의 두께가 작을 경우에는 제어 모터를 역방향으로 감아 펜의 두께를 제어한다.

**표 5. Algorithm Stroke-Width()알고리즘**

```
Algorithm Stroke-Width()
/* SVG 파일 정보에 따른 펜의 두께 값에 따라 모터 제어 */
{
   두께의 초기 값 : stroke-width = "1"
   두께 값에 따라 중앙 모터 C를 제어하여 두께 표현.
   if 현재 두께 값 보다 다음 두께 값이 클 때 = forward();
      모터를 정방향으로 감는다;
   else if 현재 두께 값 보다 다음 두께 값이 작을 때 = backward();
      모터를 역방향으로 감는다;
}
```

아래 [그림 6]은 이미지 정보 값에 따라 두께를 나타내기 위해 펜을 제어하는 시스템의 한 부분이다.

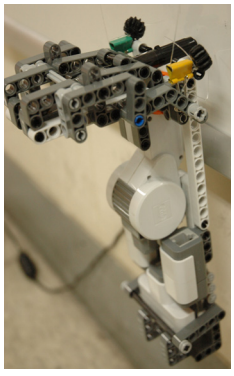

**그림 6. 펜 제어 시스템**

## Ⅵ. 구현 결과

### 1. 시스템 구현 환경

로봇을 이용한 드로잉 시스템은 다음과 같은 환경 하에서 개발되었다.

**표 6. 로봇을 이용한 드로잉 시스템 구조**

| 분류 | 종류 |
|---|---|
| 개발 운영체제 | 윈도우즈 XP Professional SP2 |
| 개발 도구 | NetBean 6.15, JDK 1.5 |
| 개발 언어 | JAVA, Processing, AWT, Icommend6.0 |
| 기타 H/W | Webcam, Bluetooth, Mindstorm NXT |

### 1. 로봇 드로잉 시스템 UI 환경 구현

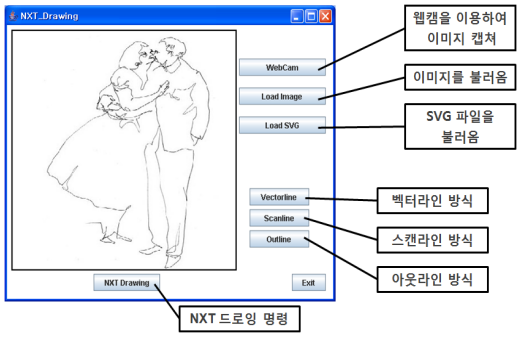


**그림 7. 로봇을 이용한 드로잉 시스템 UI**

로봇을 이용한 드로잉 시스템의 UI 메뉴는 다음과 같다.

① 웹캠을 이용하여 이미지를 저장할 수가 있다.
② 이미지를 불러올 수가 있다.
③ SVG 파일을 불러올 수가 있다.
④ 스타일에 따라 드로잉을 할 수 있도록 한다.
⑤ 로봇이 실제적으로 드로잉을 하도록 명령한다.
⑥ 종료.

### 3. 로봇 드로잉 응용 결과

로봇을 이용하여 전처리 과정을 통하여 벡터라인과 스캔라인, 아웃라인 스타일로 드로잉을 작업을 하였다. 아래 [그림 8]은 벡터라인 방식으로 드로잉 한 결과이다.

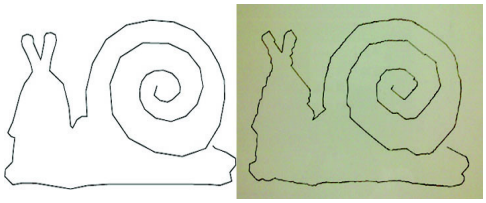

그림 8. 벡터라인 방식 원본 이미지 & 드로잉 결과

아래 [그림 9]는 스캔라인 방식으로 드로잉 한 결과이다.

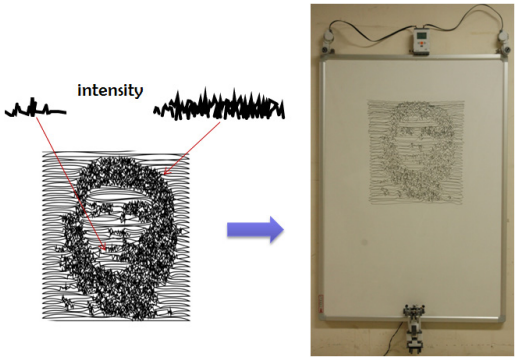


그림 9. 스캔라인 방식 원본 이미지 & 드로잉 결과

아래 [그림 10]은 아웃라인 방식으로 드로잉 한 결과이다.

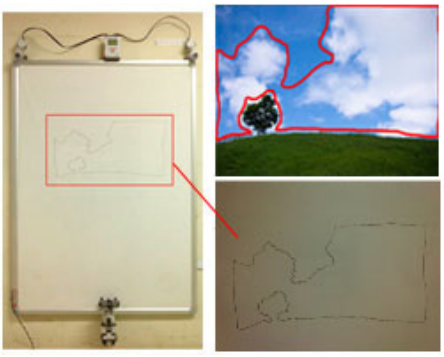

그림 10. 아웃라인 방식 원본 이미지 & 드로잉 결과

## Ⅶ. 결론 및 향후 연구

이미지를 벡터 이미지로 변환한 SVG 파일을 이용하는 부분은 수동적인 시스템이었다. 이를 드로잉 과정에서 한 번에 처리될 수 있도록 전처리 과정을 삽입하여 자동화 시스템을 구현하였다. 컴퓨터 상에서의 작업과 실제 드로잉 환경을 연동해주는 UI 메뉴로 사용자의 편의를 돕고 전처리 과정과 드로잉 과정을 연동하는 시스템으로 구현하였다.

원하는 이미지를 불러올 수가 있고 실제 환경에서 웹캠을 이용하여 원하는 모습을 이미지로 저장이 가능하다. 이러한 이미지를 UI 메뉴를 통해 벡터 이미지인 SVG 파일로 변환이 가능하며 원하는 벡터 이미지를 불러올 수도 있다. 이미지에 따른 드로잉 할 스타일을 선택하면 알고리즘에 따라 원본 이미지가 전처리 과정을 거쳐 벡터 이미지와 일치하는 최종 이미지가 결과물로 생성된다.

또한, 이 작업은 사용자가 원하는 이미지와 원하는 스타일로 사용자와의 통신을 통하여 실시간으로 진행되는 드로잉 과정 자체를 하나의 퍼포먼스로 볼 수가 있겠다. 그러나 NXT 로봇에서 지원하는 모터로 드로잉 하는 작업 과정에서 불안정함과 속도에 따른 모터 제어가 힘든 단점이 있다. 모터 제어가 섬세하게 가능하도록 해야겠다.

추후에는 단순히 캡쳐된 이미지뿐만 아니라 전통적인 애니메이션 기법인 페인팅 온 글래스(Painting on glass)를 연상시키듯 두 이미지의 차영상을 계산하여 변화된 부분만을 그리는 새로운 스타일의 드로잉이 가능할 것이다. 또한, NXT 로봇 하나만으로 드로잉 작업을 하는 것뿐만 아니라 로봇 하나에 각각의 로봇 3개를 NXJ 라이브러리(Library)로 연동시켜 하나의 로봇으로 동시에 3개의 로봇을 제어하여 각각의 다른 공간에 다른 색으로 드로잉 하는 작업이 가능할 것이다.

## 참 고 문 헌

[1] 이기석, “환경디자인으로서 도자조형물의 시도”,

한국디자인학회, 통권, 제22호, pp.467-472, 1997.
[2] 서동수, “피지컬컴퓨팅의 개념과 기술적 기초”, 한국디자인학회 가을 학술발표대회논문집, pp.270-271, 2006.
[3] 장호연, 류승택, 박진완, “NXT 로봇을 이용한 SVG 기반 실시간 드로잉”, 한국콘텐츠학회 춘계 종합학술대회, 제7권, 제1호, pp.146-151, 2009.
[4] 배희재, 송병규, 김윤기, 김창수, 정회경, “XML 기반의 구조화된 그래픽 표현을 위한 SVG 문서 저작 시스템”, 한국정보과학회 봄 학술발표논문집, 제31권, 제1호, pp.817-819, 2004.
[5] 문외식, 유승한, 성영훈, *LEGO MINDSTORMS NXT로 로봇만들기*, ALCO, 2008.
[6] 케이시 시에라, 버트 베이츠, 서환수, *Head First Java(뇌 회로를 자극하는 자바 학습법)*, 한빛미디어, 2005.
[7] Dan O'Sullivan, Tom Igoe, *Physical Computin g (Sensing and Controlling the Physical World with Computers)*, Course Technology Ptr, 200 4.
[8] Alan Watt, *3D Computer Graphics*, ADDiSON-WESLEY, 1999.
[9] John C. *Hansen, Lego Mindstorms NXT Powe r Programming: Robotics i n C(Robotics in C)*, Variant Press, 2007.
[10] http://blog.daum.net/doyota91/12645912
[11] http://www.hektor.ch/
[12] http://robotics.benedettelli.com/portrayer.htm
[13] http://www.youtube.com/watch?v=WTGHyOt2Ekk
[14] http://www.youtube.com/watch?v=fKK933KK6Gg
[15] http://mindstorms.lego.com
[16] http://www.w3.org/TR/SVG
[17] http://100.naver.com/
[18] http://www.wikipedia.org/
[19] http://cafe.naver.com/javachobostudy.cafe
[20] http://cafe.naver.com/javacircle.cafe
[21] http://cafe.naver.com/netbean.cafe

## 저 자 소 개

**장 호 연(Ho-Yeon Jang)** 정회원

- 2008년 2월 : 한신대학교 소프트웨어학과 이학사
- 2008년 2월 ~ 현재 : 중앙대학교 첨단영상대학원 영상학과(석사재학 중)

<관심분야> : 인터랙티브 미디어, 게임

**류 승 택(Seung-Taek Ryoo)** 정회원

- 1996년 2월 : 중앙대학교 전자계산학과 공학사
- 1998년 2월 : 중앙대학교 컴퓨터공학과 공학석사
- 2002년 8월 : 중앙대학교 영상공학과 컴퓨터그래픽스전공 공학박사
- 2002년 9월 : 중앙대학교 정보통신 연구원 연구 전담 교수
- 2004년 3월 ~ 현재 : 한신대학교 컴퓨터공학부 부교수

<관심분야> : Image Based Rendering, Realtime Rendering, Non-Photorealistic Rendering

**박 진 완(Jin-Wan Park)** 정회원

- 1995년 : 중앙대학교 컴퓨터 공학과 졸업
- 1998년 : Pratt CGIM(Computer Media)1995-1998. MFA
- 2003년 : CWI, Art Director/General ManagerGraphics & Inter active
- 2003년 : Pratt Institute CGIM Instructor
- 2003년 : VR and Game Graphics
- 2003년 3월 ~ 현재 : 중앙대학교 첨단영상대학원 교수

<관심분야> : Digital Animation, NPR, Digital Art