NXT 로봇을 이용한 SVG 기반 실시간 드로잉

SVG Based Realtime Drawing Using NXT Robot

장호연, 류승택*, 박진완
중앙대학교, 한신대학교*, 중앙대학교

Jang ho-yeon, Ryoo seung-taek*, Park jin-wan
CHUNG-ANG Univ., HANSHIN Univ.*,
CHUNG-ANG Univ.

요약

현대 설치 예술 분야에서 피지컬 컴퓨팅(physical computing)을 이용한 작품의 사례가 많아지고 있다. 하지만 인터랙션(interaction)을 위한 로봇 사용이 아닌 인터페이스(interface)가 장착된 드로잉(drawing)의 도구로 로봇을 사용하여 작업한 예는 쉽게 찾아 볼 수가 없다. 본 논문에서는 사용자와 통신할 수 있는 드로잉 작업용 시스템 설계와 개발 과정에 대해 언급하고자 한다. 작업 환경을 구성하는 로봇으로는 레고(Lego) 社에서 나온 마인드스톰(Mindstorm) 지능형 로봇 NXT 시스템을 이용하였고, 현실에서의 실제 드로잉 환경과 이를 예측하여 운동을 시뮬레이트(simulate)하는 가상 환경으로 구분하였다. 실제 환경에서 드로잉을 하기 위하여 NXT 시스템을 제어할 수 있도록 하는 아이커맨드(Icommand) 라이브러리(library)를 이용하였고, 가상 환경을 표현하기 위하여 이미지 표현이 쉬운 프로세싱(processing) 라이브러리를 이용하였다. 라인(line) 드로잉을 위하여 벡터(vector) 방식 SVG(Scalable Vector Graphics) 파일을 기반으로 이미지 정보를 얻어 표현하였다. 이 시스템은 블루투스(bluetooth) 연동으로 실시간 통신이 가능하여 사용자의 요구에 따라 원하는 이미지를 만들어 낼 수 있다. 이러한 모습은 이미지의 결과에 그치는 것이 아니라 드로잉을 하는 과정에서 하나의 퍼포먼스(performance)로 작용할 수가 있다.

Abstract

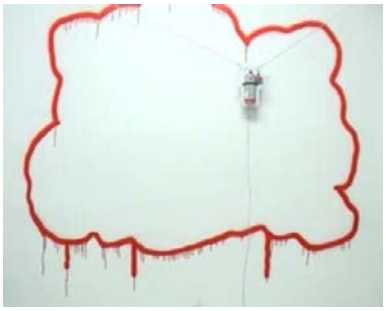
A reward of the work which used physical computing in fields artistic modern installation is becoming large. But the example that used a robot cannot try to easily look it up on a tool of the drawing which isn't robot use for interaction. I will mention it about a user and a drawing system design and the developmental process which I can communicate with at these papers. I used a Mindstorm NXT system to have been said in Lego companies in the robot which composed working environment, and I classified it to virtual environments that I forecasted environmental actual drawing this, and I did simulate. I used an Icommand library to control a NXT system, and drawing used a processing library in actual environments for the purpose of in order to to express to virtual environment. I got image information, and I expressed to a vector method SVG file to bases for line drawing. This system can have already made it according to demand of a user as real-time communication is possible by bluetooth working together. These figures can work to one performance not being stopping to the results of already in processes doing drawing.

Ⅰ. 서론

최근 미디어 아트에서 피지컬 컴퓨팅(physical compu ting)[1] 기술을 이용한 인터페이스 작품의 사례가 다양하게 발표되고 있다. 이러한 작품에는 주변 센서(sensor)를 구동할 수 있는 입출력 인터페이스 보드를 사용하는데, 이 보드는 블루투스, 무선전신(wireless), USB, 시리얼 통신, 1394 등 각각의 사용자 환경에 맞는 시스템으로 구성된다[3]. 또한 센서를 제어하는 부분에서 독립적인 지능형 시스템이 아닌 사용자의 입력에 의하여 상호소통하며 제어하는 시스템은 쉽게 찾아 볼 수가 없으며, 이러한 시스템을 적용하여 자동생성 드로잉이 가능한 시스템을 구축하려고 한다. 이러한 시스템을 구축하기 위해서 사용자와 통신할 수 있는 자동생성 드로잉 시스템 설계와 개발 과정에 대해 언급하고자 한다.

Ⅱ. 관련 연구

로봇을 이용한 드로잉 작업 중에는 여러 작업이 있다. 그 중에 피지컬 컴퓨팅[2]을 이용한 로봇으로 드로잉 작업을 하는 작품들을 드로잉에 쓰인 재료에 따라 분리하여 살펴보겠다.



▶▶ 그림 1. Jurg Lehni & Uli Franke - In a Beautiful Place Out in the Country

첫 번째, 직접 제작한 헥터(hektor)[6] 기기로 스프레이를 사용하여 작업을 한 작품이 있다. 이 작품은 일러스트 환경에서 추출한 벡터 값을 이용하여 경로를 미리 구해주는 스크립트(script) 알고리즘(Algorithm)을 이용하여 구현한 작품으로써, 간단한 구조로 비교적 정확하게 그려지나 드로잉의 재료가 스프레이라는 점에서 벽에 그림을 그릴 때 분사되는 양의 조절이 미흡하여 흘러 내림과 번짐이 있다. 또한 미리 지정된 경로를 따라 선을 그리는 방식으로, 먼저 입력을 받고 작동되는 환경으로 실시간적인 응용이 불가능하다.

▶▶ 그림 2. Adam & Jamie[7] - GPU와 CPU의 성능 비교

두 번째, NVIDIA의 후원으로 CPU와 GPU 성능을 비교하기 위해 페인트 볼을 이용한 작업이 있다. 이는 단순히 평행하게 한 가지 색의 볼을 던져 결과를 내는 CPU와 1100개의 여러 가지 색 페인트 볼들을 한꺼번에 던져 결과를 내는 GPU를 예로 들어 설명하는 작업이다. 취지는 CPU와 GPU의 차이를 비교한 것이지만 예술적 측면으로 기술과 접목하여 만들어낸 작품으로 볼 수가 있다. 색이 다양하고 한꺼번에 처리 된다는 점에서는 빠르고 간단한 시스템이지만 페인트볼을 사용함으로써 상세한 그림이라기보다 크기가 큰 추상적인 점묘화의 느낌이 강하고 로봇의 크기가 크기 때문에 다른 곳으로 이동하는 작업에는 다소 어렵다.

▶▶ 그림 3. Pindar Van Arman[8] - 붓을 이용한 드로잉

세 번째, 원본의 대상으로부터 색상을 추출하여 붓으로 이미지를 제작하는 시스템이다.

첫 번째에 설명한 제작한 기기로 스프레이를 사용한 작품에 비해 다양한 색상과 유화 느낌을 표현할 수가 있다. 하지만 붓을 이용하여 단순히 찍는 작업으로는 정밀 이미지의 표현이 힘들고 무엇보다 12시간에서 최고 36시간까지 오랜 시간에 걸쳐 이미지가 완성된다는 단점이 있다.

Ⅲ. NXT 시스템을 이용한 SVG 기반 실시간 드로잉

1. 로봇 프로그래밍 개발 환경

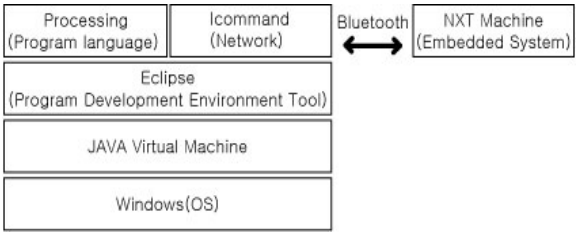
로봇 프로그래밍 환경은 소프트웨어를 보다 쉽고, 지원되는 라이브러리를 가져와 다양한 언어로 프로그래밍이 가능하게 하는 이클립스(Eclipse)[9] 개발 툴 안에서 이루어졌다. 이는 자바 기반의 프로그램으로 JDK(Java Virtual Machine) [10]를 설치하여 자바 개발 환경 안에서 이루어지도록 셋팅하였다.

▶▶ 그림 4. 로봇 프로그래밍 개발 환경

여기에 애플릿 환경에 보다 쉽게 접근 가능한 프로세싱[11]과 아이커맨드[13]라는 비교적 간단하게 구현할 수 있도록 정리가 잘 되어 있는 라이브러리를 이용하였다. 아이커맨드는 레고 사에서 나온 마인드스톰 지능형 로봇으로 프로그램을 구현하며, 모터와 센서를 제어 가능하게 하는 임베디드 시스템(Embedded System) NXT[16] 시스템으로써 자바 언어로 작성된 블루투스 기반의 NXT 시스템 표준 명령 라이브러리이다. NXT 시스템은 프로그램을 구현하고 주변 센서를 컨트롤 하는 환경의 표준화가 적절하게 이루어져 있다. 또한 제공되는 라이브러리가 효율적으로 정비되어 있어 프로그래밍의 접근이 용이하다. NXT 시스템을 제어하기 위한 아이커맨드 라이브러리를 사용하려면, 자바 API로 지원되는 블루투스 라이브러리 bluecove.jar 파일과 bluecove를 이용하여 NXT 시스템에 표준 명령을 전달하기 위한 API를 제공하는 Icommand.jar 파일이 필요하다.

NXT 시스템의 센서에는 터치, 사운드, 라이트, 거리 센서가 있고 모터의 회전수, 회전 각도를 제어할 수 있으며 블루투스 연동이 가능하다. 또한 조립 부품들은 각 세트로 나누어져 목적에 맞게 다양한 형태로 제작이 가능하다. 플라스틱의 레고 블록 형식이기 때문에 장난감처럼 느껴질 수 있으나 컨트롤 하는 데 있어 어려운 다른 기기들보다는 프로토타입에 가깝더라도 간단한 구조로 쉽게 접근 가능한 좋은 프로그램이 될 수가 있다.

이러한 임베디드 시스템 NXT 시스템과 개발 환경 사이의 시리얼 포트(serial profile)를 지원하는 USB 블루투스 동글(Dongle)을 이용하여 연결이 가능하다. 이 개발 환경에서는 NXT 시스템 사이의 시리얼 포트를 지원하는 ASUS 블루투스 동글로 실험을 하였다.

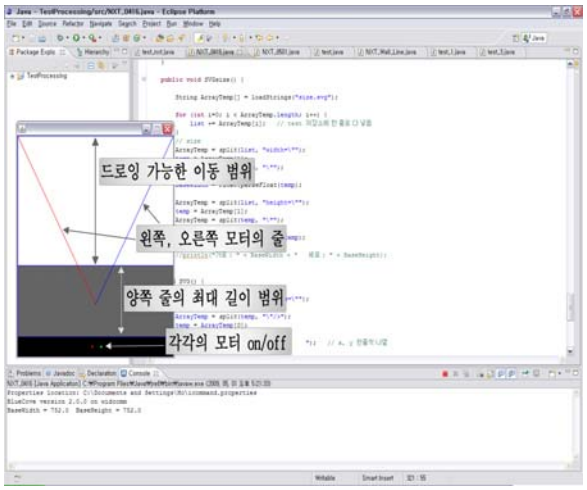


▶▶ 그림 5. 로봇 프로그래밍 개발 구조[4]

2. 실험 환경

2.1 가상 환경

현실에서의 구축된 운영 환경을 입력하고 이를 예측하여 운동을 시뮬레이트(Simulate)하는 환경이다.

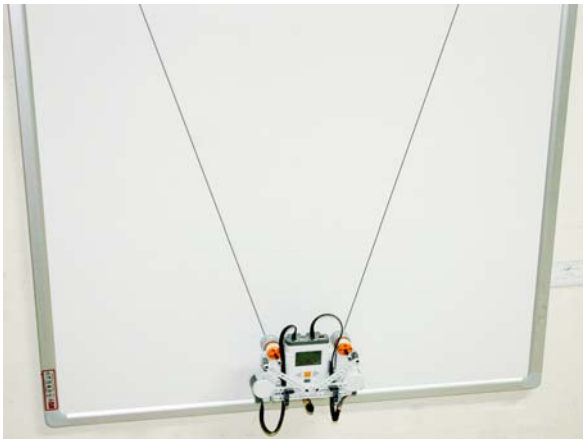


▶▶ 그림 6. 가상 환경

위에 보이는 그림과 같이 왼쪽의 줄과 오른쪽의 줄은 실제 환경에서 양쪽 각각의 줄을 의미한다. 진회색 범위는 드로잉이 가능한 흰색 범위 안에서 줄의 최대 길이 범위이다. 그 아래 검은 범위는 현재 상태를 알려주기 위한 공간으로써 각각의 네모는 줄의 길이가 변화할 때마다 즉, 모터가 감기거나 풀리고 있을 때 움직임을 나타내기 위한 상태 표시이다. 움직일 때는 네모에 불이 들어오게 되고 움직이지 않을 때에는 불이 들어오지 않게 된다.

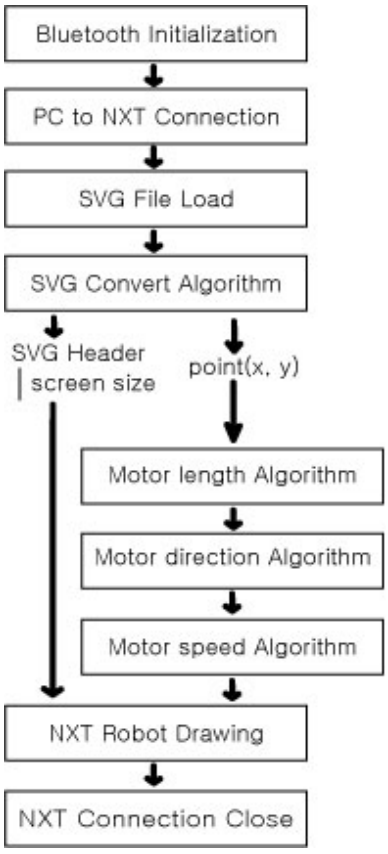
2.2 실제 환경

직접적으로 구현한 시스템과 블루투스와의 연동으로 명령에 따라 드로잉하는 환경이다.



▶▶ 그림 7. 실제 환경

실제 환경에서는 800*1200 보드에 NXT 시스템의 양쪽 모터가 반응하여 그에 따라 줄을 감았다 풀었다 하며 좌표로 이동을 하게 된다.



▶▶ 그림 8. 개발 환경 시스템

블루투스가 초기화 되고 PC와 NXT 시스템을 연결하게 된다. 프로그래밍한 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 명령을 하게 되고 드로잉 작업을 마치면 NXT 시스템과 연결이 끊어지고 프로그램은 종료한다.

3. 드로잉 알고리즘

3.1 SVG 데이터 추출

1) SVG 변환 알고리즘(SVG Convert Algorithm)

SVG[5]란 2차원 벡터 그래픽을 기술하기 위한 XML 마크업 언어로 웹 브라우저 상에서 열람할 수가 있고 텍스트 편집이 가능하며, 하이퍼링크나 자바스크립트 등과도 연동시킬 수가 있고, 벡터 그래픽스로 확대나 축소를 해도 화질에 변화가 없는 언어이다. 값을 추출하기 위해 이미지를 SVG 파일로 변환[17]하여 문자열로 저장되어 있는 정보를 스캔하여 처음부터 끝까지 데이터 검색을 하게 된다. 데이터에서 이미지 크기 값, 색상 값, 좌표 값 등을 따로 추출하여 각각의 데이터를 하나의 수치 값으로 변환시키는 알고리즘이다.

3.2 NXT 시스템의 모터 구동

1) 모터 길이 측정 알고리즘
(Motor Length Algorithm)

SVG 변환 알고리즘에서 구한 X와 y의 좌표값을 이용하여 기준으로부터 거리를 측정하여 현재의 x와 y의 좌표값에 따른 거리를 계산해 주는 알고리즘이다. 현재 상태의 줄의 길이와 다음으로 이동되는 줄의 길이를 비교하여 길이를 구하고, 그에 따른 카운트 값을 계산하게 된다.

사용되는 NXT 시스템의 모터는 1도 모터로써, 실제 드로잉을 하는 줄의 길이에서 모터가 1바퀴, 즉 360도를 회전했을 때의 줄의 길이가 86.4㎜가 나오게 된다. 이에 따라 1도에 따른 줄의 길이는 약 0.24㎜로써 이에 따라 정규화를 한 알고리즘 시스템이다.

$$L(X) = (X` - X) * 0.24 \qquad (1)$$

$$360^{\circ} : 86.4\text{㎜} = 1^{\circ} : X\text{㎜} \qquad (2)$$

2) 모터 방향 측정 알고리즘
(Motor Direction Algorithm)

최대 줄 길이에서 모터가 감긴 줄 길이를 뺀 값을 다음으로 이동해야 하는 줄 길이에서 빼면 양수나 음수로 값이 나오게 된다. 많이 감아져 있을수록 줄의 길이는 짧고 조금 감아져 있을수록 줄의 길이는 길다. 계산된 방향 측정값이 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 때 모터는 정방향의 상위 방향으로 줄을 감게 되고, 양수 일 때 모터는 역방향의 아래 방향으로 줄을 풀어주게 되는 모터의 방향을 측정해주는 알고리즘이다.

$$Direction(X) = L_{next} - (L_{MAX} - Count * 0.24) \qquad (3)$$

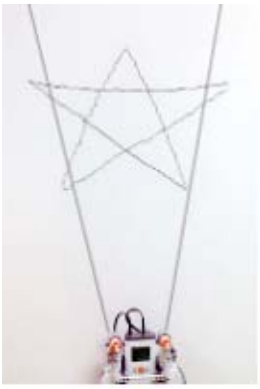
3) 모터 속도 측정 알고리즘
(Motor speed Algorithm)

양쪽 각각의 모터 A와 C가 동시에 도착할 수 있도록 거리에 따른 모터의 속도 값을 계산해 주는 알고리즘이다. 움직여야 할 A 모터의 길이와 C 모터의 길이의 차를 계산하여 값을 정하고, 절대값으로 처리하여 각각의 모터 속도값을 측정해주는 알고리즘이다.

$$Speed(X) = (\frac{L_{move}}{L_{MAX}} * Speed_{MAX}) + Speed_{MIN} \qquad (4)$$

Ⅳ. 구현 결과

본 연구는 가로 750㎜, 세로 750㎜ 크기 영역 안에서 이미지 작업이 진행되었다. 실험을 하는 데에 있어서는 마카를 이용하여 보드에 드로잉을 하였다.



▶▶그림 9. 드로잉 과정 그림 10. 드로잉 결과

마카를 이용하여 별 모양의 이미지를 실험해보았다. 위의 [그림 9]는 드로잉을 하고 있는 과정 중의 하나이다. 단순하게 포인트에서 포인트로의 직선으로 보이지만 실제 이미지에는 한 직선 사이사이에 많은 포인트가 있다. 최대 속도를 높일수록 빠르게 움직이나, 줄을 감고 풀며 이동하는 과정에서 흔들림이 좀 더 생기기 때문에 최대 속도를 낮춰서 실험하였다. [그림 10]은 드로잉 된 결과 이미지이다.

Ⅴ. 결론 및 향후 연구

가상환경에서는 좌표로 이동하는 동안에 이미지 그대로의 모습으로 진행되었기 때문에 실제 환경에서의 드로잉 작업을 끝내기 전까지는 구현 결과가 어떤 모습으로 나올지 알 수 없었다. 이러한 문제점으로 인하여 후에 가상환경에서도 실제 환경에서의 드로잉 모습을 그대로 나타나게 하는 방식으로 프로그래밍을 수정하였다. 즉, 가상 환경과 실제 환경이 실시간으로 변하여 진행되는 과정이 같은 결과를 얻을 수가 있었다.

지원하는 SVG 포맷의 이미지의 좌표 값을 받아 이미지를 보다 정확한 좌표 값으로 향하여 드로잉 작업을 하기 위해 중간 중간에 속도를 줄여 멈추게 함으로써 아직까지는 불안정한 움직임으로 흔들림이 나타났다. 추후에는 부드러운 움직임으로 보다 깔끔하고 정교하게 드로잉 작업을 할 수 있도록 개선해야 하겠다. 하지만 결과물이 원본 이미지와 같지 않다고 하더라도 가상 환경에서와 실제 환경에서 실시간으로 작동하며 원하는 모습으로 드로잉을 하는 이러한 작업은 그러한 과정 자체가 하나의 퍼포먼스로 작용할 수가 있다.

■ 참 고 문 헌 ■

[1] Physical Computing (Sensing and Controlling the Physical World with Computers), Dan O'Sullivan, Tom Igoe, 2004.

[2] 서동수, "피지컬컴퓨팅의 개념과 기술적 기초", 한국디자인학회, pp.270-271, 2006.

[3] 홍성대, "뇌파 및 사운드 신호에 반응하는 인터랙티브 영상에 관한 연구", 박사학위논문, pp.30-34, 2008.

[4] 장호연, 홍성대, 류승택, 박진완, "피지컬 로봇을 이용한 예술 작품 구현 환경 연구", 한국컴퓨터게임학회논문지, 제 15호, pp.161-178, 2008.

[5] 배희재, 송병규, 김윤기, 김창수, 정회경, . "XML 기반의 구조화된 그래픽 표현을 위한 SVG 문서 저작 시스템", pp.817-819, 2004.

[6] Hektor, http://www.hektor.ch/

[7] NVIDIA: Adam & Jamie draw a MONA LISA, http://www.youtube.com/watch?v=fKK933KK6Gg

[8] Artistic Painting Robot Paints Realism, http://www.vanarman.com/

[9] Processing in Eclipse, http://processing.org/learning/tutorials/eclipse/

[10] Processing, http://www.processing.org/

[11] Java - Icommand, http://folog.egloos.com/1296765

[12] Icommand, http://lejos.sourcefor gez.net/pztechnologies/nxt/icommand/api/ index.html

[13] Lego NXT, http://mindstorms.lego.com/

[14] SVG, http://www.w3.org/TR/SVG/

[15] brickinside, http://www.brickinside.com/

이 논문은 2008년 정부(교육과학기술부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임 (KRF-2008-321-H00001)