Hadoop에서 3D 프린팅용 G-code 생성 소프트웨어 개발

이규영*, 남기원*, 김건영*, 김성석**, 양순옥***
*서경대학교 컴퓨터과학과
**서경대학교 컴퓨터과학과, 교신저자
***가천대학교 교양학부
e-mail: {lky4644, jtt36, c2619zz}@naver.com, sskim03@skuniv.ac.kr, soyang@gachon.ac.kr

Development of G-code generating software for 3D printer in Hadoop

Kyuyoung Lee*, Kiwon Nam*, Gunyoung Kim*, Sungsuk Kim**
*Dept of Computer Science, Seokyeong University
**Dept of Computer Science, Seokyeong University(corresponding author)
***Liberal college, Gachon University

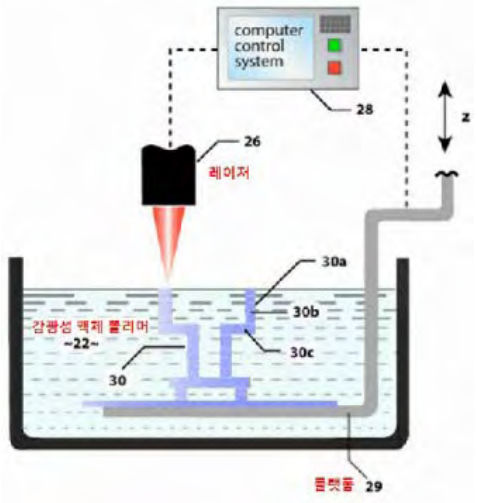
요 약

3D 프린터를 이용하여 출력을 하기 위해서는 3D 모델 데이터를 G-code로 변환하는 과정을 수행해야 한다. 일반적으로 3D 모델은 STL 파일 형식으로 저장되는데, 이 파일은 대개 삼각형 형식인 페이셋들의 좌표 데이터를 포함하고 있다. 만약 3D 모델의 크기가 커지거나 정밀도가 높아진다면, 페이셋의 수가 매우 많아지게 되고, 결과적으로 3D 모델에서 G-code로 변환하는 시간이 길어지게 된다. 본 논문에서는 널리 활용되고 있는 Hadoop에서 변환 소프트웨어를 개발하고자 하였다. Hadoop은 마스터 노드와 여러 데이터 노드들이 Map-Reduce 방식으로 작업을 수행한다. 이러한 노드들은 하둡 파일시스템(HDFS)을 공유할 수 있어 작업을 효율적으로 수행할 수 있다. 이에 본 논문에서는 이 시스템의 기능을 활용하여 기존에 개발된 분산 알고리즘을 변형한 후 이를 구현하고자 한다.

1. 서론

2차원 공간에 잉크를 뿌려 출력하는 일반적인 프린터와 달리, 3D 프린터는 평면(플레이트)을 Z축으로 이동하면서 3차원 물체를 출력할 수 있는 장치이다[1]. 출력방식은 입체 형식을 쌓아가는 적층형과 큰 재료를 깍아서 대상을 만드는 절삭형으로 구분된다. 본 논문에서는 적층형 방식을 고려하며, 따라서 프린터는 출력물을 받치는 플레이트와 잉크를 녹여 출력시키는 헤드로 구성된다(그림 1 참고)[2].

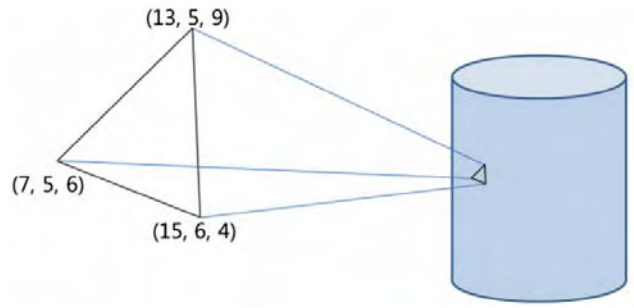
3D 프린팅은 모델링, 프린팅, 마무리의 3단계를 거쳐 진행된다. 모델링은 CAD와 같은 소프트웨어를 이용하여 출력하고자 하는 3차원 대상을 생성하는 과정이다. 이렇게 생성된 3차원 모델은 프린터용 G-code로 변환 과정을 거쳐 출력(printing)되고, 마무리 과정을 거쳐 최종적인 3차원 물체를 얻을 수 있다. 대개 3D 모델 데이터는 STL 파일 포맷으로 저장되는데, 이 포맷은 3차원 대상을 페이셋



(그림 1) 적층형 3D 프린터의 예 : SLA 방식[3]

(Facet)들의 집합으로 표현한다(그림 2 참고). 이 페이셋들은 특별한 규칙없이 저장되어 있다. 이에 반해 (적층형 방식의) 3D 프린터는 헤드가 이동하면서 잉크를 뿌려 대상의 아래쪽부터 출력해야 한다. 프린터를 위한 명령어는 대

* "이 논문은 2015년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. 2015R1D1A1A01061173)"



(그림 2) 삼각형 기반의 페이셋의 예

개 G-code로 작성된다. 따라서 3D 모델의 페이셋들로부터 대상의 밑면부터 최대 높이까지 플레이트와 헤드의 이동 및 출력을 위한 G-code 변환 작업은 많은 시간이 소요된다. 특히 대상의 높이가 커지거나 정밀도가 높아질 경우 페이셋의 개수가 많아지면서 상당한 변환 시간이 필요하다.

우리의 이전 연구에서 이러한 변환시간을 줄이기 위한 2단계 분산 처리 알고리즘을 제안하였다[5]. 1단계에서는, STL 파일에 저장된 페이셋들의 집합을 데이터 처리 노드의 개수(N)만큼 N등분하여 전달한다. 데이터 노드들은 받은 페이셋들을 제일 낮은 Z값을 기준으로 정렬하여 돌려준다. 마스터 노드들은 최종적으로 정렬된 페이셋 집합을 만든다. 2단계에서는 정렬된 페이셋을 다시 N등분하여 데이터 노드에게 전달한다. 각 데이터 노드들은 받은 페이셋을 이용하여 완전 분산 방식으로 G-code 변환 작업을 수행한다.

본 연구에서는 기 개발된 분산 알고리즘을 클라우드 시스템에 적용할 수 있도록 알고리즘을 변형한 후, 이를 Hadoop에서 구현하고자 한다.

2. Hadoop에서 변환 소프트웨어 개발

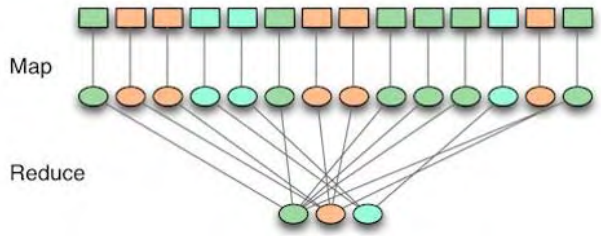
2.1 Hadoop

Hadoop은 대용량 데이터를 분산 처리할 수 있는 자바 기반의 오픈소스 프레임 워크를 의미한다. Hadoop은 데이터를 분산 저장할 수 있는 기술(HDFS)과 분산 처리기술(MapReduce)이 있어 대용량의 데이터라도 여러 데이터 노드들이 분산하여 처리하기 편리하다.

HDFS는 여러 서버에 설치되어 동작하며, 분산 환경에 최적화된 모듈로 파일을 안전하게 저장할 수 있도록 한다. MapReduce는 대규모 데이터를 수직화하여, 그 데이터를 각각의 종류 별로 모으고(Map), 필터링과 정렬 과정을 거쳐 데이터를 뽑아내는(Reduce) 분산처리 기술을 의미한다(그림 3 참고).

2.2 개발환경

개발을 위해 3대의 컴퓨터에 Hadoop을 설치하였다. PC1은 마스터 노드로서, jobtracker 및 name node 역할을 수행하며, 나머지 PC들은 데이터 노드로 동작한다. Java



(그림 3) Hadoop의 맵 리듀스 기술 [6]

로 개발을 진행하기 위해 hadoop/conf/hadoop-env.sh 파일을 수정하였으며, name node 속성은 core-site.xml에서 다루도록 하였다. 그리고 mapred-site.xml에서 jobtracker를 설정하였다.

3. 변환 알고리즘

이전 연구[5]에서 개발한 분산 변환 알고리즘은 STL 파일에 저장된 페이셋 데이터들이 완전 분산 방식으로 변환될 수 있도록 전처리 과정을 거친다. 이것은 저장된 페이셋들은 출력물의 높이(Z축 값)에 어떤 규칙성도 보이지 않지만, 이에 반해 3D 프린팅 작업이 대개 Z축을 기준으로 0부터 최대 높이까지 순차적으로 진행된다는 속성을 이용한 것이다.

이를 위해 1단계에서는 각 페이셋의 세 좌표값에서 제일 낮은 Z값(z_{min})과 제일 큰 Z값(z_{mzx})을 추출한 후, 이 값을 이용하여 정렬을 수행한다. 2단계에서는 출력 대상의 높이 값을 N등분한 후, 페이셋들의 z_{min}값에 해당하는 구간을 데이터 노드에게 전달하여 변환작업을 수행하도록 한다.

Hadoop은 모든 데이터 노드들이 HDFS에 저장된 데이터를 읽거나 쓰기 작업을 할 수 있다. Map의 결과로 (Key, value)쌍의 데이터가 생성되며, 이 데이터 집합은 Shuffling이 수행되어 Key값으로 정렬된 후 HDFS에 저장된다. 이 데이터는 Reduce에 의해 읽혀지게 된다.

이 과정을 4단계로 구분하여 설명한다.

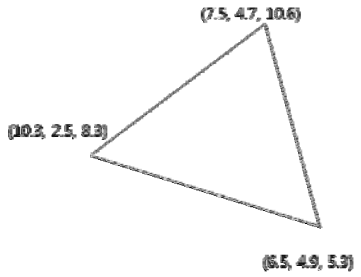
3.1 전처리

STL 파일에 저장된 페이셋은 3개의 공간 좌표값으로 구성되어 있다. 이 파일로부터 각 페이셋에 대한 공간 좌표값만 추출하여 문자열 형식으로 저장한다. 예를 들면, 파일로부터 하나의 페이셋의 좌표값(예를 들면, (10.3, 2.5, 8.3), (6.5, 4.9, 5.3), (7.5, 4.7, 10.6))을 읽어들인 후, 이를 문자열 “10.3 2.5 5.3 6.5 4.9 8.3 7.5 4.7 10.6”로 저장한다.

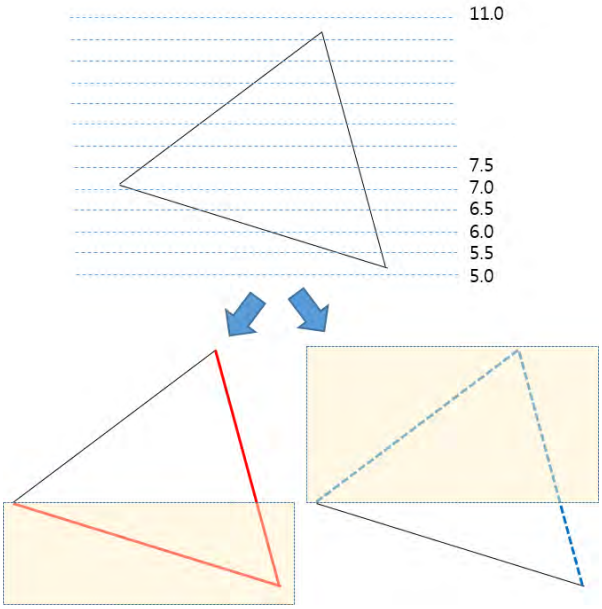
3.2 데이터 노드에서 Map

이 단계에서 각 데이터 노드는 N등분된 전처리 파일을 읽어서 직접 G-code를 생성한다. 읽어들은 각 (Key, Value)에서 Value가 이전 전처리 단계에서 저장한 좌표값이다. (그림 4)는 전처리 전 하나의 페이셋을 보여주고

있다. Map이 이 페이셋을 읽으면 (그림 5)와 같은 과정에 따라 G-code를 생성한다.



(그림 4) 페이셋의 한 예



(그림 5) 각 페이셋으로부터 G-code 생성

예제로 든 페이셋은 높이(Z축값)가 5.3 - 10.6에 해당한다. 하지만 3D 프린터에서 한번에 출력하게 되는 잉크 자체가 일정 두께를 가진다. (그림 5)에서는 그 두께가 0.5라고 가정한다. 따라서 그림의 윗부분처럼 0.5 단위마다 만나는 두 점의 좌표가 바로 프린터 헤드가 이동하면서 잉크를 뿌리는 위치가 된다. 그림에서 제일 아래 Z값은 5.3이지만, 플레이트의 위치는 5.0, 5.5, 6.0처럼 0.5씩 증가하게 된다. 따라서 Z값이 5.5 위치부터 교점의 좌표를 계산하여 그에 해당하는 G-code를 생성한다.

교점의 좌표를 계산하기 위해서 두 점을 지나는 직선의 공식을 이용하여 계산한다.

$$\frac{x-x_2}{x_2-x_1}=\frac{y-y_2}{y_2-y_1}=\frac{z-z_2}{z_2-z_1} \qquad (1)$$

즉, 수식 (1)에서 z 변수에 현재의 플레이트 높이 값을 대입하면 교점의 좌표를 얻을 수 있다.

이렇게 얻어진 두 좌표 정보는 이후 Reduce를 위해 다시 (Key, value)로 저장된다. 이때 Key는 높이이고, value는 교점의 좌표가 된다.

3.3 Shuffling

Map 단계는 하나의 페이셋으로부터 다수의 (key, value) 쌍을 생성하게 되며, 또한 다수의 페이셋이 변환 처리해야 한다. Shuffling은 생성된 (key, value) 쌍을 key 값을 기준으로 정렬한다. 여기서 key는 출력이 이루어질 플레이트의 높이 값이라고 해석할 수 있다. 따라서 동일한 key값(즉, 동일한 높이)을 가지는 여러 value들이 묶여서 $(key, [value_1, value_2, ...value_n])$ 처럼 표현된다.

3.4 Reduce

이 단계는 Shuffling의 결과로 동일한 높이에 해당하는 G-code에 대한 최적화 작업을 수행한다. 현재 프린터의 플레이트 높이에서 헤드의 이동이 최소가 될 수 있도록 G-code 명령을 조절한다. 이는 key에 대한 value 값들의 위치를 변경하거나, 출력 위치와 위치 사이에 헤드를 이동하는 명령을 추가하는 것을 의미한다.

4. 결론 및 향후 작업

본 논문에서는 Hadoop을 활용하여 3D 모델로부터 G-code를 생성하기 위한 변형된 알고리즘을 개발하였다. 즉, 이전 분산 환경에서 개발된 알고리즘은 필요한 모든 작업을 노드들이 수행해야 했다. 하지만 Hadoop 시스템의 강력한 HDFS과 MapReduce 기능을 활용하면 일부 기능을 개발자가 고려하지 않아도 되어서 프로그램 개발하기가 훨씬 용이해졌다.

향후 개발한 프로그램에 대하여 다양한 상황을 가정하여 성능을 평가하고자 한다. 즉, 대상 객체의 크기가 커지거나 정밀도가 높아진다면 변환에 걸리는 시간은 매우 커진다. 각 경우마다 최적화할 수 있는 방안을 별도로 마련하여 프로그램의 성능을 높일 수 있도록 하고자 한다.

참고문헌

[1] 박세환, “3D 프린팅 표준기술 및 시장동향”, TTA Journal Vol. 149, pp. 80-84, 2013

[2] Jacobs, Paul Francis, "Rapid Prototyping & Manufacturing: Fundamentals of Stereolithography". Society of Manufacturing Engineers. 1992

[3] “StereoLithography Interface Specification”, 3D Systems, Inc., 1988

[4] Lynch, Mike, "When programmers should know G code", Modern Machine Shop (online ed.), 2010

[5] S. Kim and S.O. Yang, “Transforming algorithm of 3D model data into G-code for 3D printers in distributed systems”, 11th Int'l Conf. on Ubiquitous Information Technologies and Applicatioin(CUTE), 2016

[6] 정재화, “시작하세요! 하둡 프로그래밍 빅데이터 분석을 위한 하둡 기초부터 YARN까지”, 위키북스, 2016