JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 25, No. 5, pp. 1263-1271, May. 2024

CPU 병렬 처리를 이용한 SPH 유체와 단일보드 컴퓨터의 실시간 상호작용 시스템 개발

이 건 희[1] · 최 웅[2*]
[1]강남대학교 소프트웨어응용학부 공학사
[2*]강남대학교 ICT융합공학부 부교수

Development of Real-Time Interaction System for SPH Fluid Using CPU Parallel Processing and Single-Board Computer

Geon-Hee Lee[1] · Woong Choi[2*]

[1]Bachelor's Degree, Department of Software Application, Kangnam University, Yongin 16979, Korea
[2*]Associate Professor, College of ICT Construction & Welfare Convergence, Kangnam University, Yongin 16979, Korea

[요 약]

가상현실에서 사용자와 직접 상호작용하는 콘텐츠를 통해서 현실감을 향상시키는 것이 중요하게 여겨지고 있다. 강체와 같이 독립적인 객체가 아닌 수많은 입자의 형태로 이루어진 유체와 실시간으로 충돌을 다루는 실감 콘텐츠의 연구는 미흡한 것으로 보인다. 본 연구에서는 입자 형태의 유체를 시뮬레이션으로 구현하고 충돌 처리를 통해 사용자와 상호작용하는 시스템을 구현하였다. 상호작용을 위해서는 유체를 이루고 있는 입자의 데이터를 각각 처리하는 속도를 고려해야하기 때문에 상호작용이 가능하도록 데이터 처리 방식을 개선하였다. 이에 더해 초음파 센서에서 측정된 값을 통해 유체 시뮬레이션의 동적 변화를 준다. 이를 통해 실시간 상호작용의 가능성을 확인하였다. 따라서 본 연구는 향후 유체를 활용하는 가상현실 콘텐츠에서 상호작용 시스템의 기반으로 활용되기를 기대한다.

[Abstract]

In virtual reality, enhancing realism through content that directly interacts with the user is increasingly considered important. Research on realistic content that deals with real-time collisions with fluids, which are not independent objects like rigid bodies but are made up of numerous particles, seems to be insufficient. In this study, a system that simulates fluids in particle form and interacts with users through collision processing was developed. To enable interaction, the data processing method was improved to consider the speed of processing the data of each particle that makes up the fluid. Additionally, dynamic changes in the fluid simulation were induced through values measured by ultrasonic sensors. This confirmed the possibility of real-time interaction. Therefore, this study is expected to serve as a basis for interaction systems in virtual reality content utilizing fluids in the future.

색인어 : 가상현실, 인간 컴퓨터 상호작용, 유체 시뮬레이션, 충돌 처리, 병렬 처리
Keyword : Virtual Reality, Human Computer Interaction, Fluid Simulation, Collision Detection, Parallel Processing

http://dx.doi.org/10.9728/dcs.2024.25.5.1263

Received 15 February 2024; **Revised** 14 March 2024
Accepted 21 March 2024

***Corresponding Author; Woong Choi**

Tel: +82-31-280-3753
E-mail: wchoi@kangnam.ac.kr

Ⅰ. 서 론

기존의 시뮬레이션 및 가상현실(VR; Virtual Reality) 콘텐츠는 주로 시각과 청각 자극에 초점을 맞춘 사용자 상호작용 위주로 이루어졌다. HMD(Head Mounted Display) 같은 장비의 상용화로 시각과 청각 외에도 다른 감각을 자극하여 사용자의 몰입감을 높이는 방향으로 발전하였다[1]. 특히, 촉각에 대한 연구가 지속적으로 진행되며, '고무손 착각(Rubber-Hand Illusion)' 실험과 같은 신경과학 연구를 통해 촉각과 시각의 결합이 인간의 감각 인식에 중요한 역할을 한다는 것이 입증되었다[2]. 이러한 연구 결과를 바탕으로 메타버스 콘텐츠 개발은 인간의 다양한 감각 경험을 가상 환경에 구현하여 여러 분야에 적용할 수 있는 가능성을 열었다.

이전 연구들은 가상 환경에서 사용자의 손과 특정 강체가 상호작용하는 시스템을 개발하여 실험해 보았다[3]. 저비용 시스템 구축을 위해 아두이노 보드를 활용하며, 이 보드는 MCU(Micro Controller Unit)로 사용된다. 사용자가 장비를 착용하고 움직일 때, 여러 센서들이 반응하여 가상 환경 속 사용자의 손 객체가 움직이고, 이 손 객체는 다른 강체 객체와의 충돌을 통해 촉각적 피드백을 제공한다. 가상 환경에서의 이러한 상호작용은 진동을 통한 촉각 피드백을 제공한다[4]. 이 시스템을 통해 디지털 환경에서 실제와 같은 상호작용 경험을 재현할 수 있으며, 이는 라이프로깅과 미러 월드 개념과 결합되어 사용자에게 더욱 실감나는 경험을 제공한다[5]. 이는 HCI(Human Computer Interaction) 분야에서 몰입도를 향상시킬 수 있는 방향을 제시한다. 또한, 사용자의 손 움직임을 통해 로봇 손을 직관적으로 제어하는 연구와[6] 가상 환경에서 햅틱 장비를 이용한 반복적인 경험을 통해 교육 및 훈련을 실현하는 연구도 진행되었다[7].

그러나 물체 간 충돌을 기반으로 한 상호작용 시스템 구축에 관한 연구는 이루어졌지만, 강체가 아닌 유체를 다루는 연구는 상대적으로 부족한 상황이다. 본 연구는 강체가 아닌 유체와의 충돌을 다루며, 가상 환경에서 유체를 구현하고 접근하는 방법 및 활용 방안에 있어 강체와는 다른 접근이 필요함을 지적한다. 입자 기반의 유체 시뮬레이션 모델을 통해 사용자 상호작용을 구현하기 위해서는 유체를 구성하는 각 입자에 접근하는 처리 방식이 필요하다. 본 연구는 유체역학 기반의 실시간 물리 시뮬레이션을 구현하고, 사용자와의 상호작용을 가능하게 하는 두 가지 시스템을 제안한다: 첫째는 사용자가 조종할 수 있는 충돌 객체와 유체의 충돌을 감지하여 실시간으로 처리하는 시스템이고, 다른 하나는 사용자의 신체 일부를 초음파 센서로 감지하여 충돌 객체를 조종하는 시스템이다.

본 논문의 내용 구성은 다음과 같다. 우선, 2장에서는 유체 시뮬레이션에서 충돌 처리 기능을 포함하여 구현하는 방법을 소개하고 시뮬레이션 시스템과 단일보드 컴퓨터의 결합 방법 과정을 설명한다. 3장에서는 제안한 방법으로 구현된 전체적인 시스템의 시뮬레이션 결과를 설명하고 4장에서 연구의 결론을 맺는다.

Ⅱ. 시뮬레이션 시스템 및 단일보드 컴퓨터 구성

2-1 시뮬레이션 시스템 구성

표 1. 기본 시뮬레이션 구조
Table 1. Configuration of the basic simulation

Class	Function	Purpose
Camera	Camera	Responsible for vision of a user within the simulation
Geometry	Mesh	Building a objects and surface of the objects in the simulation
	Shader	
Particle	Particle	Initialization of 3-Dimensional particles sequence
System(Solver)	Building tables	Building the hash table for computation among particles
	Update	drawing particles and parallel processing data
Engine	Main	The main function of the simulation system

표 1은 실험에 사용한 기본적인 시뮬레이션 구조를 표현한다. 시뮬레이션은 C++로 구조화하였고 유체 시뮬레이션 동작에 필요한 클래스의 이름과 용도를 설명한다.

1) SPH

유체 시뮬레이션을 구현하는 방식인 Smoothed Particle Hydrodynamics(SPH)는 격자 방법을 사용하지 않는 방법 중의 하나로써, 입자 간의 연산을 이용한 순수 라그랑지안 기법이다. 일반적으로 비압축성을 필수적으로 고려한다. 따라서 SPH 유체에 사용되는 입자의 형태는 기존 연구에서 제안된 기법으로 구현한다[8].

2) 충돌 처리

사용자와 상호작용을 위한 시스템으로써 먼저 사용자가 직접 조종할 수 있는 충돌 객체를 생성해야한다. 유체 입자를 그릴 때 사용한 3D 오브젝트 파일과 쉐이더 파일을 사용해서 충돌 객체의 형태를 그리는 새로운 클래스 'Controller'를 디자인한다. 충돌 객체는 단일보드 컴퓨터를 사용하기 이전에 컴퓨터에 연결된 마우스로 위치 정보를 변환시킬 수 있다.

표 2는 시뮬레이션 환경에서 사용자가 직접 조종할 수 있는 충돌 객체를 생성하기 위한 클래스의 구성이고, 앞으로 소개할 충돌 객체의 기능들을 포함한다.

사용자가 마우스 오른쪽 버튼을 클릭한 상태로 움직이게 된다면 마우스의 이전 위치 정보의 변화량에 따라 충돌 객체의 y좌표가 변하게 된다.

표 2. 충돌 감지 객체에 대한 클래스 구성
Table 2. Configuration of class to collision detection object

Class	Function	Purpose
Controller	Controller Initialization	information Initialization of the collider to control by the user in the simulation
	Moving Controller	Moving and calculating the position of the collider

y좌표는 OpenGL 3D 환경에서 상-하 방향이므로, 유체의 입자들과 충돌 객체가 서로 만나 충돌할 수 있도록 하는 핵심적인 방향이다. 또한, 클릭 없이 마우스를 조작할 경우, 충돌 객체의 x좌표와 z좌표가 변하게 된다. x좌표와 z좌표는 OpenGL 3D 환경에서 중력의 영향을 받지 않는 평면을 의미하기 때문에 마우스로 휘젓는 듯 동선을 그리게 되면 충돌 객체는 유체의 입자와 지속적으로 충돌이 이루어지며 결과적으로 유체를 휘젓는 시뮬레이션이 가능하다.

충돌 객체가 유체의 입자와 충돌이 감지되는 알고리즘은 유체의 입자의 반지름과 충돌 객체의 반지름의 합을 기준으로 충돌 여부의 참과 거짓을 구별할 수 있다.

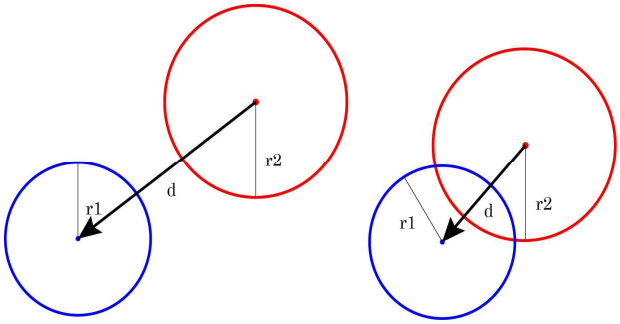

그림 1. 충돌 객체와 유체의 입자의 충돌 처리 원리
Fig. 1. The principle of collision detection among the collision object and a particle of fluid

$$r_1 + r_2 \geq d \tag{1}$$

식 (1)에서 r_1은 유체의 입자 객체의 반지름이고 r_2는 충돌 객체의 반지름이다. d는 측정한 유체의 입자 객체의 반지름과 충돌 객체의 반지름의 합이다. 모든 입자들에 접근을 해서 중심을 기준으로 현재 입자와 충돌 객체와의 거리가 현재 입자의 반지름과 충돌 객체의 반지름의 합보다 작으면 이는 충돌로 감지한다. 충돌된 순간 해당 유체의 입자와 충돌 객체의 방향 벡터를 정규화하고 충돌된 반대 방향 즉 충돌 객체가 유체의 입자를 밀어내는 방향으로 유체의 입자 간에 압력을 연산하는 수식을 동일하게 적용해서 새로운 압력 값을 적용한다. 이 충돌 처리 과정은 특정 유체의 입자가 다른 유체의 입자 간에 충돌로 유체의 현상을 위해 연산되는 동시에 새로운 압력 값이 중복 적용되는 것이기 때문에 해당 입자가 갖게 되는 압력 값이 거대해지는 경우가 발생한다. 이는 단순히 충돌 객체와 충돌하여 유체의 입자가 밀리는 현상이 발생하는 것이 아닌 충돌하는 순간 유체가 폭발 하는듯한 현상이 발생한다. 식 (2)에서 F_i는 유체의 현재 입자에 추가적으로 발생하는 충돌 압력의 힘이다. $\overrightarrow{p_i}$는 현재 유체의 입자의 방향 벡터, $\overrightarrow{p_H}$는 고정적으로 충돌 객체의 방향 벡터이며 P_i와 P_j는 각각 충돌 되는 유체의 입자의 현재 저장되어 있는 압력 값이다. w는 가중치이다. D_j는 유체의 이웃 입자의 점도 값이고 S_{grad}는 수식에 적용하는 고정 상수 값이다. r_H는 충돌 객체의 반지름이고, $\overrightarrow{p_H}$는 충돌되는 유체의 이웃 입자의 방향 벡터이다. 따라서 위 식과 같이, 중복 적용되는 압력 값이 거대해지는 현상을 방지하기 위한 적절한 가중치, w를 정해서 수식에 적용한다. 본 연구의 실험에서는 충돌 객체의 충돌 압력 가중치를 '10000'으로 정한다.

$$F_i = \frac{\overrightarrow{p_i} - \overrightarrow{p_H}}{\|\overrightarrow{p_i}\| - \|\overrightarrow{p_H}\|} \{(P_i + P_j)w/(2D_j)(S_{grad})\}(r_H - \sqrt{\overrightarrow{p_H} - \overrightarrow{r_j}}) \tag{2}$$

2-2 상호작용을 위한 단일보드 컴퓨터 구성

단일보드 컴퓨터를 이용해서 사용자와 상호작용하기 위해서는 시뮬레이션에서 충돌을 감지했을 때 해당 감지 신호를 데이터화해서 단일보드 컴퓨터와 통신할 수 있어야 한다. 단일보드 컴퓨터와 시뮬레이션간의 통신 수단은 연결형 TCP/IP(Transmission Control Protocol/Internet Protocol) 프로토콜을 기반으로 하는 소켓 통신을 사용한다.

1) TCP / IP 프로토콜 소켓 통신

시뮬레이션 환경과 단일보드 컴퓨터는 서로 같은 와이 파이(Wi-Fi; Wireless Fidelity) 서버를 접속해야 서로 갖고 있는 고유의 소켓 즉 클라이언트 소켓과 서버 소켓이 연결이 가능하다. 본 연구의 제안 시스템에서는 유체 시뮬레이션이 실행되는 환경에서 Microsoft Visual Studio 2019 버전을 사용하여 C++로 클라이언트 소켓을 구축한다. 그리고 통신을 시도하는 단일보드 컴퓨터는 라즈베리파이 4(Raspberry Pi 4)를 이용한다. 라즈베리파이 4에서 터미널을 이용하여 C++로 서버 소켓을 구축하고 클라이언트 소켓과 연결이 될 수 있도록 시스템을 구성한다.

2) 데이터 처리 최적화 전략 1 – 패킷화(Packetization)

시뮬레이션에서 'Particle' 클래스는 해당 입자의 충돌 정보를 저장 및 관리하는 boolean 변수가 있다. 충돌 객체와 해당 유체의 입자가 서로 충돌되었을 때, 충돌 정보 변수의 값은 기본 값 false에서 true로 변환된다. 시뮬레이션은 라즈베

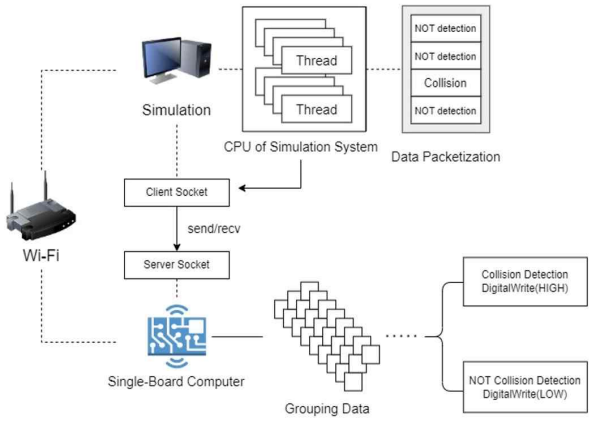

그림 2. 연구 실험으로 사용하게 된 TCP/IP 통신 기반 제안 시스템의 개요

Fig. 2. Overview of the proposal system based on TCP/IP communication, using as research experimental

리파이 4에게 이 변수 값을 전송해야한다. 충돌 객체가 유체의 입자와 충돌할 때 서로의 반지름의 합을 계산하기 위해서 모든 유체의 입자에 접근하여 측정하는 것처럼, 충돌 데이터를 전송하는 과정 또한 모든 유체의 입자에 접근해서 해당 유체의 입자의 충돌 정보를 저장하는 변수의 값을 전송한다. 전송되는 데이터의 형태는 충돌이 감지되었을 시 문자형 '1'을 전송하고 충돌이 감지되지 않았을 시 문자형 '0'을 전송한다. 충돌 정보를 전송하는 모든 과정은 시뮬레이션 환경에서 병렬 처리로 이루어진다. 라즈베리파이 4에서 해당 충돌 장비를 수신 받았을 때, 현재 수신 받은 데이터의 값이 '1'일 경우 디지털 신호를 보내고 '0'일 경우는 디지털 신호를 끊는다. 따라서 시뮬레이션에서 충돌이 감지됨에 따라 라즈베리파이 4에 조립된 발광다이오드(LED; Light Emitting Diode) 가 점등되고 소등되는 것을 반복한다.

그러나 이러한 방식은 전체적인 시뮬레이션 실행 시간에 영향을 미친다. 추가적으로 유체의 입자와 충돌 객체가 서로 충돌했음에도 불구하고 LED가 지속적으로 점등되는 것이 아닌 깜빡거리는 현상이 발생하였다. 이러한 이유는 현재 처리되고 있는 유체의 입자의 충돌 정보는 충돌 감지가 된 상태이지만, 충돌되지 않은 다른 유체의 입자 충돌 정보도 계속해서 전송되어 처리되고 있기 때문에 오히려 깜빡거리는 현상이 나타난다. 실행 시간 지연 문제와 LED가 깜빡거리는 현상을 해결하기 위해 전송되는 데이터를 패킷화하였다. 충돌 정보를 여러 개를 패킷하고 패킷된 데이터는 한 개의 데이터로 간주한다. 패킷 내부 데이터는 유체의 입자의 충돌 정보로 이루어져있는데 단 한 개라도 충돌이 감지 된 데이터라면 이는 충돌이 된 것으로 간주하고 라즈베리파이 4에 전송한다. 패킷된 데이터 중에서 충돌 데이터를 찾는 알고리즘은 패킷된 충돌 정보 저장하는 변수의 합을 구하고 해당 결과가 1 이상이면 충돌 데이터가 있는 것으로 간주한다. 실험에서는 패킷의 사이즈를 '4'로 고정하였는데, 패킷의 사이즈가 커지면 커질수록 패킷화를 하는데 시간이 많이 소요가 되기 때문에 오히려 역효과가 발생하였다. 따라서 시행착오를 통해 가장 적절한 사이즈인 '4'로 고정하였다.

3) 데이터 처리 최적화 전략 2 – 그룹화(Grouping)

라즈베리파이 4에서 패킷화한 데이터를 수신하고 처리하게 되었을 때 이전보다 개선된 LED 점등 상태를 보여주었지만 아직 기능적으로 미흡한 부분이 있었다. 따라서 패킷 데이터를 사용하는 방식에 영감을 받아, 단일 보드 컴퓨터에서도 해당 데이터를 수신 받는 즉시 처리하는 것이 아닌 그룹화를 하여 처리하도록 구성한다. 수신 받는 데이터의 형태는 문자형으로 '0'과 '1'이기 때문에 그룹화한 데이터들을 각각 ASCII(American Standard Code for Information Interchange) 코드에 따라 정수형 0과 1로 치환한다. 그룹화한 데이터들의 정수 합을 구하고 합의 결과가 1 이상의 값으로 도출된다면 이는 충돌로 간주하여 디지털 신호를 발생시킨다. 그룹화를 할 때의 단위는 적절하게 설정하는 것이 중요하다. 따라서 본 연구에서는 그룹화 단위를 '100'으로 설정한다.

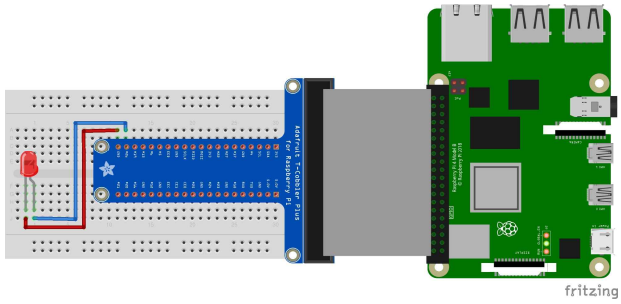

그림 3. GPIO 확장 보드를 사용한 라즈베리파이 4에서 LED 조립한 설계도

Fig. 3. Blueprint for LED assembly on Raspberry Pi 4 using GPIO extension board

4) 단일보드 컴퓨터

라즈베리파이 4와 함께 사용하는 부품은 다용도 입출력 포트(GPIO; General Purpose Input/Output) 확장 보드와 LED 그리고 초음파 센서를 사용한다. LED를 사용해서 시뮬레이션으로부터 데이터를 원활하게 수신하는지 시각적으로 확인한다. 시뮬레이션에서 충돌 객체와 유체의 입자가 충돌되었을 때 충돌 데이터를 전송하고 라즈베리파이 4는 수신 받은 데이터에 따라 LED를 점등 및 소등을 시킨다. LED가 자연스럽게 점등 및 소등이 되는 것을 확인할 수 있다.

5) 초음파 센서 활용

현재 과정까지 시스템은 시뮬레이션에서 라즈베리파이 4로 패킷화된 충돌 정보 데이터를 전송하고 라즈베리파이 4에서는 해당 데이터를 그룹화 하여 처리한 결과로 디지털 신호를 발생시키는 방법을 사용한다. 사용자가 마우스를 이용해서 시뮬레이션 내의 존재하는 충돌 객체를 조종하는 시스템이다. 다음으로 제안하고자 하는 시스템은 초음파 센서를 사용해서

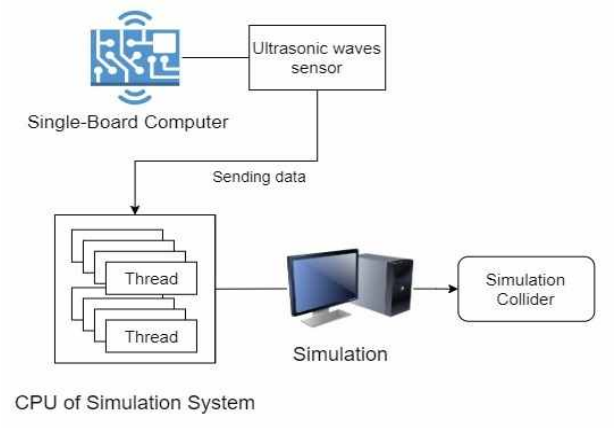

그림 4. 단일보드 컴퓨터의 초음파 센서로 측정한 거리 값을 시뮬레이션 환경으로 전송하는 구상도

Fig. 4. The conception of sending a distance values measured by an ultrasonic sensor of the single-board computer to the simulation environment

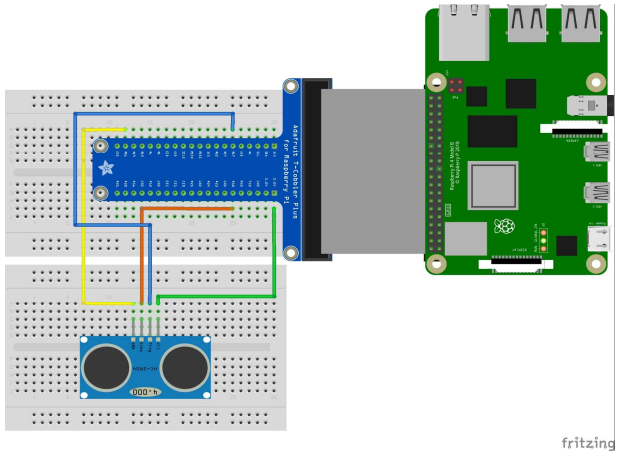

그림 5. GPIO 확장 보드를 사용한 라즈베리파이 4에서 초음파 센서를 조립한 설계도

Fig. 5. Blueprint for an ultrasonic waves sensor assembly on Raspberry Pi 4 using GPIO extension board

사용자의 신체 일부를 감지하고, 측정된 초음파 센서와 사용자의 신체 일부와의 거리 값을 통해 시뮬레이션 내의 존재하는 충돌 객체를 조종하는 시스템이다.

초음파 센서가 측정한 거리 값은 시뮬레이션 내의 충돌 객체의 y좌표를 변환하는데 사용된다. 라즈베리파이 4의 브레드보드에서 초음파 센서는 하늘을 가리키게 조립을 하고 사용자의 손이 상-하 방향으로 움직임에 따라 거리 값이 변하도록 설계를 한다. 사용자가 특정 공간에 담겨있는 액체를 손으로 직접 접촉을 시도할 때, 실제로 중력의 영향으로 액체의 위치는 사용자의 손의 위치보다 아래에 있을 수밖에 없기 때문에 사용자는 손을 직접 액체의 방향으로 뻗어야한다.

이러한 원리를 이용해서 초음파 센서가 측정하는 사용자의 손과의 거리의 값이 작아질수록 충돌 객체의 y좌표 값은 점점 감소하고 값이 커질수록 충돌 객체의 y좌표 값은 점점 증가한다. 따라서 시뮬레이션 환경에서는 충돌 객체의 위치가 사용자의 손의 높이에 따라 상-하 방향으로 움직이는 것을 확인할 수 있다.

초음파 센서가 기본적으로 측정하는 빈도수를 시스템 그대로 적용하게 되면 빈도수가 많아 시뮬레이션 실행 속도 전반에 영향을 미치는 문제가 발생한다. 초음파 센서는 40 kHz의 주파수를 사용하기 때문에 한 번의 거리를 측정하는 시간은 1 ms이하로 계산된다[9].

$$Distance = \frac{Speed}{170.15m} \times \frac{Meters}{100cm} \times \frac{\leq 6\mu S}{17015m} \times \frac{58.772\mu S}{cm} \quad (3)$$

시스템 실행 상태를 원활하게 유지하기 위해 초음파 센서가 측정하는 빈도수를 줄일 필요가 있다. 초음파 센서의 측정 빈도를 줄이기 위해서 한 번의 측정이 끝나면 10 ms의 시스템 지연을 발생시킨다. 초음파 센서의 측정 빈도를 줄이기만 해도 전체적인 시뮬레이션 실행 시간이 개선된다.

Ⅲ. 연구 결과 및 고찰

3-1 연구 결과

시뮬레이션에서 각 입자간의 연산을 병렬로 처리함으로써 자연스러운 유체의 움직임이 실시간으로 표현되었다. 또한, 유체와 충돌 객체간의 충돌 연산 과정에서 적절한 가중치를 적용함으로써 충돌 시 폭발하는 모습이 아닌 자연스럽게 휘젓는 모습이 표현되었다. 그림 6은 압력 수식에 가중치를 적용하지 않은 유체의 상태를 보여주고 그림 7은 압력 수식에 가중치를 적용하여 자연스럽게 흐르는 유체를 보여준다.

TCP/IP 프로토콜을 사용한 소켓 통신으로 데이터를 라즈베리파이 4로 전송하였을 때, 제안한 두 가지의 데이터 처리 최적화 알고리즘을 통해 실시간으로 데이터를 처리할 수 있게 되었으며 요구했던 사항 즉 데이터 처리 결과로 LED를 점등 및 소등을 하는데 충돌 발생 시 깜빡거리는 현상이 아닌 지속적으로 점등이 되는 사항을 만족하였다.

초음파 센서를 사용한 시스템은 충돌 객체가 사용자의 신체 일부, 즉 사용자의 손이라는 가정하고 센서로부터 사용자의 손까지의 거리를 측정하고 거리 값에 따라 시뮬레이션 내의 충돌 객체가 움직임을 확인하였다. 이를 통해 사용자가 마우스를 사용하여 충돌 객체를 움직인다는 이질감을 해결하였다.

3-2 고찰

본 연구에서 제시한 두 가지 시스템은 몇 가지 실험적 제약을 가지고 있다. 첫 번째로, 라즈베리파이 4를 사용하여 두 시스템 간의 충돌 데이터를 주고받는 시스템을 구축하려 할 때, 초기에는 아두이노 레오나르도 보드를 활용한 실험이 진행되었다. USB(Universal Serial Bus) 연결을 통한 통신 시도에서,

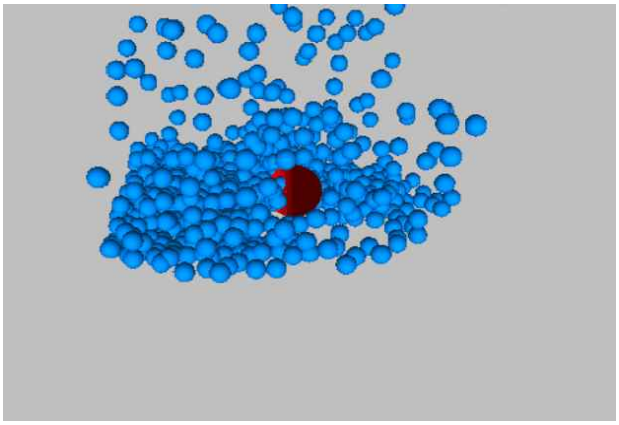

그림 6. 각 입자의 압력 값에 가중치를 적용하지 않아 유체가 폭발하는 것 같은 움직임을 보여주는 시뮬레이션
Fig. 6. The fluid in the simulation without weighted pressure for natural motion

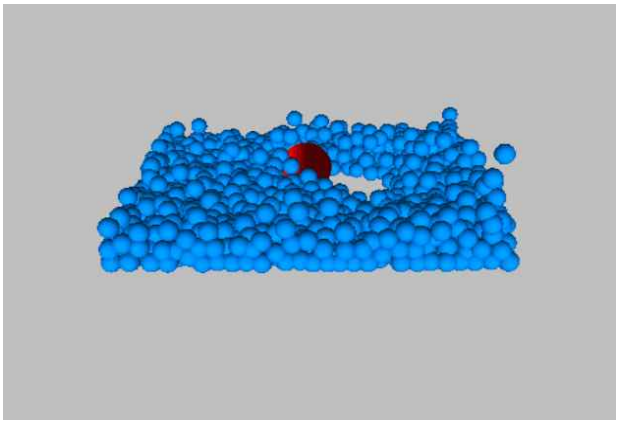

그림 7. 각 입자의 압력 값에 가중치를 적용하여 유체의 자연스러운 움직임을 보여주는 시뮬레이션
Fig. 7. The fluid that is applied weight to pressure equation in the simulation

UART(Universal Asynchronous Receiver/Transmitter)를 지원하는 아두이노 보드는 C++로 처리되는 데이터의 실시간 처리에 속도적 제한을 보였다.

이러한 문제를 해결하기 위해, 중앙 처리 장치(CPU; Central Processing Unit)를 내장한 라즈베리파이 4로 실험 장비를 교체하였고, TCP/IP 소켓 통신을 사용하여 데이터 송수신을 구현하였다. 아두이노 보드를 사용했을 때는 직렬 통신을 통해 데이터 수신은 빠르나, 수신된 데이터를 즉각 처리하는데 있어서 실시간 처리가 어려운 단점이 있었다. 반면, 라즈베리파이 4를 사용함으로써 물리적 연결 없이 소켓 연결로 데이터를 주고받게 되어, 첫째로 시뮬레이션 실행 환경과 사용자 상호작용이 가능한 단일보드 컴퓨터 간의 거리적 제약이 해소되었다. 둘째, CPU가 있는 라즈베리파이 4가 수신받는 데이터를 처리해주기 때문에 시뮬레이션에서 전송하는 데이터의 형태를 적절히 패킷화 한다면 실시간에 근접하게 처리해줄 수 있다. 따라서 아두이노 보드의 성능적 한계를 극복하고 다양한 시스템을 구축하고 실험할 수 있는 환경이라는 기대로써 유의미하다.

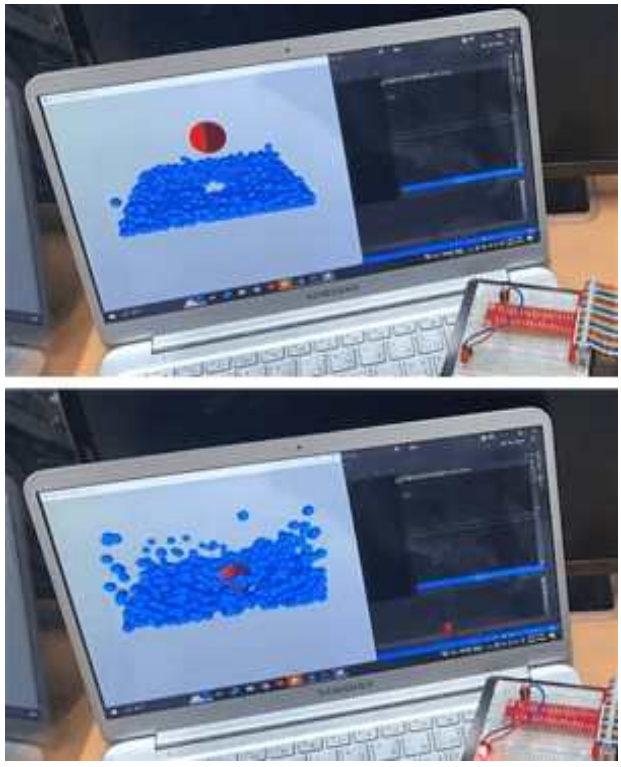

그림 8. 시뮬레이션 내의 충돌 객체가 유체의 입자와 충돌을 감지했을 때 LED 점등 및 소등
Fig. 8. LED on/off when the collision object detect collision with a particle of fluid in the simulation

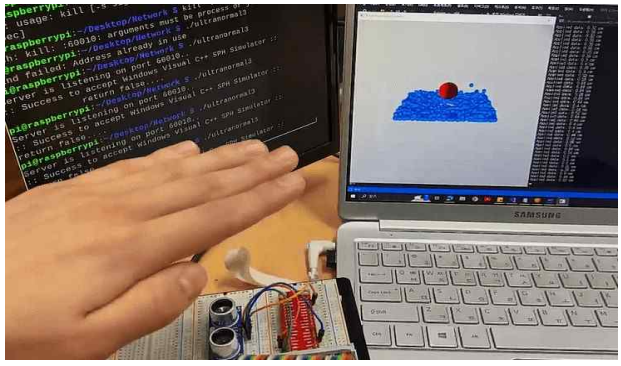

그림 9. 사용자의 신체 일부와의 거리 값에 따라 위치가 변하는 충돌 객체
Fig. 9. The collision object which is transferred the position according to the distance value the piece of user's body

시뮬레이션 내에서 특정 유체의 입자 충돌을 감지하는 충돌 객체가 유체의 입자와 서로 충돌하였을 때, 동시간대에 충돌이 감지되지 않은 유체의 입자 때문에 라즈베리파이 4가 출력하는 디지털 신호가 일정하지 않고 사용자를 혼란스럽게 하는 결과를 초래하였다. 이 부분은 앞서 소개한 데이터 패킷화 및 그룹화 하여 처리하는 최적화 전략으로 인해 실시간에

가깝게 속도가 개선된 점으로 보아 강체와 같이 단순히 독립적인 특정 물체만 충돌 대상으로 가능한 것이 아니라 유체와 같은 많은 입자를 다루는 시뮬레이션에서도 촉각 피드백 제공과 같은 사용자와의 상호작용이 가능하다는 것을 확인하였다. 시뮬레이션은 각 입자마다 충돌 데이터를 저장 및 관리하는 특정 변수가 존재해서 모든 입자에 접근하는 방식으로 구현했는데, 실제로 시뮬레이션 내 렌더링 된 모든 입자에 접근하는 것은 효율성이 좋지 않다.

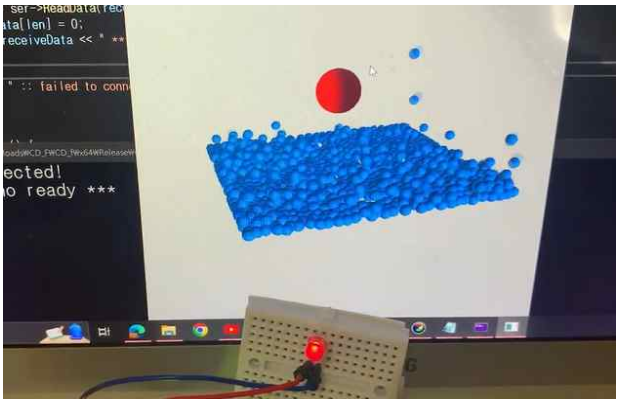

그림 10. 충돌 발생 이후, 충돌 객체와 유체의 입자가 서로 분리가 되어 있음에도 불구하고 아두이노 보드는 수신 받은 데이터를 아직 처리 중이다.

Fig. 10. After occurring collision, the Arduino board is still processing data that is sent from the simulation despite the collision object and the particle of a fluid are divided each other

만약 유체의 입자충돌 여부를 저장 및 관리하는 클래스를 유체의 입자를 구성하는 'Particle' 클래스의 부모 클래스로 하고 유체의 입자가 충돌이 감지되었을 때 해당 입자의 부모 클래스의 충돌 변수의 값을 거짓에서 참으로 재설정한다면 실제로 충돌 감지 여부를 확인하기 위해 모든 유체의 입자에 접근할 필요가 없을 뿐만 아니라 전송하는 빈도수도 낮아져 시뮬레이션 실행 속도를 크게 개선할 수 있다는 것을 기대한다. 실험에 사용되었던 초음파 센서의 모델은 'HC-SR04' 부품이다. 실제로 교육 목적으로 사용되는 초음파 거리 감지 센서이기 때문에 정확하지 않은 데이터가 측정될 가능성을 감안하고 실험을 진행한 점이 첫 번째 한계점이고, 측정 빈도수를 적절하게 지연시키지 않는다면 전체적인 시스템 속도에 영향이 가는 큰 지연이 발생된다는 것이 두 번째 한계점이다. 그러나 시스템 지연 시간을 적용해 초음파 센서의 측정 빈도수를 낮추고 시뮬레이션의 실행 속도를 개선했더라도 엔터테인먼트 콘텐츠나 교육 등 실용적인 분야에서 사용이 어려울 정도의 성능을 보이고 있다.

유체 시뮬레이션은 유체를 구성하는 입자의 개수가 많을수록 사실적인 표현이 가능하다. 현재 시뮬레이션의 구현 방식은 이웃 입자 간의 연산을 실시간으로 처리하여 결과를 도출하는 방식이고 모든 연산은 CPU를 사용하여 병렬 처리로 구현되어있다. 하지만 이러한 산술 논리 장치(ALU; Arithmetic Logic Unit) 의 개수가 현저하게 부족한 CPU에서는 고성능 CPU를 사용하더라도 입자의 개수가 많아지게 되었을 때 실시간으로 처리하지 못하고 초당 출력 수(FPS; Frame Per Second) 가 기하급수적으로 낮아지는 형태를 확인할 수 있다. 따라서 NVIDIA에서 제공하는 CUDA(Compute Unified Device Architecture) 를 사용해서 그래픽 처리 장치(GPU; Graphic Processing Unit) 를 사용한 병렬 처리를 사용하게 된다면 CPU보다 ALU의 개수가 많은 GPU로 고비용의 연산을 병렬로 처리하기 때문에 시뮬레이션의 성능을 기하급수적으로 향상시킬 수 있다. 단, 시뮬레이션 시스템을 구동하기 위해 반드시 NVIDIA 제품의 GPU가 내장되어 있어야 한다는 시스템 제한이 존재한다.

본 연구는 유체를 구성하는 많은 입자와 충돌 관련 데이터를 실시간으로 다루는 시스템에 핵심을 두고 있다. 이는 앞으로 사용자와 실시간으로 상호작용을 할 수 있는 콘텐츠를 개발하게 될 때 중점적으로 고찰해 볼 수 있는 유형이다. 예를 들어, 닌텐도 스포츠 콘텐츠 테니스에서 날아오는 테니스공을 라켓으로 타격을 했을 때 사용자가 콘텐츠에 강하게 몰입할 수 있도록 컨트롤러로부터 사용자는 감각 기관에 따라 피드백을 제공받는다[10]. 이러한 엔터테인먼트 분야에서 특히 스포츠와 같은 역동적인 영역은 인간이 감각적으로 인지할 수 있는 영역이지만, 인간이 유체와 접촉함을 감각적으로 표현해야할 때 실제로 촉감적인 표현은 스포츠와 같이 특정한 신체 활동 및 충돌 시 진동으로 촉각적인 피드백을 제공하는 방식과는 본질적으로 다른 접근을 해야 한다.

Ⅳ. 결 론

본 연구에서는 실시간으로 구동되는 유체역학 기반의 물리 시뮬레이션을 CPU 병렬 처리를 통해 실현하고, 라즈베리파이 4를 이용하여 시뮬레이션 중 발생하는 충돌 정보를 기반으로 사용자에게 즉각적인 피드백을 제공하는 데이터 처리 과정을 최적화했다. 고비용 연산이 요구되는 유체 시뮬레이션에서 발생하는 특정 이벤트를 LED의 점등 및 소등으로 실시간 처리할 수 있음을 확인함으로써, 하드웨어 기반의 실시간 처리 가능성을 입증했다. 또한, 초음파 센서를 통해 측정된 거리 값을 실시간으로 시뮬레이션에 전달하고, 이를 기반으로 시뮬레이션 내 충돌 객체의 위치를 조정함으로써, 사용자가 마우스 대신 신체의 일부를 이용해 충돌 객체를 조작할 수 있는 방법을 제시했다. 사용자에게 다양한 피드백을 제공함으로써 사용자와 직접 상호작용을 하는 HCI 적인 실감 콘텐츠 시스템을 저비용으로 구현할 수 있고, 높은 수준의 컴퓨터 사양을 사용하지 않고도 시뮬레이션과 상호작용 시스템을 실행할 수 있다는 점에서 특징이 있다.

유체와 같이 시뮬레이션을 위해 고비용의 연산이 요구되는 시스템에서 실시간으로 상호작용할 수 있다는 점은 앞으로

연구되는 실감형 콘텐츠에서 사용자에 대한 제한적이지 않은 다양한 경험을 제공할 수 있음을 시사한다. 또한, 시뮬레이션에서 표현된 유체의 형태가 더 액체의 형태에 가깝고 충돌 객체의 형태가 사용자의 손을 감지한 가상의 손 모양이었을 때 더 이상적인 결과를 낼 수 있으며 그에 대한 디테일한 개발이 필요할 수 있다. 따라서 향후 연구에서는 본 연구에서 제안한 두 가지 시스템을 하나의 콘텐츠로 결합함으로써 콘텐츠의 완성도를 높여 사용자에게 제공되는 피드백을 분석하고 만족도를 설문하는 연구를 진행할 계획이다. 또한, 사용자의 신체 일부를 보다 정확히 감지하는 립 모션(Leap Motion) 기술을 이용한 피드백 콘텐츠들이 많이 개발되고 있는데[11]-[13], 단순한 형태의 충돌 객체를 다루는 것이 아닌 실시간으로 감지하는 사용자의 손을 충돌 객체로 대체하여 시스템의 완성도를 높일 것을 기대한다.

감사의 글

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2022R1A2C1092178).

참고문헌

[1] S. H. Jeon, “Haptic Feedback Technology for Hyper-Realistic Metaverse,” *Communications of the Korean Institute of Information Scientists and Engineers*, Vol. 40, No. 4, pp. 8-13, April 2022.

[2] D.-H. Chung, “User-Based Theories and Practices on Virtual Reality,” *Informatization Policy,* Vol. 24, No. 1, pp. 3-29, March 2017. https://doi.org/10.22693/NIAIP.2017.24.1.003

[3] J. S. Lee, S. H. Kim, S. W. Jang, J. Y. Moon, and D. Y. Suh, “Metaverse Interface with Haptic and Rigid Sense Feedback at a Low Cost,” *Journal of Appropriate Technology*, Vol. 8, No. 2, pp. 91-98, August 2022. https://doi.org/10.37675/jat.2022.00171

[4] Y.-K. Park, D.-I. Kang, and H.-S. Yang, “Touch Variation due to Vibration,” in *Proceedings of the Korean Society for Emotion and Sensibility Conference*, Seoul, pp. 379-382, November 1999.

[5] J. Smart, J. Cascio, and J. Paffendorf, Metaverse Roadmap Overview: Pathways to the 3D Web, Acceleration Studies Foundation, San Pedro: CA, 2007.

[6] J. Y. Shim, D.-G. Yuk, and J. W. Sohn, “Robot Hand Control Using Haptic Glove,” *Transactions of the Korean Society for Noise and Vibration Engineering*, Vol. 33, No. 4, pp. 372-379, August 2023. https://doi.org/10.5050/KSNVE.2023.33.4.372

[7] S. Y. Yoon, S. K. Sung, and B. S. Shin, “Dental Surgery Simulation Using Haptic Feedback Device,” *KIPS Transactions on Software and Data Engineering*, Vol. 12, No. 6, pp. 275-284, June 2023. https://doi.org/10.3745/KTSDE.2023.12.6.275

[8] M. Ihmsen, J. Orthmann, B. Solenthaler, A. Kolb, and M. Teschner, “SPH Fluids in Computer Graphics,” in *Proceedings of the 35th Annual Conference of the European Association for Computer Graphics (Eurographics 2014)*, Strasbourg, France, pp. 21-42, April 2014. http://dx.doi.org/10.2312/egst.20141034

[9] How to Mechatronics. Ultrasonic Sensor HC-SR04 and Arduino - Complete Guide [Internet]. Available: https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/.

[10] M. Carrasco, N. Ortiz-Maqués, and S. Martínez-Rodríguez, “Playing with Nintendo Wii Sports: Impact on Physical Activity, Perceived Health and Cognitive Functioning of a Group of Community-Dwelling Older Adults,” *Activities, Adaptation & Aging*, Vol. 44, No. 2, pp. 119-131, 2020. https://doi.org/10.1080/01924788.2019.1595261

[11] S. B. Yang, J. Y. Jang, and S. H. Kim, “Virtual Reality Game Interface Using Leap Motion Sensor and Arduino,” in *Proceedings of Korea Software Congress (KSC 2019)*, Pyeongchang, pp. 1343-1345, December 2019.

[12] H. Joo, M. Cho, S. K. In, K. Cho, and J.-K. Min, “Development of Baseball Game Using Leap Motion Controllers,” *KIISE Transactions on Computing Practices*, Vol. 21, No. 5, pp. 343-350, May 2015.

[13] S. Park, S. Park, and J.-H. Kim, “Leap-Motion Based Tracking Framework for Practice and Analysis of User’s Arm Muscle,” in *Proceedings of the 62nd Summer Conference of Korean Society of Computer Information*, Jeju, pp. 469-472, July 2020.

이건희(Geon-Hee Lee)

2024년 : 강남대학교 소프트웨어응용학부(공학사)

※관심분야 : 가상현실(Virtual Reality), 휴먼 컴퓨터 인터랙션(Human Computer Interaction), 등

최웅(Woong Choi)

2000년 : 조선대학교 대학(공학석사)
2005년 : Tokyo Institute of Technology(공학박사-지능시스템과학)
2005년～2010년: Ritsumeikan University
2010년～2022년: National Institute of Technology, Gunma College
2022년～현　재: 강남대학교 ICT융합공학부 부교수
※관심분야 : 가상현실(Virtual Reality), 휴먼 컴퓨터 인터랙션(Human Computer Interaction), 등