2022년 2월
석사학위 논문

삼각함수를 활용한 크리에이티브 코딩 웹 어플리케이션 개발

조선대학교 대학원

창의공학디자인융합학과

김 현 주

삼각함수를 활용한 크리에이티브 코딩 웹 어플리케이션 개발

Developing a Creative Coding Web Application
Using a Trigonometry Function

2022년 2월 25일

조선대학교 대학원

창의공학디자인융합학과

김 현 주

삼각함수를 활용한 크리에이티브 코딩 웹 어플리케이션 개발

지도교수 김 병 욱

이 논문을 디자인학석사학위 신청논문으로 제출함

2021년 10월

조선대학교 대학원

창의공학디자인융합학과

김 현 주

김현주의 석사학위논문을 인준함

위원장 조선대학교 교수 **이 진 렬**

위 원 조선대학교 교수 **김 병 욱**

위 원 조선대학교 교수 **손 장 완**

2021년 12월

조선대학교 대학원

목 차

표목차

그림목차

ABSTRACT

Developing a Creative Coding Web Application Using a Trigonometry Function

Kim, Hyeonju
Supervisor : Prof. Kim, Byounguk
School of Design & Creative Engineering,
Graduate School of Chosun Univ.

Designers use a computer with GUI-based software, but there has been a stream of movement to utilize a computer beyond software. The spread of coding education has increased the demand for creative coding, as well. In a coding process, mathematical algorithms are used to express the movements of nature-based on physics. A trigonometric function is one of the most popular algorithms for that purpose. Designers, however, have a difficult time understanding materials focused on mathematical aspects too much. They need an approach to understand the characteristics of a mathematical algorithm and have easy access to one.

Against this backdrop, this study aimed to develop a creative codin g web application based on a trigonometric function so that design ers could understand it with ease. For that purpose, the investigator analyzed the cases of web animation coding algorithm web applicat ions that were the most familiar to designers and set the screen an d menu composition of the developed creative coding web applicati on(CCWA). Also analyzed in the study were 200 creative coding w orks to figure out the patterns of designers using a trigonometric fu nction. The analysis results were used to organize a menu to allow designers to recognize the utilization patterns of a trigonometric fun ction and choose one that wanted.

In Short, the creative coding web application developed in the stud y is based on a trigonometric function and characterized by suppor ts for open-source coding and intuitive GUI. Using this application,

designers will be able to apply a trigonometric function to their co ding easily with the intuitive GUI and the menu organized around their needs.

제 1장
서 론

제 1장
서 론

제 1절 | 연구배경 및 목적

컴퓨터는 이제 디자인 분야에서 빼놓을 수 없게 되었다. Adobe사의 소프트웨어들은 디자인 분야에서 필수적인 도구로써 GUI 기반으로 효율적인 작업을 할 수 있도록 한다. 그러나 소프트웨어를 넘어 컴퓨터를 적극적으로 활용하고자 하는 움직임은 이어져 왔으며 코딩교육이 확산되면서 크리에이티브 코딩의 수요도 함께 증가하고 있다. 크리에이티브 코딩은 디자이너가 프로그래밍을 활용하여 자신의 아이디어를 확장할 수 있다는 점에서 디자이너에게 각광받고 있다. 실시간으로 미디어를 사용하고 변화시키는 컴퓨터의 처리 능력은 창의적인 아이디어의 원천을 형성한다.[1] 존 마에다(John Maeda)는 그의 저서 『Design by Numbers』(2001)에서 컴퓨터를 이용하여 성공적으로 디자인하기 위해서는 자신이 사용하는 프로그램을 설계하거나 적어도 완벽하게 이해할 수 있어야 한다고 말했다.

프로그래밍에서 수학적 알고리즘은 물리학에 기초하여 자연법칙의 움직임을 표현하기 위하여 사용된다. 그 중 삼각함수 알고리즘은 유용하게 사용되어 자주 마주하는 알고리즘이다. 그러나 디자이너들에게 익숙해지기 힘든 영역 중 하나이며 삼각함수 알고리즘을 어려워하는 이유는 크리에이티브 코딩에 활용되는 특성을 알지 못하거나 자료가 수학적인 부분에 지나치게 초점이 맞춰져 있기 때문이다. 이에 삼각함수 알고리즘의 특성을 인지하고 쉽게 접근할 수 있는 방법이 필요하다.

이에 본 연구는 크리에이티브 코딩에서 삼각함수를 쉽게 활용할 수 있는 방법을 제안하는 것을 목표로 한다. 크리에이티브 코딩 작품에서 삼각함수를 활용하

1) Andrew Richardson, Date-driven Graphic Design: Creative Coding for Visual Communication, 2017, pp.10

는 영역을 살펴보고 분류하여 삼각함수를 활용한 크리에이티브 코딩 웹 어플리케이션을 개발하고자 한다.

제 2절 | 연구내용 및 방법

본 연구는 크리에이티브 코딩에 있어 삼각함수 알고리즘 웹 어플리케이션을 개발하는 것을 목적으로 다음과 같은 방법으로 연구를 진행하였다.

제 1장 서론

연구의 배경과 목적 및 연구 방법을 서술하였다.

제 2장 크리에이티브 코딩에서 삼각함수의 이해

문헌연구를 통해 크리에이티브 코딩의 기본 개념과 크리에이티브 코딩에서 삼각함수를 설명하고 있는 방식에 대해 살펴보았다. 디자이너가 활용할 수 있는 삼각함수의 특성을 조사하였다.

제 3장 삼각함수 활용 방법 제안을 위한 사례연구

크리에이티브 코딩 웹 어플리케이션 개발을 위하여 디자이너에게 웹 애니메이션 코딩 알고리즘을 제안하는 시스템들의 방식을 조사하였다. 사례를 분석하여 기존 제안의 기본 구성을 파악하였다. 또한 애니메이션에 변화를 주는 파라미터의 구성을 파악하였다. 다음으로, 크리에이티브 코딩 작품 중 삼각함수를 활용한 사례 200개를 수집하여 삼각함수 적용 유형에 따라 작품을 분류하였다.

제 4장 크리에이티브 코딩 웹 어플리케이션 개발

웹 어플리케이션의 특징을 정의하고 3장에서 도출한 시스템의 필수 구성을 통하여 기본 화면을 구성하였다. 3장의 사례에서 도출한 유형에 따라 메뉴를 구성하고 웹 어플리케이션 적용 사례를 제시하였다.

제 5장 결 론

연구의 발견점과 시사점 및 향후 연구 과제에 대하여 서술하였다.

제 2장

크리에이티브 코딩에서 삼각함수의 이해

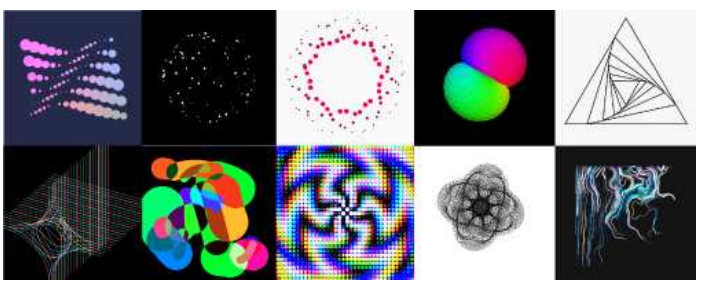
제 2장
크리에이티브 코딩에서 삼각함수의 이해

제 1절 | 크리에이티브 코딩의 개념

1. 크리에이티브 코딩의 개념

1) 크리에이티브 코딩의 개념

크리에이티브 코딩은 소프트웨어, 코드 및 계산 프로세스를 사용하여 표현하고자 하는 것을 만들어내는 것이다. 컴퓨터를 창의적인 목적으로 사용하는 개념은 1960년대부터 있었지만, 예술 프로젝트에 프로그래밍을 사용하는 것에 대한 관심이 높아지면서 크리에이티브 코딩이라는 용어는 지난 15년부터 사용되기 시작했다. 크리에이티브 코딩은 주로 기능적인 목적보다 표현적인 목적으로 코드를 작성하는 것으로 기술과 예술의 결합이다. 일반적인 프로그래밍 언어는 텍스트를 입력하고 그 결과를 다시 텍스트로 출력하는 방식인 반면 창의적인 코딩언어들은 텍스트 입력을 통해 시각적인 결과물을 만들어 낸다.[2] 디자이너는 크리에이티브 코딩을 통해 단순한 시각적 표현을 넘어 반응하거나 물리적으로 움직이는 결과물을 만들어 낼 수 있다.



[그림 2-1] 크리에이티브 코딩 작품 이미지[3]

2) 김혜란, 예술 활동을 통한 창의적 코딩에 관한 연구 - 교육 프로그램 사례를 기반으로 , 한국영상학회논문집, 19(4), 2021, p.26

2) 크리에이티브 코딩 언어 종류

[표 2-1] 크리에이티브 코딩 언어 종류

이름	특징	언어
OpenFrameWorks	단순하고 직관적인 크리에이티브 코딩을 위해 디자인된 오픈 소스 프레임	C++
Vvvv	손쉬운 프로토타이핑 및 개발을 위한 시각적/텍스트 라이브 프로그래밍 환경	비주얼 프로그래밍 언어
Processing	프로그래밍을 시각적 개념으로 교육할 목적으로 개발된 오픈 소스 프로그래밍 언어이자 통합 개발 환경	Java
p5.js	웹에서 코딩하고 창의적으로 표현할 수 있도록 돕는 플랫폼 프로세싱을 기반으로 함	JavaScript
Three.js	웹 페이지에 3D 객체를 쉽게 렌더링하도록 돕는 라이브러리	JavaScript

프로세싱(Processing)은 크리에이티브 코딩에 사용되는 가장 대중적인 언어 중 하나이다. 그래픽 디자이너이자 프로그래머인 존 마에다(John maeda)는 시각적인 표현을 위한 프로그래밍 언어 DBN를 개발하였고 이는 제자인 케이시 리아스(Casey Rea)와 벤 프라이(Ben Fry)에 의해 자바(Java) 기반 오픈소스 프로그래밍 언어 프로세싱(Processing)으로 발전하였다. 그 이후 웹과 연결을 위해 프로세싱(Processing)을 기반으로 하는 자바스크립트(Javascript) 프레임 p5.js이 개발되었다. 그 이외에도 C++ 언어를 사용하는 Openframeworks와 비주얼 프로그래밍 방식인 Vvvv, WebGL을 이용하여 3D그래픽을 만드는 자바스크립트 라이브러리 Three.js 등이 있다.

본 논문에서는 오픈 소스 프로그래밍 언어인 프로세싱(Processing)과 p5.js를 기반으로 하여 삼각함수 활용법을 제안하고자 한다.

3) Openprocessing, “Openprocessing - Creative Coding for Curious Mind.” Openprocessing, accessed September 29, 2021, https://openprocessing.org/

제 2절 | 크리에이티브 코딩에서 삼각함수에 대한 이해

1. 크리에이티브 코딩에서 삼각함수의 개념

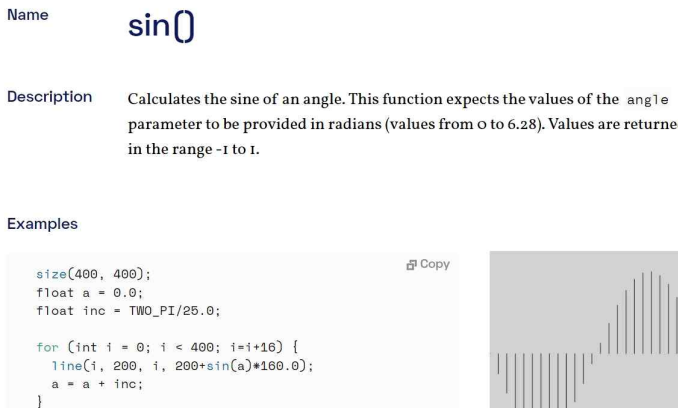
프로세싱은 삼각함수 파트에서 acos, asin, atan2, atan, cos, sin, tan 함수를 제공하고 있다. 이 중 가장 많이 쓰이는 함수는 sin, cos. atan2 이다. 프로세싱의 레퍼런스(reference)에서 cos과 sin 함수는 각도의 코사인과 사인을 계산하며 이 값은 -1 에서 1 범위 사이에 있다고 설명한다. atan2 함수는 단순히 각도를 계산하는 함수이므로 본 연구에서 제외하였다.

[표 2-2] Processing 레퍼런스의 삼각함수[4)]

함수명	설명
acos()	cos()의 역함수로, 아크 코사인을 반환
asin()	sin()의 역함수로, 아크 사인을 반환
atan2()	지정된 점에서 좌표 원점까지의 각도(라디안) 계산
atan()	tan()의 역함수로, 아크 탄젠트를 반환
cos()	각도의 코사인 계산
sin()	각도의 사인 계산
tan()	각도의 사인과 코사인의 비율 계산

이는 단순히 제공되는 기능들의 나열로 삼각함수의 특성이나 활용 방법은 기술되어 있지 않다.

4) Processing, “Processing.” Referece, accessed September 10, 2021, https://processing.org/reference



[그림 2-2] Processing 레퍼런스의 sin() 설명[5)]

다니엘 쉬프만의 'Nature of code'에서는 진동(Oscillation) 파트에서 삼각함수를 다룬다. 이 서적에서 삼각함수는 컴퓨터 그래픽의 기초이며 각도를 계산하거나 두 점의 거리를 구하거나 원을 사용할 때 항상 사용된다고 설명한다. 이는 진동, 파동, 극좌표 변환, 움직임의 방향을 만든다고 설명하였다.[6)]

프로세싱을 개발한 케이시 리아스(Casey Reas)와 벤 프라이(Ben Fry)의 '프로세싱 교과서'에서는 사인과 코사인은 물결, 원을 그리고 방향을 설정하기 위해 사용한다고 설명한다.[7)]

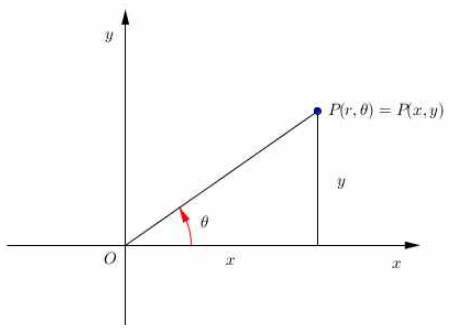
2. 크리에이티브 코딩에서 삼각함수의 특성

크리에이티브 코딩에서 사용하는 삼각함수의 특성은 크게 총 3가지이다. 첫 번째는 극좌표를 사용하기 위해서이다. 두 번째는 애니메이션의 가속도를 설정하는 것이다. 세 번째는 삼각함수는 주기함수로 애니메이션에서 주기를 활용하기 위한 것이다.

5) Processing, "Processing." Referece, accessed September 10, 2021, https://processing.org/reference/sin_.html
6) 다니엘 쉬프만, Nature of Code, 한빛미디어, 2015, pp.145-173
7) 케이시 리아스 외 1, 프로세싱 교과서, 유엑스리뷰, 2018, pp.271-279

1) 극좌표와 삼각함수

프로세싱의 좌표 시스템은 x, y 좌표로 픽셀의 위치를 지정해야 한다. 이 좌표를 데카르트 좌표라고 한다. 극좌표 시스템은 위치를 각도와 반지름을 통해 나타낸다. 극좌표계를 사용하면 원형이나 곡선 경로를 따라서 물체를 배열하거나 이동하는 것을 각도와 반지름이라는 변수만으로 쉽게 만들 수 있다.



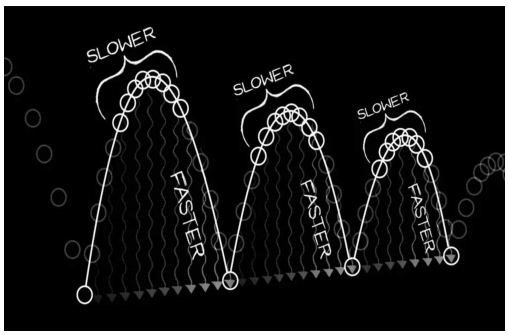
[그림 2-3] 극좌표와 데카르트 좌표의 관계[8)]

2) 가속도와 삼각함수

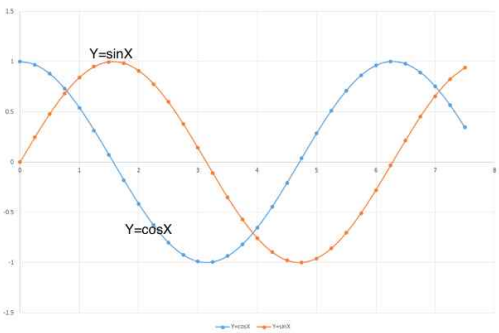
움직임은 공간과 시간을 전제로 한 개념으로서, 정해진 프레임 안에서 그래픽 요소들을 적절히 사용하여 움직임을 부여하게 된다.[9)] 그 중 삼각함수는 시간과 연관이 있다. 움직임의 기본서로 불리는 프랭크 토머스(Frank Thomas)와 올리 존스턴(Ollie Johnston)의 『 The Illusion of Life 』(1981)에서 제시한 12가지의 원칙 중 'Slow In and Slow Out'과 'Timing'에서 삼각함수 알고리즘과의 유사성을 발견하였다. 'Slow In and Slow Out'은 물체에 감속과 가속의 원리를 구현함으로써 현실 세계와 비슷한 움직임을 표현하는 것이며 'Timing'은 어떠한 동작을 할 때 소요되는 시간을 말한다. 두 원칙은 스페이싱(Spacing)과 관련된다. 스페이싱은 움직임이 일정한 시간 동안 일어났을 때 그 사이의 간격을 뜻한

8) 수학과 사는 이야기, 2015. 5. 28, 이미지, https://suhak.tistory.com/300
9) 함혜연, 모션그래픽 움직임의 시지각 효과 연구, 중앙대학교 석사학위논문, 2004, p.6

다. 스페이싱을 동등한 간격으로 나눈다면 물체의 움직임이 일정하며 스페이싱의 간격을 점차 넓혀주거나 좁혀준다면 감속, 가속을 적용할 수 있다. 마찬가지로 'Slow In and Slow Out'에서 스페이싱을 넓혀주거나 좁혀주면 감속, 가속을 조절할 수 있다. 감속은 속도의 값이 줄어드는 것이고 가속은 속도의 값이 늘어나는 것이다. 그림 2-4의 'Slow In and Slow Out', 'Timing'의 그래프와 그림 2-5 삼각함수의 sin 그래프는 유사한 모습을 보인다. 가속도는 속도의 값을 변화시키는 것을 의미하므로 삼각함수는 가속도를 설정할 수 있다는 특성을 가지고 있다. 이는 삼각함수의 각도 값에 변수를 더하거나 곱하여 움직임의 가속도를 조절할 수 있다는 의미를 가진다.



[그림 2-4] Timing and Spacing in Animation[10)]



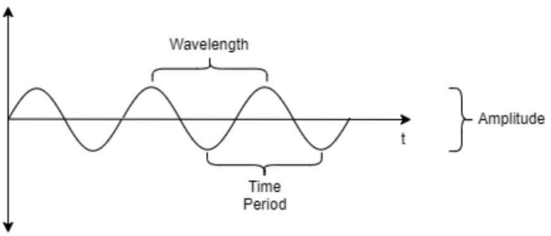
[그림 2-5] Y = sinX과 Y =cosX의 그래프[11)]

10) TED-Ed. "Animation basics: The art of timing and spacing" January 29, 2014. video. 4:10. https://www.youtube.com/watch?v=KRVhtlVkQWRs

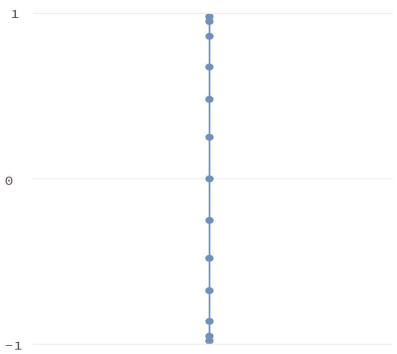
11) "SINE AND COSINE GRAPHS", fizzics.org, accessed November 25, 2021, https://www.fizzics.org/sine-and-cosine-graphs/

3) 주기와 삼각함수

그림 2-5에서 보여지는 Y = sinX, Y = cosX 그래프는 2π의 주기를 가진다. 주기는 반복되는 패턴의 길이를 말한다. 이 식은 각도를 매개변수로 설정하고 각도 값이 증가하거나 감소하면 주기 2π마다 반복적으로 그래프를 그린다. 그림 2-6와 같이 삼각함수는 −1에서 1의 진폭 값을 가진다. 이는 −1과 1 사이를 주기적으로 움직일 수 있음을 의미한다. 움직임의 범위를 변경하고 싶다면 이 값에 범위 값을 곱하면 된다. 예를 들어 −50에서 50까지 주기적으로 이동시키고 싶다면 50을 곱해주기만 하면 된다. 이 값을 물체의 위치로 지정하면 두 지점을 주기적으로 움직이도록 만든다.



[그림 2-6] 삼각함수의 진폭(Amplitude)과 주기(Period)[12]

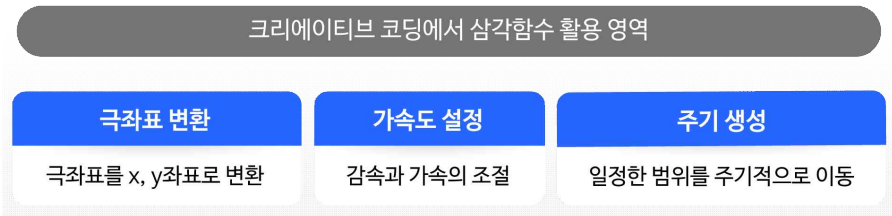


[그림 2-7] sin() x축 압축 그래프

12) Beny Basumatary, “Sound Waves: Nauture, Speed, Reflection Of Sound, Intensity, Loudness, Reverberation, Superposition.” testbook, Jun 10, 2021, https://testbook.com/learn/physics-sound-waves/

그림 2-7는 그림 2-5의 그래프에서 x축을 압축시킨 모습이다. x축이 고정되었을 때 y축의 속도변화를 보여준다. 삼각함수는 항상 주기적 특성을 가짐과 동시에 가속도를 적용할 수 있으며 변수의 설정으로 값을 조절할 수 있다.[13)]

크리에이티브 코딩에서 삼각함수를 활용할 수 있는 특성은 다음과 같다.



[그림 2-8] 크리에이티브 코딩에서 삼각함수 활용 특성

13) 케이시 리아스 외 1, op.cit, 2015, pp.304

제 3장

삼각함수 활용 방법 제안을 위한 사례연구

제 3장
삼각함수 활용 방법 제안을 위한 사례연구

3장의 1절에서는 삼각함수의 활용 방법 제안을 위하여 디자이너에게 애니메이션 코딩 알고리즘을 제안하는 시스템을 조사하여 필수 구성을 도출하였다. 2절에서는 카테고리를 구성하기 위하여 크리에이티브 코딩 작품이 삼각함수를 활용한 사례를 분석하여 카테고리를 정의하였다.

제 1절 | 활용 방법 제안 방식의 사례 연구

본 연구의 목적인 삼각함수 활용 웹 어플리케이션의 구성의 단서를 얻기 위하여 디자이너들이 코딩을 가장 많이 접하는 분야인 웹 프론트엔드에서 애니메이션을 위한 알고리즘을 제안하는 시스템을 수집하였다. 사례 수집을 기반으로 디자이너에게 알고리즘을 제안할 때 필요한 구성과 메뉴 분류 기준, 파라미터의 구성을 도출하였다.

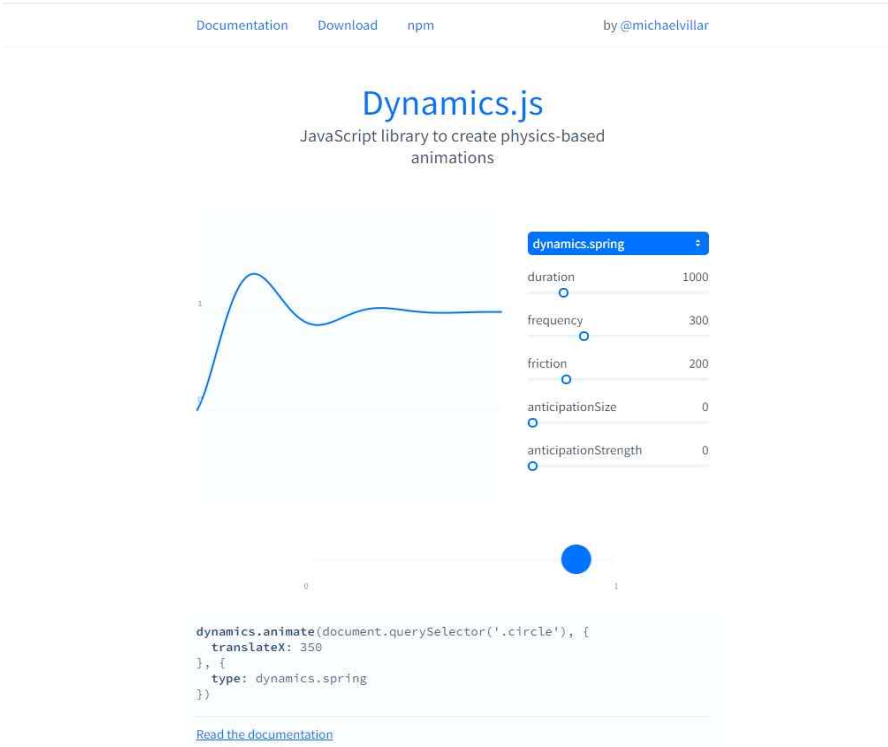
1) Dynamics.js[14)]

물리를 기반으로 움직임에 적용할 수 있는 알고리즘을 위한 자바스크립트 라이브러리이다. 움직임 값을 그래프로 표현하였고 드롭 다운 메뉴를 클릭하면 애니메이션의 움직임에 따른 카테고리를 선택할 수 있다. 메뉴의 구성은 움직임의 특성에 따라 정의하였다. 각 파라미터를 조정하여 그래프를 변형시키고 애니메이션을 수정할 수 있다. 파라미터의 구성은 메뉴마다 다르지만 기본적으로 duration(지속 시간), frequency(진동수)[15)], friction(마찰)로 구성되었다. 그래프와 파라미터 슬라이더 아래에는 물체에 움직임을 적용한 모습을 보여준다. 다음

14) http://dynamicsjs.com/

15) 주기가 단위시간에 몇 번 반복되었는지를 뜻하는 말이다.

에는 적용할 수 있는 코드가 있으며 버튼을 통해 코드 파일을 다운로드 할 수 있도록 하였다. 마지막으로는 라이브러리를 사용한 예제를 보여준다.

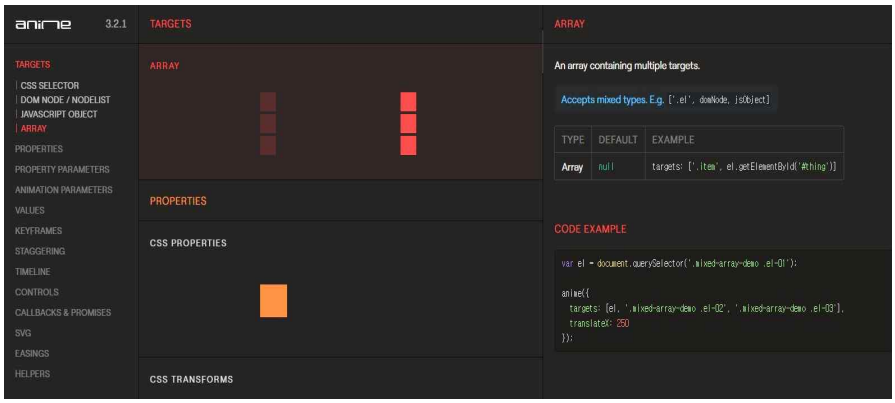


[그림 3-1] Dynamics.js 라이브러리 제안 사이트의 구성

2) Anime.js[16)]

애니메이션의 쉬운 구현을 위해 만든 자바스크립트 라이브러리이다. 제일 좌측에는 애니메이션이 적용되는 요소에 따라 분류한 메뉴가 있다. 메뉴를 선택하면 해당 코드를 적용하였을 때 표현되는 움직임을 보여준다. 우측에는 가벼운 설명과 함께 적용된 코드, 예시에 적용된 값의 수치를 보여준다.

16) https://animejs.com/



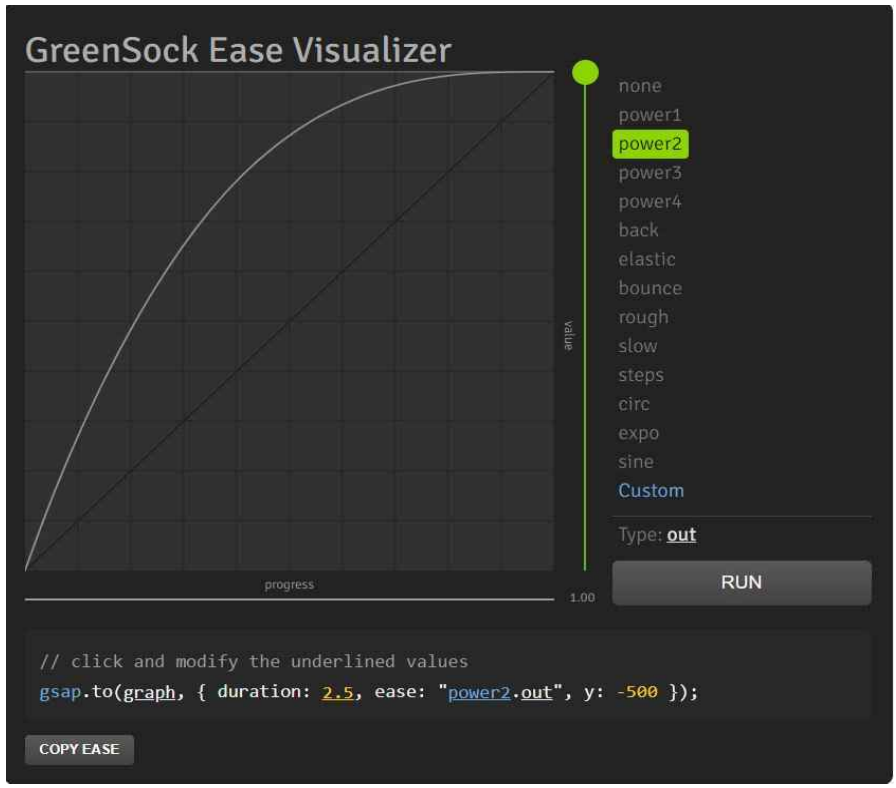
[그림 3-2] Anime.js 라이브러리 제안 사이트의 구성

3) GreenSock[17)]

애니메이션을 쉽게 사용할 수 있는 타임라인 기반의 자바스크립트 라이브러리이다. 가장 유명한 애니메이션 라이브러리 중 하나로 기능이 방대하여 Ease 메뉴를 중심으로 살펴보았다.

GreenSock의 Ease 메뉴에서는 움직임 라이브러리를 쉽게 적용할 수 있도록 'Ease Visualizer'를 제공한다. 움직임 속도의 특성에 따라 이름을 정의하고 기본형을 제안했다. 좌측의 속도 그래프는 시간 진행에 따라 속도가 변화하는 모습을 보여준다. 그래프 옆에는 해당 움직임을 물체에 적용한 모습을 표현했다. 아래에는 해당 움직임을 적용할 수 있는 코드를 보여준다.

17) https://greensock.com/docs/v3/Eases

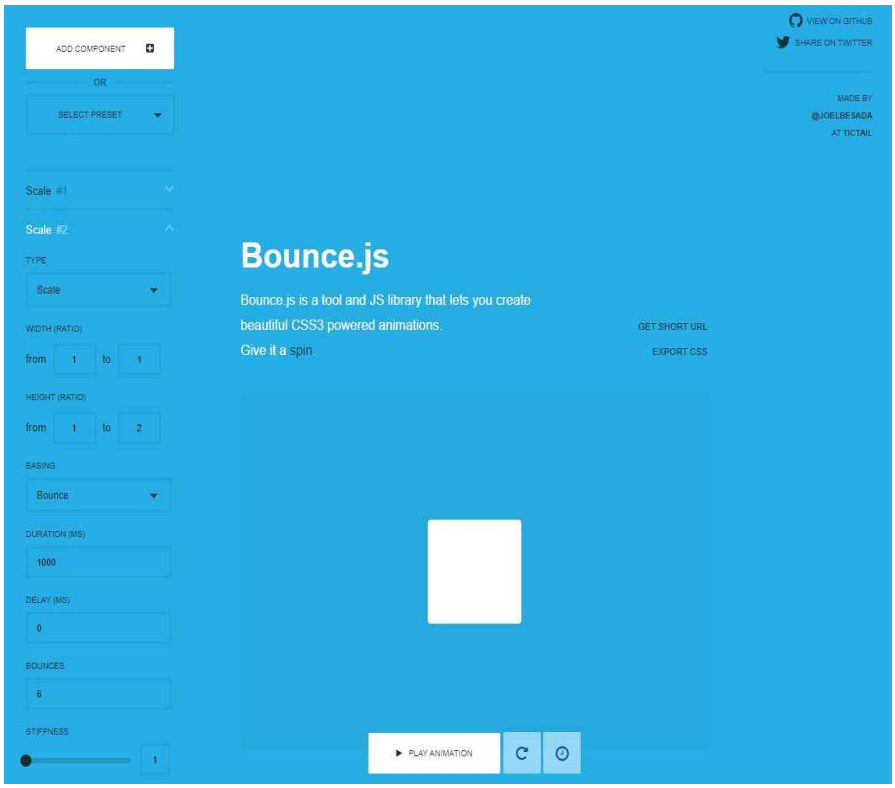


[그림 3-3] GreenSock 라이브러리의 Ease Visualizer

4) Bounce.js[18)]

자바스크립트로 쉽게 CSS 애니메이션을 만드는 라이브러리이다. 좌측의 메뉴에서 컴포넌트를 추가한 뒤 움직임 유형을 선택하면 적용된 움직임이 재생된다. duration(지속 시간), width, height(움직임의 범위), Easing(속도 변화 값) 등의 파라미터를 제공하여 움직임을 조정할 수 있다. Easing 파라미터는 4가지의 기본 패턴을 제공하여 적용할 수 있도록 했다. 하지만 주어진 Easing만 사용할 수 있다는 점이 아쉬웠다. EXPORT CSS 버튼을 이용하여 움직임을 적용하기 위한 소스코드를 다운받을 수 있다.

18) http://bouncejs.com/

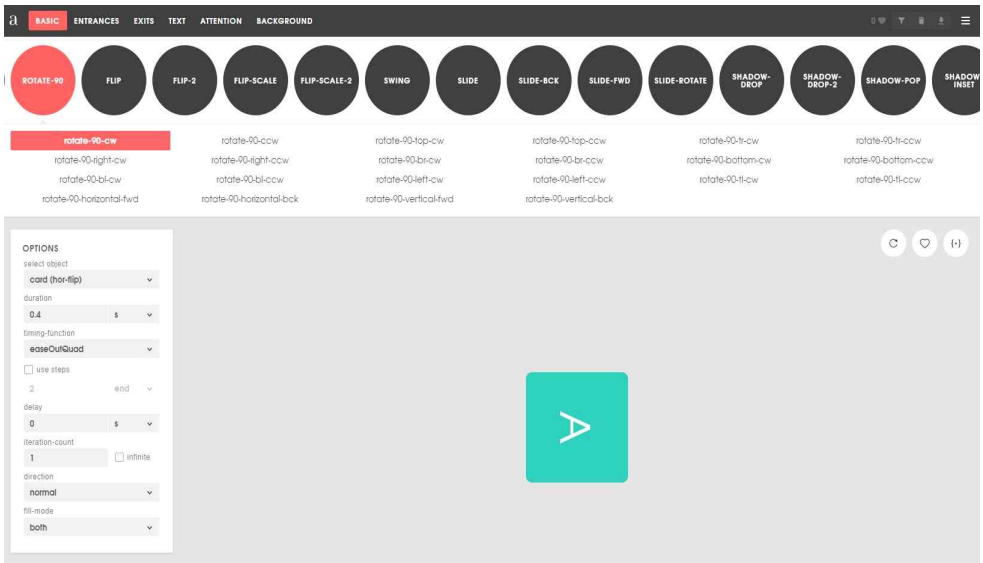


[그림 3-4] Bounce.js 라이브러리 제안 사이트의 구성

5) animista[19)]

CSS 애니메이션을 쉽고 간단하게 만들 수 있는 웹 사이트이다. 상단의 메뉴는 움직임을 적용하는 요소에 따라 구성되어 있으며, 하위 메뉴는 움직임의 유형으로 구성되어 있다. 좌측의 duration(지속 시간), direction(방향), select object(움직임을 적용할 물체), 시간함수(timing function) 등의 파라미터들을 통해 움직임 값을 조정할 수 있다. 시간함수(timing function)는 Easing, 즉 속도 변화 값과 같다. 다양한 속도 곡선의 이름을 정의하고 이를 적용하도록 했다. 버튼을 통해 해당 움직임을 구현하는 코드를 복사할 수 있다.

19) https://animista.net/



[그림 3-5] animista 사이트의 구성

사이트의 공통적인 구성 요소는 물체에 움직임을 적용한 시뮬레이션, 애니메이션에 적용한 코드이다. 이외에 움직임 속도 그래프, 움직임 조정 파라미터, 적용 사례 등이 있다.

[표 3-1] 애니메이션 알고리즘 제안 사이트 구성 요소

시스템 명	속도 그래프	조정 파라미터	시뮬레이션	코드	적용 사례
Dynamics.js	○	○	○	○	○
Anime.js			○	○	
GreenSock	○	○	○	○	
Bounce.js		○	○	○	
animista		○	○	○	

공통적인 구성 요소인 시뮬레이션과 코드는 필수적인 요소이며, 나머지 요소들은 애니메이션의 이해를 돕기 위해 추가적으로 구성할 수 있다.

다음은 각 사이트의 메뉴 분류 기준을 살펴보았다. 움직임의 중요한 요소인 속

도 값의 설정에 따른 분류와 움직임이 적용되는 요소에 따른 분류 두 가지로 도출되었다.

[표 3-2] 애니메이션 알고리즘 제안 사이트 카테고리 분류 기준

시스템 명	카테고리 분류 요인
Dynamics.js	적용하는 속도 값의 설정에 따라 분류
Anime.js	움직임이 적용되는 요소에 따라 분류
GreenSock	적용하는 속도 값의 설정에 따라 분류
Bounce.js	적용하는 속도 값의 설정에 따라 분류
animista	움직임이 적용되는 요소에 따라 분류한 뒤 움직임 유형에 따라 분류

마지막으로 각 사이트의 파라미터 구성을 정리하였다. 파라미터는 지속기간(Duration), 주기(Frequency), 가동 범위(Width) 등으로 움직임의 속도 곡선을 조정할 수 있게 구성하거나 Easing 파라미터를 통해 정의한 속도 곡선을 적용할 수 있도록 구성되었다.

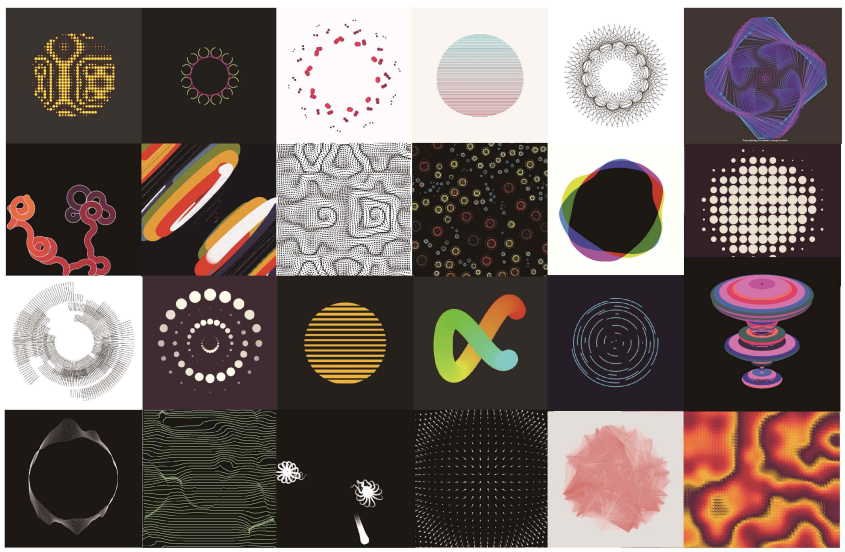
[표 3-3] 애니메이션 알고리즘 제안 사이트 파라미터 기본 구성

시스템 명	파라미터 구성
Dynamics.js	지속기간(Duration), 진동수(Frequency), 마찰(Friction) 등
Anime.js	노드 없음
GreenSock	파라미터 없음
Bounce.js	애니메이션의 유형(Type), 가동 범위(Width, Height), 속도 곡선 적용(Easing), 지속기간(Duration) 등
animista	애니메이션 적용 물체(Object), 지속 기간(Duration), 속도 곡선 적용 (Timing function), 방향(Direction) 등

파라미터는 움직임의 속도 곡선을 조정할 수 있게 구성되도록 삼각함수의 공식을 활용하여 조정할 수 있도록 설정했다.

제 2절 | 삼각함수 활용 패턴 도출을 위한 사례 연구

디자이너들이 삼각함수를 활용하는 유형을 도출하기 위해서는 기존의 크리에이티브 코딩 작품에서 삼각함수를 사용하고 있는 사례의 분석이 필요하다. 프로세싱과 p5.js를 이용한 크리에이티브 코딩 작품을 공유하는 커뮤니티 오픈프로세싱(www.openprocessing.org) 에서 사례를 수집했다. 2021년 3월 기준으로 총 74338 작품 중 좋아요를 많이 많은 상위 1000개의 작품에서 삼각함수를 활용한 작품을 수집하였다. 이 중 총 389개의 작품이 삼각함수를 사용하였다. 1000개의 작품 중 삼각함수를 활용한 작품이 389개라는 점에서 삼각함수 알고리즘은 크리에이티브 코딩에 필수적임을 다시 확인하였다. 389개의 작품에서 같은 작가가 시리즈로 만든 작품과 오류가 발생하는 작품을 제외하고 총 200개의 작품을 수집하였다.



[그림 3-6] 크리에이티브 코딩 작품 사례 수집 예시

첫 번째 분류는 디자이너가 삼각함수를 쉽게 이해하는 것을 목적으로 시각적으로 패턴을 형성하는 요소에 따라 '라인으로 패턴 형성', '원으로 패턴 형성', '기

타 도형으로 패턴 형성', '파티클 패턴 형성'으로 분류한 뒤 패턴을 만드는 방법에 따라 재분류 하였다. 그러나 이 분류는 삼각함수 소스코드의 기본 유형을 도출할 수 없었다.

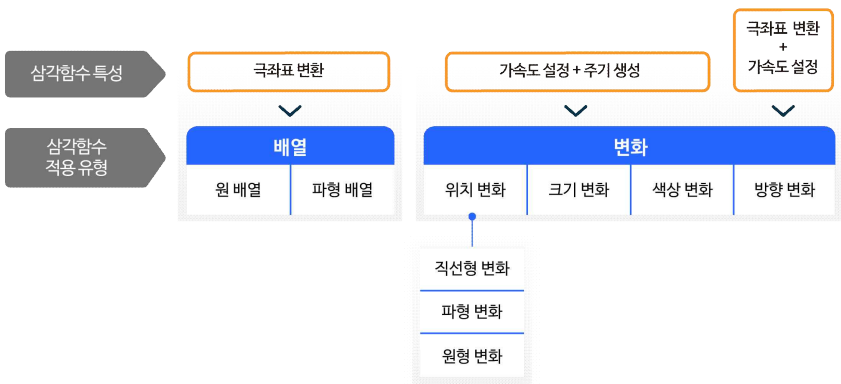
분류 기준	하위 분류	No	수	
라인으로 패턴 형성	라인을 움직임	13, 16, 17, 21, 24, 27, 36, 40, 51, 60, 61, 73, 76, 82, 86, 92, 101, 104, 107, 111, 112, 113, 117, 140, 144, 147, 148, 150, 154, 161, 162, 169, 173, 178, 181, 184, 188, 192, 207, 232, 193, 221, 293, 249, 211, 178, 198, 292, 188, 236, 316, 154, 286, 254, 272, 247	53	
	라인이 그려짐	2, 10, 20, 29, 30, 34, 63, 68, 70, 71, 72, 74, 75, 108, 130, 134, 142, 152, 277, 256, 263, 208, 299, 206, 302, 275, 284, 142, 183, 287, 214, 226, 282, 108, 195, 303, 306	35	
	인터랙션 반응	1, 9, 45, 94, 95, 102, 126, 274, 255, 315, 268, 194	12	100
원으로 패턴 형성	원이 움직임	69, 52, 213, 119, 143, 46, 115, 157, 79, 131, 98, 218, 237	13	
	원이 그려짐	172, 170, 145, 12, 53, 317, 83, 39, 291, 176	10	
	원의 크기 변화	64, 32, 96, 99, 88, 90, 220, 43, 49, 55, 209	11	
	원으로 다른 패턴 생성	191, 233, 262, 243, 241, 100, 311, 65, 133, 301, 48, 149	12	
	인터랙션 반응	167, 196, 50, 18, 202, 261, 264, 127, 253, 295, 174, 35, 41, 210	14	60
기타 도형으로 패턴 형성	기타 도형이 원 형태를 이룸	267, 307, 93, 297, 230, 216, 77, 124, 279, 120, 31	11	
	움직이는 패턴 그림	223, 239, 5, 190, 240, 66, 189, 245, 165, 304, 116, 276, 187, 54, 80, 310, 103	17	
	정적인 패턴 그림	175, 44, 227, 136, 123, 139, 163, 8, 225, 129, 219, 212, 234	13	
	인터랙션 반응	300, 33, 278, 42, 283, 171, 180, 118	8	49
파티클 패턴	파티클 흐름	294, 151, 135, 105, 37, 280, 201, 91, 224, 6, 285, 266, 260, 11, 146, 89, 114, 14, 26, 251	20	
	인터랙션 반응	81, 257, 296, 231, 106, 7, 235, 22, 138, 305	10	
	기타	185, 78, 110, 197, 28, 122, 137, 271, 290	9	39
패턴 없음	인터랙션 반응 없음	265, 85, 4, 270, 47, 62, 84, 258, 244, 281, 156, 205, 203	13	
	인터랙션 반응	19, 15, 182, 312, 121, 199, 215, 168, 308, 179, 217, 204, 153	13	26

[그림 3-7] 패턴을 이루는 요소에 따라 사례 분류

최종분류는 소스코드를 분석하여 삼각함수 특성이 적용되는 요소에 따라 분류하고 애니메이션의 유형에 따라 재분류하였다. 삼각함수가 변화시키는 요소에 따라 배열, 변화 2개의 카테고리로 분류하였다. 변화시키는 영역이 좌표의 배열이면 배열, 애니메이션이면 변화로 정의하였다. 배열은 극좌표의 특성을 활용하였으며 변화는 가속도 설정과 주기 생성을 활용하였음을 도출할 수 있었다. 배열과 변화로 사례를 분류한 뒤 배열에서는 배열 형태에 따라 원형 배열과 파형 배열, 변화에서는 애니메이션 유형에 따라 위치 변화, 크기 변화, 색상 변화, 방향 변화로 나누어져 활용하고 있었다. 그 중 위치 변화는 직선형 변화, 파형 변화, 원형 변화로 나누었다. 방향 변화는 극좌표의 특성을 통해 방향을 변경하고 가속도의 설정을 통하여 속도를 변화시키는 것으로 방향 변화로 정의하였다. 각 카테고리에 따라 분류한 결과 소스코드들이 유사하게 사용되고 있는 것을 발견하였다. 이를 종합하여 각 카테고리마다 기본형 코드를 정의하였다.

No		원 배열	파형 배열	회전위치변화	직선형위치변화	파형위치변화	크기변화	방향변화	컬러변화
2	단일			○					
4	단일								○
5	단일								
6	단일	○							
10	단일							○	
11	단일		○					○	
14	단일			○					
15	중복	○						○	
16	중복	○			○				
17	단일	○							
18	단일							○	
19	단일								○
20	단일	○							
21	중복						○		○
22	단일							○	
23	단일			○					

[그림 3-8] 삼각함수 특성이 적용되는 요소에 따라 사례 분류

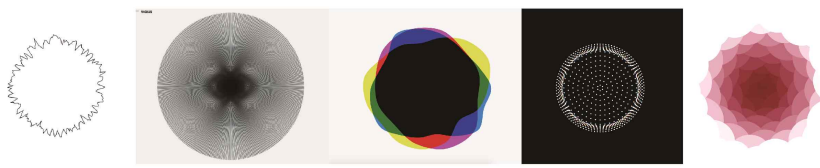


[그림 3-9] 삼각함수 특성이 적용되는 요소에 따른 분류와 특성의 연결

1) 배열

크리에이티브 코딩에서 원형이나 파형으로 배열시킬 때, x, y좌표 값을 구하기 위하여 삼각함수의 극좌표 변환을 사용하였다. 프로세싱과 p5.js는 좌표시스템을 사용한다. 프로세싱에서 원형의 좌표나 파형 모양의 좌표가 필요할 때 극좌표를 사용하여 쉽게 표현할 수 있다. 200개의 작품 중 94개의 작품이 극좌표 변환을 활용하고 있었다. 극좌표를 활용하여 원을 그리면 그림 3-10과 같이 원 좌표 위에 배치된 점들을 이용하여 다양한 그래픽을 표현할 수 있다. 각도와 반지름을 변수로 설정하고 변수들의 조정을 통해 다양한 형태의 배열을 만들었다. 극좌표계는 두 점 사이를 각도나 반지름으로 쉽게 표현하는 경우에 가장 유용하다.

배열은 '원형 배열'과 '파형 배열'로 나누어져 활용되고 있었다. '원형 배열'은 물체를 원형의 형태로 배열하기 위해 원형의 좌표 값을 구하는 과정에서 삼각함수를 활용한 것이며, '파형 배열'은 물체를 파형의 형태로 배열하기 위해 파형의 좌표 값을 구하는 과정에서 삼각함수를 활용하고 있었다.



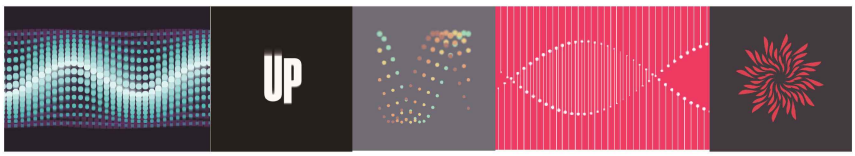
[그림 3-10] 원형 배열 사례 수집 예시

[그림 3-11] 파형 배열 사례 수집 예시

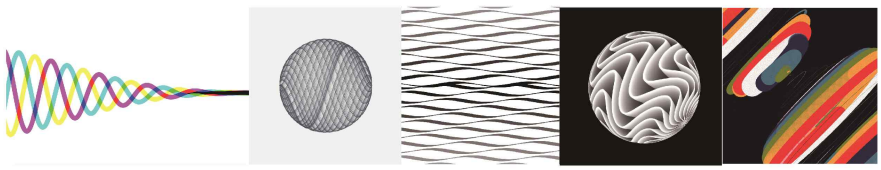
2) 변화

삼각함수의 움직임의 특성인 '주기'와 '가속도'는 2장에서 설명하였듯이 동시에 활용된다. 주기와 가속도는 애니메이션을 변화시키기 위해 활용되는 요소들로 삼각함수를 통하여 애니메이션을 설정한 작품 모두 주기와 가속도를 활용한 움직임을 보여준다. 삼각함수로 움직임을 설정한 작품은 총 150개이다. 작품들은 변수를 설정하여 삼각함수의 진폭과 주기, 위상을 조정하여 움직임을 설정하였다.

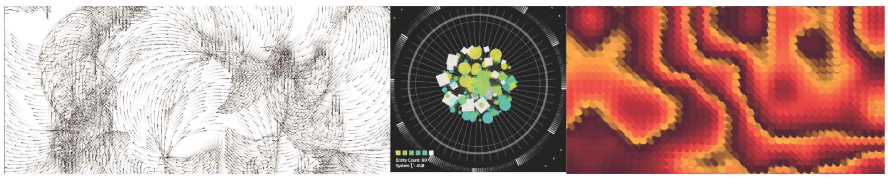
변화는 '위치변화', '크기변화', '색상변화'로 나누어져 활용되고 있었으며 그 중 위치 카테고리는 '직선형위치변화', '파형위치변화', '원형위치변화'로 나누어졌다. '위치변화'는 위치를 변화시켜 움직임을 만드는 것이다. 이 중 '직선형위치변화'는 물체가 직선의 모양으로 주기적으로 움직이면서 양 끝점에서 속도 값을 변화시키는 움직임을 만드는 것이다. '파형위치변화'는 물체가 파형의 모양으로 한 축의 좌표 값이 증가하면서 다른 축의 좌표 값에 속도의 변화를 가지고 주기적으로 움직이는 것이다. '원형위치변화'는 물체가 원형의 모양으로 속도 변화와 동시에 주기적으로 움직임을 의미한다. '크기변화'는 도형의 길이나 원의 지름을 변화시켜 크기가 변화하는 움직임을 만드는 것이며 '색상변화'는 색상의 그라데이션을 변화시켜 움직임을 만드는 것으로 활용되고 있었다.



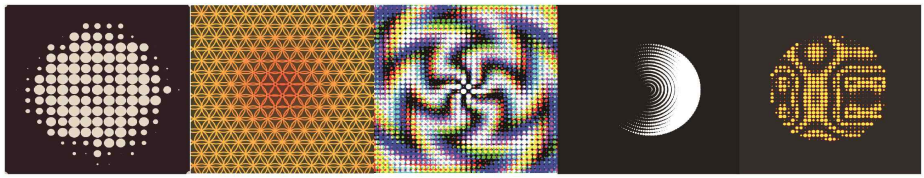
[그림 3-12] 직선형 위치 변화 사례 수집 예시



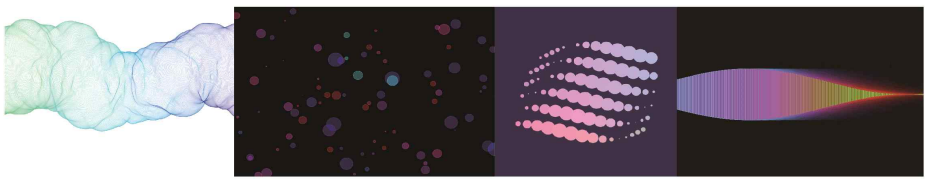
[그림 3-13] 파형 위치 변화 사례 수집 예시



[그림 3-14] 원형 위치 변화 사례 수집 예시



[그림 3-15] 크기 변화 사례 수집 예시



[그림 3-16] 색상 변화 사례 수집 예시

방향 변화는 극좌표의 특성으로 가리키는 방향을 설정하고 가속도 설정의 특성으로 물체 움직임의 빠르기를 설정하는 것이다. 방향은 벡터와 연관된다. 벡터는 크기와 방향을 가지고 있다. 벡터에서는 속도(Velocity)와 빠르기(Speed)의 개념을 다르게 보는데 속도는 특정 방향의 빠르기이고 빠르기는 프레임에서 다음 프레임으로 위치가 변화할 때 어느 픽셀만큼 변화하는지에 대한 개념이다. 벡터에서 각도를 이용하여 움직임을 만들기 위해 삼각함수를 활용한다. 삼각함수를 벡터의 방향 변화에 활용한 작품은 총 29개이다. 방향 변화는 파티클이나 벡터 필드와 결합하여 활용되는 비중이 크다.

[그림 3-17] 방향 변화 사례 수집 예시

[표 3-4] 삼각함수 활용 패턴 사용 빈도 (중복가능)

배열		변화					
원형 배열	파형 배열	직선형위치변화	파형위치변화	원형위치변화	크기변화	색상변화	방향변화
94	8	78	15	25	21	11	29

제 4장

크리에이티브 코딩
웹 어플리케이션 개발

제 4장
크리에이티브 코딩 웹 어플리케이션 개발

제 1절 | CCWA의 특징

크리에이티브 코딩 웹 어플리케이션(CCWA)은 크리에이티브 코딩에서 삼각함수를 사용하고자 할 때 디자이너에게 필요한 특성을 빠르고 효과적으로 활용할 수 있도록 돕는 웹 어플리케이션이다.

[그림 4-1] CCWA의 특징

1) 삼각함수 기반 코딩 알고리즘 어플리케이션

삼각함수의 특성인 '극좌표의 변환', '가속도의 설정', '주기 생성'을 활용하여 크리에이티브 코딩에 쉽게 적용할 수 있도록 적용 유형을 분류하여 선택할 수 있게 한다. 삼각함수에 대한 기초지식의 이해가 부족해도 삼각함수를 적용하고 싶은 영역에 원하는 특성을 적용할 수 있다. 또한 디자이너가 삼각함수를 활용하기 위한 학습을 쉽게 도와준다.

2) 오픈소스(Processing) 기반 코딩지원 플랫폼

오픈소스 기반 언어로 작성되어 크리에이티브 코딩 작품이 삼각함수를 활용한 방법에 대해 이해하고 이를 활용할 수 있다. CCWA는 오픈소스 언어인 Processing과 p5.js로 코드를 제공한다. Processing과 p5.js에서는 만든 프로그램 코드를 공유하도록 권장하는데 이는 다른 사람들이 제작한 코드에 접근하여 학습을 진행할 수 있도록 한다. CCWA를 통하여 다른 작품들의 삼각함수 적용 유형을 파악하고 응용하는데 도움을 줄 수 있다.

3) 직관적인 GUI 인터페이스

직관적인 GUI 인터페이스로 구성하여 웹 어플리케이션을 쉽고 빠르게 활용할 수 있도록 하였다. 디자이너는 컬러, 오브젝트의 모양 및 위치, 애니메이션의 속도 등 디자인적 요소가 구현되는 것에 관심을 가지고 있기[20] 때문에 메뉴와 코드적용 그래픽 시뮬레이터, 그래프를 변화시키는 파라미터, 적용 코드, 적용 사례 순으로 흐름이 연결되도록 구성하였다.

기존의 학습법은 코드 텍스트를 직접 수정해가며 코드에 따라 애니메이션이 변화하는 정도를 관찰하고 이를 자신의 작품에 응용하는 것이었다. CCWA는 애니메이션과 변화 파라미터를 활용하여 마우스로 파라미터의 조작만을 통해 삼각함수 코드의 변화에 따른 애니메이션의 변화를 직관적으로 관찰할 수 있다. 또한 코드적용 그래픽 시뮬레이터에서 보조선을 제공하여 애니메이션이 삼각함수 그래프와 연결되고 있음을 보여준다.

이는 디자이너가 파라미터의 조정만으로 삼각함수의 변화 값을 조정할 수 있으며 이를 응용할 수 있도록 돕는다. 연결되는 코드와 파라미터 이름의 컬러지정을 통하여 파라미터의 조정에 따른 코드의 변화가 바로 느껴질 수 있도록 하였다.

20) 권진아, 디자이너를 위한 이러닝 기반의 프로그래밍 학습 서비스 디자인, 이화여자대학교 석사학위논문, 2019, p.7

제 2절 | CCWA의 기본 화면 구성

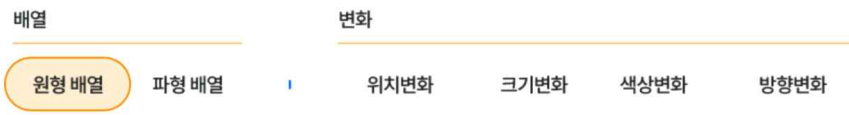
CCWA의 인터페이스 구성은 [그림4-1]과 같이 구성하였다. 1)메뉴와 2)코드적용 그래픽 시뮬레이터 3)파라미터 슬라이더 4)적용 코드 5)적용 사례로 구성되어 있다.

[그림 4-2] 크리에이티브 코딩 웹 어플리케이션 레이아웃

1) 메뉴 구성

메뉴는 삼각함수를 적용하고 싶은 영역에 활용할 수 있도록 적용 유형에 따라

나누어 배열과 변화로 나누어지며 유사한 배열, 변화 유형으로 그룹핑 한 것을 기반으로 원형 배열, 파형 배열, 위치변화, 크기변화, 색상변화, 방향변화로 구성하였다. 하위 메뉴가 있는 메뉴인 위치변화는 해당 메뉴를 클릭하게 되면 직선형변화, 파형변화, 원형변화 버튼이 나타난다. 메뉴 구성에 대한 자세한 설명은 3절에서 이어진다.

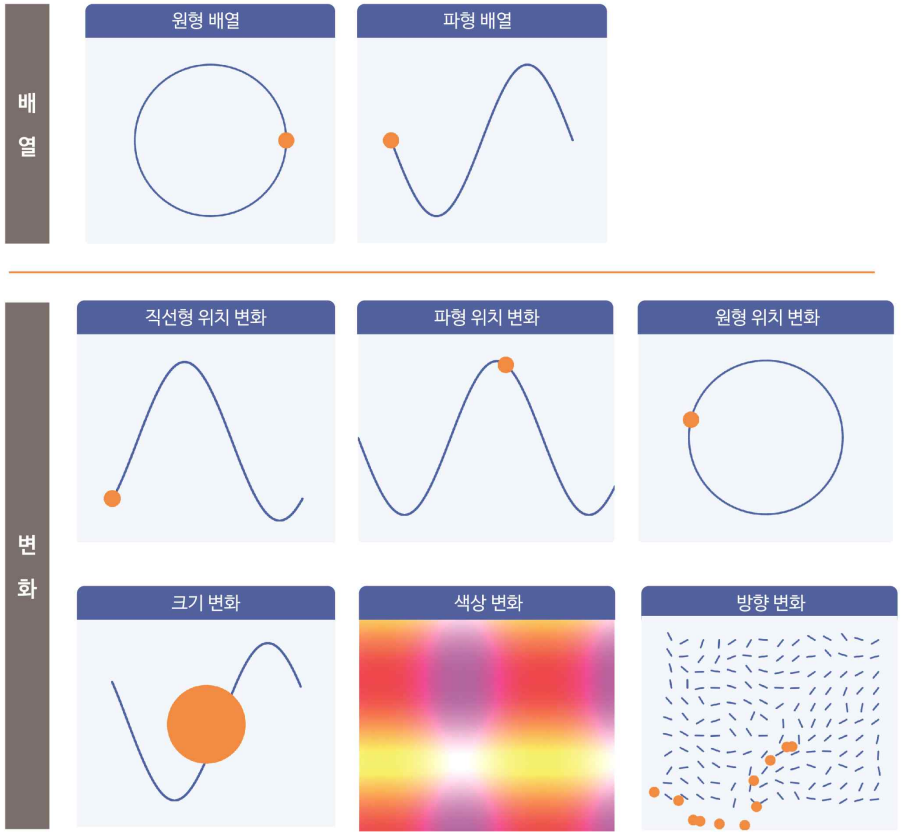


[그림 4-3] CCWA의 메뉴 구성

[그림 4-4] 위치변화 하위 메뉴 구성

2) 코드적용 그래픽 시뮬레이터

삼각함수의 가속도 설정과 주기 생성의 특성이 드러나려면 코드적용 그래픽 시뮬레이터가 필수적이라고 판단하여 구성하였다. 코드적용 그래픽 시뮬레이터는 보조선과 적용 애니메이션을 함께 표현하여 움직임의 원리를 이해하기 쉽도록 구성하였다. 색상변화 카테고리는 보조선이 필요하지 않아 제외하였다. 코드적용 그래픽 시뮬레이터는 파라미터 슬라이더의 조정을 통해 그래프의 진폭의 범위, 주기의 변화 등 다양한 변화 값을 가질 수 있으며 시뮬레이션 애니메이션에 바로 적용된다. 각 메뉴에 해당하는 코드적용 그래픽 시뮬레이터의 구성은 다음과 같다.



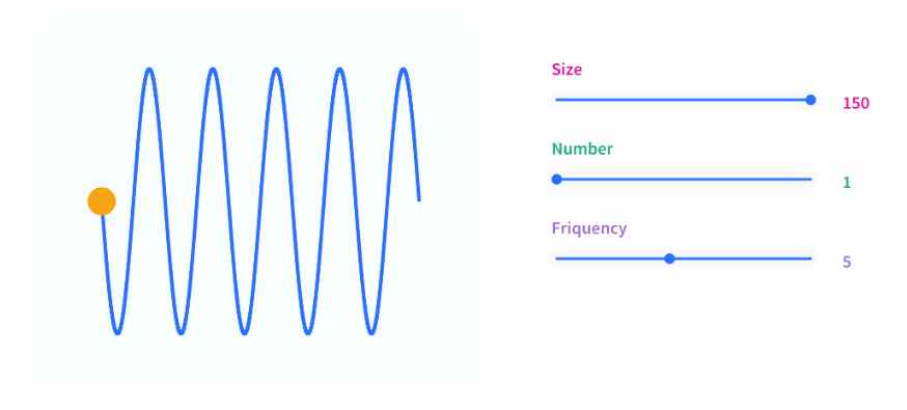
[그림 4-5] 각 메뉴의 코드적용 그래픽 시뮬레이터

3) 파라미터 슬라이더

코드적용 그래픽 시뮬레이터와 함께 이를 변화시킬 수 있는 파라미터를 구성하여 변화 값을 컨트롤 할 수 있도록 하였다. 메뉴마다 변화시킬 수 있는 영역이 달라 파라미터 슬라이더의 구성을 다르게 설정하였다. [표 4-1]은 메뉴 카테고리별 변화 파라미터 슬라이더 구성이다. 파라미터 슬라이더의 컨트롤을 통해 코드적용 그래픽 시뮬레이터의 변화, 적용 코드의 변수값 변화를 연결하여 보여주었다.

[표 4-1] 메뉴 카테고리별 변화 파라미터 슬라이더 구성

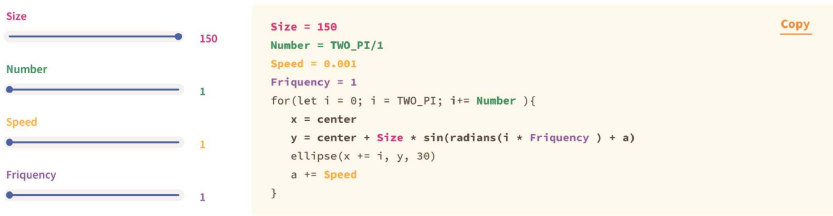
메뉴 카테고리	Size	Number	Speed	Friquency
원형 배열	○	○		
파형 배열	○	○		○
직선형 위치 변화	○	○	○	○
파형 위치 변화	○	○	○	○
원형 위치 변화	○	○	○	
크기 변화	○		○	
색상 변화				○
방향 변화	○		○	



[그림 4-6] 파라미터 슬라이더의 변화 값에 따른 시뮬레이터

4) 적용 코드

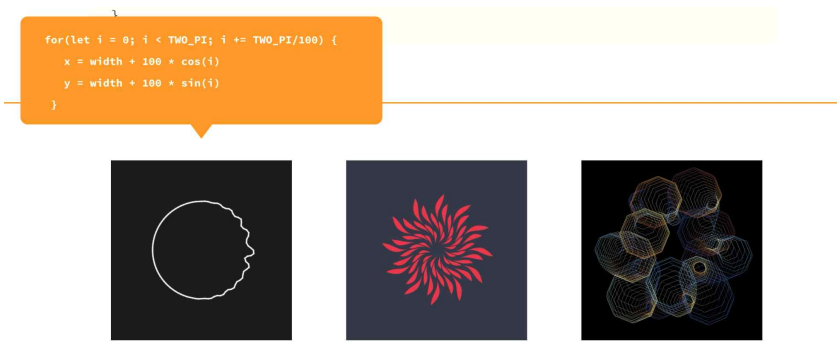
크리에이티브 코딩에 삼각함수를 쉽게 적용할 수 있도록 코드의 기본형을 제공한다. 이 코드는 파라미터 슬라이더 조정에 따라 변화한다. 이는 단순히 기본형만을 제시하는 것보다 응용 학습을 돕는다. 코드 우측의 Copy 버튼을 누르면 기본형 코드를 다운받을 수 있는 페이지로 연결된다.



[그림 4-7] 파라미터 슬라이더와 적용 코드의 변수 값 연결

5) 적용 사례

삼각함수의 기본형에서 확장할 수 있도록 돕기 위하여 해당 메뉴를 활용한 사례를 추가하였다. 사례들의 이미지를 클릭하고 있으면 사례들이 삼각함수를 어떻게 사용하였는지 볼 수 있다. 애니메이션만으로는 삼각함수의 다양한 활용 방법이 드러나지 않아 코드를 확장해 나갈 수 있음을 보여주기 위하여 기존 작품에서 활용하고 있는 코드 일부를 추가하였다.



[그림 4-8] 적용 사례

배열

원형 배열

```
Size = 150
number = TWO_PI/1
for(let i = 0; i = TWO_PI; i+= Number ){
   x = center + Size * cos(radians(i))
   y = center + Size * sin(radians(i))
   ellipse(x, y, 30)

}
```

파형 배열

```
Size = 150
Number = TWO_PI/1
Speed = 0.001
Friquency = 1
for(let i = 0; i = TWO_PI; i+= Number ){
   x = center
   y = center + Size * sin(radians(i * Friquency ))
   ellipse(x += i, y, 30)
}
```

변화

직선형 위치 변화

```
Size = 150
Number = TWO_PI/1
Speed = 0.001
Friquency = 1
for(let i = 0; i = TWO_PI; i+= Number ){
   x = center
   y = center + Size * sin(radians(i * Friquency ) + a)
   ellipse(x += i, y, 30)
   a += Speed
}
```

파형 위치 변화

```
Size = 150
Number = TWO_PI/1
Speed = 1.000
Friquency = 1
for(let i = 0; i = TWO_PI; i+= Number ){
   x = center
   y = center + Size * sin(radians( position * Friquency + i ))
   ellipse(x, y, 30)
   x += i + position
   position += Speed
}
```

원형 위치 변화

```
Size = 150
Number = TWO_PI/1
Speed = 0.001
for(let i = 0; i = TWO_PI; i+= Number ){
   x = center + Size * cos(radians(i) + a)
   y = center + Size * sin(radians(i) + a)
   ellipse(x, y, 30)
   a += speed
}
```

크기 변화

```
Size = 150
Speed = 0.001
Friquency = 1
r = Size * sin(a * Friquency))
   ellipse(x, y, r)
   a += Speed
}
```

색상 변화

```
friquency = 50
for(let i = 0; i = 400; i += 3) {
 for(let j = 0; i = 300; i += 3) {
  g = 255 - ((sin(i/ Friquency )+1)*100)
  b = 255 - ((sin(i/ Friquency )+1)*100)
  fill(255, g, b)
  rect(i, j, 10, 10
 }
}
```

방향 변화

```
angle = frameCound(30)
force = 0.05
vx, vy = 0
ax = sin(angle) * force
ay = cos(angle) * force
vx += ax
vy += ay
x += vx
y += vy
ellipse(x, y, 20, 20)
```

[그림 4-9] 각 메뉴 카테고리의 기본형 코드

제 3절 | 메뉴의 구성

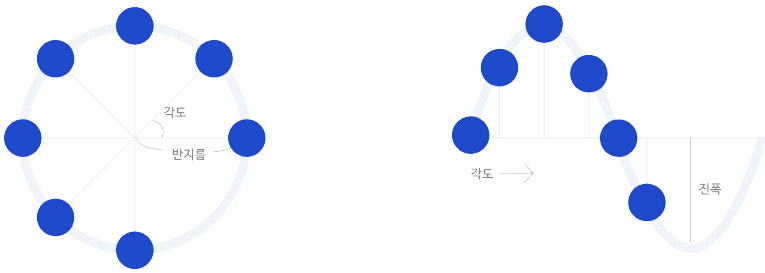
CCWA 메뉴는 디자이너가 쉽게 활용 할 수 있도록 하기 위하여 앞의 3장에서 작품 사례 연구를 기반으로 하여 배열의 '원형배열', '파형배열'과 변화의 '위치변화', '크기변화', '색상변화', '방향변화'로 구성하였다. 디자이너가 삼각함수 알고리즘을 사용하려는 목적은 삼각함수 알고리즘을 통해 자신이 구현하고자 하는 결과물에 적용하는 방법을 알기 위한 것이므로 삼각함수 관련 용어의 사용을 줄이고 디자이너가 구현하고 싶은 유형을 선택하여 메뉴에 바로 접근할 수 있도록 구성하였다.

1) 원형배열

원형배열은 극좌표를 이용하여 원 좌표를 만들고 좌표 위의 점들을 이용하여 원을 형성하거나 좌표 위에 물체를 배치하여 결과물을 만드는 방법을 뜻한다. 원형배열에서 삼각함수를 통해 조정할 수 있는 영역은 각도 값을 입력하여 물체를 원 좌표 위에 배치하는 것과 반지름 값의 입력을 통해서 원 좌표의 원 크기를 설정하는 것이다. 각도 값을 for문의 반복에 적용하여 배치하는 물체의 개수를 조정하고 진폭 값을 변경하여 원 좌표의 크기를 조정할 수 있도록 구성하였다.

2) 파형배열

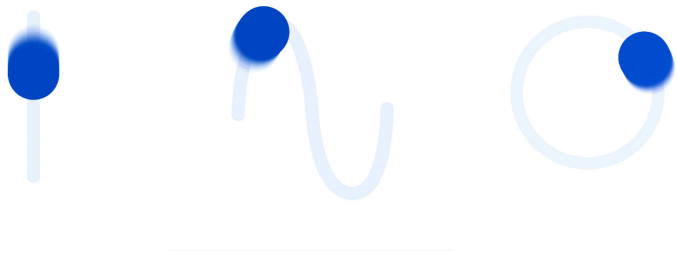
파형배열은 극좌표를 이용하여 파형의 좌표를 만들고 좌표 위에 점들을 이용하여 파형을 형성하거나 파형 좌표 위에 물체를 배치하여 결과물을 만드는 방법이다. 파형배열에서 삼각함수를 통해 조정할 수 있는 영역은 각도 값을 입력하여 물체를 파형 좌표 위에 배치하는 것과 진폭 값의 입력을 통해서 파형의 범위를 설정하는 것, 주기에 변화를 주어 파형을 변형시키는 것이 있다. 각도 값을 for문의 반복에 적용하여 배치하는 물체의 개수를 조정하고 진폭 값을 변경하여 파형 좌표의 범위, 주기를 조정하여 파형의 변형을 변경할 수 있도록 구성하였다.



[그림 4-10] 원형 배열과 파형 배열의 원리

3) 위치변화

위치변화는 변화 유형에 따라 '직선형 위치변화', '파형 위치변화', '원형 위치변화'로 하위 메뉴를 구성하였다. '직선형 위치변화'는 상하나 좌우로 한 축에만 변화 값을 주어 물체의 위치를 변화시키는 것이다. 위치를 변화시킬 때 삼각함수를 x축이나 y축 한 축에만 적용하여 범위를 설정하고 그 범위 안에서 가속도를 더하고 빼주어 범위의 양 끝으로 갈수록 속도가 빨라지고 중간으로 갈수록 속도가 느려지도록 조정한 것이다. '파형 위치변화'는 직선형 위치변화와 마찬가지로 x축이나 y축에 삼각함수 각도에 속도를 더하여 값의 변화를 조정하는데 다른 한 축에 증가하는 값을 더하여 물체가 움직이는 모양이 파형이 된다. '원형 위치변화'는 x축과 y축 모두에 삼각함수 각도에 속도를 더하여 원형의 애니메이션을 만드는 것이다. 물체의 각도 값을 for문의 반복에 적용하여 배치하는 물체의 개수를 조정하고 진폭 값을 변경하여 범위, 주기를 조정하여 움직이는 파형의 변형과 함께 각도에 더하는 속도를 변경할 수 있도록 구성하였다.



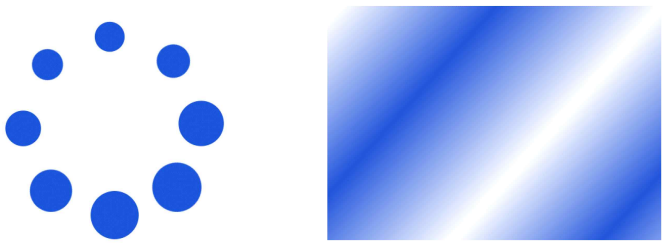
[그림 4-11] 위치 변화의 기본 애니메이션

4) 크기변화

크기변화는 삼각함수의 변화 값을 원의 지름이나 도형의 길이에 적용하여 크기를 변화시키는 것이다. 삼각함수의 진폭을 활용하여 도형의 크기의 범위를 설정하고 각도 값에 변화하는 속도 값을 더해주어 따라 크기가 변화하는 속도를 조정한다. 주기의 조정을 활용하여 펄스효과를 만들어 낼 수 있다. 또한 도형을 배열하고 각 도형에 오프셋 값이 다른 삼각함수 식을 적용하면 시간차 애니메이션을 만들어 낼 수 있다.

5) 색상변화

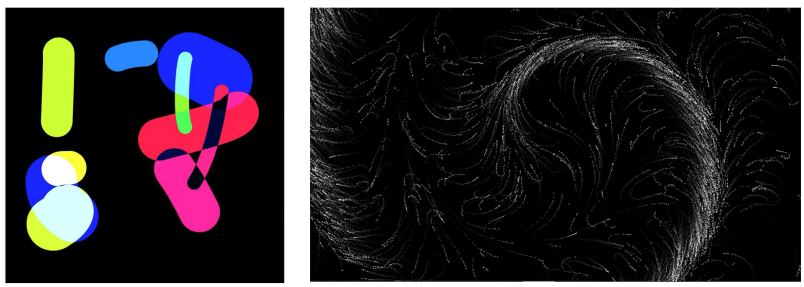
색상변화는 삼각함수를 통하여 색상의 그라데이션을 설정하는 것이다. 코딩에서는 그라데이션을 위하여 다양한 계산법을 활용하고 있다. 삼각함수도 그라데이션을 설정할 수 있다. 이는 보간법과 관련이 있는데 그라데이션은 한 점의 색과 다른 점의 색이 있을 때 그 사이의 중간 값들의 모임이라고 할 수 있다. 삼각함수를 여기에 활용하면 각도 값의 변화에 따라 색상 값을 설정하여 그라데이션을 만들 수 있다. 삼각함수의 주기 값의 변화로 그라데이션의 주기도 변화한다. 이 변화를 관찰할 수 있도록 파라미터로 주기 값을 변화시킬 수 있도록 구성하였다.



[그림 4-12] 오프셋 값을 더한 크기 변화와 색상 변화 그라데이션

6) 방향변화

방향변화는 삼각함수의 극좌표 특성으로 물체가 향하는 방향을 설정하고 가속도의 특성으로 물체 움직임의 빠르기를 설정하는 것이다. 극좌표의 개념을 가져와 물체의 x축 위치 좌표에 sin(각도)*빠르기를 더하고 y축 위치 좌표에 cos(각도)*빠르기를 더하면 원하는 각도가 가리키는 방향으로 빠르기만큼 물체를 움직일 수 있다. 3장의 사례 분석 결과 방향 변화는 랜덤을 활용하여 무작위성을 나타내거나 벡터 필드에 따라 움직이도록 하는 경우가 많았다. 이는 삼각함수 값의 조정과는 관련이 없어 방향변화에는 변화 파라미터를 구성하지 않았다.



[그림 4-13] 랜덤과 벡터 필드를 활용한 방향 변화 사례

[표 4-2] 메뉴 카테고리별 삼각함수 코드의 역할

메뉴 카테고리	기본 코드

원형배열	for(let I = 0; I 〈 TWO_PI; I += 원형에 배열하는 물체의 간격(각도)) {
	x = 위치 중심좌표값 + 반지름의 크기(원의 크기) * sin(radians(i))
	y = 위치 중심좌표값 + 반지름의 크기(원의 크기) * cos(radians(i))
	배열하고자 하는 물체(x, y, 30, 30)
	}

파형배열	for(let I = 0; I 〈 TWO_PI; I += 파형에 배열하는 물체의 간격(각도)) {
	x = 위치 좌표값
	y = 위치 좌표값 + 진폭의 크기 * sin(radians(i * 주기 변경값))
	배열하고자 하는 물체(x, y, 30, 30)
	}

직선형
위치변화
	for(let I = 0; I 〈 TWO_PI; I += 파형에 배열하는 물체의 간격(각도)) {
	x = 위치 좌표값
	y = 위치 좌표값 + 진폭의 크기 * sin(radians(i * 주기 변경값) + a)
	배열하고자 하는 물체(x, y, 30, 30)
	a += 속도값
	}

파형
위치변화
	for(let I = 0; I 〈 TWO_PI; I += 파형에 배열하는 물체의 간격(각도)) {
	x = 위치 좌표값
	y = 위치 좌표값 + 진폭의 크기 * sin(radians(a + i * 주기 변경값))
	배열하고자 하는 물체(x, y, 30, 30)
	x += I + a
	a += 속도값
	}

원형
위치변화
	for(let I = 0; I 〈 TWO_PI; I += 파형에 배열하는 물체의 간격(각도)) {
	x = 위치 중심좌표값 + 진폭의 크기 * sin(radians(i) + a)
	y = 위치 중심좌표값 + 진폭의 크기 * cos(radians(i) + a)
	배열하고자 하는 물체(x, y, 30, 30)
	a += 속도값
	}

크기변화	x = 위치 좌표값 y = 위치 좌표값 r = 진폭의 크기 * sin(각도값 * 주기 변경 값) 크기변화 물체(x, y, r, r) a += 속도값

색상변화	for(let x = 0; x < 400; x += 3) {
	for(let y = 0; y < 400; y += 3) {
	r = 255
	g = 색상기본값 - ((sin(x/주기 변화 값)+1)*100)
	b = 색상기본값 - ((sin(y/주기 변화 값)+1)*100)
	fill(r, g, b)
	rect(x, y, 10, 10)
	}

방향변화	ax = sin(증가하는 각도값) * 힘(임의의 값을 설정한다) ay = cos(증가하는 각도값) * 힘 vx += ax vy += ay x += vx y += vy 움직이는 물체(x, y, 30, 30)

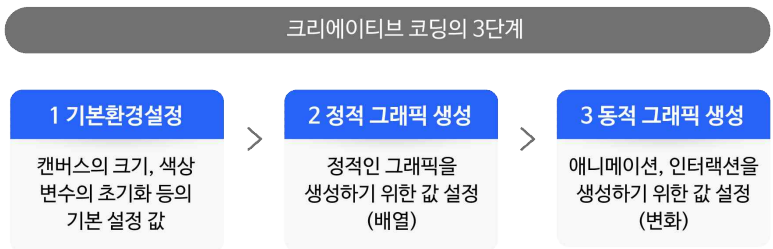
제 4절 | CCWA 적용 작품구현 사례

CCWA의 활용 시나리오 제안을 위하여 3장에서 수집했던 사례를 통하여 CCWA 활용 과정을 설명하였다. 크리에이티브 코딩의 단계를 총 3단계로 설정하고 CCWA를 어떤 단계에서 활용할 수 있는지와 코드에서 적용 가능한 방법을 설명하였다.

1단계는 기본환경설정 단계로 코딩에 캔버스의 사이즈나 색상, 코딩에 필요한 변수 초기화 등을 설정하는 단계이다. 2단계는 정적인 그래픽을 생성하는 단계

로 코딩에서 도형의 배치를 설정하는 단계이다. 3단계는 정적인 그래픽에 애니메이션이나 인터랙션 등 변화할 수 있는 값을 설정하는 단계이다.

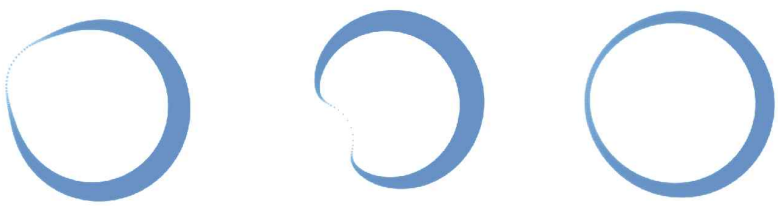
CCWA의 메뉴에서 배열은 2단계, 변화는 3단계에서 적용할 수 있다.



[그림 4-14] 크리에이티브 코딩의 3단계

다음은 3장에서 수집했던 사례 중 일부를 가져와 코드에서 CCWA를 적용하는 시나리오에 대하여 설명하였다.

1) 사례 1 : Shape bend[21)]

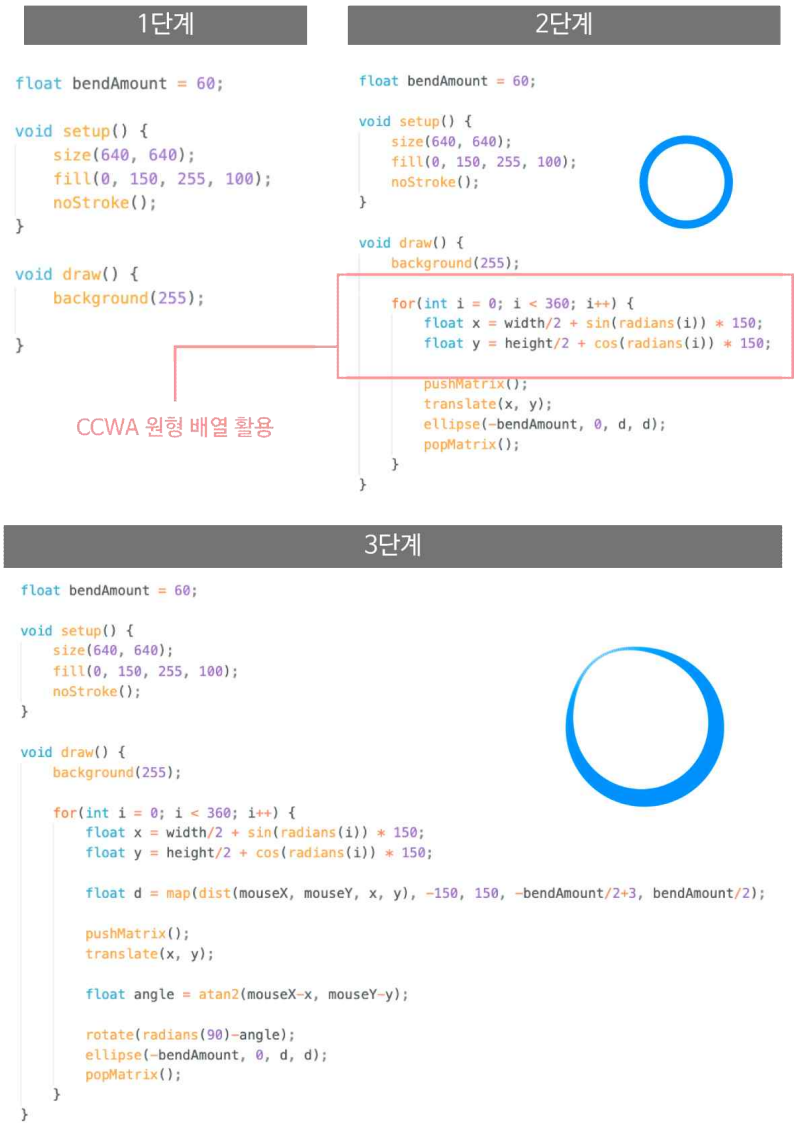


[그림 4-15] 사례1 이미지

그림 3-2는 마우스의 움직임에 따라 원형으로 배열된 원들의 위치가 변화하는 인터랙션 반응을 보여주는 작품이다. 이 작품은 2단계에서 원형배열을 설정하기

21) https://openprocessing.org/sketch/166967

위하여 CCWA의 원형배열 코드 기본형에서 각도 값의 설정을 변경하여 사용하였다. 3단계에서는 인터랙션을 위한 변수 설정과 인터랙션에 반응하는 변화를 설정하였다.




```
float bendAmount = 60;

void setup() {
    size(640, 640);
    fill(0, 150, 255, 100);
    noStroke();
}

void draw() {
    background(255);

}
```

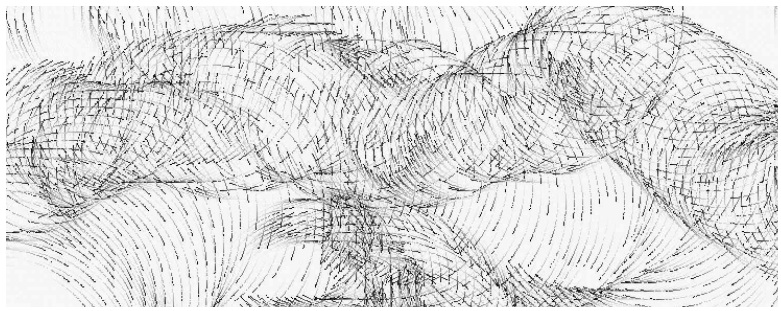
```
float bendAmount = 60;

void setup() {
    size(640, 640);
    fill(0, 150, 255, 100);
    noStroke();
}

void draw() {
    background(255);

    for(int i = 0; i < 360; i++) {
        float x = width/2 + sin(radians(i)) * 150;
        float y = height/2 + cos(radians(i)) * 150;

        pushMatrix();
        translate(x, y);
        ellipse(-bendAmount, 0, d, d);
        popMatrix();
    }
}
```

```
float bendAmount = 60;

void setup() {
    size(640, 640);
    fill(0, 150, 255, 100);
    noStroke();
}

void draw() {
    background(255);

    for(int i = 0; i < 360; i++) {
        float x = width/2 + sin(radians(i)) * 150;
        float y = height/2 + cos(radians(i)) * 150;

        float d = map(dist(mouseX, mouseY, x, y), -150, 150, -bendAmount/2+3, bendAmount/2);

        pushMatrix();
        translate(x, y);

        float angle = atan2(mouseX-x, mouseY-y);

        rotate(radians(90)-angle);
        ellipse(-bendAmount, 0, d, d);
        popMatrix();
    }
}
```

[그림 4-16] 사례1 코딩 시나리오

먼저 1단계에서 캔버스의 사이즈, 색 등을 설정하고 변수를 설정한다. 사례 1의 경우 2단계에서 CCWA의 원형 배열을 활용하였다. 캔버스 중앙에 크기 150의 원형 배열이 올 수 있도록 좌표를 설정하였다. 사례 1은 이후 3단계에서 마우스 인터랙션을 이용하기 위하여 물체의 좌표를 직접 설정하지 않고 translate() 함수를 통해 원형으로 배열하였다. 3단계에서는 마우스 인터랙션의 설정을 위하여 마우스와 좌표의 거리 값에 따른 이동 범위를 map() 함수를 통해 재설정하였다. 마우스가 원형에 가까워지면 원의 지름이 줄어드는 인터랙션을 생성한다.

2) 사례 2 : Fake Particles[22)]



[그림 4-17] 사례2 이미지

그림 5-4는 원형위치변화를 활용한 작품이다. 1단계에서 캔버스의 사이즈, 색 등을 설정하고 변수를 초기화 한다. 사례 2의 경우 3단계에서 CCWA의 원형 위치 변화를 활용하였다. 2단계에서 x, y의 기본 좌표 값을 지정하고 3단계에서 각도에 노이즈값을 곱하여 위치변화에 시간차를 설정하였다. b 변수에 속도 값을 더해주어 애니메이션을 설정하였다.

22) https://openprocessing.org/sketch/108501

1단계

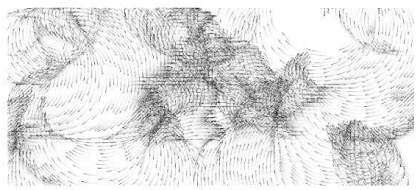
```
float b = 0;

void setup() {
    size(900, 400);
    background(255);
    fill(255, 15);
    noSmooth();
}

void draw() {
    noStroke();
    rect(0, 0, width, height)
}
```

2단계

```
float b = 0;

void setup() {
    size(900, 400);
    background(255);
    fill(255, 15);
    noSmooth();
}

void draw() {
    noStroke();
    rect(0, 0, width, height)

    stroke(0);
    for(int x = 0; x < width; x += 8) {
        for(int y = 0; y < height; y += 8) {
        point( x, y);
      }
    }
```

3단계

```
float b = 0;

void setup() {
    size(900, 400);
    background(255);
    fill(255, 15);
    noSmooth();
}

void draw() {
    noStroke();
    rect(0, 0, width, height)

    stroke(0);
    for(int x = 0; x < width; x += 8) {
        for(int y = 0; y < height; y += 8) {
        float a = 4 * noise(x / 300.0, y / 300.0) * TWO_PI + b;
        point( x + 100 * cos(a), y + 100 * sina(a));
      }
    }

    b += 0.01;
}
```



CCWA 원형 위치 변화 활용

[그림 4-18] 사례2 코딩 시나리오

제 5장

결론

제 5장
결 론

제 1절 | 연구의 발견점

본 연구에서는 크리에이티브 코딩에 삼각함수 알고리즘 활용 방법을 개발하여 디자이너가 삼각함수를 쉽게 활용하는 것을 돕고자 한다. 문헌연구를 통해 삼각함수가 크리에이티브 프로그래밍에 활용되는 특성을 조사하였다. 삼각함수의 극좌표 변환, 가속도 설정, 주기 생성의 특성을 크리에이티브 코딩에서 활용하고 있었다. 삼각함수의 특성은 움직임과 연관이 있다. 디자이너들의 접근성이 가장 높은 웹 프로그래밍 애니메이션 알고리즘을 적용하기 쉽게 안내하는 사이트 5개를 조사하여 사이트의 구성, 메뉴 분류 기준, 변화 파라미터의 설정 기준을 파악하였다. 또한 기존 크리에이티브 코딩 작품 사례를 수집하여 유사한 움직임 유형끼리 분류하여 디자이너가 필요에 의해 활용할 수 있도록 메뉴를 구성하였다. 디자이너들은 삼각함수의 특성인 극좌표 변환, 가속도 설정, 주기 생성을 배열, 위치, 크기, 색상, 방향 변화에 활용하고 있음을 도출할 수 있었다. 각 메뉴 카테고리별로 가장 기본형의 애니메이션을 만들고 이를 파라미터를 통해 조정할 수 있도록 하였다. 또한 파라미터를 통해 애니메이션이 조정될 때 적용 코드도 함께 변하여 기본형에서 쉽게 응용할 수 있도록 하였다. 해당 코드를 사용한 작품의 사례를 추가하여 기본형에서 간단한 응용에 이어 아이디어를 확장해 나갈 수 있도록 도왔다.

이를 통한 연구의 시사점은 다음과 같다.

1. 크리에이티브 코딩을 위한 직관적 GUI 환경의 웹 어플리케이션 개발이 필요하며 이는 디자이너가 쉽게 이해할 수 있도록 돕는다.

2. 디자이너를 위한 수학적 알고리즘 활용 기반 웹 어플리케이션은 함수의 특성 분석과 디자이너에게 익숙한 메뉴 구성 및 파라미터 설정 기준을 정의해야 한다.
3. 디자이너를 위한 수학적 알고리즘 웹 어플리케이션의 개발 연구가 확대되어야 한다.

제 2절 | 향후 연구 과제

본 연구에서는 기본형 코드를 제공하고 가벼운 응용 단계까지 안내하는 웹 어플리케이션을 개발하였다. 그러나 적용과정에서 아직 다루지 못한 부분도 많다. 삼각함수 기본형의 코드에 노이즈 값을 곱하거나 여러 변수를 연산하여 애니메이션을 제어하기 위해서는 추가적인 학습이 필요하다. 추가 연구를 통하여 다양한 응용 방법의 안내로 이어진다면 디자이너의 크리에이티브 코딩 학습에 크게 기여할 수 있을 것이다. 또한 크리에이티브 코딩에서 활용하고 있는 다른 수학적 알고리즘도 GUI 기반의 웹 어플리케이션으로 활용방법을 제안하는 연구가 필요하다.

참고문헌

도 서

- Andrew Richardson, Date-driven Graphic Design: Creative Coding for Visual Communicatio, 2017
- 다니엘 쉬프만, Nature of Code, 한빛미디어, 2015
- 케이시 리아스 외 1, 프로세싱 교과서, 유엑스리뷰, 2018

학위논문

- 함혜연, 모션그래픽 움직임의 시지각 효과 연구, 중앙대학교 학위논문, 2004
- 권진아, 디자이너를 위한 이러닝 기반의 프로그래밍 학습 서비스 디자인, 이화여자대학교 학위논문, 2019

학술논문

- 김혜란, 예술 활동을 통한 창의적 코딩에 관한 연구 – 교육 프로그램 사례를 기반으로, 한국영상학회논문집, 19(4), 2021

웹사이트

- https://openprocessing.org/
- https://processing.org/reference
- https://processing.org/reference/sin_.html
- https://suhak.tistory.com/300
- https://www.youtube.com/watch?v=KRVhtMxQWRs
- https://www.fizzics.org/sine-and-cosine-graphs/
- https://testbook.com/learn/physcis-sound-waves/
- https://dynamicsjs.com/
- https://animejs.com/
- https://greensock.com/docs/v3/Eases
- https://bouncejs.com/
- https://animista.net/

- 국문초록 -

삼각함수를 활용한 크리에이티브 코딩 웹 어플리케이션 개발

Developing a Creative Coding Web Application Using a Trigonometry Function

디자이너들은 GUI 기반의 소프트웨어를 통해 컴퓨터를 활용하고 있다. 하지만 소프트웨어를 넘어 컴퓨터를 활용하고자 하는 움직임은 꾸준히 이어져 왔으며 코딩교육의 확산으로 크리에이티브 코딩의 수요도 증가하고 있다. 코딩에서 수학적 알고리즘은 물리학에 기초하여 자연의 움직임을 표현하기 위하여 사용되는데 이 중 삼각함수 알고리즘은 가장 많이 사용되는 알고리즘 중 하나이다. 그러나 디자이너에게 지나치게 수학적 부분에만 초점이 맞춰져 있는 자료들은 이해하기 어렵다. 디자이너들이 수학적 알고리즘의 특징을 이해하고 쉽게 접근할 수 있는 방법이 필요하다.

이러한 배경을 바탕으로 본 연구에서는 디자이너가 이해하기 쉽도록 삼각함수를 활용한 크리에이티브 코딩 웹 어플리케이션 개발을 목적으로 진행하였다. 이를 개발하기 위하여 디자이너에게 가장 익숙한 웹 애니메이션 코딩 알고리즘웹 어플리케이션 사례를 분석하여 개발할 크리에이티브 코딩 웹 어플리케이션(CCWA)의 화면 구성과 메뉴 구성의 방향성을 설정하였다. 이후 디자이너가 삼각함수를 활용하고 있는 유형을 도출하기 위하여 크리에이티브 코딩 작품 200개를 분석하였다. 이를 통해 디자이너가 삼각함수가 활용되는 유형을 인지하고 원하는 유형을 선택할 수 있도록 메뉴를 구성하였다.

결과적으로 본 논문에서 개발한 크리에이티브 코딩 웹 어플리케이션은 삼각함수를 기반으로 하며, 오픈소스 코딩지원, 직관적인 GUI 인터페이스를 특징으로 하는 웹 어플리케이션을 개발하였다. 디자이너는 직관적인 GUI 인터페이스와 디자이너의 필요 위주로 구성한 메뉴를 통하여 삼각함수를 쉽게 코딩에 적용하고 응용할 수 있다.