



프론트엔드 미니프로젝트 v1.0

디지털 영상 처리 w/ Javascript

by 심윤경

2021년 상공회의소 서울기술교육원
자바기반 빅데이터 시각화 시스템 과정

프론트엔드 프로그래밍의 세계로 풍덩...

HTML5 / CSS
Javascript

화소 점 처리
기하학 변환

화소 영역 처리

디지털 영상처리 기법

...했더니 영상처리의 해일이 함께 밀려왔습니다

프로젝트 목표

영상처리 프로젝트

파일 선택

선택된 파일 없음

이미지 다운받기

변화 저장

변화 취소

처음으로

세피아

모노톤

흑백

네거티브

이중노출 준비

이중노출 진행

영역반전

영역모노톤

영역흑백

히스토그램

화소 영역 처리

에지 검출

밝기 조정

컨트라스트 조정

투명도 조정

R

G

B

축소

확대

회전

반전

좌우

상하

대각선

Javascript 익히기

HTML5/CSS를 활용한 UI 만들기

다양한 크기의 배열 다루기

알고리즘 구현하고 응용하기

프로젝트의 개요 및 특징

흔치 않은 조합인 자바스크립트와 영상처리 알고리즘의 상호역학 관계 탐색

v1.0은 코드의 효율성과 메모리 사용 정도에 초점을 두지 않고,
다양한 알고리즘을 구현하여 사용자가 위화감 없이 작업할 수 있는지에 주력



프로그램 기능 소개

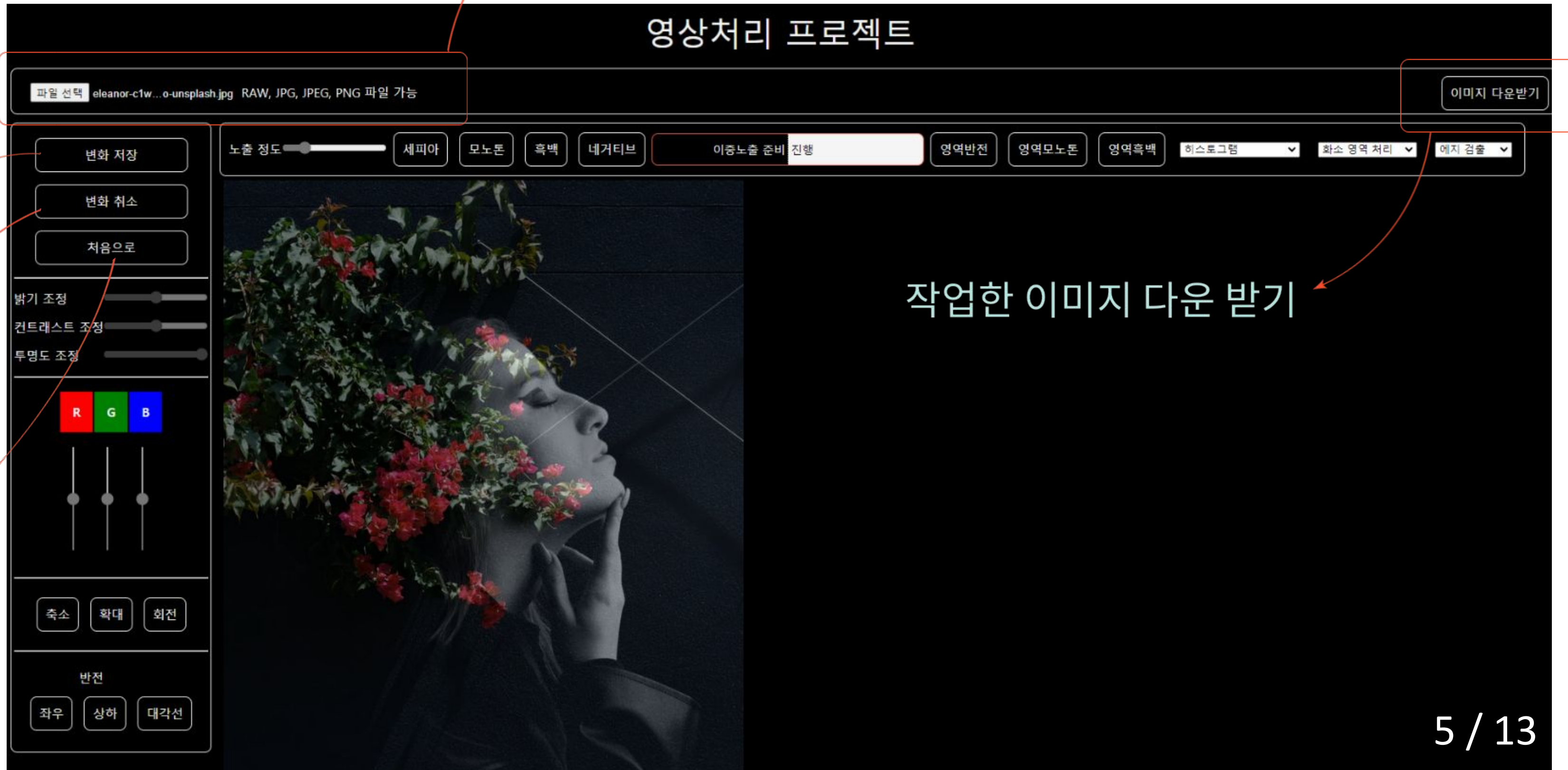
사용자 편의 기능

작업할 이미지 파일 선택:
RAW, PNG, JPG, JPEG 형식 가능

방금 편집
저장

편집 취소

원본으로
되돌리기



영상 처리 기능 - 화소 점 처리

세피아 필터
모노톤 필터
흑백 변환
반전 (네거티브 필름)

히스토그램 스트레칭
엔드-인 탐색
히스토그램 평활화

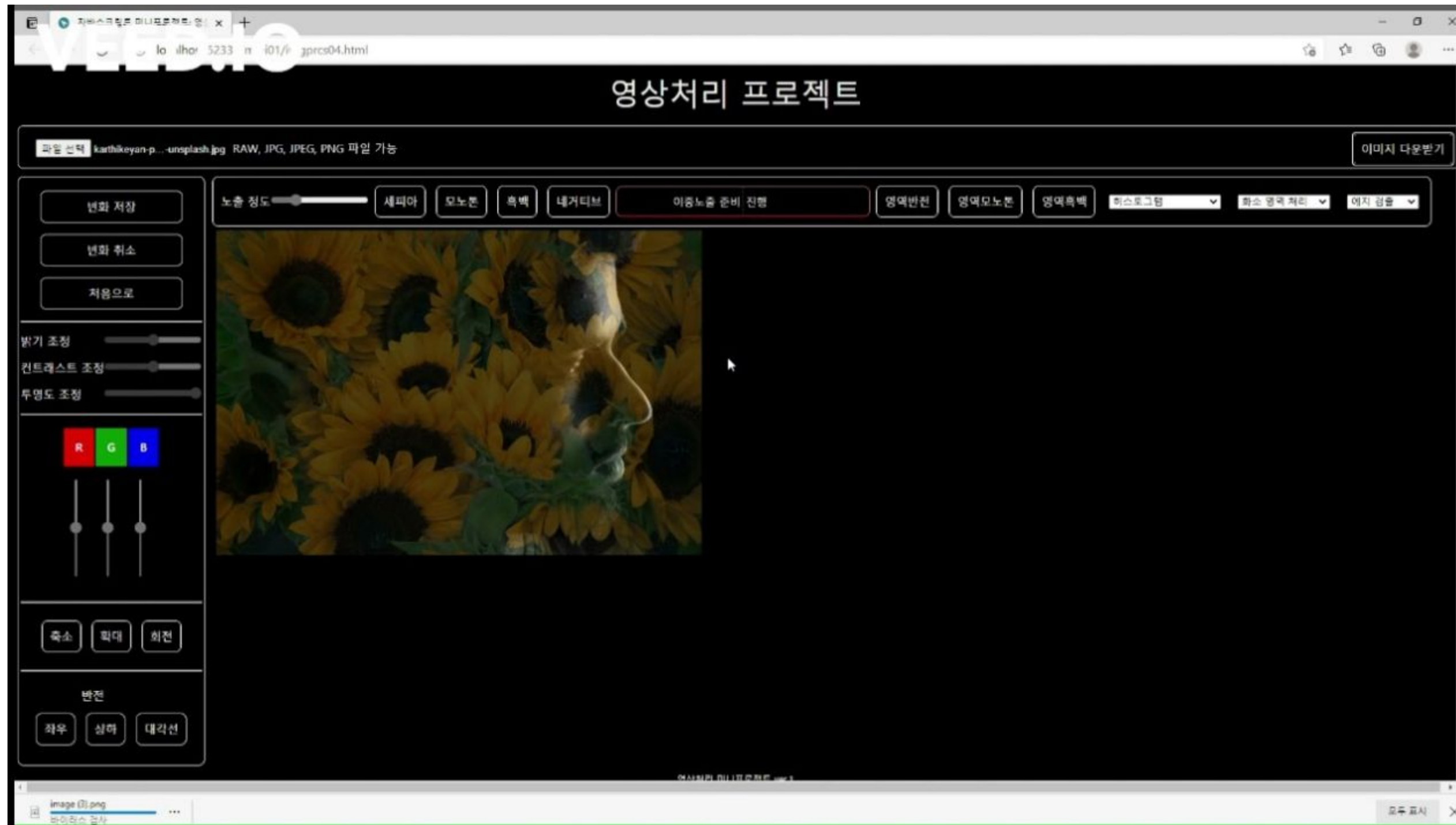


밝기 조정
명암대비 조정
투명도 조정
색채별 비율 조정

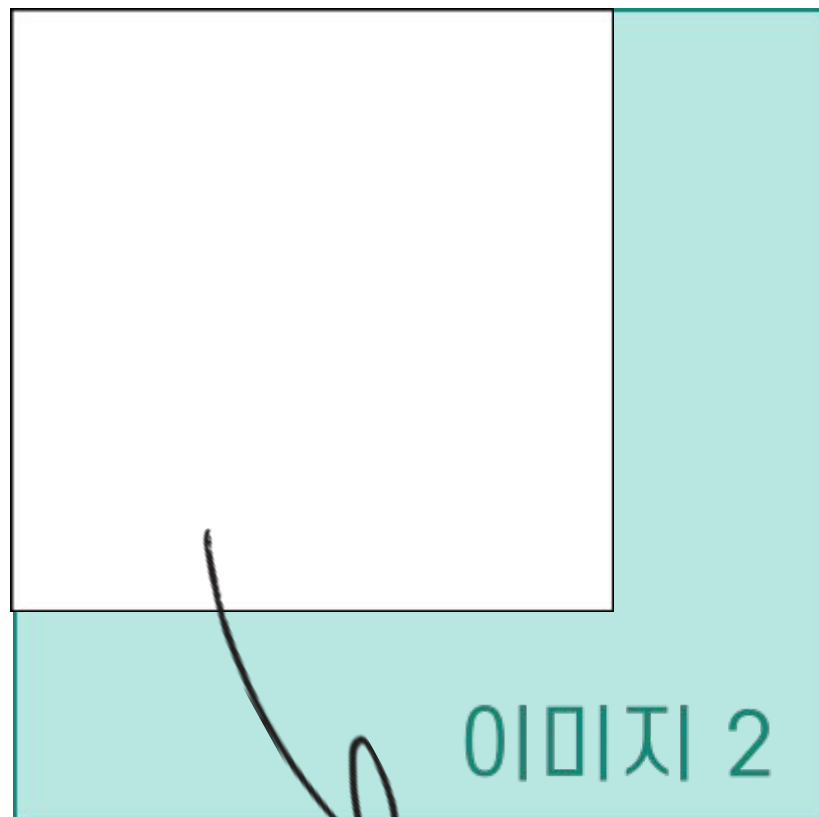
마우스로 영역
지정해서 효과
입히기

이중 노출 효과

<https://youtu.be/qOFR6IDiDWo>



이중노출 효과 구현



사용자가 슬라이드바로 입력한 노출
정도를 받아, 겹치는 부분은 상대적인
노출 비율만큼씩 픽셀값을 합한다.

```
traverse_baseImage (rgb < 4, i < baseImage.height, k < baseImage.width)
{
    if (i < layerImage.height && k < layerImage.width)
    {
        resultImg[rgb][i][k] = baseImage[rgb][i][k] * (1 - blendRatio)
                               + layerImage[rgb][i][k] * blendRatio;
    } else
    {
        resultImg[rgb][i][k] = baseImage[rgb][i][k];
    }
}
```

영상 처리 기능 - 기하학 변환

영상처리 프로젝트

파일 선택 etc09_512.png

이미지 다운받기

변화 저장

변화 취소

처음으로

밝기 조정

컨트레스트 조정

투명도 조정

R

G

B

축소

확대

회전

반전

좌우

상하

대각선

세피아

모노톤

흑백

네거티브

이중노출 준비

진행

영역반전


영역모노톤

영역흑백

히스토그램

화소 영역 처리

에지 검출



축소
확대
회전
반전 (좌우/상하/대각선)

회전 구현

반시계 방향 회전.
백워딩, 중심점, 출력 크기 보완.

$$\begin{bmatrix} h_{out} \\ w_{out} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} h_{in} - C_h \\ w_{in} - C_w \end{bmatrix} + \begin{bmatrix} C'_h \\ C'_w \end{bmatrix}$$

입력 이미지 배열의 각 픽셀 (h_in, w_in)마다
출력 이미지 배열의 픽셀 (h_out, w_out)은
위 식에 따라 도출된다.

C_h, C_w는 (더 커진) 출력 이미지 배열의 중심점,
C'_h, C'_w는 기존 입력 이미지 배열의 중심점

```
/* 입력받은 각도만큼 이미지 반시계 방향으로 회전. 백워딩, 중심점, 출력 크기 보완. */
function toRotate() {
  let degree = parseFloat(prompt("회전 각도 (90도 이하)", 45)); //연산을 위해서 실수 처리
  //하지만 거의 모든 컴에서는 degree가 아닌 radian을 쓴다
  let radian = degree * Math.PI / 180.0 * -1; //f(degree) -> radian
  //출력 영상의 크기
  let w = Math.ceil(inHeight * Math.sin(-radian) + inWidth * Math.cos(-radian))
  let h = Math.ceil(inHeight * Math.cos(-radian) + inWidth * Math.sin(-radian));
  decOutVars2(w, h);
  traverseImgArr(function (rgb, i, k) { //출력 배열 255로 값 초기화
    if (rgb == 3)
      outImageArray[rgb][i][k] = 0; // 이미지 없는 부분은 투명하게
    else
      outImageArray[rgb][i][k] = 255;
  }, outHeight, outWidth);
  /* 역방향 회전 공식
  xs = cos * xd - sin * yd;
  ys = -sin * xd + cos * yd; */
  var xd, yd, xs, ys;
  //중앙점 처리
  var inCh = Math.round(inHeight / 2);
  var inCw = Math.round(inWidth / 2);
  var outCh = Math.round(outHeight / 2);
  var outCw = Math.round(outWidth / 2);
  //영상처리 알고리즘
  traverseImgArr(function (rgb, i, k) {
    xs = i;
    ys = k;
    xd = parseInt(Math.cos(radian) * (xs - outCh) - Math.sin(radian) * (ys - outCw) + inCh);
    yd = parseInt(Math.sin(radian) * (xs - outCh) + Math.cos(radian) * (ys - outCw) + inCw);
    if ((xd > -1 && xd < inHeight) && (yd > -1 && yd < inWidth))
      outImageArray[rgb][xs][ys] = inImageArray[rgb][xd][yd];
  }, outHeight, outWidth);
  displayImage();
}
```

영상 처리 기능 - 화소 영역 처리

영상처리 프로젝트

파일 선택 cat12_512.png

이미지 다운받기

변화 저장

변화 취소

처음으로

밝기 조정

컨트래스트 조정

투명도 조정

R

G

B

축소

확대

회전

반전

좌우

상하

대각선

세피아

모노톤

흑백

네거티브

이중노출 준비

진행

영역반전

영역모노톤

영역흑백

히스토그램

화소 영역 처리

LoG 5x5

에지 검출

이동과 차분

로버츠

프리윗


소벨

라플라시안

LoG 5x5

DoG 7x7

DoG 9x9



엠보싱
블러링 (마스크 크기 임의 지정)
샤프닝 (두 가지 버전)
가우시안 스무딩
에지 검출



이중노출 효과 업그레이드

- layer 이미지 위치를 사용자가 지정하도록

ver.2 목표

HSV 변환과 색채
관련 기능 구현

마우스로 자유 영역
지정해서 효과 입히기
기능 구현



코드 다운받기

<https://github.com/YS12/not-photoshop>

이 프로젝트에 대해 더 보기

<https://ys12.github.io/not-photoshop/>