



# OpenCV를 이용한 객체 움직임 감지 프로젝트

# CONTENTS

---

01 Introduction

02 Results

03 References

01



Introduction

# 01 Introduction

## 01 Dataset

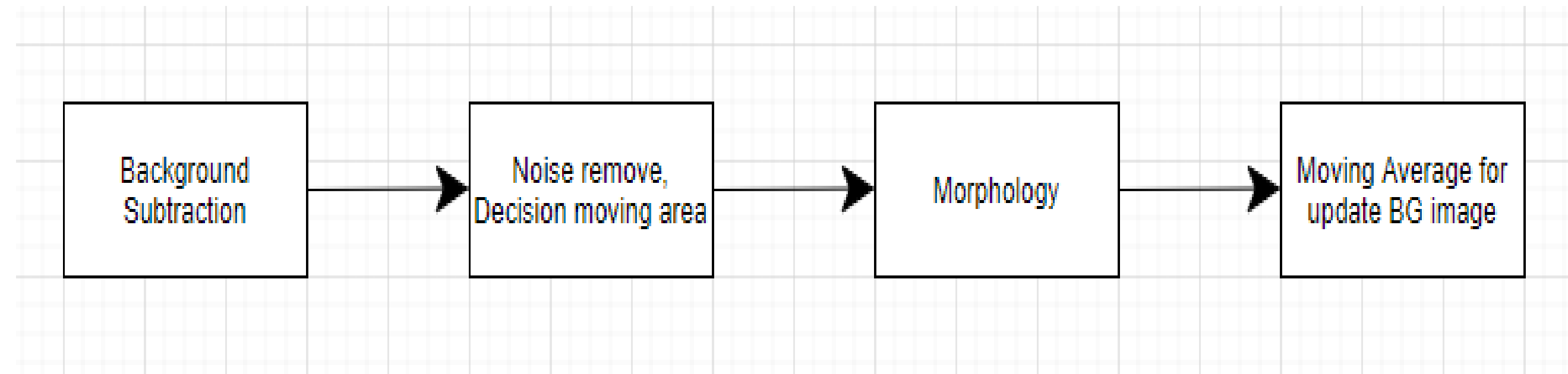
## 02 Background Subtraction

## 03 Noise remove, Decision moving area

## 04 Morphology

## 05 Moving Average for update BG image

# Dataset



위와 같은 과정 순서대로 적용하여 동영상 속의 움직임이 있는 객체를 탐지해냄.

# 01 Motion Detection Process

01 Introduction

02 Background Subtraction

03 Noise remove,  
Decision moving area

04 Morphology

05 Moving Average  
for update BG image

## Background Subtraction

객체를 포함하는 영상에서 객체가 없는 배경 영상을 빼는 방법을 통해 배경을 모두 제거하고 객체만 남기는 방법을 사용함.

자료 사진	핵심 코드
	<pre>while (1) {     if (!stream1.read(frame))         break;     if (old_frame.empty())     {         old_frame = frame.clone();         continue;     }     subtract(old_frame, frame, sub_frame);     imshow("frame", frame);     imshow("sub_frame", sub_frame);     old_frame = frame.clone();     if (waitKey(5) &gt;= 0)         break; }</pre>

# 01 Motion Detection Process

01 Introduction

02 Background Subtraction

03 Noise remove,  
Decision moving area

04 Morphology

05 Moving Average  
for update BG image

## Noise remove

이미지에서 이미지 픽셀값이 문턱값보다 크면 고정된 값으로 할당하고, 작으면 다른 고정값으로 할당하는 방식을 적용하여 Noise를 제거함.

자료 사진	핵심 코드
	<pre>cvtColor(frame, frame_binary, COLOR_BGR2GRAY); absdiff(bg_frame_binary, frame_binary, sub_frame); threshold(sub_frame, sub_frame, 80, 255, THRESH_BINARY);</pre>

# 01 Motion Detection Process

01 Introduction

02 Background Subtraction

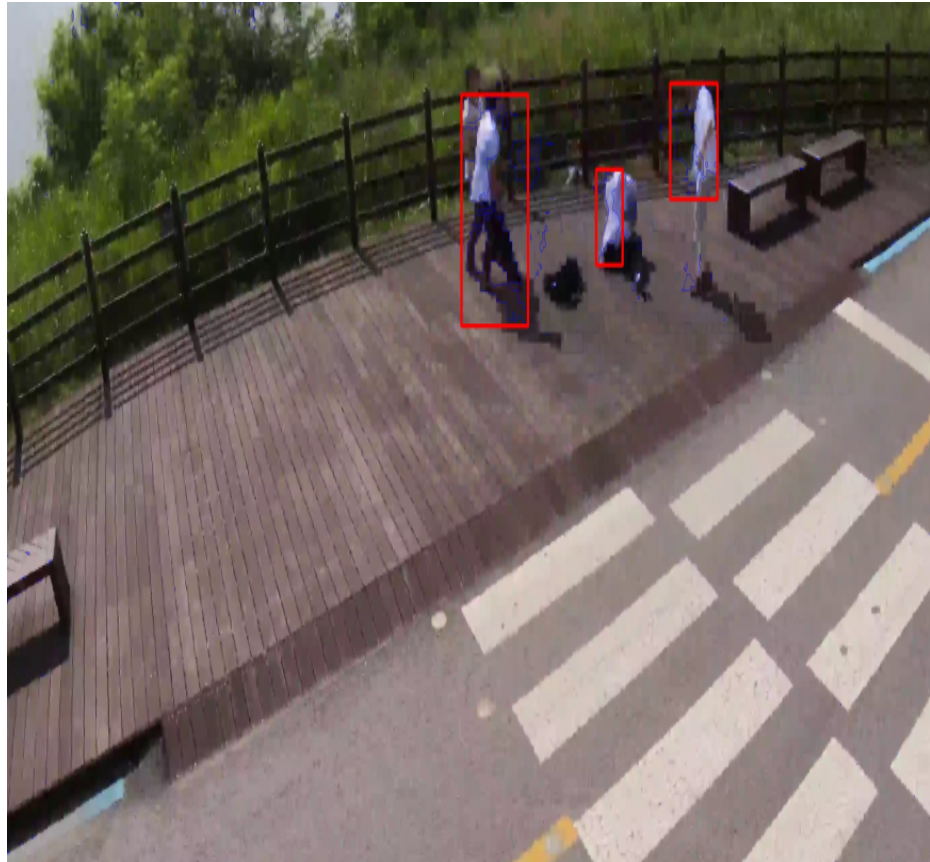
03 Noise remove,  
Decision moving area

04 Morphology

05 Moving Average  
for update BG image

## Decision moving area

영상 내 움직이는 객체의 윤곽선(컨투어)을 검출하고 윤곽선 좌표 정보를 이용해 객체를 사각형으로 감쌈.

자료 사진	핵심 코드
	<pre>// find contour vector&lt; vector&lt; Point&gt; &gt; contours; vector&lt; Vec4i&gt; hierarchy; findContours(sub_frame.clone(), contours, hierarchy, RETR_CCOMP, CHAIN_APPROX_SIMPLE); drawContours(frame, contours, -1, CV_RGB(0, 0, 255), 1, 8, hierarchy); //Blob labeling vector&lt; Rect &gt; v_rect; for (auto it : contours) {     Rect mr = boundingRect(Mat(it));     if (mr.width &gt; TH_WIDTH    mr.height &gt; TH_HEIGHT) {         v_rect.push_back(mr);     } } draw_rect(frame, v_rect)</pre>

# 01 Motion Detection Process

01 Introduction

02 Background Subtraction

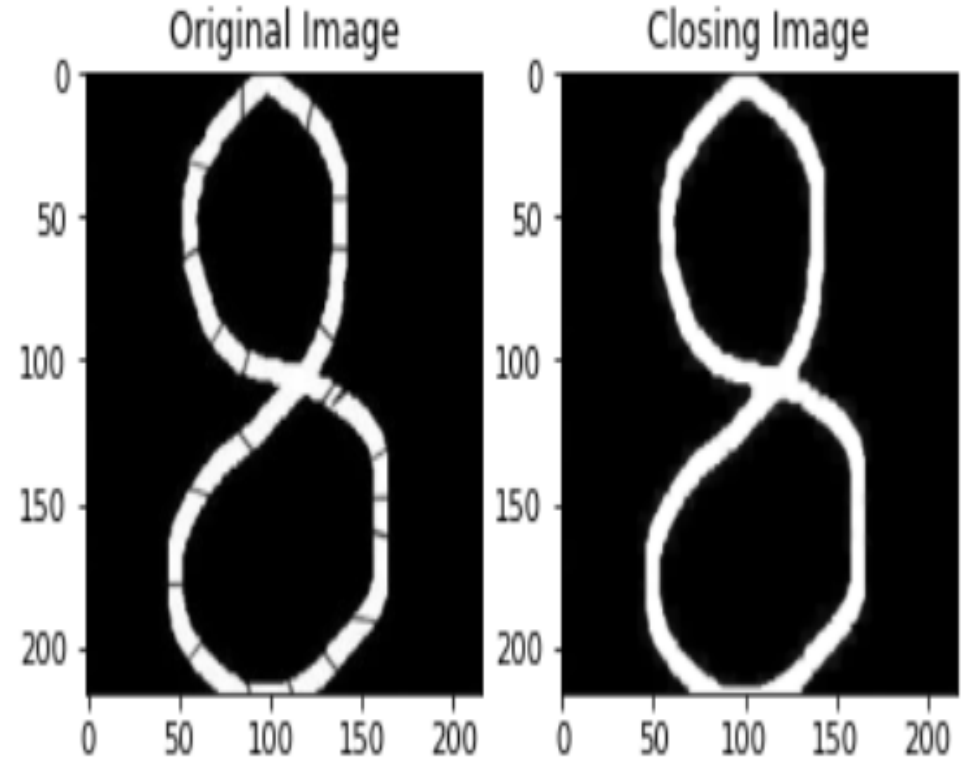
03 Noise remove,  
Decision moving area

04 Morphology

05 Moving Average  
for update BG image

## Morphology

Morphology 연산에서 팽창 연산 후 침식 연산을 적용하는 것을 닫힘(closing) 연산이라고 하는데 닫힘 연산을 사용하여 주변보다 어두운 노이즈를 제거하고 끊어져 보이는 개체를 연결하거나 구멍을 메움

적용 이론	핵심 코드
	<pre>morphologyEx(sub_frame, sub_frame, MORPH_CLOSE, element);</pre>



# 01 Motion Detection Process

01 Introduction

02 Background Subtraction

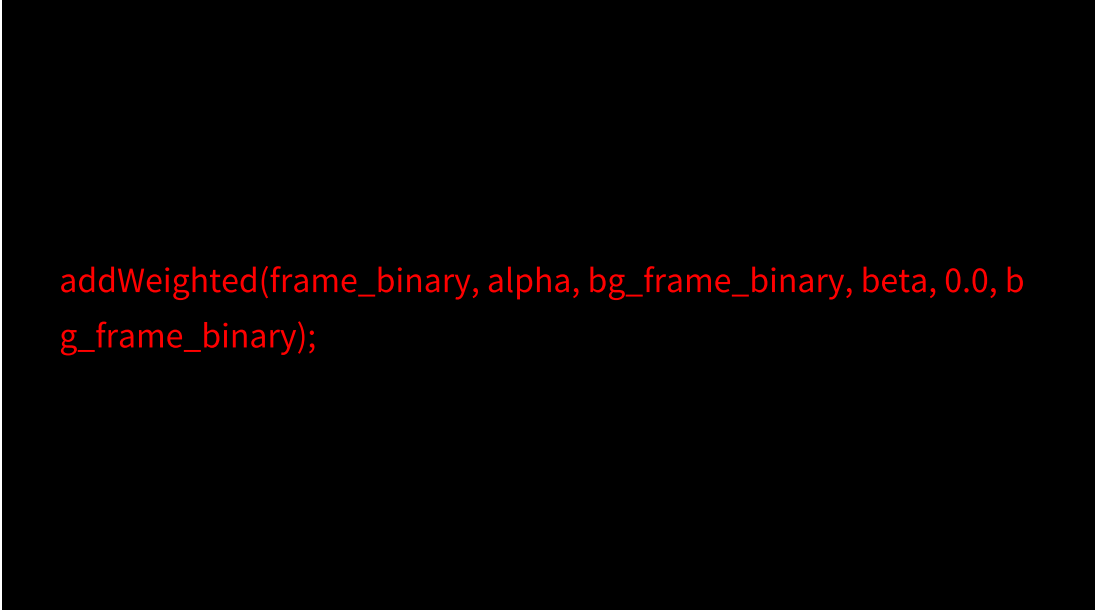
03 Noise remove,  
Decision moving area

04 Morphology

05 Moving Average  
for update BG image

## Moving Average for update BG image

정적 배경 모델 사용시 문제점은 새로 나타난 객체가 고정되었을 때 이것을 지속적으로 객체로 인식한다는 것이다.  
이를 해결하기 위해 현재 프레임과 배경 영상의 가중치 합을 이용해서 배경 영상을 업데이트하는 방법을 사용하였다.

적용 이론	핵심 코드
$B(x,y,t) = \alpha \cdot I(x,y,t) + (1 - \alpha) \cdot B(x,y,t-1)$  $B(x,y,t)$ : 갱신된 배경 영상 $\alpha$ : 현재 프레임에 대한 가중치 $I(x,y,t)$ : 현재 프레임 $B(x,y,t-1)$ : 이전 프레임까지의 배경 영상	 <pre>addWeighted(frame_binary, alpha, bg_frame_binary, beta, 0.0, b g_frame_binary);</pre>

# 02 | Results

# Results



03

참조

# References

---

[1] <https://opencv.org/>

[2] <https://deep-learning-study.tistory.com/273>

[3] <https://bkshin.tistory.com/category/OpenCV>

---

감사합니다

---