# SamOpenCVWeb

## 비주얼 OpenCV 에디터 교재

노드 기반 시각적 프로그래밍으로 배우는 영상처리

본 교재는 SamOpenCVWeb의 모든 기능을 다룹니다.
83개 노드 레퍼런스 + 30개 실습 튜토리얼

# Part 1: 소개

## SamOpenCVWeb이란?

SamOpenCVWeb은 Node-RED 스타일의 비주얼 프로그래밍 환경에서 OpenCV 영상처리를 수행할 수 있는 웹 기반 도구입니다. 코드를 작성하지 않고도 노드를 드래그, 연결하여 복잡한 영상처리 파이프라인을 구성할 수 있습니다.

Python Flask 백엔드와 HTML5 Canvas 프론트엔드로 구성되어 있으며, 브라우저에서 바로 사용할 수 있습니다.

## 시스템 요구사항

- Python 3.8 이상
- OpenCV (cv2) 라이브러리
- Flask 웹 프레임워크
- 최신 웹 브라우저 (Chrome, Firefox, Edge 등)

## 설치 및 실행

1. 필수 패키지 설치: pip install flask opencv-python numpy
2. 서버 실행: python app.py
3. 브라우저에서 http://localhost:5000 접속

## 화면 구성

- 노드 팔레트 (왼쪽): 17개 카테고리, 83개 노드를 드래그하여 추가
- 캔버스 (중앙): 노드를 배치하고 연결하는 작업 영역
- 속성 패널 (오른쪽): 선택한 노드의 속성을 편집
- 미리보기 패널 (오른쪽 하단): 실행 결과 이미지를 표시
- 도구 모음 (상단): Execute, Clear, Save, Load, Code 버튼

# Part 2: 기본 사용법

## 노드 추가

왼쪽 팔레트에서 원하는 노드를 캔버스로 드래그&드롭합니다. 또는 노드를 클릭하면 캔버스 중앙에 추가됩니다.

## 노드 연결

노드의 출력 포트(오른쪽 원)를 다른 노드의 입력 포트(왼쪽 원)로 드래그하면 연결선이 만들어집니다. 가까이 배치하면 자동 연결됩니다.

## 속성 편집

노드를 클릭하면 오른쪽 속성 패널에 해당 노드의 속성이 표시됩니다. 값을 변경하면 실행 시 반영됩니다.

## 실행 & 미리보기

- Execute (Ctrl+Enter): 전체 파이프라인을 실행합니다.
- 노드를 더블클릭하면 해당 노드까지만 실행합니다.
- 파일 노드(Image Read 등)를 더블클릭하면 파일 선택 대화상자가 열립니다.

## 저장 & 불러오기

- Save: 현재 파이프라인을 JSON 파일로 저장합니다.
- Load: 저장된 파이프라인을 불러옵니다.
- 브라우저의 로컬 저장소를 사용합니다.

## 코드 생성

Code 버튼을 클릭하면 현재 파이프라인에 해당하는 Python/OpenCV 코드가 자동 생성됩니다. 이 코드를 복사하여 독립 스크립트로 사용할 수 있습니다.

## 단축키

| | |
|---|---|
| **Ctrl+Enter** | 파이프라인 실행 |
| **Ctrl+Z** | 실행 취소 (Undo) |
| **Ctrl+C / X / V** | 복사 / 잘라내기 / 붙여넣기 |
| **Ctrl+A** | 전체 선택 |
| **Delete** | 선택 노드 삭제 |
| **Ctrl+S** | 파이프라인 저장 |
| **마우스 휠** | 캔버스 확대/축소 |
| **마우스 드래그 (빈 영역)** | 캔버스 이동 |
| **Shift+드래그** | 다중 선택 |

# Part 3: 노드 레퍼런스

SamOpenCVWeb에서 사용 가능한 83개 노드를 카테고리별로 설명합니다.

## 입출력 (I/O)

### 이미지 읽기 (Image Read)

cv2.imread(filename, flags)

파일에서 이미지를 읽어옵니다. BMP, JPEG, PNG, TIFF 등을 지원합니다.

입력: 없음 | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| File Path | text | | 파일 경로 |

### 이미지 표시 (Image Show)

cv2.imshow(winname, mat)

이미지를 미리보기 패널에 표시합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Window Name | text | Output | 윈도우 이름 |

### 이미지 저장 (Image Write)

cv2.imwrite(filename, img)

이미지를 파일로 저장합니다. 비디오 출력 모드도 지원합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| File Path | text | output.png | 저장 경로 |
| Format | select | PNG | 형식 |
| Quality | number | 95 | 품질 |
| Video Output | checkbox | false | 비디오 출력 |
| Codec | select | mp4v | 코덱 |
| FPS | number | 30 | 프레임율 |

### 비디오 읽기 (Video Read)

cv2.VideoCapture(filename)

비디오 파일에서 프레임을 읽습니다. 단일/루프 모드 지원.

입력: 없음 | 출력: frame

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Video Path | text | | 경로 |
| Mode | select | single | 모드 |
| Frame Index | number | 0 | 프레임 |
| Start | number | 0 | 시작 |
| End | number | -1 | 끝 |
| Step | number | 1 | 스텝 |

---

### 카메라 캡처 (Camera Capture)

cv2.VideoCapture(index)

웹캠에서 한 프레임을 캡처합니다.

입력: 없음 | 출력: frame

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Camera Index | number | 0 | 카메라 인덱스 |

---

## 색상 (Color)

### 색상 변환 (CvtColor)

cv2.cvtColor(src, code)

이미지의 색상 공간을 변환합니다. BGR↔Gray, BGR↔HSV 등.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Color Code | select | COLOR_BGR2GRAY | 변환 코드 |

---

## 범위 필터 (InRange)

cv2.inRange(src, lowerb, upperb)

지정 범위 내 픽셀만 선택하는 이진 마스크를 생성합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Lower B/H | number | 0 | 하한 |
| Lower G/S | number | 0 | |
| Lower R/V | number | 0 | |
| Upper B/H | number | 255 | 상한 |
| Upper G/S | number | 255 | |
| Upper R/V | number | 255 | |

---

## 히스토그램 평활화 (Histogram EQ)

cv2.equalizeHist / cv2.createCLAHE

히스토그램을 균일하게 분포시켜 대비를 개선합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Use CLAHE | checkbox | false | CLAHE 사용 |
| Clip Limit | number | 2.0 | 클립 제한 |
| Tile Size | number | 8 | 타일 크기 |

---

## 채널 분리 (Split Channels)

cv2.split(m)

다채널 이미지를 개별 채널로 분리합니다.

입력: image | 출력: ch 0, ch 1, ch 2

---

## 채널 병합 (Merge Channels)

cv2.merge(mv)

단일 채널들을 합쳐 다채널 이미지를 만듭니다.

입력: ch 0, ch 1, ch 2 | 출력: image

---

# 필터/블러 (Filter)

## 가우시안 블러 (Gaussian Blur)

cv2.GaussianBlur(src, ksize, sigmaX)

가우시안 필터로 이미지를 부드럽게 합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Kernel Size | number | 5 | 커널 크기 (홀수) |
| Sigma X | number | 0 | 표준편차 |

---

## 미디언 블러 (Median Blur)

cv2.medianBlur(src, ksize)

미디언 필터로 소금-후추 노이즈를 제거합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Kernel Size | number | 5 | 커널 크기 |

---

## 양방향 필터 (Bilateral Filter)

cv2.bilateralFilter(src, d, sigmaColor, sigmaSpace)

에지를 보존하면서 노이즈를 제거합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Diameter | number | 9 | 직경 |
| Sigma Color | number | 75 | 색상 시그마 |
| Sigma Space | number | 75 | 공간 시그마 |

---

## 박스 필터 (Box Filter)

cv2.boxFilter(src, ddepth, ksize)

평균값 필터로 이미지를 부드럽게 합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Width | number | 5 | 너비 |
| Height | number | 5 | 높이 |
| Normalize | checkbox | true | 정규화 |

---

### 샤프닝 (Sharpen)

Unsharp Mask

언샤프 마스크로 이미지를 선명하게 만듭니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Strength | number | 1.5 | 강도 |

---

### 2D 필터 (Filter2D)

cv2.filter2D(src, ddepth, kernel)

커스텀 커널로 컨볼루션을 수행합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Kernel Size | number | 3 | 커널 크기 |
| Preset | select | sharpen | 프리셋 |

---

## 에지 검출 (Edge)

### 캐니 에지 (Canny)

cv2.Canny(image, threshold1, threshold2)

캐니 알고리즘으로 에지를 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Threshold 1 | number | 100 | 하위 임계값 |
| Threshold 2 | number | 200 | 상위 임계값 |

---

### 소벨 (Sobel)

cv2.Sobel(src, ddepth, dx, dy, ksize)

소벨 연산자로 이미지 미분을 계산합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| dx | number | 1 | X 미분 차수 |
| dy | number | 0 | Y 미분 차수 |
| Kernel Size | number | 3 | 커널 |

---

### 라플라시안 (Laplacian)

cv2.Laplacian(src, ddepth, ksize)

라플라시안으로 2차 미분 에지를 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Kernel Size | number | 3 | 커널 |

---

### 샤르 (Scharr)

cv2.Scharr(src, ddepth, dx, dy)

소벨보다 정밀한 3x3 1차 미분을 계산합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| dx | number | 1 | X |
| dy | number | 0 | Y |

---

## 임계값 (Threshold)

### 임계값 (Threshold)

cv2.threshold(src, thresh, maxval, type)

고정 임계값으로 이진 이미지를 생성합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Threshold | number | 127 | 임계값 |
| Max Value | number | 255 | 최대값 |
| Type | select | THRESH_BINARY | 유형 |

---

### 적응형 임계값 (Adaptive Threshold)

cv2.adaptiveThreshold(src, maxValue, method, type, blockSize, C)

영역별 적응적 임계값을 적용합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Max Value | number | 255 | 최대값 |
| Method | select | ADAPTIVE_THRESH_GAUSSIAN_C | 방법 |
| Type | select | THRESH_BINARY | 유형 |
| Block Size | number | 11 | 블록 |
| C | number | 2 | 상수 |

---

### 오츠 임계값 (Otsu Threshold)

cv2.threshold(src, 0, maxval, THRESH_OTSU)

오츠 알고리즘으로 최적 임계값을 자동 결정합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Max Value | number | 255 | 최대값 |

---

## 형태학 (Morphology)

### 형태학 연산 (Morphology Ex)

cv2.morphologyEx(src, op, kernel)

열기, 닫기, 그래디언트 등 형태학 변환을 수행합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Operation | select | MORPH_OPEN | 연산 |
| Kernel Size | number | 5 | 커널 |
| Shape | select | MORPH_RECT | 모양 |
| Iterations | number | 1 | 반복 |

---

### 팽창 (Dilate)

cv2.dilate(src, kernel, iterations)

밝은 영역을 확장합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Kernel Size | number | 5 | 커널 |
| Iterations | number | 1 | 반복 |

---

### 침식 (Erode)

cv2.erode(src, kernel, iterations)

밝은 영역을 축소합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Kernel Size | number | 5 | 커널 |
| Iterations | number | 1 | 반복 |

---

### 구조 요소 (Structuring Element)

cv2.getStructuringElement(shape, ksize)

형태학 연산용 커널을 생성합니다.

입력: 없음 | 출력: element

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Shape | select | MORPH_RECT | 모양 |
| Width | number | 5 | 너비 |
| Height | number | 5 | 높이 |

---

## 컨투어 (Contour)

## 컨투어 그리기 (Draw Contours)

cv2.findContours + cv2.drawContours

이진 이미지에서 윤곽선을 찾아 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | RETR_EXTERNAL | 검색 |
| Approx | select | CHAIN_APPROX_SIMPLE | 근사 |
| Index | number | -1 | 인덱스 |
| Thickness | number | 2 | 두께 |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |

## 바운딩 박스 (Bounding Rect)

cv2.boundingRect + cv2.rectangle

컨투어의 외접 사각형을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | RETR_EXTERNAL | |
| Approx | select | CHAIN_APPROX_SIMPLE | |
| Thickness | number | 2 | |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |

## 최소 외접원 (Min Enclosing Circle)

cv2.minEnclosingCircle + cv2.circle

컨투어의 최소 외접원을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | RETR_EXTERNAL | |
| Approx | select | CHAIN_APPROX_SIMPLE | |
| Thickness | number | 2 | |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |

## 볼록 껍질 (Convex Hull)

cv2.convexHull + cv2.drawContours

컨투어의 볼록 껍질을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | RETR_EXTERNAL | |
| Approx | select | CHAIN_APPROX_SIMPLE | |
| Thickness | number | 2 | |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |

---

## 다각형 근사 (Approx Poly)

cv2.approxPolyDP

컨투어를 다각형으로 근사합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Epsilon | number | 0.02 | 정밀도 |
| Closed | checkbox | true | 닫힘 |
| Mode | select | RETR_EXTERNAL | |
| Approx | select | CHAIN_APPROX_SIMPLE | |
| Thickness | number | 2 | |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |

---

## 면적 필터 (Contour Area)

cv2.contourArea + cv2.drawContours

면적 범위로 컨투어를 필터링합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | RETR_EXTERNAL | |
| Approx | select | CHAIN_APPROX_SIMPLE | |
| Min | number | 100 | 최소 |
| Max | number | 100000 | 최대 |
| Thickness | number | 2 | |
| R | number | 0 | |
| G | number | 255 | |

| | | | |
|---|---|---|---|
| B | number | 0 | |

### 컨투어 속성 (Contour Properties)

cv2.contourArea + cv2.arcLength + cv2.moments

면적, 둘레, 중심점 등의 속성을 표시합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | RETR_EXTERNAL | |
| Approx | select | CHAIN_APPROX_SIMPLE | |
| Area | checkbox | true | 면적 |
| Perimeter | checkbox | true | 둘레 |
| Center | checkbox | true | 중심 |

## 변환 (Transform)

### 리사이즈 (Resize)

cv2.resize(src, dsize, fx, fy)

이미지 크기를 변경합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Width | number | 0 | 너비 |
| Height | number | 0 | 높이 |
| Scale X | number | 0.5 | X비율 |
| Scale Y | number | 0.5 | Y비율 |
| Interp | select | INTER_LINEAR | 보간 |

### 회전 (Rotate)

cv2.getRotationMatrix2D + cv2.warpAffine

이미지를 중심 기준으로 회전합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Angle | number | 90 | 각도 |

## 뒤집기 (Flip)

cv2.flip(src, flipCode)

이미지를 수평/수직으로 뒤집습니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | Horizontal (1) | 방향 |

---

## 자르기 (Crop)

img[y:y+h, x:x+w]

이미지의 관심 영역(ROI)을 잘라냅니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| X | number | 0 | X |
| Y | number | 0 | Y |
| Width | number | 100 | 너비 |
| Height | number | 100 | 높이 |

---

## 아핀 변환 (Warp Affine)

cv2.warpAffine(src, M, dsize)

2x3 변환 행렬로 아핀 변환을 적용합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| M[0,0] | number | 1 | |
| M[0,1] | number | 0 | |
| tx | number | 0 | X이동 |
| M[1,0] | number | 0 | |
| M[1,1] | number | 1 | |
| ty | number | 0 | Y이동 |

---

## 투시 변환 (Warp Perspective)

cv2.getPerspectiveTransform + cv2.warpPerspective

4꼭짓점 대응으로 투시 변환을 적용합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| SrcX1 | number | 0 | |
| SrcY1 | number | 0 | |
| SrcX2 | number | 300 | |
| SrcY2 | number | 0 | |
| SrcX3 | number | 300 | |
| SrcY3 | number | 300 | |
| SrcX4 | number | 0 | |
| SrcY4 | number | 300 | |
| DstX1 | number | 0 | |
| DstY1 | number | 0 | |
| DstX2 | number | 300 | |
| DstY2 | number | 0 | |
| DstX3 | number | 300 | |
| DstY3 | number | 300 | |
| DstX4 | number | 0 | |
| DstY4 | number | 300 | |

---

## 리맵 (Remap)

cv2.remap(src, map1, map2, interpolation)

배럴/핀쿠션 왜곡을 보정합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| K | number | 0.5 | 왜곡계수 |
| Interp | select | INTER_LINEAR | 보간 |

---

### 이미지 붙여넣기 (Paste Image)

Image Paste (overlay at x,y)

작은 이미지를 큰 이미지 위에 지정 좌표에 덮어씌웁니다. overwrite/blend/alpha 모드.

입력: image, overlay | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| X | number | 0 | X좌표 |
| Y | number | 0 | Y좌표 |
| Mode | select | overwrite | 모드 |
| Opacity | number | 1.0 | 투명도 |

---

## 히스토그램 (Histogram)

### 히스토그램 계산 (Calc Histogram)

cv2.calcHist

이미지 히스토그램을 계산하고 시각화합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Bins | number | 256 | 빈 |
| Normalize | checkbox | true | 정규화 |

---

## 특징 검출 (Feature)

### 컨투어 찾기 (Find Contours)

cv2.findContours(image, mode, method)

이진 이미지에서 윤곽선을 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | RETR_EXTERNAL | 검색 |
| Approx | select | CHAIN_APPROX_SIMPLE | 근사 |

---

## 허프 직선 (Hough Lines)

cv2.HoughLines(image, rho, theta, threshold)

허프 변환으로 직선을 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Rho | number | 1 | 거리 |
| Theta (π/N) | number | 180 | 각도 |
| Threshold | number | 100 | 임계 |

---

## 해리스 코너 (Harris Corner)

cv2.cornerHarris(src, blockSize, ksize, k)

해리스 코너 검출기로 모서리를 찾습니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Block | number | 2 | 블록 |
| Sobel K | number | 3 | 커널 |
| K | number | 0.04 | 파라미터 |

---

## 좋은 특징점 (Good Features)

cv2.goodFeaturesToTrack

Shi-Tomasi 알고리즘으로 강한 코너를 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Max | number | 100 | 최대 |
| Quality | number | 0.01 | 품질 |
| MinDist | number | 10 | 거리 |

---

## ORB 특징점 (ORB)

cv2.ORB_create(nfeatures)

ORB 특징점을 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Features | number | 500 | 특징수 |

---

### FAST 코너 (FAST)

cv2.FastFeatureDetector_create

FAST 알고리즘으로 빠르게 코너를 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Threshold | number | 25 | 임계값 |
| NMS | checkbox | true | 비극대 |

---

### 특징 매칭 (Match Features)

cv2.ORB_create + cv2.BFMatcher

ORB로 두 이미지의 특징점을 매칭합니다.

입력: image, image2 | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Features | number | 500 | 수 |
| Ratio | number | 0.75 | 비율 |

---

## 드로잉 (Drawing)

### 직선 그리기 (Draw Line)

cv2.line(img, pt1, pt2, color, thickness)

이미지에 직선을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| X1 | number | 10 | |
| Y1 | number | 10 | |
| X2 | number | 200 | |
| Y2 | number | 200 | |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |
| Thickness | number | 2 | |

---

## 사각형 그리기 (Draw Rectangle)

cv2.rectangle(img, pt1, pt2, color, thickness)

이미지에 사각형을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| X | number | 10 | |
| Y | number | 10 | |
| W | number | 100 | |
| H | number | 100 | |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |
| Thickness | number | 2 | |

---

## 원 그리기 (Draw Circle)

cv2.circle(img, center, radius, color, thickness)

이미지에 원을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| CX | number | 100 | |
| CY | number | 100 | |
| R | number | 50 | |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |
| Thickness | number | 2 | |

---

## 타원 그리기 (Draw Ellipse)

cv2.ellipse(img, center, axes, angle, ...)

이미지에 타원을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| CX | number | 100 | |
| CY | number | 100 | |
| W | number | 80 | |
| H | number | 40 | |
| Angle | number | 0 | |
| R | number | 0 | |

| | | | |
|---|---|---|---|
| G | number | 255 | |
| B | number | 0 | |
| Thickness | number | 2 | |

### 텍스트 그리기 (Draw Text)

cv2.putText(img, text, org, font, scale, color, thickness)

이미지에 텍스트를 표시합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Text | text | Hello | 텍스트 |
| X | number | 50 | |
| Y | number | 50 | |
| Scale | number | 1.0 | 크기 |
| R | number | 255 | |
| G | number | 255 | |
| B | number | 255 | |
| Thickness | number | 2 | |

### 다각선 그리기 (Draw Polylines)

cv2.polylines(img, [pts], isClosed, color, thickness)

다각형 또는 다각선을 그립니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Points | text | 10,10;100,50;50,100 | 좌표 |
| Closed | checkbox | true | 닫힘 |
| R | number | 0 | |
| G | number | 255 | |
| B | number | 0 | |
| Thickness | number | 2 | |

## 산술 연산 (Arithmetic)

## 가중 합성 (Add Weighted)

cv2.addWeighted(src1, alpha, src2, beta, gamma)

두 이미지를 가중치로 합성합니다. 크기 불일치 시 sizeMismatch 옵션 사용.

입력: image, image2 | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Alpha | number | 0.5 | |
| Beta | number | 0.5 | |
| Gamma | number | 0 | |
| Size Mismatch | select | error | 크기불일치 |

---

## 비트 AND (Bitwise AND)

cv2.bitwise_and(src1, src2)

비트 AND 연산. 마스크 적용에 사용합니다.

입력: image, image2 | 출력: image

---

## 비트 OR (Bitwise OR)

cv2.bitwise_or(src1, src2)

비트 OR 연산을 수행합니다.

입력: image, image2 | 출력: image

---

## 비트 NOT (Bitwise NOT)

cv2.bitwise_not(src)

모든 비트를 반전합니다.

입력: image | 출력: image

---

## 덧셈 (Add)

cv2.add(src1, src2)

픽셀별 덧셈 (포화).

입력: image, image2 | 출력: image

---

### 뺄셈 (Subtract)

cv2.subtract(src1, src2)

픽셀별 뺄셈 (포화).

입력: image, image2 | 출력: image

---

### 곱셈 (Multiply)

cv2.multiply(src1, src2, scale)

픽셀별 곱셈.

입력: image, image2 | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Scale | number | 1.0 | 스케일 |

---

### 절대 차이 (AbsDiff)

cv2.absdiff(src1, src2)

절대 차이. 변화 감지에 사용합니다.

입력: image, image2 | 출력: image

---

### 비트 XOR (Bitwise XOR)

cv2.bitwise_xor(src1, src2)

비트 XOR 연산.

입력: image, image2 | 출력: image

---

## 검출 (Detection)

## 하르 캐스케이드 (Haar Cascade)

cv2.CascadeClassifier.detectMultiScale()

얼굴, 눈, 미소 등을 검출합니다. coords 포트로 좌표 리스트 출력.

입력: image | 출력: image, coords

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Type | select | face | 유형 |
| Scale | number | 1.1 | 스케일 |
| Neighbors | number | 5 | 이웃 |
| MinW | number | 30 | 최소폭 |
| MinH | number | 30 | 최소높이 |

---

## 허프 원 (Hough Circles)

cv2.HoughCircles(image, method, dp, minDist, ...)

허프 변환으로 원을 검출합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| dp | number | 1.2 | 역비율 |
| MinDist | number | 50 | 최소간격 |
| P1 | number | 100 | 캐니 |
| P2 | number | 30 | 축적기 |
| MinR | number | 0 | 최소R |
| MaxR | number | 0 | 최대R |

---

## 템플릿 매칭 (Template Match)

cv2.matchTemplate(image, templ, method)

템플릿으로 일치 영역을 찾습니다. matches 포트로 좌표 리스트 출력.

입력: image, template | 출력: image, matches

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Method | select | TM_CCOEFF_NORMED | 방법 |
| Threshold | number | 0.8 | 임계 |

---

### 이미지 추출 (Image Extract)

Image Extract Node

좌표 [x1,y1,x2,y2]로 이미지 영역을 추출합니다.

입력: image, coords | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Padding | number | 0 | 패딩 |

---

## 분할 (Segmentation)

### 플러드 필 (Flood Fill)

cv2.floodFill(image, mask, seedPoint, newVal, ...)

시드 점에서 연결 영역을 채웁니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| SeedX | number | 0 | X |
| SeedY | number | 0 | Y |
| R | number | 255 | |
| G | number | 0 | |
| B | number | 0 | |
| LoDiff | number | 20 | 하한 |
| UpDiff | number | 20 | 상한 |

---

### 그랩컷 (GrabCut)

cv2.grabCut(img, mask, rect, ...)

GrabCut으로 전경/배경을 분리합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| X | number | 10 | X |
| Y | number | 10 | Y |
| W | number | 200 | 너비 |
| H | number | 200 | 높이 |
| Iter | number | 5 | 반복 |

---

### 워터쉐드 (Watershed)

cv2.watershed(image, markers)

워터쉐드로 이미지를 영역 분할합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Marker K | number | 10 | 커널 |

---

## 값 (Value)

### 정수 (Integer)

Integer Value

정수 상수를 출력합니다.

입력: 없음 | 출력: value

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Value | number | 0 | 값 |
| Min | number | 0 | 최소 |
| Max | number | 255 | 최대 |

---

### 실수 (Float)

Float Value

실수 상수를 출력합니다.

입력: 없음 | 출력: value

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Value | number | 0.0 | 값 |

---

### 불리언 (Boolean)

Boolean Value

True/False 값을 출력합니다.

입력: 없음 | 출력: value

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Value | checkbox | false | 값 |

---

## 점 좌표 (Point)

Point Value

(x, y) 좌표를 출력합니다.

입력: 없음 | 출력: point

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| X | number | 0 | X |
| Y | number | 0 | Y |

---

## 스칼라 (Scalar)

Scalar Value

4원소 스칼라 값을 출력합니다.

입력: 없음 | 출력: scalar

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| V0 | number | 0 | |
| V1 | number | 0 | |
| V2 | number | 0 | |
| V3 | number | 0 | |

---

## 수학 연산 (Math Operation)

Math Operation

기본 수학 연산을 수행합니다.

입력: A, B | 출력: result

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Op | select | add | 연산 |

---

## 좌표 입력 (Val Coords)

Val Coords (x1,y1,x2,y2)

수동 좌표 입력. Image Extract의 coords 포트에 연결합니다. single/multi 모드.

입력: 없음 | 출력: coords

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | single | 모드 |
| X1 | number | 0 | |
| Y1 | number | 0 | |
| X2 | number | 100 | |
| Y2 | number | 100 | |

---

### 리스트 인덱싱 (List Index/Slice)

List Index/Slice

리스트에서 인덱싱 또는 슬라이싱합니다.

입력: list | 출력: result

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Mode | select | index | 모드 |
| Index | number | 0 | 인덱스 |
| Start | number | 0 | 시작 |
| Stop | number | -1 | 끝 |
| Step | number | 1 | 스텝 |

---

## 제어 흐름 (Control)

### 조건 분기 (If)

If (condition) -> True / False

조건에 따라 True/False 경로로 분기합니다.

입력: image | 출력: true, false

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Condition | select | not_empty | 조건 |
| Value | number | 100 | 비교값 |
| Custom | text | img.shape[0]>100 | 식 |

---

### For 반복 (For Loop)

for i in range(N): operation

연산을 N번 반복 적용합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| N | number | 3 | 반복 |
| Op | select | gaussian_blur | 연산 |
| K | number | 3 | 커널 |
| Code | text | | 코드 |

---

### While 반복 (While Loop)

while (cond): operation

조건이 참인 동안 반복합니다.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Cond | select | mean_gt | 조건 |
| Val | number | 128 | 임계 |
| Op | select | gaussian_blur | 연산 |
| K | number | 3 | |
| Max | number | 50 | 제한 |
| CCond | text | | 조건식 |
| CCode | text | | 코드 |

---

### 스위치-케이스 (Switch-Case)

switch -> case 0/1/2

조건에 따라 3개 출력 중 하나로 분기합니다.

입력: image | 출력: case 0, case 1, case 2

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Switch | select | channels | 기준 |
| Custom | text | | 식 |

---

## 스크립트 (Script)

### 파이썬 스크립트 (Python Script)

Custom Python Script

사용자 정의 Python 코드를 실행합니다. img_input/img_output 사용.

입력: image | 출력: image

| 속성 | 타입 | 기본값 | 설명 |
|---|---|---|---|
| Code | textarea | # img_input → img_output | 코드 |

---

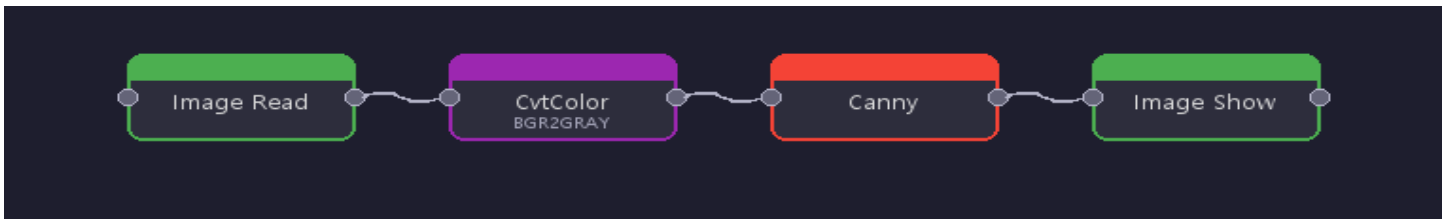
# Part 4: 튜토리얼

30개의 실습 튜토리얼을 통해 SamOpenCVWeb의 사용법을 단계별로 배웁니다.

## Tutorial 1: 첫 이미지 처리 — 그레이스케일 & 에지 검출

**학습 목표: CvtColor와 Canny를 사용하여 컬러 이미지에서 에지를 검출합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. 팔레트 → I/O에서 Image Read를 캔버스에 드래그합니다.
2. Image Read를 더블클릭하여 이미지를 업로드합니다.
3. 팔레트 → Color에서 CvtColor를 추가하고 Color Code를 COLOR_BGR2GRAY로 설정합니다.
4. Image Read의 출력 포트를 CvtColor의 입력 포트로 드래그하여 연결합니다.
5. 팔레트 → Edge에서 Canny를 추가합니다. Threshold1=100, Threshold2=200.
6. CvtColor → Canny를 연결합니다.
7. Image Show를 추가하고 Canny → Image Show를 연결합니다.
8. Execute 버튼을 클릭하면 에지 검출 결과가 표시됩니다.

[Tip] 더 해보기: Threshold 값을 변경하여 에지 감도를 조절해 보세요.

---

## Tutorial 2: 블러 & 임계값 — 이진 마스크 생성

**학습 목표: 가우시안 블러로 노이즈를 제거한 후 임계값으로 이진 이미지를 만듭니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read → CvtColor(BGR2GRAY)를 구성합니다.
2. Gaussian Blur를 추가합니다. Kernel Size=5.
3. CvtColor → Gaussian Blur를 연결합니다.
4. Threshold를 추가합니다. Thresh=127, MaxVal=255, Type=THRESH_BINARY.
5. Gaussian Blur → Threshold → Image Show를 연결합니다.
6. Execute하면 흰/검 이진 마스크가 표시됩니다.

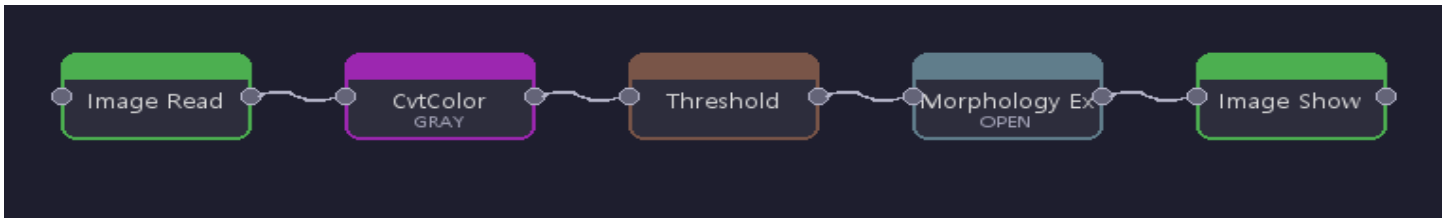
[Tip] 더 해보기: THRESH_OTSU를 사용하면 자동으로 최적 임계값을 찾습니다.

---

# Tutorial 3: 형태학 연산 — 이진 이미지 정리

**학습 목표: Opening 연산으로 이진 이미지의 노이즈를 제거합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Tutorial 2의 파이프라인(Image Read → CvtColor → Threshold)을 구성합니다.
2. Morphology Ex를 추가합니다. Operation=MORPH_OPEN, Kernel=5, Shape=MORPH_RECT.
3. Threshold → Morphology Ex → Image Show를 연결합니다.
4. Execute하면 작은 노이즈가 제거된 깨끗한 이진 이미지가 표시됩니다.

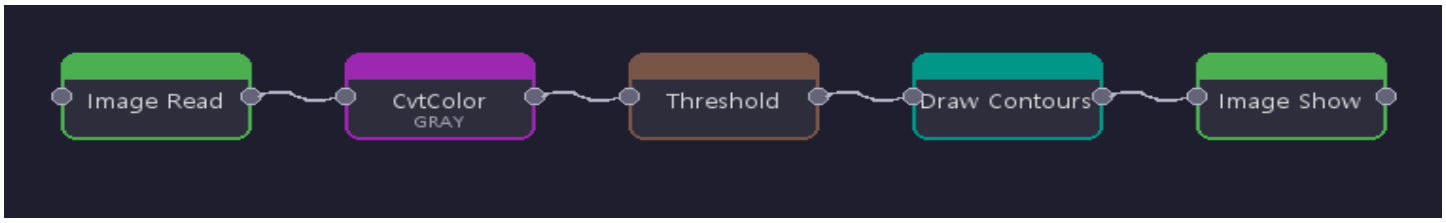
[Tip] 더 해보기: MORPH_CLOSE로 변경하면 작은 구멍이 메워집니다.

---

# Tutorial 4: 컨투어 검출 — 물체 외곽선 찾기

**학습 목표: 이진 이미지에서 컨투어를 찾아 원본 이미지 위에 그립니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read → CvtColor → Threshold로 이진 이미지를 준비합니다.
2. Draw Contours를 추가합니다. Mode=RETR_EXTERNAL, Thickness=2.
3. Threshold → Draw Contours → Image Show를 연결합니다.
4. Execute하면 윤곽선이 녹색으로 그려집니다.

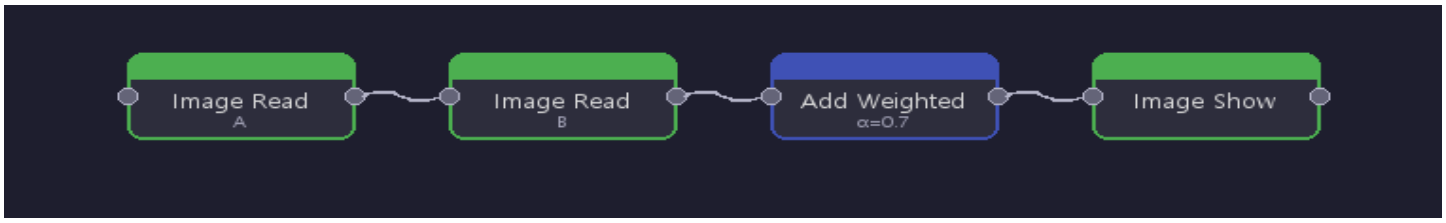
[Tip] 더 해보기: Bounding Rect나 Min Enclosing Circle로 다른 표현을 시도해 보세요.

---

# Tutorial 5: 이미지 합성 — 두 이미지 블렌딩

**학습 목표: Add Weighted로 두 이미지를 투명도를 조절하여 합성합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read 2개를 추가하고 각각 다른 이미지를 업로드합니다.
2. 두 이미지의 크기가 같아야 합니다 (Resize로 맞출 수 있습니다).
3. Add Weighted를 추가합니다. Alpha=0.7, Beta=0.3, Gamma=0.
4. 첫 번째 Image Read → image 포트, 두 번째 → image2 포트에 연결합니다.
5. Add Weighted → Image Show를 연결하고 Execute합니다.

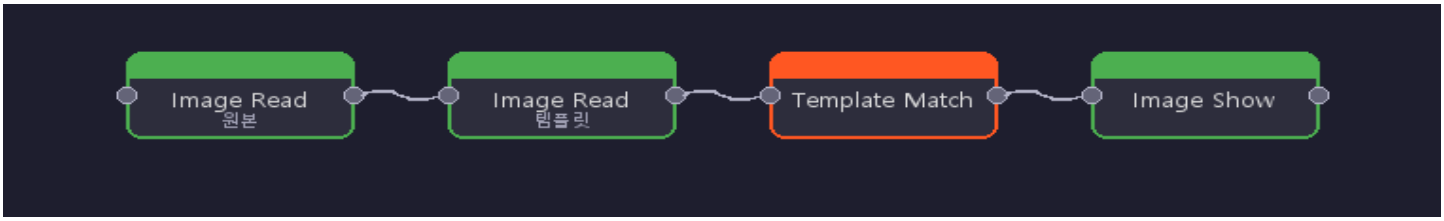
[Tip] 더 해보기: Alpha+Beta=1이면 전체 밝기가 유지됩니다.

---

# Tutorial 6: 템플릿 매칭 & 이미지 추출

**학습 목표: Template Match로 패턴을 찾고 Image Extract로 해당 영역을 추출합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read 2개: 원본 이미지와 찾을 템플릿 이미지.
2. Template Match를 추가합니다. Method=TM_CCOEFF_NORMED, Threshold=0.8.
3. 원본 → image 포트, 템플릿 → template 포트에 연결합니다.
4. Image Extract를 추가합니다.
5. 원본 → Image Extract의 image, Template Match의 matches → coords에 연결합니다.
6. Image Extract → Image Show를 연결하고 Execute합니다.

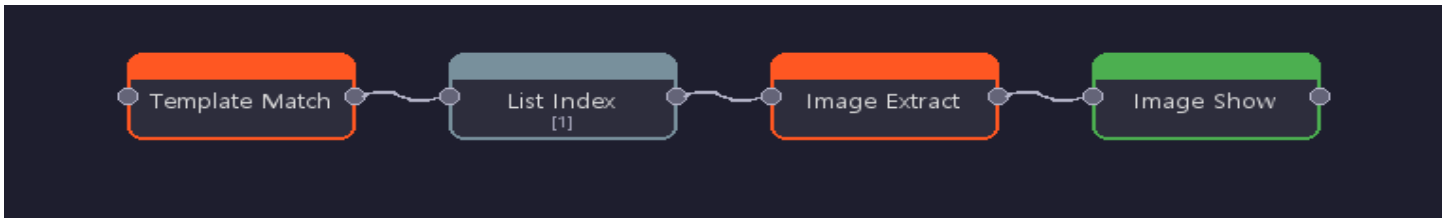
[Tip] 더 해보기: Threshold를 낮추면 더 많은 매칭이 검출됩니다.

---

# Tutorial 7: 리스트 인덱싱과 템플릿 매칭 결합

**학습 목표: matches 리스트에서 특정 인덱스의 좌표를 선택합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Tutorial 6의 Template Match 파이프라인을 구성합니다.
2. List Index/Slice 노드를 추가합니다. Mode=index, Index=1.
3. matches → List Index의 list 입력에 연결합니다.
4. result → Image Extract의 coords에 연결합니다.
5. Execute하면 두 번째 매칭 영역이 추출됩니다.

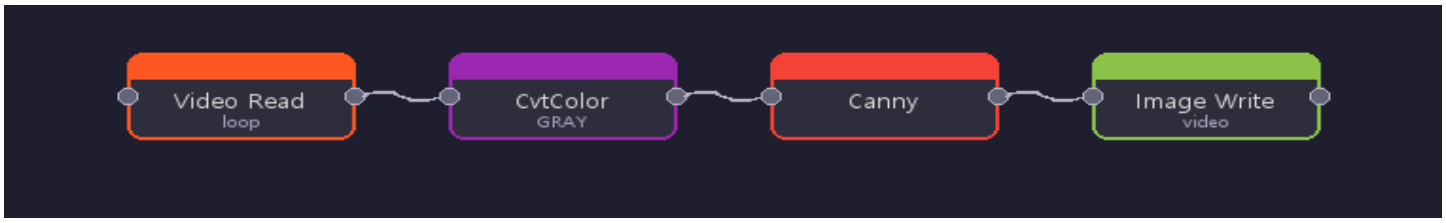
[Tip] 더 해보기: slice 모드로 여러 매칭을 한번에 처리할 수 있습니다.

---

# Tutorial 8: 비디오 프레임별 처리

**학습 목표: Video Read 루프 모드로 모든 프레임에 에지 검출을 적용합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Video Read를 추가하고 비디오를 업로드합니다.
2. Mode=loop, Start Frame=0, End Frame=-1, Step=1로 설정합니다.
3. CvtColor(BGR2GRAY) → Canny(100,200)를 연결합니다.
4. Image Write를 추가합니다. Video Output 체크, 경로=output.mp4.
5. Canny → Image Write를 연결하고 Execute합니다.

[Tip] 더 해보기: Step을 변경하면 프레임을 건너뛸 수 있습니다.

---

# Tutorial 9: 얼굴 검출 (Haar Cascade)

**학습 목표: Haar Cascade 분류기로 이미지에서 얼굴을 검출합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 사람 얼굴이 있는 사진을 업로드합니다.
2. Haar Cascade를 추가합니다. Type=face, Scale=1.1, Neighbors=5.
3. Image Read → Haar Cascade → Image Show를 연결합니다.
4. Execute하면 검출된 얼굴에 사각형이 그려집니다.

[Tip] 더 해보기: Type을 eye나 smile로 변경해 보세요.

---

# Tutorial 10: 커스텀 Python 스크립트

**학습 목표: Python Script 노드로 직접 코드를 작성하여 실행합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 이미지를 업로드합니다.
2. Python Script를 추가합니다.
3. 코드: img_output = cv2.Canny(img_input, 50, 150)
4. Image Read → Python Script → Image Show를 연결합니다.
5. Execute하면 커스텀 코드 결과가 표시됩니다.

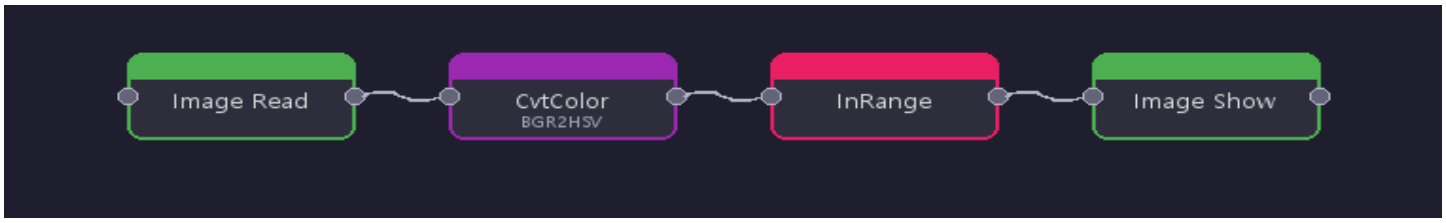
[Tip] 더 해보기: cv2, np를 자유롭게 사용할 수 있습니다.

---

# Tutorial 11: HSV 색상 필터링 (InRange)

**학습 목표: HSV 공간에서 특정 색상 범위만 추출합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

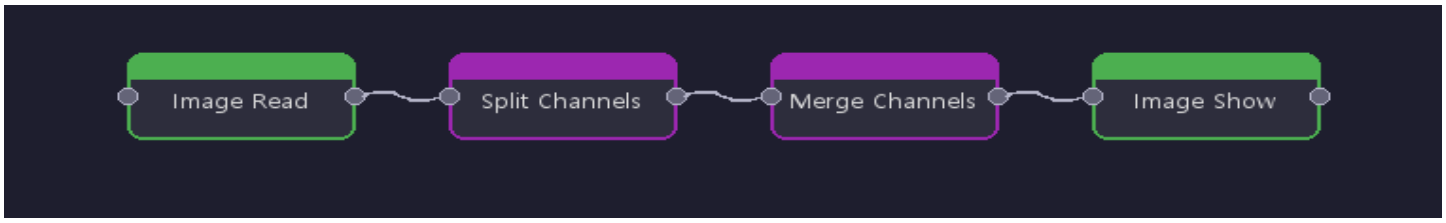
1. Image Read에 컬러 이미지를 업로드합니다.
2. CvtColor를 추가하고 COLOR_BGR2HSV로 설정합니다.
3. InRange를 추가합니다. 예: 빨간색 → Lower=(0,100,100), Upper=(10,255,255).
4. Image Read → CvtColor → InRange → Image Show를 연결합니다.
5. Execute하면 해당 색상만 흰색인 마스크가 표시됩니다.

[Tip] 더 해보기: 결과 마스크를 Bitwise AND에 사용하면 원본에서 해당 색상만 추출합니다.

# Tutorial 12: 채널 분리 & 병합

**학습 목표: Split/Merge Channels로 BGR 채널을 분리하고 재조합합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 컬러 이미지를 업로드합니다.
2. Split Channels를 추가하고 연결합니다. ch0=B, ch1=G, ch2=R.
3. Merge Channels를 추가합니다.
4. Split의 ch0→Merge의 ch2, ch2→ch0으로 교차 연결합니다 (R↔B 교환).
5. Merge → Image Show를 연결하고 Execute합니다.

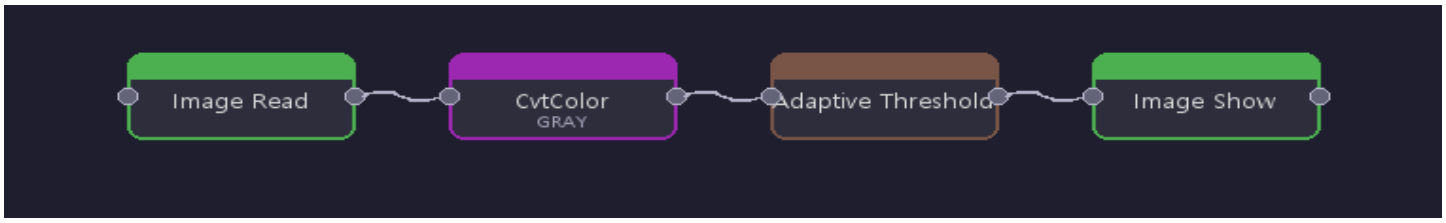
[Tip] 더 해보기: 한 채널만 Image Show에 연결하면 해당 채널의 그레이스케일 이미지를 볼 수 있습니다.

---

# Tutorial 13: 적응형 임계값 처리

**학습 목표: Adaptive Threshold로 조명이 불균일한 이미지를 이진화합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 문서/텍스트 이미지를 업로드합니다.
2. CvtColor(BGR2GRAY) → Adaptive Threshold를 연결합니다.
3. Method=GAUSSIAN_C, Block Size=11, C=2로 설정합니다.
4. Execute하면 조명 차이에 관계없이 깨끗하게 이진화됩니다.

[Tip] 더 해보기: Block Size를 키우면 더 넓은 영역을 참조합니다.

---

# Tutorial 14: 이미지 샤프닝

**학습 목표: Sharpen 노드로 흐릿한 이미지를 선명하게 합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 흐릿한 이미지를 업로드합니다.
2. Sharpen을 추가합니다. Strength=1.5.
3. Image Read → Sharpen → Image Show를 연결합니다.
4. Execute하면 선명해진 이미지가 표시됩니다.

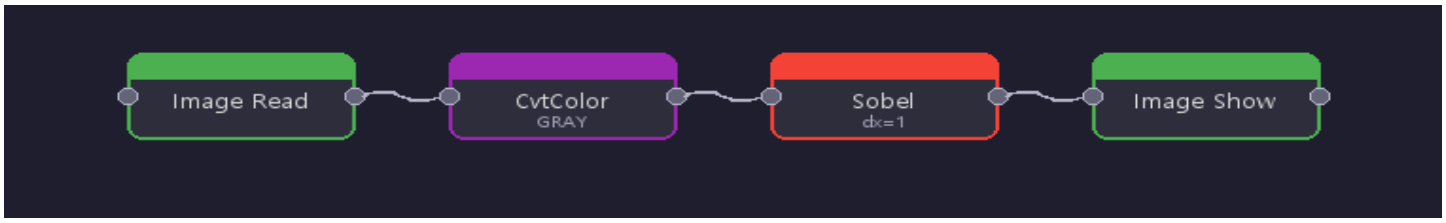
[Tip] 더 해보기: Gaussian Blur → Sharpen 조합으로 노이즈 제거+선명화를 할 수 있습니다.

---

# Tutorial 15: 소벨 에지 그래디언트

**학습 목표: Sobel로 X, Y 방향 에지 그래디언트를 시각화합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read → CvtColor(BGR2GRAY)를 구성합니다.
2. Sobel 2개 추가: 하나는 dx=1,dy=0(수직 에지), 다른 하나는 dx=0,dy=1(수평 에지).
3. CvtColor에서 두 Sobel로 각각 연결합니다.
4. Add Weighted(α=0.5,β=0.5)로 합성합니다.
5. Execute하면 합쳐진 에지 이미지가 표시됩니다.

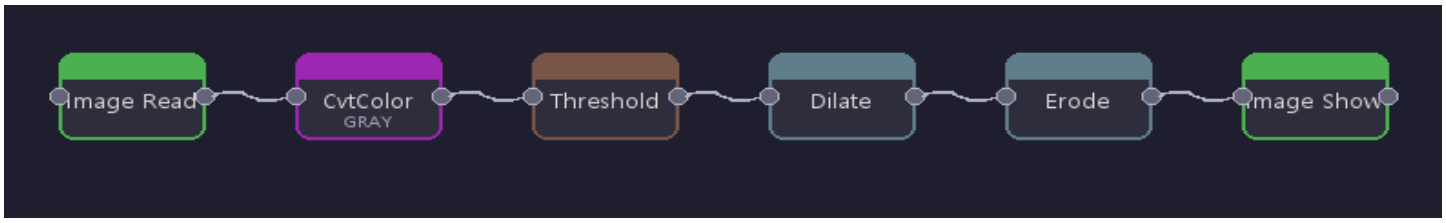
[Tip] 더 해보기: X 방향은 수직 에지, Y 방향은 수평 에지를 검출합니다.

---

# Tutorial 16: 팽창 & 침식

**학습 목표: Dilate와 Erode로 이진 이미지 형태를 변형합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read → CvtColor → Threshold로 이진 이미지를 만듭니다.
2. Dilate를 추가합니다. Kernel=5, Iterations=1.
3. Erode를 추가합니다. 같은 설정.
4. Threshold → Dilate → Erode → Image Show를 연결합니다.
5. Execute하면 팽창 후 침식 결과가 표시됩니다.

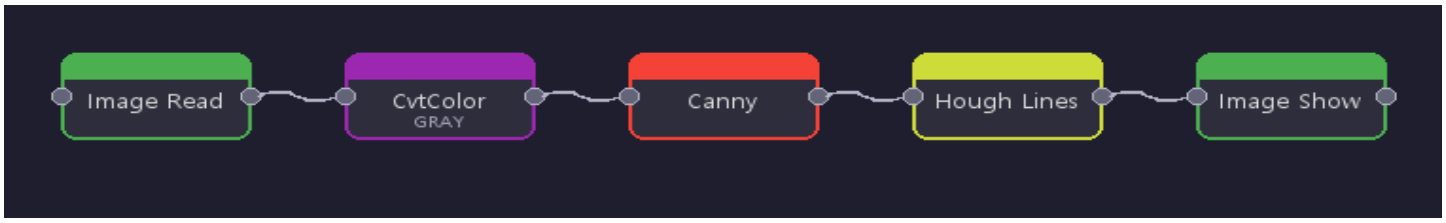
[Tip] 더 해보기: 순서를 Erode → Dilate로 바꾸면 닫기 연산이 됩니다.

---

# Tutorial 17: 허프 직선 검출

**학습 목표: Canny 에지에서 Hough Lines로 직선을 검출합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 직선이 많은 이미지를 업로드합니다.
2. CvtColor(GRAY) → Canny를 연결합니다.
3. Hough Lines를 추가합니다. Rho=1, Theta=180, Threshold=100.
4. Canny → Hough Lines → Image Show를 연결합니다.
5. Execute하면 검출 직선이 빨간색으로 표시됩니다.

[Tip] 더 해보기: Threshold를 낮추면 더 많은 직선이 검출됩니다.

---

# Tutorial 18: 허프 원 검출

**학습 목표: Hough Circles로 원형 물체를 검출합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 원형 물체가 있는 이미지를 업로드합니다.
2. Hough Circles를 추가합니다. dp=1.2, MinDist=50, P1=100, P2=30.
3. Image Read → Hough Circles → Image Show를 연결합니다.
4. Execute하면 원이 녹색으로 표시됩니다.

[Tip] 더 해보기: Min/Max Radius로 검출할 원 크기를 제한할 수 있습니다.

---

# Tutorial 19: 리사이즈 & 스케일

**학습 목표: Resize 노드로 이미지 크기를 변경합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 이미지를 업로드합니다.
2. Resize를 추가합니다. Scale X=0.5, Scale Y=0.5.
3. Image Read → Resize → Image Show를 연결합니다.
4. Execute하면 절반 크기 이미지가 표시됩니다.

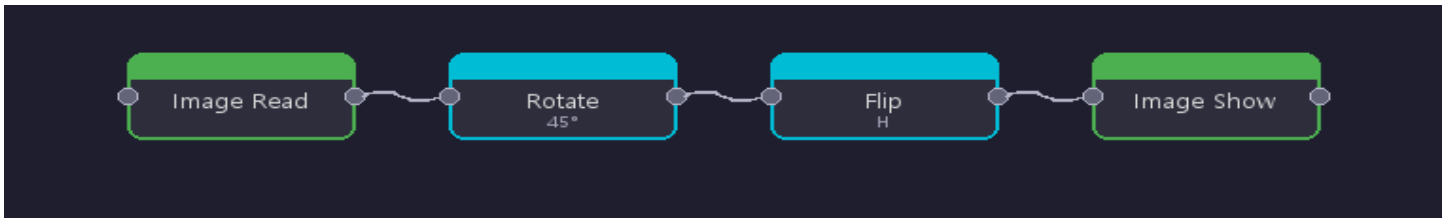
[Tip] 더 해보기: INTER_CUBIC은 확대에, INTER_AREA는 축소에 적합합니다.

---

# Tutorial 20: 회전 & 뒤집기

**학습 목표: Rotate와 Flip으로 이미지 방향을 변경합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 이미지를 업로드합니다.
2. Rotate를 추가합니다. Angle=45.
3. Flip을 추가합니다. Mode=Horizontal (1).
4. Image Read → Rotate → Flip → Image Show를 연결합니다.
5. Execute합니다.

[Tip] 더 해보기: Vertical (0)은 상하 반전, Both (-1)은 180도 회전과 같습니다.

---

# Tutorial 21: 자르기 — 관심 영역 (ROI)

**학습 목표: Crop으로 이미지의 특정 영역만 잘라냅니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 이미지를 업로드합니다.
2. Crop을 추가합니다. X=50, Y=50, Width=200, Height=150.
3. Image Read → Crop → Image Show를 연결합니다.
4. Execute하면 지정 영역만 표시됩니다.

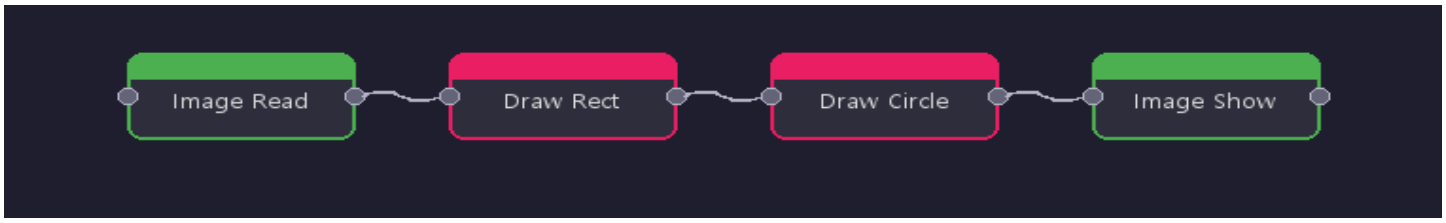
[Tip] 더 해보기: Template Match + Image Extract와 조합하면 자동 추출이 가능합니다.

---

# Tutorial 22: 도형 그리기

**학습 목표: Draw 노드들로 이미지 위에 도형을 그립니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 이미지를 업로드합니다.
2. Draw Rectangle을 추가합니다. X=10, Y=10, W=100, H=80, Color=(0,255,0).
3. Draw Circle을 추가합니다. Center(200,150), R=40, Color=(0,0,255).
4. Image Read → Rectangle → Circle → Image Show를 연결합니다.
5. Execute하면 도형이 그려집니다.

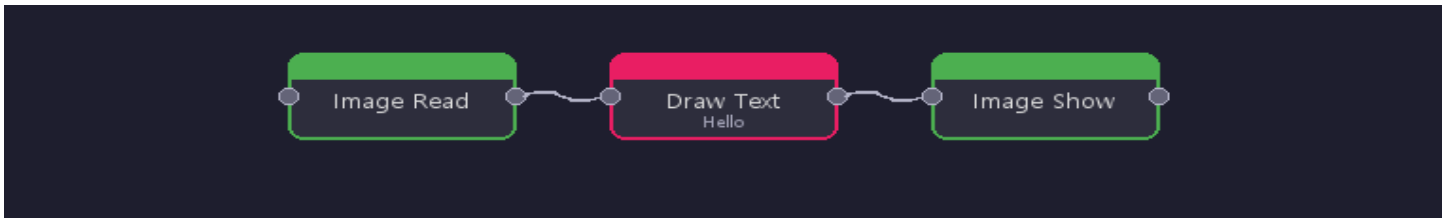
[Tip] 더 해보기: Thickness=-1이면 채워진 도형이 됩니다.

---

# Tutorial 23: 텍스트 오버레이

**학습 목표: Draw Text로 이미지에 텍스트를 표시합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 이미지를 업로드합니다.
2. Draw Text를 추가합니다. Text="Hello OpenCV", (50,50), Scale=1.5, Color=(255,255,255).
3. Image Read → Draw Text → Image Show를 연결합니다.
4. Execute합니다.

[Tip] 더 해보기: 여러 Draw Text를 연결하면 여러 줄을 추가할 수 있습니다.

---

# Tutorial 24: 비트 마스크 연산

**학습 목표: InRange 마스크를 Bitwise AND로 적용하여 특정 색상만 추출합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read → CvtColor(BGR2HSV) → InRange를 구성합니다.
2. InRange에서 원하는 색상의 HSV 범위를 설정합니다.
3. Bitwise AND를 추가합니다.
4. 원본 Image Read → image, InRange 결과 → image2에 연결합니다.
5. Execute하면 해당 색상만 원본 색상으로 표시됩니다.

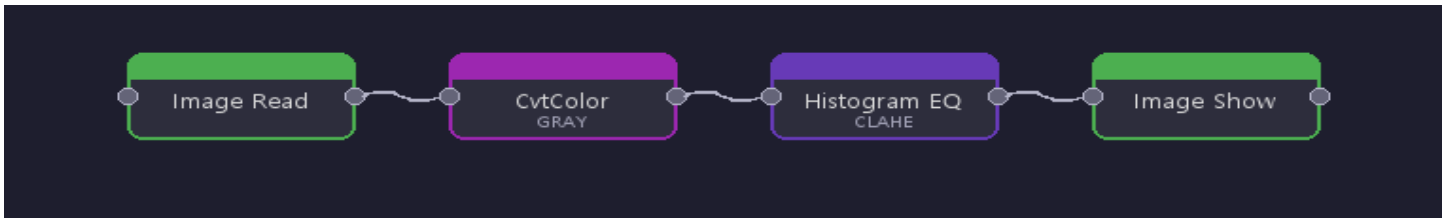
[Tip] 더 해보기: Bitwise NOT으로 마스크를 반전하면 반대 영역을 추출합니다.

---

# Tutorial 25: 히스토그램 평활화 (CLAHE)

**학습 목표: CLAHE로 대비가 낮은 이미지를 개선합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 대비가 낮은 이미지를 업로드합니다.
2. CvtColor(BGR2GRAY) → Histogram EQ를 연결합니다.
3. Use CLAHE 체크, Clip Limit=2.0, Tile Size=8.
4. Execute하면 대비가 개선된 이미지가 표시됩니다.

[Tip] 더 해보기: Clip Limit를 높이면 대비가 더 강해집니다.

---

# Tutorial 26: 이미지 차이 — 변화 감지 (AbsDiff)

**학습 목표: AbsDiff로 두 이미지의 차이를 계산합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read 2개에 같은 장면의 전후 이미지를 업로드합니다.
2. AbsDiff를 추가합니다. A → image, B → image2.
3. Threshold를 추가합니다. 임계값=30.
4. AbsDiff → Threshold → Image Show를 연결합니다.
5. Execute하면 변화 영역이 흰색으로 표시됩니다.

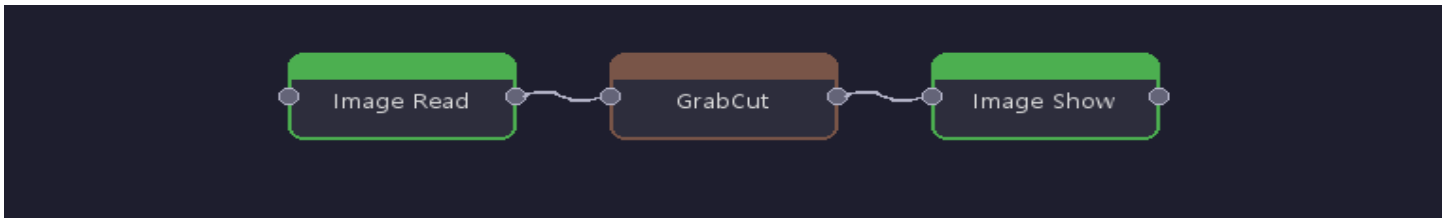
[Tip] 더 해보기: CvtColor(GRAY) 후 AbsDiff를 적용하면 더 정확합니다.

---

# Tutorial 27: GrabCut 배경 제거

**학습 목표: GrabCut으로 전경 물체를 분리합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 물체가 중앙에 있는 이미지를 업로드합니다.
2. GrabCut을 추가합니다. 물체를 포함하는 사각영역 설정.
3. Iterations=5로 설정합니다.
4. Image Read → GrabCut → Image Show를 연결합니다.
5. Execute하면 배경이 제거됩니다.

[Tip] 더 해보기: 사각영역이 물체를 잘 감싸야 정확한 결과가 나옵니다.

---

# Tutorial 28: Watershed 이미지 분할

**학습 목표: Watershed로 겹치는 물체를 분리합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 여러 물체가 겹친 이미지를 업로드합니다.
2. Watershed를 추가합니다. Marker Kernel=10.
3. Image Read → Watershed → Image Show를 연결합니다.
4. Execute하면 경계가 빨간 선으로 표시됩니다.

[Tip] 더 해보기: Marker Kernel이 클수록 분할 영역이 적어집니다.

---

# Tutorial 29: 투시 변환 (Perspective Transform)

**학습 목표: Warp Perspective로 비스듬한 이미지를 정면으로 보정합니다.**

**파이프라인 다이어그램:**


**단계별 설명:**

1. Image Read에 비스듬히 촬영된 문서 이미지를 업로드합니다.
2. Warp Perspective를 추가합니다.
3. Src 4꼭짓점을 문서 모서리로, Dst를 직사각형으로 설정합니다.
4. Image Read → Warp Perspective → Image Show를 연결합니다.
5. Execute하면 정면으로 보정됩니다.

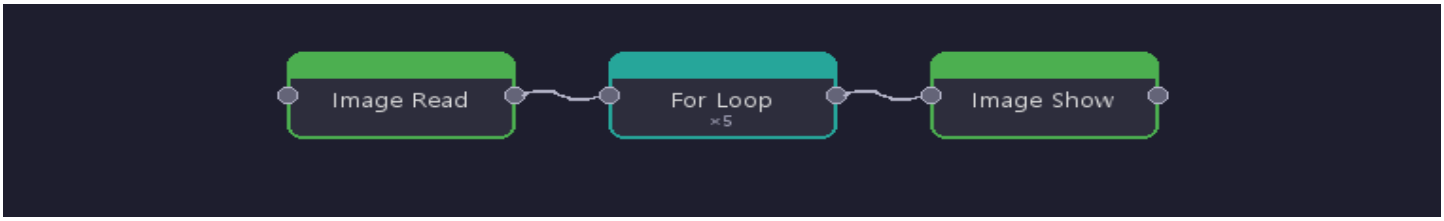
[Tip] 더 해보기: 명함, 간판 등의 기울어진 촬영물 보정에 활용하세요.

---

# Tutorial 30: For Loop — 반복 블러 처리

**학습 목표: For Loop으로 블러를 여러 번 반복 적용합니다.**

**파이프라인 다이어그램:**




**단계별 설명:**

1. Image Read에 이미지를 업로드합니다.
2. For Loop을 추가합니다. N=5, Operation=gaussian_blur, K=3.
3. Image Read → For Loop → Image Show를 연결합니다.
4. Execute하면 블러가 5번 적용된 결과가 표시됩니다.

[Tip] 더 해보기: custom 모드에서 직접 코드를 작성하면 복잡한 반복도 가능합니다.

---

# Part 5: 단축키 & 팁

## 키보드 단축키

| | |
|---|---|
| **Ctrl+Enter** | 파이프라인 실행 |
| **Ctrl+Z** | 실행 취소 |
| **Ctrl+C** | 선택 노드 복사 |
| **Ctrl+X** | 선택 노드 잘라내기 |
| **Ctrl+V** | 붙여넣기 |
| **Ctrl+A** | 전체 선택 |
| **Delete / Backspace** | 선택 삭제 |
| **Ctrl+S** | 파이프라인 저장 |
| **Esc** | 선택 해제 |

## 마우스 조작

| | |
|---|---|
| **왼쪽 클릭** | 노드 선택 |
| **더블 클릭** | 파일 노드: 파일 선택 / 기타: 미리보기 실행 |
| **드래그 (노드)** | 노드 이동 |
| **드래그 (포트)** | 연결선 생성 |
| **드래그 (빈 영역)** | 캔버스 이동 |
| **Shift+드래그** | 다중 선택 영역 |
| **마우스 휠** | 확대/축소 |

## 유용한 팁

- 노드를 가까이 배치하면 자동으로 연결됩니다.
- 이미지 크기가 다른 경우 Resize 노드로 맞춰주세요.
- Python Script 노드로 커스텀 처리를 구현할 수 있습니다.
- Code 버튼으로 생성된 코드를 독립 스크립트로 활용하세요.
- Video Read의 loop 모드로 비디오 전체를 한번에 처리합니다.

- 컨투어 분석 시 이진화를 먼저 수행하면 결과가 좋습니다.
- HSV 색상 공간에서 InRange 필터링이 더 효과적입니다.

# Part 6: 응용 예제 프로젝트

실전에서 활용할 수 있는 34개의 복합 예제 프로젝트입니다. 각 예제는 JSON 저장 파일로 제공되며, Load 기능으로 바로 불러올 수 있습니다.

**[안내] examples/ 폴더의 JSON 파일을 Load하면 예제 파이프라인이 자동으로 구성됩니다.**

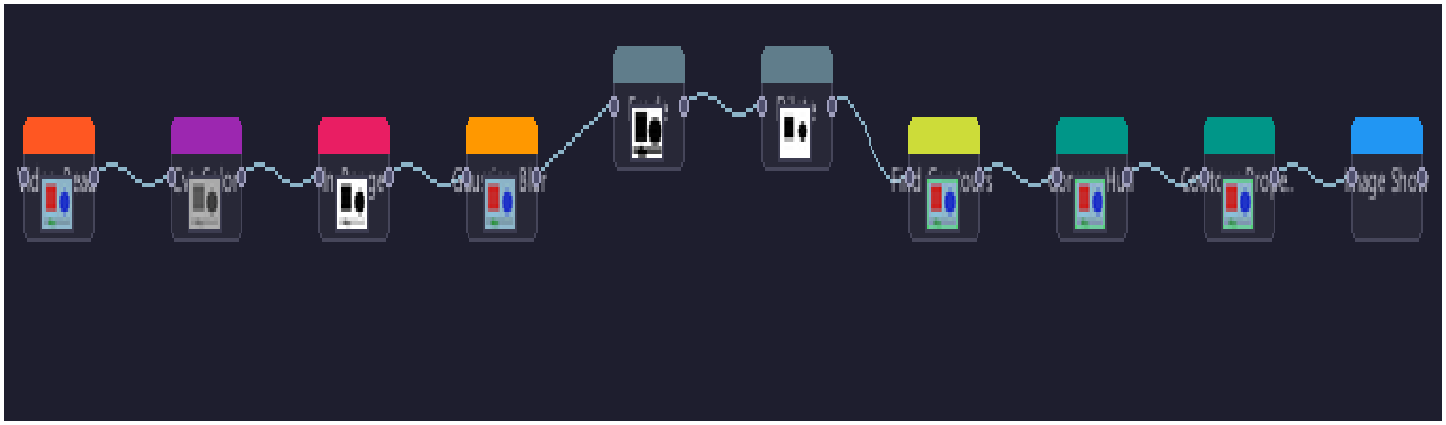
## 예제 1: 손가락 개수 세기 (Finger Counter)

난이도: ★★★ | 저장파일: 01_finger_counter.json

**학습 목표: 비디오에서 피부색을 감지하고 컨투어/볼록 껍질 분석으로 손가락 개수를 추정합니다.**

**핵심 개념: HSV 색상 공간 | 피부색 범위 필터링 | 침식/팽창 노이즈 제거 | 컨투어 분석 | 볼록 껍질(Convex Hull)**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Video Read 노드를 추가하고 손 영상 파일을 업로드합니다.
2. CvtColor (BGR->HSV)을 연결합니다. HSV 색상 공간은 피부색 감지에 효과적입니다.
3. InRange를 추가합니다. 피부색 HSV 범위: H=0~20, S=30~150, V=60~255.
4. Gaussian Blur (ksize=5)로 노이즈를 줄입니다.
5. Erode (ksize=3, iterations=2)로 잔여 노이즈를 제거합니다.
6. Dilate (ksize=3, iterations=3)로 손 영역을 확장합니다.
7. Find Contours (RETR_EXTERNAL)로 가장 큰 윤곽선(손)을 찾습니다.
8. Convex Hull로 볼록 껍질을 그립니다. 손가락 사이 오목한 부분이 손가락 수를 결정합니다.
9. Contour Properties를 추가하면 면적/중심점이 표시됩니다.
10. Execute하면 손 위에 볼록 껍질이 그려집니다.

[Tip] 조명 조건에 따라 InRange의 HSV 범위를 조절하세요. 어두운 환경에서는 V 하한을 낮추세요.

---

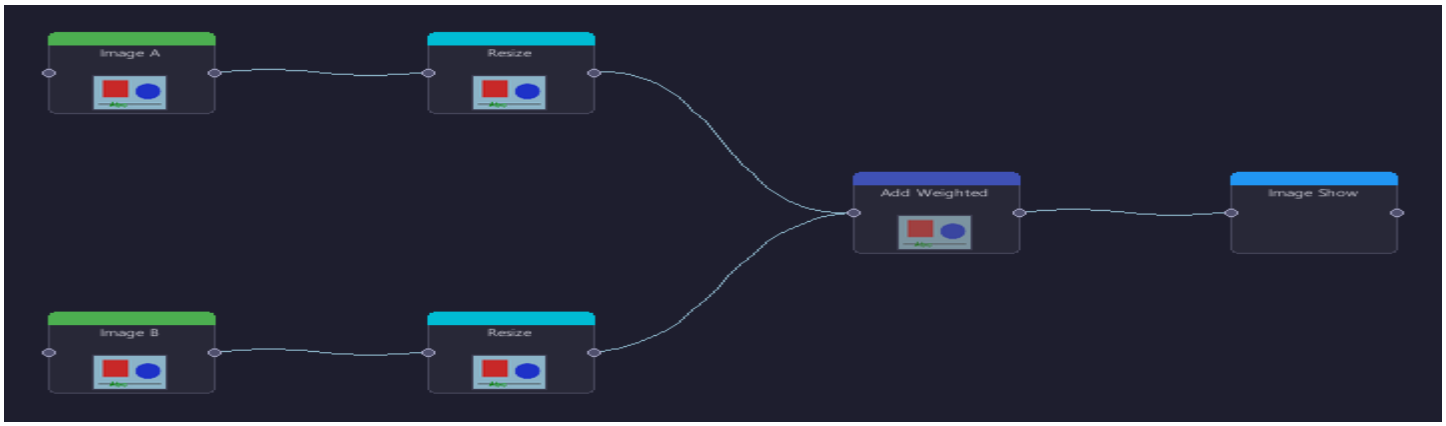
## 예제 2: 이미지 페이드 전환 (Fade In/Out)

난이도: ★★ | 저장파일: 02_fade_transition.json

**학습 목표: 두 이미지를 Alpha 블렌딩으로 부드럽게 전환합니다.**

**핵심 개념: Alpha 블렌딩 | 가중치 합산 | 이미지 크기 매칭**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read 두 개를 추가합니다. 각각 Image A, Image B를 업로드합니다.
2. Resize 두 개를 추가합니다. 두 이미지를 같은 크기(640x480)로 통일합니다.
3. Add Weighted를 추가합니다. Alpha=0.5, Beta=0.5, Gamma=0.
4. Image A의 Resize -> Add Weighted의 image 포트에 연결합니다.
5. Image B의 Resize -> Add Weighted의 image2 포트에 연결합니다.
6. Add Weighted -> Image Show를 연결하고 Execute합니다.
7. Alpha를 0.0->1.0으로 변경하면 Fade-in/out 효과를 확인할 수 있습니다.

[Tip] Alpha+Beta=1.0이면 원래 밝기가 유지됩니다. Alpha를 0.0에서 1.0까지 변경하며 전환 효과를 확인하세요.

---

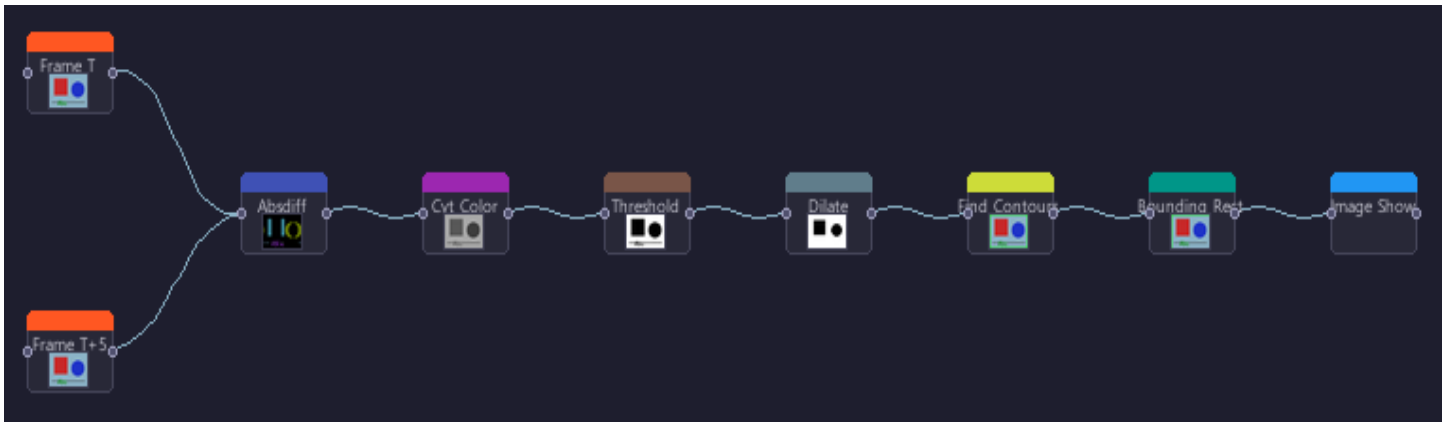
## 예제 3: 움직임 감지 (Motion Detection)

난이도: ★★★ | 저장파일: 03_motion_detection.json

**학습 목표: 비디오의 연속 프레임 차이를 이용하여 움직이는 물체를 감지합니다.**

**핵심 개념: 프레임 차분 | 절대값 차이(AbsDiff) | 임계값 처리 | 바운딩 박스**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Video Read 두 개를 추가합니다. 같은 비디오를 로드하되 Frame Index를 0과 5로 설정합니다.
2. AbsDiff 노드를 추가합니다. 두 프레임의 차이를 계산합니다.
3. Video(0) -> AbsDiff의 image, Video(5) -> AbsDiff의 image2에 연결합니다.
4. CvtColor(BGR2GRAY)로 그레이스케일 변환합니다.
5. Threshold (thresh=30, BINARY)로 유의미한 변화만 추출합니다.
6. Dilate (ksize=5, iterations=3)로 감지 영역을 확장합니다.
7. Find Contours -> Bounding Rect를 연결합니다.
8. Execute하면 움직임이 있는 영역에 녹색 사각형이 표시됩니다.

[Tip] Frame Index 차이를 크게 하면 큰 움직임, 작게 하면 미세한 움직임을 감지합니다.

---

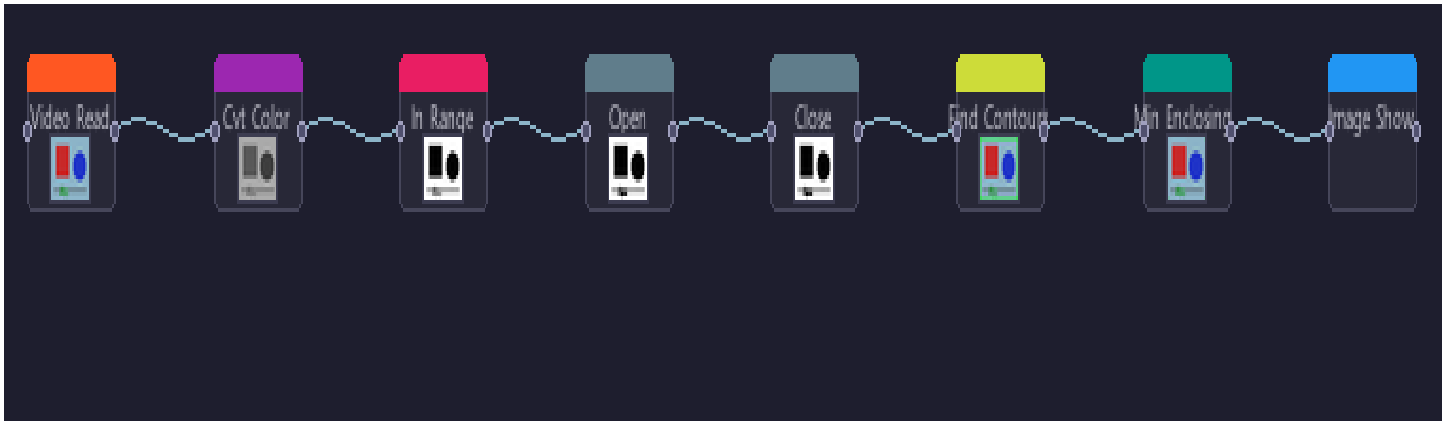
## 예제 4: 색상 객체 추적 (Color Tracking)

난이도: ★★★ | 저장파일: 04_color_tracking.json

**학습 목표: HSV 색상 범위로 특정 색상의 물체를 실시간 추적합니다.**

**핵심 개념: HSV 필터링 | 모폴로지 열기/닫기 | 최소 외접원**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Video Read를 추가하고 파란색 물체가 있는 영상을 업로드합니다. Mode=loop.
2. CvtColor (BGR->HSV)을 연결합니다.
3. InRange를 추가합니다. 파란색 HSV: H=100~130, S=50~255, V=50~255.
4. Morphology (MORPH_OPEN, ksize=5, Ellipse)로 작은 노이즈를 제거합니다.
5. Morphology (MORPH_CLOSE, ksize=5, Ellipse)로 내부 빈 공간을 채웁니다.
6. Find Contours (RETR_EXTERNAL) -> Min Enclosing Circle을 연결합니다.
7. Image Show를 연결하고 Execute합니다.
8. 파란색 물체 주위에 원이 그려집니다.

[Tip] 빨간색: H=0~10/170~180, 녹색: H=35~85. 환경 조명에 따라 S,V 범위 조절이 필요합니다.

---

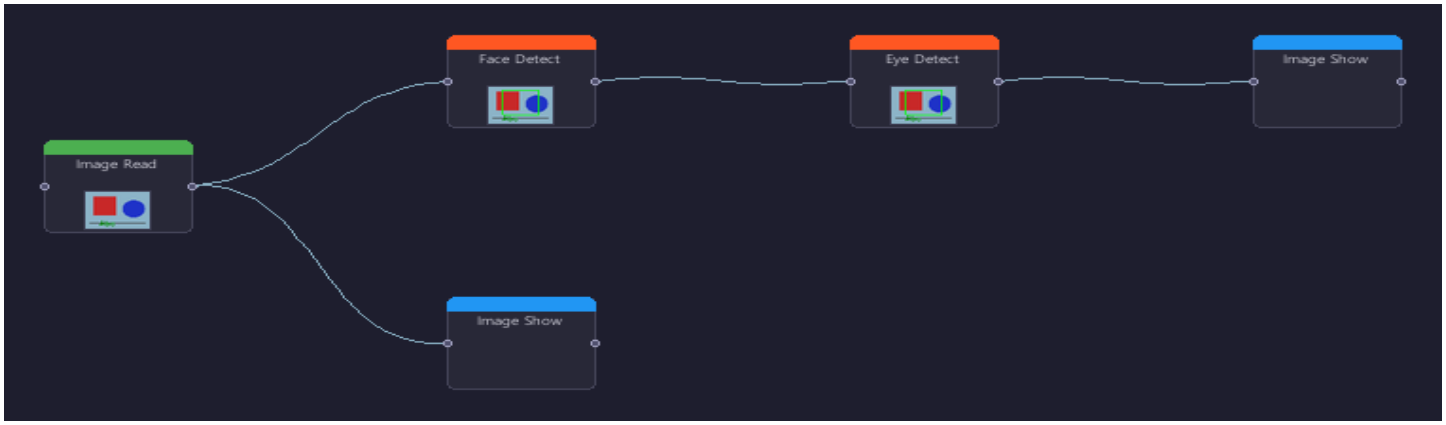
## 예제 5: 얼굴 + 눈 검출 (Face & Eye Detection)

난이도: ★★ | 저장파일: 05_face_detection.json

**학습 목표: Haar Cascade 분류기를 이용하여 얼굴과 눈을 동시에 검출합니다.**

**핵심 개념: Haar Cascade | 캐스케이드 분류기 | 얼굴 검출 | 눈 검출**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 사람 얼굴이 포함된 이미지를 업로드합니다.
2. Haar Cascade (Face)를 추가합니다. Type=frontalface, Scale=1.3, Neighbors=5.
3. Image Read -> Haar Cascade (Face)를 연결합니다.
4. Haar Cascade (Eye)를 추가합니다. Type=eye, Scale=1.1, Neighbors=5.
5. Haar Cascade (Face) 출력 -> Haar Cascade (Eye) 입력에 연결합니다.
6. 이렇게 하면 얼굴 영역 안에서 눈을 더 정확하게 감지합니다.
7. Haar Cascade (Eye) -> Image Show를 연결하고 Execute합니다.
8. 얼굴에 큰 사각형, 눈에 작은 사각형이 그려집니다.

[Tip] 단체 사진에서 minNeighbors를 높이면 오검출이 줄어듭니다.

---

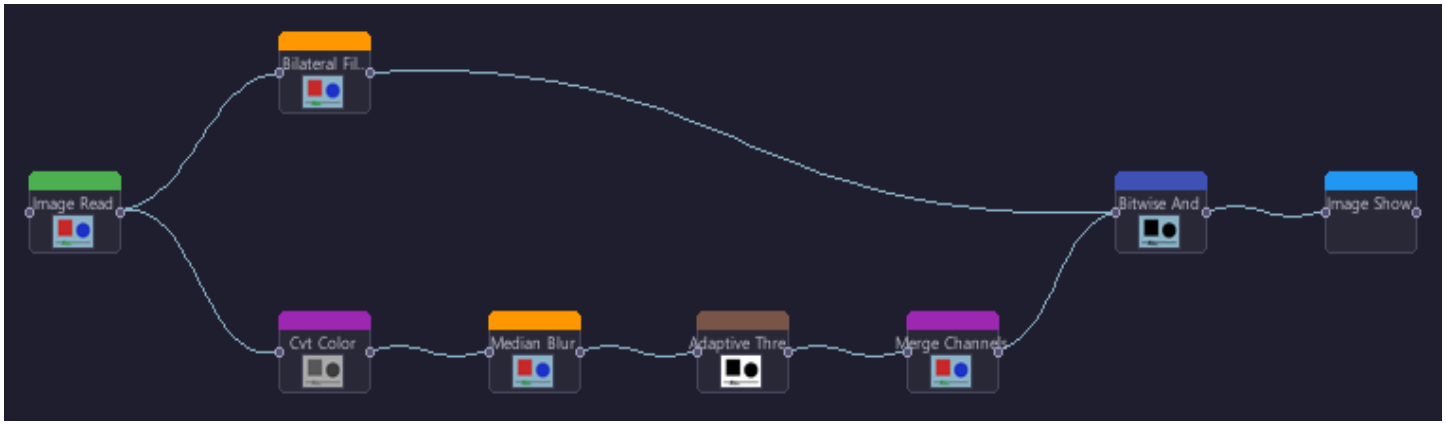
## 예제 6: 카툰 효과 (Cartoon Effect)

난이도: ★★★★ | 저장파일: 06_cartoon_effect.json

**학습 목표: 양방향 필터 + 에지 마스크를 결합하여 만화 스타일 효과를 만듭니다.**

**핵심 개념: 양방향 필터링 | 적응형 임계값 | 채널 병합 | Bitwise AND 마스킹**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 이미지를 업로드합니다.
2. [색상 경로] Bilateral Filter (d=9, sigmaColor=75, sigmaSpace=75)를 추가합니다.
3. Image Read -> Bilateral Filter를 연결합니다. 에지를 보존하며 부드럽게 합니다.
4. [에지 경로] CvtColor(BGR2GRAY) -> Median Blur(ksize=7) -> Adaptive Threshold를 연결합니다.
5. Adaptive Threshold: Method=MEAN_C, Block Size=9, C=2. 에지가 검은 선으로 표현됩니다.
6. Merge Channels를 추가합니다. Adaptive Threshold 출력을 ch0, ch1, ch2 모두에 연결합니다.
7. 이렇게 하면 1채널 에지 이미지가 3채널(BGR)로 변환됩니다.
8. [합성] Bitwise AND를 추가합니다.
9. Bilateral -> image, Merge Channels -> image2에 연결합니다.
10. Bitwise AND -> Image Show를 연결하고 Execute합니다.
11. 만화처럼 색상은 부드럽고 에지가 선명한 이미지가 생성됩니다.

[Tip] Bilateral Filter의 반복 횟수를 늘리면(d 값 증가) 더 강한 카툰 효과를 얻습니다.

---

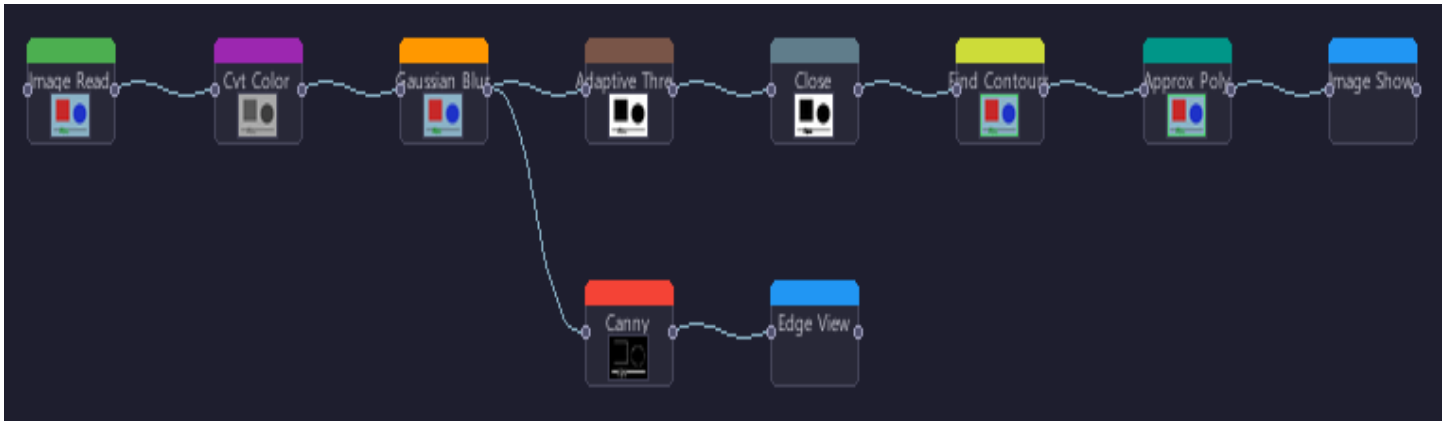
## 예제 7: 문서 스캐너 (Document Scanner)

난이도: ★★★ | 저장파일: 07_document_scanner.json

**학습 목표: 에지 검출과 컨투어 분석으로 문서 윤곽을 감지합니다.**

**핵심 개념: 적응형 임계값 | 모폴로지 닫기 | 컨투어 검출 | 다각형 근사**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**




**단계별 구성 방법:**

1. Image Read에 문서 사진을 업로드합니다. 배경과 문서의 대비가 중요합니다.
2. CvtColor (BGR->GRAY)으로 변환합니다.
3. Gaussian Blur (ksize=5)로 노이즈를 줄입니다.
4. Adaptive Threshold (GAUSSIAN_C, blockSize=11, C=2)를 적용합니다.
5. Morphology (MORPH_CLOSE, ksize=3, iterations=2)로 에지 끊김을 연결합니다.
6. Find Contours (RETR_EXTERNAL)로 외곽 윤곽을 찾습니다.
7. Approx Poly (epsilon=0.02)를 연결합니다. 4개 꼭지점이 감지되면 문서 윤곽입니다.
8. Execute하면 문서 영역에 녹색 다각형이 그려집니다.
9. * Gaussian에서 분기하여 Canny -> Image Show로 에지를 보조 확인할 수 있습니다.

[Tip] 감지된 4꼭지점을 Warp Perspective에 적용하면 문서를 정면 뷰로 변환할 수 있습니다.

# 예제 8: 히스토그램 CLAHE 보정

난이도: ★★ | 저장파일: 08_histogram_clahe.json

**학습 목표: CLAHE 알고리즘으로 이미지 대비를 개선하고 히스토그램으로 비교합니다.**

**핵심 개념: CLAHE | 히스토그램 평활화 | 대비 보정 | 히스토그램 시각화**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**


**단계별 구성 방법:**

1. Image Read에 어둡거나 대비가 낮은 이미지를 업로드합니다.
2. Histogram EQ를 추가합니다. CLAHE=true, Clip Limit=2.0, Tile Size=8.
3. Image Read -> Histogram EQ -> Image Show(보정 후)를 연결합니다.
4. 원본 비교: Image Read -> 다른 Image Show(원본)에도 연결합니다.
5. Calc Histogram 두 개를 추가합니다.
6. 하나는 Image Read -> Calc Histogram -> Image Show (원본 히스토그램).
7. 다른 하나는 Histogram EQ -> Calc Histogram -> Image Show (보정 후 히스토그램).

8. Execute하면 원본/보정 이미지와 히스토그램을 비교할 수 있습니다.

[Tip] Clip Limit을 높이면 대비가 강해지지만, 너무 높으면 노이즈가 증폭됩니다.

---

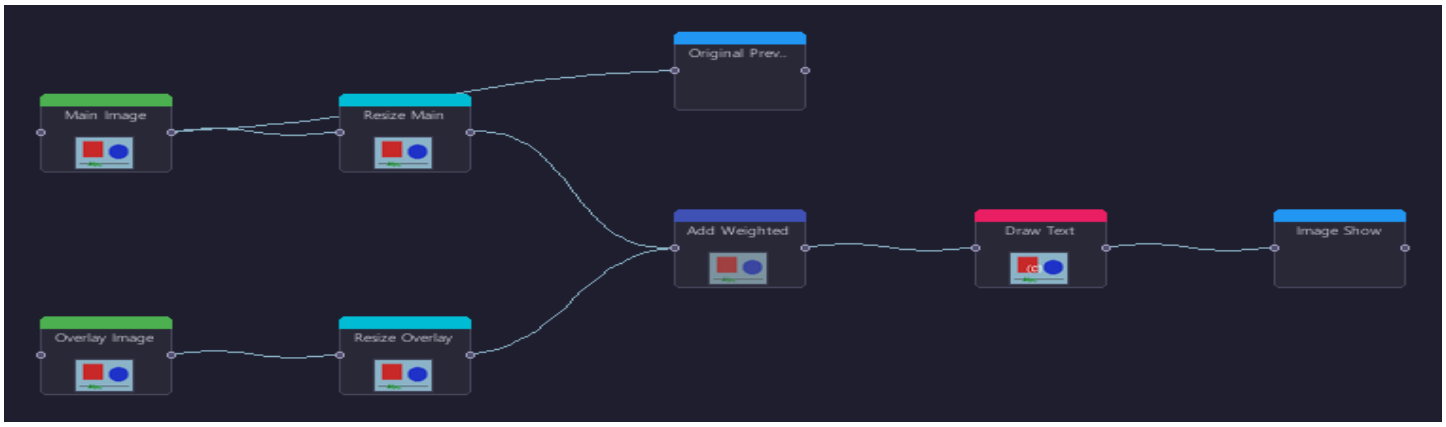
## 예제 9: 이미지 워터마크 합성

난이도: ★★ | 저장파일: 09_watermark_blend.json

**학습 목표: 두 이미지를 블렌딩하고 텍스트 워터마크를 추가합니다.**

**핵심 개념: Alpha 블렌딩 | 텍스트 드로잉 | 워터마크 합성**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**




**단계별 구성 방법:**

1. Image Read 두 개를 추가합니다. 메인 이미지와 워터마크/로고 이미지를 업로드합니다.
2. 각각 Resize로 동일한 크기(640x480)로 맞춥니다.
3. Add Weighted를 추가합니다. Alpha=0.7(메인), Beta=0.3(오버레이).

4. 메인 Resize -> image, 오버레이 Resize -> image2에 연결합니다.
5. Draw Text를 추가합니다. Text="(C) SamOpenCV", 위치=(20,460), 색상=흰색.
6. Add Weighted -> Draw Text -> Image Show를 연결합니다.
7. Execute하면 두 이미지가 합성되고 텍스트 워터마크가 추가됩니다.

[Tip] Alpha를 1.0에 가깝게 하면 메인 이미지가 더 선명하게 보입니다.

---

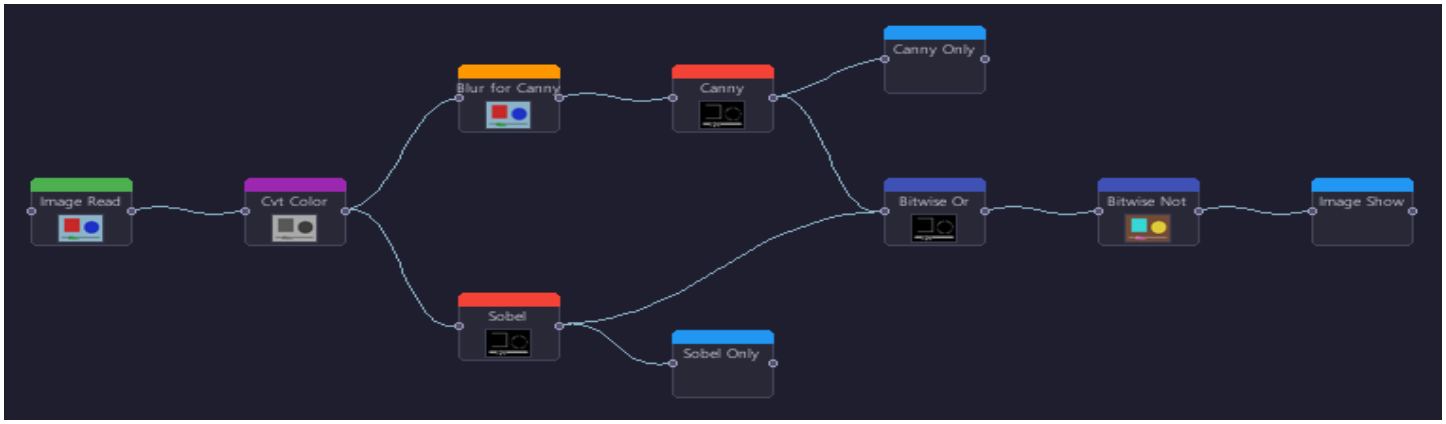
## 예제 10: 에지 아트 (Multi-Edge Art)

난이도: ★★★ | 저장파일: 10_edge_art.json

**학습 목표: 다중 에지 검출기의 결과를 결합하여 예술적 효과를 만듭니다.**

**핵심 개념: 캐니 에지 | 소벨 에지 | Bitwise OR 결합 | 비트 반전**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**




**단계별 구성 방법:**

1. Image Read에 풍경 또는 인물 이미지를 업로드합니다.
2. CvtColor (BGR->GRAY)로 그레이스케일 변환합니다.
3. [캐니 경로] Gaussian Blur(ksize=3) -> Canny(50, 150)을 연결합니다.
4. [소벨 경로] CvtColor에서 Sobel(dx=1, dy=1)을 분기합니다.
5. Bitwise OR를 추가합니다. Canny -> image, Sobel -> image2에 연결합니다.
6. 두 에지 검출 결과가 합성됩니다.
7. Bitwise NOT을 추가합니다. 흰 배경에 검은 선으로 반전됩니다.
8. Image Show를 연결하고 Execute합니다.
9. 마치 연필 스케치처럼 보이는 에지 아트가 생성됩니다.

[Tip] Laplacian이나 Scharr도 Bitwise OR에 추가하면 더 풍부한 에지 표현을 얻습니다.

---

## 예제 11: 노이즈 제거 비교 (Blur Comparison)

난이도: ★★ | 저장파일: 11_noise_removal.json

**학습 목표: 세 가지 블러 필터(Gaussian, Median, Bilateral)의 노이즈 제거 효과를 비교합니다.**

**핵심 개념: 가우시안 블러 | 미디언 블러 | 양방향 필터 | 노이즈 제거**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**

**단계별 구성 방법:**

1. Image Read에 노이즈가 있는 이미지를 업로드합니다.
2. Gaussian Blur (ksize=5)를 연결합니다. 전체적으로 균일하게 흐려집니다.
3. Median Blur (ksize=5)를 별도 연결합니다. 소금-후추 노이즈에 효과적입니다.
4. Bilateral Filter (d=9, sigmaColor=75)를 별도 연결합니다. 에지를 보존합니다.
5. 각각 Image Show에 연결하고 Execute합니다.
6. 세 가지 필터의 결과를 비교하여 상황에 맞는 필터를 선택합니다.

[Tip] Gaussian은 범용, Median은 소금-후추, Bilateral은 에지 보존에 적합합니다.

---

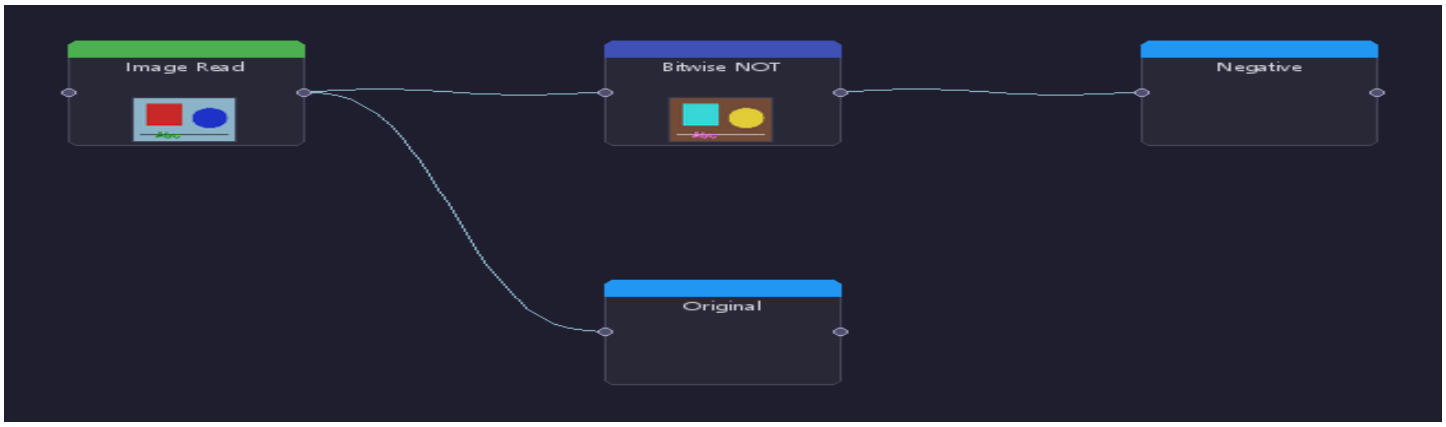
## 예제 12: 이미지 네거티브 (반전)

난이도: ★ | 저장파일: 12_image_negative.json

**학습 목표: Bitwise NOT으로 이미지 색상을 반전시킵니다.**

**핵심 개념: 비트 반전 | 네거티브 이미지 | Bitwise NOT**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 이미지를 업로드합니다.
2. Bitwise NOT을 연결합니다. 모든 픽셀 값이 255에서 빼진 값으로 변환됩니다.
3. Image Show를 연결하고 Execute합니다.
4. 원본과 비교: 별도 Image Show에 Image Read를 직접 연결합니다.

[Tip] X-ray 이미지 분석이나 예술적 효과에 활용됩니다.

---

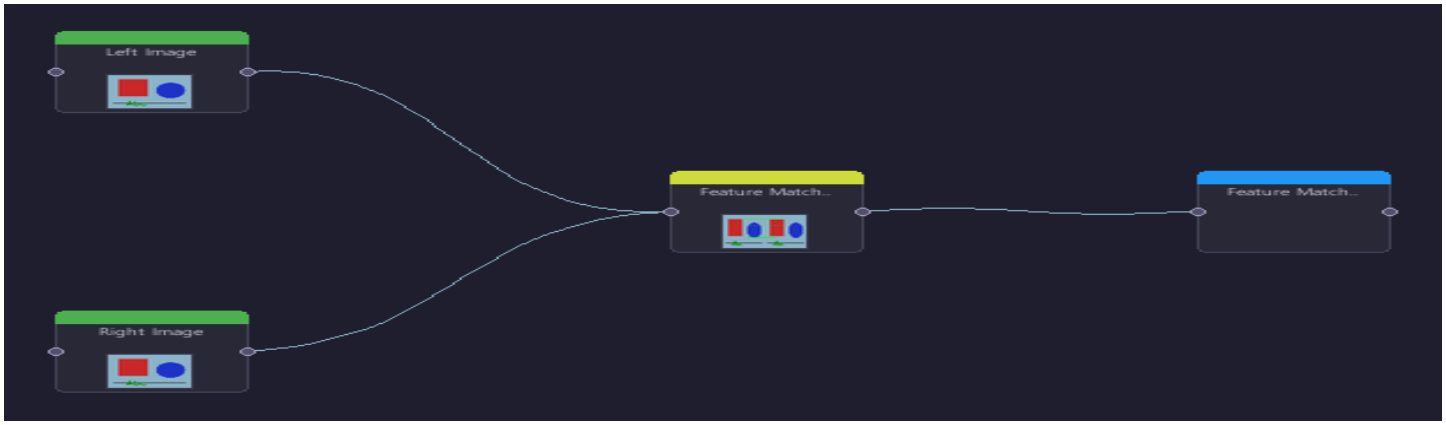
## 예제 13: 파노라마용 특징 매칭 (Feature Matching)

난이도: ★★★ | 저장파일: 13_panorama_matching.json

**학습 목표: 두 이미지의 ORB 특징점을 매칭하여 파노라마 합성 가능성을 확인합니다.**

**핵심 개념: ORB 특징점 | 특징 매칭 | 파노라마 | 매칭 비율**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read 두 개를 추가합니다. 겹치는 영역이 있는 좌/우 이미지를 업로드합니다.
2. Match Features를 추가합니다. nFeatures=500, Match Ratio=0.75.
3. 왼쪽 이미지 -> image, 오른쪽 이미지 -> image2에 연결합니다.
4. Image Show를 연결하고 Execute합니다.
5. 매칭된 특징점 쌍이 선으로 연결되어 표시됩니다.

[Tip] Match Ratio를 낮추면 더 정확한 매칭만 남고, 높이면 더 많은 매칭이 표시됩니다.

---

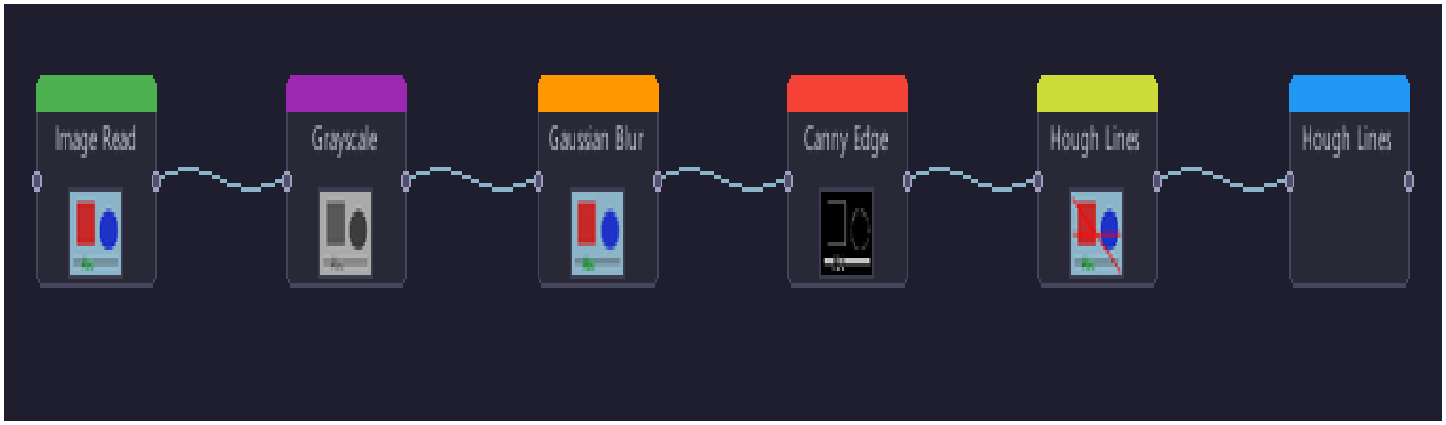
## 예제 14: 허프 직선 검출 (Hough Lines)

난이도: ★★★ | 저장파일: 14_hough_line_detection.json

**학습 목표: Hough 변환으로 이미지에서 직선을 검출합니다.**

**핵심 개념: 허프 변환 | 직선 검출 | 캐니 에지 | 파라미터 공간**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 건물이나 도로 사진을 업로드합니다.
2. CvtColor(BGR2GRAY) -> Gaussian Blur(ksize=5) -> Canny(50, 150)를 연결합니다.
3. Hough Lines를 추가합니다. Rho=1, Theta Divisor=180, Threshold=100.
4. Canny -> Hough Lines -> Image Show를 연결하고 Execute합니다.
5. 검출된 직선이 이미지 위에 빨간 선으로 표시됩니다.

[Tip] Threshold를 낮추면 더 많은 직선이 검출되고, 높이면 확실한 직선만 검출됩니다.

---

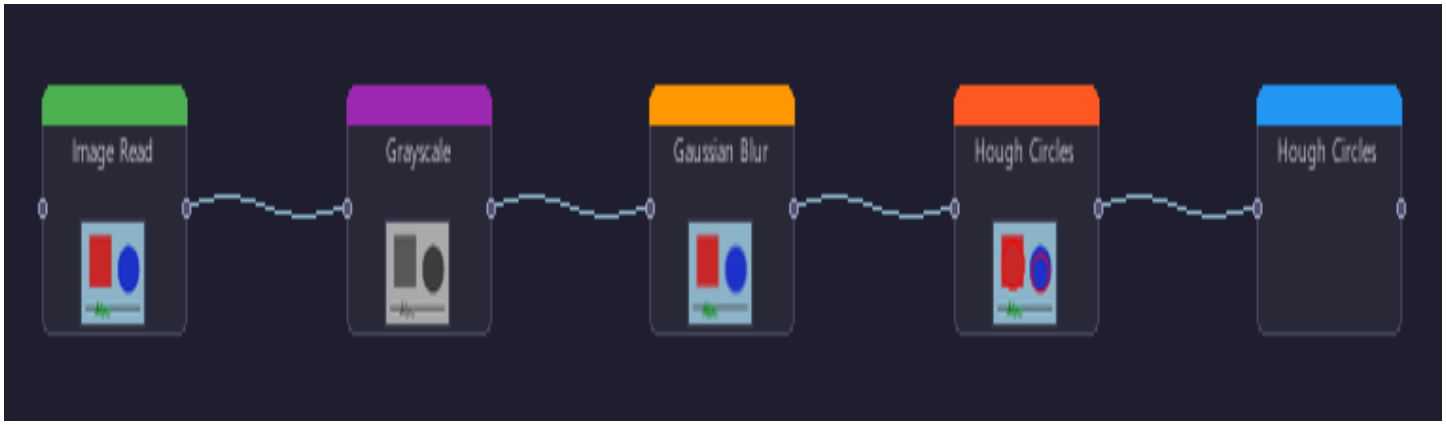
## 예제 15: 허프 원 검출 (Hough Circles)

난이도: ★★★ | 저장파일: 15_hough_circle_detection.json

**학습 목표: Hough 원 변환으로 이미지에서 원형 객체를 검출합니다.**

**핵심 개념: 허프 원 변환 | 원 검출 | 가우시안 블러 | 파라미터 튜닝**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 동전이나 공 등 원형 물체 이미지를 업로드합니다.
2. CvtColor(BGR2GRAY)로 변환합니다.
3. Gaussian Blur (ksize=9)로 노이즈를 충분히 줄입니다. 원 검출에 중요합니다.
4. Hough Circles를 추가합니다. dp=1.2, minDist=50, param2=30, minRadius=10, maxRadius=100.
5. Execute하면 검출된 원이 표시됩니다.

[Tip] minDist는 원 사이 최소 거리, param2를 낮추면 더 많은 원이 검출됩니다.

---

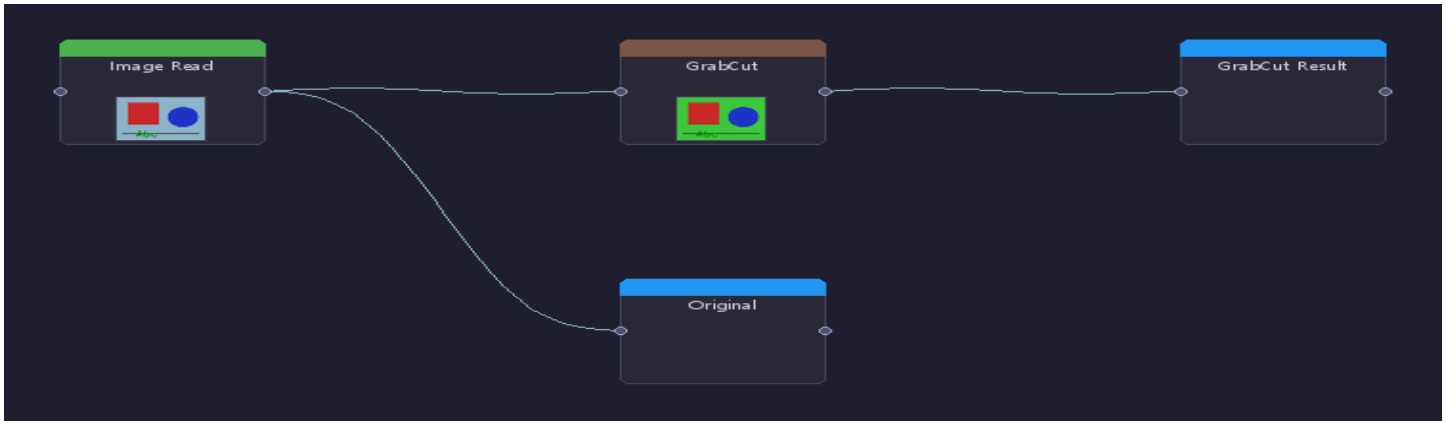
## 예제 16: GrabCut 전경 추출

난이도: ★★ | 저장파일: 16_grabcut_segmentation.json

**학습 목표: GrabCut 알고리즘으로 이미지에서 전경 객체를 자동 분리합니다.**

**핵심 개념: GrabCut | 전경 분리 | ROI 영역 | 반복 최적화**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 전경 객체가 있는 이미지를 업로드합니다.
2. GrabCut을 추가합니다. 전경 객체를 포함하는 ROI를 설정합니다: x=50, y=50, width=400, height=300.
3. Iterations=5로 설정합니다. 반복 횟수가 많을수록 정확합니다.
4. Image Read -> GrabCut -> Image Show를 연결하고 Execute합니다.
5. 전경 객체만 추출되어 표시됩니다.

[Tip] ROI가 전경 객체를 충분히 포함해야 합니다. Iterations를 높이면 결과가 개선됩니다.

---

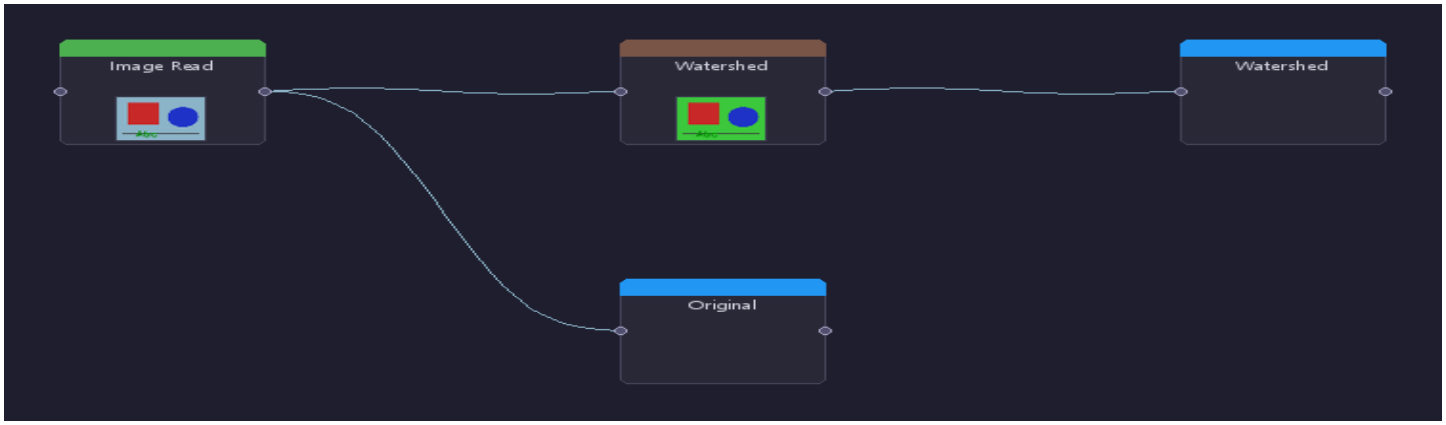
## 예제 17: Watershed 분할

난이도: ★★★ | 저장파일: 17_watershed_segmentation.json

**학습 목표: Watershed 알고리즘으로 이미지를 영역별로 분할합니다.**

**핵심 개념: Watershed | 영역 분할 | 마커 기반 분할**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 여러 객체가 있는 이미지를 업로드합니다.
2. Watershed를 추가합니다. Marker Size=3.
3. Image Read -> Watershed -> Image Show를 연결합니다.
4. Execute하면 영역 경계가 색상으로 표시됩니다.
5. 별도 Image Show에 원본을 연결하여 비교합니다.

[Tip] 겹쳐진 동전, 셀 등의 분리에 효과적입니다.

---

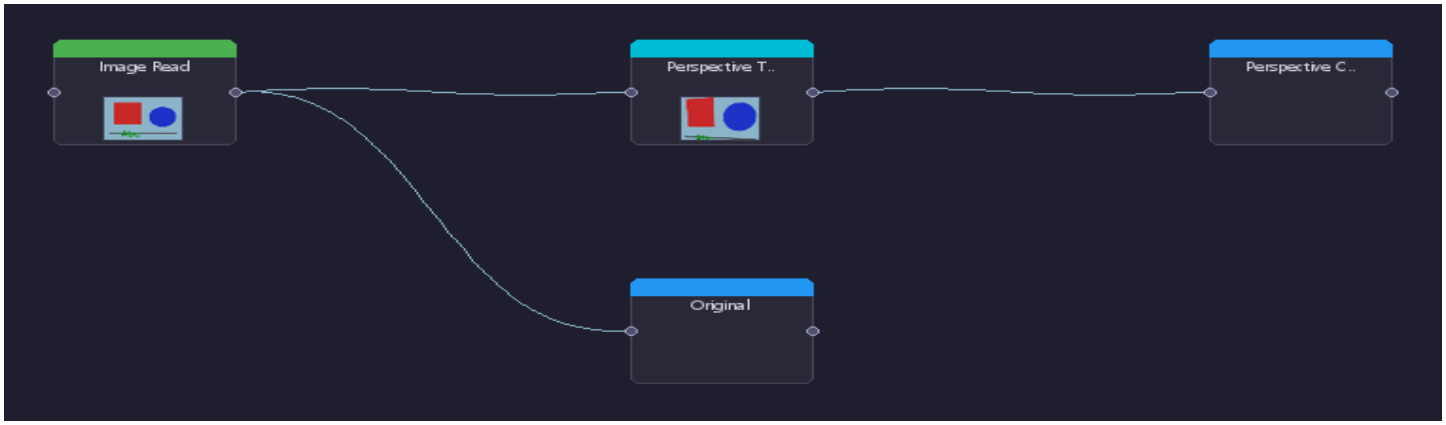
## 예제 18: 원근 변환 보정 (Perspective Transform)

난이도: ★★★ | 저장파일: 18_perspective_transform.json

**학습 목표: 비스듬하게 촬영된 이미지를 정면 뷰로 변환합니다.**

**핵심 개념: 원근 변환 | 4점 변환 | 호모그래피**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 비스듬하게 촬영된 문서/간판 이미지를 업로드합니다.
2. Warp Perspective를 추가합니다.
3. 소스 좌표 4개: 원본 이미지에서 문서의 네 꼭지점을 지정합니다.
4. 대상 좌표 4개: 변환 후 위치를 직사각형으로 지정합니다.
5. Execute하면 원근이 보정된 정면 뷰 이미지가 생성됩니다.

[Tip] 문서 스캐너(예제 7)의 Approx Poly로 감지된 꼭지점을 여기에 활용할 수 있습니다.

---

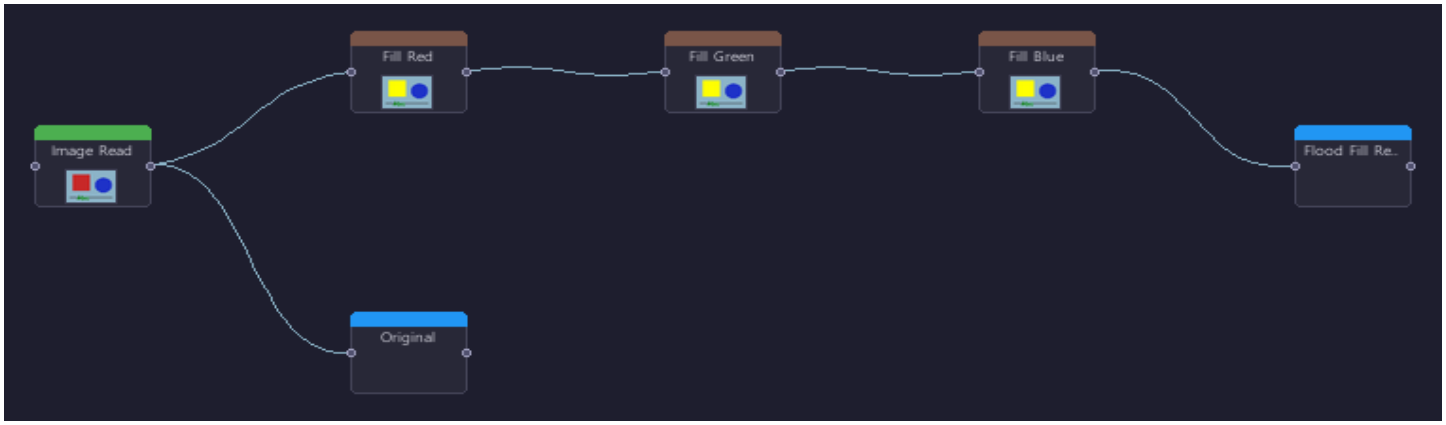
## 예제 19: Flood Fill 색칠

난이도: ★★ | 저장파일: 19_flood_fill_coloring.json

**학습 목표: Flood Fill로 이미지의 특정 영역을 색칠합니다.**

**핵심 개념: Flood Fill | 시드 포인트 | 색상 허용 범위 | 영역 채우기**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 이미지를 업로드합니다.
2. 첫 번째 Flood Fill: seedX=100, seedY=100, Color=Red(255,0,0), loDiff=30, upDiff=30.
3. 두 번째 Flood Fill: seedX=200, seedY=200, Color=Green(0,255,0).
4. 세 번째 Flood Fill: seedX=300, seedY=150, Color=Blue(0,0,255).
5. 직렬로 연결하여 3개 영역을 순차적으로 색칠합니다.
6. Execute하면 각 시드 포인트 주변의 유사 색상 영역이 채워집니다.

[Tip] loDiff와 upDiff를 높이면 더 넓은 색상 범위가 채워집니다.

---

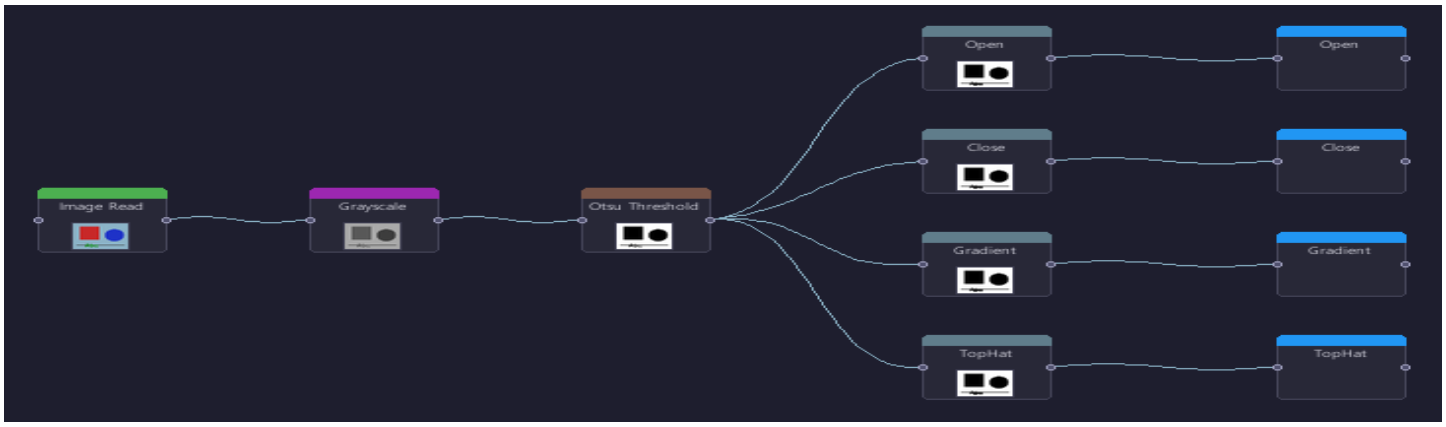
## 예제 20: 형태학 연산 비교 (Morphology Comparison)

난이도: ★★ | 저장파일: 20_morphology_comparison.json

**학습 목표: Open, Close, Gradient, Top Hat 형태학 연산의 차이를 비교합니다.**

**핵심 개념: 열기(Open) | 닫기(Close) | 그래디언트 | Top Hat | 이진화**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 이미지를 업로드합니다.
2. CvtColor(BGR2GRAY) -> Otsu Threshold로 이진화합니다.
3. Morphology 4개를 추가합니다: MORPH_OPEN, MORPH_CLOSE, MORPH_GRADIENT, MORPH_TOPHAT.
4. Otsu 출력에서 4개로 분기하여 각각 Image Show에 연결합니다.
5. Execute하면 4가지 연산 결과를 비교할 수 있습니다.
6. Open: 노이즈 제거, Close: 빈 공간 채움, Gradient: 에지, TopHat: 밝은 요소 추출.

[Tip] ksize와 iterations를 조정하면 연산 강도가 달라집니다.

---

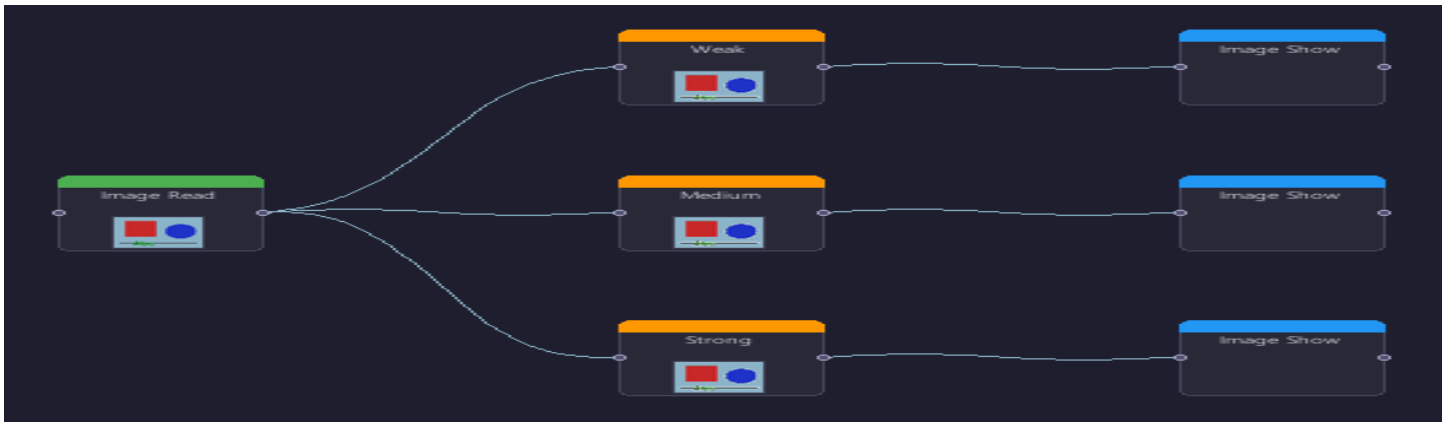
## 예제 21: 샤프닝 강도 비교 (Sharpen Comparison)

난이도: ★ | 저장파일: 21_sharpen_compare.json

**학습 목표: 다양한 샤프닝 강도의 효과를 비교합니다.**

**핵심 개념: 언샤프 마스크 | 샤프닝 강도 | 이미지 선명화**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 약간 흐릿한 이미지를 업로드합니다.
2. Sharpen 3개를 추가합니다: Strength=0.5(약), 1.5(중), 3.0(강).
3. Image Read에서 3개로 분기하여 각각 Image Show에 연결합니다.
4. Execute하면 3단계 샤프닝 강도를 비교할 수 있습니다.
5. 너무 강하면 노이즈가 증폭되므로 적절한 값을 선택합니다.

[Tip] Strength=1.0~2.0이 대부분의 경우에 적합합니다.

---

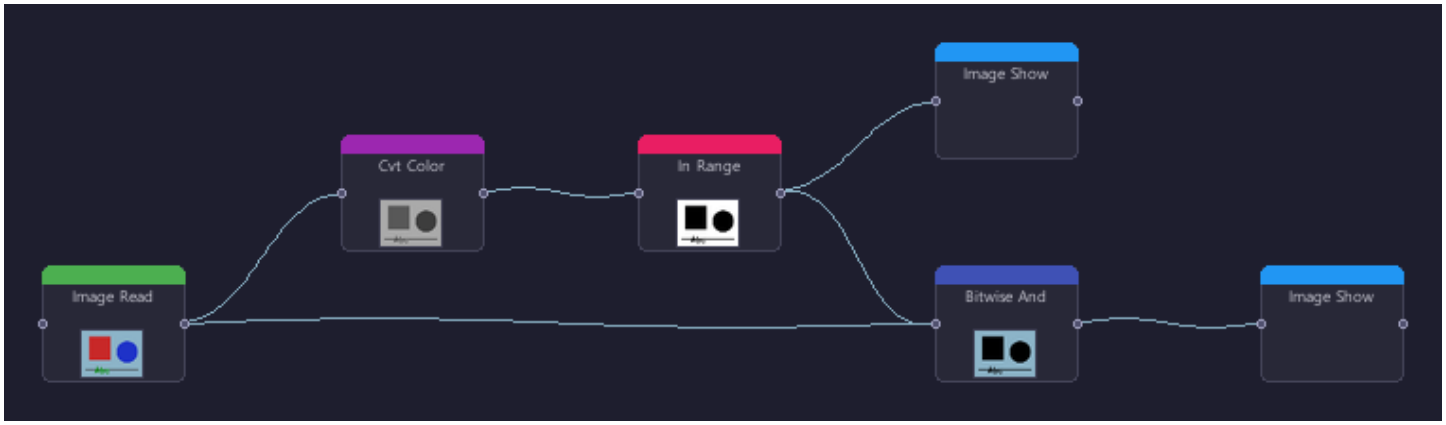
## 예제 22: 색상 추출 (Color Extraction)

난이도: ★★★ | 저장파일: 22_color_extraction.json

**학습 목표: HSV 마스크로 특정 색상만 추출하여 원본에 적용합니다.**

**핵심 개념: HSV 마스크 | Bitwise AND 마스킹 | 색상 분리**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 다양한 색상이 있는 이미지를 업로드합니다.
2. CvtColor(BGR2HSV) -> InRange로 빨간색 마스크를 생성합니다: H=0~10, S=100~255, V=100~255.
3. Bitwise AND를 추가합니다.
4. Image Read(원본) -> image 포트, InRange(마스크) -> image2 포트에 연결합니다.
5. Execute하면 빨간색 부분만 원래 색상으로, 나머지는 검정으로 표시됩니다.
6. InRange의 범위를 변경하면 다른 색상도 추출할 수 있습니다.

[Tip] 여러 InRange + Bitwise OR로 복수 색상을 동시에 추출할 수 있습니다.

---

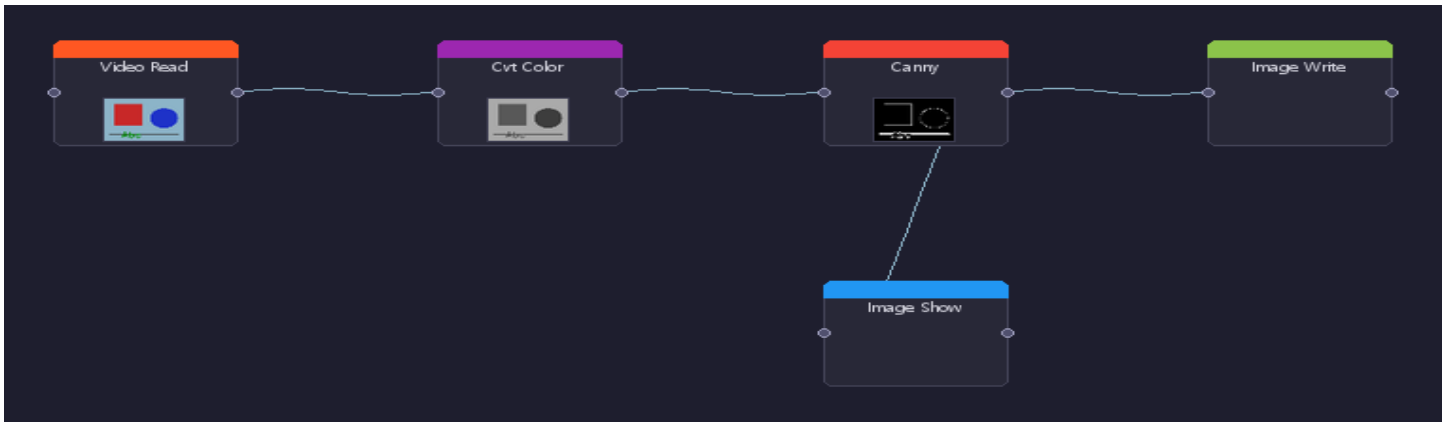
## 예제 23: 비디오 에지 출력 (Video Edge Export)

난이도: ★★ | 저장파일: 23_video_edge_output.json

**학습 목표: 비디오의 모든 프레임에 에지 검출을 적용하여 새 비디오로 저장합니다.**

**핵심 개념: 비디오 처리 | 프레임별 에지 검출 | 비디오 출력**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Video Read를 추가하고 비디오를 업로드합니다. Mode=loop.
2. CvtColor(BGR2GRAY) -> Canny(100, 200)를 연결합니다.
3. Image Write를 추가합니다. Video Output=true, filepath=output_edges.mp4, Codec=mp4v, FPS=30.
4. Canny -> Image Write를 연결합니다.
5. 분기하여 Canny -> Image Show를 연결하면 미리보기도 됩니다.
6. Execute하면 에지 검출된 비디오가 저장됩니다.

[Tip] Canny 대신 다른 에지/필터 노드를 연결하면 다양한 효과의 비디오를 생성합니다.

---

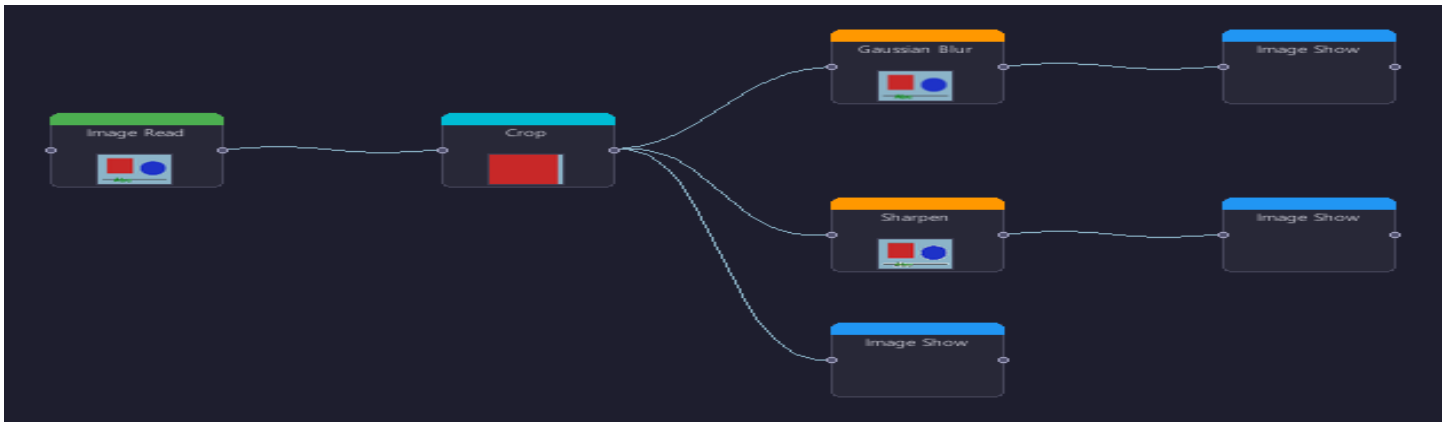
## 예제 24: ROI 영역 집중 처리

난이도: ★★ | 저장파일: 24_roi_focus.json

**학습 목표: 이미지의 특정 영역(ROI)을 잘라내어 집중적으로 처리합니다.**

**핵심 개념: ROI(Region of Interest) | Crop | 블러 | 샤프닝**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 이미지를 업로드합니다.
2. Crop을 추가합니다. x=100, y=100, width=200, height=200으로 관심 영역을 잘라냅니다.
3. Crop에서 분기하여 Gaussian Blur(ksize=11)와 Sharpen(strength=2.0)을 각각 연결합니다.
4. 각각 Image Show에 연결하고 Execute합니다.
5. ROI 영역에 대해 블러와 샤프닝 효과를 비교할 수 있습니다.

[Tip] Crop 좌표를 변경하여 다양한 영역을 분석할 수 있습니다.

---

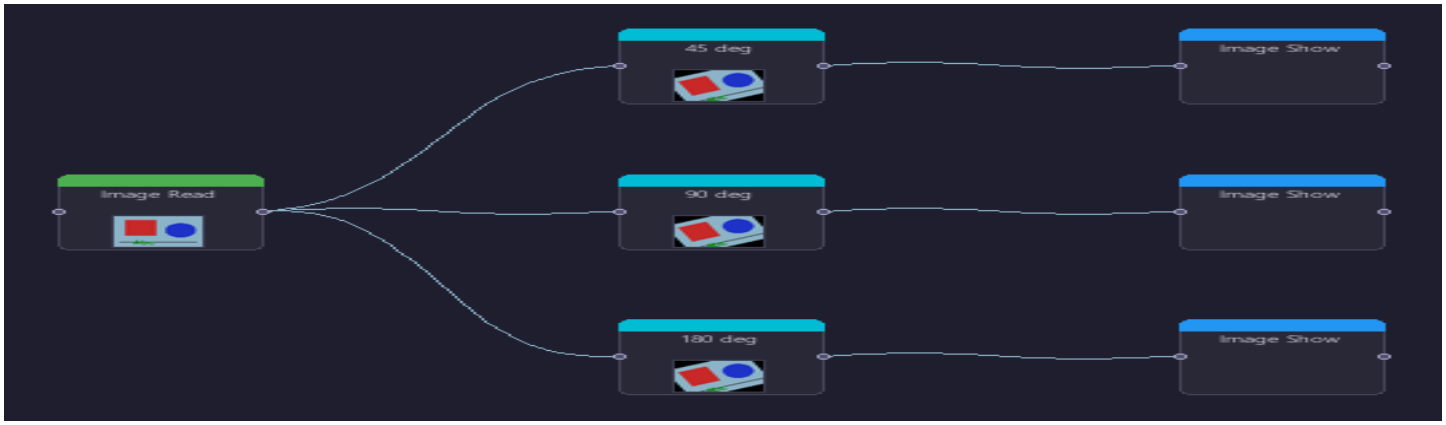
## 예제 25: 이미지 회전 갤러리 (Rotation Gallery)

난이도: ★ | 저장파일: 25_image_rotation_gallery.json

**학습 목표: 다양한 각도로 이미지를 회전시켜 갤러리를 만듭니다.**

**핵심 개념: 이미지 회전 | 각도 변환 | 보간법**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 이미지를 업로드합니다.
2. Rotate 3개를 추가합니다: 45도, 90도, 180도.
3. Image Read에서 3개로 분기하여 각각 Image Show에 연결합니다.
4. Execute하면 3가지 회전 결과를 동시에 비교할 수 있습니다.

[Tip] Flip 노드를 추가하면 좌우/상하 반전도 함께 비교할 수 있습니다.

---

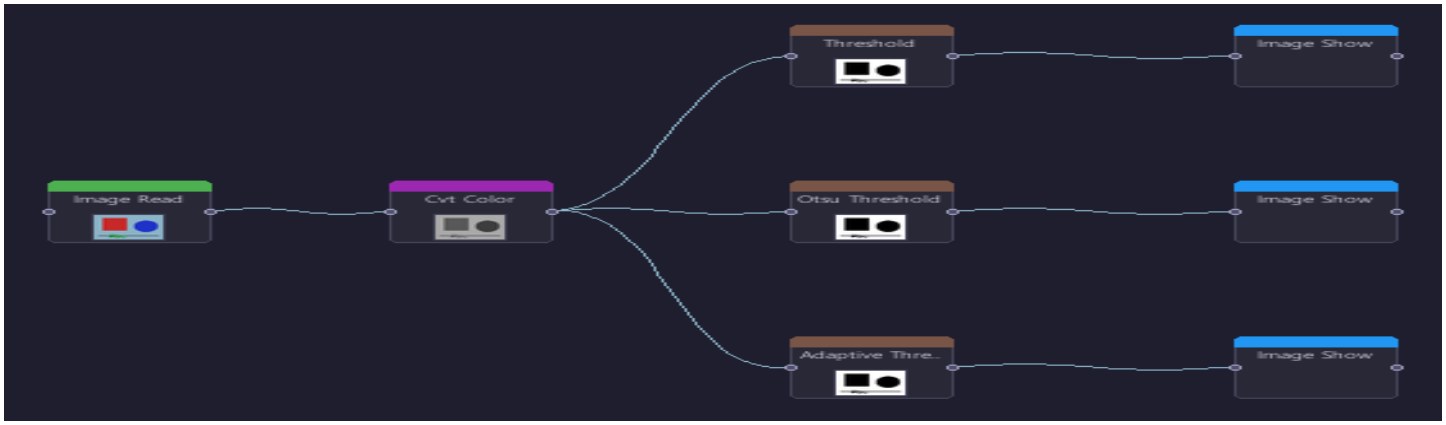
## 예제 26: 임계값 방법 비교 (Threshold Comparison)

난이도: ★★ | 저장파일: 26_threshold_comparison.json

**학습 목표: Binary, Otsu, Adaptive 세 가지 임계값 방법을 비교합니다.**

**핵심 개념: 고정 임계값 | Otsu 자동 임계값 | 적응형 임계값**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 문서 또는 텍스트 이미지를 업로드합니다.
2. CvtColor(BGR2GRAY)로 변환합니다.
3. CvtColor에서 3개로 분기합니다:
4. 1) Threshold (thresh=127, BINARY): 고정값 이진화.
5. 2) Otsu Threshold: 자동으로 최적 임계값 결정.
6. 3) Adaptive Threshold (GAUSSIAN_C, blockSize=11, C=2): 영역별 적응적 이진화.
7. 각각 Image Show에 연결하고 Execute합니다.
8. 조명이 불균일한 이미지에서는 Adaptive가 가장 좋은 결과를 보입니다.

[Tip] blockSize를 변경하면 Adaptive Threshold의 참조 영역이 달라집니다.

---

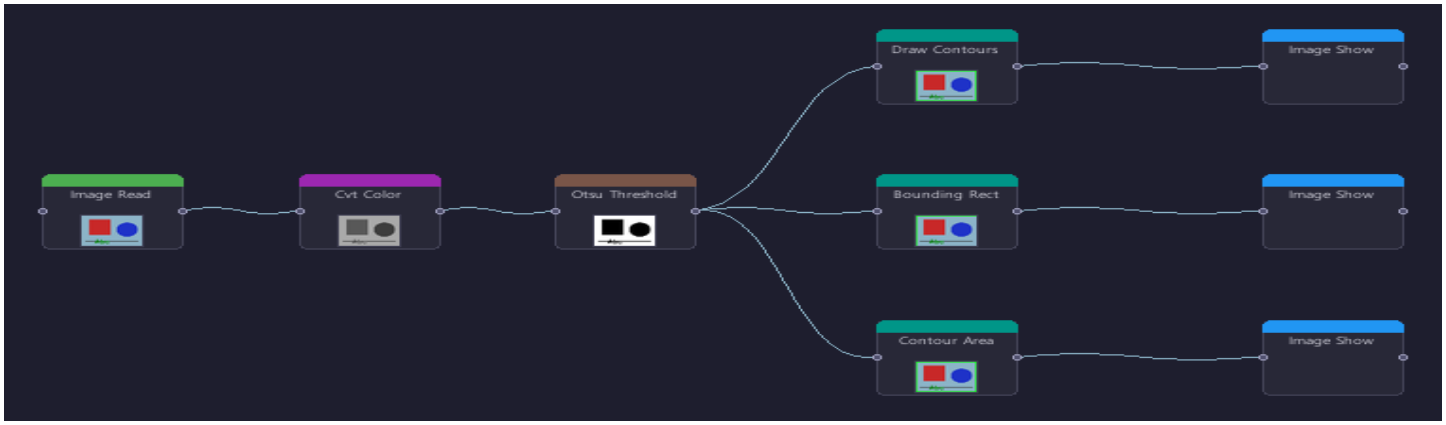
## 예제 27: 컨투어 분석 종합

난이도: ★★★ | 저장파일: 27_contour_analysis.json

**학습 목표: 다양한 컨투어 분석 도구의 결과를 비교합니다.**

**핵심 개념: 컨투어 그리기 | 바운딩 박스 | 면적 필터링 | 윤곽 분석**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 여러 객체가 있는 이미지를 업로드합니다.
2. CvtColor(BGR2GRAY) -> Otsu Threshold로 이진화합니다.
3. Otsu에서 3개로 분기합니다:
4. 1) Draw Contours (녹색): 모든 윤곽선을 그립니다.
5. 2) Bounding Rect (빨간색): 외접 사각형을 그립니다.
6. 3) Contour Area (파란색, minArea=500): 작은 노이즈를 필터링합니다.
7. 각각 Image Show에 연결하고 Execute합니다.

[Tip] Contour Properties를 추가하면 면적, 둘레, 중심점 수치를 확인할 수 있습니다.

---

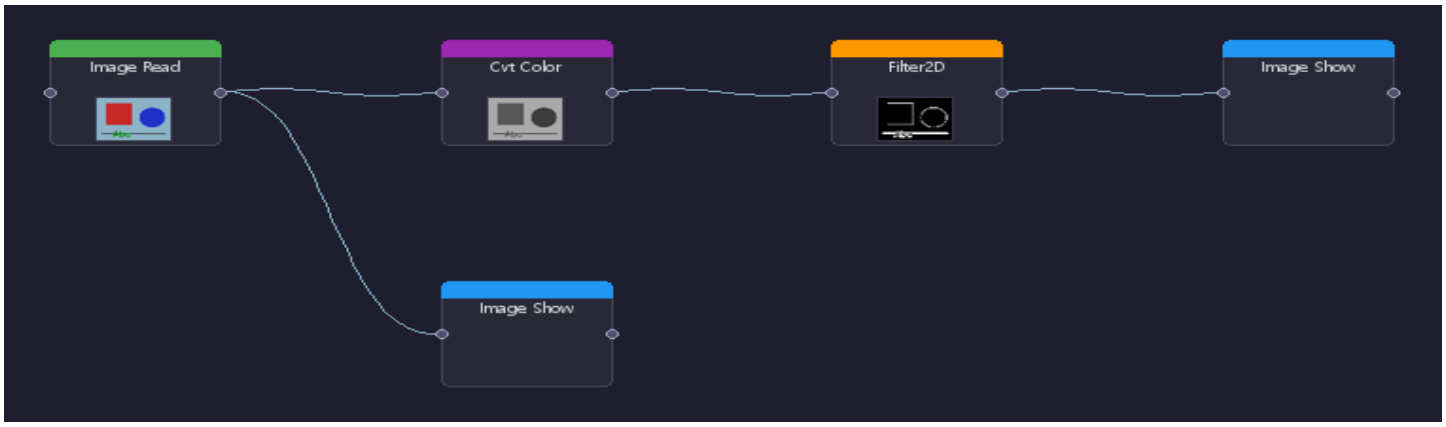
## 예제 28: 엠보싱 효과 (Emboss Effect)

난이도: ★ | 저장파일: 28_emboss_effect.json

**학습 목표: Filter2D 엠보스 프리셋으로 입체 느낌의 효과를 만듭니다.**

**핵심 개념: 커스텀 커널 | 컨볼루션 | 엠보싱 | Filter2D**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 이미지를 업로드합니다.
2. CvtColor(BGR2GRAY)로 그레이스케일 변환합니다.
3. Filter2D를 추가합니다. Preset=emboss, Kernel Size=3.
4. CvtColor -> Filter2D -> Image Show를 연결하고 Execute합니다.
5. 입체적인 양각 효과가 적용됩니다.

[Tip] Preset을 sharpen이나 edge_detect로 변경하면 다른 커널 효과를 볼 수 있습니다.

---

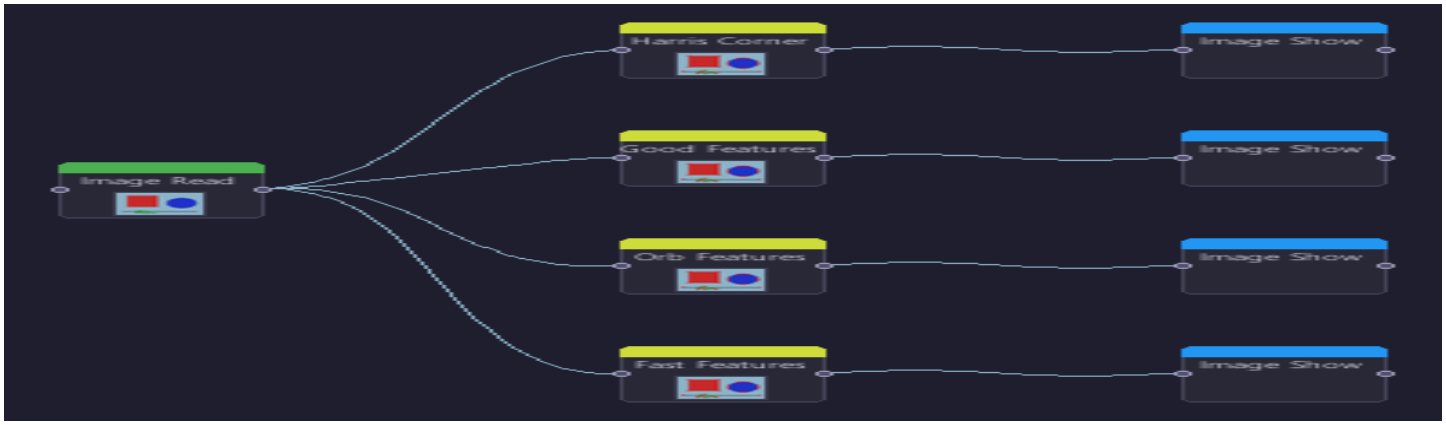
## 예제 29: 특징점 검출 비교 (Feature Detection)

난이도: ★★★ | 저장파일: 29_feature_detection_compare.json

**학습 목표: Harris, Shi-Tomasi, ORB, FAST 네 가지 특징점 검출기를 비교합니다.**

**핵심 개념: Harris 코너 | Shi-Tomasi | ORB | FAST | 특징점 검출**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 건물이나 패턴이 많은 이미지를 업로드합니다.
2. Image Read에서 4개로 분기합니다:
3. 1) Harris Corner: blockSize=2, ksize=3, k=0.04.
4. 2) Good Features: maxCorners=100, qualityLevel=0.01, minDistance=10.
5. 3) ORB Features: nFeatures=500.
6. 4) FAST Features: threshold=25.
7. 각각 Image Show에 연결하고 Execute합니다.
8. 특징점의 수, 분포, 정확도를 비교할 수 있습니다.

[Tip] ORB는 회전 불변, FAST는 속도가 빠르며, Harris는 코너에 강합니다.

---

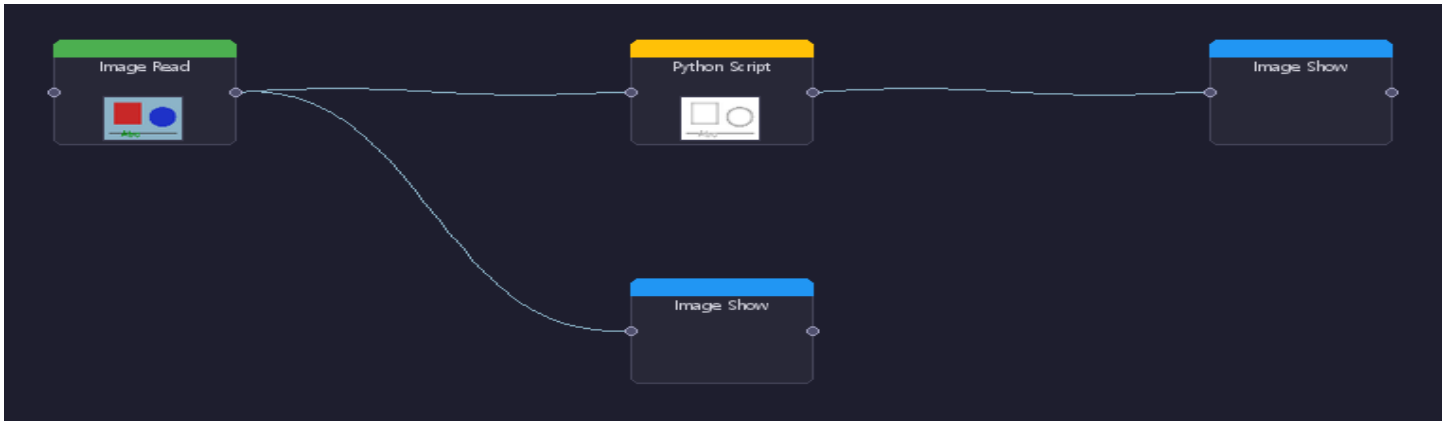
## 예제 30: Python 스크립트 연필 스케치

난이도: ★★★★ | 저장파일: 30_custom_filter.json

**학습 목표: Python Script 노드로 연필 드로잉 효과를 직접 구현합니다.**

**핵심 개념: 커스텀 코드 | 연필 스케치 | cv2.divide | 가우시안 반전**

**노드 그래프 (각 노드에 처리 결과 미리보기 포함):**



**단계별 구성 방법:**

1. Image Read에 인물 또는 풍경 이미지를 업로드합니다.
2. Python Script를 추가하고 아래 코드를 입력합니다:
3. gray = cv2.cvtColor(img_input, cv2.COLOR_BGR2GRAY)
4. inv = 255 - gray
5. blur = cv2.GaussianBlur(inv, (21,21), 0)
6. sketch = cv2.divide(gray, 255-blur, scale=256)
7. img_output = cv2.cvtColor(sketch, cv2.COLOR_GRAY2BGR)
8. Image Read -> Python Script -> Image Show를 연결하고 Execute합니다.
9. 연필로 스케치한 듯한 예술적 효과가 생성됩니다.

[Tip] GaussianBlur 커널 크기를 변경하면 스케치 선의 굵기가 달라집니다.

---

## 예제 31: 이미지 오버레이 (Paste Image)

난이도: ★★ | 저장파일:

**학습 목표: Crop으로 추출한 이미지를 다른 이미지 위에 지정 좌표에 덮어씌웁니다.**

**핵심 개념: 이미지 합성 | overlay | blend | alpha channel | 좌표 지정**

**파이프라인 구성:**


**단계별 구성 방법:**

1. 배경용 Image Read와 오버레이용 Image Read 두 개를 추가합니다.
2. Paste Image 노드를 추가하고, 배경 이미지를 image 포트에, 작은 이미지를 overlay 포트에 연결합니다.
3. x, y 좌표를 설정하여 붙여넣기 위치를 지정합니다.
4. Mode를 선택합니다: overwrite (완전 덮어쓰기), blend (투명도 합성), alpha_channel (PNG 알파 사용).
5. blend 모드에서 Opacity를 0.5로 설정하면 반투명 합성이 됩니다.
6. Execute로 결과를 확인합니다.

[Tip] 로고 삽입, 워터마크, Crop 결과를 다른 이미지에 합성할 때 유용합니다.

---

## 예제 32: 수동 좌표 입력 (Val Coords + Image Extract)

난이도: ★★ | 저장파일:

**학습 목표: Val Coords 노드로 직접 좌표를 입력하여 Image Extract로 영역을 추출합니다.**

**핵심 개념: 좌표 입력 | Image Extract | coords 포트 | 영역 추출**

**파이프라인 구성:**


**단계별 구성 방법:**

1. Image Read로 이미지를 로드합니다.
2. Val Coords 노드를 추가합니다. Single 모드에서 x1, y1, x2, y2 값을 입력합니다.
3. Image Extract 노드를 추가합니다.
4. Image Read → Image Extract(image), Val Coords → Image Extract(coords)로 연결합니다.

5. Execute하면 지정 좌표 영역이 추출됩니다.
6. Multi 모드를 사용하면 여러 좌표를 한 줄에 하나씩 입력 가능합니다.

[Tip] Haar Cascade, Bounding Rect, Template Match의 coords/matches 출력도 Image Extract에 연결 가능합니다.

---

## 예제 33: 크기 불일치 처리 (Size Mismatch)

난이도: ★★ | 저장파일:

**학습 목표: 크기가 다른 두 이미지의 산술 연산에서 Size Mismatch 옵션을 활용합니다.**

**핵심 개념: 크기 불일치 | resize | 산술 연산 | error 모드**

**파이프라인 구성:**


**단계별 구성 방법:**

1. 서로 다른 해상도의 이미지를 Image Read 두 개로 로드합니다 (예: 640x480 + 320x240).
2. Add Weighted를 추가하고 두 이미지를 연결합니다.
3. 기본 설정(Size Mismatch = error)에서는 크기 불일치 에러가 발생합니다.
4. Properties에서 Size Mismatch를 resize_img2로 변경하면 img2를 img1 크기로 자동 리사이즈합니다.
5. resize_img1은 반대로 img1을 img2 크기로 리사이즈합니다.
6. 명시적으로 크기를 맞추려면 별도의 Resize 노드를 사용하는 것을 권장합니다.

[Tip] 모든 두-이미지 산술 노드(Add, Subtract, Multiply, AbsDiff, Bitwise AND/OR/XOR)에 동일한 옵션이 있습니다.

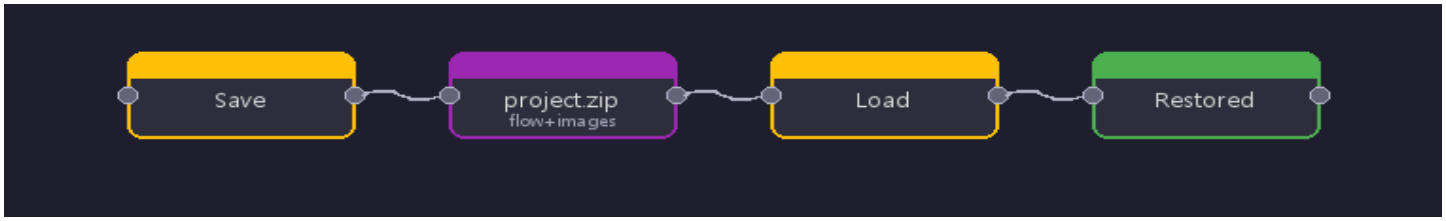
# 예제 34: 프로젝트 저장/불러오기 (Save/Load with Images)

난이도: ★ | 저장파일:

**학습 목표: 이미지가 포함된 프로젝트를 ZIP으로 저장하고 완전히 복원합니다.**

**핵심 개념: 프로젝트 저장 | ZIP | 이미지 포함 | 프리뷰 복원**

**파이프라인 구성:**




**단계별 구성 방법:**

1. 파이프라인을 구성하고 Execute로 실행합니다.
2. 툴바에서 Save를 클릭합니다. 이미지가 있으면 자동으로 .zip 파일로 저장됩니다.
3. ZIP 내부: flow.json (노드 그래프) + images/ 폴더 (원본 이미지들).
4. Load를 클릭하고 저장한 .zip 파일을 선택합니다.
5. 이미지가 자동 복원되고 프리뷰가 표시됩니다. 바로 Execute로 재실행 가능합니다.
6. 이미지가 없는 플로우는 기존처럼 .json으로 저장됩니다.

[Tip] 다른 컴퓨터나 학생 간에 프로젝트를 공유할 때 ZIP 파일 하나로 완전한 전달이 가능합니다.