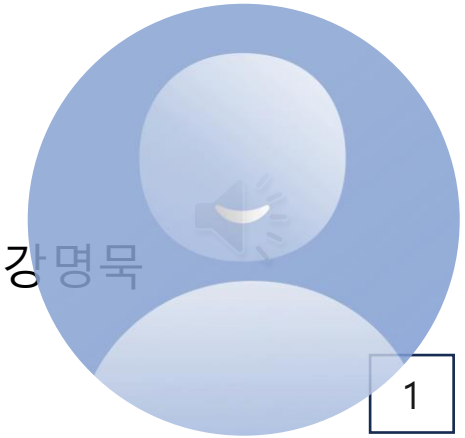
Computer Vision and Deep Learning

Ch 06



강명묵

Contents

매칭

1. 비디오에서 프레임 잡아서 저장

```
class Video(QMainWindow):
    def __init__(self) :
        super().__init__()
        self.setWindowTitle('비디오에서 프레임 수집') # 윈도우 이름과 위치 지정
        self.setGeometry(200,200,500,100)

        videoButton=QPushButton('비디오 켜기',self) # 버튼 생성
        captureButton=QPushButton('프레임 잡기',self)
        saveButton=QPushButton('프레임 저장',self)
        quitButton=QPushButton('나가기',self)

        videoButton.setGeometry(10,10,100,30)   # 버튼 위치와 크기 지정
        captureButton.setGeometry(110,10,100,30)
        saveButton.setGeometry(210,10,100,30)
        quitButton.setGeometry(310,10,100,30)

        videoButton.clicked.connect(self.videoFunction) # 콜백 함수 지정
        captureButton.clicked.connect(self.captureFunction)
        saveButton.clicked.connect(self.saveFunction)
        quitButton.clicked.connect(self.quitFunction)
```

```
    def videoFunction(self):
        self.cap=cv.VideoCapture(0,cv.CAP_DSHOW)  # 카메라와 연결 시도
        if not self.cap.isOpened(): self.close()

        while True:
            ret,self.frame=self.cap.read()
            if not ret: break
            cv.imshow('video display',self.frame)
            cv.waitKey(1)

    def captureFunction(self):
        self.capturedFrame=self.frame
        cv.imshow('Captured Frame',self.capturedFrame)

    def saveFunction(self):       # 파일 저장
        fname=QFileDialog.getSaveFileName(self,'파일 저장','./')
        cv.imwrite(fname[0],self.capturedFrame)

    def quitFunction(self):
        self.cap.release()        # 카메라와 연결을 끊음
        cv.destroyAllWindows()
        self.close()

app=QApplication(sys.argv)
win=Video()
win.show()
app.exec_()
```

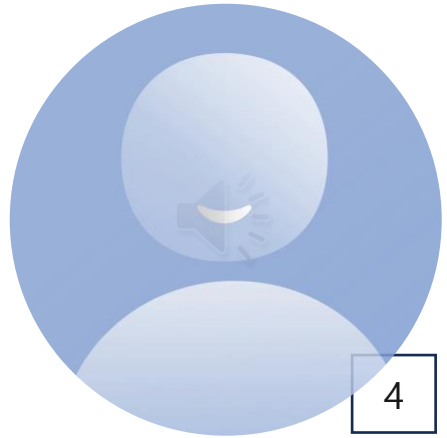
매칭

2. 이미지 크기 조정

```python
def resize_and_save(input_path, output_path):
    # 이미지 읽기
    original_image = cv2.imread(input_path)

    # 이미지 크기를 10분의 1로 조절
    new_width = original_image.shape[1] // 10
    new_height = original_image.shape[0] // 10
    resized_image = cv2.resize(original_image, (new_width, new_height), interpolation=cv2.INTER_AREA)

    # 크기가 조절된 이미지를 저장
    cv2.imwrite(output_path, resized_image)

# 이미지 파일들의 리스트
image_files = ["01.jpg", "02.jpg", "03.jpg","04.jpg"]

# 입력 폴더와 출력 폴더 설정
input_folder = "input_images"
output_folder = "output_images"

# 출력 폴더가 없으면 생성
os.makedirs(output_folder, exist_ok=True)

# 각 이미지에 대해 크기를 조절하고 저장
for image_file in image_files:
    input_path = os.path.join(input_folder, image_file)
    output_path = os.path.join(output_folder, f"resized_{image_file}")
    resize_and_save(input_path, output_path)
```

cv2.INTER_AREA

- 이미지 축소에 효과적!



매칭

3. 오림을 이용하여 흰색 배경을 가진 타겟 물체 만들기

```
class Orim(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle('오림')
        self.setGeometry(200, 200, 700, 200)

        fileButton = QPushButton('파일', self)
        paintButton = QPushButton('페인팅', self)
        cutButton = QPushButton('오림', self)
        incButton = QPushButton('+', self)
        decButton = QPushButton('-', self)
        saveButton = QPushButton('저장', self)
        quitButton = QPushButton('나가기', self)

        fileButton.setGeometry(10, 10, 100, 30)
        paintButton.setGeometry(110, 10, 100, 30)
        cutButton.setGeometry(210, 10, 100, 30)
        incButton.setGeometry(310, 10, 50, 30)
        decButton.setGeometry(360, 10, 50, 30)
        saveButton.setGeometry(410, 10, 100, 30)
        quitButton.setGeometry(510, 10, 100, 30)

        fileButton.clicked.connect(self.fileOpenFunction)
        paintButton.clicked.connect(self.paintFunction)
        cutButton.clicked.connect(self.cutFunction)
        incButton.clicked.connect(self.incFunction)
        decButton.clicked.connect(self.decFunction)
        saveButton.clicked.connect(self.saveFunction)
        quitButton.clicked.connect(self.quitFunction)

        self.BrushSiz = 5  # 페인팅 붓의 크기
        self.LColor, self.RColor = (255, 0, 0), (0, 0, 255)  # 파란색 물체, 빨간색 배경
```

```
def fileOpenFunction(self):
    fname = QFileDialog.getOpenFileName(self, 'Open file', './')
    self.img = cv.imread(fname[0])
    if self.img is None: sys.exit('파일을 찾을 수 없습니다.')

    self.img_show = np.copy(self.img)  # 표시용 영상
    cv.imshow('Painting', self.img_show)

    self.mask = np.zeros((self.img.shape[0], self.img.shape[1]), np.uint8)
    self.mask[:, :] = cv.GC_PR_BGD  # 모든 화소를 배경일 것 같음으로 초기화

def paintFunction(self):
    cv.setMouseCallback('Painting', self.painting)

def painting(self, event, x, y, flags, param):
    if event == cv.EVENT_LBUTTONDOWN:
        cv.circle(self.img_show, (x, y), self.BrushSiz, self.LColor, -1)  # 왼쪽 버튼을 클릭하면 파란색
        cv.circle(self.mask, (x, y), self.BrushSiz, cv.GC_FGD, -1)
    elif event == cv.EVENT_RBUTTONDOWN:
        cv.circle(self.img_show, (x, y), self.BrushSiz, self.RColor, -1)  # 오른쪽 버튼을 클릭하면 빨간색
        cv.circle(self.mask, (x, y), self.BrushSiz, cv.GC_BGD, -1)
    elif event == cv.EVENT_MOUSEMOVE and flags == cv.EVENT_FLAG_LBUTTON:
        cv.circle(self.img_show, (x, y), self.BrushSiz, self.LColor, -1)  # 왼쪽 버튼을 클릭하고 이동하면 파란색
        cv.circle(self.mask, (x, y), self.BrushSiz, cv.GC_FGD, -1)
    elif event == cv.EVENT_MOUSEMOVE and flags == cv.EVENT_FLAG_RBUTTON:
        cv.circle(self.img_show, (x, y), self.BrushSiz, self.RColor, -1)  # 오른쪽 버튼을 클릭하고 이동하면 빨간색
        cv.circle(self.mask, (x, y), self.BrushSiz, cv.GC_BGD, -1)

    cv.imshow('Painting', self.img_show)
```

매칭

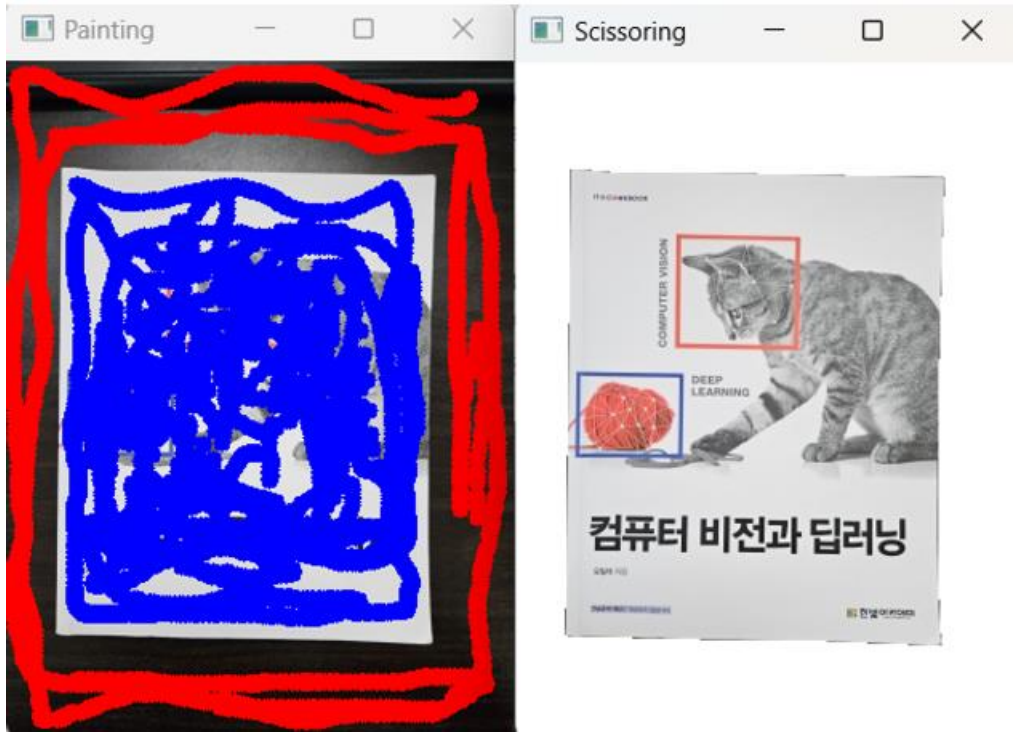
3. 오림을 이용하여 흰색 배경을 가진 타겟 물체 만들기

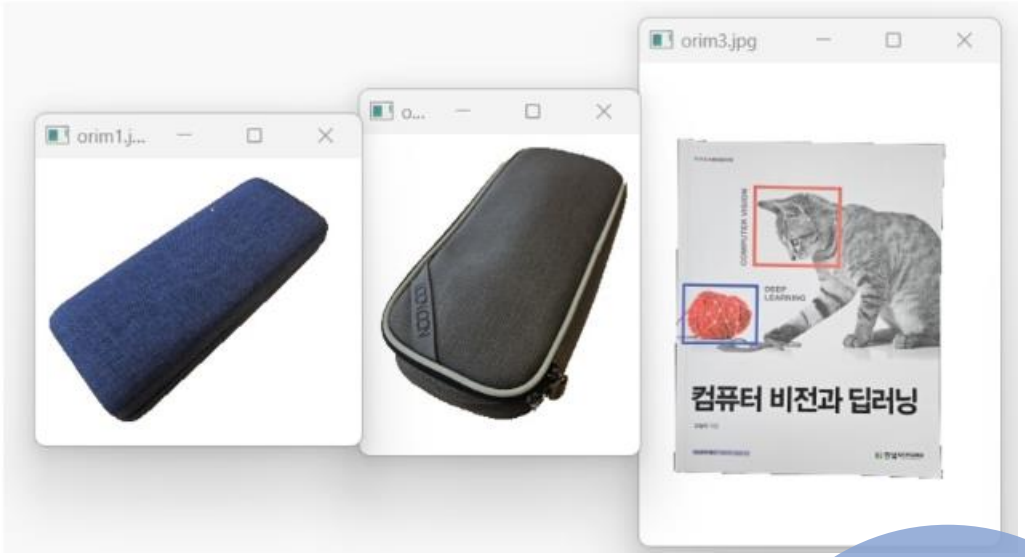
```
def cutFunction(self):
    background = np.zeros((1, 65), np.float64)
    foreground = np.zeros((1, 65), np.float64)
    cv.grabCut(self.img, self.mask, None, background, foreground, 5, cv.GC_INIT_WITH_MASK)
    mask2 = np.where((self.mask == 2) | (self.mask == 0), 0, 1).astype('uint8')
    self.grabImg = self.img * mask2[:, :, np.newaxis]

    self.grabImg[np.where((self.grabImg == [0, 0, 0]).all(axis=2))] = [255, 255, 255]

    cv.imshow('Scissoring', self.grabImg)

def incFunction(self):
    self.BrushSiz = min(20, self.BrushSiz + 1)

def decFunction(self):
    self.BrushSiz = max(1, self.BrushSiz - 1)

def saveFunction(self):
    fname = QFileDialog.getSaveFileName(self, '파일 저장', './')
    cv.imwrite(fname[0], self.grabImg)

def quitFunction(self):
    cv.destroyAllWindows()
    self.close()


app = QApplication(sys.argv)
win = Orim()
win.show()
app.exec_()
```

1. cv.grabCut(self.img, self.mask, None, background, foreground, 5, cv.GC_INIT_WITH_MASK)
   - 그랩컷 알고리즘을 사용하여 이미지를 세분화하고 제공된 마스크 및 초기화 진행
2. mask2 = np.where((self.mask == 2) | (self.mask == 0), 0, 1).astype('uint8')
   - 원본 마스크에서 값이 0 또는 2인 부분을 0으로 설정하고, 나머지를 1로 설정한 이진 마스크 생성
   - 0 : 배경, 1 : 전경 , 2 : 배경 초기화 되었지만 전경 확률 , 3 : 전경 초기화 되었지만 배경 확률
3. self.grabImg = self.img * mask2[:, :, np.newaxis]
   - 원본 이미지에 바이너리 마스크를 적용하여 세분화된 객체를 얻음
4. self.grabImg[np.where((self.grabImg == [0, 0, 0]).all(axis=2))] = [255, 255, 255]
   - 픽셀 값이 [0, 0, 0]인 경우 [255, 255, 255]로 대체하여 세분화된 객체 처리

매칭

3. 오림을 이용하여 흰색 배경을 가진 타겟 물체 만들기

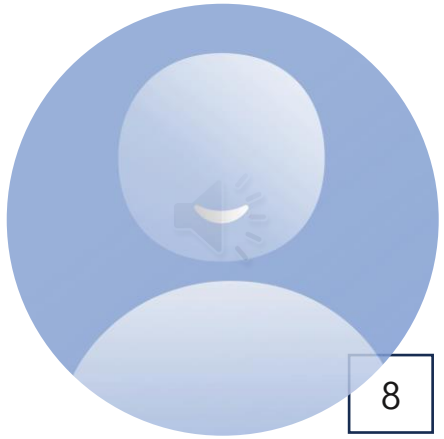




매칭

4. 물체 탐지

```
class SingleDetect(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle('물체 탐지')
        self.setGeometry(200, 200, 700, 200)

        signButton = QPushButton('타겟 영상 등록', self)
        imageButton = QPushButton('영상 불러옴', self)
        recognitionButton = QPushButton('인식', self)
        quitButton = QPushButton('나가기', self)
        self.label = QLabel('환영합니다!', self)

        signButton.setGeometry(10, 10, 100, 30)
        imageButton.setGeometry(110, 10, 100, 30)
        recognitionButton.setGeometry(210, 10, 100, 30)
        quitButton.setGeometry(510, 10, 100, 30)
        self.label.setGeometry(10, 40, 600, 170)

        signButton.clicked.connect(self.signFunction)
        imageButton.clicked.connect(self.imageFunction)
        recognitionButton.clicked.connect(self.recognitionFunction)
        quitButton.clicked.connect(self.quitFunction)

        self.signFiles = [['orim1.jpg', '안경케이스'], ['orim2.jpg', '필통케이스'], ['orim3.jpg', '책']]  # 타겟 영상
        self.signImgs = []  # 모델 영상 저장
```



매칭

4. 물체 탐지

```python
def recognitionFunction(self):
    if self.roadImg is None:
        self.label.setText('먼저 영상을 입력하세요.')
    else:
        sift = cv.SIFT_create()

        KD = []  # 여러 표지판 영상의 키포인트와 기술자 저장
        for img in self.signImgs:
            gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
            KD.append(sift.detectAndCompute(gray, None))

        grayRoad = cv.cvtColor(self.roadImg, cv.COLOR_BGR2GRAY)  # 명암으로 변환
        road_kp, road_des = sift.detectAndCompute(grayRoad, None)  # 키포인트와 기술자 추출

        matcher = cv.DescriptorMatcher_create(cv.DescriptorMatcher_FLANNBASED)
        GM = []  # 여러 표지판 영상의 good match를 저장
        for sign_kp, sign_des in KD:
            knn_match = matcher.knnMatch(sign_des, road_des, 2)
            T = 0.7
            good_match = []
            for nearest1, nearest2 in knn_match:
                if (nearest1.distance / nearest2.distance) < T:
                    good_match.append(nearest1)
            GM.append(good_match)

        best = GM.index(max(GM, key=len))  # 매칭 쌍 개수가 최대인 번호판 찾기
```

SIFT Descriptor 생성

SIFT **Keypoint**, **Descriptor** 추출

FLANN 기반 Descriptor matcher 생성

임계값을 이용하여 좋은 match 선별

매칭

4. 물체 탐지

```
if len(GM[best]) < 4:  # 최선의 물체가 매칭 쌍 4개 미만이면 실패
    self.label.setText('타겟물체가 없습니다.')
else:  # 성공(호모그래피 찾아 영상에 표시)
    sign_kp = KD[best][0]
    good_match = GM[best]

    points1 = np.float32([sign_kp[gm.queryIdx].pt for gm in good_match])
    points2 = np.float32([road_kp[gm.trainIdx].pt for gm in good_match])

    H, _ = cv.findHomography(points1, points2, cv.RANSAC)

    h1, w1 = self.signImgs[best].shape[0], self.signImgs[best].shape[1]  # 타겟 영상의 크기
    h2, w2 = self.roadImg.shape[0], self.roadImg.shape[1]  # 영상의 크기

    box1 = np.float32([[0, 0], [0, h1 - 1], [w1 - 1, h1 - 1], [w1 - 1, 0]]).reshape(4, 1, 2)
    box2 = cv.perspectiveTransform(box1, H)

    self.roadImg = cv.polylines(self.roadImg, [np.int32(box2)], True, (0, 255, 0), 4)

    img_match = np.empty((max(h1, h2), w1 + w2, 3), dtype=np.uint8)
    cv.drawMatches(self.signImgs[best], sign_kp, self.roadImg, road_kp, good_match, img_match,
                   flags=cv.DrawMatchesFlags_NOT_DRAW_SINGLE_POINTS)
    cv.imshow('Matches and Homography', img_match)

    self.label.setText(self.signFiles[best][1] + '을 잡아내었습니다.')
    winsound.Beep(3000, 500)
```

좋은 매치에 해당하는 keypoint 추출

호모그래피 행렬 계산(RANSAC 이용)

good_match와 호모그래피에 적용된 경계 상자를 사용하여 매칭 결과 그리기

매칭

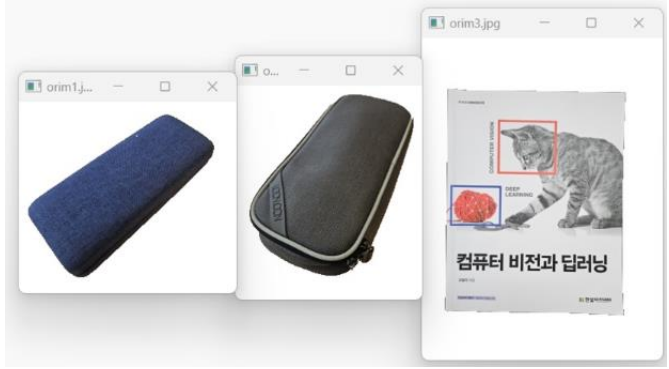
4. 물체 탐지

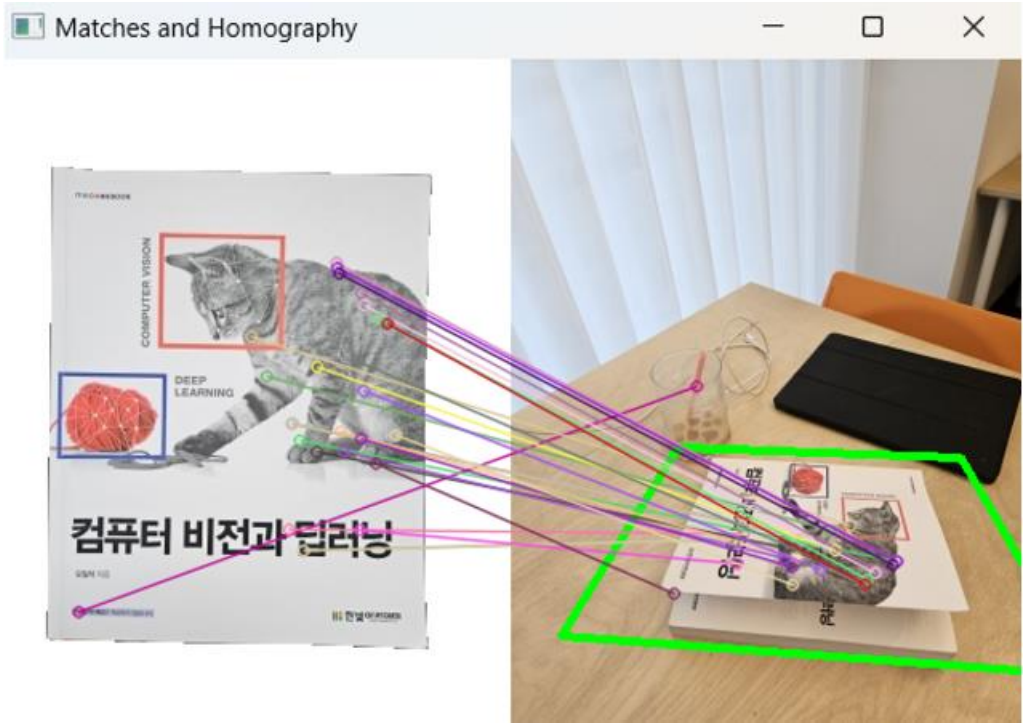
```
def signFunction(self):
    self.label.clear()
    self.label.setText('물체를 등록합니다.')

    for fname, _ in self.signFiles:
        self.signImgs.append(cv.imread(fname))
        cv.imshow(fname, self.signImgs[-1])

def imageFunction(self):
    if self.signImgs == []:
        self.label.setText('먼저 타겟 물체를 등록하세요.')
    else:
        fname = QFileDialog.getOpenFileName(self, '파일 읽기', './')
        self.roadImg = cv.imread(fname[0])
        if self.roadImg is None: sys.exit('파일을 찾을 수 없습니다.')

        cv.imshow('Road scene', self.roadImg)
```

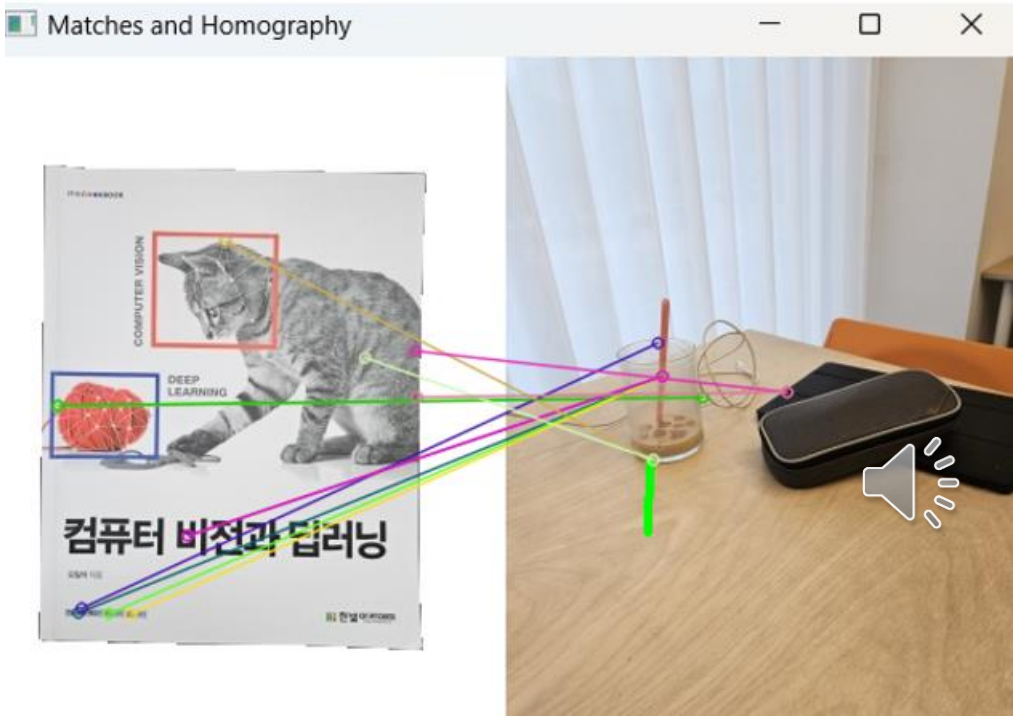
```
def quitFunction(self):
    cv.destroyAllWindows()
    self.close()


app = QApplication(sys.argv)
win = SingleDetect()
win.show()
app.exec_()
```

매칭

4. 물체 탐지


orim1.j...
o...
orim3.jpg
컴퓨터 비전과 딥러닝


Matches and Homography
COMPUTER VISION
DEEP LEARNING
컴퓨터 비전과 딥러닝


Matches and Homography
COMPUTER VISION
DEEP LEARNING
컴퓨터 비전과 딥러닝

파노라마

```
class Panorama(QMainWindow) :
    def __init__(self) :
        super().__init__()
        self.setWindowTitle('파노라마 영상')
        self.setGeometry(200,200,700,200)

        collectButton=QPushButton('영상 수집',self)
        self.showButton=QPushButton('영상 보기',self)
        self.stitchButton=QPushButton('봉합',self)
        self.saveButton=QPushButton('저장',self)
        quitButton=QPushButton('나가기',self)
        self.label=QLabel('환영합니다!',self)

        collectButton.setGeometry(10,25,100,30)
        self.showButton.setGeometry(110,25,100,30)
        self.stitchButton.setGeometry(210,25,100,30)
        self.saveButton.setGeometry(310,25,100,30)
        quitButton.setGeometry(450,25,100,30)
        self.label.setGeometry(10,70,600,170)

        self.showButton.setEnabled(False)
        self.stitchButton.setEnabled(False)
        self.saveButton.setEnabled(False)

        collectButton.clicked.connect(self.collectFunction)
        self.showButton.clicked.connect(self.showFunction)
        self.stitchButton.clicked.connect(self.stitchFunction)
        self.saveButton.clicked.connect(self.saveFunction)
        quitButton.clicked.connect(self.quitFunction)
```

```
    def collectFunction(self):
        self.showButton.setEnabled(False)
        self.stitchButton.setEnabled(False)
        self.saveButton.setEnabled(False)
        self.label.setText('c를 여러 번 눌러 수집하고 끝나면 q를 눌러 비디오를 끕니다.')

        self.cap=cv.VideoCapture(0,cv.CAP_DSHOW)
        if not self.cap.isOpened(): sys.exit('카메라 연결 실패')

        self.imgs=[]
        while True:
            ret,frame=self.cap.read()
            if not ret: break

            cv.imshow('video display', frame)

            key=cv.waitKey(1)
            if key==ord('c'):
                self.imgs.append(frame) # 영상 저장
            elif key==ord('q'):
                self.cap.release()
                cv.destroyWindow('video display')
                break

        if len(self.imgs)>=2:   # 수집한 영상이 2장 이상이면
            self.showButton.setEnabled(True)
            self.stitchButton.setEnabled(True)
            self.saveButton.setEnabled(True)
```

파노라마

```
    def showFunction(self):
        self.label.setText('수집된 영상은 '+str(len(self.imgs))+'장 입니다.')
        stack=cv.resize(self.imgs[0],dsize=(0,0),fx=0.25,fy=0.25)
        for i in range(1,len(self.imgs)):
            stack=np.hstack((stack,cv.resize(self.imgs[i],dsize=(0,0),fx=0.25,fy=0.25)))
        cv.imshow('Image collection',stack)

    def stitchFunction(self):
        stitcher=cv.Stitcher_create()
        status,self.img_stitched=stitcher.stitch(self.imgs)
        if status==cv.STITCHER_OK:
            cv.imshow('Image stitched panorama',self.img_stitched)
        else:
            winsound.Beep(3000,500)
            self.label.setText('파노라마 제작에 실패했습니다. 다시 시도하세요.')

    def saveFunction(self):
        fname=QFileDialog.getSaveFileName(self,'파일 저장','./')
        cv.imwrite(fname[0],self.img_stitched)

    def quitFunction(self):
        self.cap.release()
        cv.destroyAllWindows()
        self.close()

app=QApplication(sys.argv)
win=Panorama()
win.show()
app.exec_()
```

파노라마

특수 효과

```
class VideoSpecialEffect(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle('비디오 특수 효과')
        self.setGeometry(200,200,400,100)

        videoButton=QPushButton('비디오 시작',self)
        self.pickCombo=QComboBox(self)
        self.pickCombo.addItems(['엠보싱','카툰','연필 스케치(명암)','연필 스케치(컬러)','유화'])
        quitButton=QPushButton('나가기',self)

        videoButton.setGeometry(10,10,140,30)
        self.pickCombo.setGeometry(150,10,110,30)
        quitButton.setGeometry(280,10,100,30)

        videoButton.clicked.connect(self.videoSpecialEffectFunction)
        quitButton.clicked.connect(self.quitFunction)
```

특수 효과

```python
def videoSpecialEffectFunction(self):
    self.cap=cv.VideoCapture(0,cv.CAP_DSHOW)
    if not self.cap.isOpened(): sys.exit('카메라 연결 실패')

    while True:
        ret,frame=self.cap.read()
        if not ret: break

        pick_effect=self.pickCombo.currentIndex()
        if pick_effect==0:
            femboss=np.array([[-1.0, 0.0, 0.0],[0.0, 0.0, 0.0],[0.0, 0.0, 1.0]])
            gray=cv.cvtColor(frame,cv.COLOR_BGR2GRAY)
            gray16=np.int16(gray)
            special_img=np.uint8(np.clip(cv.filter2D(gray16,-1,femboss)+128,0,255))
        elif pick_effect==1:
            special_img=cv.stylization(frame,sigma_s=60,sigma_r=0.45)
        elif pick_effect==2:
            special_img,_=cv.pencilSketch(frame,sigma_s=60,sigma_r=0.07,shade_factor=0.02)
        elif pick_effect==3:
            _,special_img=cv.pencilSketch(frame,sigma_s=60,sigma_r=0.07,shade_factor=0.02)
        elif pick_effect==4:
            special_img=cv.xphoto.oilPainting(frame,10,1,cv.COLOR_BGR2Lab)

        cv.imshow('Special effect',special_img)
        cv.waitKey(1)
```

엠보싱

카툰

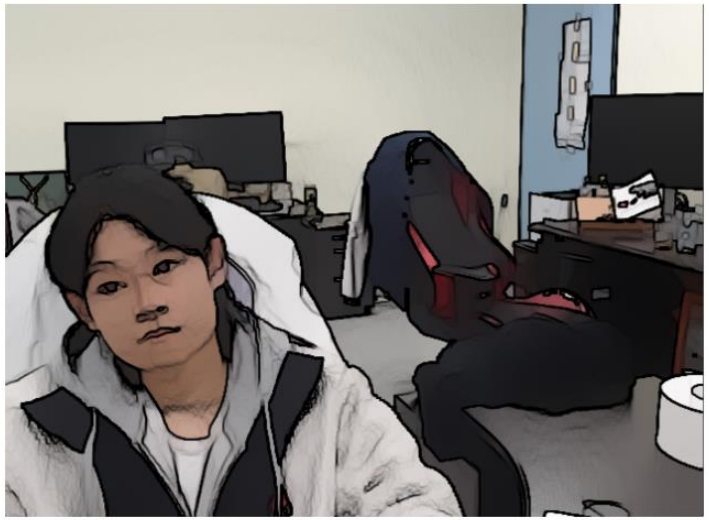
연필 스케치(명암)

연필 스케치(컬러)

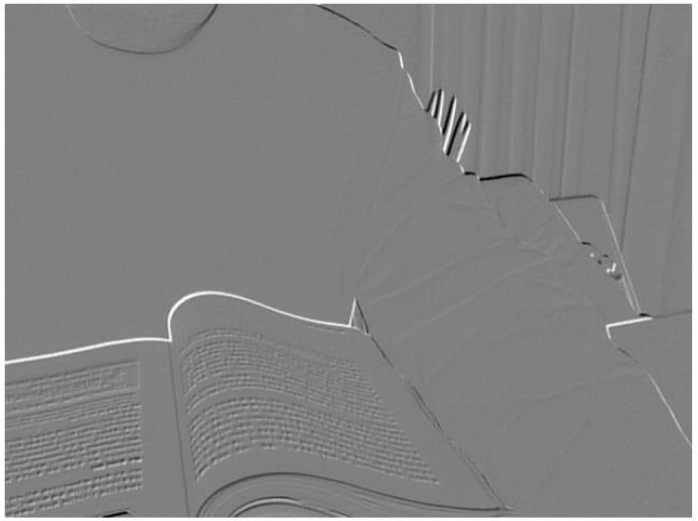
유화

```python
    def quitFunction(self):
        self.cap.release()
        cv.destroyAllWindows()
        self.close()

app=QApplication(sys.argv)
win=VideoSpecialEffect()
win.show()
app.exec_()
```

특수 효과

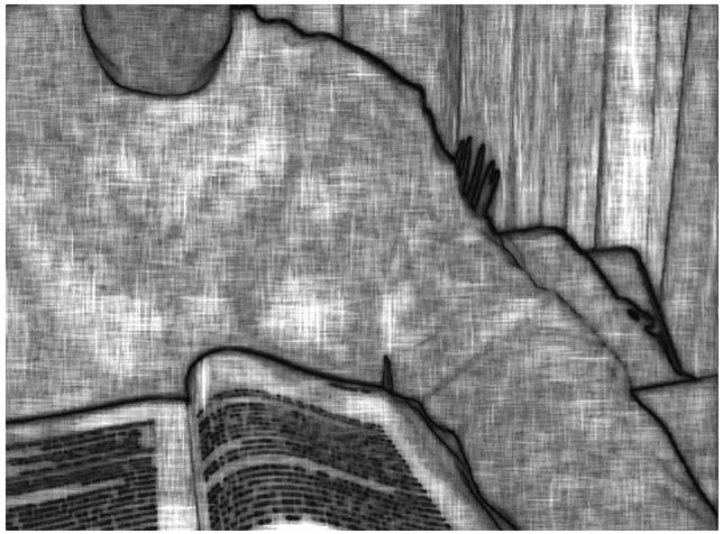


카툰



엠보싱

연필 스케치(컬러)



연필 스케치(흑백)

감사합니다.