[과제 3] 차선 검출

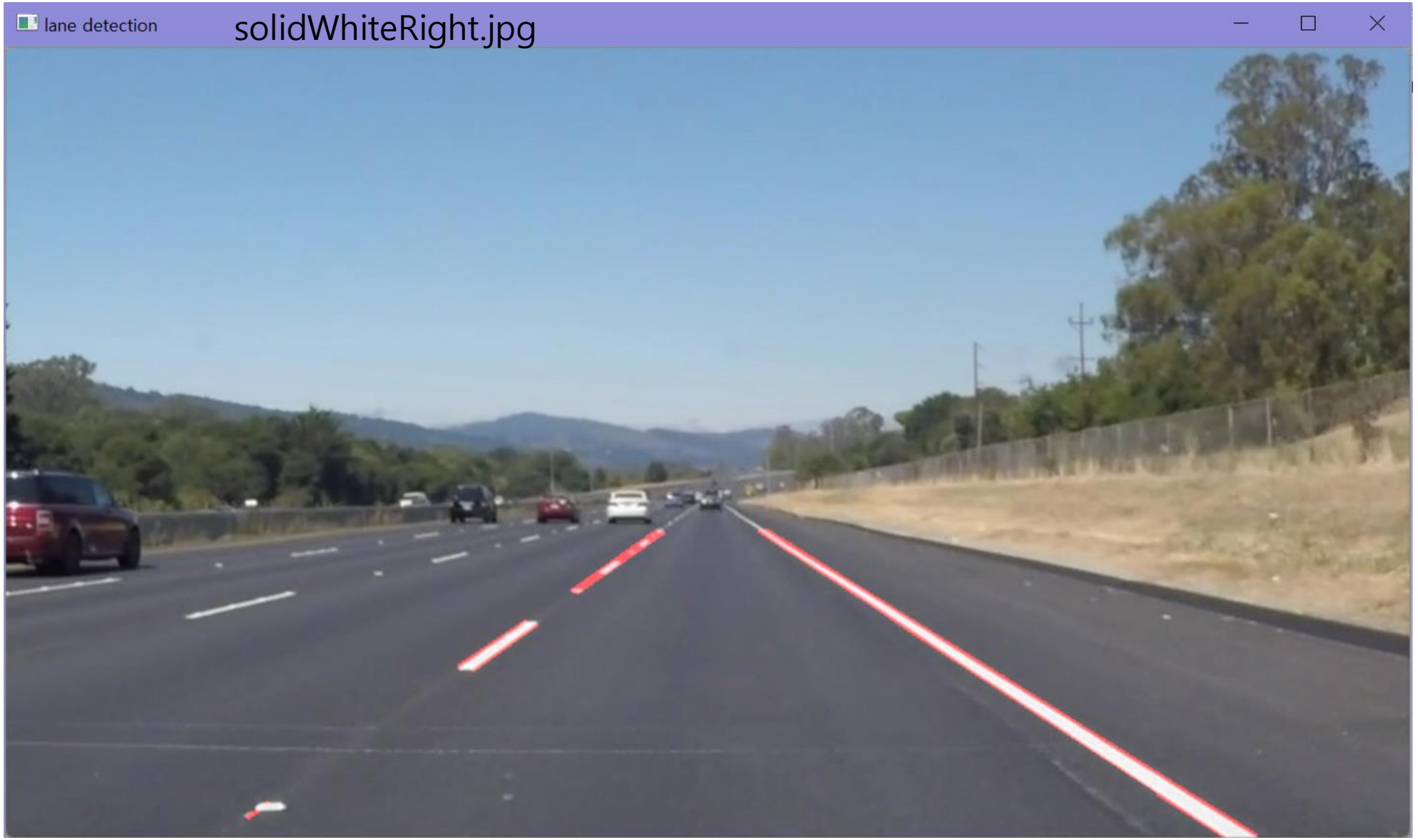
차선 검출영상

201810800 이혜인

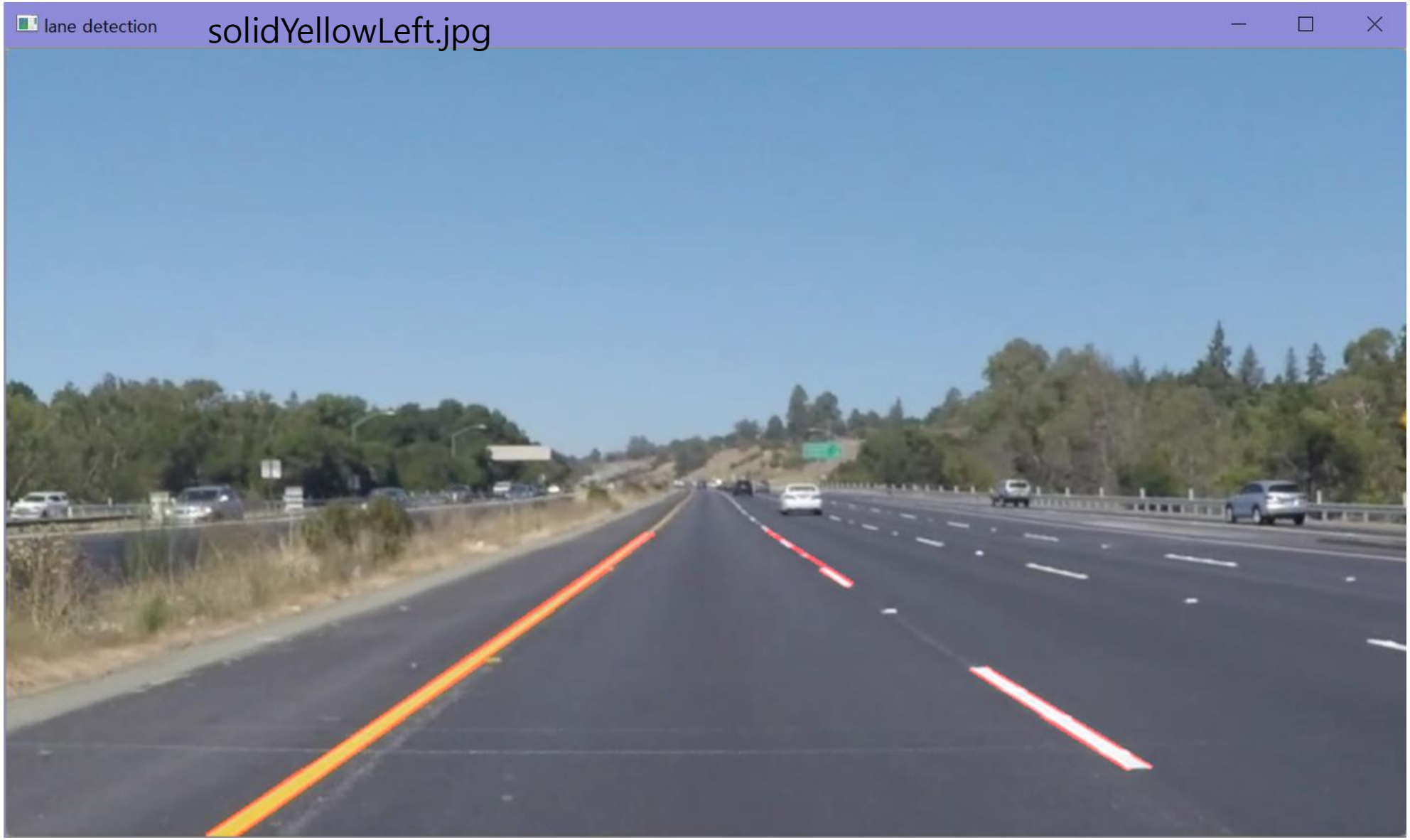
4) Write your own codes for Lane Detection

```
def hough_line(img, rho, theta, threshold, min_len, max_gap): # 허프 변환하기
    lines = cv2.HoughLinesP(img, rho, theta, threshold, np.array([]), minLineLength=min_len, maxLineGap=max_gap)
    #HoughLinesP함수를 사용해서 허프 변환
    line_img = np.zeros((img.shape[0], img.shape[1], 3), dtype = np.uint8)
    draw_line(line_img, lines)#선 그리기 함수 이용
    return line_img

def draw_line(img, lines, color = [0, 0, 255], thickness = 2): # 빨간색 선 그리기
    for line in lines:
        for x1,y1,x2,y2 in line:
            cv2.line(img, (x1, y1), (x2, y2), color, thickness)#범위 내에 라인 그리기

gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)#영상을 흑백영상으로 만들기
img_canny = cv2.Canny(gray_img, 100, 200)#Canny를 이용해서 에지 검출
roi_img = region_of_interest(img_canny, vertices)#지정범위 내에 에지만 검출하기 위해 범위 설정
hough_img = hough_line(roi_img, 1, 1 * np.pi/180, 30, 10, 20)#허프 변환을 통해 라인 그리기
result = cv2.addWeighted(hough_img, 1, img, 1.0, 0.0)#원본 영상에 라인 겹치기

return result
```

4) Write your own codes for Lane Detection



4) Write your own codes for Lane Detection



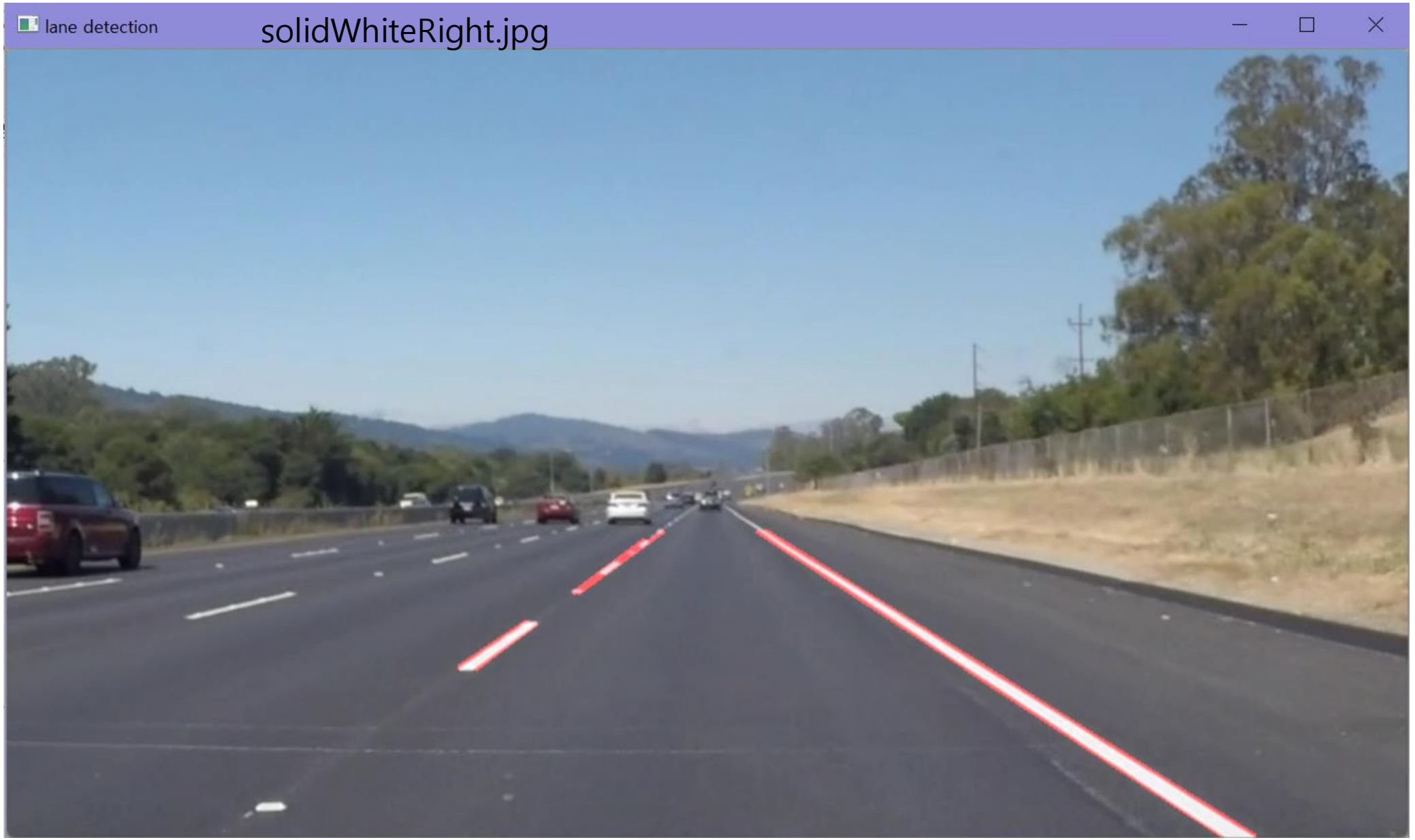
※ Further improvements

```
def hough_line(img, rho, theta, threshold, min_len, max_gap): # 허프 변환하기
    lines = cv2.HoughLinesP(img, rho, theta, threshold, np.array([]), minLineLength=min_len, maxLineGap=max_gap)
    #HoughLinesP함수를 사용해서 허프 변환
    return lines

def draw_line(img, lines, color = [0, 0, 255], thickness = 2): # 빨간색 선 그리기
    for line in lines:
        for x1,y1,x2,y2 in line:
            cv2.line(img, (x1, y1), (x2, y2), color, thickness)#범위 내에 라인 그리기

gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)#영상을 흑백영상으로 만들기
blur_img = cv2.GaussianBlur(gray_img, (3, 3), 0)#GaussianBlur를 통해서 영상 블러처리
img_canny = cv2.Canny(blur_img, 100, 200)#Canny를 이용해서 에지 검출
roi_img = region_of_interest(img_canny, vertices)#지정범위 내에 에지만 검출하기 위해 범위 설정
hough_img = hough_line(roi_img, 1, 1 * np.pi/180, 30, 10, 20)#허프 변환을 통해 라인 그리기

hough_img = np.squeeze(hough_img)#hough_img를 np.squeeze를 통해 배열에서 1차원 항목제거
slope = (np.arctan2(hough_img[:,1] - hough_img[:,3], hough_img[:,0] - hough_img[:,2]) * 180) / np.pi# 기울기 구하기
```

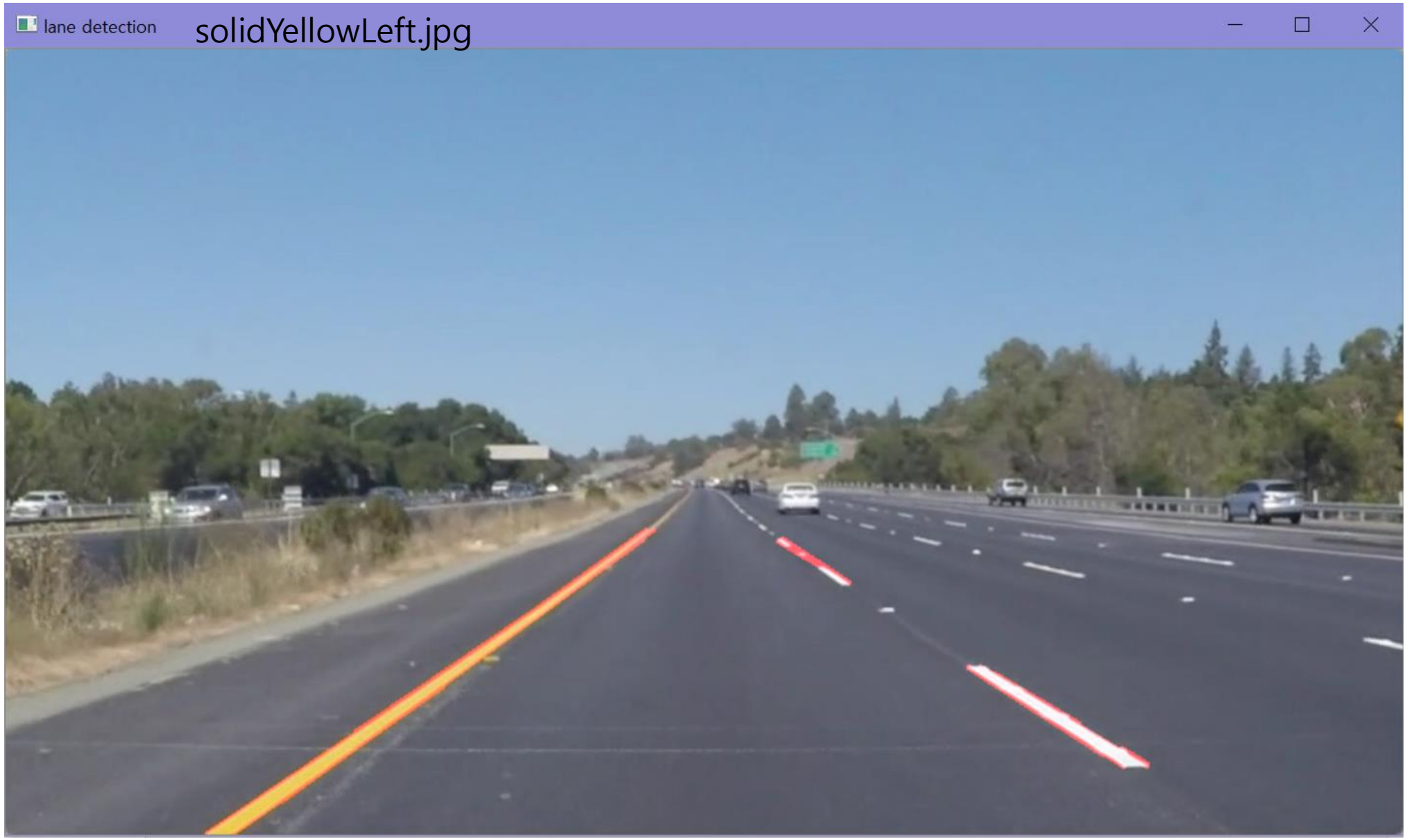
※ Further improvements

```
# 수평 기울기 제한
  hough_img = hough_img[np.abs(slope)<150]#150미만으로 기울기 제한
  slope = slope[np.abs(slope)<150]

  # 수직 기울기 제한
  hough_img = hough_img[np.abs(slope)>85]#85초과로 기울기 제한
  slope = slope[np.abs(slope)>85]

  # 필터링된 직선 버리기(왼쪽, 오른쪽 나눠서 진행)
  left = hough_img[(slope>0),:]
  right = hough_img[(slope<0),:]
  temp = np.zeros((img.shape[0], img.shape[1], 3), dtype=np.uint8)
  left = left[:,None]
  right = right[:,None]

  # 직선 그리기(왼쪽, 오른쪽 각각 그리기)
  draw_line(temp, left)
  draw_line(temp, right)
  result = cv2.addWeighted(temp, 1, img, 1.0, 0.0)#원본 영상에 라인 겹치기
  return result
```

※ Further improvements



※ Further improvements



※ Further improvements