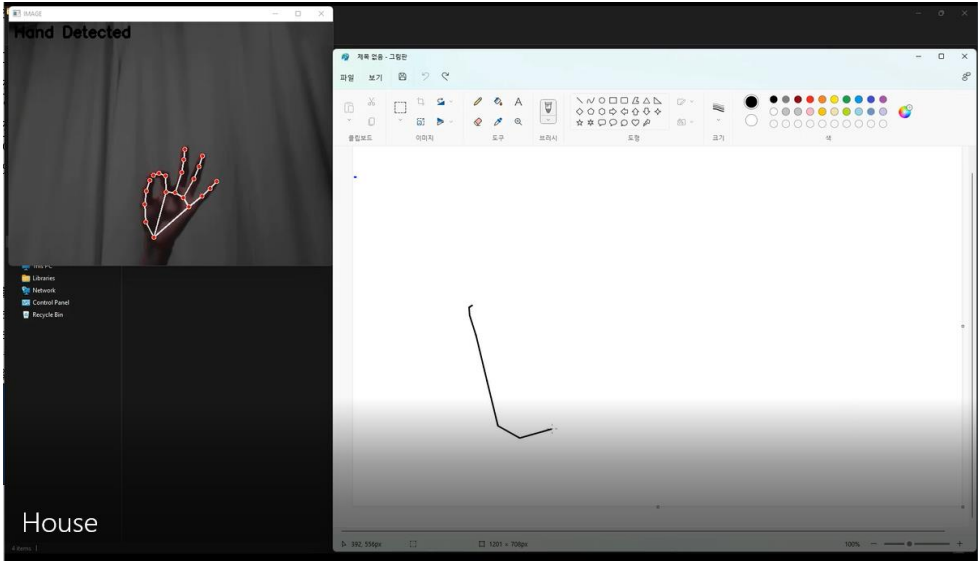
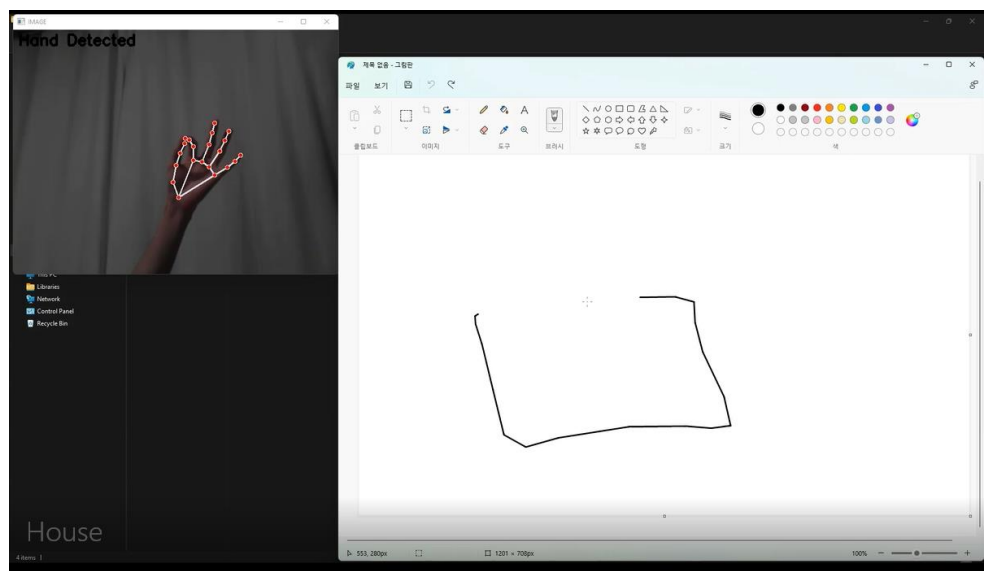
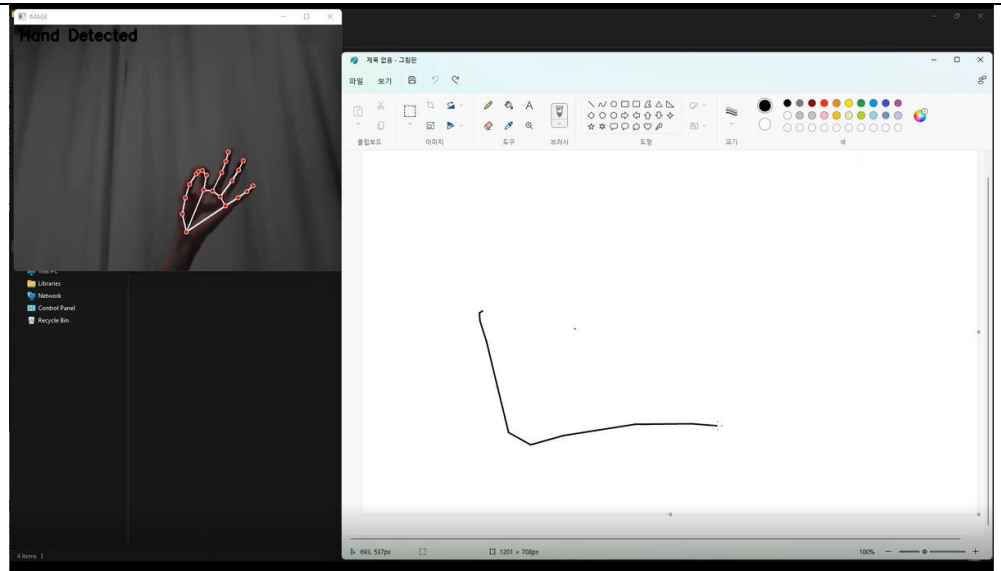


캡스톤 디자인 11월 1주차 보고서

주제명	제스처인식 기반 커서제어		
4조	박세준, 임준혁, 이재운	기간	2021.10.31~2021.11.06

기본 계획	제스처 인식을 하기 위한 코드를 작성하고, 마우스 이벤트에 관한 제스처 설정 구현
금주 계획	영상 인식을 위한 코드를 작성하고, 커서 이동 이벤트를 처리할 제스처를 설정하고, 인식을 하여 커서를 움직일 수 있는 기능 구현
금주의 수행 내용 및 이슈 이슈 해결 방 안	<p>커서 이동 제스처를 인식하고, 커서가 사용자 의도대로 움직이게끔 설정을 하였다.</p> 



[문제점]

커서를 화면에 끝 부분에 가져갈 경우, 영상에서 손이 벗어나 커서를 이동하는 경우에 다시 손을 영상에 보여준 뒤, 커서를 움직이는 불편함이 있었는데, 실제 화면에 크기를 영상에 설정한 부분으로 계산하여, 영상 안에서 손이 움직이면 화면에 원하는 곳 어디든 커서를 가져갈 수 있게 설정을 하였다.

```

41 camWidth, camHeight =
    cap.get(cv2.CAP_PROP_FRAME_WIDTH),
43         cap.get(cv2.CAP_PROP_FRAME_HEIGHT)
44 winWidth, winHeight = pg.size()
45 curX, curY = pg.position()
   ratio = 0.7
  
```

```

70 curX[1] = (((curX[1] - 0.5) / ratio) + 0.5) * winWidth
71 curY[1] = (((curY[1] - 0.5) / ratio) + 0.5) *
72 winHeight
pg.moveTo(int(curX[1]), int(curY[1]))

```

그리고 커서 이동을 하기 위해서, 검지의 끝 마디를 인식하고, 움직이는 과정에서 커서가 부드럽게 움직이지 않고, 자꾸 튕는 모습을 보였는데, 이를 개선하기 위해서 손의 검지 포인트가 아니라 인식되는 손의 21개의 좌표의 평균을 구하여 움직이도록 하였다.

```

31 def arithMean(landmarks):
32     sum_x = 0
33     sum_y = 0
34     for landmark in landmarks:
35         sum_x += landmark.x
36         sum_y += landmark.y
37     return sum_x / 21, sum_y / 21

```

```

69 curX, curY = arithMean(hand_landmarks.landmark)
70 curX[1] = (((curX[1] - 0.5) / ratio) + 0.5) * winWidth
71 curY[1] = (((curY[1] - 0.5) / ratio) + 0.5) *
72 winHeight
pg.moveTo(int(curX[1]), int(curY[1]))

```

차주 수행 계획	1. 좌클릭, 우클릭, 휠클릭 이벤트 사용 가능하게 구현 2. 키보드 이벤트를 수행할 제스처 모델 만들기
참고 문헌	https://github.com/google/mediapipe https://deep-learning-study.tistory.com/99 https://github.com/opencv/opencv/tree/4.5.4