입회시험(결과)성적서

시험목적 : 알고리즘의 방법 상의 적합성 검증 및 산출된 결과의 적합성 검증

시험내용 : 1) 충격전 임계값 기반 추락 검출 알고리즘 검증 (충격 전에 추락동작을 검출함)

가) 측정된 데이터 불러오기
나) 데이터 필터링
다) 특성값 연산 (각도, 수직속도)
라) 임계값 최적화 및 테스트 데이터셋 분리
마) 추락 검출 알고리즘
바) 결과값 산출 (추락 검출 알고리즘 정확도, secure time)

2) 충격후 머신러닝 기반 추락/전도 검출 알고리즘 검증
(충격 후에 일반작업동작과 추락/전도동작을 분류함)
가) 측정된 데이터 불러오기
나) 데이터 필터링
다) 특성값 연산 (3축 가속도 및 3축 각속도)
라) 훈련 및 테스트 데이터 분리
마) 딥러닝 알고리즘
아) 분류기 훈련 및 평가 (추락/전도 검출 알고리즘 정확도)

<시제품 관성센서>

<시제품 착용 사진>

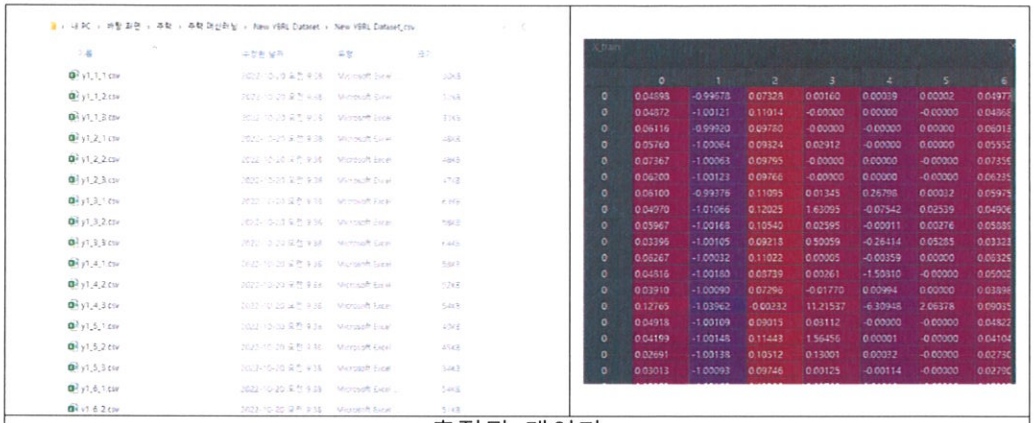
<실험 동작>

Class	Name			
일반작업동작	D01	쭈그려 앉기	D09	빔 위 걷기 (편도)
	D02	완전히 앉기	D10	짐들고 빔 위 걷기 (편도)
	D03	계단 오르내리기	D11	삽질
	D04	사다리 타기	D12	스트래칭
	D05	곡괭이질	D13	비계 오르기
	D06	짐들기 (앞)	D14	점프 후 착지
	D07	짐들기 (옆)	D15	제자리 뛰기
	D08	짐들기 (뒤)	D16	앉았다 일어나기
			D17	의자 위에 빠르게 올라갔다 내려오기
전도동작	LF01	걸어가다 걸려 전도 (앞)	LF03	걸어가다 미끄러져 전도 (뒤)
	LF02	걸어가다 걸려 전도 (옆)	LF04	걸어가다 미끄러져 전도 (앞)
			LF05	기절
추락동작	HF01	추락	HF02	사다리에서 후방 회전 추락

시험결과 : 1) 충격전 임계값 기반 추락 검출 알고리즘 검증

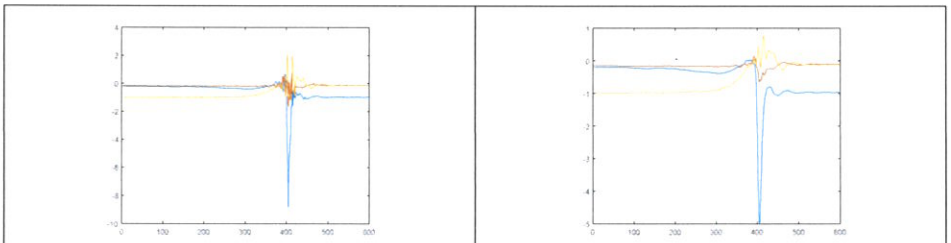
가) 측정된 데이터 불러오기

- 데이터셋: 10명



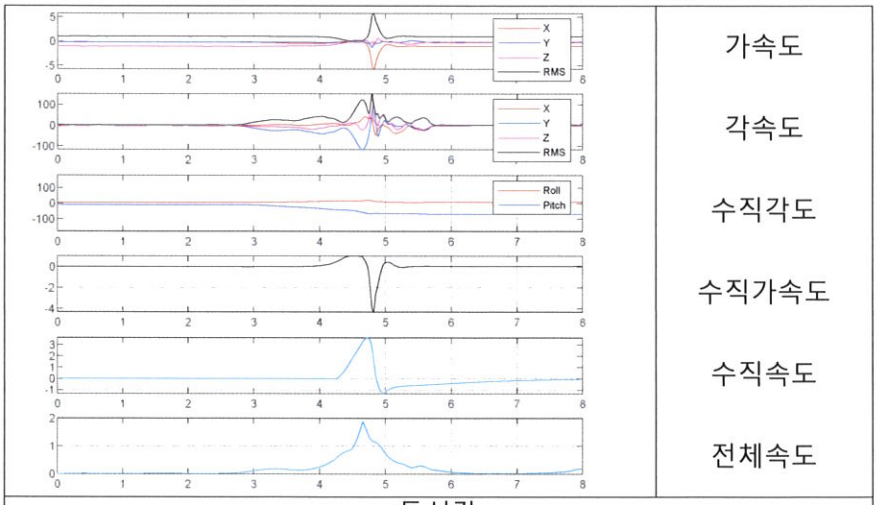
<측정된 데이터>

나) 데이터 필터링



<필터링 전> <필터링 후>

다) 특성값 연산 (각도, 수직속도)



<특성값>

라) 임계값 최적화 및 테스트 데이터셋 분리

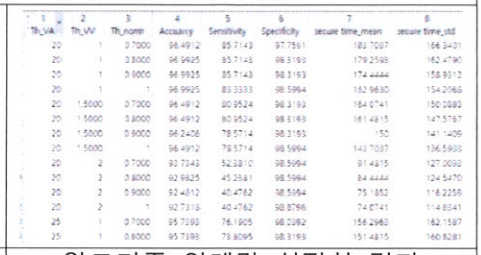
```
Subject_list = list(range(1,11))
random.seed(1)
test_set = random.sample(Subject_list,3)
print(test_set)

% c = [1, 4, 6, 7, 8, 9, 10]   % 임계값 설정 피험자 번호
% c = [2, 3, 5]    % 임계값 검증 피험자 번호
```

<임계값 최적화 및 테스트 데이터셋 분리 코드>

```
for Th_VA = 20:5:30
    for Th_VV = 1.0:0.5:2.0
        for Th_norm = 0.7:0.1:1.0
```

<알고리즘 임계값 최적화 코드>



Th_VA	Th_VV	Th_norm	Accuracy	Sensitivity	Specificity	secure time_mean	secure time_std
20	1	0.7000	96.4912	85.7143	97.7561	183.7037	166.3431
20	1	0.8000	96.9825	85.7143	98.3193	179.2593	162.4790
20	1	0.9000	96.9825	85.7143	98.3193	174.4444	158.9313
20	1	1	96.9825	83.3333	98.5994	162.9630	154.2068
20	1.5000	0.7000	96.4912	80.9524	98.3193	164.0741	150.0880
20	1.5000	0.8000	96.4912	80.9524	98.3193	161.4815	147.5767
20	1.5000	0.9000	96.2406	78.5714	98.3193	153	141.1406
20	1.5000	1	96.4912	78.5714	98.5994	143.7037	136.5938
20	2	0.7000	93.7343	52.3810	98.5994	91.4815	127.0093
20	2	0.8000	92.9825	45.2381	98.5994	84.4444	124.5470
20	2	0.9000	92.4812	40.4762	98.5994	75.1852	118.2258
20	2	1	92.7318	40.4762	98.8796	74.0741	114.8341
25	1	0.7000	95.7393	76.1905	98.0392	156.2963	162.1587
25	1	0.8000	95.7393	73.8095	98.3193	151.4815	160.8281

<알고리즘 임계값 최적화 결과>

라) 추락검출 알고리즘

```
for i = 1 : 1 : length(Vel_V)   % 추락 감지 알고리즘
    if( (Vel_V(i) > Th_VV && (abs(ThetaSaved_Comp(i)) > Th_VA || abs(PhiSaved_Comp(i)) > Th_VA)) || Vel_V(i) > 4.5)
        if( norm_deb(i) >= 0.9)
            R_indu(c,x,y) = 1;
            Time_indu(c,x,y) = i;
            break;
        end
    end
end

if ( R_indu(c,x,y) == 1 && x == 10 )  % 추락인데 추락이라고 감지
    TP(th_idx) = TP(th_idx) +1;
end
if ( R_indu(c,x,y) == 1 && x ~= 10 )  % 추락이 아닌데 추락이라고 감지
    FP(th_idx) = FP(th_idx) +1;
end
if ( R_indu(c,x,y) == 0 && x ~= 10 )  % 추락이 아닌데 추락이 아니라고 감지
    TN(th_idx) = TN(th_idx) +1;
end
if ( R_indu(c,x,y) == 0 && x == 10 )  % 추락인데 추락이 아니라고 감지
    FN(th_idx) = FN(th_idx) +1;
end
```

<충격전 임계값 기반 추락 검출 알고리즘 코드>

마) 결과값 산출 (민감도, 특이도, 정확도, secure time)

```
'추락 검출 알고리즘 민감도: 100%
 추락 검출 알고리즘 특이도: 100%
 추락 검출 알고리즘 정확도: 100%
 Secure time: 354.44+-94.26 ms
```

추락 검출 민감도: 18/18*100 = 100.00%

추락 검출 특이도: 151/151*100 = 100.00%

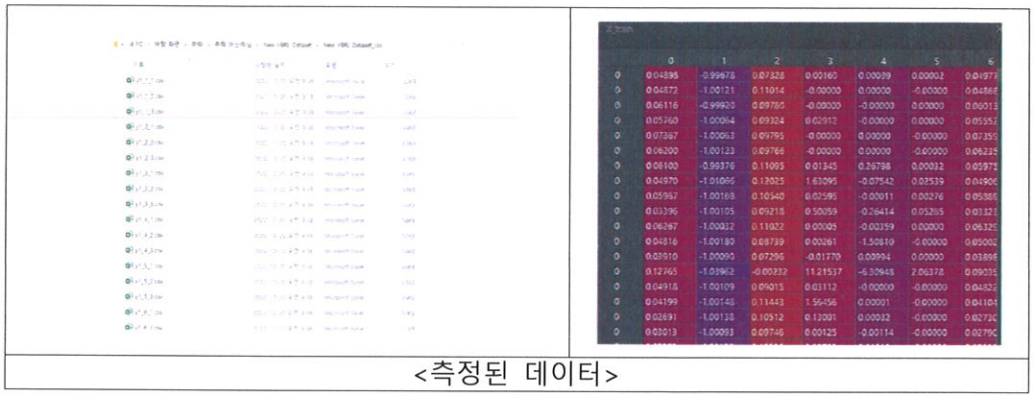
추락 검출 정확도: 169/169*100 = 100.00%

Secure time: 354.44 ± 94.26 ms

<충격전 임계값 기반 추락 검출 알고리즘 정확도 결과>

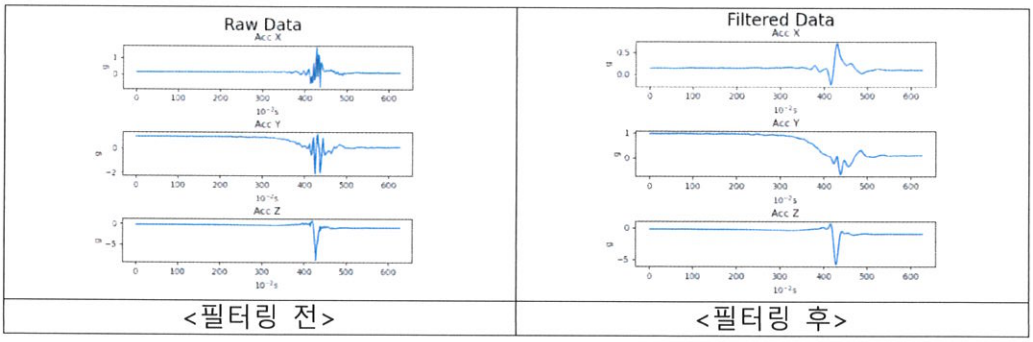
2) 딥러닝 기반 추락/전도 검출 알고리즘 검증

가) 측정된 데이터 불러오기



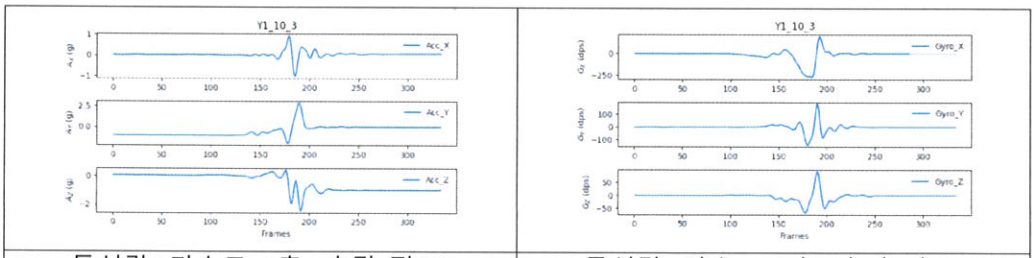
<측정된 데이터>

나) 데이터 필터링



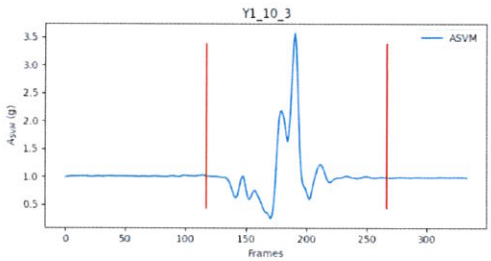
<필터링 전> <필터링 후>

다) 특성값 연산 (3축 가속도 및 3축 각속도)



<특성값: 가속도 3축 (추락/전도)>

<특성값: 각속도 3축 (추락/전도)>



<Windowing: 추락/전도 데이터 (150 frames)>

라) 훈련 및 테스트 데이터 분리

```
if Subject <= 8:
    X_train = X_train.append(pd.DataFrame(Tmp4).T)
elif Subject > 8:
    X_test = X_test.append(pd.DataFrame(Tmp4).T)

X_train_input = np.array(X_train.iloc[:,0:900])
Y_train_2class = np.array(X_train.iloc[:,904:])

X_test_input = np.array(X_test.iloc[:,0:900])
Y_test_2class = np.array(X_test.iloc[:,904:])
```

```
X_test = {DataFrame: (145, 905)} 0  1
X_test_input = {ndarray: (145, 900)} [[ 1.8266451
X_train = {DataFrame: (576, 905)} 0  1
X_train_input = {ndarray: (576, 900)} [[ 4.897748
Y_test_2class = {DataFrame: (145, 1)} 904 [0  1.0
Y_train_2class = {ndarray: (576, 1)} [[0.], [0.], [0.],
```

<훈련 및 테스트 데이터 분리 코드>

마) 딥러닝 알고리즘

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(4, 4), strides=(2, 2), padding='same',
                 activation='relu',
                 input_shape=(150, 6, 1)))
model.add(MaxPooling2D(padding='SAME'))
model.add(Conv2D(32, (2, 2), activation='relu',
                 padding='same'))
model.add(MaxPooling2D(padding='SAME'))
model.add(Conv2D(64, (2, 2), activation='relu',
                 padding='same'))
model.add(MaxPooling2D(padding='SAME'))
model.add(Flatten())
model.add(Dense(50, activation='relu'))
model.add(Dense(2, activation='softmax'))

model.compile(optimizer=adam, loss=categorical_crossentropy, metrics=METRICS)
model.summary()
```

<CNN 기반 딥러닝 코드>

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_22 (Conv2D)          (None, 75, 3, 32)         544

 max_pooling2d_13 (MaxPoolin  (None, 38, 2, 32)        0
 g2D)

 conv2d_23 (Conv2D)          (None, 38, 2, 32)         4128

 max_pooling2d_14 (MaxPoolin  (None, 19, 1, 32)        0
 g2D)

 conv2d_24 (Conv2D)          (None, 19, 1, 64)         8256

 max_pooling2d_15 (MaxPoolin  (None, 10, 1, 64)        0
 g2D)

 flatten_3 (Flatten)         (None, 640)               0

 dense_8 (Dense)             (None, 50)                32050

 dense_9 (Dense)             (None, 2)                 102

=================================================================
Total params: 45,080
Trainable params: 45,080
Non-trainable params: 0
_________________________________________________________________
```

<CNN 기반 모델 요약>

바) 분류기 훈련 및 평가 (추락/전도 검출 알고리즘 정확도)

Result

	0	1
0	1.00000	0.00000
1	1.00000	0.00000
2	1.00000	0.00000
3	1.00000	0.00000
4	1.00000	0.00000
5	1.00000	0.00000
6	1.00000	0.00000
7	1.00000	0.00000
8	1.00000	0.00000
9	1.00000	0.00000
10	1.00000	0.00000
11	1.00000	0.00000
12	1.00000	0.00000
13	1.00000	0.00000
14	1.00000	0.00000

<Softmax 예측 결과>

```
Y_test_2class = np.array(Y_test_2class)
pred_data = np.array(pred_data)

FN_name = []
for i in range(1,len(X_test_input)+1):
    if Y_test_2class[i-1] == 0:
        if pred_data[i-1] == 0:
            TN += 1
        elif pred_data[i-1] == 1:
            FP += 1
    elif Y_test_2class[i-1] == 1:
        if pred_data[i-1] == 0:
            FN += 1
        elif pred_data[i-1] == 1:
            TP += 1

Acc = (TP + TN) / (TP + TN + FP + FN)
Sen = (TP) / (TP + FN)
Sep = (TN) / (TN + FP)
print(Acc, Sen, Sep)
```

<충격 후 추락/전도 검출 알고리즘 정확도 계산 코드>

```
추락/전도 검출 알고리즘 민감도: 100.0 %
추락/전도 검출 알고리즘 특이도: 100.0 %
추락/전도 검출 알고리즘 정확도: 100.0 %
```

추락/전도 검출 민감도: 42/42*100 = 100.00%
추락/전도 검출 특이도: 102/102*100 = 100.00%
추락/전도 검출 정확도: 144/144*100 = 100.00%

<충격후 딥러닝 기반 추락/전도 검출 알고리즘 정확도 결과>

확인자 : 최 기 원 (서명)

입회 시험 사진