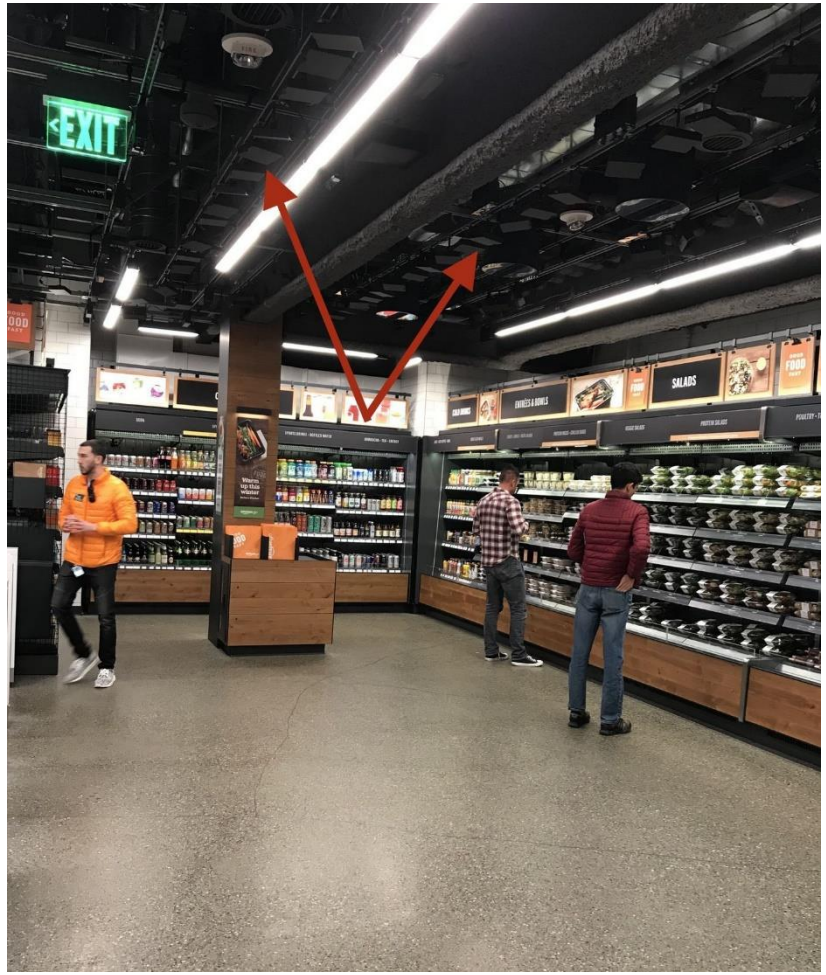
TDOA CONV-LSTM MODEL(Indoor Tracking)

22510108 이성호

목차 Contents

Introduction



- IT 기술의 발달로 무인 시스템은 점차 많은 분야에서 활용되고 있다. 특히 아마존 고와 같은 무인 매장에서는 고객이 물건을 장바구니에 넣고 체크아웃 과정 없이 바로 출구로 나가면 구매한 상품이 확인돼 자동으로 결제가 이뤄진다.

- 기존의 여러 연구는 정확한 TDOA 값을 수학적으로 추론하고 쌍곡선 방정식을 사용하여 위치를 추적했다.

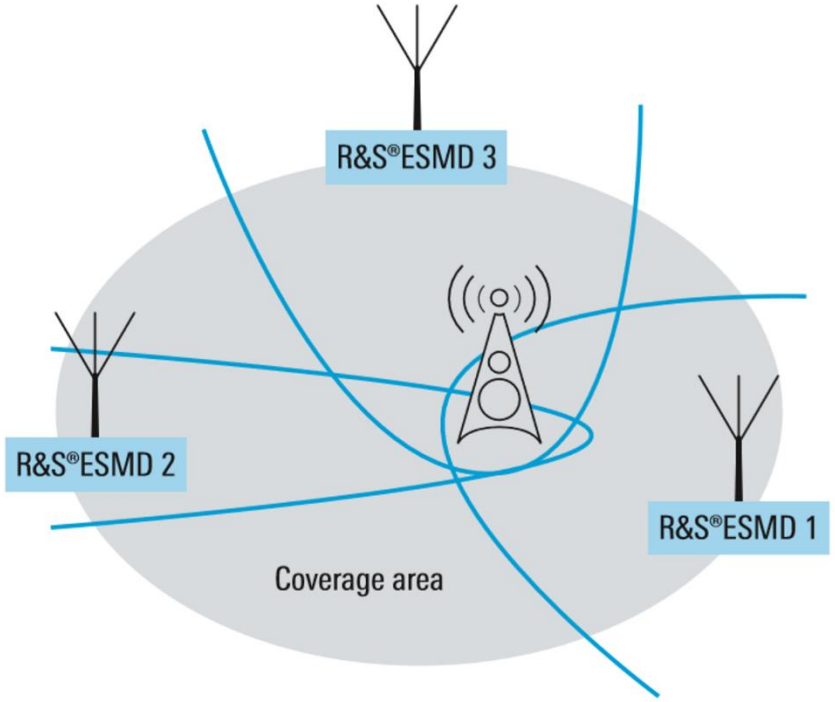
- 쌍곡선 방정식의 교점을 찾기 위해서는 TDOA를 통한 위치 추정 방법이 필수적이지만, 이 과정은 복잡한 비선형 방정식을 풀기 위해서는 많은 컴퓨터 자원이 필요하다.

- 이러한 문제를 해결하기 위해 복잡한 실내 공간에서 추출된 노이즈가 많은 TDOA 값을 보정할 수 있는 모델이 필요로 한다.

Domain Knowledge - ToA

- 태그(슈카)에서 쏜 신호를 앵커(인공위성)가 측정하기 까지의 시간을 이용해서 계산하는 방식이다 (v=d/t에서 속도 v는 빛 혹은 전파의 속도는 c로 일정하기 때문에 시간 t만 재면 두 점 사이의 거리를 잴 수 있다.)
- 인공위성(앵커)는 최소 3개가 필요로 하며 정확한 시간 동기화 필요하다.
- 기술 발전(PHY Layer, 반도체의 난수 생성 및 이용 등)으로 인해 최근 Wi-F를 대체하여 디지털 키 같은 IOT 산업, 스포츠 (이동 거리)에서 다시 사용하려는 트렌드이다.

Domain Knowledge - TDoA

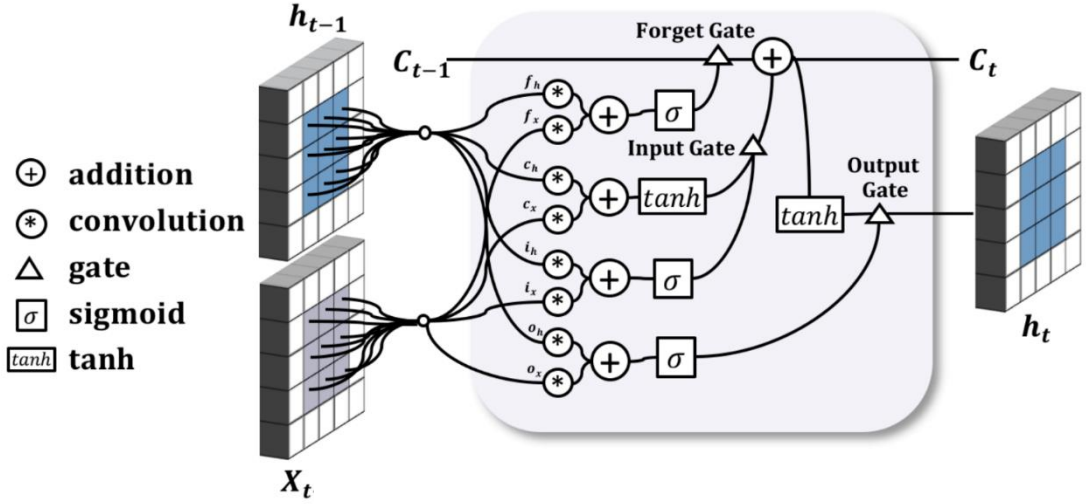


- 이용자가 속한 기지국 신호와 인접 기지국으로 부터의 신호 도달 시간(ToA)의 차이를 이용하는 기술
- 최소 3개의 쌍곡선들의 교점을 통해 사용자의 위치 파악
- TDOA 장점
  1. 태그는 앵커와 개별적으로 통신하지 않으며, 신호만 전송하여 배터리 소모가 적다.

  2. 태그와 앵커 사이에 이전 주소 바인딩이 없으므로 동일 공간의 앵커의 수의 확장이 가능

  3. 태그는 신호를 보내기 위해 짧은 시간만 사용하므로 많은 수의 태그가 한 번의 재생 빈도로 신호를 전송하기에 동시에 처리할 수 있는 태그의 수가 매우 많다

Domain Knowledge – Convolutional LSTM



$$i_t = \sigma\left(W_{xi} * X_t + W_{hi} * h_{t-1} + b_i\right)$$
$$f_t = \sigma\left(W_{xf} * X_t + W_{hf} * h_{t-1} + b_f\right)$$
$$o_t = \sigma\left(W_{xo} * X_t + W_{ho} * h_{t-1} + b_o\right)$$
$$C_t = f_t \circ C_{t-1} + i_t \circ tanh\left(W_{xc} * X_t + W_{hc} * h_{t-1} + b_c\right)$$
$$h_t = o_t \circ tanh\left(C_t\right)$$

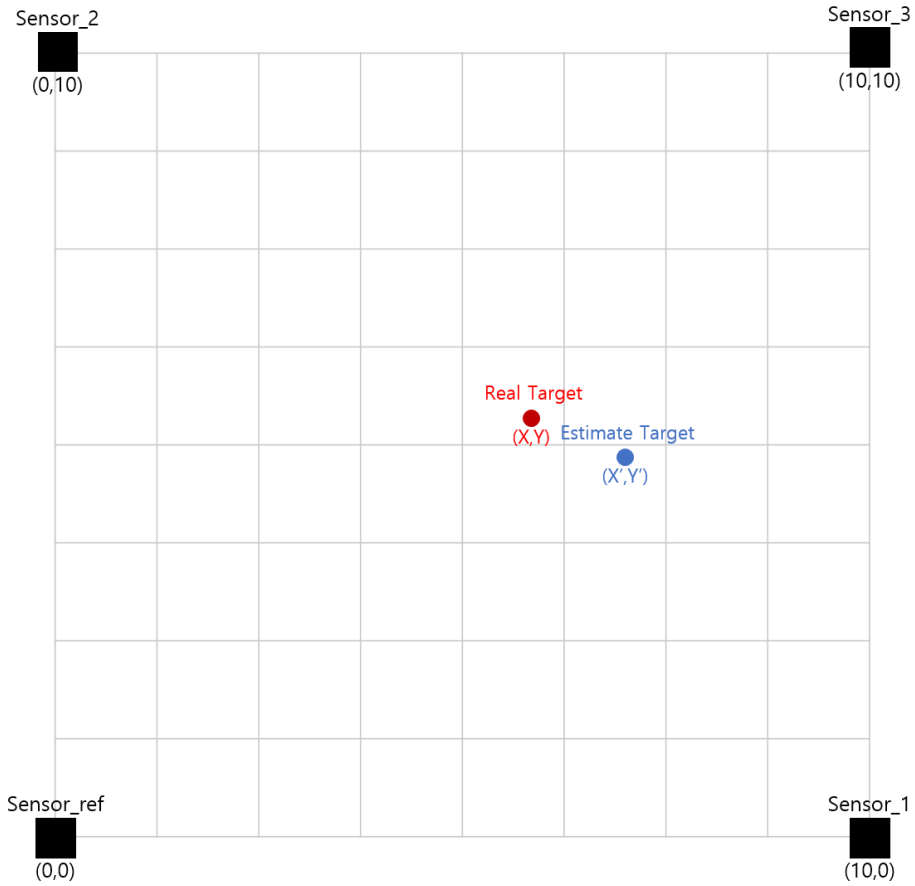
Figure 3: The inner structure of a ConvLSTM cell.

- W는 convolution 연산에서 나오는 필터의 weight를 의미하고 *는 convolution 연산을 의미
- h(t−1), X(t−1) 두 개가 concat하게 conv 연산에 들어가서 같은 크기로 4개가 나눠져 생기고 이게 각각 연산을 통해 input_gate, forget_gate, output_gate 다음 cell state, hidden state가 만들어지는 FCNNs
- 즉, conv layer에서 4종류의 필터를 이용해 값을 만든 뒤 그 값들을 각각의 gate에 넣는 구조(lstm과 input space만 다르다)

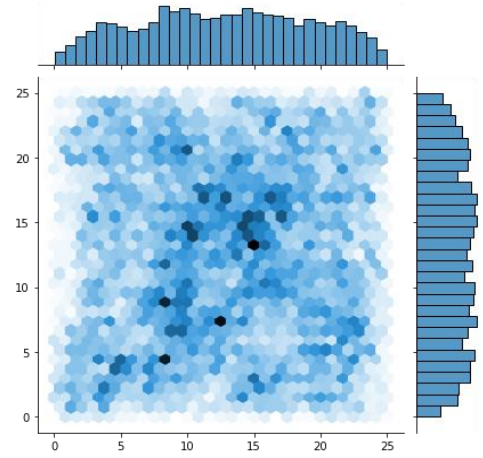
Simulation Architecture

- 필자가 정의한 환경에서 노이즈가 많은 상황 가정 및 데이터셋 생성
- 각각의 환경에서 만든 데이터셋을 이용해 각각의 모델 학습 & 예측
- 모델의 종류는 FCNNs, CNN, LSTM , CONV-LSTM으로 정의
- 학습 및 예측한 데이터들의 평균 오차 계산
- 이를 통해 기존 연구에서 고안한 모델들과 필자가 제안한 모델 성능 비교 및 장단점 분석

Simulation Architecture – Environment Setting

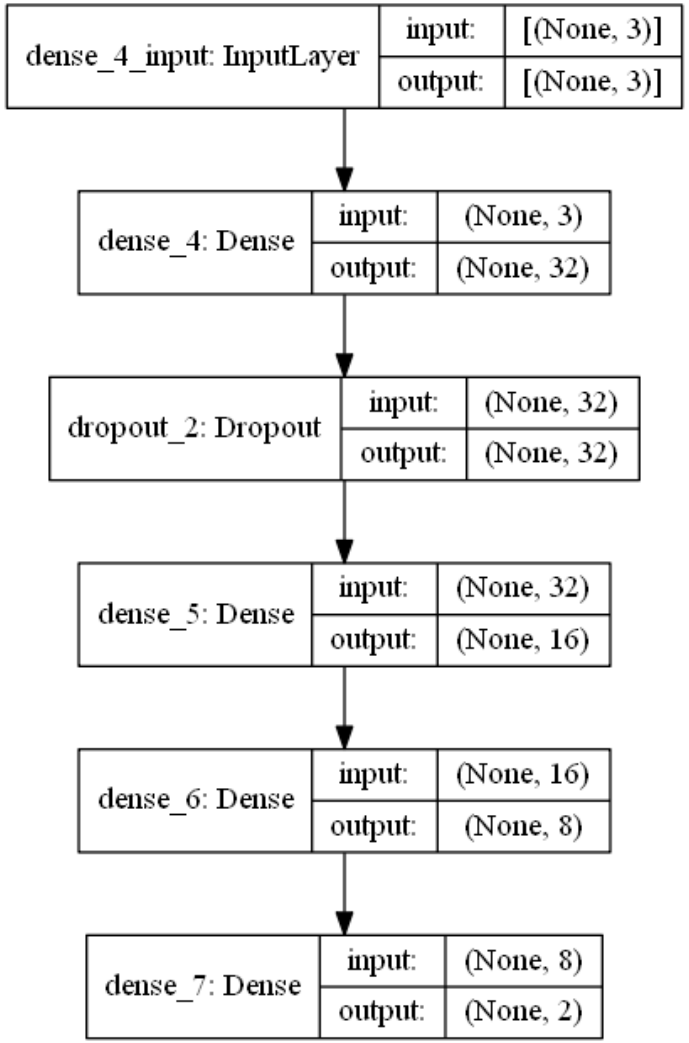


- 0~10 사이의 타겟의 X와 Y 좌표 데이터 생성
- 해당 타겟은 1초마다 걷거나 뛰고 멈추는 가속도 운동 진행
- Additive white Gaussian noise(평균 0 & 표준편차 2)을 활용해 신호에 대한 Noise 데이터 생성
- 좌표와 noise 데이터를 활용해 복잡한 실내 환경에 대한 TDOA 데이터셋 생성
- 모델은 타겟의 위치 좌표 추정



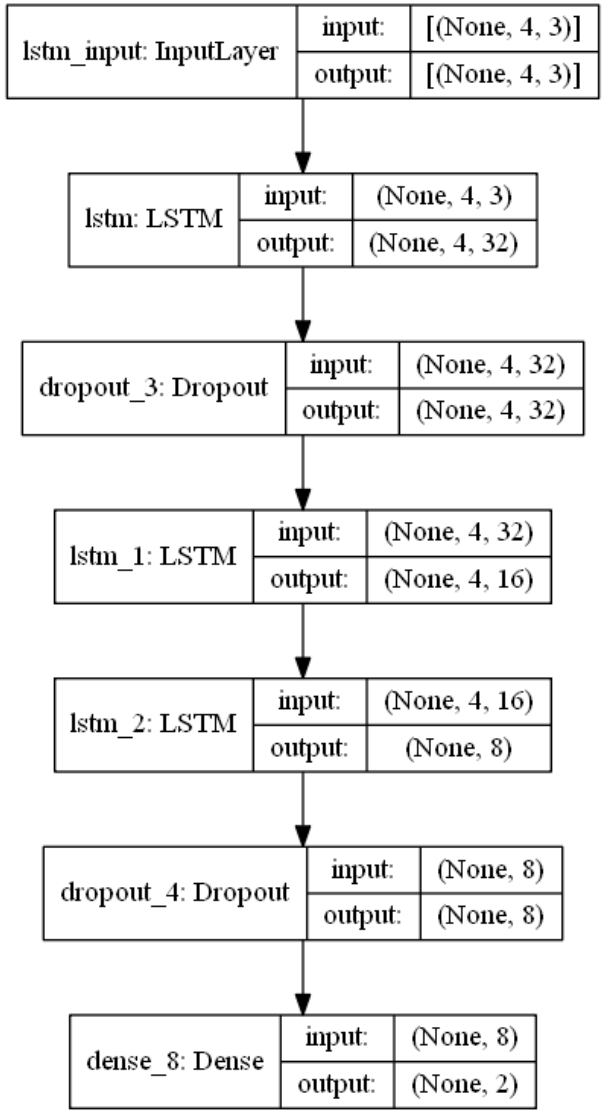
<좌표 데이터 분포>

Simulation Architecture – FCNNs



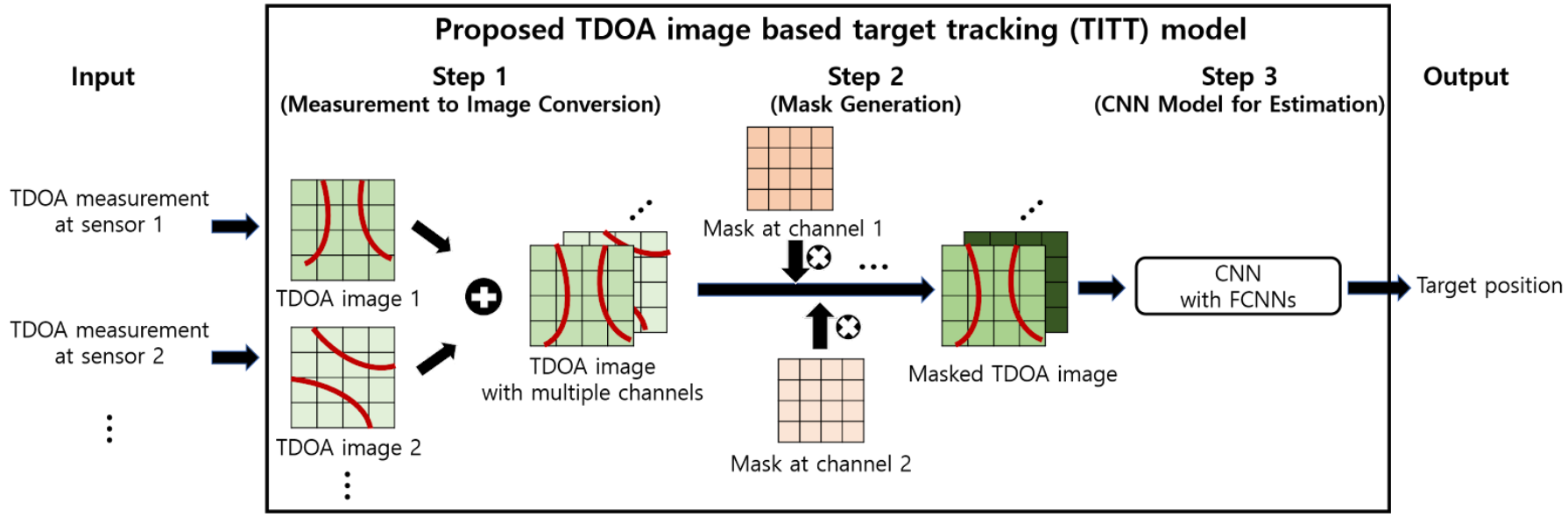
- TDOA값을 인풋, x와 y좌표를 아웃풋으로 하는 FCNNs
- Loss는 MSE로 계산이 되며 사전에 8천개의 데이터로 사전 학습
- Optimizer는 Adam(lr =1 * 10^-3), 배치사이즈는 15, epoch = 30
- 그후 2천개의 데이터를 통해 타겟의 좌표 예측
- 실제 좌표와 타겟의 좌표 차이를 평균으로 해서 모델의 성능을 종합적으로 평가

Simulation Architecture – LSTM



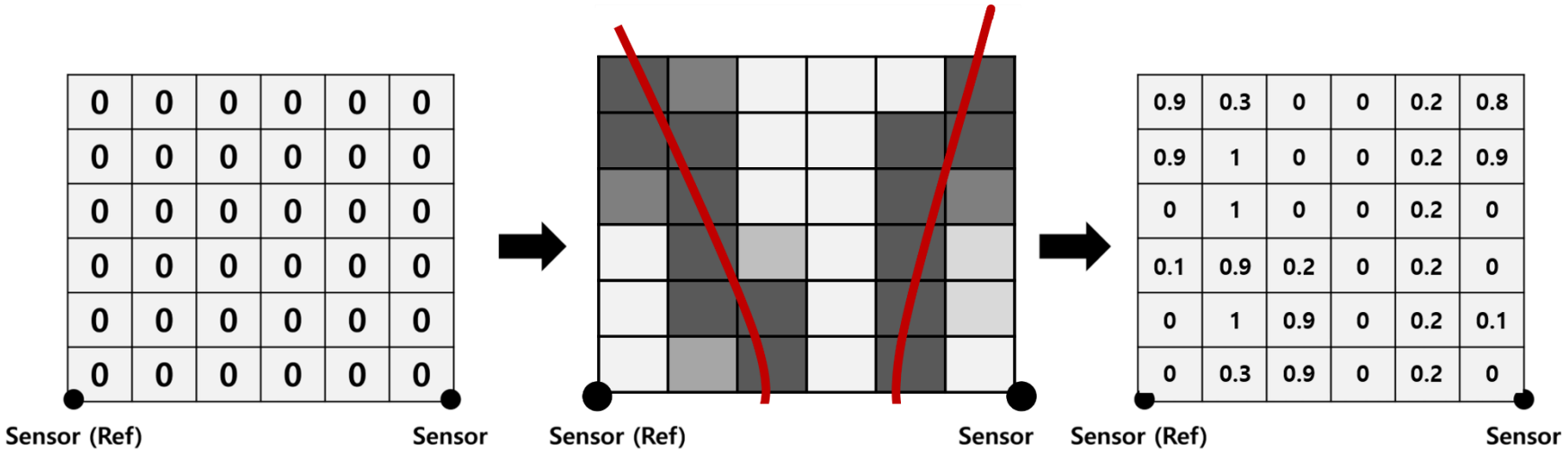
- 타겟은 시간에 따라 인간이 할 수 있는 속도 내에서 가속도 운동 진행
- 그렇기에 연속된 시간의 데이터 셋이기에 RNN 모델 사용 가능
- 해당 구조를 지닌 모델 구축
- Optimizer는 Adam(lr = 1 * 10^-3), 배치사이즈는 15, epoch = 30
- 2천개의 데이터를 통해 타겟의 좌표 예측하고 실제 좌표와의 차이를 평균으로 해서 모델의 성능을 평가

Simulation Architecture – CNN



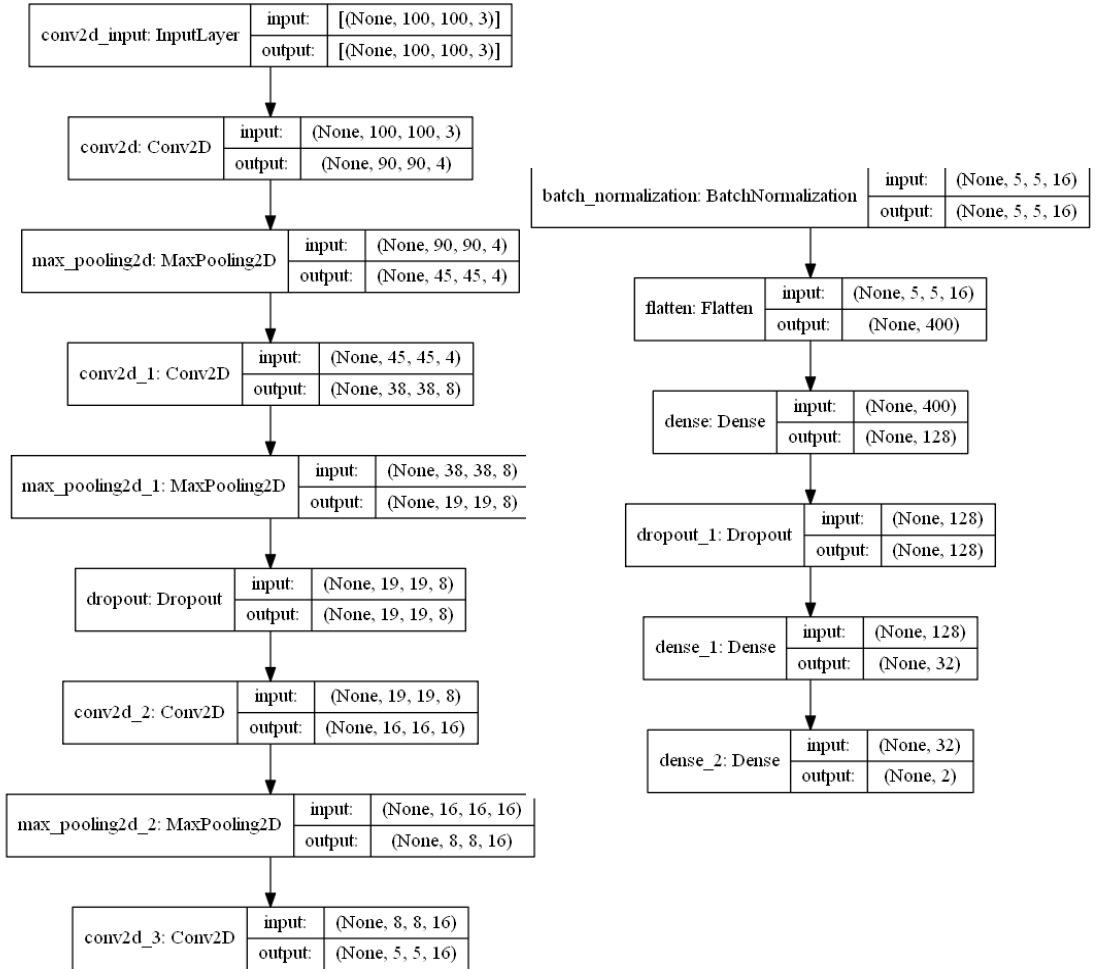
- TDOA 값들을 이미지로 변환하여 3장의 이미지를 텐서로 하여 모델에 삽입
- 그후 conv layer를 지나고 [x좌표, y좌표]를 아웃풋으로 하는 FCNN layer 통과

Simulation Architecture – CNN



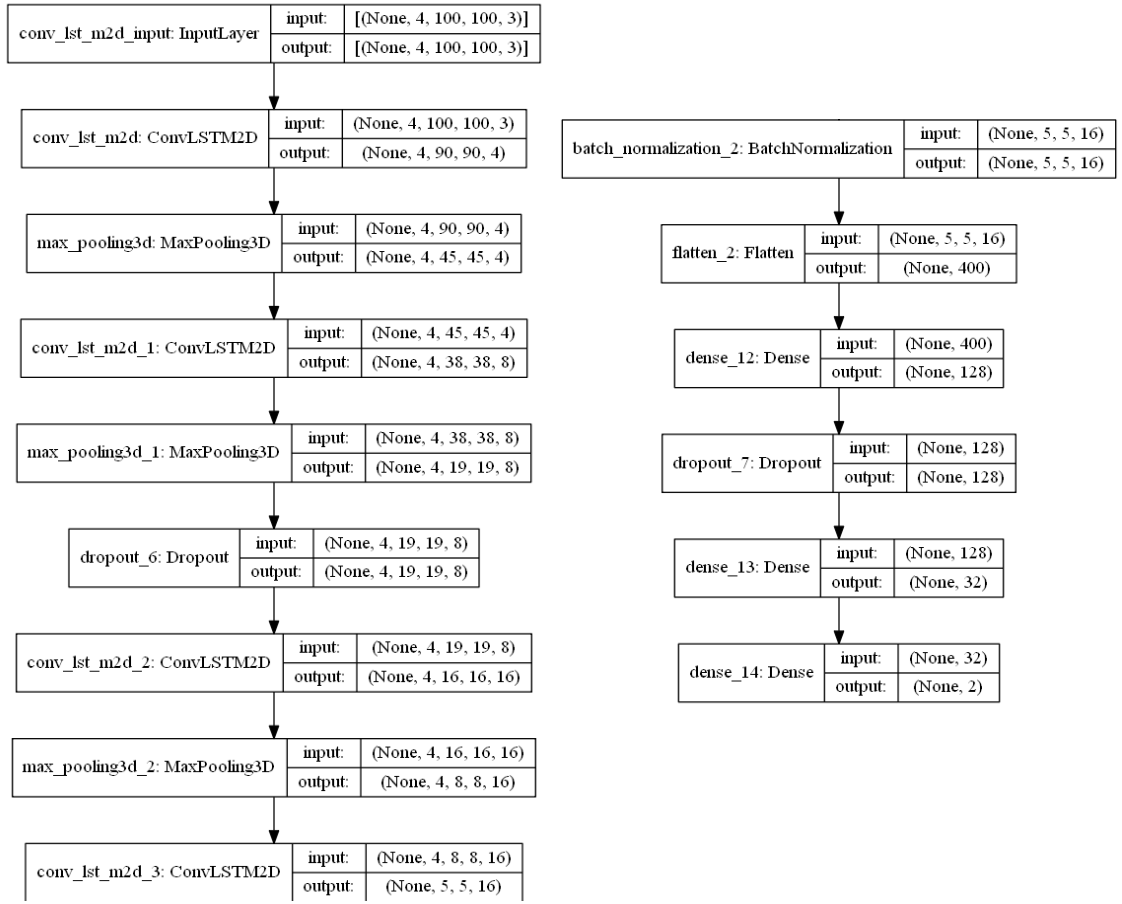
- TDOA 값들을 활용하여 해당 grid를 지나는지 표시하는 쌍곡선 이미지를 생성
- 방식은 grid의 중점에 대한 TDOA값이 인풋 TDOA값과 유사하다면 해당 grid를 1로 표시
- 추가적으로 orbital 같이 grid를 지날 확률로도 변경 가능

Simulation Architecture – CNN



- 해당 구조를 지닌 모델 구축
- Optimizer는 Adam(lr = 1 * 10^-3), 배치사이즈는 15, epoch = 30
- 1만 2천개의 데이터를 통해 학습
- 4천개의 데이터를 통해 검정
- 4천개의 데이터를 통해 타겟의 좌표 예측하고 실제 좌표와의 차이를 평균으로 해서 모델의 성능을 평가

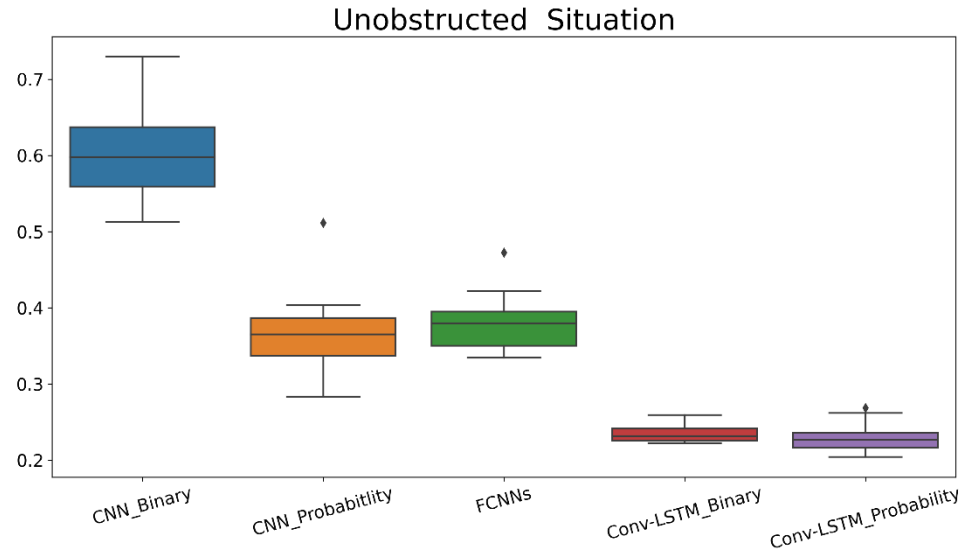
Simulation Architecture – Convolutional LSTM



- 해당 구조를 지닌 모델 구축
- Optimizer는 Adam(lr = 1 * 10^-3), 배치사이즈는 15, epoch = 30
- 1만 2천개의 데이터를 통해 학습
- 4천개의 데이터를 통해 검정
- 4천개의 데이터를 통해 타겟의 좌표 예측하고 실제 좌표와의 차이를 평균으로 해서 모델의 성능을 평가

Simulation Result – Noise Little

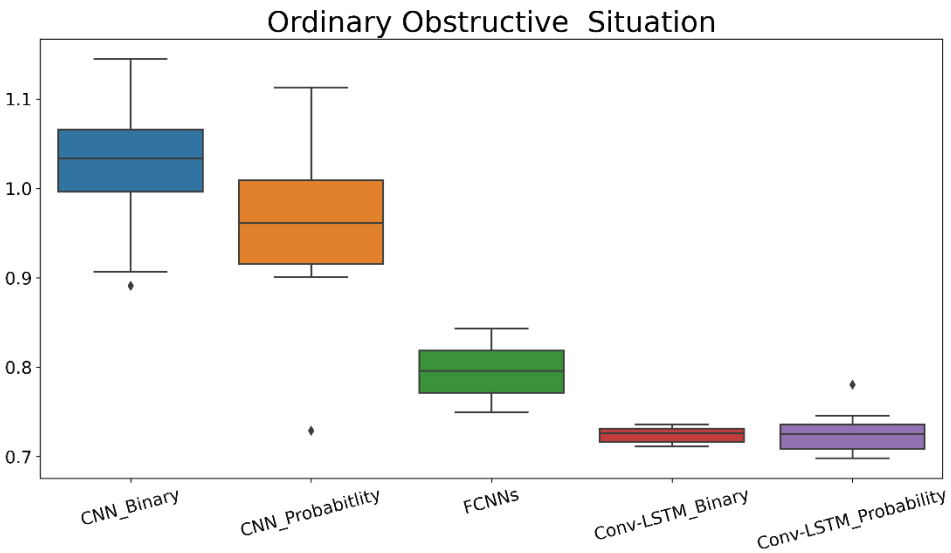
	Mean	Std	Min	Q1	Median	Q3	Max
CNN(Bin)	0.6019	0.0596	0.5128	0.5593	0.5977	0.6372	0.7298
CNN(Pro)	0.3691	0.0588	0.2835	0.3372	0.3652	0.3866	0.5112
FCNNs	0.3814	0.04	0.3346	0.3502	0.3796	0.3951	0.4723
LSTM	3.018	0.0526	2.9287	2.9943	2.9999	3.0338	3.1164
Conv-LSTM(Bin)	0.236	0.0127	0.2225	0.2256	0.2315	0.2415	0.2592
Conv-LSTM(Pro)	0.2296	0.0205	0.2041	0.2161	0.227	0.2362	0.2686



- Conv-LSTM(Pro)이 0.2296m로 가장 좋은 결과를 보임, Conv-LSTM(Bin)이 2등
- 표준편차가 0.01-2로 되게 강건하게 학습
- CNN(Pro)가 0.3691m로 사전 연구와 다르게 FCNNs 모델보다 좋은 성능을 보였다.

Simulation Result – Noise Middle

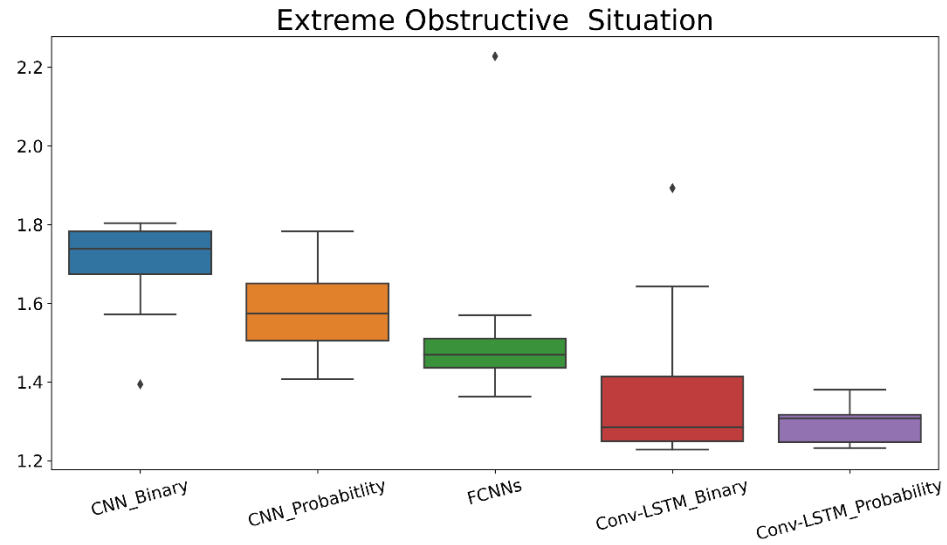
	Mean	Std	Min	Q1	Median	Q3	Max
CNN(Bin)	1.0217	0.0737	0.8909	0.9963	1.0332	1.0656	1.1449
CNN(Pro)	0.9564	0.0976	0.729	0.9156	0.9611	1.0091	1.1131
FCNNs	0.7497	0.0292	0.7497	0.7713	0.7959	0.8186	0.8433
LSTM	2.999	0.0441	2.9163	2.9768	2.9962	3.0235	3.0668
Conv-LSTM(Bin)	0.7239	0.0085	0.7114	0.7161	0.726	0.7315	0.7361
Conv-LSTM(Pro)	0.7207	0.0147	0.6981	0.7084	0.7241	0.73	0.7458



- Conv-LSTM(Pro)와 Conv-LSTM이 비슷한 성능으로 학습이 되었다
- CNN(Pro)가 0.9564m, FCNNs이 0.7497로 앞선 결과와 반대로 되었다.

Simulation Result – Noise Many

	Mean	Std	Min	Q1	Median	Q3	Max
CNN(Bin)	1.696	0.1209	1.3944	1.674	1.7377	1.7826	1.8035
CNN(Pro)	1.5874	0.1106	1.4075	1.5047	1.5736	1.6505	1.7825
FCNNs	1.5662	0.2364	1.3644	1.4131	1.4619	1.5476	2.4506
LSTM	3.0374	0.0718	2.9423	2.9805	3.0306	3.0724	3.1528
Conv-LSTM(Bin)	1.3837	0.2084	1.2281	1.2497	1.2849	1.4137	1.8925
Conv-LSTM(Pro)	1.2913	0.0459	1.2328	1.2476	1.3079	1.3174	1.3805

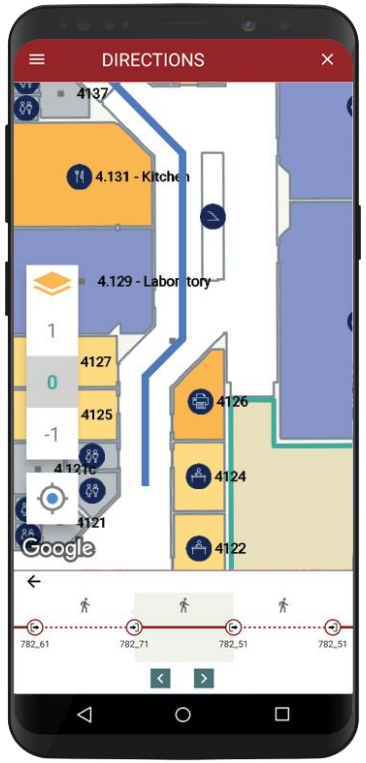


- Conv-LSTM(Pro)이 1.2913으로 가장 좋은 성능을 보였으며 표준편차는 0.04로 다른 모델에 비해 압도적으로 낮았다.
- 그 다음으로 성능이 좋은 모델은 Conv-LSTM(Bin)이며 1.3877m의 오차를 보였다.

Simulation Results - Final

- numerical value를 이용하는 모델(FCNNs, LSTM) 보다 이미지를 활용하는 모델이 더욱 성능이 좋았다.
- LSTM을 활용해 신호에서 정확한 TDOA를 구하는 연구와 다르게 LSTM은 압도적으로 안 좋은 결과를 보였다. => 노이즈가 많은 경우에서는 타겟의 운동성이 노이즈에 의해 가려진 것으로 추측!
- 이미지 모델의 경우 사전의 연구와 다르게 오히려 TDOA 그 자체를 사용하는 모델보다 더 좋은 성능을 보임 => noise에 robust한 모델 생성
- 또한, 이미지로 변환하는 과정에서 노이즈의 영향력이 감소하여 타겟의 운동성 또한 살아나게 된 것으로 추측 가능
- 앞선 추측의 결과로 Conv-lstm 모델에선 시간의 연속성을 파악 가능하여, 모델의 아웃풋에 대한 이상치가 줄어들어 평균적으로 좋은 성능을 보인 것으로 추측

Real life application



1. Indoor Navigation : 실내 경로를 찾는 경우 오차에 robust한 모델 구축 가능하기에 성능 향상 가능

2. **Gym tracking** : 운동을 할 때는 따로 기록할 필요 없이 위치 정보와 동기화하면 몇 분 동안 어떤 운동을 했는지 기록 가능

3. **Location tracking in emergency** : 화재 등 비상시에 유용하게 사용할 수 있다. 시야가 확보되지 않은 상황에서 목표 위치를 파악해 인명 구조에 도움을 줄 수 있다.


Thank
you