特 別 研 究 報 告 題 目

CNNを用いた画像による屋内測位

Indoor Location by Images using CNN

指 導 教 員　　主査 出口 利憲　教授
副査 山田 博文　准教授

岐阜工業高等専門学校 専攻科 先端融合開発専攻

2018Y07 海老澤 颯

令 和　2　年（2020　年 ）　2　月　3　日 提 出

Abstract

Positioning system for outdoor is certainly established by GNSS. It has been studied for more accuracy and cost reduction. On the other hand, Received Signal Strength Indicator (RSSI) is often adopted for indoor which could not receive the data from GNSS such as skyscraper city, underground town. the margin of error of RSSI is generally 1 10m because of multipath, wave interference and obstacle. it need to improve accuracy.

We propose the method of using images. A human can estimate the places which have seen before. In other words, the image has the location information by Positional relationship like wall and floor. We use Convolutional Neural Network (CNN) to find the features from these complex relationship for localization.

The accuracy is not bad compared with existing method under the conditions that the camera direction and hight are fixed in this experiments.

目 次

第1章　序論

屋外向けの測位は QNSS などによる測位技術がおおよそ確立されていて，さらなる測位精度の改善や受信機の低コスト化が行われている. 一方で，GNSS のデータを正確に反映できない環境 (高層ビル街の谷間や屋内，地下街) において，位置情報は，主に RSSI と IMES による測位手法が注目されている．RSSI は，ビーコンなどの Bluetooth Low Energy (BLE) 端末や Wi-Fi などの電波強度を利用した測位方法で，誤差は 1〜10m で障害物などに大きく左右されるというデメリットがある [1]．Indoor MEssaging System (IMES) [2] は. 屋内に置かれた送信機からの信号を受信することで， 受信機は送信機の場所を自己推定する手法である．誤差 は一般に 10m とされている．どちらの手法もマルチパス，電波干渉や障害物などが原因となり，測位精度には改善の必要がある.

今回提案するのは，画像を使った自己位置推定の手法である．我々は，以前に見たことある景色の画像を見た時，画像が撮影された場所を推測することができる．それは，短時間で動かないもの (窓，柱，壁，看板など) の位置関係や大きさから相対的な位置を推測している．つまり, 画像は位置を特定するための情報を備えていると考えた.

ディープラーニング [3] による画像認識技術は，年々発達していて，ImageNet などのコンペでは毎年記録が更新されている．本研究では，Convolutional Neural Network(CNN) を用いて画像から自己位置の推定を行う．CNN によって，画像から特徴量を抽出し，各々の場所をクラスとして分類することで，屋内での位置情報として取得する.

既存の屋内測位手法に比べて，電波の悪影響を受けないため，

- 測位精度の改善
- 多くの電波が行き交う環境での利用
- 障害物が多くランドマークを固定できない環境での利用
- マルチパスが起きやすい環境での利用などの場所での利用が期待できる.

第2章　屋内測位

屋内測位において現在，主流な方法は Received Signal Strength Indicator(RSSI) である．他にも Indoor MEssaging System(IMES) などが提案されている．

2.1　RSSI

Wi-Fi や Bluetooth などの電波は，アクセスポイントまたはルータからの距離に応じで強度が減衰する特性がある．発振器からの信号を受信器で受け取ることで，その強度から距離を測定する手法である．近年では Bluetooth Low Energy(BLE) による屋内位置測位の実務導入が進んでいる [1]．しかし，マルチパスや電波干渉，障害物などから影響を受けやすく，実際に利用する環境では，測位精度が大きく低下することも多い．平均誤差は 1〜10m と言われており，利用する環境により変化する．

2.2　IMES

IMES [2] は，GPS 衛星と同じ電波形式を用いた屋内 GPS 送信機 (モジュール) を設置し，送信機からは時刻情報の代わりに送信機の位置情報を送信する．これにより受信機側では屋外で行われる時差の計算を行わず，屋内 GPS 送信機の位置情報を受信機の位置としてそのまま受け取り，受信機の屋内外でのシームレスな利用を可能にしたものである．屋内に置かれた送信機からの 信号を受信することで，受信機は送信機の場所を 自己位置とする手法である．誤差は一般に 1〜10m とされている．利用は日本国内のみに限られている．

第3章　Deep Learning

近年，Deep Learning(以下，DL) は画像認識，自然言語処理，物体検知，強化学習，GAN など様々な分野で日々目覚まし速度で発展している．画像認識を応用し，今まで人の手で行われていた異常検知や良，不良の分類などが実社会への導入が進んでいる．

3.1　Convolutional Neural Network

Convolutional Neural Network(以下，CNN) は，小領域をとり，これを 1 つの特徴量として圧縮を行う畳み込み層を含んだモデルであり，画像認識の分野で広く使われているモデルである．近年では，画像以外にも応用例が広がっている．今回の実験では構造がシンプルで実装が容易な VGG16 をベースにしている．

3.1.1　VGG16

VGG [4] は，Convolution 層と Maxpooling 層を重ねた構造で，2014 年の画像認識のコンペティション (ILSVRC) のクラス分類の部門で特に高評価を得たモデルである．

3.1.2　Batch Normalization

Batch Normalization(BN) は，深層学習において、内部共変量シフトと呼ばれる，層を重ねるほど分布が不安定になること学習データと検証 (評価) データ間で各層の分布が変わってしまう現象の対策として考えられた隠れ層の正規化手法である．ある層への入力を $X \in \mathcal{R}^{H \times W \times C \times N}$($H$:縦の次元数，$W$:横の次元数，$C$:チャンネル数，$N$:バッチサイズ) とすると，それぞれのチャンネルに対して平均・分散 μ、$\sigma^2 \in \mathcal{R}^{H \times W \times C}$ を計算し，正規化を行う．

$$\hat{X} = \frac{X - \mu}{\sigma} \tag{3.1}$$

その後，平均・分散を調整する学習パラメータ γ、$\beta \in \mathcal{R}^{H \times W \times C}$ を用いて出力を計算する．

$$Y = \gamma \hat{X} + \beta \tag{3.2}$$

3.1.3　dropout

dropout [3] は，モデルの学習時に，一定割合のノードを不活性化させながら学習を行う手法である．過学習の防止が出来る．

3.2 最適化手法，損失関数と評価関数

3.2.1 最適化手法 [5]

最適化手法とは，最適解に近づくためのアルゴルズムであり，素早く最適解に近づけるものがすぐれていると言える．単純に，学習率が大きければ、勾配方向への移動距離が大きくなり、より早く学習し最適解に近づくことが期待されるが、このとき学習率が大きすぎると最適解を通り過ぎてしまい収束に時間が余計にかかる，もしくは収束しない可能性が大きくなる．

したがって、学習の序盤は大きな学習率で素早く解の付近にたどり着き、次第に学習率を減らして慎重に解に位置を合わせるのが適切である．そこで、学習率の調整が組み込まれた最適化手法が複数提案されていて，その中から代表的なものを載せる．

- SGD
- momentum
- Adagrad
- RMSprop
- Adadelta
- Adam

SGD

SGD は最もオーソドックスな最適化手法である．各更新毎にランダムにミニバッチ（訓練データから抜粋したデータの集合）を構成し、その平均損失を使用することが SGD の特徴である．

$$\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta \nabla \mathrm{E}^{(t)}(\boldsymbol{w}^{(t)}) \quad \left(\nabla \mathrm{E}^{(t)}(\boldsymbol{w}^{(t)}) = \left. \frac{\partial \mathrm{E}^{(t)}(\boldsymbol{w})}{\partial \boldsymbol{w}} \right|_{\boldsymbol{w}=\boldsymbol{w}^{(t)}} \right) \tag{3.3}$$

$$\mathrm{E}^{(t)}(\boldsymbol{w}^{(t)}) = \frac{1}{|\mathfrak{B}^{(t)}|} \sum_{n \in \mathfrak{B}^{(t)}} \mathrm{E}_n(\boldsymbol{w}^{(t)}) \tag{3.4}$$

ここで，$\boldsymbol{w}^{(t)}$ は t 回目の更新時のパラメータ，η は学習率，E は損失関数，$\nabla \mathrm{E}(\boldsymbol{w}^{(t)})$ はパラメータの空間で損失関数が最も増加する方向 (勾配)，η はパラメータの空間で損失関数が最も減少する方向 $\left(-\nabla \mathrm{E}(\boldsymbol{w}^{(t)})\right)$ にどれだけ進むかの比率を表している．SGD では学習率は明示的に与える必要があり，どのような学習率が最適かパラメータを調整する必要がある．また、単純な SGD では、勾配が小さい平坦な場所に来ると更新がほとんど行われなくなってしまうといった問題もあるが、こちらについても以下に示すアルゴリズムで同時に対処がなされている．

momentum

前ステップの更新を加味することで、 勾配の変化を滑らかにする．これによって、最適解を通り過ぎることで，loss が収束せずに行ったり来たりする振動を抑制することができる．また、勾配変化の少ない斜面においては、他の勾配変化の大きい斜面と比較して学習率が上昇し、加速的に学習が進むという効果を持っている．実験はすべてこの momentum を使用した．

更新式は以下に示す．

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \Delta\boldsymbol{w}^{(t)} \Delta\boldsymbol{w}^{(t)} = \mu\Delta\boldsymbol{w}^{(t-1)} - (1-\mu)\eta\nabla \mathrm{E}(\boldsymbol{w}^{(t)}) \tag{3.5}$$

ここで，$\boldsymbol{w}^{(t)}$ は t 回目の更新時のパラメータ，η は学習率，E は損失関数，$\nabla \mathrm{E}(\boldsymbol{w}^{(t)})$ はパラメータの空間で損失関数が最も増加する方向 (勾配)，η はパラメータの空間で損失関数が最も減少する方向 $\left(-\nabla \mathrm{E}(\boldsymbol{w}^{(t)})\right)$ にどれだけ進むかの比率を表している．Nesterov's accelerated gradient method (NAG) により

$$\Delta\boldsymbol{w}^{(t)} = \mu\Delta\boldsymbol{w}^{(t-1)} - (1-\mu)\eta\nabla \mathrm{E}(\boldsymbol{w}^{(t)} + \mu\boldsymbol{w}^{(t-1)}) \tag{3.6}$$

と更新式が変化し、$t+1$ での位置を概算した $\boldsymbol{w}^{(t)} + \mu\boldsymbol{w}^{(t-1)}$ での勾配を計算し、より変化を緩やかにする．

AdaGrad

これまでは全パラメータに対して一様な学習率が設定されていたが、各方向に対して勾配が異なることを考慮すれば、各パラメータごとに学習率を変化させることができると，より効率的な最適化ができるという考えで発展したのが AdaGrad である．AdaGrad では、全体の学習率を各方向ごとに過去の勾配の累積で割り引くことで、勾配が大きかった方向の学習率を下げ、小さかった方向の学習率を上げる工夫を導入している．

これによって、たとえ鞍点のようなある方向には勾配が小さいような状況でも、学習が進みやすくなることが期待される．

なお、AdaGrad は学習の初期に勾配が大きいとすぐに更新量が小さくなってしまい、学習がストップしてしまうという欠点がある．つまり結局のところ，SGD と同様に学習率の選択、また重みの初期値の選択が重要になるので，パラメータ調整が必須である．

更新式は次の通りである

$$\Delta\boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{\sum_{s=1}^{t}\left(\nabla \mathrm{E}(\boldsymbol{w}^{(s)})_i\right)^2 + \varepsilon}} \nabla \mathrm{E}(\boldsymbol{w}^{(t)})_i \tag{3.7}$$

ここで，$\boldsymbol{w}^{(t)}$ は t 回目の更新時のパラメータ，η は学習率，E は損失関数，$\nabla \mathrm{E}(\boldsymbol{w}^{(t)})$ はパラメータの空間で損失関数が最も増加する方向 (勾配)，η はパラメータの空間で損失関数が最も減少する方向 $\left(-\nabla \mathrm{E}(\boldsymbol{w}^{(t)})\right)$ にどれだけ進むかの比率を表している．なお、ε は計算機上での 0 割りを回避するためのもので、ごく小さい値（$\varepsilon = 10^-8$）を指定する．

RMSprop

AdaGrad では勾配の蓄積が大きくなり、更新量が小さくなると二度と大きくなることがないという欠点があった．RMSprop では、この点に対処するため、勾配の情報が指数的な減衰によって次第に忘却されるように更新式を変更したことが特徴的になっている．更新式は通りである．

$$v_i^{(t)} = \rho v_i^{(t-1)} + (1-\rho)(\nabla \mathrm{E}(\boldsymbol{w}^{(t)})_i)^2 \quad (v_i^{(0)} = 0) \tag{3.8}$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{v_i^{(t)} + \varepsilon}} \nabla \mathrm{E}(\boldsymbol{w}^{(t)})_i \tag{3.9}$$

ここで，$\boldsymbol{w}^{(t)}$ は t 回目の更新時のパラメータ，η は学習率，E は損失関数，$\nabla \mathrm{E}(\boldsymbol{w}^{(t)})$ はパラメータの空間で損失関数が最も増加する方向 (勾配)，η はパラメータの空間で損失関数が最も減少する方向 $\left(-\nabla \mathrm{E}(\boldsymbol{w}^{(t)})\right)$ にどれだけ進むかの比率，ρ は勾配情報の減衰率を表している．

Adadelta

RMSprop によって学習率が不可逆的に悪化することを防ぐことができたが、AdaGrad の全体の学習率に鋭敏であるという性質はそのままである．この全体の学習率への鋭敏性、つまり問題設定毎に適切な学習率が変化してしまうという問題は、実は更新量と勾配の次元の不一致を学習率で調整していることによるものである．そこで、AdaDelta ではそうした次元の不一致を加味して自動的に適切な学習率が設定されるようにしている．

具体的には，勾配の 2 乗の指数移動平均に加えて，更新量の 2 乗の指数移動平均をもちい，両者の比を学習率として設定している．更新式は次の通りである．

$$u_i^{(t)} = \rho u_i^{(t-1)} + (1-\rho)(\Delta \boldsymbol{w}_i^{(t)})^2 \quad (u_i^{(0)} = 0) \tag{3.10}$$

$$v_i^{(t)} = \rho v_i^{(t-1)} + (1-\rho)(\nabla \mathrm{E}(\boldsymbol{w}^{(t)})_i)^2 \quad (v_i^{(0)} = 0) \tag{3.11}$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\sqrt{u_i^{(t)} + \varepsilon}}{\sqrt{v_i^{(t)} + \varepsilon}} \nabla \mathrm{E}(\boldsymbol{w}^{(t)})_i \tag{3.12}$$

ここで，$\boldsymbol{w}^{(t)}$ は t 回目の更新時のパラメータ，η は学習率，E は損失関数，$\nabla \mathrm{E}(\boldsymbol{w}^{(t)})$ はパラメータの空間で損失関数が最も増加する方向 (勾配)，η はパラメータの空間で損失関数が最も減少する方向 $\left(-\nabla \mathrm{E}(\boldsymbol{w}^{(t)})\right)$ にどれだけ進むかの比率，ρ は勾配情報の減衰率を表している．

Adam

AdaDelta とは異なる RMSprop の改良法として Adam が挙げられる．Adam では、各方向への勾配の 2 乗に加えて勾配自身も、指数移動平均による推定値に置き換えている．こ

れにより、ある種 Momentum と似た効果が期待できる．更新式は次の通りである．

$$m_i^{(t)} = \rho_1 m_i^{(t-1)} + (1-\rho_1)\nabla \mathrm{E}(\boldsymbol{w}^{(t)})_i \quad (m_i^{(0)} = 0) \tag{3.13}$$

$$v_i^{(t)} = \rho_2 v_i^{(t-1)} + (1-\rho_2)(\nabla \mathrm{E}(\boldsymbol{w}^{(t)})_i)^2 \quad (v_i^{(0)} = 0) \tag{3.14}$$

$$\hat{m}_i^{(t)} = \frac{m_i^{(t)}}{1-(\rho_1)^t} \tag{3.15}$$

$$\hat{v}_i^{(t)} = \frac{v_i^{(t)}}{1-(\rho_2)^t} \tag{3.16}$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{\hat{v}_i^{(t)} + \varepsilon}} \hat{m}_i^{(t)} \tag{3.17}$$

ここで，$\boldsymbol{w}^{(t)}$ は t 回目の更新時のパラメータ，η は学習率，E は損失関数，$\nabla \mathrm{E}(\boldsymbol{w}^{(t)})$ はパラメータの空間で損失関数が最も増加する方向 (勾配)，η はパラメータの空間で損失関数が最も減少する方向 $\left(-\nabla \mathrm{E}(\boldsymbol{w}^{(t)})\right)$ にどれだけ進むかの比率，ρ_1, ρ_2 は勾配情報の減衰率を表している．この中から，いくつかの最適化手法を試したが，学習経過が安定していたので momentum を採用した．

3.2.2 損失関数

モデルの学習は、本質的には損失関数の最小化を行うことである．今回の実験で出力は領域，すなわち，離散的な数値になるので損失関数は多クラス交差エントロピー

$$E = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_{nk} \tag{3.18}$$

を使用する．

3.2.3 評価関数

評価関数 (metric) はモデルの出力の良し悪しを評価する．

損失関数もモデルの良し悪しの指標となるという点では同じだが、損失関数は最適化計算をとおして学習に直接的に影響するのに対して、評価関数は学習には使用されず、あくまでその時点でのモデルの評価指標を出力するのみであるという違いがある．

$$\text{正解率 } (accuracy) = \frac{\text{予測値が答えと一致した数}}{\text{全体のデータ数}}$$

で出力される．

3.2.4 検証方法

全体のデータから訓練用のデータと検証用のデータに分割し，学習に使うのは，訓練用のデータのみである．訓練データをもとに学習したモデルに，検証用データを入力し，出力を評価関数で数値化することでモデルの評価をする．比率は 7:3 もしくは 8:2 が使われることが多い．

3.2.5 可視化

GradCAM [6, 7] は畳み込み層の出力特徴マップと入力画像をもとに，その特徴マップの各チャネルを重み付けする手法である．重み付けは，そのチャネルに関するクラスの勾配に基づいて行われる．つまり，ある画像を入力したときの畳み込み層の活性化の様子をヒートマップ化して可視化したものである．以下の式で表される．

(c：クラス，y^c：確率スコア，A^k_{ij}：k 番目の特徴マップの (i,j) ピクセルにおける強度，α^c_k：クラス c の k 番目のフィルタに関する重み係数)

$$\alpha^c_k = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \frac{\partial y^c}{\partial A^k_{ij}} \tag{3.19}$$

$$L^c_{\text{Grad-CAM}} = ReLU \underbrace{\left(\sum_k \alpha^c_k A^k\right)}_{\text{linear conbination}} \tag{3.20}$$

この可視化を行うことで，

- 入力画像に対してどの部分に注目してネットワークが判断したのか
- クラスに属する部分はどの部分であるか

という，ブラックボックスとされることが多い CNN モデルに対する，重要な問題の答えになる．

第4章　実験

画像を使った屋内測位の実現のために実験1では，画像による自己位置推定が可能がどうか確かめ，実験2では，障害物のある環境下での検証する．実験3では精度の改善を行った．

4.1　実験1

領域の推定を多クラス分類問題として捉えることで，入力の屋内で撮影した画像からクラスの出力を得て，そのまま位置情報として利用するアルゴリズムを検証する．DL を用いた画像による屋内測位が出来るのか可能性を示すことが目的である．

4.1.1　実験方法

この実験の目的は，画像による屋内測位の可能性を示すことである．画像による屋内での位置推定を行うために, 岐阜工業高等専門学校の第一体育館を Figure 4.1 のように 1m ごとの各領域に分割した．横方向 (以下 X 軸) 方向に 25 分割， 縦方向 (以下 Y 軸) 方向に 34 分割し，床からカメラまでの高さは 1m，カメラの向きはステージ方向に固定と した．撮影は，太陽光の変化が分類に影響するのを防ぐため日没以降に行い，領域ごとに端から端まで 2 回行った. 画像のサイズは 192x108，領域ごとに訓練用画像が 80 枚，検証用画像が 20 枚で，学習に使用したモデルの構造を Figure 4.2 に示す．VGG16 [4] をベースに Batch Normalization を Convolution と MaxPooling の層の間に設置した [6].

4.1.2　X 軸方向の結果

X 軸を 1m ごとに分割した学習の過程は Figure 4.3 のようになった．Varidation accuracy と Validation loss はそれぞれ，検証用画像を学習モデルに適用したときの結果であり，epoch は学習モデルの訓練回数である．Varidation accuracy は最高 92.5%であった．検証用画像の 500 枚から間違えた画像 38 枚の一部を，Figure 4.4 にまとめた．GradCAM により可視化した画像が Figure 4.5 である．間違えた画像のほとんどは，答えとなる場所から ±1 までに収まっていて，±3 以上の間違いをした画像は 3 枚であった．

4.1.3　Y 軸方向の結果

Y 軸を 1m ごとに分割した学習の過程は Figure 4.6 のようになった. 検証用データの 680 枚から間違えた画像は 8 枚であり，Figure 4.7 にまとめた画像が全てである．正解した画像を GradCAM によって可視化したものが Figure 4.8 である．間違えた画像は答えとなる

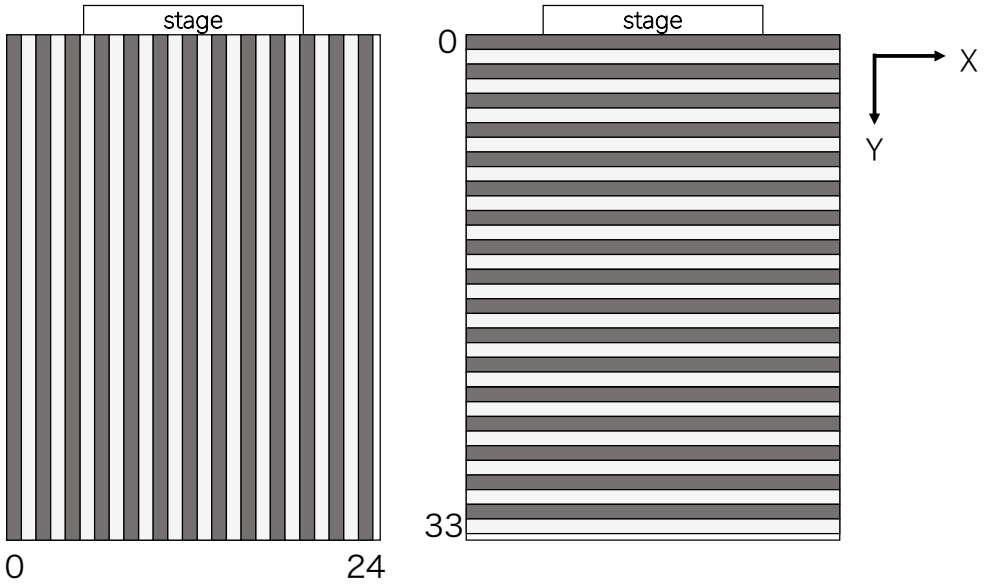

Figure 4.1: division sketch X-axis (left) Y-axis (right).

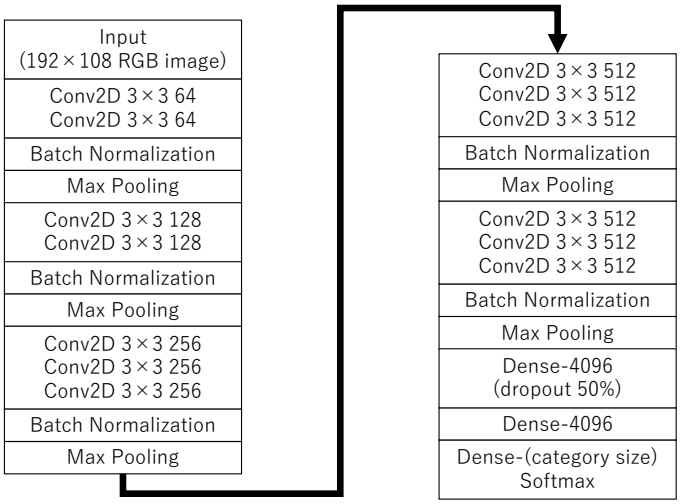

Figure 4.2: VGG16 and Batch Normalization model.

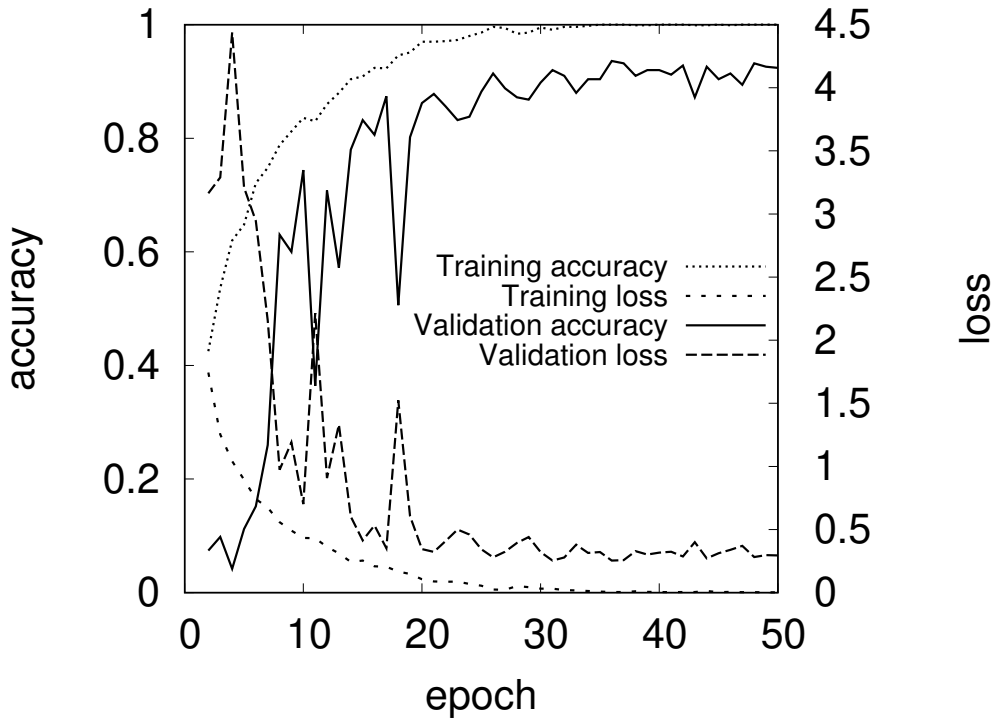

Figure 4.3: X-axis accuracy rate.

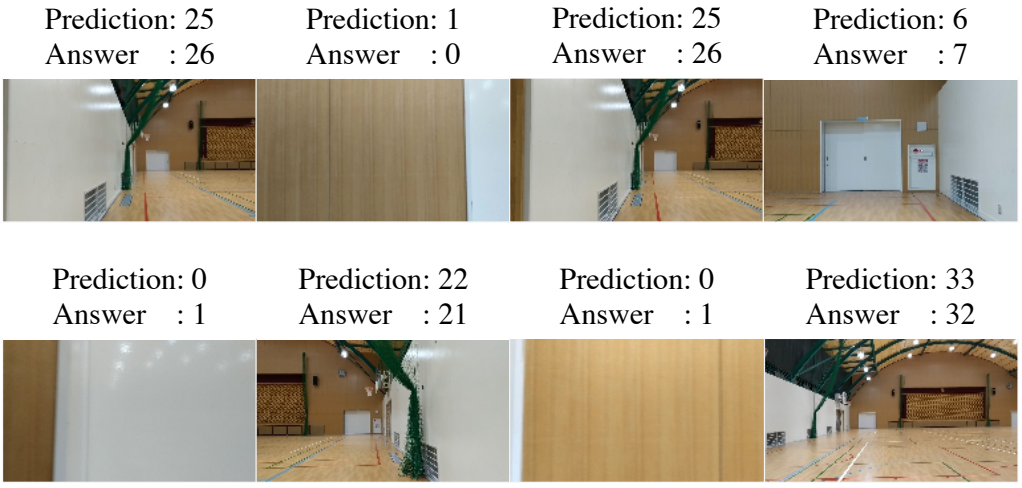

Figure 4.4: X-axis miss images.

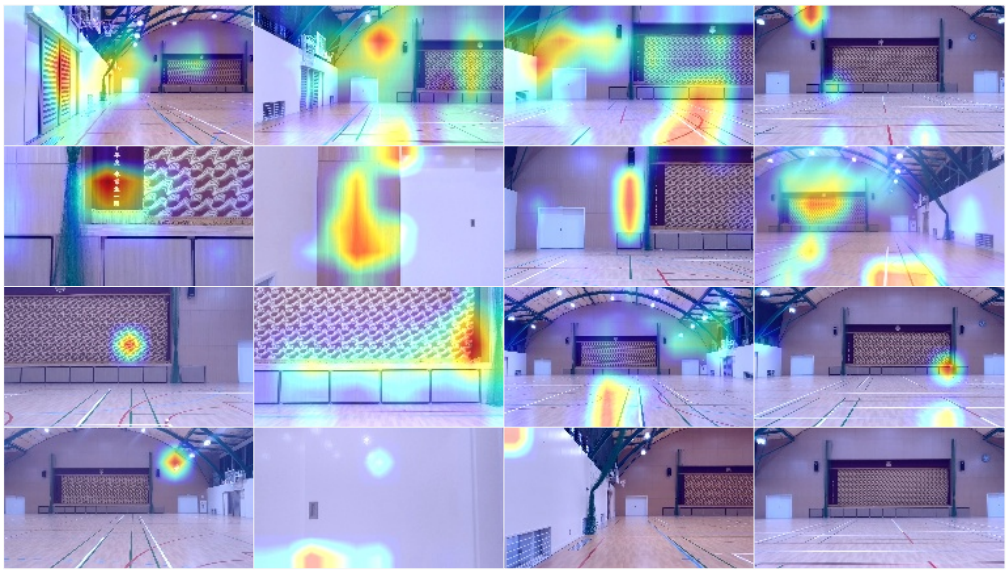

Figure 4.5: X-axis visualized images.

場所から ±1 までに収まっていた．間違えた画像 8 枚の内 3 枚は，壁からの距離が 0〜2m と非常に近い領域のため，特徴となる情報がほぼ存在しな いことから分類が困難であると考えられる.

4.1.4 考察

Grad-CAM を使って可視化した画像を見ると，Y 軸方向の分割のほうが活性化されている部分が広く，CNN にとって判別の基準になるパターンを見つけやすかったのではないかと考える. 今回の実験において，画像を用いて屋内測位をすることができる可能性が示せた.

4.2 実際 2

実験 1 より，障害物のない理想的な環境下においては，既存の技術と比べても悪くない精度を出すことが出来た．一方で，序章で述べたように，画像による屋内測位のメリットは，障害となるものが多い環境化での利用することである．実験 1 で作成した学習済みモデルを使用して，障害物が映り込む環境下での精度の変化を検証する.

4.2.1 実験方法

Figure 4.9 のように，(X 軸，Y 軸) とした，9 つのポイントから Figure 4.10 のような画像を，撮影した．撮影した時間帯は，実験 1 と同じだが，実験 1 用の訓練データを撮影した日付から 1ヶ月ほど経過してから撮影した．中心に著者が立ち，障害物とした.

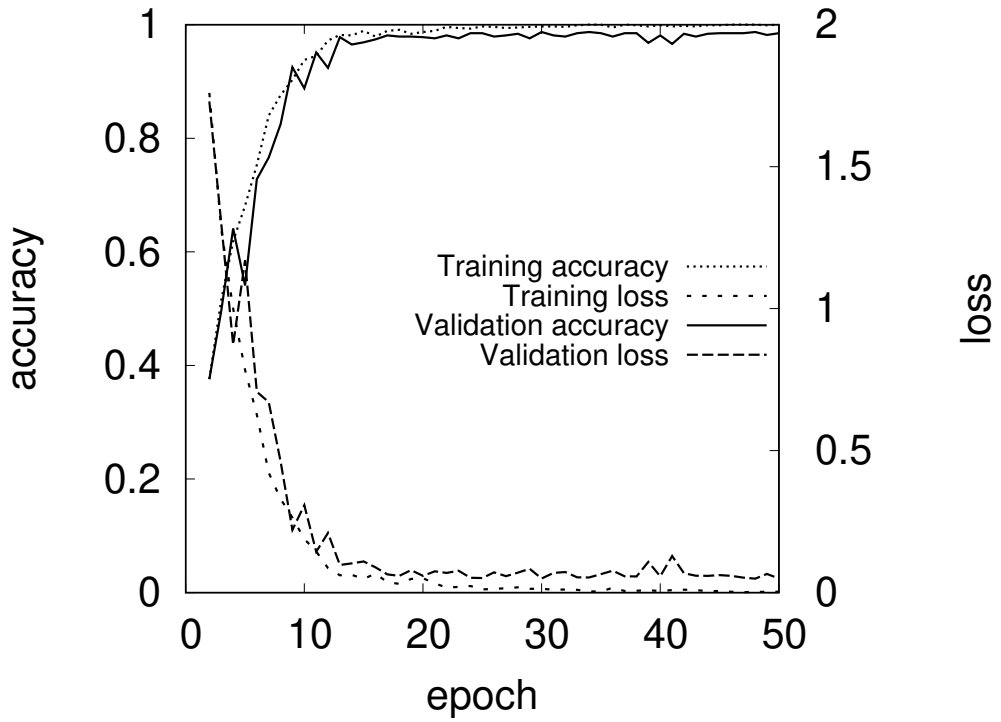

Figure 4.6: Y-axis accuracy rate.

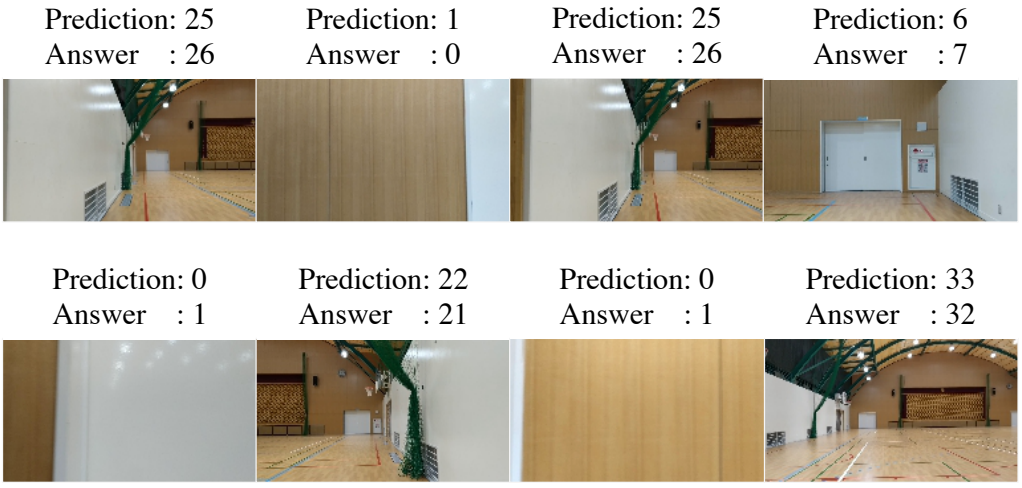

Figure 4.7: Y-axis miss images.

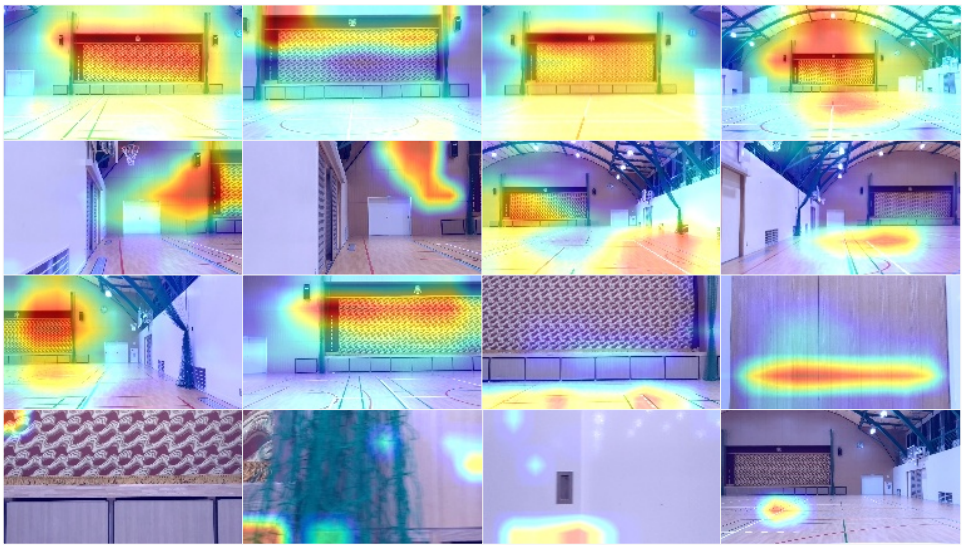

Figure 4.8: Y-axis miss images.

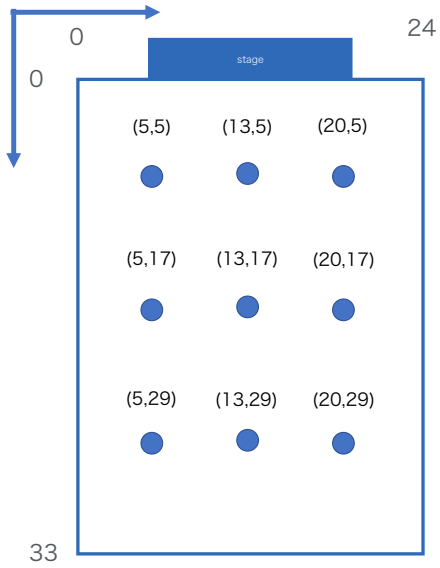

Figure 4.9: sample points.

Figure 4.10: obsticle image.

4.2.2　結果

結果は，Figure 4.11 のようになった．実験 1 で 680 枚中 8 枚のみの間違いであったことを考えると，今回の実験では 9 枚中 2 枚の間違いであるため，Y 軸方向の精度の低下している，しかし，概ね位置推定が出来ているように見える．問題は X 軸方向で，全体的に予想が左側によっている．

4.3　実験 3

実験 2 より，Y 軸方向に比べて，X 軸方向の精度が著しくわるいので，モデルを変えて精度の改善を試みる．

4.3.1　実験方法

新たに検証するモデルは以下である．

- Figure 4.12 のように Batch Normalization(BN) と dropout をなくした，オリジナルの VGG16

dropout の役割は，学習中にニューロンの繋がりをランダムに断つことで，より頑健なモデルを作ることである．つまり，本質的には障害物が画像に写っている状態と似た状態を再現出来ると考え，Convolution 層と Maxpooing 層の間に dropout を行うモデルを用意した．

- Figure 4.13 のように．dropout を 30%に設定したモデル．
- Figure 4.14 のように，dropout を 50%に設定したモデル．

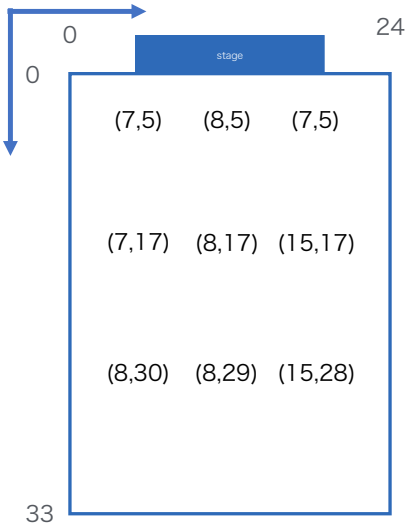

Figure 4.11: Prediction with obsticle images.

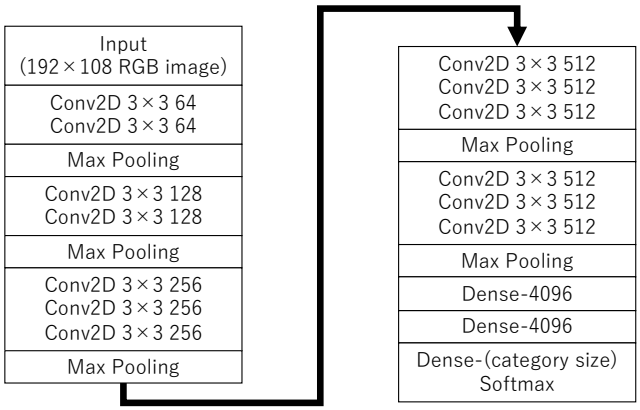

Figure 4.12: VGG16 model.

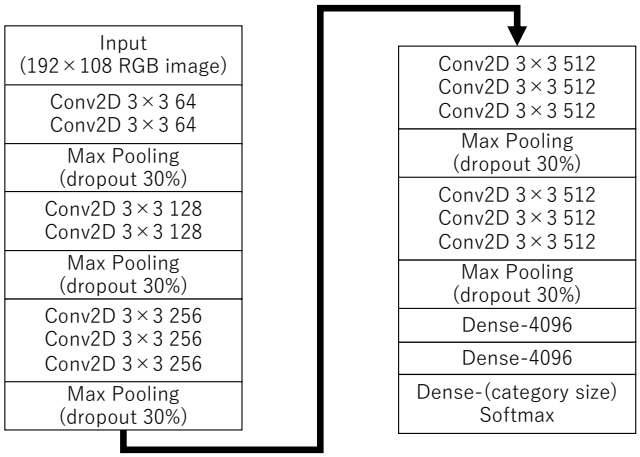

Figure 4.13: VGG16 dropout30% model.

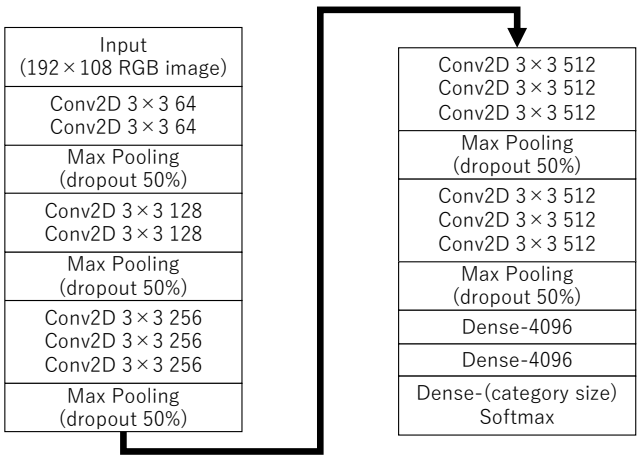

Figure 4.14: VGG16 dropout50% model.

上記のモデルを使って，実験 1 と同じ訓練データを使い，精度の変化を確認する．

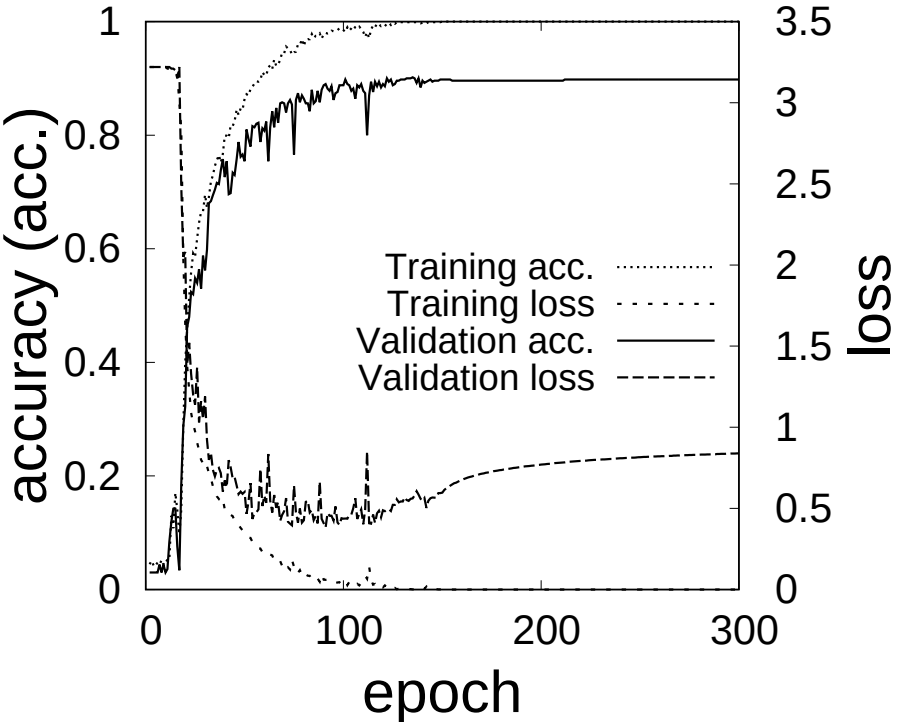

Figure 4.15: VGG16 training trainsition.

4.3.2 学習結果

学習結果を Figure 4.15, 4.16, 4.17 に示す．Validation accuracy の最高値はそれぞれ，90.2%, 90.2%, 89.2%であった．Figure 4.3 の validatition accuracy は最高で 92.5%になるので，Figure 4.3 よりも劣っている．オリジナルの VGG16 は，epoch100 前後から validatition loss 増加傾向に転じているので過学習の傾向が見える．dropout 学習が収束しにくくなっている．

4.3.3 サンプルポイントでの結果

X 軸の結果のみを示す．各マスの表記は DL の予測/真の値 である．全体の誤差を 絶対値誤差の平均値 で評価する Figure 4.18， 4.19， 4.20， 4.21 の絶対値誤差の平均値は，順に 5, 2, 1.45, 1.77 である．

4.3.4 考察

validation accuracy が BN を加えたモデルよりも低いにも関わらず，障害物のある環境下で，改善が見られたのは，意外な結果である．結果的には，学習結果の通り，Y 軸方向ほどの精度は出ないが，既存の測位方法と変わらない精度になったと考えている．

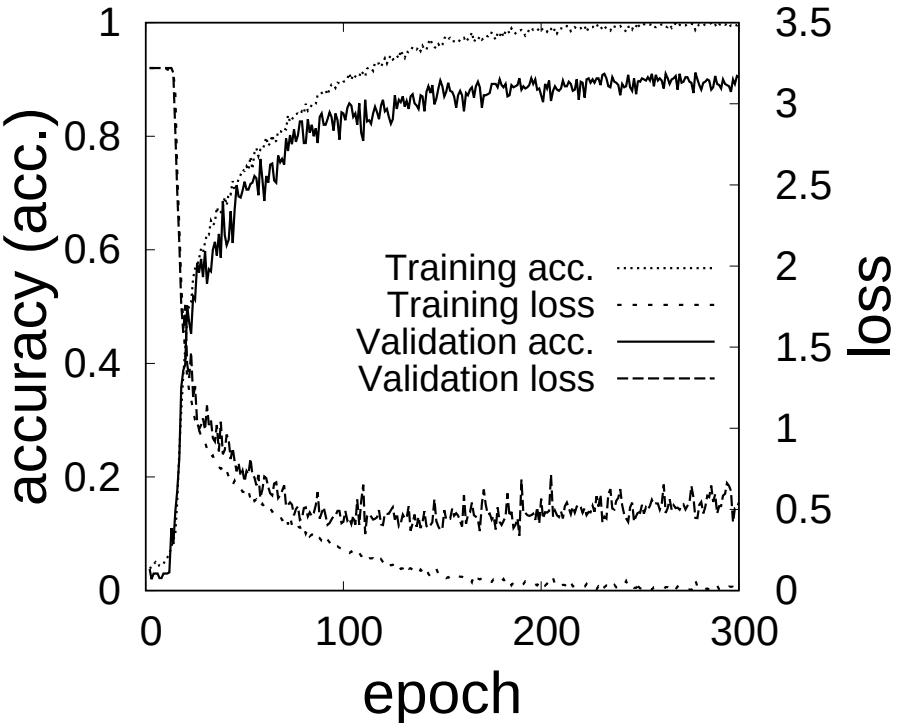

Figure 4.16: VGG16(dropout30%) training transition.

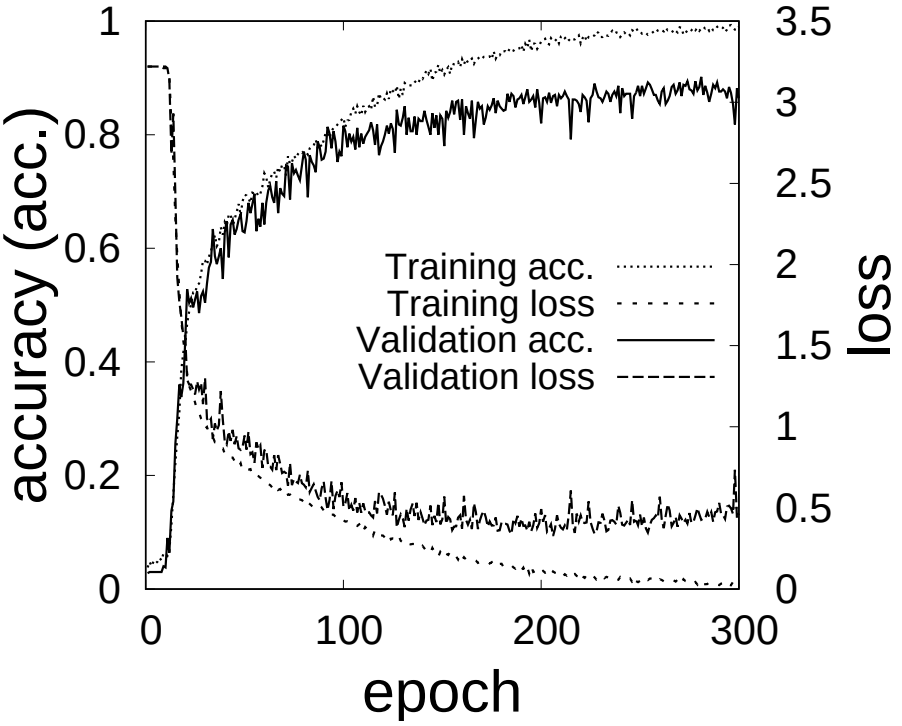

Figure 4.17: VGG16(dropout50%) training transition.

7/5	8/13	7/20
7/5	8/13	15/20
8/5	8/13	15/20

Figure 4.18: VGG16 BN.

7/5	10/13	23/20
5/5	13/13	15/20
5/5	13/13	15/20

Figure 4.19: VGG16 original.

9/5	10/13	15/20
5/5	13/13	20/20
5/5	13/13	19/20

Figure 4.20: VGG16 dropout30%.

3/5	10/13	15/20
5/5	13/13	15/20
5/5	13/13	19/20

Figure 4.21: VGG16 dropout50%.

4.4 考察

障害物が写り込まない理想的な環境よりは精度が下がることが容易に予想出来るが，CNNの特性を考えれば，障害物が写り込んでいる教師データを作成する必要がある．教師データを改善すれば，実験 1 の学習結果のような障害物が写り込まない状態と同レベルの精度まで改善するのではないだろうか．また，今回は出力層の値をそのまま利用したが，実際の環境においては，連続で撮影した複数の画像を利用することや，突然，遠くの位置に移動してしまうようなことはないことを利用した誤差の緩和ができるのではないかと考えている．

第5章　結論

本研究では，画像から屋内測位を行う手法の可能性を提示することが出来た．画像を使った自己位置推定は，Y 軸方向の分割のほうが X 方向の分割に比べて強い傾向がある．Y 軸方向の分割のほうが活性化されている部分が広く，CNN にとって判別の基準になるパターンを見つけやすかったのではないかと考える．実際に使うためには，カメラの向きを固定した状態での精度が必要である．角度をラベル情報として，学習させることで今回の実験と同水準まで精度が出せる可能性は十分にあると考えている．また，学習データを用意するという測位の環境を作るコストが，ビーコンの設置よりもはるかに高いので，より少ないデータで精度を保てるような，新しいアプローチが必要である．

謝辞

最後に，本研究を進めるにあたり，ご多忙中にも関わらず多大なご指導を賜わりました出口利憲先生に深く感謝するとともに，同研究室において共に勉学に励んだ皆様に厚く御礼を申し上げます.

参考文献

[1] 古舘 達也，堀川 三好，菅原光政, “歩行者を対象とした屋内測位手法の提案”，第 77 回全国大会講演論文集,1,pp.313–314, 2015

[2] Naohiko Kohtake, Shusuke Morimoto, Satoshi Kogure, Dinesh Manandhar, “indoor and Outdoor Seamless Positioning using Indoor Messaging System and GPS”, 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN), pp.21–23, 2011

[3] KRIZHEVSKY Alex, SUTSKEVER Ilya, HINTON,Geoffrey E, “Imagenet classification with deep convolutional neural networks,” In: Advances in neural information processing systems, pp.1097–1105, 2012

[4] Karen Simonyan, Andrew Zisserman,“Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv:1409.1556

[5] https://weblab.t.u-tokyo.ac.jp/dl4us/

[6] Francois Chollet ,“Deep Learning with Python,” Manning Publications, 2017.

[7] SELVARAJU, Ramprasaath R., et al. “Grad-cam: Visual explanations from deep networks via gradient-based localization,” In: Proceedings of the IEEE international conference on computer vision, pp. 618-626, 2017