自動監視システムのための CNN の分散処理による送信データ量削減手法

池田佳弘[1] 柳沢豊[2] 岸野泰恵[2] 水谷伸[2] 白井良成[2] 須山敬之[2] 松村耕平[1] 野間春生[1]

概要：大規模な一過性のイベント会場においては，安全にイベントを開催するため，カメラやセンサノードを用いて会場を自動で監視できるシステムを仮設で構築する需要が高い．このような会場内で人物追跡を行う場合，必要なセンサノードの数が膨大となるため，キャリブレーションの労力，およびデータを収集する際の帯域幅への負担が課題となる．本研究ではこれらの負担を低減する自動監視システムの構築を目標として，Convolutional Neural Network（CNN）により個々の監視映像から同一人物の位置と進行方向を推定し，それらの情報をつなぎ合わせていくことでキャリブレーションの必要なく人物の移動経路を取得できる方式を提案する．さらに， CNN の分割部の畳み込み数を意図的に減らし，その計算プロセスをセンサノードとサーバで分散処理することで，送信データ量低減も同時に実現する．提案する CNN の識別精度を実験によって検証したところ，センサノード上で何も処理しない場合に対して送信データ量を 1/900 に低減した場合，人の進行方向を 8 方向に識別する場合であれば 96%程度，人の CG モデルの位置を 16 区画に識別する場合であれば 98%程度の精度で推定可能であった．また，1 つの監視映像にオクルージョンが生じる場合であっても，異なる監視映像からの情報を利用することで進行方向の識別を行うことが可能であった．送信データ量を低減しても 96%以上の精度で識別が行えたことから，目標とするシステムにおいて提案手法は有効であるといえる．

1. はじめに

オリンピックのような大規模な一過性のイベントにおいては，不審者の発見や追跡といったセキュリティの強化に向けて，カメラを備えたセンサノードを仮設して自動で監視を行うことへの需要が高まっている．このように大規模なイベント会場内で人物追跡を行う場合，利用するセンサノードの数が膨大となる．近年ではセンサノードが安価に入手可能となり運用コストは現実的となった．しかし，設置後の初期設定であるキャリブレーションにかかる労力とデータを収集する際の帯域幅への負担が依然として課題となる．

そこで本研究では，複数のセンサノードで取得した画像を統合して処理する上で，キャリブレーションを行うことの労力と帯域幅への負担を同時に低減する自動監視システムの構築を目標とする．このシステムでは Convolutional Neural Network（CNN）によって個々のセンサノードの監視映像から歩行者の位置と進行方向を推定し，同一の歩行者の位置と進行方向をつなぎ合わせていくことで，キャリブレーションの必要なくセンサノード間を跨った歩行者の移動経路を取得することを可能にする．さらに，計算資源を持ったセンサノードとサーバで CNN の計算プロセスを分散処理し，図 1 のようにセンサノードから処理された CNN の特徴量をサーバに送信することで，送信データ量低減も同時に達成する．これによりキャリブレーションの負担と帯域幅への負担の双方を低減し，センサノードを仮設するだけで簡易に人物追跡を行うことを可能にする．

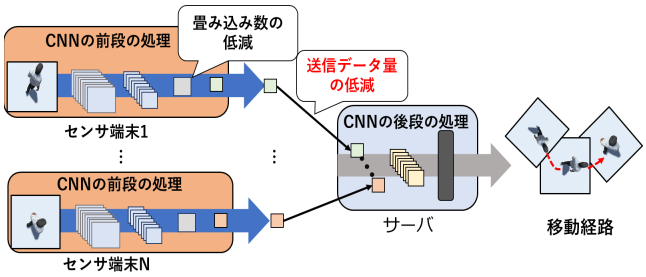

図 1 人の移動経路を自動で取得するシステムの例

2. 関連研究と本研究の位置づけ

2.1 複数のカメラによる人物追跡

複数のカメラを利用した人物追跡は様々な研究が行われている．加藤らは 3 次元情報が既知の楕円体モデルを利

[1] 立命館大学
yoshihiro.ikeda.ap@hco.ntt.co.jp
[2] NTT コミュニケーション科学基礎研究所
NTT Communication Science Laboratories

用して様々なシミュレーション画像を作成し，そのシミュレーション画像と実際のカメラ映像から人物のみを抽出した差分画像を比較することで，人の位置を推定した．[1]．また森らは計算資源を取り付けた個々のカメラ上で人物の特徴を独立に取得し，その特徴と取得時刻をサーバで結びつけることで，観測時刻が非同期であっても同一人物追跡を行うことが可能なシステムを提案した．[2]．さらに石塚らは，カメラセンサネットワーク内において，特定のオブジェクトに着目し，そのオブジェクトを撮影しているカメラを検索するシステムを開発した．[3]．

このような従来の研究では，複数のカメラ画像を統合的に処理することによって広範囲な人物追跡を可能にしているが，事前にカメラのキャリブレーションを行う必要があった．しかし大規模なイベント会場にセンサノードを仮設して人物追跡を行う場合，利用するセンサノードの数は膨大になるため，個々のセンサノードをキャリブレーションすることの負担は大きい．そこで本稿では，Convolutional Neural Network（CNN）を利用することでキャリブレーションなく映像から人物追跡を行うことを目標とする．

2.2 CNN による人物追跡

従来の一般的な画像処理の手法においては，人があらかじめ画像処理に用いる特徴量を決定する必要があり，画像処理手法の実現には，試行錯誤の労力が必要であったり，ノウハウがなければ実現できないという課題があった．これに対し，この CNN では，画像に対して複数回の畳み込み処理を行うことで，機械が自動的に画像処理に用いる特徴量を抽出することが可能になった．これによって画像識別 [4], [5], [6], [7], [8]，物体検出 [9], [10], [11], [12]，そして同一人物の対応付け [13], [14], [15] を自動で行うことが可能になった．例えば Karen らの研究では，画像に対して 64 回以上の畳み込み処理を行う層を 16 層以上重ねることで，1000 クラスの画像識別において 74%以上の識別精度を達成した．[6]．また Joseph らの研究では，CNN によって映像から 63.4%以上の精度で特定の物体の位置とカテゴリを推定した．[11]．さらに Wei らの研究では，物体検出によって切り出された 2 枚の人物画像それぞれに対して畳み込みを行ない，処理された特徴量を比較することで，同一人物かどうかを自動で識別するネットワークを提案した．[13]．このように，CNN によって個々のカメラ映像から人物の位置や進行方向を推定し，同一人物を対応付けていくことで，キャリブレーションなく映像から人物追跡することが可能になる．

3. 提案手法の設計

本稿が目標とするシステムは，複数のセンサノードから取得した映像を CNN で処理して人の位置と進行方向を推定し，推定結果をもとに映像をつなぎ合わせることで人物の移動経路を取得する．しかし，本稿が処理することを想定する映像はデータ量が大きいため，複数のセンサノードから映像データを収集する場合には帯域幅の制限が課題となる．そこで本稿ではエッジコンピューティングによって個々のセンサノードからサーバに送信するデータ量を低減し，帯域幅への負担を低減する．

3.1 エッジコンピューティングによるデータ処理手法

単純なクラウドコンピューティングシステムにおいては，センサノードから得られたデータを処理する場合，データを豊富な計算資源を持つサーバやクラウドに送信することが一般的である．しかし，本稿が処理することを想定する映像はデータ量が大きいため，複数のセンサノードから映像データを収集する場合には帯域幅の制限が課題となる．これに対し，近年では JeVois Smart Machine Vision [*1] のように，センサノード上で従来の画像処理や CNN の処理を行うことが可能なデバイスが開発された．このエッジコンピューティングという手法は，このようなデバイス上でデータを処理し，必要な場合のみデータをサーバに送信することで，送信データ量の削減や，サーバ上における処理量の低減などを行うものである．[16]．例えば Surat らは，センサノードに小規模な CNN を実装してセンサノード上で取得したデータの推論を行い，推論の精度が十分でない場合にだけデータをサーバに送信して処理することで，送信データ量を低減した．[17], [18]．そこで本研究でも，センサノードに簡単な画像処理や CNN の一部を実装し，取得したカメラ画像をあらかじめ処理したうえでサーバに送信することで，センサノードが送信するデータ量を削減することを考える．

本稿が目標とする通信システムでは，バッテリーとカメラセンサを備えたセンサノードを利用して映像を取得し，その映像を CNN で処理することで人物追跡を行う．そのため，現段階では計算資源や電源の限られたセンサノード上ですべての CNN の処理を行うことは難しい．そこで，一部の CNN の処理をセンサノードで行い，中間層の段階でデータをサーバに送ることで処理の分担を行うことを検討する．しかし，一般的な CNN では精度向上に向けて多くの畳み込み処理を行うため，センサノード上の計算量を抑えるために入力層に近い層で処理を分割すると送信データ量が増えてしまう場合がある．

そこで本研究では，意図的に分割部の畳み込み数を減らした CNN の計算プロセスをセンサノードとサーバで分散処理し，センサノードから CNN の特徴量を送信することで送信データ量の低減とセンサノード上の処理の低減を両立させる手法を提案する．提案する手法の実装の一例を図 4，図 9 に示す．

[*1] JeVois Smart Machine Vision. https://www.jevoisinc.com, 2019.

このように畳み込み数を減らしたCNNの計算プロセスを分散処理することで送信データ量低減と，センサノード上の計算量低減が可能になるが，畳み込み数を減らした場合には識別精度が低下する可能性がある．そこで本稿では，意図的に畳み込み数を減らしたCNNを作成し，その識別精度を検証することで，このようにCNNをセンサノードとサーバに分割して配置することによって通信量の削減と精度の維持が両立することを検証する．

3.2 サーバ上における特徴量の統合処理

提案手法を検討する上でもう一つ重要なのは，個々のカメラの細かなキャリブレーションをしなくても複数のカメラ画像を組み合わせて人物の追跡が行えるか，という課題である．しかし，複数のセンサノードから得た特徴量をサーバで統合的に処理する場合，統合がノイズとなって単体のセンサノードから得られる特徴量のみを処理するよりも精度が低下する可能性がある．そこで本稿では単体と2個のセンサノード-サーバ間でデータを処理する場合の識別精度を比較し，特徴量の統合が精度にどのような影響を与えるかについても検証する．

4. 送信データ量と精度の関係性を検証

本実験ではまず，1対のセンサノードとサーバ間でCNNの分散処理を行うことに着目し，人の進行方向と位置を推定する場合の送信データ量と識別精度の関係性を検証した．以降は実験に使用した3種類のデータセットと，それらのデータセットを識別するために使用したCNNについて述べる．本実験では実際にセンサノードとサーバに実装したわけではなく，一つのPC上で仮想的に実験を行なった．

4.1 実画像を用いたデータセット（データセット1）

進行方向を識別するデータセット作成に向けて，5mの高さにある天井に5台のカメラを真下向きに取り付け，640 × 480サイズで歩行者を撮影した．歩行者は成人男性4名，成人女性2名の計6名である．歩行者は指示に従い，床に設置されたラインに沿って上下左右，斜めの8方向に直進した．我々はこの映像からフレーム間差分を用いて歩行者を発見し，矩形で切り出しを行い，200 × 200にリサイズした．そして，体がおおむね9割以上映るものを目視で選択し，「左斜め上」～「人が映っていない」にラベル付けした．図2に切り出した画像とラベル付けの例を示す．ここで「人が映っていない」画像は，1台のカメラ映像から人が映っていないフレームを1枚取り出し，200 × 200サイズでランダムに切り出した．

我々はこのデータセットをデータセット1（進行方向-実画像）とし，計13,500枚（1,500枚× 9クラス）用意した．そして学習用として9,000枚（1000枚× 9クラス），検証用として1,800枚（200枚× 9クラス），テスト用として2,700枚（300枚× 9クラス）に分けて使用した．

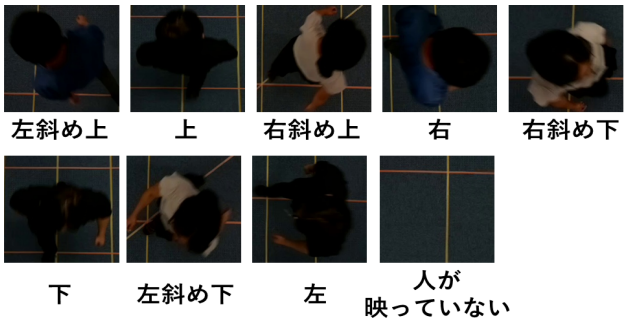

図 2 進行方向推定に使用した画像の例

4.2 CGを用いたデータセット（データセット2）

CGの環境では，現実には撮影することが困難な様々な環境を容易に構築して仮想的に撮影することが可能となる．そのため，本稿では以降の実験においてCGで作成したデータセットも利用する．しかし，現実の画像を使用した場合とCGの画像を使用した場合で，CNNの識別精度の出方に差が出る可能性がある．そこで，データセット1（進行方向-実画像）の画像に似せた画像をUnity上でCGで作成し，CG画像に対するCNNの識別精度を検証し，現実の画像とCGの画像を利用した場合に識別精度の出方に差が出るかを検証した．以降はこのCGのデータセットのことをデータセット2（進行方向-CG画像）とする．

CGによるデータセット作成のために，Unity上に1台のカメラを真下向きに設置し，640 × 480サイズで撮影した．そしてその映像から，データセット1（進行方向-実画像）と同様に切り出し，リサイズ，ラベル付けを行なった．撮影にはCGモデルを6体用意し，1体ずつUnity上で歩かせた．我々はデータセット2をデータセット1と同量用意し，同じ設定で学習用，検証用，テスト用に分けて使用した．

4.3 位置推定実験用データセット（データセット3）

人の位置を識別するためのデータセット作成に向けて，まずUnity上に1台のカメラを設置し，進行方向推定と同じ6体のCGモデルが画面上から画面下に向かって直進する様子を640 × 480のサイズで撮影した．そして，撮影した映像からCGモデルが映るフレームを選択し，そのフレーム内を4 × 4の16分割し，CGモデルの足が存在している区画にラベルづけした．図3に撮影した映像から取り出したフレームをラベル付けする例を示す．この例では，フレーム内にいるCGモデルの位置は区画6であるとラベル付けした．

以降はこの人の位置推定のために作成したデータセットを，データセット3（位置-CG画像）とする．本研究ではデータセット3（位置-CG画像）として，計24,000枚

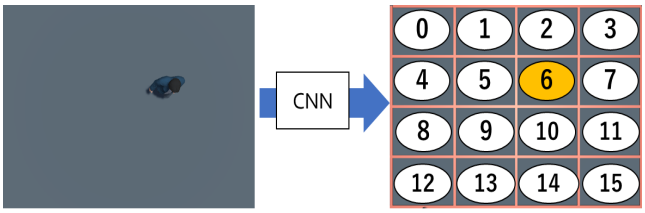

図 3 位置推定に使用した CG 画像の例

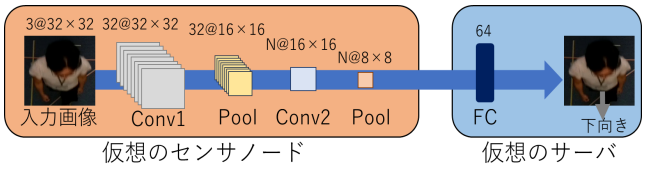

図 4 単体のセンサノード-サーバ間で情報を処理する CNN の例

(1,500 枚× 16 クラス) を用意した．本実験ではこのデータセット 3 (位置-CG 画像) を，学習用画像として 16,000 枚 (1000 枚× 16 クラス)，検証用画像として 3,200 枚 (200 枚× 16 クラス)，テスト用画像として 4,800 枚 (300 枚× 16 クラス) に分けて使用した．

4.4 実験内容

本実験では，中間層の畳み込み数と識別精度の関係性について検証するために，意図的に畳み込み数を削減した CNN を作成し，データセット 1 (進行方向-実画像)，データセット 2 (進行方向-CG 画像) に対して人の進行方向 (8 方向) と人が存在しないという計 9 クラスに識別する場合の識別精度を検証した．同様に，提案する CNN がデータセット 3 (位置-CG 画像) に対し，CG モデルが存在する区画 (16 クラス) を識別する場合の精度も検証した．

実験で使用した CNN では，図 4 に示すように畳み込み層 (Conv) 2 層の後に 64 次元の全結合を行う層を積み重ねたものである．CNN の実装には Keras[*2] を利用した．図中の@の前の数字はチャネル数 (ch) であり，後の数字はデータのサイズである．入力画像は Keras の仕様によって自動的に 32 × 32 にリサイズされる．各畳み込み層では 3 × 3 のサイズのフィルタを用いて畳み込みを行なった．また，各畳み込み層の後に 2 × 2 の max-pooling を行うプーリング層 (Pool) を導入した．そのため，プーリング層後の特徴量の総データ量は，直前の畳込み層後の特徴量のデータ量の 1/4 になる．N は二層目の畳み込みフィルタの数であり，N の数によって仮想のセンサノードの送信データ量が変化する．例えば，入力データ量を 32 × 32 × 3 (ch) × 8 (bit) (=24,576 bit) とした場合，特徴量は 32-bit の float 型で扱われるため，仮想のセンサノードの送信データ量は 8 × 8 × N (ch) × 32 (bit) (=2048 × N bit) となる．各畳み込み層の活性化関数には ReLU (Rectified Linear Unit)，出力層の活性化関数には softmax 関数を用いた．また，誤差関数には多クラス交差エントロピー誤差を用いた．

本実験では N を 1，2，4，6，8，10，12，32 に設定した CNN を，それぞれデータセット 1 (進行方向-実画像) とデータセット 2 (進行方向-CG 画像)，データセット 3 (位置-CG 画像) の学習用画像で 1,000 epoch 学習し，テスト画像に対する識別精度を検証した．N=32 のときは送信データ量が入力データ量よりも増えてしまうが，これは畳み込み数を減らさなかった場合として比較するために精度を検証した．本実験では各 N に設定した CNN を初期値を変えて 10 試行学習し，それぞれ 10 試行分の識別精度の平均値を算出した．

[*2] Chollet Franccois et al. Keras. `https://keras.io`, 2015.

4.5 実験結果

以下の実験結果は，図 4 の CNN の二層目の畳み込み数 N を 1，2，4，6，8，10，12，32 に設定して学習し，テスト画像に対する識別精度を測定することを初期値を変えて 10 試行し，それぞれの N に対して識別精度の平均値をグラフ化したものである．グラフには標本標準偏差も掲載した．また，各 N のときの入力データ量に対する送信データ量も載せている．横軸は N の数，左の縦軸はテスト画像に対する識別精度，右の縦軸は送信データ量の圧縮率である．ここで識別精度と圧縮率は以下で計算した．

$$\text{識別精度 (\%)} = \frac{\text{正しく識別された画像 (枚)}}{\text{全テスト画像 (枚)}} \times 100$$

$$\text{圧縮率 (\%)} = \left(1 - \frac{\text{送信データ量}}{\text{入力データ量}}\right) \times 100$$

例えば N=12 のときは，入力データ量と送信データ量は等しいため圧縮率は 0%となり，N=1 の時の圧縮率は 91.7%となる．以降は単に識別精度，圧縮率といった場合，これらのことを示す．

図 5 はデータセット 1 (進行方向-実画像) で実験したときの結果である．N=1 の識別精度は最も低く 88.4%であり，N=32 の時の識別精度が最も高く 97.1%であった．

図 6 はデータセット 2 (進行方向-CG 画像) で実験したときの実験結果である．N=1 の識別精度は最も低く 98.4%であり，N=32 の時の識別精度は最も高く 99.7%であった．

図 7 はデータセット 3 (位置-CG 画像) で実験したときの結果である．N=1 の識別精度は最も低く 98.4%であり，N=32 の時の識別精度が最も高く 98.6%であった．また，すべての場合において，識別精度が 98.4%を超えた．

4.6 考察

データセット 1〜データセット 3 に対する実験結果から，CNN の畳み込み数を減らして送信データ量を低減すると，畳込み数の削減の割合に応じて識別精度が低下することが

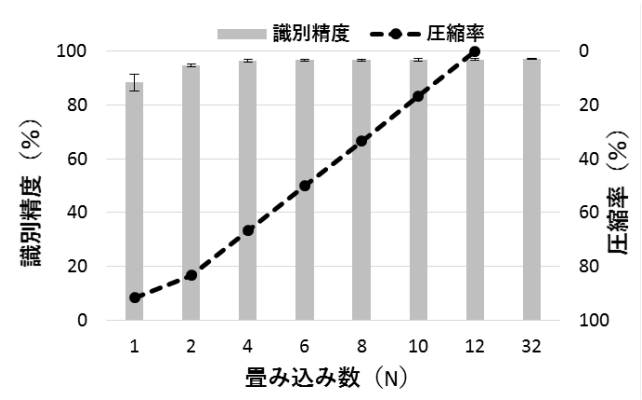

図 5　データセット 1 に対する実験結果

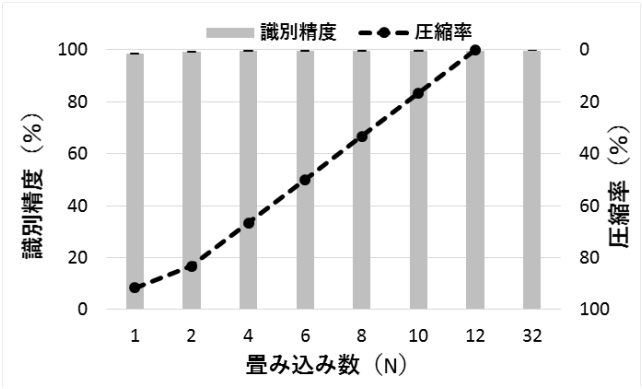

図 6　データセット 2 に対する実験結果

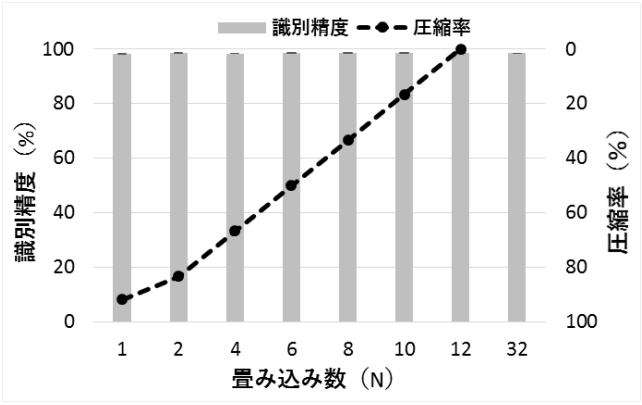

図 7　データセット 3 に対する実験結果

分かった．また，データセット 2 に対する実験結果を見ると，データセット 1 に対する実験結果と同様に，畳み込み数を減らすと識別精度が低下した．このように，CG 画像を利用した場合の実験結果は現実のデータセットに対して行う場合と同様の傾向が得られるため，今後の実験において CG 画像を利用することに問題はないと考える。

個々の実験結果を見ると，データセット 1 に対する実験結果は N=4 と N=32 を比較したときに精度の差が 0.6%程度であった．また，データセット 2 とデータセット 3 に対する実験結果は N=4 と N=32 の精度を比較したときにどちらも精度に 1%以上の差はなかった．このことから，人の進行方向と位置を推定する場合であれば，送信データ量を入力の 1/12 まで低減することが可能であると考える．ここで，センサノード上で何も処理をしない場合のデータ量は（640 × 480 × 1 (Byte) × 3 (ch) =）921,600 (Byte) であり，N=4 のときの送信データ量が（8 × 8 × 4 (Byte) × 4 (ch) =）1024 (Byte) であるため，送信データ量を提案手法によって 1/900 に低減した場合，96%以上の精度で人の進行方向や位置を推定することが可能であると考えられる．

5. 特徴量の統合処理による影響を検証

本実験では第 3.2 節に述べるように，特徴量の統合処理が精度にどのような影響を与えるかを検証するため，単体と 2 個のセンサノード-サーバ間でデータを処理する場合の識別精度を比較した．以降は検証に利用した 4 種類のデータセットと，そのデータセットを分類するために作成した CNN について述べる．実験は実際のセンサノードとサーバは使用せずに，一つの PC 上で仮想的に行なった．

5.1 統合処理実験用データセット（データセット 4）

本実験では後に述べる CNN を用いて，画像を進行方向（8 方向）と人が映っていないという計 9 クラスに識別するタスクによって統合処理の影響を検証した．進行方向を推定するデータセット作成のために，図 8 に示すように，1 体の CG モデルの正面と背面に等距離にカメラを設置し，それぞれの位置から CG モデルが歩行する様子を撮影して作成した．利用した CG モデルは第 4 章でも利用した男女計 6 体である．

本稿ではデータセット作成のための環境として，両方のカメラと CG モデルの間に遮蔽物がない場合（以下壁なし）と，カメラ 1 と CG モデルの間に遮蔽物がある場合（以下壁あり）の計 2 パターンに設定して撮影を行なった．壁ありの環境を作るために，図 8 と全く同じ環境に 6 体すべての CG モデル以上の高さを持つ CG の壁を 1 つ設置した．これによってカメラ 2 からは CG モデルが完全に見えるが，カメラ 1 からは CG モデルが完全に見えないという状況を作り出した．

我々はデータのバリエーションを増やすために，CG モデルの初期位置を図 8 に示した位置だけではなく，左右に 50cm ずつずらした位置でも撮影を行なった．ここで壁ありのデータセット作成時には，CG の壁も CG モデルと同じだけ動かすことで，カメラ 1 からは CG モデルが完全に見えないようにした．また，モデルの移動によって拡縮が生じると精度に影響を与える可能性があるため，本実験では CG モデルを初期位置に固定して撮影した．そのため CG モデルが初期位置から移動せずに歩行モーションのみを行う状況を撮影した．

データのラベル付けを行う際には，カメラ 1 とカメラ 2

図 8　2 台のカメラで撮影する例

が撮影する映像から同時刻のフレームを選択し，CG モデルの進行方向を基に「左斜め上」～「人が映っていない」という 9 クラスの中から 1 つに設定した．ここで「人が映っていない」画像については，撮影した映像から CG モデルが映っていないフレームを選択した．カメラ 2 の画像をラベル付けするにあたり，カメラ 1 からの CG モデルの向きをラベルとして与えた．そのため，カメラ 1 とカメラ 2 で同時刻に同一ラベルがついた画像は，CG モデルの見え方が点対称になる．本稿ではカメラ 1，2 から壁なし設定で作成したデータセットを 4-A（カメラ 1-壁なし），4-B（カメラ 2-壁なし）とし，壁あり設定で作成したデータセットを 4-C（カメラ 1-壁あり），4-D（カメラ 2-壁あり）とする．

本稿ではデータセット 4-A（カメラ 1-壁なし）～4-D（カメラ 2-壁あり）として，それぞれ計 6,318 枚（702 枚× 9 クラス）ずつ用意した．本実験ではそれぞれのデータセットを，学習用画像として 4,518 枚（502 枚× 9 クラス），検証用画像として 900 枚（100 枚× 9 クラス），テスト用画像として 900 枚（100 枚× 9 クラス）に分けて使用した．

5.2　実験内容

本実験では，単体のセンサノード-サーバ間でデータを処理して，画像を人の進行方向（8 方向）と人が存在しないという計 9 クラスに分類する場合と，複数のセンサノード-サーバ間でデータを処理して，画像を 9 クラスに分類する場合で識別精度を比較した．以下に単体のセンサノード-サーバ間で処理を行う場合に使用した CNN と，複数のセンサノード-サーバ間で処理を行う場合に使用した CNN について述べる．

5.2.1　単体の処理を検証するための CNN

対照条件とする単体のセンサノード-サーバ間の処理に使用した CNN は，図 4 で使用したものとと全く同じ構造のものである．本実験では N を 2，4，6，8，10，12，32 に設定した CNN を，データセット 4-A（カメラ 1-壁なし）～データセット 4-D（カメラ 2-壁あり）の学習用画像で 1000 epoch 学習した．そして，学習したデータセットのテスト画像に対する識別精度を検証した．例えば，データセット 4-A（カメラ 1-壁なし）の学習用画像を使用して CNN を学習した場合，データセット 4-A（カメラ 1-壁なし）のテスト画像に対する識別精度を求めた．N=32 のときは送信データ量が入力データ量よりも増えてしまうが，これは畳み込み数を減らさなかった場合として比較するために精度

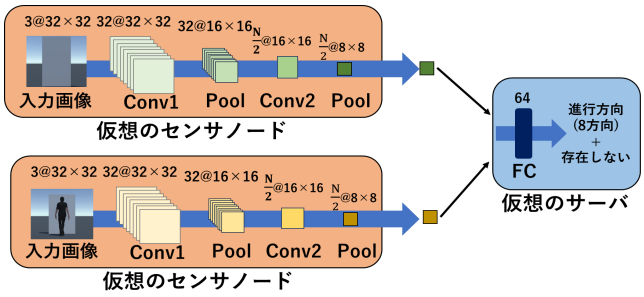

図 9　2 個のセンサノードの情報を統合処理する CNN の例

を検証した．本実験では各 N に設定した CNN を，初期値を変えて 10 試行学習し，それぞれの CNN に対し識別精度の平均を算出した．

5.2.2　統合処理を検証するための CNN

複数のセンサノード-サーバ間でデータを処理するために，図 9 に示すような CNN を作成した．この CNN はまず，2 個の仮想のセンサノードから仮想のサーバにデータを送信する．そしてこの仮想のサーバは複数のセンサノードから得たデータを統合的に処理して進行方向（8 方向）＋人が存在しないという 9 クラスに分類する．図中の@の前の数字はチャネル数（ch）であり，後の数字はデータのサイズである．それぞれの仮想のセンサノードの入力画像は 32 × 32 のサイズである．各畳み込み層では 3 × 3 のサイズのフィルタを用いて畳み込みを行なった．また，各畳み込み層の後に 2 × 2 の max-pooling を行うプーリング層（Pool）を導入した．

本実験ではそれぞれのセンサノード上の処理として，畳み込み層（Conv）2 層を積み重ねた．図 9 中の N は 2 層目の畳み込み層（Conv2）の畳み込み数である．本実験では単体のセンサノード-サーバ間で CNN の計算プロセスを処理する場合とネットワーク上の送信データ量を揃えるために，Conv2 の畳み込み数を N/2 に設定した．例えば，N=2 の時は，単体のセンサノード-サーバ間の送信データ量は 8 × 8 × 2（ch）× 32（bit）となり，2 個のセンサノード-サーバ間の処理では，送信データ量は 8 × 8 × 1（ch）× 32（bit）× 2（センサノード数（個））となる．また仮想のサーバ側の処理として，2 個のセンサノードから得られる計 N 枚の特徴量に対して 64 次元の全結合を行う全結合層を積み重ねた．各畳み込み層の活性化関数には ReLU（Rectified Linear Unit），出力層の活性化関数には softmax 関数を用いた．また，誤差関数には多クラス交差エントロピー誤差を用いた．

本実験では N を 2，4，6，8，10，12，32 に設定し，データセット 4-A（カメラ 1-壁なし）～データセット 4-D（カメラ 2-壁あり）を利用して 1000 epoch 学習した．学習時には図 9 の CNN に対し，2 種類のデータセットの学習用画像を 1 組として同時に入力し，9 クラスに分類した．ここで本稿では，同時に入力するデータセットの組として 2

組用意した．1 組目はデータセット 4-A（カメラ 1-壁なし）とデータセット 4-B（カメラ 2-壁なし），2 組目はデータセット 4-C（カメラ 1-壁あり）とデータセット 4-D（カメラ 2-壁あり）である．学習後には，学習した組のテスト画像に対する識別精度を検証した．例えば，データセット 4-A（カメラ 1-壁なし）とデータセット 4-B（カメラ 2-壁なし）の組で CNN を学習した場合，データセット 4-A（カメラ 1-壁なし）とデータセット 4-B（カメラ 2-壁なし）のテスト画像を同時に入力して 9 クラスに分類する際の識別精度を求めた．本実験では各 N に設定した CNN を，初期値を変えて 10 試行学習し，それぞれの CNN に対し 10 試行分の識別精度の平均を算出した．

5.3 実験結果

以下の実験結果は単体の CNN と統合処理のために作成した CNN の，二層目（Conv2）の畳み込み数 N を 2，4，6，8，10，12，32 に設定し，それぞれの N に対して 10 回初期値を変えて学習し，テスト画像に対する識別精度の 10 試行分の平均値をグラフ化したものである．横軸は N の数であり，左の縦軸はテスト画像に対する識別精度の 10 試行の平均値である．右の縦軸は，単体のセンサノードからサーバにデータを送信する場合の送信データ量の圧縮率である．グラフには標本標準偏差を載せている．

図 10 中の single1，single2 はそれぞれ単体の CNN をデータセット 4-A（カメラ 1-壁なし），データセット 4-B（カメラ 2-壁なし）で学習とテストを行った結果であり，Multi は統合処理用の CNN に対し 2 つのカメラからのデータセットを用いて学習とテストを行った結果である．N=2 の時の Single 1，Single 2，Multi の識別精度の平均値はすべて 99.9%であり，N=32 の時の識別精度の平均値はすべて 100.0%である．また，すべての N において識別精度が 99%を超える．

図 11 中の single1, single2 はそれぞれ単体の CNN をデータセット 4-C（カメラ 1-壁あり），データセット 4-D（カメラ 2-壁あり）で学習とテストを行った結果であり，Multi は統合処理用の CNN に対して 2 つのカメラからのデータセットを用いて学習とテストを行った結果である．N=2 の時の Single 1，Single 2，Multi の識別精度の平均値はそれぞれ，11.1%，99.8%，99.5%である．また，N=32 の時の識別精度の平均値はそれぞれ，10.9%，100.0%，100.0%である．

5.4 考察

図 10 の結果から分かるように，CG モデルを遮蔽物なく真横から撮影した場合には，単体のセンサノードと 2 個のセンサノードを利用する場合で識別精度に差がなく，99%以上で識別が行えた．図 11 の Single1 の実験結果では，すべての場合において識別精度が 11%程度であったが，この状

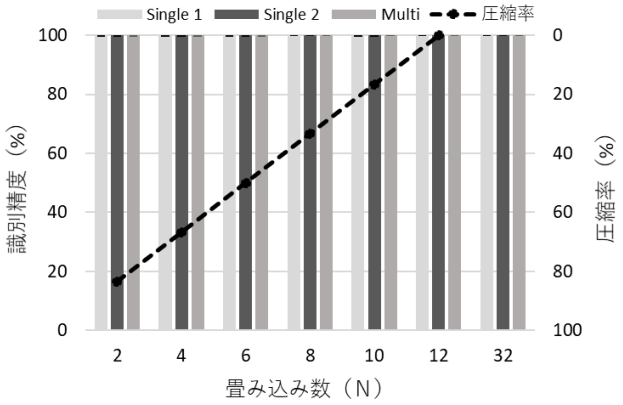

図 10 データセット 4-A，4-B に対する実験結果

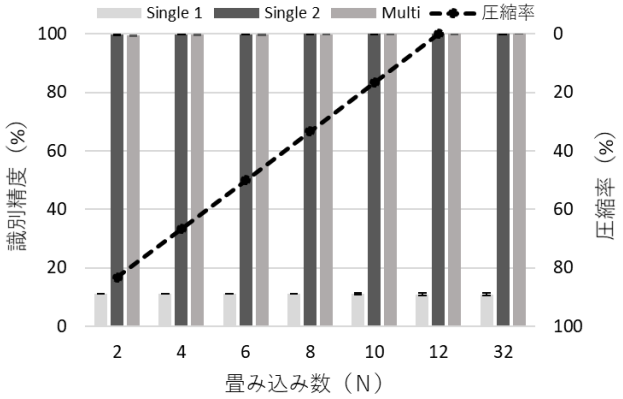

図 11 データセット 4-C，4-D に対する実験結果

況でカメラ 1 には人は映っておらず，画像をすべて「人がいない」というクラスに識別し，「人がいない」クラスのみが正解したためこの値になったと考えられる．

図 11 内の Single 1, Single 2, Multi を比較すると，Single 1 は識別精度が 11%程度であったが，Multi の識別精度は Single 2 と同様に 99%以上の精度であった．このことから Multi は，カメラ 2 から得たデータを主に利用して識別を行なったと考える．以上の結果から，複数のセンサノード-サーバ間でデータを統合的に処理する場合の識別精度は，個々のセンサノード-サーバ間でデータを処理する場合の識別精度の中で，最も高い識別精度に近づくと考える．

以上から，本稿が想定する環境のように大規模なイベント会場に複数のセンサノードを仮設して人物追跡を行う場合，複数のデータをサーバで統合的に処理することによって人物への遮蔽による影響を自動的に低減することが可能なため，複数のセンサノードで圧縮した CNN の途中の層のデータをサーバで統合して処理することにより，細かな設定を行わずとも，高い方の識別精度に沿って画像識別を行える可能性が示せた．

6. 終わりに

本稿では，大規模なイベント会場に複数のセンサノード

を仮設し，映像だけから人物追跡を行うことが可能な通信システムの構築を目標として，複数のセンサノードから映像を同時にサーバに送信する場合に生じる帯域幅への負担を低減するため，意図的に畳み込み数を減らした CNN の計算プロセスをセンサノードとサーバで分散処理する手法を提案した．意図的に畳み込み数を減らした CNN の識別精度を実験によって検証したところ，送信データ量をセンサノード上で何も処理しない場合と比較して 1/900 に低減した場合，人の進行方向推定を行う場合であれば 96%程度，人の位置推定を行う場合であれば 98%程度の精度で推定可能であった．ただし，本稿では人の位置については CG で作成した画像に対してしか検証を行なっていない．現実の画像の方が CG の画像よりも光などによってノイズが生じるため，本稿で検証した結果よりも精度が低下する可能性がある．また，人物の位置識別で使用した CG のデータセットでは，人以外にオブジェクトが存在していない．実際のイベント会場ではパネルや展示物といったオブジェクトが映像中に映ることが考えられるため，今後はこれらのオブジェクトが映った場合でも人の位置を検出できるかを検討する必要がある．加えて，本稿では人の進行方向推定と人物位置推定を独立に行なったが，今後は人の位置と進行方向を同時に推定する場合についての識別精度も検証する必要がある．さらに，本稿では複数のセンサノードが共通の視野を持つことを前提に同一人物をつなげていくことで人物追跡を行うことを想定しているが，実際のイベント会場ではセンサノードどうしが視野を共有していないことも考えられる．そこで，そのような場合には時系列情報を利用し，あるセンサノードの撮影範囲から出た人が，異なるセンサノードの撮影範囲に現れたという情報をつなぎ合わせることで，同一人物の移動経路を取得することが可能になると考える．以上の検証を通して，最終的には大規模なイベント会場にセンサノードを仮設するだけで人物追跡を簡易に行うことを可能にする．

参考文献

[1] 加藤博一, 中澤篤志, 井口征士. 楕円体モデルを用いたリアルタイム人物追跡. 情報処理学会論文誌, Vol. 40, No. 11, pp. 4087–4096, 1999.

[2] 森大樹, 内海章, 大谷淳, 谷内田正彦, 中津良平. 非同期多視点画像による人物追跡システムの構築. 電子情報通信学会論文誌 D, Vol. 84, No. 1, pp. 102–110, 2001.

[3] 石塚宏紀, 戸辺義人. カメラセンサネットワークにおける地理位置情報を利用した経路制御機構の設計と実装. 情報処理学会論文誌, Vol. 49, No. 6, pp. 1907–1919, 2008.

[4] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object Recognition with Gradient-Based Learning. In *Shape, contour and grouping in computer vision*, pp. 319–345. 1999.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in neural information processing systems (NIPS)*, pp. 1097–1105, 2012.

[6] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[7] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 770–778, 2016.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 37, No. 9, pp. 1904–1916, 2015.

[10] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in neural information processing systems (NIPS)*, pp. 91–99, 2015.

[11] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2016.

[12] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single Shot Multibox Detector. In *European conference on computer vision (ECCV)*, pp. 21–37. Springer, 2016.

[13] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deepreid: Deep Filter Pairing Neural Network for Person Re-Identification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 152–159, 2014.

[14] Schroff F, Kalenichenko D, and Philbin J. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823, 2015.

[15] De Cheng, Yihong Gong, Sanping Zhou, Jinjun Wang, and Nanning Zheng. Person Re-Identification by Multi-Channel Parts-Based CNN with Improved Triplet Loss Function. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1335–1344, 2016.

[16] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal*, Vol. 3, No. 5, pp. 637–646, 2016.

[17] Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. Branchynet: Fast Inference via Early Exiting from Deep Neural Networks. In *International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469, 2016.

[18] Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. Distributed Deep Neural Networks over the Cloud, the Edge and End Devices. In *International Conference on Distributed Computing Systems (ICDCS)*, pp. 328–339, 2017.