特徴量損失の少ないボクセル表現を用いた三次元点群による物体検出*

Voxel Representation with low feature loss for 3D Object Detection from Point Clouds

重中 亨介 Ryosuke SHIGENAKA
米司 健一 Kenichi YONEJI
塚田 明宏 Akihiro TSUKADA

In this paper we propose a novel method that can operate at high speed for 3D object detection using LiDAR point clouds. In the detection method in which the point cloud is expressed by voxels and a convolutional neural network is applied, the processing speed can be increased by increasing the grid size of the voxels, but the detection accuracy tends to decrease. To solve this problem, we propose a novel voxel feature expression with little loss of point cloud features. As a result of the evaluation, the processing speed was increased from 32Hz to 64Hz while maintaining the detection accuracy.

Key words :
Autonomous driving, LiDAR perception, 3D object detection, Deep neural networks

特集

1. はじめに

都市が世界中で成長するにつれて車両の交通量は増加し続けており，交通事故や温室効果ガスによる環境汚染などが社会問題となっている．これに対して，自動運転技術は次世代モビリティの中核を担う技術として，交通事故の削減だけでなく効率的な運転による温室効果ガスの削減にも寄与すると期待されている．

近年，自動運転の実現に向けて LiDAR によって得られる点群データから Deep Neural Network による深層学習を用いて三次元物体検出を行なう技術が盛んに研究されている．点群データから深層学習を用いて三次元物体検出を行なう手法は，PointRCNN[1] のように点群から特徴量を直接抽出する方式と VoxelNet[2] のように点群をボクセルに変換してから特徴抽出を行なう方式に分けることができるが，後者のほうが一般的に高速に処理できるためリアルタイム処理が求められる自動運転には向いている．ボクセルに基づく方式では，点群をグリッドで分割した後に Voxel Feature Encoding (VFE)[2] などによってボクセルを表現し，カメラ画像と同様に畳み込みニューラルネットワーク (CNN) を用いて物体検出が行なわれる．PointPillars[3] では，処理負荷をさらに軽減するためにボクセルのグリッドサイズを大きくすることでボクセルマップの解像度を落とすことが提案されている．このようにボクセルを用いる手法ではグリッドサイズを調整することで容易に処理を高速化できるが，グリッドサイズを大きくすると点群情報の損失が生じるため，特に小物体に対して検出精度が低下しやすいという課題があった．

本報告では，この課題に対して特徴量の損失が少な

*情報処理学会 CVIM 研究会の了承を得て，第 23 回画像の認識・理解シンポジウムの予稿集 IS2-1-8 より一部加筆して転載

いボクセル表現方法を提案する．具体的には，VFE に対して Hu らによって提案された注意機構 [4] と global average pooling を組み合わせることで特徴量の損失を抑えて検出精度を維持しながらグリッドサイズを大きくすることを可能にする．さらに，FPN[5] のような CNN を用いることで処理の効率化を実現する．

2. 関連研究

2.1 点群ベース手法

ピクセルが規則的にグリッド上に配置されているカメラ画像とは異なり，点群は不規則かつ順不同なデータであるため一般的な CNN を適用することが難しく，従来では点群を直接扱う手法はコンピュータビジョンの分野において多くの注目を集めてこなかった．しかし，PointNet[6] の登場によって近年では点群を直接扱う手法が提案されている [1) 8)]．

これらの手法では PointNet++[7] を backbone に用いることで点群を直接ニューラルネットワークに入力できるため，前処理による点群情報の損失がなく，高い検出精度を実現できる．しかしながら，PointNet++ は処理負荷が大きいために処理が遅いという課題がある．

2.2 ボクセルベース手法

点群をグリッド分割してボクセルに変換することで，画像と同様に CNN を適用可能にする手法も提案されている [2) 3) 9)]．このとき，ボクセルは点群の平均値 [9] や VFE に基づく特徴量 [2] によって表現され，いずれも軽量な演算によって点群が空間解像度の低いボクセルに変換されるため，点群を直接扱う手法よりも処理時間を短縮することが可能である．例えば PointPillars[3] では，0.16m × 0.16m の柱型のボクセルに対して 1 層のみの VFE による特徴量を用いることで 62Hz の処理速度を実現している．さらに，ボクセルサイズを 0.28m × 0.28m に拡大することで処理速度を 105Hz に高速化できることも示している．しかしながら，Fig. 4 に示すようにボクセルのグリッドサイズを大きくすると検出精度が低下するという課題があった．検出精度を上げるために，三種類の注意機構を適用する手法 [10] や物体境界に基づいた deformable convolution を用いる手法 [11] が提案されているが，いずれも計算負荷が大きいため処理速度が低下している．

本報告では，PointPillars をベースとして，処理速度を落とすことなく検出精度を向上する方法を提案する．

3. 提案手法

3.1 概要

提案手法における全体的な処理の流れを Fig. 1 に示す．ボクセル化 (Voxelization)，ボクセル特徴抽出 (voxel feature extraction)，2D CNN，detection head の 4 つのパーツから構成されるが，提案手法は PointPillars[3] をベースとしてボクセル特徴抽出と 2D CNN を改良した手法であるため，ボクセル化と detection head は PointPillars と同じ方法を用いる．

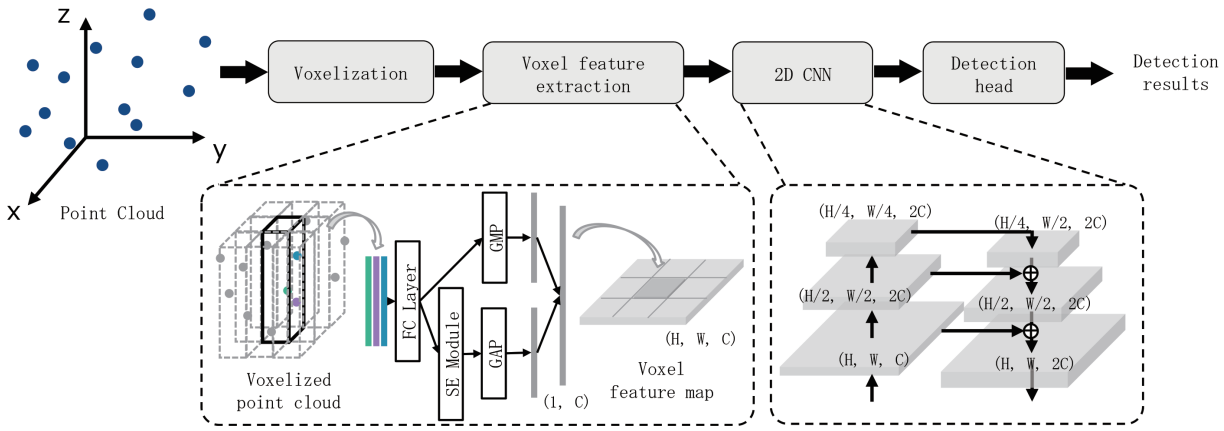


Fig. 1 Our method consists of four phases: voxelization, voxel feature extraction, 2D CNN and detection head. GMP, GAP, FC Layer and SE Module stand for global max pooling, global average pooling, fully-connected layer and channel attention[4], respectively

3.2 準備

ここでは，次節で説明するボクセル特徴抽出の前処理となる点群のボクセル化について述べる．

まず点群を $\mathbf{P}=\{\boldsymbol{p}_i=[x_i, y_i, z_i, r_i]^{\mathrm{T}}\}_{i=1,\cdots,M}$ とする．x_i, y_i, z_i はそれぞれ X, Y, Z 軸における i 番目の点群の座標値であり，r_i は i 番目の点群の反射強度である．また，M は点群数を表す．

次に点群 P が三次元空間の X, Y, Z 軸に対してそれぞれ W^*, H^*, D^* の範囲内に分布しているとするとき，X, Y, Z 軸に対してそれぞれ v^*_W, v^*_H, v^*_D のグリッドを用いて点群を $W=W^*/v^*_W$, $H=H^*/v^*_H$, $D=D^*/v^*_D$ のサイズのボクセルに分割することができる．なお，PointPillars[3] に従って Z 軸方向にはグリッドを分割せず v^*_D は D が 1 になるように設定する．またボクセルごとに点群数を N で揃え，点群は 9 次元の特徴に拡張する．

従って，k 番目のボクセル内の点群を $\mathbf{V}_k \in \mathbb{R}^{9\times N}$ $(k=[1,2,\cdots,H\times W])$ とすると，ボクセル化された点群全体 $\mathbf{V}$ はサイズ $(H,W,9,N)$ のテンソルとなる．

3.3 ボクセル特徴抽出

前節では点群をボクセルに分割する手順について説明した．ここでは各ボクセル内の点群 $\mathbf{V}_k \in \mathbb{R}^{9\times N}$ から特徴量 $\mathbf{f}_k \in \mathbb{R}^C$ を抽出することによって，点群 $\mathbf{V} \in \mathbb{R}^{N\times W\times 9\times N}$ からボクセルの特徴マップ $\mathbf{F} \in \mathbb{R}^{H\times W\times C}$ を出力することを考える．

PointPillars では，Fig. 2(a) に示すように全結合層と global max pooling (GMP) を用いてボクセル特徴を抽出している．ここで全結合層による特徴抽出は，δ を ReLU 関数，θ をバッチ正規化，$\mathbf{W} \in \mathbb{R}^{C\times 9}$ を全結合層における重みパラメータとすると以下の式で記述でき，point-wise な特徴 $\tilde{\mathbf{F}}_k \in \mathbb{R}^{C\times N}$ を抽出していると解釈できる．

$$\tilde{\mathbf{F}}_k = \delta\big(\beta(\mathbf{W}\mathbf{V}_k)\big). \tag{1}$$

そして，global max pooling では N 個の特徴量から 1 つの最大特徴量 $\mathbf{f}^*_k \in \mathbb{R}^C$ のみを取り出す．j 列目の特徴量 $\tilde{\mathbf{F}}_k$ を $\tilde{\mathbf{f}}_{k,j}$ $(j=[1,\cdots,N])$ とすると，以下の式で記述できる．

$$\mathbf{f}^*_k = \max_{j\in[1,\cdots,N]} \tilde{\mathbf{f}}_{k,j}. \tag{2}$$

従って，多数の点群情報はボクセル特徴抽出の過程で失われることとなる．特に，物体検出において重要な点群は物体境界に沿って局所的に分布する傾向があり，ボクセルのグリッドサイズが大きいと 1 つのボクセル内に重要な点が複数含まれる可能性が高くなるため，global max pooling による特徴量の損失によって検出精度の低下を引き起こす．

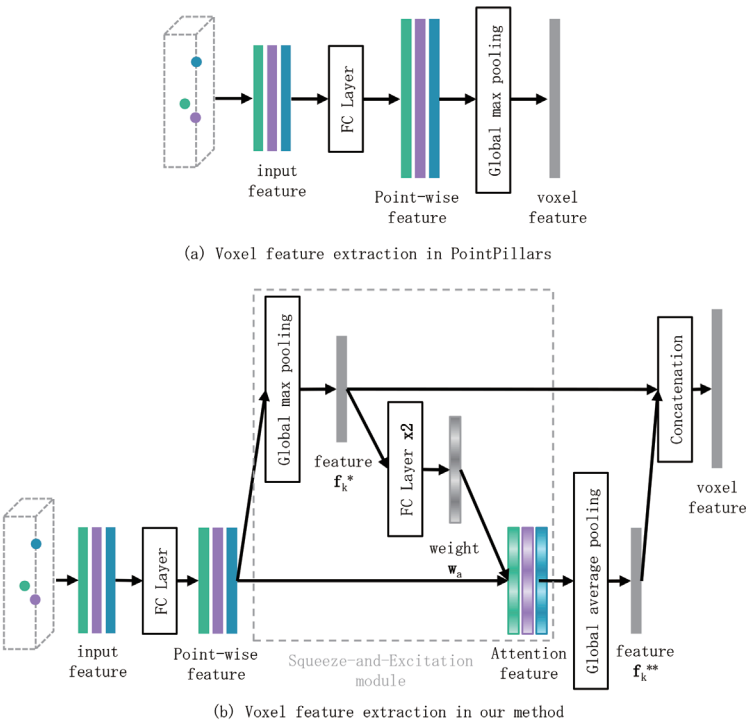


Fig. 2 Comparison of voxel feature extraction between PointPillars and our proposed method

提案手法では，Fig. 2(b) に示すように global average pooling (GAP) を追加することにより特徴量の損失を抑える．そのために，まず式 (1) における出力特徴次元数を $C'=C/2$ にしておき，式 (2) で得られる特徴 $\mathbf{f}_k^* \in \mathbb{R}^{C'}$ と global average pooling によって得られる特徴 $\mathbf{f}_k^{**} \in \mathbb{R}^{C'}$ を合わせた特徴が C 次元になるようにする．但し，global average pooling は活性値の小さい特徴も含めてすべての特徴量の平均を取るため，物体検出に不要な特徴まで抽出してしまう．そこで，不要な特徴を排除することを目的として注意機構を導入し，重み付け global average pooling とする．本来は point-wise に attention weight をかけることが望ましいが，点群数が多く処理負荷が大きくなるため channel-wise な注意機構 [4] を代用する．注意機構における重み $\mathbf{w}_\alpha$ は以下のように記述できる．

$$\mathbf{w}_a = \sigma\left(\mathbf{W}_2\left(\delta(\mathbf{W}_1\mathbf{f}_k^*)\right)\right). \tag{3}$$

ここで，σ はシグモイド関数を表す．$\mathbf{W}_1 \in \mathbb{R}^{C/8\times C}$ と $\mathbf{W}_2 \in \mathbb{R}^{C\times C/8}$ は注意機構における全結合層の重みパラメータであり，特徴次元数を 1／8 に圧縮してから元に戻す．そして，特徴量 $\tilde{\mathbf{F}}_k$ を重み $\mathbf{w}_\alpha$ で重み付けした後に global average pooling を適用する．j 列目の特徴量 $\tilde{\mathbf{F}}_k$ を $\tilde{\mathbf{f}}_{k,j}$，⊙を要素ごとの乗算とすると，以下のように記述できる．

$$\mathbf{f}_k^{**} = \frac{1}{N}\sum_{j=1}^{N} \mathbf{w}_a \odot \tilde{\mathbf{f}}_{k,j}. \tag{4}$$

最終的に k 番目のボクセル特徴量は二つの特徴量 $\mathbf{f}_k^*$ と $\mathbf{f}_k^{**}$ をチャネル方向に連結することで得られる．

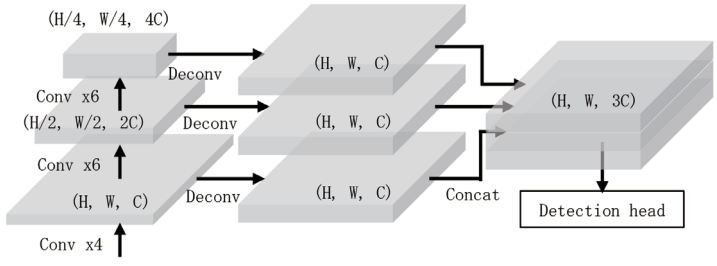
$$\mathbf{f}_k = \left[\mathbf{f}_k^{*\mathrm{T}}, \mathbf{f}_k^{**\mathrm{T}}\right]^{\mathrm{T}}. \tag{5}$$

3.4 2D CNN

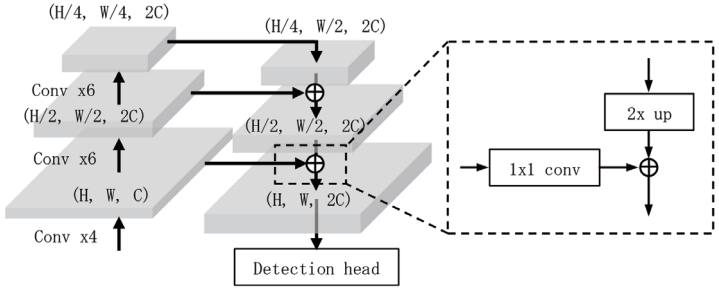
前節で説明したボクセル特徴の改良では演算量の増加に伴い処理負荷が大きくなるため，本節では CNN の簡略化による処理負荷の軽減を行なう．

PointPillars では，Fig. 3(a) に示すようにダウンサンプリングによって特徴マップの解像度を落としながら畳み込みを行ない，得られたマルチスケールの特徴マップを逆畳み込みによって同じ解像度に合わせてから統合している．これによってマルチスケールの特徴量を獲得することが可能であるが，逆畳み込みによるアップサンプリングは演算量が大きいためスケールごとに独立に処理するのは非効率である．

そこで提案手法では，Fig. 3(b) に示すようにダウンサンプリングされた特徴マップをアップサンプリングしながら，skip connection を用いてマルチスケール特徴量の統合を行なう．このとき，アップサンプリングには最近傍補間を用いるため，PointPillars における逆



(a) 2D CNN in PointPillars



(b) 2D CNN in our method

Fig. 3 Comparison of 2D CNN between PointPillars and our proposed method. 'Conv' , 'Deconv' , 'Concat' , 'up' and ⊕ stand for convolution, transposed convolution, concatenation, upsampling and element-wise summation, respectively

Table 1 Evaluation results on KITTI validation set. The evaluation metrics follow KITTI[12] where the evaluation is based on the difficulty level of Easy, Moderate (Mod.) and Hard. The mAP is the mean value of AP for all categories.

Method	Voxel grid size [m]	mAP	Car			Pedestrian			Cyclist		
		Mod.	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
PointPillars[3]	0.16	67.07	86.48	76.37	69.77	67.73	61.16	54.95	84.06	63.69	59.77
	0.28	60.91	81.73	70.79	67.63	55.86	51.10	47.68	78.90	60.85	57.02
Ours	0.16	68.45	87.49	77.02	70.21	68.31	62.52	58.69	84.01	65.82	61.74
	0.28	63.66	84.56	72.52	67.85	62.68	57.59	52.70	81.59	60.88	57.18

畳み込みよりも演算量が小さい．また，二つの特徴マップを足し合わせるために，提案手法ではカーネルサイズが 1 × 1 の軽量な畳み込みを用いてチャネル数を合わせている．これは FPN[5] に着想を得たものであるが，FPN とは異なり detection head は最も解像度の高い特徴マップにしか付けない．これは鳥瞰視点の特性を活かしたものであり，鳥瞰視点では物体のスケール変化が小さいため FPN のような複数の detection head は不要となる．

4. 評価実験

ここでは提案手法の有効性を検証するために，二つの実験を行なう．一つ目は，ボクセルのグリッドサイズが小さい場合 (0.16m) と大きい場合 (0.28m) でそれぞれ PointPillars と提案手法の検出精度を評価し，グリッドサイズが大きい場合に提案手法のほうが小物体に対して精度の低下が小さいことを示す．また二つ目の実験では，検出精度と処理速度の評価を行ない，同じ検出精度のときに提案手法のほうが PointPillars に対して高速であることを示す．なお，これらの実験は KITTI データセット[12] を PointPillars の評価実験[3] に従って train set と validation set に分割し，validation set を評価データとして使用した．学習に使用する誤差関数や最適化手法，データ拡張手法とその他のハイパーパラメータはすべて PointPillars[3] に従った．但し，ボクセル特徴の次元数は C=64 とした．また，本実験では Geforce GTX 1080Ti の GPU および Core i7-7700K の CPU を用いて，CUDA9.0 と cuDNN7.6.4.38 の環境において処理時間計測を行なった．

Table 1 に各対象物体における検出精度比較結果を示す．ボクセルのグリッドサイズが 0.16m と 0.28m のとき，PointPillars の mAP がそれぞれ 67.07% と 60.91% であるのに対して提案手法の mAP はそれぞれ 68.45% と 63.66% となり，どちらも提案手法のほうが高精度となった．特に Pedestrian に対して，グリッドサイズを 0.16m から 0.28m に大きくしたとき，PointPillars では Moderate の AP が 61.16% から 51.10% まで 10.06% 低下するのに対して提案手法では 62.52% から 57.59% までの 4.93% の低下に抑えられており，提案手法は小さい物体に対して精度低下を抑える効果が大きいことがわかる．

また，Fig. 4 にボクセルのグリッドサイズを 0.16m から 0.28m まで変化させた場合の処理速度と検出精度の推移を示す*．この結果より，すべてのグリッドサイズにおいて提案手法は PointPillars よりも高速かつ高精度であることがわかる．そして，グリッドサイズが 0.16m のときの PointPillars の精度は 62.42% であり，これはグリッドサイズが 0.20m のときの提案手法（62.87%）と同程度の検出精度である．このとき，処理速度は PointPillars が 32.09Hz であるのに対して提案手法は 45.51Hz であるため，提案手法は PointPillars に対して高速化したことになり，検出精度を維持しながら処理速度を向上することができた．

*PointPillars の論文中での処理速度は TensorRT を用いて計測したものであるため，今回の計測とは処理速度が異なる．

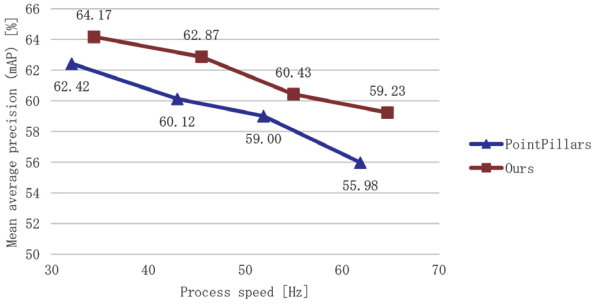


Fig. 4 Balance between processing speed and detection accuracy when voxel grid size is set to 0.16m, 0.20m, 0.24m, 0.28m, respectively

5. おわりに

本報告では，PointPillars をベースとしてボクセル特徴抽出 (voxel feature extraction) と 2D CNN を改良した手法を提案し，性能の比較実験を行なった．KITTI データセットにおいて，提案手法が高速かつ高精度であることを確認し，PointPillars を上回る処理速度を実現した．また，歩行者のような小物体に対して提案手法がより有効であることを示した．

参考文献

1) S. Shi, X. Wang, and H. Li：“PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud”, International Conference on Computer Vision and Pattern Recognition, (2019)
2) Y. Zhou and O. Tuzel：“VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection”, International Conference on Computer Vision and Pattern Recognition, (2018)
3) A. H. Lang, S. Vora, H. Caesar, L. Zhou, and J. Yang：“PointPillars: Fast Encoder for Object Detection from Point Clouds”, International Conference on Computer Vision and Pattern Recognition, (2019)
4) J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu：“Squeeze-and-Excitation Networks”, International Conference on Computer Vision and Pattern Recognition, (2018)
5) T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie：“Feature Pyramid Networks for Object Detection”, International Conference on Computer Vision and Pattern Recognition, (2017)
6) C. R. Qi, H. Su, K. Mo, and L. J. Guibas：“PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”, International Conference on Computer Vision and Pattern Recognition, (2017)
7) C. R. Qi, L. Yi, H. Su, and L. J. Guibas：“PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”, Neural Information Processing Systems, (2017)
8) Z. Yang, Y. Sun, S. Liu, and J. Jia：“3DSSD: Point-based 3D Single Stage Object Detector”, International Conference on Computer Vision and Pattern Recognition, (2020)
9) S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li：“From Points to Parts: 3D Object Detection from Point Cloud with Part-aware and Part-aggregation Network”, IEEE Trans. Pattern Analysis and Machine Intelligence, (2020)
10) Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai：“TANet: Robust 3D Object Detection from Point Clouds with Triple Attention”, AAAI Conference on Artificial Intelligence, (2020)
11) G.Xu, W. Wang, Z. Liu, L. Xie, Z. Yang, H. Liu, and D. Cai：“Boundary-Aware Feature Indicator for Single-Shape 3D Object Detection from Point Clouds”, arXiv, (2020)
12) A. Geiger, P. Lenz, and R. Urtasun：“Are you ready for Autonomous Driving? The KITTI Vision Benchmark Suite”, International Conference on Computer Vision and Pattern Recognition, (2012)

著者

重中 亨介
しげなか りょうすけ

AI 研究部
LiDAR 認識関連の要素技術開発に従事

米司 健一
よねじ けんいち

AI 研究部
画像認識関連の要素技術開発に従事

塚田 明宏
つかだ あきひろ

AI 研究部
コンピュータビジョン／画像認識関連の要素技術開発に従事