生活支援ロボットによる物体配置タスクにおける危険性予測および視覚的説明生成

○畑中駿平 * †，上田雄斗 * †，植田有咲 †，平川翼 ‡，山下隆義 ‡，藤吉弘亘 ‡，杉浦孔明 †
† 慶應義塾大学　　‡ 中部大学

1. はじめに

少子高齢化社会では，今後さらに介助者の人手不足が問題となると予想される．高齢者や障碍者を支える人手不足の解決策として家庭用の生活支援ロボットは有望視されており，活発に研究が行われている．生活支援ロボットにとって物体配置は基本的動作の一つであるため，配置における高い安全性が求められる．

本研究では，生活支援ロボットが物体配置を行う際の，物体-ロボット-環境間の衝突に関する物理的推論 (physical reasoning) タスクを扱う．ここでは，軽微な接触の連鎖から生じる危険な衝突も含む．例えば，食器が置かれているテーブルに物体を置く場合における，食器と物体との衝突や，ロボットの腕と食器との衝突の推論を扱う．環境内に複数の物体が存在する場合，物理的相互作用の連鎖を考慮し，衝突を予測をすることは難しい．実際に 2D 環境であっても物理的相互作用の連鎖の予測精度は不十分である [1].

そこで本研究では，衝突確率の低い安全な領域の候補を可視化する手法を提案する． 図 1 に提案手法の概略を示す．提案手法では対象物体と配置領域の画像のみから物体配置における衝突確率を推定することができる．また，提案手法を用いることで家庭用ロボットの物体配置における衝突確率と安全領域を事前にユーザに提示することができ，それに対する判断を仰ぐことも可能となる．

既存手法の Transformer PonNet [2] では衝突に関連する部分に着目した可視化を行っているが，安全領域と危険領域が混合して可視化されていたため，安全である領域のみを可視化することができなかった．また，既存手法では RGB と Depth に対してそれぞれ別個の Attention Branch を持っていたため，パラメータ利用の効率が悪かった．本研究の新規性は以下である．

- 平面検出 [3] と Attention map を組み合わせることにより，衝突確率の低い安全な領域の候補を可視化する．
- RGB と Depth の Attention Branch および Transformer Perception Branch のパラメータ共有することによりパラメータ利用を効率化する．

2. 関連研究

物体操作分野のサーベイ論文として, [4] があげられる．[4] では，視覚を元としたロボット把持における物体姿勢推定, 物体検出, 物体把持タスクの紹介, 従来手法と RGBD 画像入力に基づいた最先端の手法について比較検討を行なっている．[5] ではデータ駆動型の把持

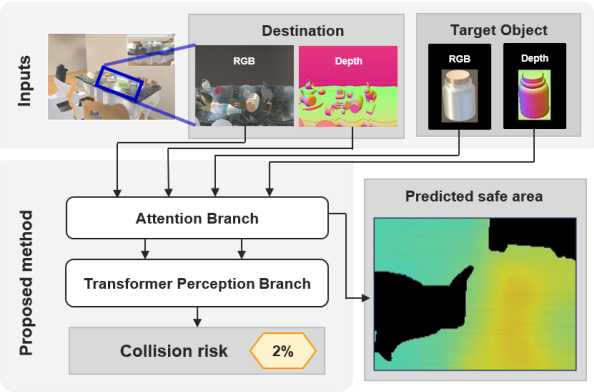

図 1　提案手法の概略

動作計画に関する研究紹介を行なっている．[6] では複数台のロボットからのデータを組み合わせることで, 画像のみから未知物体の把持に成功しており, リアルタイムでの効率的な把持学習が可能であることを示している．

物体操作分野は物体把持, 物体配置分野にカテゴリ化することができる．把持分野の研究として，[7] ではソフトハンドを用いて未知の物体を把持するための 3 次元畳み込みニューラルネットワークを用いた手法を提案している．配置分野の研究として，[8] ではサンプリングベースの運動計画と局所最適化を MCTS(Monte Carlo tree search) に基づいた階層的サンプリングアルゴリズムと組み合わせることで, 把持した物体を安定した配置領域に移動させるための動作アルゴリズムの提案を行なっている．[9] では家庭内での物体操作タスクを対象としており，未知単語を含む音声入力と画像入力から未知物体について学習する手法の提案を行っている．物体配置タスクにおいて PonNet では安全な物体配置を行うため，物体の衝突可能性を推定する研究を行っている [10]．また，[2] では PonNet を大きさが未知の対象物体についても扱えるように改良し, Perception Branch 部分に Transformer を組み込んだ Transformer PonNet の提案を行っている．

物体操作では空間内の物体位置や物体間の物理的な相互作用を理解し, 推論する必要がある．物体の力学的側面を考慮しているデータセットとして PHYRE [1] があげられる．PHYRE は 2D 環境内のパズルで構成されており, パズル間の相互作用を考慮したデータセットである．YCB データセット [11] は日常生活で利用される様々な物体の高解像度の RGBD スキャン, 物理的特性, 幾何学的モデルを含んでおり, シミュレーション環境で容易に 3D モデルを利用することが可能なデータセットである．

*両者は同等に貢献した.

図 2　対象タスクの例．左：実験環境．右：カメラ画像

3.　問題設定

本論文は，対象物体を指定した場所に配置する際の物体間の物理的推論 (physical reasoning) タスクを対象とする．本タスクでは，対象物体と配置領域の画像から，衝突確率が高い場合には衝突，低い場合には接触と予測することが望ましい．その際に，衝突確率の高い領域と，衝突確率の低い安全な領域に注目した可視化を行う．

図 2 に本対象タスクの代表例を示す．図 2 の状況において，配置領域に対象物体を置く際の衝突確率を予測するとともに，安全に物体を配置することが可能な場所に注目した可視化を行う．

本研究では，以下の入出力を想定する．

- 入力： 対象物体の RGBD 画像と，それを配置する領域の RGBD 画像
- 出力： 衝突が起こる確率の予測値

本論文で使用する用語を以下に定義する．

- 配置領域：ロボットが対象物体を置く場所
- 対象物体： 配置領域に置く物体
- 障害物：配置領域にすでに置かれている物体
- 衝突：危険な接触で，物体の相対速度が閾値を超えた接触を指す．ここでの物体は対象物体および障害物を表す．

本論文では評価尺度としては精度を使用する．また，物体配置を行なった際の衝突確率の予測が目的であるため，ロボットは対象物体を持った状態で配置領域の前にいることを前提とする．さらに，水の入ったコップなど，衝突に付随した別の危険性のある物体は対象とせず，配置方索も扱わない．

4.　手法

本研究では Transformer PonNet [2] を拡張する．Transformer PonNet は，画像から物体配置における衝突確率を推定するモデルである PonNet [10] に Transformer を組み込んだモデルである．本研究では，配置領域および対象物体の RGBD 画像の撮影を行い，提案手法から危険な衝突確率を推定し Attention map を生成する．生成した Attention map と平面検出法 [3] を組み合わせることで衝突確率の低い安全な領域の候補を可視化する．

4.1　Transformer PonNet の入力

ネットワークの入力を以下のように定義する．

$$\boldsymbol{x}(i) = \left\{ \boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{dst} \rangle}(i), \boldsymbol{x}_{\mathrm{depth}}^{\langle \mathrm{dst} \rangle}(i), \boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{trg} \rangle}(i), \boldsymbol{x}_{\mathrm{depth}}^{\langle \mathrm{trg} \rangle}(i) \right\} \quad (1)$$

ここに，$\boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{dst} \rangle}(i)$ および $\boldsymbol{x}_{\mathrm{depth}}^{\langle \mathrm{dst} \rangle}(i)$ は配置領域の RGB および Depth 画像，$\boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{trg} \rangle}(i)$ および $\boldsymbol{x}_{\mathrm{depth}}^{\langle \mathrm{trg} \rangle}(i)$ は対象物体の RGB および Depth 画像を表す．対象物体と配置領域の RGB および Depth 画像を正規化後，224×224 の大きさにサイズを変換した．本研究では ResNet18 の Residual unit6 の出力から特徴を抽出した．

提案ネットワークは Feature Extractor，Attention Branch，Transformer Perception Branch の 3 つからなる．Feature Extractor は ResNet18 の前半部分で構成される．それぞれに $224 \times 224 \times 3$ 次元の画像が入力され，$14 \times 14 \times 256$ 次元の特徴量マップが出力される．本手法において注意機構を用いることで衝突確率の予測に関連する視覚的特徴のうち最も顕著な部分を強調することができる．

4.2　Attention Branch

Attention Branch の入力 $\boldsymbol{f}_{\cdot} \in \mathbb{R}^{14 \times 14 \times 256}$ は ResNet18 の 6 層目からの出力である．ここに，$\cdot$ は rgb または depth を表す．Attention Branch では，まず，$\boldsymbol{f}_{\cdot}$ を ResBlock に入力し，Batch 正規化と畳み込みを行う．ここで，ResBlock の構造として文献 [12] で提案されたものを採用した．次に sigmoid 活性化関数に入力し，Attention Map $\boldsymbol{a}_{\cdot}$ を得る．Attention Branch の出力 $\boldsymbol{w}_{\cdot}$ を $\boldsymbol{w}_{\cdot} = (1 + \boldsymbol{a}_{\cdot}) \odot \boldsymbol{f}_{\cdot}$ と定義する．ここに，$\odot$ はアダマール積を表す．

4.3　安全領域の可視化

平面検出において平面と検出されたピクセルの集合を $\boldsymbol{h}$ とする．$\boldsymbol{h}$ と Attention map $\boldsymbol{a}_{\cdot}$ より，以下のように安全領域 $\boldsymbol{s}$ を出力する．

$$\boldsymbol{s} = \frac{(\boldsymbol{a}_{\mathrm{rgb}} + \boldsymbol{a}_{\mathrm{depth}})}{2} \odot \boldsymbol{h} \quad (2)$$

4.4　Transformer Perception Branch

$\boldsymbol{w}_{\cdot}$ は ResNet18 の後半部分および Global Average Pooling(GAP) 層に入力される．GAP 層の出力を $\boldsymbol{o}_{\cdot}$ とする．Transformer 層では $\boldsymbol{o}_{\cdot}$ から，Query $\boldsymbol{Q}_{\cdot}^{(i)} = \boldsymbol{W}_{Q}^{(i)} \boldsymbol{o}_{\cdot}$，Key $\boldsymbol{K}_{\cdot}^{(i)} = \boldsymbol{W}_{K}^{(i)} \boldsymbol{o}_{\cdot}$，Value $\boldsymbol{V}_{\cdot}^{(i)} = \boldsymbol{W}_{V}^{(i)} \boldsymbol{o}_{\cdot}$ を生成する．

attention $\boldsymbol{\Omega} = \left\{ \boldsymbol{w}_{\cdot}^{(1)}, \ldots, \boldsymbol{w}_{\cdot}^{(A)} \right\}$ を以下で得る．

$$\boldsymbol{w}_{\cdot}^{(i)} = \boldsymbol{V}_{\cdot}^{(i)} \mathrm{softmax}\left(\frac{\boldsymbol{Q}_{\cdot}^{(i)} \boldsymbol{K}_{\cdot}^{(i)\mathrm{T}}}{\sqrt{d_{\cdot}}} \right) \quad (3)$$

ここに，H は入力 $\boldsymbol{o}_{\cdot}$ の次元数，A はヘッド数を表し，$\sqrt{d_{\cdot}} = H/A$ である．また，attention $\boldsymbol{\Omega}$ を 2 層の全結合層に入力し，その出力を $\boldsymbol{\alpha}_{\cdot}$ とする．

Multi-Head Attention 層の出力 $\boldsymbol{m}_{\cdot}$ を以下の式で定義する．

$$\boldsymbol{m}_{\cdot} = (1 + \boldsymbol{\alpha}_{\cdot}) \odot \boldsymbol{Q}_{\cdot}^{(i)} \quad (4)$$

以上で定義した Attention Branch と Transformer Perception Branch に RGB 画像と Depth 画像をそれぞれ入力し，得られた出力をチャネル方向に結合する．その後，全結合層に入力することで，最終的な出力 $p(\hat{y})$ を得る．これより予測結果 $y^* = \underset{\hat{y}}{\mathrm{argmax}}\, p(\hat{y})$ を得る．

損失関数として，以下の $\mathcal{L}$ を用いる．

$$\mathcal{L} = \lambda_{\mathrm{rgb}}^{\langle \mathrm{att} \rangle} J_{\mathrm{rgb}}^{\langle \mathrm{att} \rangle} + \lambda_{\mathrm{depth}}^{\langle \mathrm{att} \rangle} J_{\mathrm{depth}}^{\langle \mathrm{att} \rangle} + \lambda_{\mathrm{rgb}}^{\langle \mathrm{tra} \rangle} J_{\mathrm{rgb}}^{\langle \mathrm{tra} \rangle} + \lambda_{\mathrm{depth}}^{\langle \mathrm{tra} \rangle} J_{\mathrm{depth}}^{\langle \mathrm{tra} \rangle} + \lambda_{\mathrm{total}} J_{\mathrm{total}} \quad (5)$$

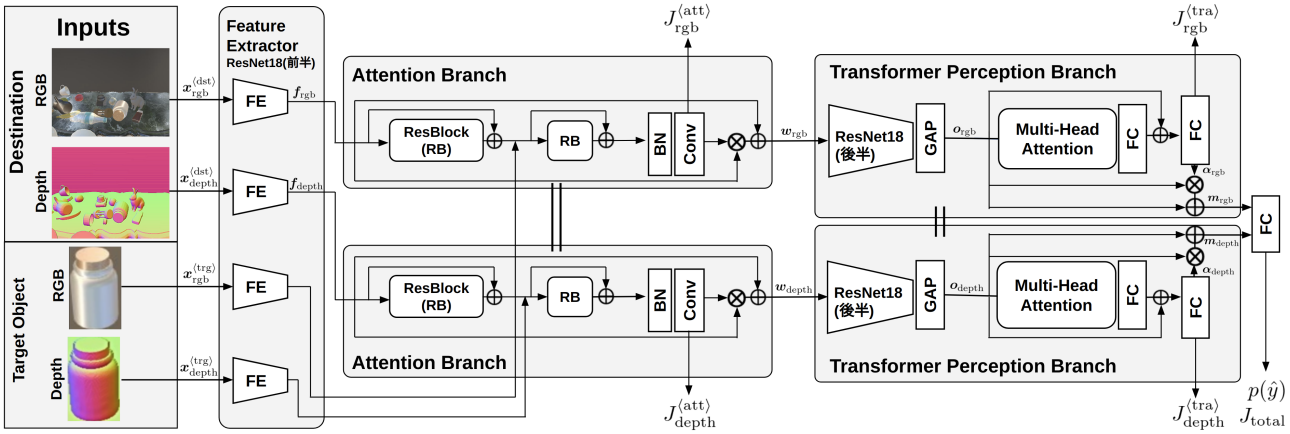

図 3 提案手法のネットワーク構造．提案手法は Feature Extractor，Attention Branch，Transformer Perception Branch からなる．ただし，二重線はパラメータ共有を示す．

表 1　ハイパーパラメータ設定

Optimizer	Adam (learning rate = 0.0003)
Backbone CNN	ResNet18
Batch size	64
Attention Branch	Input: $[14 \times 14 \times 512]$
	Att.layer: $[14 \times 14 \times 1]$
	Output: $[14 \times 14 \times 256]$
Transformer Perception Branch	Input: $[1 \times 1 \times 512]$
	FC: 512, 2
loss weights	$\lambda_{\mathrm{rgb}}^{\langle \mathrm{att} \rangle} = 1$, $\lambda_{\mathrm{depth}}^{\langle \mathrm{att} \rangle} = 1$, $\lambda_{\mathrm{rgb}}^{\langle \mathrm{tra} \rangle} = 1$, $\lambda_{\mathrm{depth}}^{\langle \mathrm{tra} \rangle} = 1$, $\lambda_{\mathrm{total}} = 1$
epochs	50

ここに，J は交差エントロピー誤差である．また，$\lambda_{\mathrm{rgb}}^{\langle \mathrm{att} \rangle}$ および $\lambda_{\mathrm{depth}}^{\langle \mathrm{att} \rangle}$ は RGB および Depth の Attention Branch からの出力の損失関数の重み，$\lambda_{\mathrm{rgb}}^{\langle \mathrm{tra} \rangle}$ および $\lambda_{\mathrm{depth}}^{\langle \mathrm{tra} \rangle}$ は RGB および Depth の Transformer Perception Branch からの出力の損失関数の重み，λ_{total} は最終出力の損失関数の重みである．

5.　実験設定

提案手法の評価のために，[2] に示すデータセットを用いる．訓練集合および検証集合をそれぞれパラメータの更新およびハイパーパラメータの選択に使用した．また，テスト集合を汎化性能の評価に使用した．

提案手法のハイパーパラメータ設定を表 1 に示す．式 (1) における $\boldsymbol{x}\,(i)$ の各要素のサイズを $\mathbb{R}^{224\times224\times3}$ とする．loss weights は損失関数の式 (5) における重みの値である．

提案手法の総パラメータ数は約 900 万である．学習は GeForce RTX 2080 Ti（メモリ 11GB） x1，64GB-RAM，Intel Corei9 9900K を搭載した計算機上で行った．学習に要した時間は約 2 時間である．また，1 サンプルあたりの推論に要した時間は 5.33ms である．最大エポック数を 50 とし，検証用データセットにおける $\mathcal{L}$ が最小となった時を学習の打ち切りとした．

6.　実験結果

6.1　定量的結果

定量的評価を表 2 に示す．ベースライン手法は平面検出 [3]，PonNet [10]，Transformer PonNet [2] である．Transformer PonNet は対象物体を指定した場所に配置する際の physical reasoning タスクにおいて良好な結果が報告されている．従って，Transformer PonNet をベースラインとした．評価尺度は精度を用いた．この理由は，データセット内の衝突と接触のデータの割合がほぼ同じであるため，精度による性能評価が合理的と考えたからである．

表 2　各データセットでの評価

	Accuracy [%]	
手法	訓練:A-Sim 訓練:A-Sim	テスト:B-Sim テスト:B-Sim
平面検出 [3]	82.5	72.30
PonNet [10]	90.94±0.22	82.29±0.68
Transformer PonNet [2] (ベースライン)	**91.26±0.21**	82.10±0.52
提案手法 (Type1)	91.16±0.77	82.82±1.52
提案手法 (Type2)	91.18±1.08	82.10±1.46
提案手法 (Type3)	91.14±0.87	**82.86±0.80**
提案手法	91.23±0.32	82.28±1.77

表 2 より，PonNet-A-Sim データセットでは，提案手法はベースライン手法と同等の精度が得られた．また，PonNet-B-Sim データセットでは，提案手法はベースライン手法に比べて精度が 0.18 ポイント向上した．一方，ベースライン手法と提案手法のパラメータ数はそれぞれおよそ 2600 万と 900 万である．ゆえに提案手法は，ベースライン手法からパラメータ数を約 3 分の 1 に削減しつつ，同等の精度またはそれを超える精度を達成した．

6.2　定性的結果

図 4 の最左列および中列に安全領域可視化の成功例を示す．上段 1 列目および上段 2 列目の目標領域画像 $\boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{dst} \rangle}\,(i)$ に対して，提案手法はそれぞれ $p(\hat{y} =$ “接触”$) = 0.728$ および $p(\hat{y} =$ “接触”$) = 0.568$ と予測した．下段 1 列目および下段 2 列目より，障害物を避けた領域および障害物の手前の領域がそれぞれ安全領域として得られた．また，図 4 の最右列に予測結果 y^* が衝突である場合の安全領域 $\boldsymbol{s}$ の可視化例を示す．上段最右列の目標領域画像 $\boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{dst} \rangle}\,(i)$ に対して，提案手法は $p(\hat{y} =$ “衝突”$) = 0.999$ でと予測した．下段最右列に示すように，提案手法は物体が密集している領域に関して注目

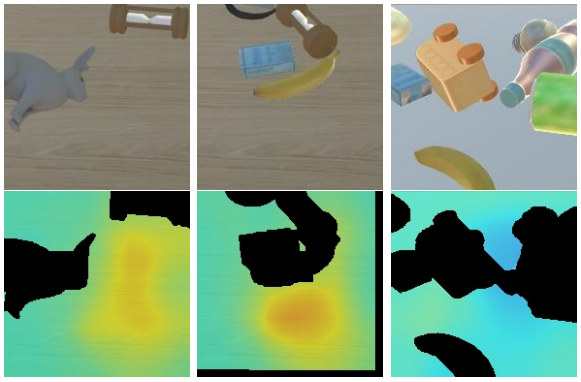

図 4 安全領域可視化の例．上段：$\boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{dst} \rangle}(i)$．下段：安全領域 $\boldsymbol{s}$

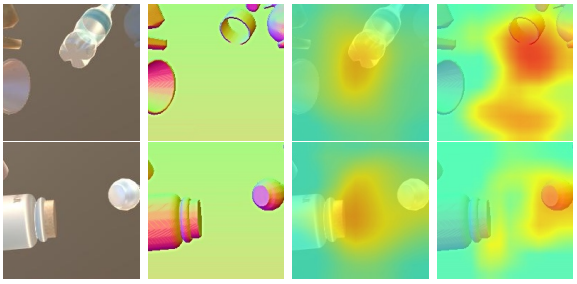

図 5 正解ラベル y が衝突に対し予測結果 y^* が接触である場合の $\boldsymbol{s}$ の可視化例．左列から順番に $\boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{dst} \rangle}(i)$，$\boldsymbol{x}_{\mathrm{depth}}^{\langle \mathrm{dst} \rangle}(i)$，$\boldsymbol{a}_{\mathrm{rgb}}$，$\boldsymbol{a}_{\mathrm{depth}}$．

をしていない．つまり，提案手法が衝突と予測する場合，顕著な安全領域は可視化されない．

6.3 Ablation Study

Ablation study として以下の 3 つの条件について検証を行った．

1. Type1：Feature Extractor のモデルを ResNet18 から ResNet34 に変更したモデル．
2. Type2：ResBlock の構造を ResNet [12] からベースラインで使用されていた改変版 ResBlock を採用したモデル．
3. Type3： Attention Branch 内で組み込んだ ResBlock 数を 2 から 3 に変更したモデル．

表 2 から，PonNet-A-Sim データセットについて，Attention Branch 内の ResBlock 数を 3 から 2 に変更することが最も性能向上に資すると考えられる．また，Feature Extractor で用いるモデルを ResNet34 よりも ResNet18 としたほうが良好な結果が得られた．一方，PonNet-B-Sim データセットに関して，PonNet-A-Sim データセットとは異なり，提案手法よりも Ablation3 のほうが精度が高いことがわかる．このことから，Attention Branch 内の ResBlock 数は 2 よりも 3 のほうが性能向上に資すると考えられる．

6.4 エラー分析

シミュレーションデータセットにおける失敗例は大きく分けて 2 種類に分類できる．1 つ目は透過する障害物を含む場合である．例えば図 5 の 1 行目のペットボトルのような透過物体の場合，$\boldsymbol{x}_{\mathrm{rgb}}^{\langle \mathrm{dst} \rangle}(i)$ ではラベル以外の透過部分が消えてしまう．そのため，ペットボトルの透過部分についても安全領域としてモデルが判断し，注目しているため，予測に失敗したと考えられる．2 つ目は衝突か接触か曖昧な位置に障害物が配置されている場合である．例えば図 5 の 2 行目で示すように，ビンが対象物体の配置領域付近に置かれている場合である．提案手法は図の中央部分を安全領域として注目し，予測結果 y^* が接触であった．しかし，実際はビンの蓋とわずかに衝突したことから正解ラベル y は衝突であり予測に失敗した．今回のデータセットには，2 つ目の失敗例のように人の目から見ても判断が難しいデータが含まれていたことにより予測に失敗したと考えられる．

7. 結論

本研究では，対象物体を指定した場所に配置する physical reasoning タスクにおいて， 衝突確率を予測するとともに，安全に物体を配置することが可能な領域の候補を可視化する手法を提案した．提案手法の貢献は以下である．

- 安全な領域を明示的に可視化できるようにした．
- 重み共有によってパラメータ数を約 3 分の 1 に削減しつつ，ベースライン手法と同等の精度が得られた．

謝辞

本研究の一部は，JSPS 科研費 20H04269，JST CREST，JST ムーンショット型研究開発事業 JPMJMS2011，NEDO の助成を受けて実施されたものである．

参 考 文 献

[1] A. Bakhtin, L. van derMaaten, J. Johnson, L. Gustafson, and R. Girshick, “Phyre: A new benchmark for physical reasoning,” NeurIPS, vol.32, pp.5082–5093, 2019.
[2] 植田有咲，M. Aly，平川翼，山下隆義他，“生活支援ロボットによる物体配置タスクにおける Transformer PonNet に基づく危険性予測および可視化，” JSAI2021，p.2J1GS8a03，2021.
[3] C. Wang and X. Guo, “Plane-based optimization of geometry and texture for rgb-d reconstruction of indoor scenes,” 2018 International Conference on 3DV, pp.533–541, 2018.
[4] G. Du, et al., “Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review,” AI Review, vol.54, no.3, pp.1677–1734, 2021.
[5] J. Bohg, A. Morales, T. Asfour, and D. Kragic, “Data-driven grasp synthesis—a survey,” T-RO, vol.30, no.2, pp.289–309, 2013.
[6] S. Levine, P. Pastor, et al., “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” IJRR, vol.37, no.4-5, pp.421–436, 2018.
[7] C. Choi, W. Schwarting, J. DelPreto, and D. Rus, “Learning object grasping for soft robot hands,” RA-L, vol.3, no.3, pp.2370–2377, 2018.
[8] J.A. Haustein, K. Hang, J. Stork, and D. Kragic, “Object placement planning and optimization for robot manipulators,” arXiv preprint arXiv:1907.02555, 2019.
[9] T. Nakamura, K. Sugiura, et al., “Learning novel objects for extended mobile manipulation,” JINT, vol.66, no.1, pp.187–204, 2012.
[10] A. Magassouba, K. Sugiura, et al., “Predicting and attending to damaging collisions for placing everyday objects in photo-realistic simulations,” AR, pp.1–13, 2021.
[11] B. Calli, et al., “Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols,” arXiv preprint arXiv:1502.03143, 2015.
[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” CVPR, pp.770–778, 2016.