第３回 とめ研究所若手研究者懸賞論文

知能移動ロボットにおける
人の反射的回避動作に基づいた
局所的経路計画手法

神戸大学大学院
海事科学研究科
Graduate School of Maritime Sciences,
Kobe University

小林 聖人
Masato KOBAYASHI

目 次

第1章　序論

近年、少子高齢社会の影響によりサービスロボットによる労働力創出が希求されている。これらサービスロボットには移動、物体操作等の知的動作が必要となる [1–4]。そこで、本論文では「移動」に焦点を当てる。サービスロボットが人と共存する空間で移動するためには、時々刻々と変化する動的な環境下に対応する必要がある [5–7]。一般的にロボットの移動方式は大別して遠隔移動と自律移動に分けられるが、本論文では労働力創出の観点からロボット操作者が不要である「自律移動」に着目して論じる。

自律移動は自己位置推定 [8]、地図生成 [9]、認識 [10]、経路計画 [11] 等の人工知能・知能情報処理技術に基づいた技術により構成されている。自己位置推定では、地図上でのロボットの位置を推定する。認識では、距離センサ等を用いてロボット周囲の環境情報の計測する。経路計画では、認識で取得した環境情報とロボットの位置等の情報をもとに、障害物回避経路の生成する。これら処理を実時間で行うことで自律移動が実現される。また、経路計画は大別して２つの手法により構成されており、大域的経路計画と局所的経路計画である。大域的経路計画では、事前に作成した地図の情報を利用して目的地まで障害物と衝突しない経路を生成する [12–14]。しかしながら大域的経路計画では、事前地図の情報のみを利用しているために、事前地図にない障害物には対応できない問題がある。そこで局所的経路計画では、リアルタイムでセンサから取得した情報を利用して経路計画を行う。つまり局所的経路計画では、事前地図上にない障害物を考慮した経路計画を行うことが可能である。

本論文では、移動経路を考える知能に該当する「局所的経路計画手法」に着目する。静的・動的障害物を考慮した多くの局所的経路計画手法が報告されている [17] [18]。Fiorini 等は、マルチロボットにおける局所的経路計画手法である Velocity Obstacle (VO) を提案した [19]。VO は速度空間で動的障害物を考慮している。Berg 等は VO を拡張し振動のない滑らかな経路探索を実現した [20]。この他にもマルチロボットのための VO に基づく有用な局所経路計画法が報告されている [19–22]。VO では障害物回避可能な経路候補を生成する場合、速度空間内で障害物と衝突しない速度を探索する。更に VO では、ロボットの速度は一定と仮定しているために、ロボットの経路候補は直線や円弧経路となる。そのため狭路環境や動的な環

境では、障害物と衝突しないロボットの経路候補が減少する。また VO ではロボットのダイナミクスに基づく速度制約を考慮していないため、生成された速度をロボットが実現できない場合も存在する。

ロボットのダイナミクスに基づく速度制約を考慮した局所的経路計画も報告されている。Fox 等は Dynamic Window Approach (DWA) を報告した。DWA はダイナミクス制約を考慮した速度空間 (VSD) を用いて経路候補を生成する。VSD はロボットが現在の速度から生成できる速度範囲を意味する。DWA は VSD 内から生成された経路候補の中から最適な経路を選択することで障害物回避を行いながらゴールへ到達可能である。Dobrevski 等は、DWA と深層強化学習に基づく局所経路計画により、経路最適化を改善することを報告した [24]。Liu 等は、jump-A*アルゴリズムと DWA を組み合わせたグローバルな動的経路計画の融合アルゴリズムを開発した [25]。この他にも DWA に基づく有用な局所的経路計画法が報告されている [23–25]。しかし、DWA は一定時間の間、一定の速度を仮定して経路候補を生成するため経路候補は直線や円弧の経路となる。つまり、経路候補生成時には障害物を考慮していないため、動的障害物等と衝突する経路が多く生成されることとなり、動的な環境や狭路環境下での移動には適さない。

非直線経路や非円弧経路を含む経路候補を生成する局所経路計画手法が存在する [26–28]。Howard 等は State Lattice Planner (SLP) [26] [27] を報告した。SLP は、ロボットの状態とロボットの制約条件のデータセットを用いて、経路候補を生成する。しかし、ロボットの正面付近に障害物がある場合、SLP は経路候補を生成できないことがある。

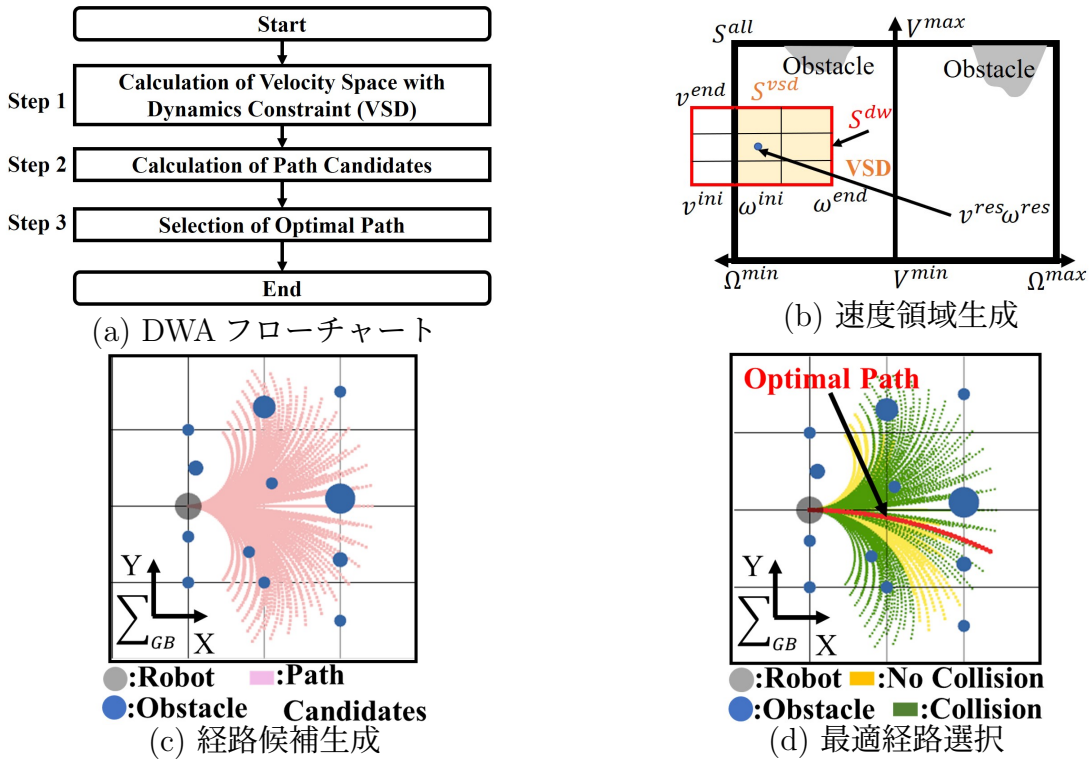
そこで本論文では、非直線・非円弧経路を含む障害物回避可能な経路生成を、人の反射的回避動作を知能情報処理技術基づき考案した、知能移動ロボットの局所的経路計画手法である Dynamic Window Approach with Virtual Manipulators (DWV) について述べる [29]。DWV は Virtual Manipulator (VM) と DWA に基づいて構成される [30]。ここで人の反射的動作を模倣するにあたり、光等が無い暗闇の中で歩いて移動する場面の想像をお願いしたい。光等が無い暗闇の中での移動する際、人は恐る恐る手や腕を用いながら手探りで移動する。暗闇の中、手探りで何か障害物を認知した場合には、障害物を手で押して避けるように移動することで、障害物回避が可能となる。本論文では、この様な反射的回避動作を生成可能な VM と、静的・動的障害物の予測位置を用いることで、ロボットの算出可能な速度領域から、非直線・非円弧経路を含む障害物回避可能な経路を生成可能とし、動的環境下での自律移動を DWV によって実現する。DWV の検証を行うためにシミュレーション及び実験により検証を行う。

本論文は第 7 章により構成されており、第 1 章では、人等が混在する自律移動に関する研究

背景と研究目的について述べる。第 2 章では、局所的経路計画手法である Dynamic Window Approach(DWA) について述べる。第 3 章では、反射的動作を生成する Virtual Manipulator(VM) について述べる。第 4 章では、動的障害物を考慮した局所的経路計画手法である Dynamic Window Approach with Virtual Manipulator(DWV) について述べる。第 5 章では、従来手法と提案手法を比較したシミュレーションについて述べる。第 6 章では、提案手法の実現性を確認するために、実機実験について述べる。第 7 章にて、本論文の結びとする。

第2章　Dynamic Window Approach

2.1　DWA 概要



(a) DWA フローチャート　(b) 速度領域生成

(c) 経路候補生成　(d) 最適経路選択

図 2.1 DWA 概要

DWA は局所的経路計画の 1 つである。図 2.1(a) に DWA のフローチャートを示す。DWA は次の 3 つステップにより実行する。

- ダイナミクスを考慮した速度領域の生成 (Step 1)
  現在の速度やロボットの加速度から算出される速度領域である (図 2.1(b))。
- 経路候補生成 (Step 2)
  Step 1 で算出した速度領域から経路候補を生成する (図 2.1(c))。
- 最適経路選択 (Step 3)
  Step 2 で算出した経路候補から最適な経路を選択する (図 2.1(d))。

2.2 ダイナミクスを考慮した速度領域

図 2.1(b) は Velocity Space with Dynamics Constraint (VSD) のイメージ図である。VSD は３つのステップにより生成される。

2.2.1 Step 1-1: 全速度領域 S^{all}

全速度領域 S^{all} は以下のように算出される。

$$S^{all} = \{(v, \omega) \quad | \quad v \in [V^{min}, V^{max}] \wedge \omega \in [\Omega^{min}, \Omega^{max}]\} \tag{2.1}$$

ここで、V^{min}、V^{max}、Ω^{min}、Ω^{max} はロボットが算出可能な最小・最大並進・旋回速度である。

2.2.2 Step 1-2: ダイナミックウィンドウ速度領域 S^{dw}

ダイナミックウィンドウ速度領域 S^{dw} は、ロボットが次のステップで算出可能な速度領域である。

$$\begin{aligned} S^{dw} &= \{(v, \omega) | v \in [v^{res} - A^{max}\Delta T, v^{res} + A^{max}\Delta T] \\ &\wedge \quad \omega \in [\omega^{res} - \Pi^{max}\Delta T, \omega^{res} + \Pi^{max}\Delta T]\} \end{aligned} \tag{2.2}$$

ここで v^{res} と ω^{res} は並進・旋回速度応答値であり、現在の速度とみなす。A^{max} と Π^{max} は最大並進・旋回加速度であり、ΔT はタイムステップである。

2.2.3 Step 1-3: ダイナミクスを考慮した速度領域 S^{vsd}

ダイナミクスを考慮した速度領域 S^{vsd} は以下のように算出される。

$$S^{vsd} = S^{all} \cap S^{dw} \tag{2.3}$$

$$= \{(v, \omega) \mid v \in [v^{ini}, v^{end}] \wedge \ \omega \in [\omega^{ini}, \omega^{end}]\} \tag{2.4}$$

ここで v^{ini} と ω^{ini} は S^{vsd} でのダイナミクスを考慮した最小並進・旋回速度である。v^{end} と ω^{end} は S^{vsd} でのダイナミクスを考慮した最大並進・旋回速度である。

2.3 経路候補生成

Algorithm 1 Path Candidates by DWA

$g \leftarrow 1, i \leftarrow 1$
// ▼*Step 2-1a*
while $N^{tlv} > g$ **do**
  $h \leftarrow 1$
  $v^{dwa} \leftarrow v^{ini} + \Delta v(g-1)$
  // ▼*Step 2-2a*
  while $N^{agv} > h$ **do**
    $\omega^{dwa} \leftarrow \omega^{ini} + \Delta\omega(h-1)$
    // ▼*Step 2-3a*
    while $f^{max} > f$ **do**
      $< x_f^{rob}, y_f^{rob}, \theta_f^{rob} > \leftarrow onePath(v^{dwa}, \omega^{dwa})$
      $< \boldsymbol{X}_i^{rob}, \boldsymbol{Y}_i^{rob}, \boldsymbol{\Theta}_i^{rob} > \leftarrow < x_f^{rob}, y_f^{rob}, \theta_f^{rob} >$
      $f \leftarrow f + 1$
    end while
    $\boldsymbol{P_i^{rob}} \leftarrow < \boldsymbol{X}_i^{rob}, \boldsymbol{Y}_i^{rob}, \boldsymbol{\Theta}_i^{rob} >$
    $\boldsymbol{O}^{rob} \leftarrow \boldsymbol{P_i^{rob}}$
    $i \leftarrow i + 1$
    $h \leftarrow h + 1$
  end while
  $g \leftarrow g + 1$
end while
return $\boldsymbol{O}^{rob}$

図 2.1(c) に DWA 経路候補のイメージ図を示す。Algorithm 1 は、DWA による経路候補の擬似コードである。N^{tlv}, N^{agv}, N^{all} は、並進速度、旋回速度、パス候補の最大数である。DWA は速度空間 S^{vsd} を並進速度軸 N^{tlv} と旋回速度軸 N^{agv} に分割し、合計で $N^{all}(= N^{tlv} \cdot N^{agv})$ の並進速度と旋回速度のペアが生成される。したがって、N^{all} 個の速度ペアは、N^{all} 個の経

路候補 $\boldsymbol{O}^{rob}$ を生成する。ロボットの経路候補 $\boldsymbol{O}^{rob}$ は以下のように表される。

$$\boldsymbol{O}^{rob} = [\boldsymbol{P}_1^{rob} \cdots \boldsymbol{P}_i^{rob} \cdots \boldsymbol{P}_{N^{all}}^{rob}]^T \tag{2.5}$$

$$\boldsymbol{P}_i^{rob} = [\boldsymbol{X}_i^{rob}\ \boldsymbol{Y}_i^{rob}\ \boldsymbol{\Theta}_i^{rob}]^T \tag{2.6}$$

ここで、$\boldsymbol{P_i^{rob}}$ は f^{max} 行 3 列の行列であり、i 番目のロボットの経路候補を示す。$f_{max}(=\frac{T^{max}}{\Delta T})$ はタイムステップの最大カウント数であり、最大予測時間 T^{max} と時間刻み ΔT から算出される。$\boldsymbol{X}_i^{rob}$, $\boldsymbol{Y}_i^{rob}$, $\boldsymbol{\Theta}_i^{rob}$ はロボットの位置と姿勢を示す。つまり、$\boldsymbol{P_i^{rob}}$ には 1 本の経路候補の位置、姿勢 $\boldsymbol{X}_i^{rob}$, $\boldsymbol{Y}_i^{rob}$, $\boldsymbol{\Theta}_i^{rob}$ が格納されている。Algorithm 1 の詳細について説明する。

2.3.1 Step 2-1a 並進速度選択 (Algorithm 1 lines 2-5)

DWA は速度領域 S^{vsd} を並進速度方向に N^{tlv} 個分割する。DWA では並進速度 v^{dwa} を次のように選択する。

$$v^{dwa} = v^{ini} + \Delta v(g-1) \tag{2.7}$$

ここで、Δv は並進速度の変化量、g $(1 \leq g \leq N^{tlv})$ は並進速度のカウント数である。

2.3.2 Step 2-2a 旋回速度選択 (Algorithm 1 lines 6-8)

DWA は速度領域 S^{vsd} を旋回速度方向に N^{agv} 個分割する。DWA では旋回速度 ω^{dwa} を次のように選択する。

$$\omega^{dwa} = \omega^{ini} + \Delta\omega(h-1) \tag{2.8}$$

ここで、$\Delta\omega$ は旋回速度の変化量、h $(1 \leq h \leq N^{agv})$ は旋回速度のカウント数である。

2.3.3 Step 2-3a 経路生成 (Algorithm 1 lines 9-13)

i 番目の経路候補 $\boldsymbol{P_i^{rob}}$ は速度ペア $[v^{dwa}, \omega^{dwa}]$ から算出される。Algorithm 1(line 11) の $onePath(v^{dwa}, \omega^{dwa})$ で次のように算出される。

$$\boldsymbol{X}_i^{rob} = [x_1^{rob} \cdots x_f^{rob} \cdots x_{f^{max}}^{rob}]^T \tag{2.9}$$

$$\boldsymbol{Y}_i^{rob} = [y_1^{rob} \cdots y_f^{rob} \cdots y_{f^{max}}^{rob}]^T \tag{2.10}$$

$$\boldsymbol{\Theta}_i^{rob} = [\theta_1^{rob} \cdots \theta_f^{rob} \cdots \theta_{f^{max}}^{rob}]^T \tag{2.11}$$

$$\theta_f = \sum_{k=1}^{f} \int_{t_{k-1}}^{t_k} \omega^{dwa}\, dt \tag{2.12}$$

$$x_f = \sum_{k=1}^{f} \int_{t_{k-1}}^{t_k} v^{dwa} \cdot \cos\theta_k\, dt \tag{2.13}$$

$$y_f = \sum_{k=1}^{f} \int_{t_{k-1}}^{t_k} v^{dwa} \cdot \sin\theta_k\, dt \tag{2.14}$$

ここで、f $(1 \leq f \leq f^{max})$ はタイムステップのカウント数である。

Step 2-1a から Step 2-3a を N^{all} 回行うことで、ロボットの経路候補 $\boldsymbol{O}^{rob}$ が算出される。

2.4 最適経路選択

図 2.1(d) に DWA の最適経路選択の様子を示す。ロボットの経路候補 $\boldsymbol{O}^{rob}$ はコスト関数によって評価される。コスト関数は次の様に導出される。

$$c^{dwa} = W^{ang} \cdot c^{ang} + W^{vel} \cdot c^{vel} + W^{obs} \cdot c^{obs} \tag{2.15}$$

ここで W^{ang}, W^{vel}, W^{obs} は重み係数である。c^{ang} はゴールへの方向を考慮したコスト、c^{vel} は選択した速度、c^{obs} は障害物とロボットとの距離である。DWA ではコスト関数 (2.15) を最大化する経路を最適な経路として選択する。

最後に最適な経路の並進速度 v^{opt}、旋回速度 ω^{opt} とし、ロボットの速度指令値を算出する。

$$v^{cmd} = v^{opt} \tag{2.16}$$

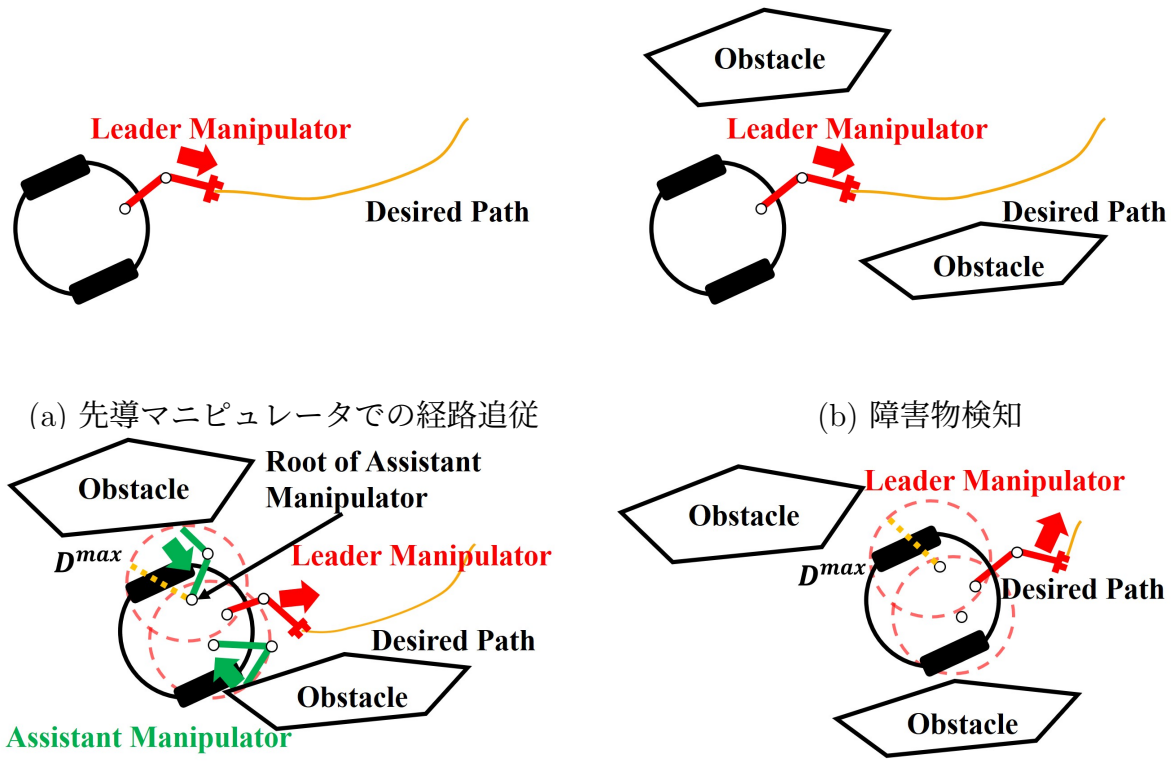
$$\omega^{cmd} = \omega^{opt} \tag{2.17}$$

ここで v^{cmd}、ω^{cmd} はロボットの並進・旋回速度指令値である。算出された速度指令値を用いることでロボットは障害物を回避しながらゴールへ到達することが可能となる。

第3章　Virtual Manipulator

本章では障害物回避手法の一つとして、山崎らが提案した仮想マニピュレータ (VM) を用いた反射的動作生成 [30] [31] について説明する。

3.1　VMの動作生成例



(a) 先導マニピュレータでの経路追従

(b) 障害物検知

(c) 補助マニピュレータによる障害物回避

(d) 補助マニピュレータ削除

図 3.1 反射的動作生成の概念図

図 3.1 は VM を用いた反射的動作生成による障害物回避の概念図である。移動ロボットには任意の数の VM を搭載可能である。VM は経路追従の役割を担う「先導マニピュレータ」と障害物回避の役割を担う「補助マニピュレータ」により構成される。図 3.1 を用いて VM について簡易に説明する。図 3.1(a) では、移動ロボットが先導マニピュレータによって所望の経路を追従することを示している。図 3.1(b) では、移動ロボットはセンサーを用いて障害物を検出する。図 3.1(c) では、障害物と補助マニピュレータの付け根までの距離が D^{max} よりも小さい障害物が存在する場合に、補助マニピュレータを生成する。D^{max} は、障害物と移動ロボットの間の距離閾値である。 移動ロボットは、補助マニピュレータにより障害物を回避する。図 3.1(d) に示すように、障害物と補助マニピュレータの根元間の距離は D^{max} より大きくなると、補助マニピュレータを削除する。 従って、ロボットは所望の経路を追従しながら、障害物を回避することが可能である。

3.2 VM を用いた反射的動作生成

自由度数の合計が m 個とした VM を 1 本搭載したロボットの状態ベクトルを $\boldsymbol{q} = [q_1, q_2, \cdots, q_m]$ とする。VM の手先状態を $\boldsymbol{p}_e = [x_e, y_e, \theta_e]$ とし、$\dot{\boldsymbol{p}}_e$ と $\dot{\boldsymbol{q}}$ の関係は式 (3.1) で表すことができる。

$$\dot{\boldsymbol{p}}_e = J(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.1}$$

ここで、$\boldsymbol{J}(\boldsymbol{q})$ はヤコビ行列である。ヤコビ行列は次のように算出される。

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} \frac{\partial x_e}{\partial q_1} & \frac{\partial x_e}{\partial q_2} & \cdots & \frac{\partial x_e}{\partial q_m} \\ \frac{\partial y_e}{\partial q_1} & \frac{\partial y_e}{\partial q_2} & \cdots & \frac{\partial y_e}{\partial q_m} \\ \frac{\partial \theta_e}{\partial q_1} & \frac{\partial \theta_e}{\partial q_2} & \cdots & \frac{\partial \theta_e}{\partial q_m} \end{bmatrix} \tag{3.2}$$

3.3 二輪駆動ロボット:移動マニピュレータのヤコビ行列

手先姿勢 $\dot{\boldsymbol{p}}_e = [\dot{x}_e, \dot{y}_e, \dot{\theta}_e]$ への速度入力によるロボット動作は次のように算出される。

$$\begin{bmatrix} v^{vm} \\ \omega^{vm} \\ \dot{\theta_{01}} \\ \dot{\theta_{02}} \end{bmatrix} = \boldsymbol{J}^{+}_{pws}\dot{\boldsymbol{p}}_{\boldsymbol{e}} \tag{3.3}$$

$\boldsymbol{J}_{pws}^{+}$ は $\boldsymbol{J}_{pws}$ の疑似逆行列であり、ロボットのノンホロミックの拘束を考慮した並進速度・旋回速度とマニピュレータの関節角速度を計算することが可能となる。

VM を用いたロボットの反射的動作生成を式 (3.4) のように構成する。

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_{w}^{+}\dot{\boldsymbol{x}} + \Lambda(\boldsymbol{I} - \boldsymbol{J}_{w}^{+}\boldsymbol{J})(\boldsymbol{\Theta}_{\boldsymbol{nj}}^{\boldsymbol{ref}} - \boldsymbol{\theta}_{\boldsymbol{nj}}^{\boldsymbol{res}}) \tag{3.4}$$

ここで、$\dot{\boldsymbol{q}} = \left[v^{vm}, \omega^{vm}, \dot{\theta}_{01}, \dot{\theta}_{02}, \dot{\theta}_{11}, \dot{\theta}_{12}, \cdots, \dot{\theta}_{n1}, \dot{\theta}_{n2}\right]^{T}$ はロボットとマニピュレータの各関節角の速度・角速度であり、$\dot{\boldsymbol{x}}$ は手先目標速度である。v^{vm} と ω^{vm} は VM により生成されたロボットの並進速度と旋回速度である。$\dot{\theta}_{n1}$ と $\dot{\theta}_{n2}$ は n 番目の VMの角速度である。$\boldsymbol{J}_{w}^{+}$ は重み行列 $\boldsymbol{W}$ を掛けたものである。Λ は零空間の重み係数である。$\boldsymbol{\Theta}_{\boldsymbol{nj}}^{\boldsymbol{ref}}$ と $\boldsymbol{\theta}_{\boldsymbol{nj}}^{\boldsymbol{res}}$ は関節角の参照値と応答値を示す。$j(=1,2)$ は第一関節と第二関節を示す。

3.4　実際の動作例

図 3.2 に VM の動作例を示す。図 3.2 の左側には物理シミュレーター内のロボットと障害物の様子を示している。また青色の線は LRF の環境取得範囲を示している。右側は可視化ツール Rviz でロボットと障害物を可視化している。また VM は左右 1 本ずつ搭載されており、左の VM を黒色、右の VM を緑色とした。ロボットから出ている赤色の線は VM により移動する経路を示している。図 3.2 が示すように、LRF 等のセンサから取得した情報を用いて、ロボット近傍に障害物がある場合には VM により反射的な動作生成を行う。

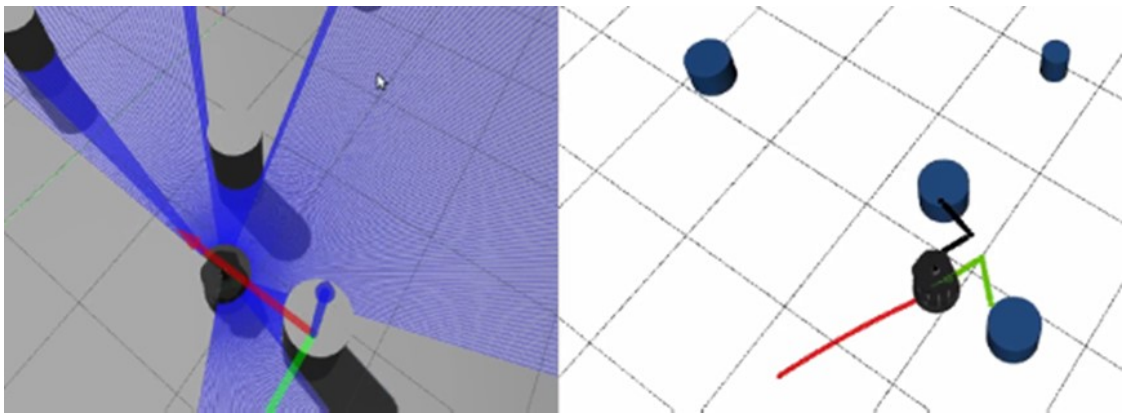
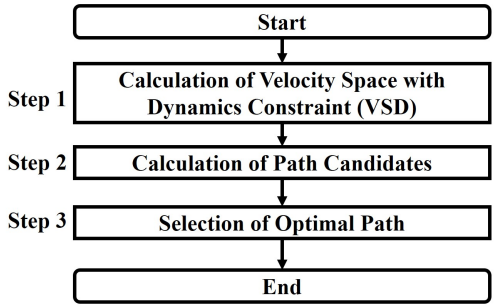


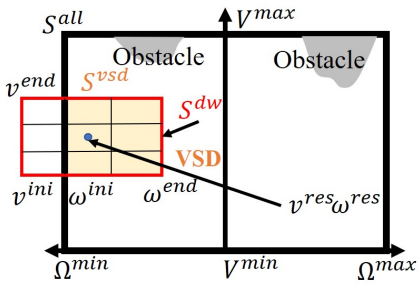
図 3.2 仮想補助マニピュレータ生成

第4章　Dynamic Window Approach with Virtual Manipulators

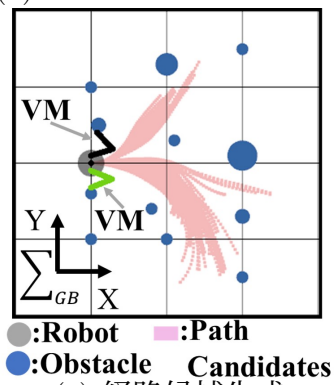
4.1　DWV 概要



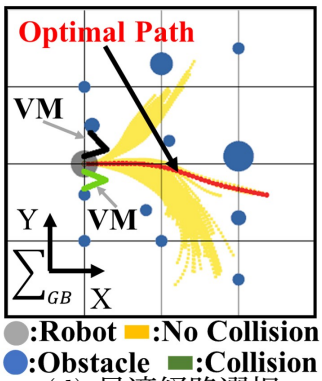
(a) DWV フローチャート



(b) 速度領域生成



(c) 経路候補生成



(d) 最適経路選択

図 4.1 DWV 概要

図 2.1(a)、4.1(a) が示すように、DWV のフローチャートは DWA と同じである。DWV は DWA から経路候補生成 (Step 2) と最適経路算出 (Step 3) を変更している。

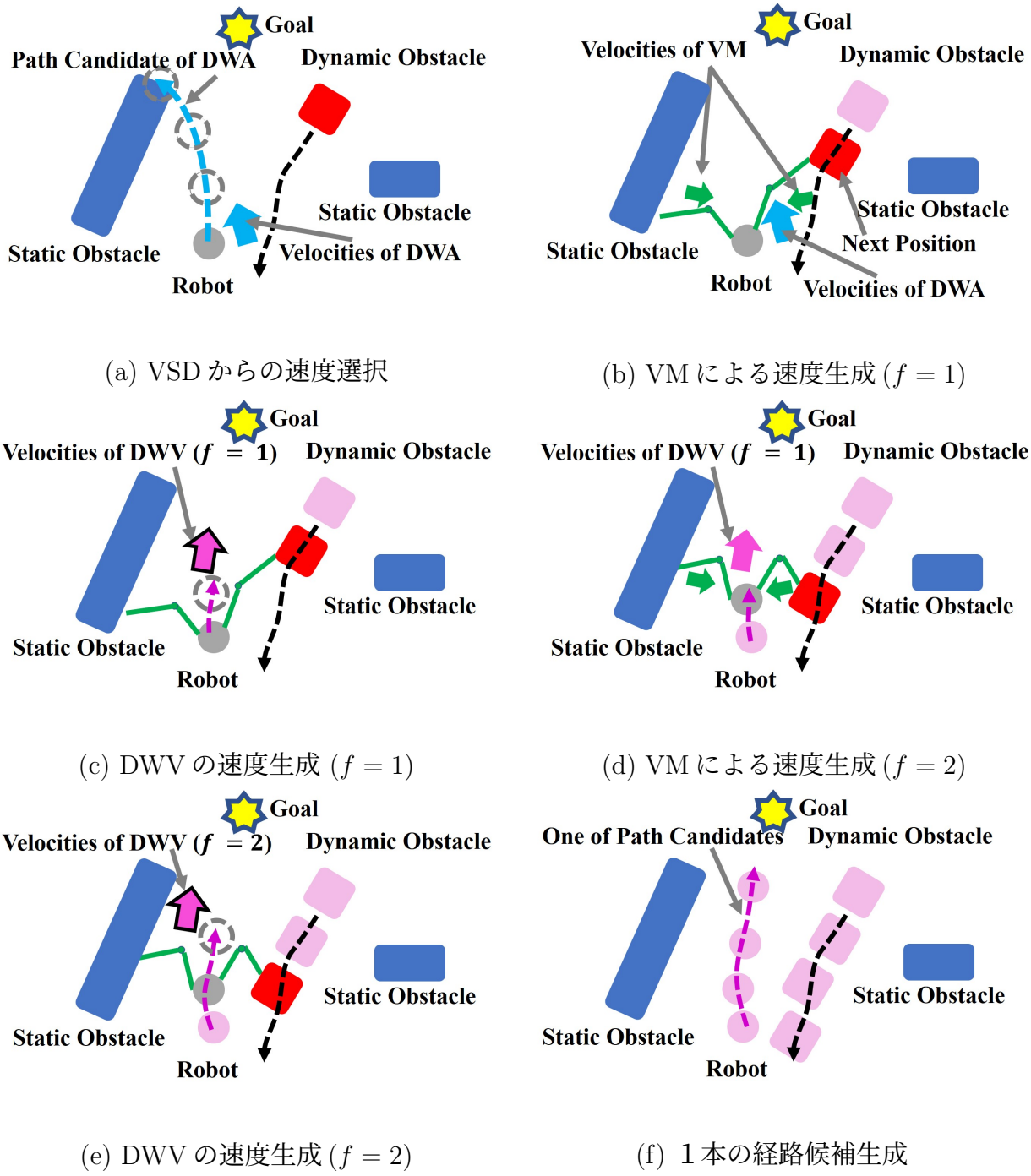
- ダイナミクスを考慮した速度領域 (Step 1)
  現在の速度やロボットの加速度から算出される速度領域である。

- 経路候補生成 (Step 2)
  Step 1 で算出した速度領域から経路候補を生成する。DWV では VM と障害物の予測経路を用いて経路候補を算出する。

- 最適経路選択 (Step 3)
  Step 2 で算出した経路候補から最適な経路を選択する。

4.2 経路候補生成

図 4.2 は DWV の経路候補生成の概念図を示している。



(a) VSD からの速度選択

(b) VM による速度生成 ($f = 1$)

(c) DWV の速度生成 ($f = 1$)

(d) VM による速度生成 ($f = 2$)

(e) DWV の速度生成 ($f = 2$)

(f) 1 本の経路候補生成

図 4.2 DWV 経路候補生成の概念図

1. 初めに DWV では DWA と同様に VSD から速度を選択する。DWA では VSD から選択した一定の速度を使用して経路候補を算出する。したがって DWA の経路候補は円弧の経路となる。障害物を考慮した経路候補を生成するために次ステップでは VM を用いた動作生成を行う (図 4.2(a))。

2. DWV はステップ f 毎に VM を用いて障害物回避可能な経路を生成する。VM は障害物の次ステップの予測位置から障害物回避を考慮した速度を生成する (図 4.2(b))。

3. $(f = 1)$ の時の DWV の速度は DWA と VM から生成する (図 4.2(c))。

4. VM は障害物の次ステップの予測位置から障害物回避を考慮した速度を生成する (図 4.2(d))。

5. $(f = 2)$ の時の DWV の速度は 1 ステップ前 $(f = 1)$ の DWV の速度と VM の速度により生成される (図 4.2(e))。

6. この処理を f^{max} 回繰り返しすことで、DWV の経路候補の一つ $\boldsymbol{P_i^{rob}}$ が生成される (図 4.2(f))。

これら操作を N^{all} の速度ペアで行うことで図 4.1(c) のように経路候補が生成される。

Algorithm 2 は DWV の経路候補算出についての疑似コードを示す (Step 2)。DWV の経路候補生成方法について Algorithm 2 を用いて説明する。

Algorithm 2 Path Candidates by DWV

$g \leftarrow 1, i \leftarrow 1$
// ▼*Step 2-1b*
while $N^{tlv} > g$ **do**
  $v^{dwa} \leftarrow v^{ini} + \Delta v(g-1)$
  $h \leftarrow 1$
  // ▼*Step 2-2b*
  while $N^{agv} > h$ **do**
    $\omega^{dwa} \leftarrow \omega^{ini} + \Delta\omega(h-1)$
    $f \leftarrow 1$
    while $f^{max} > f$ **do**
      // ▼*Step 2-3b*
      $\omega_f^{vm} \leftarrow calcVM(\boldsymbol{O}^{obs})$
      // ▼*Step 2-4b*
      $v^{dwv} \leftarrow v^{dwa}$
      if $f = 1$ **then**
        $\omega_f^{dwv} \leftarrow \omega^{dwa} + \omega_f^{vm}$
      else
        $\omega_f^{dwv} \leftarrow \omega_{f-1}^{dwv} + \omega_f^{vm}$
      end if
      // ▼*Step 2-5b*
      $< v^{dwv}, \omega_f^{dwv} > \leftarrow checkVelocity(v^{dwv}, \omega_f^{dwv})$
      // ▼*Step 2-6b*
      $< x_f^{rob}, y_f^{rob}, \theta_f^{rob} > \leftarrow onePath(v^{dwv}, \omega_f^{dwv})$
      $< \boldsymbol{X}_i^{rob}, \boldsymbol{Y}_i^{rob}, \boldsymbol{\Theta}_i^{rob} > \leftarrow < x_f^{rob}, y_f^{rob}, \theta_f^{rob} >$
      $f \leftarrow f + 1$
    end while
    $\boldsymbol{P_i^{rob}} \leftarrow < \boldsymbol{X}_i^{rob}, \boldsymbol{Y}_i^{rob}, \boldsymbol{\Theta}_i^{rob} >$
    // ▼*Step 2-7b*
    if $collision(\boldsymbol{O^{obs}}, \boldsymbol{P_i^{rob}})$ **then**
      $delete(\boldsymbol{P}_i^{rob})$
    else
      $\boldsymbol{O}^{rob} \leftarrow \boldsymbol{P_i^{rob}}$
      $i \leftarrow i + 1$
    end if
    $h \leftarrow h + 1$
  end while
  $g \leftarrow g + 1$
end while
return $\boldsymbol{O}^{rob}$

4.2.1 Step 2-1b 並進速度選択 (Algorithm 2 lines 2-4)

DWV は速度領域 S^{vsd} を並進速度方向に N^{tlv} 個分割する。DWV では並進速度 v^{dwa} を次のように選択する。

$$v^{dwa} = v^{ini} + \Delta v(g-1) \tag{4.1}$$

4.2.2 Step 2-2b 旋回速度選択 (Algorithm 2 lines 6-8)

DWV は速度領域 S^{vsd} を旋回速度方向に N^{agv} 個分割する。DWV では旋回速度 ω^{dwa} を次のように選択する。

$$\omega^{dwa} = \omega^{ini} + \Delta\omega(h-1) \tag{4.2}$$

4.2.3 Step 2-3b 仮想マニピュレータによる速度生成 (Algorithm 2 lines 11-12)

DWV では、静的・動的障害物を考慮した速度生成を障害物の予測位置 $\boldsymbol{O}^{obs}$ と VM により生成する。障害物の予測位置 $\boldsymbol{O}^{obs}$ は次のように示す。

$$\boldsymbol{O}^{obs} = [\boldsymbol{P}_1^{obs} \cdots \boldsymbol{P}_s^{obs} \cdots \boldsymbol{P}_{s^{max}}^{obs}]^T \tag{4.3}$$

$$\boldsymbol{P}_s^{obs} = [\boldsymbol{X}_s^{obs}\ \boldsymbol{Y}_s^{obs}]^T \tag{4.4}$$

$$\boldsymbol{X}_s^{obs} = [x_{s,1}^{obs} \cdots x_{s,f}^{obs} \cdots x_{s,f^{max}}^{obs}]^T \tag{4.5}$$

$$\boldsymbol{Y}_s^{obs} = [y_{s,1}^{obs} \cdots y_{s,f}^{obs} \cdots y_{s,f^{max}}^{obs}]^T \tag{4.6}$$

ここで、s $(1 \leq s \leq s^{max})$ は障害物の数を表す。DWV では VM は補助マニピュレータのみで構成される。f 番目の障害物の位置と VM の根本位置の距離 d_f^{rob} から、(3.4) を用いて旋回速度 ω^{vm} を生成する。f 番目の旋回速度 ω_f^{vm} は次のように生成される。

$$\omega_f^{vm} = \begin{cases} \omega^{vm},\ if\ d_f^{rob} \leq D^{max} \\ 0,\ otherwise \end{cases} \tag{4.7}$$

4.2.4 Step 2-4b 速度合成 (Algorithm 2 lines 13-19)

f 番目の DWV の速度は次のように生成される。

$$v^{dwv} = v^{dwa} \tag{4.8}$$

$$\omega_f^{dwv} = \begin{cases} \omega^{dwa} + \omega_f^{vm},\ if\ f = 1 \\ \omega_{f-1}^{dwv} + \omega_f^{vm},\ otherwise \end{cases} \tag{4.9}$$

ここで、v^{dwv} と ω_f^{dwv} は DWV の並進速度と旋回速度を示す。

4.2.5 Step 2-5b 速度制限 (Algorithm 2 lines 20-21)

DWV の f 番目の速度である v^{dwv} と ω_f^{dwv} を f 番目速度領域 S_f^{dwv} 内に存在するか確認する。f 番目の速度領域 S_f^{dwv} は f 番目の S_f^{pdw} と S^{all} から構成される。S_f^{dwv} は以下のように算出される。

$$\begin{aligned} S_f^{pdw} &= \{(v,\omega) | v \in [v^{dwv} - A^{max}\Delta T, v^{dwv} + A^{max}\Delta T] \\ &\wedge\ \omega \in [\omega_{f-1}^{dwv} - \Pi^{max}\Delta T, \omega_{f-1}^{dwv} + \Pi^{max}\Delta T]\} \end{aligned} \tag{4.10}$$

$$S_f^{dwv} = \begin{cases} S^{all} \cap S^{dw},\ if\ (f = 1) \\ S^{all} \cap S_f^{pdw},\ otherwise \end{cases} \tag{4.11}$$

$$= \{(v,\omega) \mid v \in [v_f^{ini}, v_f^{end}] \wedge\ \omega \in [\omega_f^{ini}, \omega_f^{end}]\} \tag{4.12}$$

ここで v_f^{ini}、ω_f^{ini} は速度領域 S_f^{dwv} 内の最小並進・旋回速度である。v_f^{end}、ω_f^{end} は速度領域 S_f^{dwv} 内の最大並進・旋回速度である。v^{dwv} か ω_f^{dwv} が速度領域 S_f^{dwv} 内に含まれない場合には、v^{dwv} と ω_f^{dwv} は次の様に算出される。

$$v^{dwv} = \begin{cases} v_f^{ini},\ if\ v^{dwv} < v_f^{ini} \\ v_f^{end},\ if\ v^{dwv} > v_f^{end} \\ v^{dwv},\ otherwise \end{cases} \tag{4.13}$$

$$\omega_f^{dwv} = \begin{cases} \omega_f^{ini},\ if\ \omega_f^{dwv} < \omega_f^{ini} \\ \omega_f^{end},\ if\ \omega_f^{dwv} > \omega_f^{end} \\ \omega_f^{dwv},\ otherwise \end{cases} \tag{4.14}$$

4.2.6 Step 2-6b 経路生成 (Algorithm 2 lines 22-24)

ロボットの位置は v^{dwv} と ω_f^{dwv} から算出し、i 番目の経路候補は次のように算出する。

$$\boldsymbol{X}_i^{rob} = [x_1^{rob} \cdots x_f^{rob} \cdots x_{f^{max}}^{rob}]^T \tag{4.15}$$

$$\boldsymbol{Y}_i^{rob} = [y_1^{rob} \cdots y_f^{rob} \cdots y_{f^{max}}^{rob}]^T \tag{4.16}$$

$$\boldsymbol{\Theta}_i^{rob} = [\theta_1^{rob} \cdots \theta_f^{rob} \cdots \theta_{f^{max}}^{rob}]^T \tag{4.17}$$

$$\theta_f = \sum_{k=1}^{f} \int_{t_{k-1}}^{t_k} \omega_k^{dwv} \, dt \tag{4.18}$$

$$x_f = \sum_{k=1}^{f} \int_{t_{k-1}}^{t_k} v^{dwv} \cdot \cos\theta_k \, dt \tag{4.19}$$

$$y_f = \sum_{k=1}^{f} \int_{t_{k-1}}^{t_k} v^{dwv} \cdot \sin\theta_k \, dt \tag{4.20}$$

v^{dwv} と ω_f^{dwv} を用いることで、DWV の経路候補には非円弧経路が含まれる。これは、旋回速度 ω_f^{dwv} を各ステップ毎に障害物の状態に合わせて変更しているからである。

4.2.7 Step 2-7b 衝突確認 (Algorithm 2 line 28-34)

ロボットが障害物と衝突しているか確認する。ロボットが障害物と衝突している経路を経路候補に含めないこととする。

Step 2-1b から Step 2-7b を N^{all} 回繰り返すことで、ロボットの経路候補 $\boldsymbol{O}^{rob}$ が算出される。

4.3 Optimal Path

図 4.1(d) に DWV の最適経路選択の様子を示す。ロボットの経路候補 $\boldsymbol{O}^{rob}$ はコスト関数によって評価される。コスト関数は次の様に導出される。

$$c^{dwv} = W^{pos} \cdot c^{pos} + W^{vel} \cdot c^{vel} + W^{sdo} \cdot c^{sdo} \tag{4.21}$$

ここで、W^{pos} と W^{sdo} は位置と障害物に関する重み係数である。c^{pos} はロボットの予測位置と目的地までの距離、c^{sdo} はロボットと障害物の予測位置との最短距離である。

最後にコスト関数を最大化する経路の並進速度 v^{opt}、旋回速度 ω_1^{opt} とし、ロボットの速度指令値を算出する。

$$v^{cmd} = v^{opt} \tag{4.22}$$

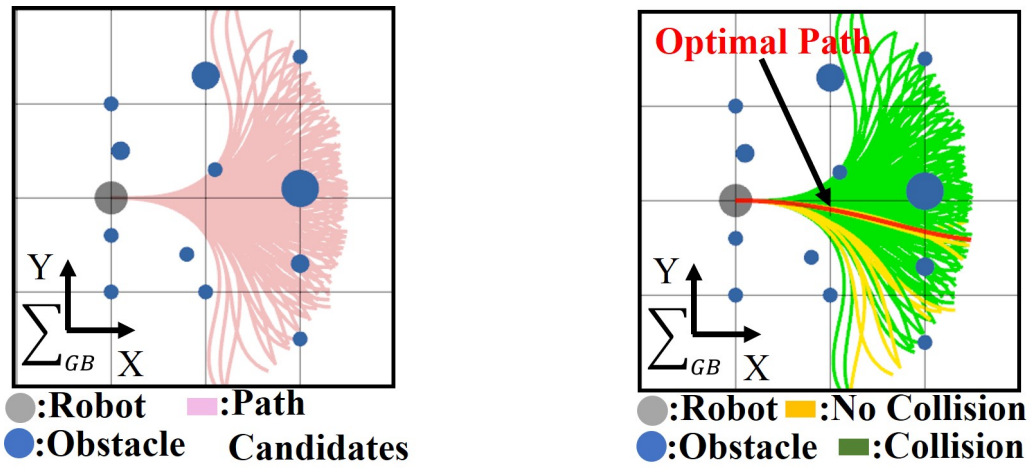
$$\omega^{cmd} = \omega_1^{opt} \tag{4.23}$$

ここで v^{cmd}、ω^{cmd} はロボットの並進・旋回速度指令値である。算出された速度指令値を用いることでロボットは障害物を回避しながらゴールへ到達することが可能となる。

第5章　シミュレーション

5.1　局所経路計画手法

本シミュレーションでは、“SLP”、“DWA”、“DWV” の 3 つの手法を比較した。SLP と DWA を従来手法、DWV を提案手法とした。表 5.1 にシミュレーションで使用したパラメータを記載する。本シミュレーションでは ROS を使用している [32–35]。図 5.1 に SLP の経路生成の様子と最適な経路選択の図を示す。本シミュレーションでは DWV に 2 つの補助マニピュレータのみの VM を搭載し、VM の左右根本位置をロボット座標系において $(0.0, -0.1)$ と $(0.0, 0.1)$ とした。



(a) 経路候補生成　　(b) 最適な候補選択

図 5.1 SLP の概要図

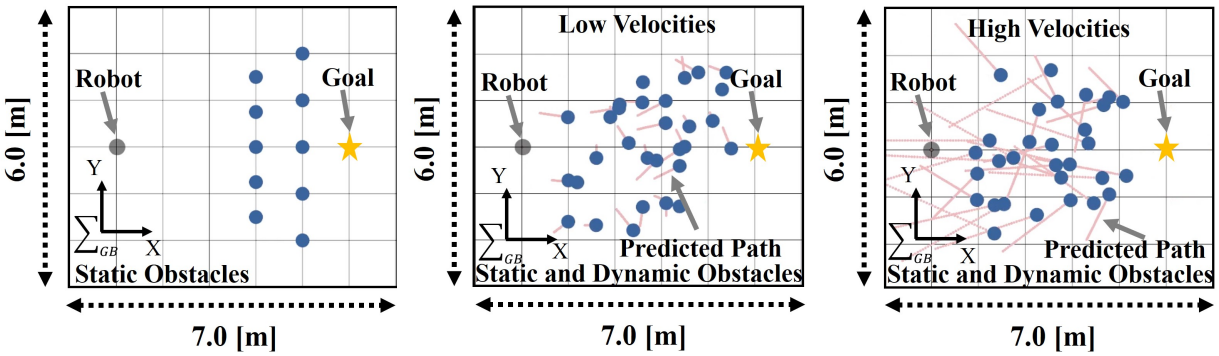
表 5.1 パラメータ

V^{max}	Maximum Translational Velocity	0.55 $[m/s]$
V^{min}	Minimize Translational Velocity	-0.3 $[m/s]$
Ω^{max}	Maximum Angular Velocity	5 $[rad/s]$
Ω^{min}	Minimize Angular Velocity	-5 $[rad/s]$
A^{max}	Maximum Translational Acceleration	2 $[m/s^2]$
Π^{max}	Maximum Angular Acceleration	5 $[rad/s^2]$
N^{tlv}	Number of Translational Velocity	6
N^{agv}	Number of Angular Velocity	20
T^{max}	Maximum Predicted Time	4 $[s]$
ΔT	Time Step	0.1 $[s]$
W^{ang}	Weight Coefficient of Angle	1
W^{vel}	Weight Coefficient of Velocity	5
W^{obs}	Weight Coefficient of Obstacle	0.1
W^{pos}	Weight Coefficient of Position	20
W^{sdo}	Weight Coefficient of Static and Dynamic Obstacle	0.1
D^{max}	Generation Value of Virtual Manipulators	0.5 $[m]$
L^{ass}	Length of Virtual Manipulators	0.3 $[m]$
Λ	Weight Coefficients of Virtual Manipulators	0.075
N^{pos}	Number of Samples in Terminal State Position	40
N^{hea}	Number of Samples in Terminal State Heading	3
L^{pat}	Path Length of SLP	2.5$[m]$
Υ^{min}	Minimum Angular Range of Terminal Position	-60$[deg]$
Υ^{max}	Maximum Angular Range of Terminal Position	60$[deg]$
Φ^{min}	Minimum Angular Range of Heading Angle	-30$[deg]$
Φ^{max}	Maximum Angular Range of Heading Angle	30$[deg]$

5.2　シミュレーション設定

表 5.2 シミュレーション条件

Case	Obstacle	Initial Position	Obstacle Velocity
*Case S*1	Static Obstacles	Constant	0
*Case S*2	Static / Dynamic Obstacles	Random	Random (Low)
*Case S*3	Static / Dynamic Obstacles	Random	Random (High)



(a) Case S1　　(b) Case S2　　(c) Case S3

図 5.2 シミュレーション環境

表 5.2 が示すように 3 パターンのシミュレーションを行った。図 5.2 は *Case S1* から *Case S3* のシミュレーション環境を示した。図 5.2 の灰色と青色の円はロボットと障害物である。図 5.2(b)-(c) ピンク色の線は障害物の移動方向である。全てのケースでのスタート位置とゴール位置は $({}^{GB}x^{start}, {}^{GB}y^{start}) = (0.0, 0.0)$ と $({}^{GB}x^{goal}, {}^{GB}y^{goal}) = (5.0, 0.0)$ とした。ロボットの位置とゴール位置が 0.3 [m] 以下になれば、ゴールに到達したとみなした。*Case S1* から *Case S3* の概要を次に示す。

- *Case S1*: 静的障害物
  図 5.2(a) が示すように、10 個の静的障害物を設置した。

- *Case S2*: 静的/動的障害物 (低速)
  図 5.2(b) が示すように、30 個の静的/動的障害物を設置した。障害物の位置と速度は

シミュレーション開始時にランダムに設定することとした。障害物の速度はロボットの最高速度より小さい 0.0 [m/s] から 0.2 [m/s] の範囲でランダムで選択された。

- *Case S3*: 静的/動的障害物 (高速)
  図 5.2(c) が示すように、30 個の静的/動的障害物を設置した。障害物の位置と速度はシミュレーション開始時にランダムに設定することとした。障害物の速度はロボットの最高速度より大きい 0.0 [m/s] から 0.6 [m/s] の範囲でランダムで選択された。

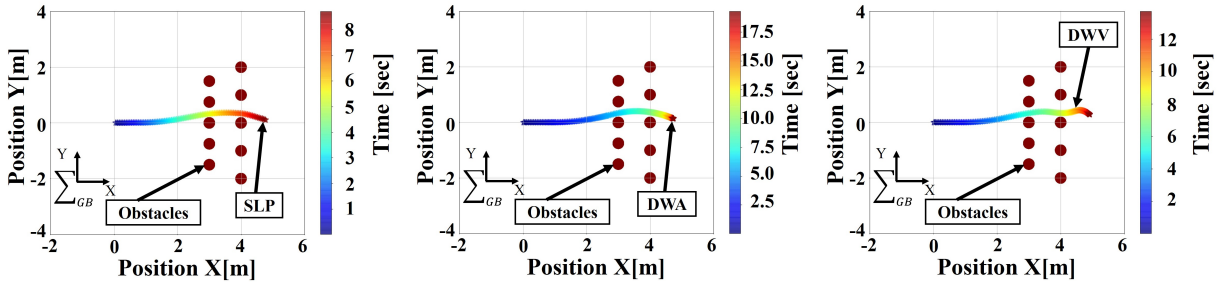
Case S1 ではシミュレーション回数を 1 回行った。*Case S2* と *Case S3* ではシミュレーション回数を 100 回行った。

5.3　シミュレーション結果

表 5.3 にシミュレーション結果を示す。表 5.3 にはゴールに到達した時の成功率、平均到達時間、平均移動距離 (TL)、平均姿勢変位 (PD) を示した。図 5.3 に *Case S1* でのロボットと障害物の軌跡を示す。図 5.4-5.5 に *Case S2* において 100 回シミュレーションを行ったロボットの軌跡を示す。

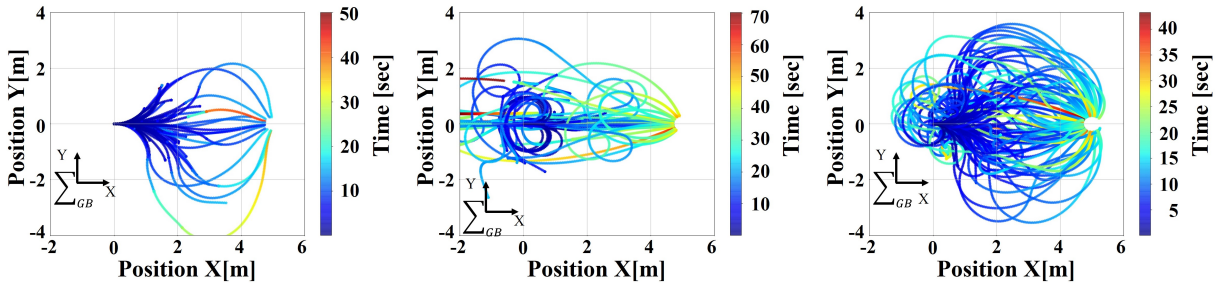
表 5.3 シミュレーション結果

Case	Method	Success [%]	Time [sec]	TL [m]	PD [rad]
*Case S*1	SLP	100	8.708	4.835	1.324
1 time	DWA	100	19.301	4.814	0.950
	DWV	100	13.711	5.009	2.362
*Case S*2	SLP	14	18.223	6.091	2.874
100 times	DWA	32	42.556	9.678	4.398
	DWV	85	20.919	7.562	7.392
*Case S*3	SLP	6	12.959	5.071	2.096
100 times	DWA	4	23.425	5.765	2.422
	DWV	70	22.135	8.003	7.877



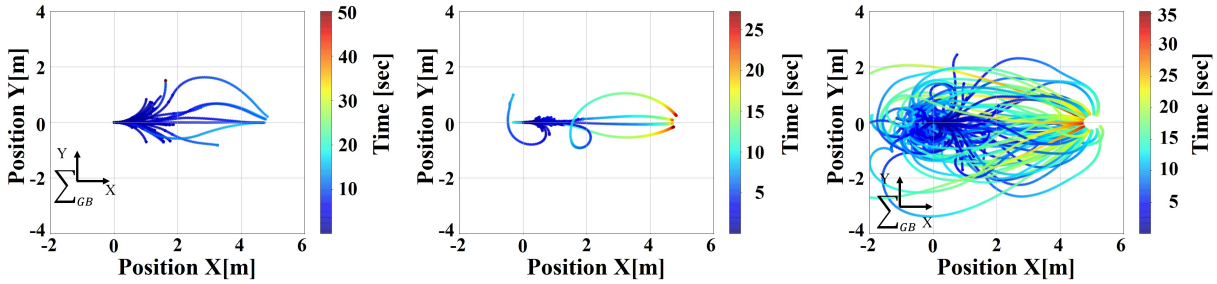
(a) SLP　　(b) DWA　　(c) DWV

図 5.3 シミュレーション結果 (*Case S1*)



(a) SLP　　(b) DWA　　(c) DWV

図 5.4 シミュレーション結果 (*Case S2*)



(a) SLP　　(b) DWA　　(c) DWV

図 5.5 シミュレーション結果 (*Case S3*)

図 5.3 に *Case S1* のシミュレーション結果を示す。図 5.3(a)-(c) には SLP、DWA、DWV 使用時のロボットと障害物の軌跡を時間毎に描画した。*Case S1* の静的環境下では、図 5.3 が

示すように、全ての手法がゴールへ到達した。到達時間はそれぞれ、SLP が約 8.7 秒、DWA が約 19.3 秒、DWV が約 13.7 秒であった。表 5.3 と図 5.3(a) より、*Case S1* で SLP は DWA や DWV よりもゴール地点に早く到達した。

Case S2 では、図 5.4 と表 5.3 が示すようにゴール到達率は SLP が 14%、DWA32%、DWV が 85%であった。図 5.4(a) は SLP の軌跡結果を示し、ゴールへの軌道は直線に近い傾向であった。障害物の位置と速度がランダムで与えられているために、100 回のシミュレーションの中で、障害物がロボットの方向へ向かわない簡単なシチュエーションが生じたためと考えられる。図 5.4(b) は DWA の軌跡結果を示している。DWA が障害物と衝突せずにゴールへ到達した時、DWA は後進の行動を生成したため、他の手法と比べてゴールへの到達平均時間が増加した。表 5.3 と図 5.4(c) のように、DWV がもっともゴール到達率 85% と高い結果になった。よって *Case S2* においてもっとも良い結果を示した手法は DWV であった。

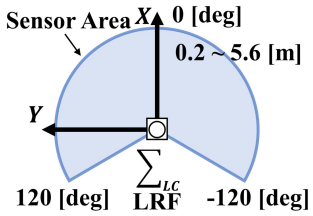
Case S3 では、図 5.5 と表 5.3 が示すようにゴール到達率は SLP が 6%、DWA4%、DWV が 70%であった。図 5.5(a)(b) は SLP と DWA の軌跡結果を示しており、ゴールへの軌道は直線に近い傾向があった。障害物の位置と速度がランダムで与えられているために、100 回のシミュレーションの中で、障害物がロボットへ接近しない簡単なシチュエーションが生じたためと考えられる。表 5.3 と図 5.5(c) のように、DWV がもっともゴール到達率 70% と高い結果になった。よって *Case S3* においてもっとも良い結果を示した手法は DWV であった。以上より、シミュレーション結果から提案手法の有効性が確認された。

第6章　実験

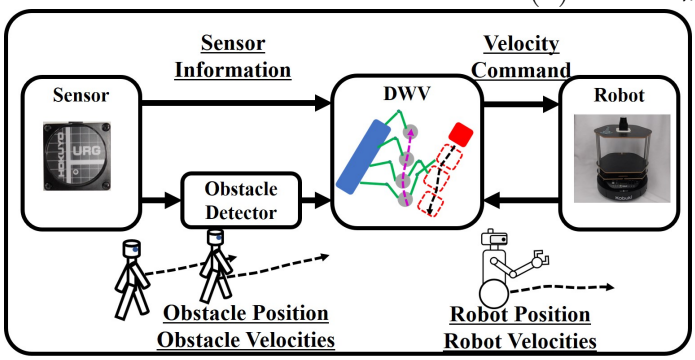
6.1　実験システム

本実験では、図 6.1(a) が示すように Turtlebot2 と ROS Melodic を使用した [36]。Turtlebot2 には URG-04LX-UG01 を搭載した [37]。図 6.1(b) に、測域センサの計測範囲を示す。図 6.1(c) に、実験で使用したシステムの概要図を示す。本システムの動作順序は、まず初めに、LRF を用いて環境情報を取得する。次に、障害物の位置と速度情報を *obastacle_detector* [38] を用いて取得する。最後に、DWV は静的・動的障害物を考慮した速度指令値を生成する。表 5.1 は実験で使用したパラメータであり、シミュレーションと同じ値を使用した。

(a) Turtlebot 2



(b) LRF の計測範囲



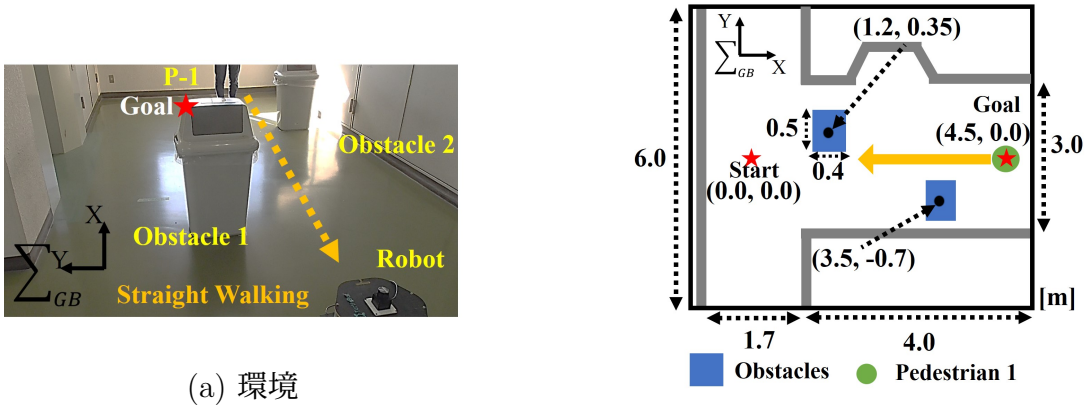
(c) System Configuration

図 6.1 Experiment Setup

6.2 実験条件

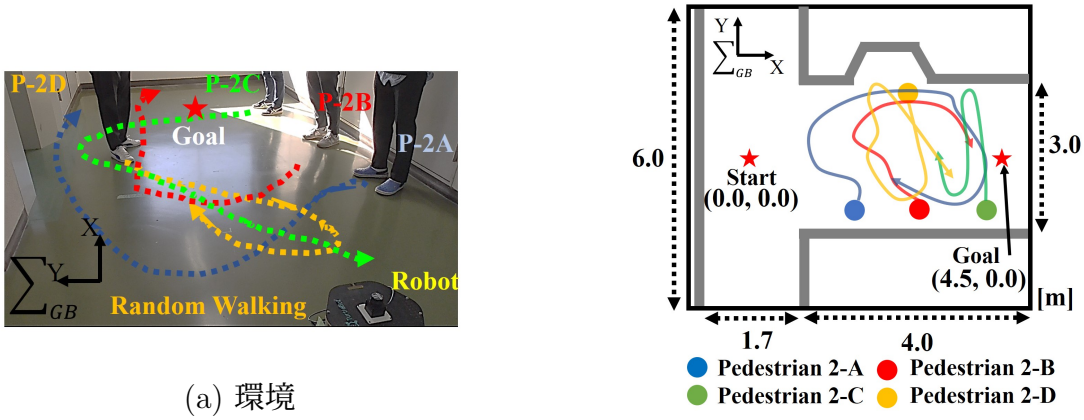
実験では 2 つの環境を使用した。図 6.2-6.3 が示すように、スタートとゴール位置はそれぞれ $({}^{GB}x^{start}, {}^{GB}y^{start}) = (0.0, 0.0)$ と $({}^{GB}x^{goal}, {}^{GB}y^{goal}) = (4.5, 0.0)$ とした。*Case E1* では、図 6.2 が示すように、2 つの静的障害物と歩行者 (P-1) を 1 名配置した。P-1 は実験環境で直線移動し、ロボットが動作すると同時に移動を開始することとした。

Case E2 では、図 6.3 が示すように、歩行者 (P-2A ∼ 2D) を 4 名配置した。4 名の歩行者は環境内をランダムに歩行し、ロボットが動作すると同時に移動を開始することとした。



(a) 環境　(b) 概略図

図 6.2 実験環境 (*Case E1*)

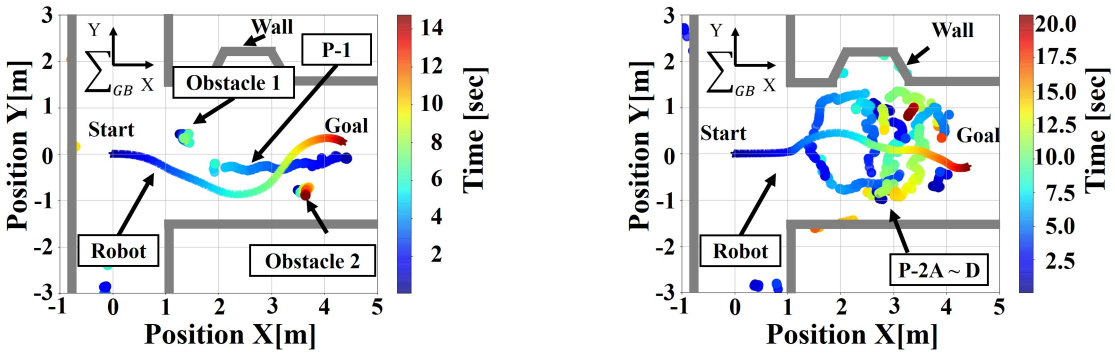


(a) 環境　(b) 概略図

図 6.3 実験環境 (*Case E2*)

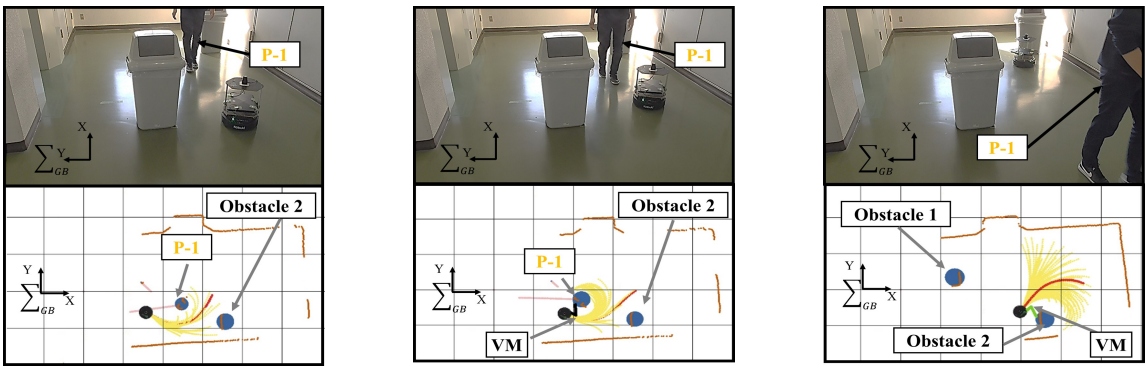
6.3 実験結果

図 6.4 はロボットと歩行者の軌跡を示す。図 6.5-6.6 は任意の時間での経路候補生成や最適経路選択の様子を示す。黒円はロボットを示し、青円は障害物を示す。黒線と緑線は VM、黄線と赤線は経路候補と最適な経路、ピンク線は歩行者の移動方向、オレンジの点は測域センサから取得した点群である。



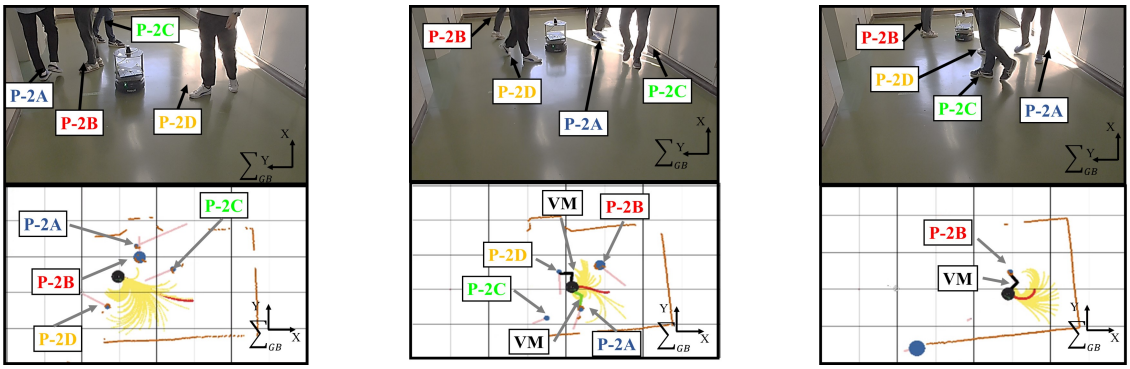
(a) *Case E1* (b) *Case E2*

図 6.4 ロボットと歩行者の軌跡結果



(a) t = 4.0 [sec] (b) t = 5.0 [sec] (c) t = 7.0 [sec]

図 6.5 実験結果 (*Case E1*)



(a) t =6.0 [sec] (b) t =12.0 [sec] (c) t =16.0 [sec]

図 6.6 実験結果 (*Case E2*)

Case E1 では 2 つの静的障害物と 1 人の歩行者を設置した。図 6.4(a) が示すように、ロボットは衝突せずにゴールまで到達した。ロボットの動作様子を図 6.5 を用いて説明する。図 6.5(a) のように、スタートから 4 秒後時点でロボットは Obstacle 1 を回避した。生成された経路候補に注目すると P-1 の進行方向には経路候補を生成せず、P-1 の背後に経路候補を生成した。また経路候補から選択された赤線の最適な経路は、Obstacle 2 と P-1 を回避する経路を選択していることから、静的障害物である Obstacle 2 と動的障害物である P-1 との衝突回避を考慮した行動を選択した。図 6.5(b) のように、スタートから 5 秒後時点でロボットは Obstacle 2 と P-1 を考慮した経路候補を生成した。生成された経路候補に注目すると P-1 の進行方向には経路候補を生成せず、P-1 の背後や Obstacle 2 を回避する経路候補を生成した。図 6.5(c) のように、スタートから 7 秒後時点では、ロボットは P-1 を回避した後に Obstacle 2 を回避する経路候補を生成した。以上より、*Case E1* では静的・動的障害物と衝突することなく、ゴール地点へ到達した。

Case E2 では 4 人の歩行者を設置した。図 6.4(b) が示すようにロボットは衝突せずにゴールまで到達した。図 6.6(a) のように、スタートから 6 秒後時点でロボットは LRF を用いて P-2A～ 2D を認識しており、ロボットは P-2A を回避し、P-2A～ 2D を考慮した経路候補を生成した。図 6.6(b) のように、スタートから 12 秒後時点でロボットは LRF を用いて P-2A～ 2D を認識しており、DWV により P-2A と P-2C を回避し、P-2A、P-2B、P-2D を考慮した経路候補を生成した。図 6.6(c) のように、スタートから 16 秒後時点でロボットは LRF を用いて P-2B を認識しており、ロボットは P-2B を考慮した経路候補を生成した。以上より、*Case E2* では障害物と衝突することなく、ゴール地点へ到達した。

以上 *Case E1* と *Case E2* の結果より、実環境において DWV の動作実現が確認された。

第7章　結論

少子高齢社会や感染症の影響により、サービスロボットによる労働力創出が希求されており、サービスロボットは人と共存する空間で移動する場面が多々求められる。そこで、本論文では「移動」に焦点を当て、知能情報処理技術基づき論じた。サービスロボットが人と共存する空間で移動するためには、時々刻々と変化する動的な環境下に対応する必要がある。一般的にロボットの移動方式は大別して遠隔移動と自律移動に分けられるが、本論文では労働力創出の観点からロボット操作者が不要である「自律移動」に着目した。

自律移動は自己位置推定、地図生成、認識、経路計画等の人工知能・知能情報処理技術に基づいた技術により構成される。そこで本論文では、特に「経路計画」に着目し、知能移動ロボットにおける人の反射的回避動作に基づいた局所的経路計画手法について述べた。

本論文では、人の反射的回避動作を知能情報処理技術基づき考案し、非直線・非円弧経路を含む障害物回避可能な経路生成を、知能移動ロボットの局所的経路計画手法である Dynamic Window Approach with Virtual Manipulators (DWV) を提案した。DWV は、VM と静的・動的障害物の予測位置によって修正された可変速度によって非直線・非円弧経路を含む経路候補を生成する。また、DWV では障害物の予測位置さえ推定できれば、障害物の予測経路が線形もしくは非線形でも対応可能である。そのため静的・動的な障害物が存在する環境においても、非直線経路や非円弧経路を含む障害物回避可能な経路を生成可能となった。様々なシミュレーションにより従来手法との比較や実環境での実機実験により、提案手法の有効性を示した。

以上より、本論文で提案した「知能情報処理技術基づいた局所的経路計画手法である DWV」を用いることで、今後、人等が混在する動的環境下での自律移動実現の一躍進となる。

参考文献

[1] C. Nam, S. Lee, J. Lee, S.H. Cheong, D.H. Kim, C. Kim, I. Kim and S. Park: “A Software Architecture for Service Robots Manipulating Objects in Human Environments”, *IEEE Access*, Vol. 8, pp. 117900-117920, 2020.

[2] M.A.V.J. Muthugala and A.G.B.P. Jayasekara: “A Review of Service Robots Coping With Uncertain Information in Natural Language Instructions”, *IEEE Access*, Vol. 6, pp. 12913-12928, 2018.

[3] L. Lestingi, M. Askarpour, M.M. Bersani and M. Rossi: “A Deployment Framework for Formally Verified Human-Robot Interactions”, *IEEE Access*, Vol. 9, pp. 136616-136635, 2021.

[4] Y. Fuse and M. Tokumaru: “Social Influence of Group Norms Developed by Human-Robot Groups”, *IEEE Access*, Vol. 8, pp. 56081-56091, 2020.

[5] C. Sirithunge, A.G.B.P. Jayasekara and D.P. Chandima: “Proactive Robots With the Perception of Nonverbal Human Behavior: A Review”, *IEEE Access*, Vol. 7, pp. 77308-77327, 2019.

[6] R. Parween, M. Vega Heredia, M.M. Rayguru, R. Enjikalayil Abdulkader and M.R. Elara: “Autonomous Self-Reconfigurable Floor Cleaning Robot”, *IEEE Access*, Vol. 8, pp. 114433-114442, 2020.

[7] M.A.K. Niloy, A. Shama, R.K. Chakrabortty, M.J. Ryan, F.R. Badal, Z. Tasneem, M.H. Ahamed, S.I. Moyeen, S.K. Das, M.F. Ali, M.R. Islam and D.K. Saha, “Critical Design and Control Issues of Indoor Autonomous Mobile Robots: A Review”, *IEEE Access*, Vol. 9, pp. 35338-35370, 2021.

[8] A. Motroni, A. Buffi and P. Nepa: “A Survey on Indoor Vehicle Localization Through RFID Technology”, *IEEE Access*, Vol. 9, pp. 17921-17942, 2021.

[9] Y. Zheng, S. Chen and H. Cheng: “Real-Time Cloud Visual Simultaneous Localization and Mapping for Indoor Service Robots”, *IEEE Access*, Vol. 8,pp. 16816-16829, 2020.

[10] M.B. Alatise and G.P. Hancke: “A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods”, *IEEE Access*, Vol. 8, pp. 39830-39846, 2020.

[11] C. Chen, J. Jiang, N. Lv and S. Li: “An Intelligent Path Planning Scheme of Autonomous Vehicles Platoon Using Deep Reinforcement Learning on Network Edge”, *IEEE Access*, Vol. 8, pp. 99059-99069, 2020.

[12] F. Li, X. Fan and Z. Hou: “A Firefly Algorithm With Self-Adaptive Population Size for Global Path Planning of Mobile Robot”, *IEEE Access*, Vol. 8, pp. 168951-168964, 2020.

[13] M. Luo, X. Hou and J. Yang: “Surface Optimal Path Planning Using an Extended Dijkstra Algorithm, ” *IEEE Access*, Vol. 8, pp. 147827-147838, 2020.

[14] Z. Liu, H. Liu, Z. Lu and Q. Zeng: “A Dynamic Fusion Pathfinding Algorithm Using Delaunay Triangulation and Improved A-Star for Mobile Robots, ” *IEEE Access*, Vol. 9, pp. 20602-20621, 2021.

[15] Q. Jin, C. Tang and W. Cai: “Research on Dynamic Path Planning Based on the Fusion Algorithm of Improved Ant Colony Optimization and Dynamic Window Method”, *IEEE Access*, Vol. 10, pp. 28322-28332, 2022.

[16] U. Patel, N.K.S. Kumar, A.J. Sathyamoorthy and D. Manocha: “DWA-RL: Dynamically Feasible Deep Reinforcement Learning Policy for Robot Navigation among Mobile Obstacles”, *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 6057-6063, 2021.

[17] D. González, J. Pérez, V. Milanés and F. Nashashibi: “A Review of Motion Planning Techniques for Automated Vehicles, ” *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, No. 4, pp. 1135-1145, 2016.

[18] K. Cai, C. Wang, J. Cheng, C.W. De Silva, and M.Q. Meng: “Mobile Robot Path Planning in Dynamic Environments: A Survey”, *Instrumentation*, Vol. 6, No. 2, pp. 92-102, 2019.

[19] P. Fiorini and Z. Shiller: “Motion Planning in Dynamic Environments using Velocity Obstacles”, *The International Journal of Robotics Research*, Vol. 17, No. 7, pp. 760-772, 1998.

[20] J. van den Berg, Ming Lin and D. Manocha: “Reciprocal Velocity Obstacles for Real-Time Multi-Agent Navigation”, *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 1928-1935, 2008.

[21] T. Xu, S. Zhang, Z. Jiang, Z. Liu and H. Cheng: “Collision Avoidance of High-Speed Obstacles for Mobile Robots via Maximum-Speed Aware Velocity Obstacle Method”, *IEEE Access*, Vol. 8, pp. 138493-138507, 2020.

[22] J. Snape, J.v.d. Berg, S.J. Guy and D. Manocha: “The Hybrid Reciprocal Velocity Obstacle”, *IEEE Transactions on Robotics*, Vol. 27, No. 4, pp. 696-706, 2011.

[23] D. Fox, W. Burgard, and S. Thrun: “The Dynamic Window Approach to Collision Avoidance”, *IEEE Robotics & Automation Magazine*, Vol. 4, pp. 23-33, 1997.

[24] M. Dobrevski and D. Skocaj: “Adaptive Dynamic Window Approach for Local Navigation”, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 6930-6936, 2020.

[25] L. Liu, J. Yao, D. He, J. Chen, J. Huang, H. Xu, B. Wang and J. Guo: “Global Dynamic Path Planning Fusion Algorithm Combining Jump-A* Algorithm and Dynamic Window Approach”, *IEEE Access*, Vol. 9, pp. 19632-19638, 2021.

[26] T.M. Howard and A. Kelly: “Optimal rough terrain trajectory generation for wheeled mobile robots”, *The International Journal of Robotics Research*, Vol. 26, No. 2, pp. 141-166, 2007.

[27] T.M. Howard, C.J. Green, A. Kelly, and D. Ferguson: “State Space Sampling of Feasible Motions for High-Performance Mobile Robot Navigation in Complex Environments”, *Journal of Field Robotics*, Vol. 25, No. 6-7, pp. 325-345, 2008.

[28] M. Kobayashi and N. Motoi: “Local Path Planning Method Based on Virtual Manipulators and Dynamic Window Approach for a Wheeled Mobile Robot”, *Proceedings of IEEE/SICE International Symposium on System Integration*, pp. 499-504, 2021.

[29] M. Kobayashi, N. Motoi: “Local Path Planning: Dynamic Window Approach with Virtual Manipulators Considering Dynamic Obstacles”, *IEEE Access*, Vol. 10, pp. 17018-17029, 2022.

[30] K. Yamazaki and M. Inaba: “Trajectory Control of Wheeled Mobile Robots Based on Virtual Manipulators”, *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, pp. 2974-2978, 2009.

[31] 山崎 公俊, 稲葉 雅幸: “仮想マニピュレータを用いた移動ロボットの反射的動作生成法”, 日本ロボット学会誌, 29 巻, 2 号, pp. 163-171, 2011.

[32] H. Zhang, C. Zhang, W. Yang and C.-Y. Chen: “Localization and navigation using QR code for mobile robot in indoor environment”, *Proceedings of IEEE International Conference on Robotics and Biomimetics*, pp. 2501-2506, 2015.

[33] J. Park, R. Delgado and B.W. Choi: “Real-Time Characteristics of ROS 2.0 in Multi-agent Robot Systems: An Empirical Study”, *IEEE Access*, Vol. 8, pp. 154637-154651, 2020.

[34] H. Lee, H. Seo and H. Kim: “Trajectory Optimization and Replanning Framework for a Micro Air Vehicle in Cluttered Environments”, *IEEE Access*, Vol. 8, pp. 135406-135415, 2020.

[35] D. Jin, Z. Fang and J. Zeng: “A Robust Autonomous Following Method for Mobile Robots in Dynamic Environments”, *IEEE Access*, Vol. 8, pp. 150311-150325, 2020.

[36] “Turtlebot2”, https://www.turtlebot.com/turtlebot2/.

[37] “URG-04LX-UG01”, https://www.hokuyo-aut.jp/search/single.php?serial=166.

[38] “obstacle_detector”, https://github.com/tysik/obstacle_detector.