

2024年3月期
関西大学卒業論文

クワッドロータ群のチョークポイント
通過に関する研究

システム理工学部 電気電子情報工学科
情報数理工学研究室

電20-77 塩田 和央

指導教員 准教授 _____ 印
教授 _____ 印

論文要旨

クワッドロータ群のチョークポイント通過に関する研究

電 20-77 塩田 和央

近年、クワッドロータの需要は拡大しており、特に災害現場での遭難者探索といった、現場に人間が侵入できない状況での活躍も期待されている。その中で効率化の観点から、複数のクワッドロータの群飛行によりタスクを遂行する研究が進められている。群飛行によるタスク遂行においては、各クワッドロータの制御入力を管制塔で計算し、全機に一括送信する方法が一般的であるが、いくつか問題点が存在している。まず、災害現場などのインフラストラクチャが整っていない環境では管制塔との通信が難しく、群れの一括制御が困難な場合がある。また、クワッドロータの数が増大すると、全機体の制御入力を管制塔で計算するには演算コストがかかる。更に、倒壊した屋内で遭難者探索を行うような状況では、チョークポイントと呼ばれる幅の狭い通行路が存在することが想定され、固定のフォーメーションで通過できない場合がある。

以上の課題を踏まえ、本研究では管制塔を用いることなく通路幅に応じて柔軟に形状変更可能なフォーメーション制御を行うことで、チョークポイントを通過することを目標とする。フォーメーション制御にはいくつかの手法があるが、本研究ではクワッドロータの群れをリーダとフォロワに分け、リーダにフォロワが追従するリーダ・フォロワ制御を採用する。リーダ・フォロワ制御の手法の一つとして、line-of-sight(LOS)による経路追従を用いた制御が提案されている。しかし、従来手法をそのまま適用した場合には群れのリーダが固定であるため群れの進行方向と逆の位置に目標点が設定された場合に機体を並び替える必要がある。この並び替えを不要とするためのアルゴリズムを提案する。また、従来手法ではフォーメーションの形が固定であるためチョークポイントを通過することができない場合がある。そのため、チョークポイントを通過するためにフォーメーションを変更するアルゴリズムを提案する。以上の提案手法についてシミュレーションによる検証を行った結果、群れの後方に目標点を設定した場合でも機体を並び替えずに進行方向を変更し、チョークポイントを検知した場合にはフォーメーションを変更して通過できることが確認された。

一方で、協調制御に基づいてチョークポイントを通過する手法が提案されている。本手法は障害物センサを搭載したリーダがフォロワを囲むことにより、フォロワは障害物センサを搭載していないなくてもチョークポイントに対応できる。ただし、従来手法では群れの進行方向を変える場合に群れ全体を回転させる必要がある。そこで、この回転を不要とするアルゴリズムを提案し、シミュレーションによる検証を行った結果、目標点が変更された場合でも群れ全体を回転させずに進行方向を変更できることが確認された。

目 次

第 1 章	序論	1
第 2 章	フォーメーション制御	3
2.1	概要	3
2.2	必要な構成要素	4
第 3 章	line-of-sight(LOS) に基づくリーダ・フォロワ制御	6
3.1	LOS による追従制御	6
3.2	追従目標点 ϵ の導出	7
3.3	フォロワの制御器の構成	9
3.4	クワッドロータ間の回避アルゴリズム	9
3.5	追従制御シミュレーション	10
第 4 章	LOS 制御におけるチョークポイント通過	14
4.1	LOS 制御における属性変更アルゴリズムの提案	14
4.2	フォーメーション変更アルゴリズムの提案	15
4.3	属性変更及びチョークポイント通過シミュレーション	16
第 5 章	協調制御に基づくチョークポイント通過	21
5.1	概要	21
5.2	制御入力	22
5.3	協調制御に基づくチョークポイント通過シミュレーション	24
第 6 章	協調制御における属性変更	27
6.1	協調制御における属性変更アルゴリズムの提案	27
6.2	協調制御における属性変更シミュレーション	29

第 7 章 結論	32
謝辞	34
参考文献	35
付 錄 A LOS 制御におけるチョークポイント通過のプログラムリスト	36
付 錄 B 協調制御に基づくチョークポイント通過のプログラムリスト	57

第1章 序論

クワッドロータの使用用途は多岐にわたり、災害時における情報収集や救援活動など様々な分野において活躍が期待されている。その中で、複数のクワッドロータの群飛行により効率的にタスクを遂行する研究が進められている[1]。

群飛行によりタスク遂行を行う場合、従来手法の多くは管制塔による一括制御を採用しているが、実際の状況を想定した場合に複数の問題点が挙げられる。まず、クワッドロータの機体数が増加するとすべての機体の情報を収集して最適な制御入力を計算するには大規模な演算が必要となる。更に、災害現場などのインフラストラクチャが整っていない環境では複数のクワッドロータを管制塔を用いて一括制御することが困難になりうる。本研究ではそのような状況でも個々のクワッドロータが周囲のクワッドロータから得られる居所的な情報のみを利用して群れとして動作することを目指す。このような制御のためにフォーメーション制御を用いる。

フォーメーション制御には大きく分けて三つの手法がある。一つは行動ベース (behavioral based) と呼ばれるもの[2]、もう一つは仮想構造ベース (virtual structure based) と呼ばれるもの[3]、もう一つはリーダ・フォロワ制御と呼ばれるものである[4]。複数の自律移動ロボット群に適用できる一般的なフォーメーション制御の手法として、line-of-sight(LOS) 誘導による経路追従を用いたリーダ・フォロワ制御が提案されている[5]。この手法を用いることで、LOS 誘導によりフォロワがリーダから一定距離だけ離れた追従目標点に向かう制御器を実装できる[6, 7]。本手法を用いた先行研究では、リーダが他の機体の上を通過したときに、リーダが発する風の影響で通過された側の機体が流され、予期しない挙動を示す現象を回避するアルゴリズムが実装されている。

先行研究の制御器では、どの機体がリーダであるのかが固定であるため、群れの進行方向と逆の位置に目標点が設定された場合に機体を並び替える必要がある。この並び替えを不要とするため、群れのリーダを変更するアルゴリズムを提案する。また、本論文では LOS に基づくリーダ・フォロワ制御を実装したうえで、クワッドロータ群が屋内を探索することを考える。探索時は広がりのあるフォーメーションで探索を行ったほうが効率が良いが、屋内では様々な障害物により、チョークポイントと呼ばれる幅の狭い通行路が存在することが想定される。広がりのあるフォーメーションではチョークポイントを通過できないためフォーメーションを変更する必要があるが、先行研究の制御器ではフォーメーションが固定であることが問題となる。そのため、チョーク

ポイント通過時にフォーメーションを変更するアルゴリズムを提案する。

他のチョークポイント通過手法として、協調制御に基づくものが提案されている[8]。この手法は障害物センサを搭載した少数のリーダがフォロワを囲むことにより、フォロワは障害物センサを搭載せずにチョークポイントに対応できる。しかし、この手法をそのまま用いた場合、群れの進行方向を変えるために群れ全体を回転させる必要がある。この回転を不要とするために、各クワッドロータの群れの中での役割を変更する属性変更アルゴリズムを提案する。本論文では上で提案したアルゴリズムを動力学シミュレーション CoppeliaSim と MATLAB を連携させてシミュレーションを行う[9]。

本論文の構成を以下に示す。第1章では本研究の背景と目的について述べる。第2章ではフォーメーション制御について述べる。第3章ではLOSに基づくリーダ・フォロワ制御について述べ、第4章ではLOS制御におけるチョークポイント通過手法について述べる。第5章では協調制御に基づいたチョークポイント通過手法について述べる。第6章では協調制御において、群れの進行方向を変更する際の回転を不要とするアルゴリズムについて述べる。第7章では本研究の総括を行う。

第2章 フォームーション制御

2.1 概要

マルチエージェントシステムとは、多数の自律的に意思決定を行うことができる構成要素からなるシステムのことである。各要素をエージェントと呼び、本研究ではクワッドロータをエージェントとする。それらのエージェントが互いに通信することでシステム全体の振る舞いを決定することができる。しかし、エージェントは物理的に分散しているため、それぞれの持つ情報が異なる。そのため、各エージェントは何らかの形で情報交換を行う必要がある。特に大規模なシステムでは、すべての情報をエージェント間で共有することは現実的ではなく、距離が近いエージェント間で局所的に共有することとなる。

マルチエージェントシステムの制御の中で最も基礎的かつ応用ができるものとして、フォームーション制御が挙げられる。フォームーション制御とは、複数のエージェントがそれぞれ得た情報をもとに自律的に移動し、フォームーションを形成、またそれを維持したまま移動することを目標とした制御である。エージェント間の安定した距離の確保による衝突回避やセンサ性能の向上はタスクを遂行するにあたってのパフォーマンスを向上させることに繋がると考えられる。

フォームーション制御の手法については様々な手法が提案されており、大きく分けて三つの手法がある。一つは、行動ベースと呼ばれるものである。この手法では、探索、フォームーション形成、衝突回避などの機能を個々のエージェントに同じように実装し、センサから得られた値を用いてそれぞれのエージェントが動作の重み付けを変えることで個別の動作を行う。もう一つは仮想構造ベースと呼ばれるものである。この手法では、複数のエージェントのフォームーションを剛体と捉えて、一つの集合として制御する。もう一つは、リーダ・フォロワ制御と呼ばれるものである。この手法では、複数のエージェントをリーダとそれ以外のフォロワに分け、フォロワがリーダに安全な距離を保ちながら追従することを目指す。本研究では複数のクワッドロータにリーダ・フォロワ制御を実装する。

2.2 必要な構成要素

本研究において採用するマルチエージェントシステムでのリーダ・フォロワ制御では管制塔を用いた一括での制御を想定していない。各エージェントの位置や速度といった状態をネットワークにおける情報交換を通じて、フォーメーションを形成し、維持しながら追従することを目指す。ここで、リーダにあたるエージェントは自由に操縦されるものとする。このような場合においてリーダ・フォロワ制御を実現するためには、リーダを含む各エージェントの状態変数をリアルタイムで測定する知覚システムや、得られた状態変数を用いて適切な制御入力を決定する制御器、制御入力を移動ロボットの動的モデルに基づき適切なアクチュエータの出力に変換する制御システムが必要となる。

図 2.1 にフォーメーション制御を実現するために必要な構成要素を示す。

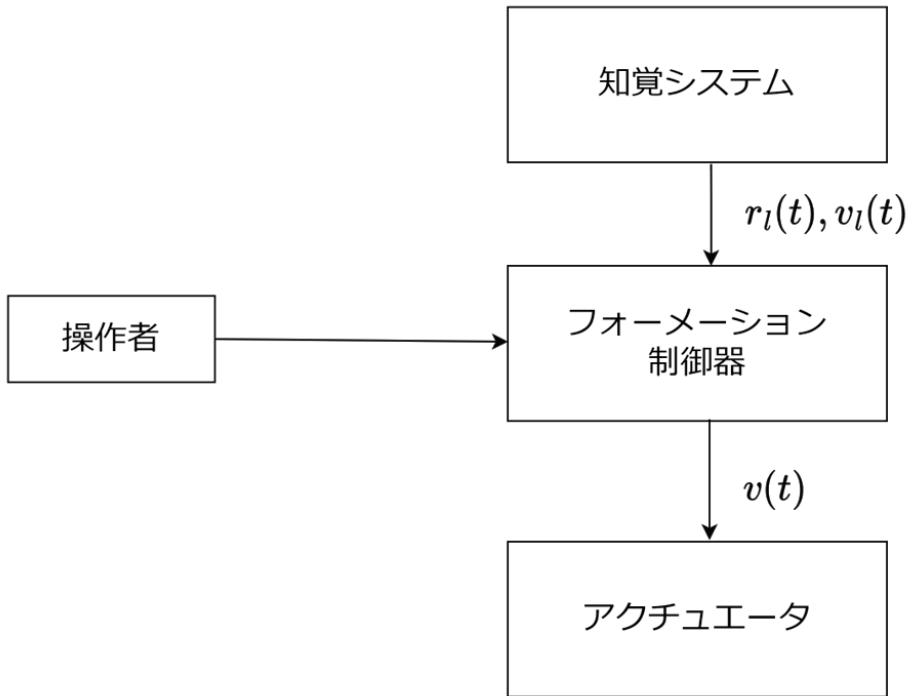


図 2.1: フォーメーション制御の構成要素

後の章では、知覚システムから必要な情報が得られることを前提として、クワッドロータを制御対象とした場合のフォーメーション制御器の設計について議論する。加えて、本研究ではクワッドロータ群がリーダ・フォロワ制御を用いてフォーメーションを形成、維持したうえで屋内を探索するような状況について考える。屋内の探索では

様々な障害物により幅の狭い通行路が存在することが想定される。これをチョークポイントと呼ぶ。本論文ではチョークポイントを通過する手法についても議論する。

第3章 line-of-sight(LOS)に基づく リーダ・フォロワ制御

リーダ・フォロワ制御とは、マルチエージェント制御において安全なフォーメーションを維持するため、リーダであるエージェントに対してフォロワが安全な距離を保ちながら追従できるようフォーメーションを形成、維持しながら走行する制御方式である。この制御の利点は、エージェントの数が増えても単一の制御器で各フォロワの動作を決定することができるため、実装が簡単な点である。外乱を除いて、フォーメーションの安定性は制御器の設定およびリーダの軌道によってのみ左右される。

3.1 LOSによる追従制御

フォロワが制御入力により指定された目標の経路を追従するために、Line-of-sight(LOS)誘導と呼ばれる経路追従方式がよく用いられる。

図 3.1 に示すように、LOS では経由点 p_1, p_2, p_3, \dots を結んだ目標経路に追従することを目指す。このとき、機体を中心とした半径 L の円が目標とする経由点 p_k とその 1 つ前の経由点 p_{k-1} を結んだ線分と交差する点 p_{los} を目標点とする。ここで p_{los} の座標は (x_{los}, y_{los}) とする。また、機体位置が p_k を中心とした半径 R_k の範囲に到達すると、目標とする経由点を p_{k+1} に切り替えて同じ手順を繰り返す。

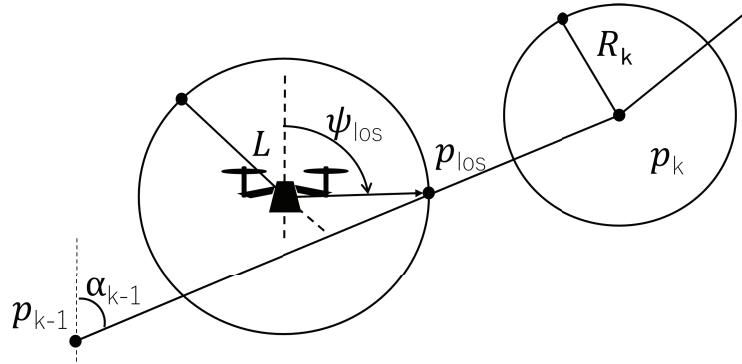


図 3.1: LOS の図解

x_{los}, y_{los} の値は式 (3.1) と式 (3.2) の連立方程式を解くことで求めることができる。

$$(y_{los} - y)^2 + (x_{los} - x)^2 = (nL_{pp})^2 \quad (3.1)$$

$$\frac{y_{los} - y_{k-1}}{x_{los} - x_{k-1}} = \frac{y_k - y_{k-1}}{x_k - x_{k-1}} = \tan(\alpha_{k-1}) \quad (3.2)$$

機体は、求めた x_{los}, y_{los} に向かって移動することで目標経路に追従する。

これが一般的な LOS 誘導の概要である。本研究では LOS に基づく制御器でフォロワを制御し、リーダからの相対位置と角度で指定された追従目標点 ε に追従させる。

3.2 追従目標点 ε の導出

LOS による追従手法においてフォロワはリーダの速度ベクトルに平行な経路 P 上の点を目指して操縦される。この点を追従目標点 ε と呼ぶ。ここでは追従目標点 ε の位置ベクトルの導出を行う。

まず、エージェントの xy 座標のみを考える。二次元でのリーダとフォロワおよび追従目標点 ε の関係を図 3.2 に示す。

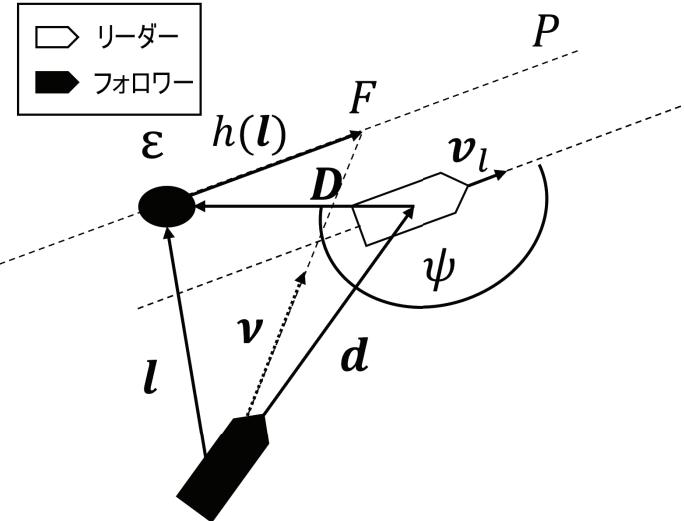


図 3.2: 二次元での追従の図解

リーダの位置から追従目標点 ε への位置ベクトル D の導出を行う。リーダの速度ベクトルを角度 ψ だけ回転させたものがベクトル D であり、その大きさは D である。なお、この角度 ψ および大きさ D は任意で決められるパラメータであるとする。このとき、ベクトル D は

$$D = DR(\hat{v}_l^z(t), \psi)\hat{v}_l(t) \quad (3.3)$$

と表すことができる。式(3.3)において、 $\hat{\mathbf{v}}_l(t)$ はリーダの速度の単位ベクトル、 $\hat{\mathbf{v}}_l^z(t)$ はリーダのローカル座標系での z 軸の単位ベクトルを表し、後述する $\mathbf{R}(\hat{\mathbf{v}}_l^z(t), \psi)$ は、軸 $\hat{\mathbf{v}}_l^z(t)$ を中心にして角度 ψ だけ回転させる回転行列を表す。回転行列 \mathbf{R} を前から掛けることで幾何学平面におけるベクトルの回転を表すことができる。

次に、三次元でのベクトル \mathbf{D} は

$$\mathbf{D} = D\mathbf{R}(\hat{\mathbf{v}}_l^x(t), \phi)\mathbf{R}(\hat{\mathbf{v}}_l^z(t), \psi)\hat{\mathbf{v}}_l(t) \quad (3.4)$$

と表せる。二次元と同じように軸 $\hat{\mathbf{v}}_l^z(t)$ を中心にして角度 ψ だけ回転させた後、リーダのローカル座標系での x 軸の単位ベクトルを中心として回転することでフォロワをリーダに対して z 座標に変化をつけ追従させる。式(3.4)内の $\hat{\mathbf{v}}_l^x(t)$ はリーダのローカル座標系の x 軸の単位ベクトルを代入する。

前述した、ベクトル \mathbf{u} を角度 θ だけ回転させる際の回転行列は

$$\mathbf{R} = \cos \theta \mathbf{I} + \sin \theta [\mathbf{u}]_{\times} + (1 - \cos \theta)(\mathbf{u} \otimes \mathbf{u}) \quad (3.5)$$

と表すことができる。式(3.5)において、 \mathbf{I} は単位行列を表し、 $[\mathbf{u}]_{\times}$ はクロス積行列

$$[\mathbf{u}]_{\times} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \quad (3.6)$$

を表す。

また、 $\mathbf{u} \otimes \mathbf{u}$ は直積

$$\mathbf{u} \otimes \mathbf{u} = \begin{bmatrix} u_x^2 & u_x u_y & u_x u_z \\ u_x u_y & u_y^2 & u_y u_z \\ u_x u_z & u_y u_z & u_z^2 \end{bmatrix} \quad (3.7)$$

を表す。

導出したベクトル \mathbf{D} およびフォロワからリーダへの位置ベクトル \mathbf{d} から、フォロワから追従目標点 ε への位置ベクトル ℓ を導出する。ここで、フォロワからリーダへの位置ベクトル \mathbf{d} は

$$\mathbf{d} = \mathbf{r}_l(t) - \mathbf{r}_f(t) \quad (3.8)$$

である。よって、フォロワから追従目標点 ε への位置ベクトル ℓ は

$$\ell = \mathbf{d}(t) + \mathbf{D} \quad (3.9)$$

で求められる。

3.3 フォロワの制御器の構成

3.2 節で求めたフォロワから追従目標点 ε への位置ベクトル ℓ を用いて、フォロワの制御入力 $\nu(t)$ を導出する制御器の設定について記述する。

本手法ではフォロワの目標速度を制御入力とし、目標速度の方向を表す単位ベクトル $\hat{\nu}(t)$ と大きさを表すスカラ $\nu(t)$ をそれぞれ計算したのち、その積が最終的な制御入力 $\nu(t)$ となる。すなわち、

$$\nu(t) = \nu(t)\hat{\nu}(t) \quad (3.10)$$

である。まず、制御入力の方向を表す単位ベクトル $\hat{\nu}(t)$ について、

$$\hat{\nu}(t) := \hat{f}(t) \quad (3.11)$$

である。ここで、 $\hat{f}(t)$ とはフォロワから経路 P に沿って進んだ点 F への位置ベクトル $f(t)$ の単位ベクトルであり、位置ベクトル $f(t)$ は

$$f(t) = \ell(t) + h(t)\hat{v}_l(t) \quad (3.12)$$

である。ここで、式 (3.12) の第二項の係数である $h(t)$ は

$$h(t) = k_0 + \frac{1}{1 + \|\ell(t)\|}k_l \quad (3.13)$$

である。式 (3.13) における k_0 および k_l はゲインパラメータと呼ばれる任意の定数である。 k_0 は $h(t)$ の初期値にあたり、 k_l はベクトル $\ell(t)$ が十分小さくなったとき、つまりフォロワが追従目標点 ε に十分接近したときの $\hat{\nu}(t)$ の値に影響を及ぼす。

次に制御入力の大きさを表すスカラ $\nu(t)$ は

$$\nu(t) = v_l(t) \left[1 + \frac{2k_p}{\pi} \tan^{-1} \left(\frac{\ell(t) \cdot \hat{v}_l(t)}{k_s} \right) \right] \quad (3.14)$$

である。式 (3.14)において、 k_p および k_s はゲインパラメータと呼ばれる任意の定数である。 k_p はベクトル $\ell(t)$ の値に関わらず常に $\nu(t)$ の値に一定の変化を与え、 k_s は $\nu(t)$ の応答性に関する。

3.4 クワッドロータ間の回避アルゴリズム

3.3 節で述べた制御器はフォロワの速度ベクトルについてのみ考慮しており、リーダとフォロワ間の距離については考慮していない。そのためリーダの軌道によっては

リーダがフォロワに対して接近しフォロワの動作の妨げとなる場合がある。具体的には、三次元の回転によりリーダの z 座標の値がフォロワの z 座標の値よりも大きい場合に、フォロワの速度が0のまま xy 平面上でリーダがフォロワに接近し、リーダがフォロワの真上にきた場合にはリーダにより発生した風の影響を強く受け、フォロワの挙動が想定と大きく異なったものになってしまう。この状況を回避するために回避アルゴリズムを実装する。

各クワッドロータに番号を割り当て、優先度を決めたうえで回避アルゴリズムを導入する。リーダには1を、フォロワには2から順に番号を割り当てる。優先度としてはリーダが最優先であり、数字が大きいクワッドロータの動作が優先され、数字の小さいクワッドロータが回避行動をとる。低優先度の機体の位置ベクトル $\mathbf{r}_a(t)$ と高優先度の機体の位置ベクトル $\mathbf{r}_p(t)$ の差は

$$\mathbf{d}_a = \mathbf{r}_a(t) - \mathbf{r}_p(t) \quad (3.15)$$

となる。この \mathbf{d}_a の単位ベクトルである $\hat{\mathbf{d}}_a$ に高優先度の機体の速さ $v_p(t)$ を掛け、

$$\boldsymbol{\nu}_a(t) = \hat{\mathbf{d}}_a v_p(t) \quad (3.16)$$

として求めた $\boldsymbol{\nu}_a(t)$ を低優先度の機体の制御入力とする。式(3.16)の制御入力を与えることにより低優先度の機体が高優先度の機体から離れていく動作となる。ここで、高優先度の機体の速さを掛け、高優先度の機体と同じ速さで回避することで高優先度の機体に追いつかれることを防いでいる。

3.5 追従制御シミュレーション

本シミュレーションは3.3節で述べた制御器を用いて追従制御を行えるかを検証する。ここでは、指定された目標点に向かって直進するリーダ1機を参照し、4機のフォロワが追従する状況を想定する。図3.3に群れが形成するフォーメーションを示す。図3.3において、 L はリーダ、 F はフォロワである。

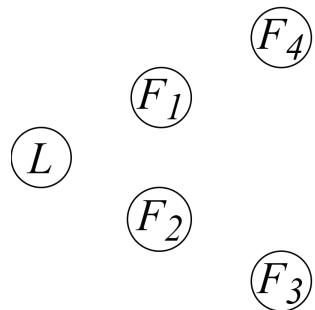


図 3.3: 形成するフォーメーション

図3.4から図3.7に追従制御時のクワッドロータの動作を示す。目標点は黒の円筒で示しており、その位置は $(5, -0.15, 2.5)$ である。リーダの制御入力は、リーダの現在位置から目標点への単位ベクトルを求め、事前に指定した速さ $v_l = 5$ を掛けて求めている。表3.1に各機体の初期位置、リーダからの相対距離 D 、リーダからの相対角度 ϕ, ψ を示す。

表 3.1: 追従制御シミュレーションのパラメータ

	初期位置	相対距離 $D[m]$	相対角度 $\phi, \psi[^{\circ}]$
リーダ	$(-4, 0, 2.2)$	—	—
フォロワ 1	$(-4.2, -1.1, 2.5)$	1	$0, -140$
フォロワ 2	$(-5, -0.6, 2.5)$	1	$0, 140$
フォロワ 3	$(-5.2, 0.45, 2.5)$	2	$0, 140$
フォロワ 4	$(-6, -1.1, 2.5)$	2	$0, -140$

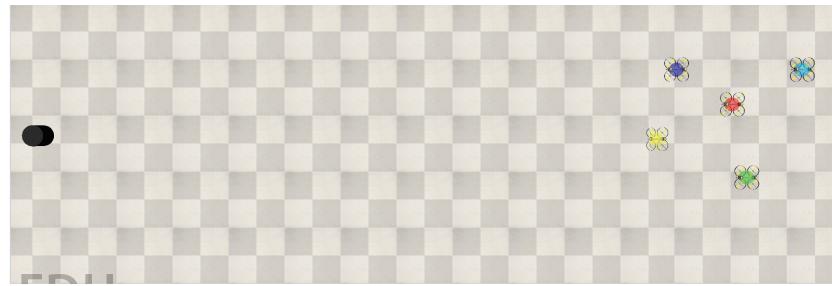


図 3.4: 初期位置

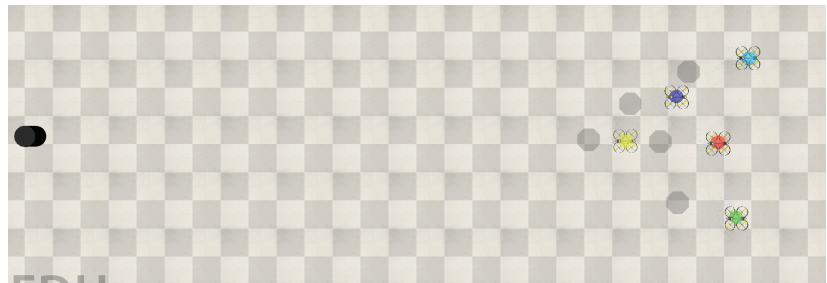


図 3.5: フォームーション形成



図 3.6: フォーメーション維持

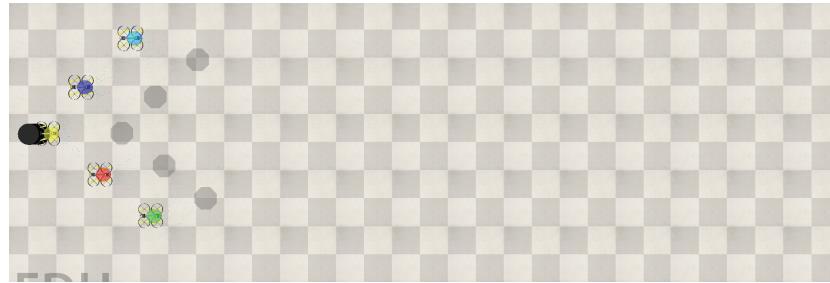


図 3.7: 目標点到達

図 3.4 は初期位置である。図 3.5 はリーダである黄の機体にフォロワが追従し、フォーメーションを形成している。図 3.6 はフォーメーションを形成したのち、維持しながら進行している。図 3.7 はフォーメーションを維持したまま進行し、目標点に到達した。

ステップ数と各フォロワの現在位置から追従目標点への距離（追従誤差）の関係を図 3.8 から図 3.11 に示す。追従誤差は式 (3.9) で求めたベクトルの大きさである。

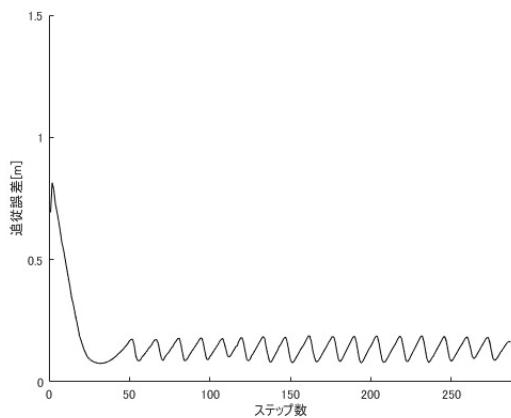


図 3.8: フォロワ 1 の追従誤差

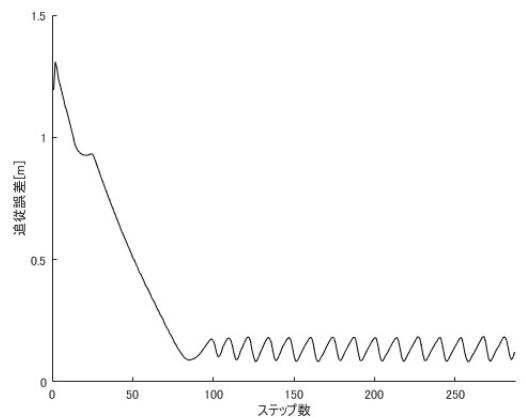


図 3.9: フォロワ 2 の追従誤差

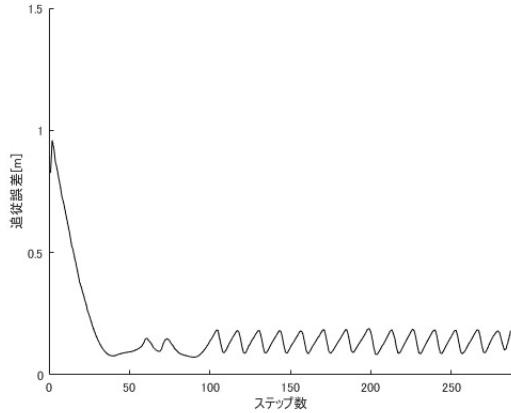


図 3.10: フォロワ 3 の追従誤差

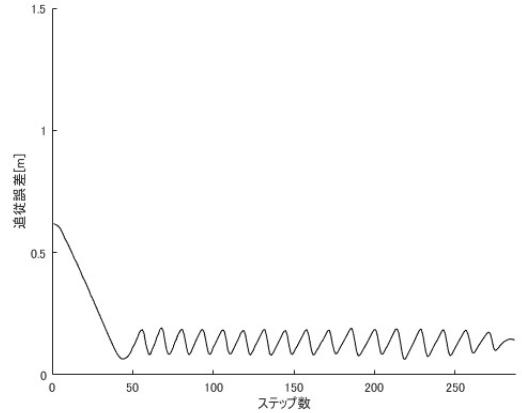


図 3.11: フォロワ 4 の追従誤差

図 3.8 から図 3.11 よりステップ数が 0 から 100 付近の範囲でステップ数の増加に伴い、追従誤差が減少しているとわかる。ステップ数が 100 以上の範囲で追従誤差が増加と減少を繰り返すのは、リーダが目標点に向かい直進する毎に目標追従点も新たに指定されるためである。以上の結果より、3.3 節で述べた制御器を用いることによりリーダにフォロワを追従させフォーメーションを形成することができた。

第4章 LOS制御におけるチョークポイント通過

4.1 LOS制御における属性変更アルゴリズムの提案

3章で述べた制御器では、どの機体がリーダまたはフォロワであるのかが固定であるため群れの進行方向と逆の位置に目標点が設定された場合には機体を並び替える必要がある。この並び替えを不要とするために属性変更アルゴリズムを提案する。3.4節で述べたように各機体には番号が割り当てられており、この番号に基づいて機体の動作の優先度が決められている。リーダには1、フォロワには2から順に番号が割り当てられており、この番号を属性としている。この番号の割り当てを変更することにより、どの機体がリーダまたはフォロワであるかを示す属性を変更する。属性は各機体と目標点との距離に基づいて決定される。目標点との距離が最小の機体をリーダとし、フォロワは目標点との距離が小さいものから順に大きい数字を割り当てる。

図4.1と図4.2を用いて具体的に述べる。図4.1と図4.2において、1を割り当てている機体がリーダであり、フォロワはリーダからの距離が近い順に3、2を割り当てている。

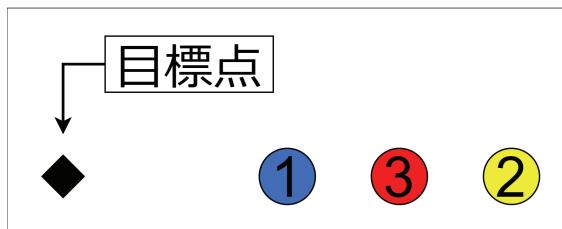


図 4.1: 属性変更前



図 4.2: 属性変更後

図4.1では目標点が左に設定されている。各機体と目標点との距離は青、赤、黄の順に小さいため属性は青は1、赤は3、黄は2が割り当てられる。ゆえにリーダとなる青の機体に赤と黄の機体がフォロワとして追従しながら目標点に向かい進行する。図4.1の左に進行している状態から目標点のみが変化し、図4.2に示すように右に設定された場合を考える。各機体と目標点との距離は黄、赤、青の順に小さい。ゆえに属性は黄

4.2. フォーメーション変更アルゴリズムの提案

関西大学 システム理工学部 電気電子情報工学科

に1, 赤に3, 青に2を割り当てる。リーダとなる黄の機体に赤と青の機体が追従して目標点に向かい進行することで機体の並び替えを不要としている。

4.2 フォーメーション変更アルゴリズムの提案

本節ではチョークポイントを通過するためのフォーメーション変更アルゴリズムについて述べる。屋内での探索を考えると、探索時は広がって探索した方が効率が良いが、そのフォーメーションのままではチョークポイントを通過することが出来ない場合がある。ゆえに探索時とチョークポイント通過時は異なるフォーメーションを組む必要がある。この問題に対して、チョークポイントを検知した際に、フォーメーションの形を変更するフォーメーション変更アルゴリズムを提案する。

図4.3と図4.4に示すような広がりのある形のフォーメーションとチョークポイントを通過するための一直線の形のフォーメーションを予め設定しておく。

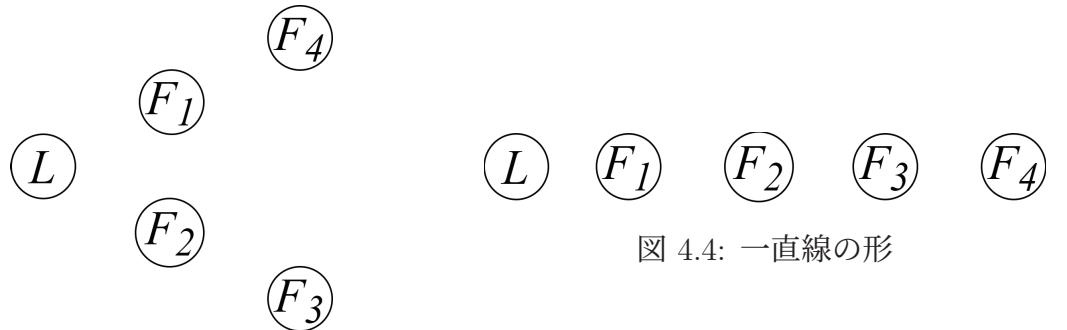


図 4.3: 広がりのある形

ここで、すべての機体は障害物を検知するためのセンサとして LiDAR を搭載していると想定する。リーダの LiDAR 情報を基にして、チョークポイントを検知していない場合は広がりのある形のフォーメーションを形成、維持しながら目標点に向かい進行する。目標点に向かい進行している中で、チョークポイントを検知した場合は通過するために一直線の形のフォーメーションを形成、維持する。チョークポイントの通過判定については、一直線の形のフォーメーションにおいて群れの先頭のリーダと群れの最後尾のフォロワの LiDAR 情報を基に、両方の機体がチョークポイントを検知しなくなった場合に通過したと判定している。屋内での探索を想定するとチョークポイント通過後は再度、広がって探索した方が効率が良いため、通過後は広がりある形のフォーメーションを形成する。

4.3. 属性変更及びチョークポイント通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

4.3 属性変更及びチョークポイント通過シミュレーション

本シミュレーションは、属性変更アルゴリズムにより目標点が群れの進行方向と逆の位置に設定された場合に並び替えが不要となるか及びフォーメーション変更アルゴリズムを用いることでチョークポイントを通過することができるかを検証する。ここでは、指定された目標点に向かって直進するリーダ1機を参照し、4機のフォロワが追従する状況を想定する。図4.5から図4.8に属性変更及びフォーメーション変更によるチョークポイント通過の動作を示す。2個の直方体の間がチョークポイントである。目標点は黒の円筒で示しており、目標点とリーダの距離が一定以下になった場合に目標点に到達したと判定し、次の目標点を設定する。1個目の目標点の位置は(11, -0.15, 2.5)で、2個目の目標点の位置は(-6.5, -0.15, 2.8)である。リーダの制御入力はリーダの現在位置から目標点への単位ベクトルを求め、速さ $v_l = 5$ を掛けて求める。表4.1に各機体の初期位置、各フォーメーションにおけるリーダからの相対距離 D 、リーダからの相対角度 ϕ, ψ を示す。

表 4.1: 属性変更及びフォーメーション変更シミュレーションのパラメータ

	初期位置	広がりのある形		一直線の形	
		相対距離 $D[m]$	相対角度 $\phi, \psi[^{\circ}]$	相対距離 $D[m]$	相対角度 $\phi, \psi[^{\circ}]$
リーダ	(-4,0,2.2)	—	—	—	—
フォロワ 1	(-4.2,-1.1,2.5)	1	0,-140	0.8	0,180
フォロワ 2	(-5,-0.6,2.5)	1	0,140	1.6	0,180
フォロワ 3	(-5.2,0.45,2.5)	2	0,140	2.4	0,180
フォロワ 4	(-6,-1.1,2.5)	2	0,-140	3.2	0,180

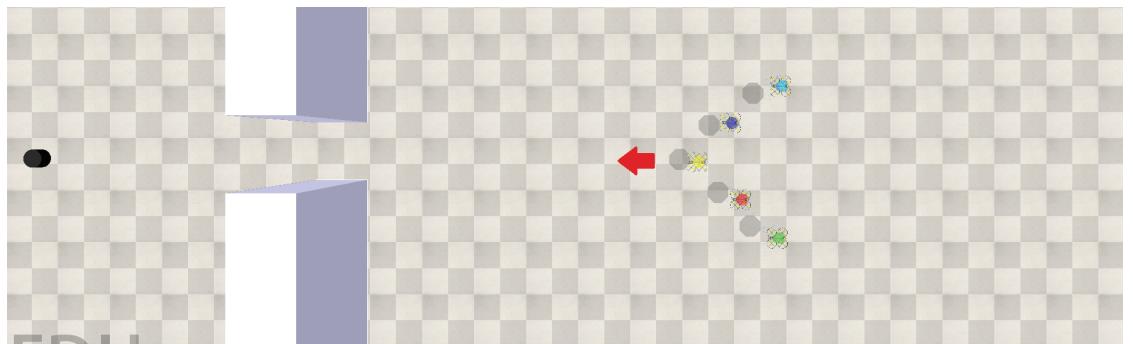


図 4.5: チョークポイント検知前

4.3. 属性変更及び choke point 通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

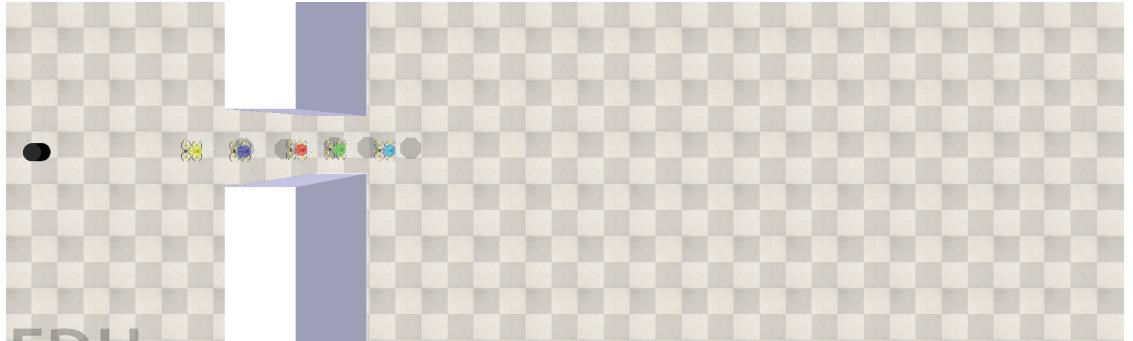


図 4.6: 一直線の形のフォーメーションを形成して choke point を通過



図 4.7: 属性変更アルゴリズム適用



図 4.8: choke point 通過後

図 4.5 は黄の機体がリーダで、その他の機体がフォロワである。この段階では群れの先頭のリーダは choke point を検知しておらず、群れは広がりのある形のフォーメーションを形成、維持して目標点に向かい進行している。図 4.6 は群れの先頭のリーダが choke point を検知したため、一直線の形のフォーメーションを形成、維持して choke point を通過している。図 4.7 は 2 個目の目標点が設定されたため、属性変更アルゴリズムを適用した。各機体と目標点との距離を考えると水色の機体が最小であるためリーダとなり、他の機体がフォロワとして追従している。図 4.8 は choke

4.3. 属性変更及び choke ポイント通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

クポイントを通過したと判定したのち、広がりのある形のフォーメーションを形成して目標点に向かい進行している。

ステップ数と各フォロワの追従誤差の関係を図 4.9 から図 4.12 に示す。

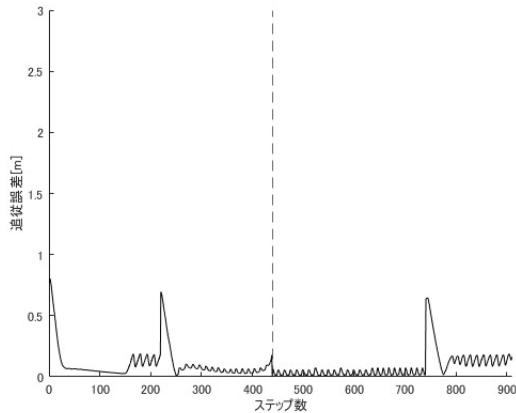


図 4.9: フォロワ 1 の追従誤差

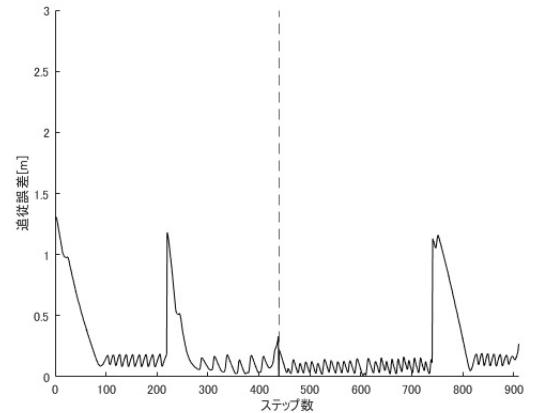


図 4.10: フォロワ 2 の追従誤差

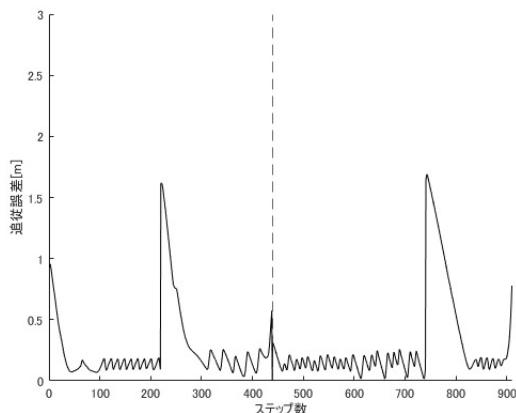


図 4.11: フォロワ 3 の追従誤差

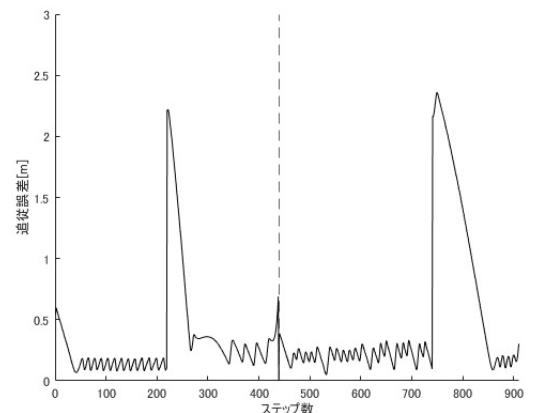


図 4.12: フォロワ 4 の追従誤差

4.3. 属性変更及び choke ポイント通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

ステップ数とリーダの速度ベクトルの各成分の関係を図 4.13 から図 4.15 に示す。

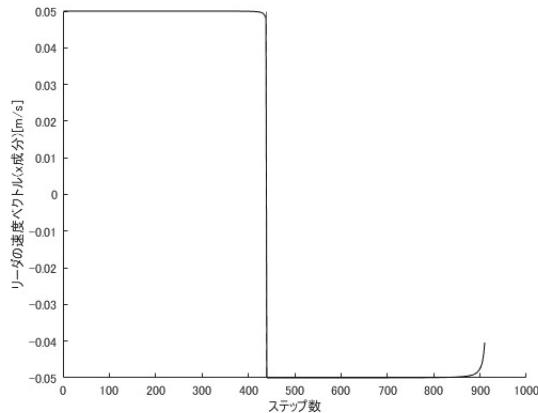


図 4.13: リーダの x 方向の速さ

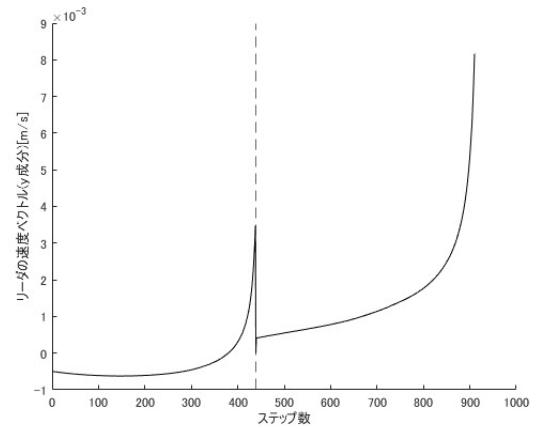


図 4.14: リーダの y 方向の速さ

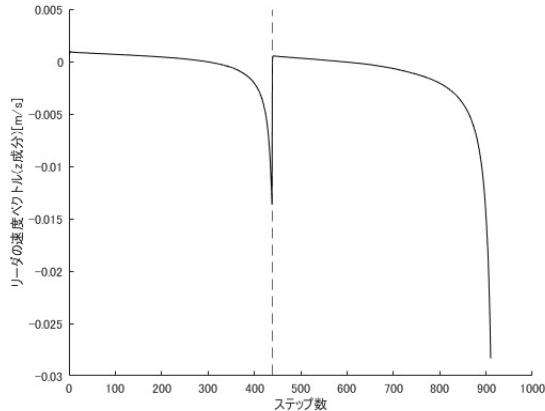


図 4.15: リーダの z 方向の速さ

図 4.9 から図 4.12 より、ステップ数が 0 から 100 付近の範囲で、ステップ数の増加に伴い、追従誤差が減少していることがわかる。ステップ数が 200 付近と 750 付近で追従誤差が増加しているのは、 choke ポイントを検知してフォーメーションを変更したためである。また、ステップ数が 450 付近で追従誤差が 0 となるのは、リーダを切り替えるタイミングで図 4.13 から図 4.15 に示すようにリーダの速度が 0 になるためである。リーダの速度が 0 になるのは 1 個目の目標点に到達し属性変更する際にクラッドロータを停留させているためである。

以上の結果より、属性を変更することにより目標点が群れの進行方向と逆の位置に設定された場合に並び替えが不要となること及びフォーメーションを変更することに

4.3. 属性変更及び choke point 通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

より choke point を通過できることが確認される。本シミュレーションでは目標点に到達した時点で属性変更アルゴリズムを適用したが、外乱等により飛行不能になった場合に適用したりすることで、より頑健な制御が可能になることが考えられる。

第5章 協調制御に基づくチョークポイント通過

4章で述べた制御器では属性変更アルゴリズムが適用されるため、どの機体がチョークポイントを検知するリーダになるのかが目標点に応じて異なる。そのため、全機体に障害物センサを搭載する必要がある。しかし、全機体に障害物センサを搭載すると費用やセンサから得た情報を処理するコストなどが問題点となる。そこで、協調制御に基づくチョークポイント通過アルゴリズムを用いて、障害物センサを搭載した少数の機体で他の機体を誘導し、チョークポイントを通過することを考える。

5.1 概要

本手法ではリーダのみが障害物センサを搭載しており、フォロワは搭載していない。図5.1に示すような初期位置から、リーダが所定のフォーメーションを形成すると同時にフォロワがフォーメーションの中に収束する。その結果、図5.2に示すように、リーダがフォロワを囲む形になる。障害物センサを搭載しているリーダがフォロワを囲むことにより、フォロワは障害物センサを搭載していなくてもチョークポイントに対応することができる。

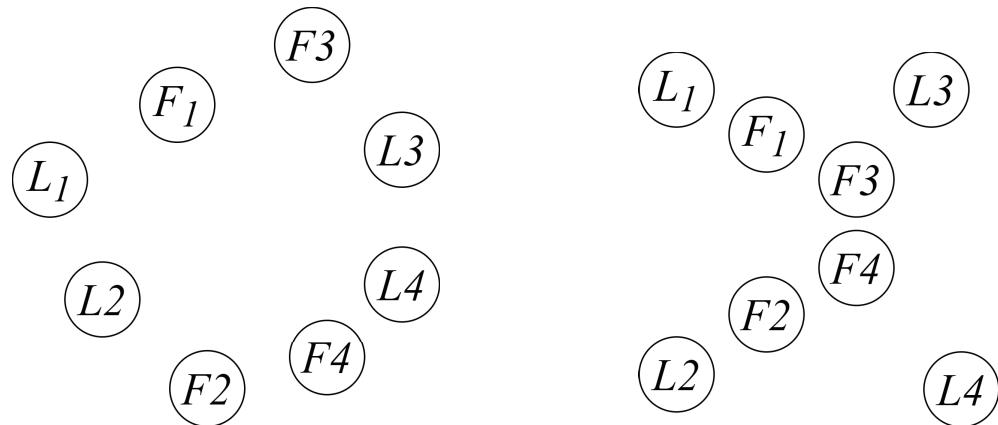


図 5.1: 初期位置の例

図 5.2: フォーメーション形成後

屋内での探索を想定しているため、チョークポイントを検知していない場合は広がりのあるフォーメーションを形成、維持しながら目標点に向かい進行する。しかし、広がりのあるフォーメーションのままではチョークポイントを通過することができないため群れの大きさを変更する必要がある。これをスケーリングと呼び、チョークポイントを検知した場合には通過するため、群れの大きさが小さくなるようにスケーリングする。チョークポイント通過後は広がって探索を行ったほうが効率が良いため、群れの大きさが元の大きさになるようにスケーリングを行う。群れ全体の動きは仮想リーダに追従する。仮想リーダとは実体がなく、外生的な制御入力を与えることができるリーダである。

5.2 制御入力

本手法では機体同士の影響を重み付き有向グラフで示しており、それを相互作用トポロジーと呼ぶ。図 5.3 に相互作用トポロジーを示す。図 5.3において V は仮想リーダ、 L はリーダ、 F はフォロワを示す。リーダに隣接するのは仮想リーダまたはリーダ、フォロワに隣接するのはリーダまたはフォロワであると仮定する。また、各フォロワに対して、少なくとも 1 機のリーダがそのフォロワへのパスを持つものとする。チョークポイントの通過判定については仮想リーダに隣接するリーダがチョークポイントを検知しなくなった後に、仮想リーダに隣接しないリーダも検知しなくなった場合に通過と判定している。

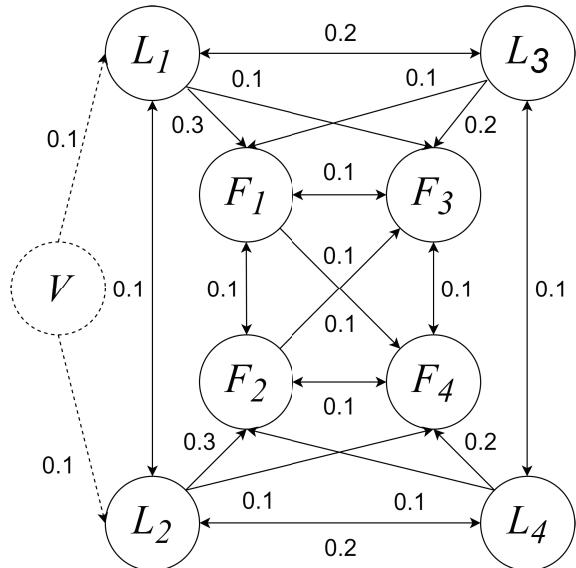


図 5.3: 相互作用トポロジー

リーダの制御入力はフォーメーションを形成, 維持する場合とスケーリングを行う場合で異なる式を用いる。フォーメーションを形成, 維持する場合の i 番目のリーダの制御入力は式 (5.1) で求められる。

$$\mathbf{u}_i = -\frac{1}{\gamma_i} \left(k_p \boldsymbol{\xi}_i - \sum_{j \in \mathcal{N}_i} a_{ij} \dot{\mathbf{p}}_j \right) \quad (\forall i \in E) \quad (5.1)$$

$$\gamma_i = \sum_{j \in \mathcal{N}_i} a_{ij} \quad (5.2)$$

$$\boldsymbol{\xi}_i = \sum_{j \in \mathcal{N}_i} a_{ij} (\mathbf{p}_i - \mathbf{p}_j - \boldsymbol{\delta}_{ij}) \quad (5.3)$$

$$k_p > 0, a_{ij} > 0 \quad (5.4)$$

E はリーダの集合, \mathcal{N}_i は i 番目のリーダに隣接するクワッドロータの集合, \mathbf{p}_j , $\dot{\mathbf{p}}_j$ はそれぞれ j 番目のクワッドロータの位置と速度である。例として図 5.3 の L_1 の制御入力を求める場合を考えると, \mathcal{N}_i の要素は仮想リーダ V , リーダ L_2, L_3 であり, それらの位置と速度を用いることになる。 k_p は制御ゲイン, a_{ij} は相互作用トポロジーにおいて j から i へのエッジの重みである。 $\boldsymbol{\delta}_{ij}$ はクワッドロータ間の位置関係を規定する変位制約であり, j からみた i の位置関係を示す。

スケーリングを行う場合の i 番目のリーダの制御入力は式 (5.3) の $\boldsymbol{\delta}$ が $\hat{\boldsymbol{\delta}}$ に変化した式 (5.5) で求められる。

$$\mathbf{u}_i = -\frac{1}{\gamma_i} \left(k_p \hat{\boldsymbol{\xi}}_i - \sum_{j \in \mathcal{N}_i} a_{ij} \dot{\mathbf{p}}_j \right) \quad (\forall i \in E) \quad (5.5)$$

$$\hat{\boldsymbol{\xi}}_i = \sum_{j \in \mathcal{N}_i} a_{ij} (\mathbf{p}_i - \mathbf{p}_j - \hat{\boldsymbol{\delta}}_{ij}) \quad (5.6)$$

$$\hat{\boldsymbol{\delta}}_{ij} = [s_x \delta_{ij}^x, s_y \delta_{ij}^y, s_z \delta_{ij}^z]^T \quad (5.7)$$

s_x, s_y, s_z はスケーリングベクトルであり, 変位制約と実際の機体間の変位の差に応じてスケーリングベクトルを調整することによりスケーリングを行う。

フォロワの制御入力はフォーメーションを形成, 維持する場合とスケーリングを行う場合で同じであり, k 番目のフォロワの制御入力は式 (5.8) で求められる。

$$\mathbf{u}_k = -\frac{1}{\gamma_k} \left(k_p \boldsymbol{\xi}_k - \sum_{j \in \mathcal{N}_k} a_{kj} \dot{\mathbf{p}}_j \right) \quad (\forall k \in F) \quad (5.8)$$

$$\gamma_k = \sum_{j \in \mathcal{N}_k} a_{kj} \quad (5.9)$$

5.3. 協調制御に基づくチョークポイント通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

$$\xi_k = \sum_{j \in \mathcal{N}_k} a_{kj} (\mathbf{p}_k - \mathbf{p}_j) \quad (5.10)$$

$$k_p > 0, a_{kj} > 0 \quad (5.11)$$

F はフォロワの集合, \mathcal{N}_k は k 番目のフォロワに隣接するクワッドロータの集合, \mathbf{p}_j , $\dot{\mathbf{p}}_j$ は j 番目のクワッドロータの位置と速度である. 例として図 5.3 の F_1 の制御入力を求める場合を考えると, \mathcal{N}_i の要素はリーダ L_1, L_3 , フォロワ F_2, F_3 であり, それらの位置と速度を用いることになる.

5.3 協調制御に基づくチョークポイント通過シミュレーション

本シミュレーションは 5.1 節で述べたアルゴリズムを用いることでチョークポイントを通過することができるかを検証する. ここでは目標点に向かい直進する仮想リーダにリーダとフォロワがフォーメーションを形成して追従する状況を想定する. 図 5.4 から図 5.9 に協調制御に基づくチョークポイント通過の動作を示す. 2 個の直方体の間がチョークポイントである. 目標点は黒の円筒でしてしており, その位置は $(9.5, 0, 2.5)$ である. 仮想リーダの制御入力は, 仮想リーダの現在位置から目標点への単位ベクトルを求め, 速さ $v_l = 0.5$ を掛けることで求めている. 相互作用トポロジーは図 5.3 に示したものを使用する. 図 5.4 から図 5.9 において, 青の 4 機がリーダ, 赤の 4 機がフォロワである. 表 5.1 と表 5.2 に各機体の初期位置を示す.

表 5.1: リーダの初期位置

	初期位置
リーダ 1	$(-5.1, 0.375, 2.5)$
リーダ 2	$(-5.5, 1.45, 2.5)$
リーダ 3	$(-7.4, 0.525, 2.5)$
リーダ 4	$(-7.5, 1.7, 2.5)$

表 5.2: フォロワの初期位置

	初期位置
フォロワ 1	$(-5.65, -0.4, 2.25)$
フォロワ 2	$(-6, 2, 2, 25)$
フォロワ 3	$(-6.5, -0.8, 2.25)$
フォロワ 4	$(-6.8, 2.2, 2.25)$

5.3. 協調制御に基づくチョークポイント通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

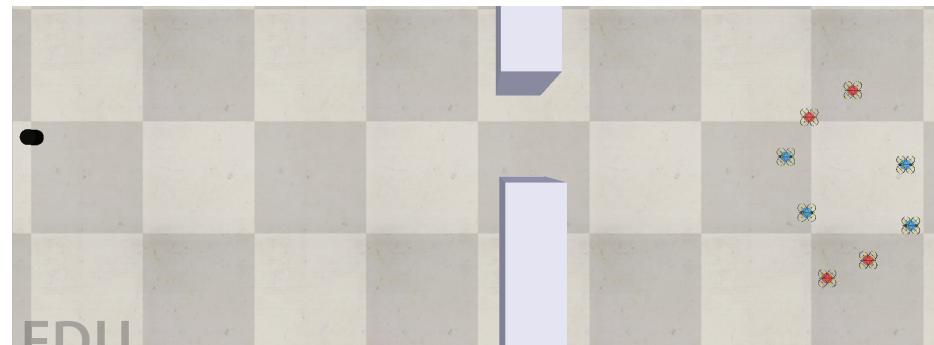


図 5.4: 初期位置

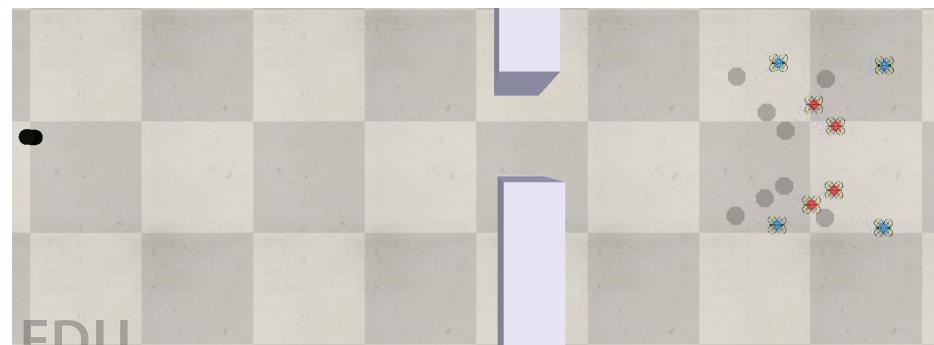


図 5.5: フォーメーション形成

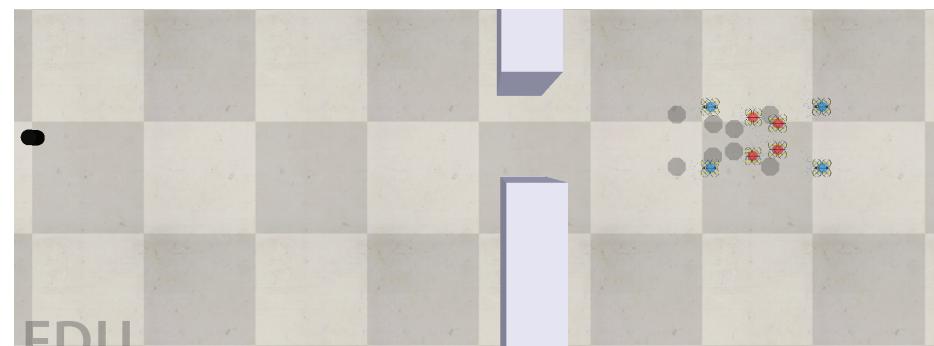


図 5.6: スケーリング後

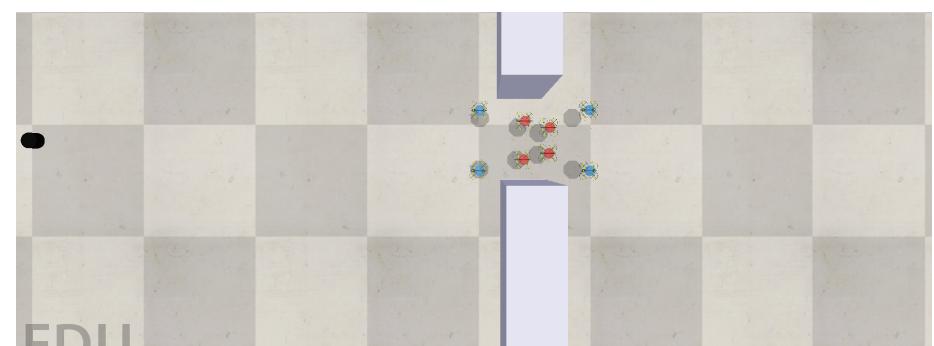


図 5.7: チョークポイント通過中

5.3. 協調制御に基づくチョークポイント通過シミュレーション

関西大学 システム理工学部 電気電子情報工学科

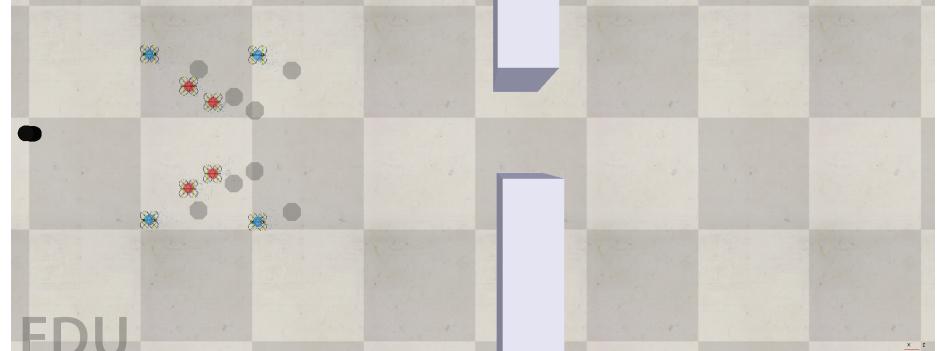


図 5.8: チョークポイント通過後

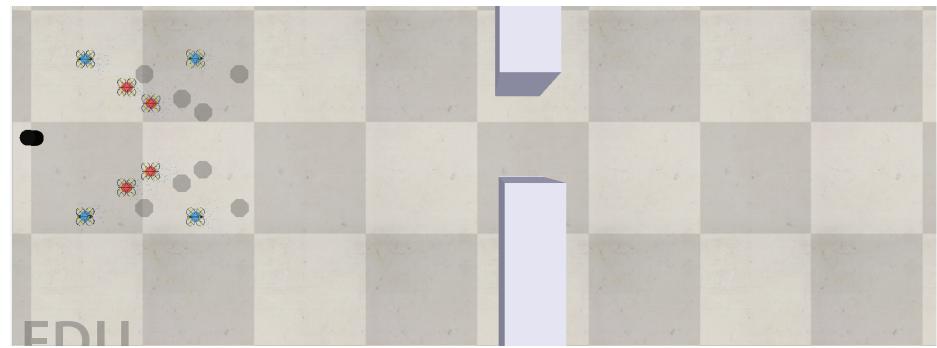


図 5.9: 目標点到達

図5.4は初期位置である。図5.5はフォーメーションを形成した様子である。青のリーダが赤のフォロワを囲む形になっている。図5.6はチョークポイントを検知した場面である。チョークポイントを通過するために、群れの大きさが小さくなるようにスケーリングした。ここで、チョークポイントの検知は先頭の2機のリーダが行っている。図5.7はスケーリングを行った後に、そのフォーメーションを維持しながらチョークポイントを通過している。図5.8はチョークポイントを通過したため、再度スケーリングを行い群れの大きさを検知前に戻している。図5.9は群れの大きさを元の大きさに戻した後に、そのフォーメーションを維持して目標点に進行し、到達した。以上の結果よりチョークポイントを検知した場合にスケーリングを行うことによりチョークポイントを通過できることが確認できる。

第6章 協調制御における属性変更

5章で述べた手法では仮想リーダの作用を受けるリーダが固定である。そのため、5.3節でのシミュレーション環境において、1個目の目標点に到達した後に2個目の目標点が下に設定された場合などには群れ全体を回転させる必要がある。この回転を不要とするために協調制御における属性変更アルゴリズムを提案する。

6.1 協調制御における属性変更アルゴリズムの提案

4.1節で述べたアルゴリズムと同様に、本アルゴリズムでも各機体に番号を割り当てる。リーダとフォロワが合わせて N 機、フォロワが M 機である状況を考える。この場合、フォロワには 1 から M まで、リーダは $M + 1$ から N まで、仮想リーダは $N + 1$ の数字が割り当てる。この番号を属性とし、番号の割り当てを変えることにより属性を変更している。仮想リーダは実体がないため、仮想リーダを表す番号が他の機体に割り当てられることはない。属性は各機体と目標点の距離に応じて決定される。目標点との距離が近い 2 機が仮想リーダに隣接する機体とし、他の機体については予め決定された相互作用トポロジーに応じて番号が割り当てられる。仮想リーダは隣接する 2 機が決定された時点で、その 2 機との変位制約を保てる位置に移動する。図 6.1 と図 6.2 を用いて例を示す。両図において目標点を黒のひし形で示す。

6.1. 協調制御における属性変更アルゴリズムの提案

関西大学 システム理工学部 電気電子情報工学科

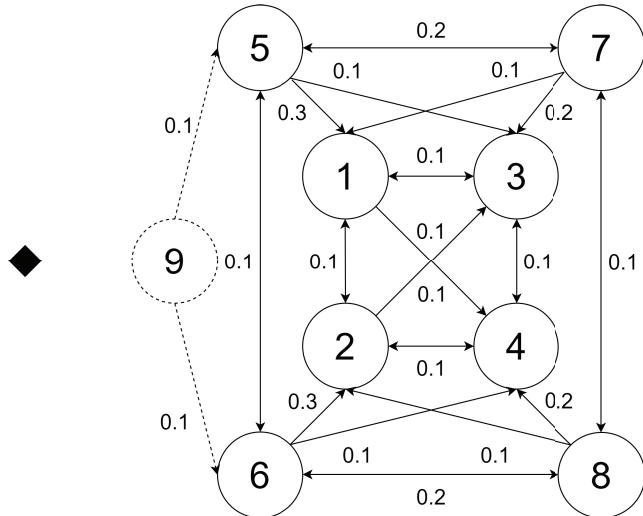


図 6.1: 属性変更前

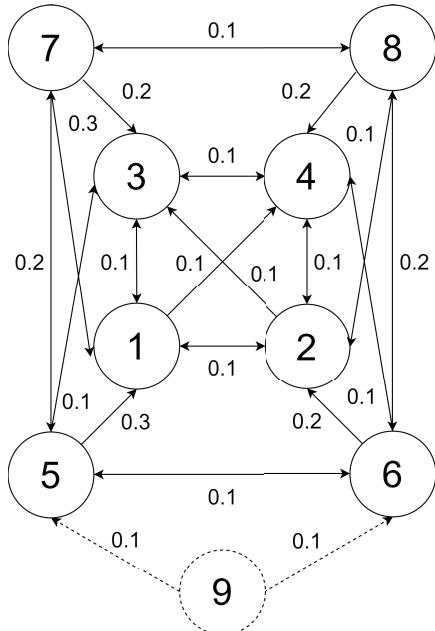


図 6.2: 属性変更後

図 6.1 に、図 5.3 を属性を用いて表したものと示す。リーダとフォロワを合計して 8 機、フォロワが 4 機であるため、図 6.1 ではフォロワに 1 から 4 を、リーダに 5 から 8

6.2. 協調制御における属性変更シミュレーション

関西大学 システム理工学部 電気電子情報工学科

を、仮想リーダに9を割り当てている。図6.1では目標点が左に設定されているが、図6.2では目標点が下に設定されている。そのため、各機体と目標点の距離を考え、図6.2では属性変更前は6, 8が与えられていた機体に5, 6が与えられている。属性変更後は変位制約を満たすようにフォーメーションを形成する。

6.2 協調制御における属性変更シミュレーション

本シミュレーションでは協調制御における属性変更アルゴリズムを用いることで群れ全体を回転させることなく進行方向を変更できるかを検証する。ここでは目標点に向かい直進する仮想リーダにリーダとフォロワがフォーメーションを形成して追従する状況を想定する。図6.3から図6.8に提案したアルゴリズムを適用した場合の動作を示す。目標点は黒の円筒で示しており、目標点と仮想リーダの距離が一定以下になった場合に目標点に到達したと判定し、次の目標点を設定する。1個目の目標点の位置は(9.5, 0, 2.5)で、2個目の目標点の位置は(8, 7.2, 2.5)である。仮想リーダの制御入力は、仮想リーダの現在位置から目標点の単位ベクトルを求め、速さ $v_l = 0.5$ を掛けることで求めている。図6.3から図6.8において、青の4機がリーダ、赤の4機がフォロワである。初期位置は表5.1、表5.2と同様である。

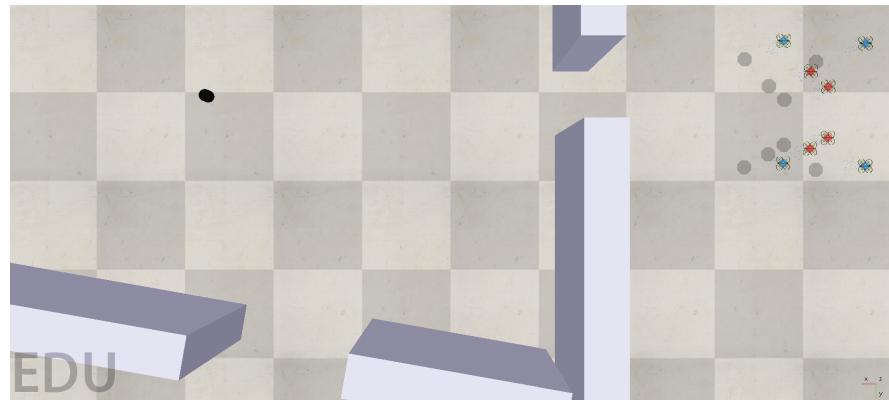


図 6.3: フォーメーション形成

6.2. 協調制御における属性変更シミュレーション

関西大学 システム理工学部 電気電子情報工学科

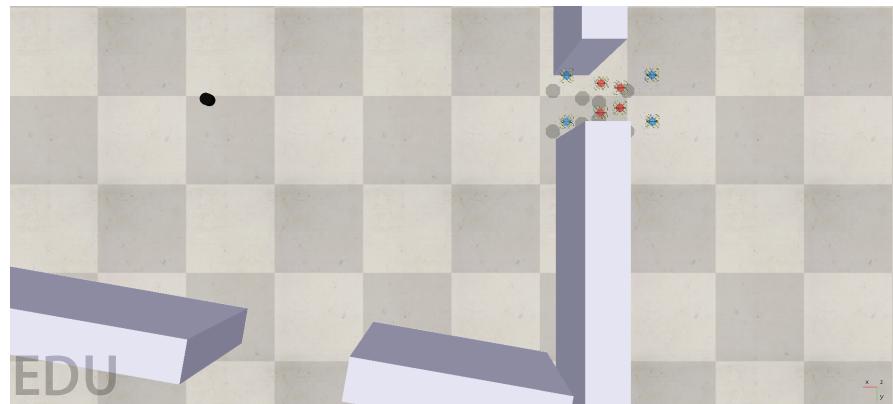


図 6.4: 1 個目のチョークポイント

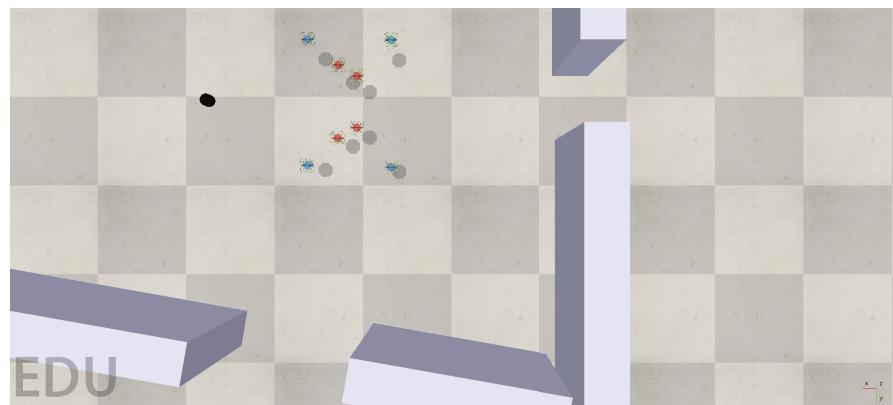


図 6.5: 広がりのあるフォーメーション

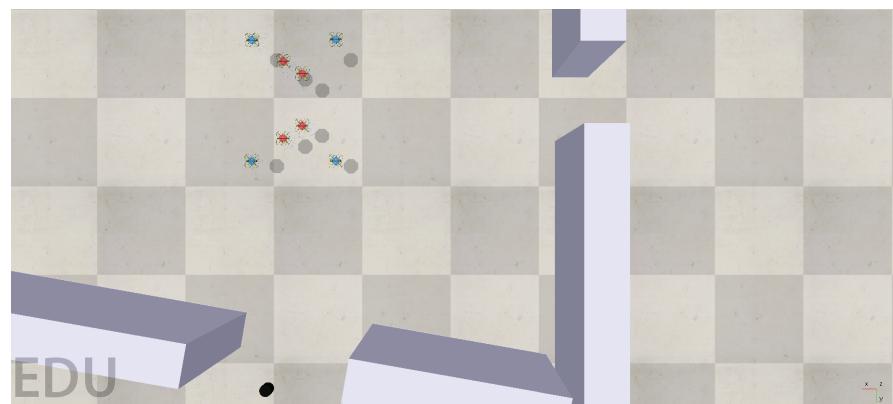


図 6.6: 属性変更アルゴリズム適用

6.2. 協調制御における属性変更シミュレーション

関西大学 システム理工学部 電気電子情報工学科

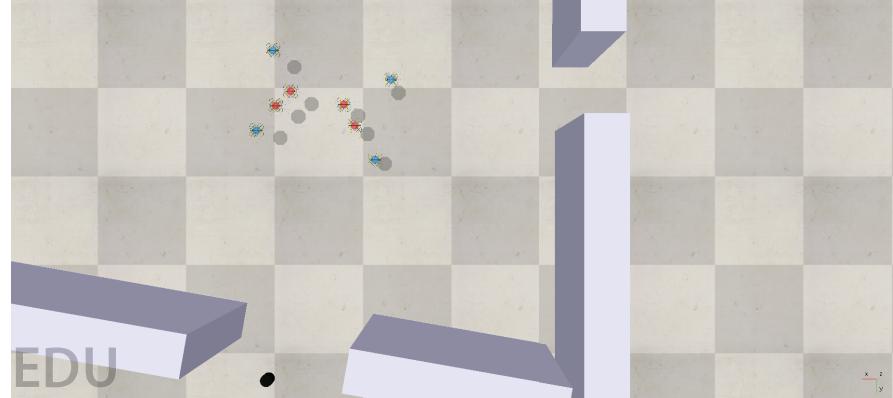


図 6.7: フォーメーション維持

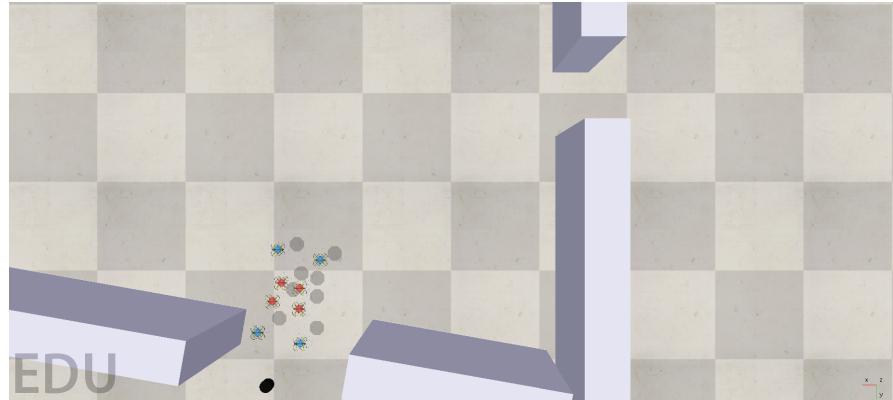


図 6.8: 2 個目のチョークポイント

図 6.3 は初期位置からフォーメーションを形成した状態である。図 6.4 はフォーメーションを形成した後に、維持して進行していく中でチョークポイントを検知したときの様子である。チョークポイントを通過するため、群れを大きさが小さくなるようにスケーリングした。そして、スケーリング後のフォーメーションを維持しながら進行し、チョークポイントを通過している。図 6.5 はチョークポイントを通過したため群れの大きさをもとに戻している。図 6.6 は 1 個目の目標点に到達したため属性変更アルゴリズムを適用した様子である。目標点に近い、下のリーダ 2 機が仮想リーダに隣接し、進行していく。図 6.7 は目標点に近いリーダ 2 機リーダに隣接したため、図 6.3 でのフォーメーションを形成するために広がっている。図 6.8 は 2 個目のチョークポイントを検知したため、通過できるようにスケーリングをしている。以上の結果より協調制御における属性変更アルゴリズムを適用することにより、群れ全体を回転させることなく進行方向を変更することができた。

第7章 結論

本論文では、まずフォーメーション制御の概要について、主に研究されている三つの手法の特徴及び違いを踏まえて説明した。その中でもリーダ・フォロワ制御を用いることで管制塔を使用せずに制御し、通信コストの節約を期待できることを述べた。

第3章では、LOSに基づくリーダ・フォロワ制御の概要及び先行研究で実装されたクワッドロータ間の回避アルゴリズムについて述べ、追従制御のシミュレーションを行った。指定された目標点に向かい直進するリーダを参照し、4機のフォロワが追従する状況で、ステップ数の増加に伴いフォロワの追従誤差が減少し、フォーメーションを形成することができた。第4章ではLOS制御に基づいてリーダ・フォロワ制御を実装した上で、群れの進行方向と逆の位置に目標点が設定された場合に機体の並び替えを不要とするための属性変更アルゴリズムを提案した。また、チョークポイント通過のためのフォーメーション変更アルゴリズムも提案した。それらのアルゴリズムの概要を述べ、シミュレーションを行った。指定された目標点に向かい直進するリーダ1機を参照し、4機のフォロワが追従する状況を想定し、チョークポイントを検知して場合は一直線のフォーメーションを形成し、チョークポイントを検知していない場合は広がりのあるフォーメーションを形成することでチョークポイントを通過できることを確認した。また、群れの進行方向とは逆の位置に目標点が設定された場合に属性変更アルゴリズムを適用することにより、機体を並び替えることなく、群れの進行方向を変えられることを確認した。本シミュレーションでは目標点が変化した場合にのみ属性変更アルゴリズムを適用したが、リーダが外乱等により飛行不能になるような場合において適用することで、より頑健なフォーメーション制御が可能となると考える。

第5章では協調制御に基づくチョークポイント通過の概要を述べ、シミュレーションを行った。4機のリーダと4機のフォロワが動作することを想定し、障害物センサを搭載したリーダがフォロワを囲むことにより、フォロワが障害物センサを搭載していないてもチョークポイントに対応できることを確認した。第6章では、第5章の手法において群れの進行方向を変える際の群れ全体の回転を不要とするため、協調制御における属性変更アルゴリズムの概要を述べ、シミュレーションを行った。1個目のチョークポイントを通過したのちに、2個目のチョークポイントに向かい進行する際に群れを回転させることなく進行方向を変えられることを確認した。

協調制御に基づくチョークポイント通過について、本論文ではリーダ4機、フォロ

ワ4機でシミュレーションを行い、より多くのフォロワが存在する場合は検討していない。フォロワの数が増えると、フォーメーション形成の際にクワッドロータ同士が衝突する可能性が高まると考えられる。ゆえに、クワッドロータ同士の回避については今後の課題である。

謝辞

本研究を進めるにあたり、関西大学システム理工学部、三好 誠司教授並びに本仲 君子准教授には大変お世話になりましたことを深くお礼申し上げます。

三好先生には研究発表に関して多くご指導をいただきました。本仲先生にはチョークポイント通過の研究に対して様々な相談に乗って頂きました。

研究室の友人並びに諸先輩方には、研究に関する助力や助言をいただき、感謝しております。

最後になりましたが、私の大学生活を金銭面、精神面の両面で支えていただき、多大なるご助力・ご支援をいただいた両親にはこの場をお借りしまして深く心よりの感謝を申し上げます。

参考文献

- [1] 桜間 一徳, “マルチエージェントシステムの制御”, システム制御情報, Vol. 57, No. 9, pp. 386-396, 2013.
- [2] 宮崎 達也, 鷹羽 浩嗣, “障害物回避を考慮した移動ロボット群のフォーメーション制御”, システム制御情報学会誌. Vol. 28, No. 2, pp.50-57, 2015.
- [3] 安田 博, 久保田 直行, “動的環境に適応するマルチロボットの動的フォーメーション”, 知能と情報(日本知能情報ファジィ学会誌) Vol24, No.1, pp.571-581 (2012).
- [4] A. S. Simonsen and E. L. M. Ruud, “The Application of a Flexible Leader-Follower Control Algorithm to Different Mobile Autonomous Robots”, Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11561-11566, 2020.
- [5] T. I. Fossen, M. Breivik, and R. Skjetne, “Line-of-sight path following of underactuated marine craft”, IFAC Proceedings Volumes, Vol. 36, No. 21, pp. 211–216, 2003.
- [6] A. Pavlov, E. Borhaug, E. Panteley, and K. Y. Pettersen, “Straight Line Path Following for Formations of Underactuated Surface Vessels”, IFAC Proceedings Volumes, Vol. 40, No. 12, pp. 521-526, 2007.
- [7] M. Breivik, V. E. Hovstein, and T. I. Fossen, “Path following for marine surface vessels. ”, Ocean '04-MTS/IEEE Techno-Ocean '04 : Bridges across the Oceans-Conference Proceedings, Vol. 4, pp. 2282–2289, 2004.
- [8] Yu-Hsiang Su and, and A. Lanzon, “Formation-containment tracking and scaling for multiple quadcopters with an application to choke-point navigation”, proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp.4908-4914, 2022.
- [9] “Remote API functions(Matlab)”, <https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsMatlab.htm>(2023.2.10)

付 錄A LOS制御におけるチョークポイント通過のプログラマリスト

ソースコード A.1: LOS 制御におけるチョークポイント通過のメインプログラム

```
1 %{
2     本プログラムでは距離等の単位を [cm]で考える
3 %}
4 %% メモリのクリア
5 clear;
6 close all
7 clc
8
9 %% coppeliasimとの連携
10 sim=remApi('remoteApi'); % using the prototype file (remoteApiProto.m
    )
11 sim.simxFinish(-1); % just in case, close all opened connections
12 clientID=sim.simxStart('127.0.0.1',19997,true,true,5000,5);
13 sim.simxStartSimulation(clientID,sim.simx_opmode_blocking);
14 pause(0.1)
15
16 %{
17     clientID1: リーダ,clientID2 以降: フォロワ 1・2・3・4
18     フォロワを増やす際にはコピペ
19     19999の場所に当たる数字のみcoppeliasimと一致させる
20 %}
21 clientID1 = sim.simxStart('127.0.0.1',19999,true,true,5000,5);
22 clientID2 = sim.simxStart('127.0.0.1',19995,true,true,5000,5);
23 clientID3 = sim.simxStart('127.0.0.1',19993,true,true,5000,5);
24 clientID4 = sim.simxStart('127.0.0.1',19991,true,true,5000,5);
25 clientID5 = sim.simxStart('127.0.0.1',19989,true,true,5000,5);
26 %リーダの目標点(円筒形)を表示する
27 clientID6 = sim.simxStart('127.0.0.1',19987,true,true,5000,5);
```

```

28
29 %% coppeliasimとの連携
30 if (clientID>-1)
31     disp('Connected to remote API server');
32
33 %{
34     handle setting
35     Quadcopter_target は coppeliasim のドローンの球
36 %}
37 [r, Quad(1)] = sim.simxGetObjectHandle(clientID1, 'Quadcopter',
38                             sim.simx_opmode_blocking);
38 [r, target(1)] = sim.simxGetObjectHandle(clientID1, ,
39                             'Quadcopter_target', sim.simx_opmode_blocking);
40
41 [r, Quad(2)] = sim.simxGetObjectHandle(clientID2, 'Quadcopter#1',
42                             sim.simx_opmode_blocking);
41 [r, target(2)] = sim.simxGetObjectHandle(clientID2, ,
42                             'Quadcopter_target#1', sim.simx_opmode_blocking);
43
44 [r, Quad(3)] = sim.simxGetObjectHandle(clientID3, 'Quadcopter#3',
45                             sim.simx_opmode_blocking);
44 [r, target(3)] = sim.simxGetObjectHandle(clientID3, ,
45                             'Quadcopter_target#3', sim.simx_opmode_blocking);
46
47 [r, Quad(4)] = sim.simxGetObjectHandle(clientID4, 'Quadcopter#5',
48                             sim.simx_opmode_blocking);
47 [r, target(4)] = sim.simxGetObjectHandle(clientID4, ,
48                             'Quadcopter_target#5', sim.simx_opmode_blocking);
49
50 [r, Quad(5)] = sim.simxGetObjectHandle(clientID5, 'Quadcopter#7',
51                             sim.simx_opmode_blocking);
50 [r, target(5)] = sim.simxGetObjectHandle(clientID5, ,
51                             'Quadcopter_target#7', sim.simx_opmode_blocking);
52
52 [r, Cylinder] = sim.simxGetObjectHandle(clientID6, 'Cylinder',
53                             sim.simx_opmode_blocking);
53
54 % Get position and orientation data Setting
55 [returnCode, QuadPos1] = sim.simxGetObjectPosition(clientID1, Quad
56     (1), -1, sim.simx_opmode_streaming);
56 [returnCode, QuadAng1] = sim.simxGetObjectOrientation(clientID1,
57     Quad(1), -1, sim.simx_opmode_streaming);
57 pause(0.1);

```

```
58 % Get URG sensor data
59 [errorCode, dist] = sim.simxGetStringSignal(clientID1, 'scan
    ranges11', sim.simx_opmode_streaming);
60 [errorCode, dist] = sim.simxGetStringSignal(clientID1, 'scan
    ranges12', sim.simx_opmode_streaming);
61 pause(0.1);
62
63 [returnCode, QuadPos2] = sim.simxGetObjectPosition(clientID2, Quad
    (2), -1, sim.simx_opmode_streaming);
64 [returnCode, QuadAng2] = sim.simxGetObjectOrientation(clientID2,
    Quad(2), -1, sim.simx_opmode_streaming);
65 pause(0.1);
66 [errorCode, dist] = sim.simxGetStringSignal(clientID2, 'scan
    ranges21', sim.simx_opmode_streaming);
67 [errorCode, dist] = sim.simxGetStringSignal(clientID2, 'scan
    ranges22', sim.simx_opmode_streaming);
68 pause(0.1);
69
70 [returnCode, QuadPos3] = sim.simxGetObjectPosition(clientID3, Quad
    (3), -1, sim.simx_opmode_streaming);
71 [returnCode, QuadAng3] = sim.simxGetObjectOrientation(clientID3,
    Quad(3), -1, sim.simx_opmode_streaming);
72 pause(0.1);
73 [errorCode, dist] = sim.simxGetStringSignal(clientID3, 'scan
    ranges31', sim.simx_opmode_streaming);
74 [errorCode, dist] = sim.simxGetStringSignal(clientID3, 'scan
    ranges32', sim.simx_opmode_streaming);
75 pause(0.1);
76
77 [returnCode, QuadPos4] = sim.simxGetObjectPosition(clientID4, Quad
    (4), -1, sim.simx_opmode_streaming);
78 [returnCode, QuadAng4] = sim.simxGetObjectOrientation(clientID4,
    Quad(4), -1, sim.simx_opmode_streaming);
79 pause(0.1);
80 [errorCode, dist] = sim.simxGetStringSignal(clientID4, 'scan
    ranges41', sim.simx_opmode_streaming);
81 [errorCode, dist] = sim.simxGetStringSignal(clientID4, 'scan
    ranges42', sim.simx_opmode_streaming);
82 pause(0.1);
83
84 [returnCode, QuadPos5] = sim.simxGetObjectPosition(clientID5, Quad
    (5), -1, sim.simx_opmode_streaming);
```

```
85 [returnCode, QuadAng5] = sim.simxGetObjectOrientation(clientID5,
86     Quad(5), -1, sim.simx_opmode_streaming);
87 pause(0.1);
88 [errorCode, dist] = sim.simxGetStringSignal(clientID5, 'scan
89     ranges51', sim.simx_opmode_streaming);
90 [errorCode, dist] = sim.simxGetStringSignal(clientID5, 'scan
91     ranges52', sim.simx_opmode_streaming);
92 pause(0.1);
93
94 %% フォーメーション制御(以下の部分が処理のメイン)
95 %% 変数の規定(シミュレーション状況により変更する)
96
97 %% ステップ数(シミュレーション時間)
98 loop_num = 20000;
99
100 %% リーダ機の機体番号を格納(値は簡易的に1としている)
101 Lnumber = 1;
102
103 %% リーダを含めたクワッドローターの数
104 Quad_num = 5;
105
106 %% 距離の閾値(侵入禁止領域の設定)
107 thresho = 80;
108
109 %% k01はフォロワの目標速度(制御入力)の方向を計算する際のゲイン(k0,k1)
110 k01 = zeros(Quad_num-1,2);
111 k01(1,:) = [5,200]; k01(2,:) = [5,200];
112 k01(3,:) = [5,200]; k01(4,:) = [5,200];
113
114 %% kpsはフォロワの目標速度(制御入力)の大きさを計算する際のゲイン(kp,
115 ks)
116 kps = zeros(Quad_num-1,2);
117 kps(1,:) = [1,1]; kps(2,:) = [1,1];
118 kps(3,:) = [1,1]; kps(4,:) = [1,1];
119
120 %% フォーメーションの数
121 formation_num = 2;
122
123 %% フォーメーションを指定
```

```

124     current_formation = 2;
125
126     %リーダと各フォロワの距離を規定
127     d = zeros(Quad_num-1,formation_num);
128     d(:,1) = [320,240,160,80];
129     d(:,2) = [200,200,100,100];
130
131     %リーダの目標点数
132     LG_num = 2;
133
134     %リーダの速さ設定
135     L_speed = 5;
136
137     %リーダの目標到達地点
138     lg = zeros(3,LG_num);
139
140     %lg(:,1) = [500,-15,250];
141     lg(:,1) = [1100,-15,250];
142     lg(:,2) = [-650,-15,300];
143
144 %% クワッドローターの構造体の定義と定数の規定
145
146 %
147     この構造体では機体番号(属性番号で無い)に紐づく情報を保持している
148     機体番号はインデックスが対応する
149     各クワッドローターの情報を格納する構造体の定義
150     Q:構造体名
151     Att:属性番号(機体番号に対応する属性番号を格納)
152         リーダ: 1
153         2以降: フォロワ
154             5~2がそれぞれフォロワ 1~4に対応
155     Coord:座標
156         第1インデックス:x,y,z
157         第2インデックス:ステップ
158         第3インデックス:機体番号
159     l_distance:自身から見たリーダのベクトル
160         インデックスは座標と同じ
161     unit_l_distance:l_distance の単位ベクトルを格納
162         インデックスは座標と同じ
163     speed:速さ,インデックスは座標と同じ
164     speed_dir:速度の方向,インデックスは座標と同じ
165     Cin:制御入力の大きさ

```

```

166      第1インデックス:ステップ
167      第2インデックス:機体番号
168      Cin_dir:Cin の単位ベクトル
169          第1インデックス:x,y,z
170          第2インデックス:ステップ
171          第3インデックス:機体番号
172      %}
173      Q.Att = zeros(Quad_num,1);
174      Q.Coord = zeros(3,loop_num+1,Quad_num);
175      Q.l_distance = zeros(3,loop_num+1,Quad_num);
176      Q.unit_l_distance = zeros(3,loop_num+1,Quad_num);
177      Q.speed = zeros(3,loop_num+1,Quad_num);
178      Q.speed_dir = zeros(3,loop_num+1,Quad_num);
179      Q.Cin = zeros(loop_num+1,Quad_num);
180      Q.Cin_dir = zeros(3,loop_num+1,Quad_num);
181
182      %}
183      フォロワ間のベクトルを格納する配列
184      フォロワ番号(属性番号の2以降に紐づく情報)
185      VBF(vector between followers):フォロワ間のベクトル
186      VBF_dir:VBF の単位ベクトル
187      %}
188      VBF = zeros(3,loop_num,Quad_num-1,Quad_num-1);
189      VBF_dir = zeros(3,loop_num,Quad_num-1,Quad_num-1);
190
191      %}
192      目標点に到達した時間を格納する配列
193      arrive_time に格納された時間を用いて目標点を表示する
194      %}
195      arrive_time = zeros(LG_num,1);
196
197      %% 初期設定
198
199      %属性番号の初期化
200      Q.Att = 1:5;
201
202      %}
203          AOR(Angle of rotation):フォロワの追従位置を決定する角度
204          一列目が $\Psi$ ,二列目が $\Phi$ 
205           $\Psi$ がローカルでのz軸中心の回転  $\theta$ がローカルでのx軸周りの回転
206      %}
207      AOR = zeros(Quad_num-1,2,formation_num);
208      AOR(1,:,:1) = [deg2rad(180),deg2rad(0)]; %フォロワ1の回転角 $\Psi$ と $\Phi$ 

```

```
209 AOR(2,:,1) = [deg2rad(180),deg2rad(0)]; %フォロワ2の回転角PsiとPhi  
210 AOR(3,:,1) = [deg2rad(180),deg2rad(0)]; %フォロワ3の回転角PsiとPhi  
211 AOR(4,:,1) = [deg2rad(180),deg2rad(0)]; %フォロワ4の回転角PsiとPhi  
212  
213  
214 AOR(1,:,2) = [deg2rad(-140), deg2rad(0)]; %フォロワ1の回転角Psiと  
215 %Phi  
216 AOR(2,:,2) = [deg2rad(140), deg2rad(0)]; %フォロワ2の回転角Psiと  
217 %Phi  
218 AOR(3,:,2) = [deg2rad(140), deg2rad(0)]; %フォロワ3の回転角Psiと  
219 %Phi  
220 AOR(4,:,2) = [deg2rad(-140), deg2rad(0)]; %フォロワ4の回転角Psiと  
221 %Phi  
222  
223 %リーダの初期座標を設定  
224 Q.Coord(1,1,1)=-400; Q.Coord(2,1,1)=0; Q.Coord(3,1,1)=220;  
225  
226 %フォロワ1,2,3,4の初期座標を設定  
227 Q.Coord(1,1,2)=-420; Q.Coord(2,1,2)=-110; Q.Coord(3,1,2)=250;  
228 Q.Coord(1,1,3)=-500; Q.Coord(2,1,3)=-60; Q.Coord(3,1,3)=250;  
229 Q.Coord(1,1,4)=-520; Q.Coord(2,1,4)= 45; Q.Coord(3,1,4)=250;  
230 Q.Coord(1,1,5)=-600; Q.Coord(2,1,5)=-110; Q.Coord(3,1,5)=250;  
231  
232 %リーダの目標地点の変更回数  
233 Change_num = 1;  
234  
235 %全目標地点に到達した時点で停滞させるための座標  
236 Comp_Coord = zeros(3,Quad_num);  
237  
238 %上記座標を取得するためのフラグ  
239 Comp_flag = 0;  
240  
241 %例外に到達したことを示すフラグ  
242 exception_flag = 0;  
243  
244 %例外時に停滞させるための座標  
245 exception_Coord = zeros(3,Quad_num);  
246  
247 %障害物センサ配列の要素数  
248 stepnum = 684;  
249  
250 %障害物があるかの判定  
251 obstacle_flag = 0;
```

```
248
249 %グラフ出力用の配列
250 l_Error = zeros(3,loop_num,4);
251
252 %ワールド座標系でみたローカル軸
253 new_aixs = zeros(3,3);
254
255 %クワッドロータの初期座標をコペラシムに反映している
256 sim.simxSetObjectPosition(clientID1, Quad(1), -1, [Q.Coord
    (1,1,1)/100, Q.Coord(2,1,1)/100,Q.Coord(3,1,1)/100], sim.
    simx_opmode_oneshot);
257 sim.simxSetObjectPosition(clientID2, Quad(2), -1, [Q.Coord
    (1,1,2)/100, Q.Coord(2,1,2)/100,Q.Coord(3,1,2)/100], sim.
    simx_opmode_oneshot);
258 sim.simxSetObjectPosition(clientID3, Quad(3), -1, [Q.Coord
    (1,1,3)/100, Q.Coord(2,1,3)/100,Q.Coord(3,1,3)/100], sim.
    simx_opmode_oneshot);
259 sim.simxSetObjectPosition(clientID4, Quad(4), -1, [Q.Coord
    (1,1,4)/100, Q.Coord(2,1,4)/100,Q.Coord(3,1,4)/100], sim.
    simx_opmode_oneshot);
260 sim.simxSetObjectPosition(clientID5, Quad(5), -1, [Q.Coord
    (1,1,5)/100, Q.Coord(2,1,5)/100,Q.Coord(3,1,5)/100], sim.
    simx_opmode_oneshot);
261 sim.simxSetObjectPosition(clientID6, Cylinder, -1, [lg(1,1)/100,
    lg(2,1)/100,lg(3,1)/100], sim.simx_opmode_oneshot);
262 sim.simxSynchronousTrigger(clientID);
263
264 %ステップ分ループを回す
265 for loop=1:loop_num
266
267 %% 実座標の取得
268 [returnCode, QuadPos1] = sim.simxGetObjectPosition(clientID1,
    Quad(1), -1, sim.simx_opmode_buffer);
269 [returnCode, QuadPos2] = sim.simxGetObjectPosition(clientID2,
    Quad(2), -1, sim.simx_opmode_buffer);
270 [returnCode, QuadPos3] = sim.simxGetObjectPosition(clientID3,
    Quad(3), -1, sim.simx_opmode_buffer);
271 [returnCode, QuadPos4] = sim.simxGetObjectPosition(clientID4,
    Quad(4), -1, sim.simx_opmode_buffer);
272 [returnCode, QuadPos5] = sim.simxGetObjectPosition(clientID5,
    Quad(5), -1, sim.simx_opmode_buffer);
273
274 Q.Coord(:,loop,1) = QuadPos1(:)*100;
```

```

275     Q.Coord(:,loop,2) = QuadPos2(:)*100;
276     Q.Coord(:,loop,3) = QuadPos3(:)*100;
277     Q.Coord(:,loop,4) = QuadPos4(:)*100;
278     Q.Coord(:,loop,5) = QuadPos5(:)*100;
279
280 %% 初期状態でのリーダとフォロワの决定
281 if loop == 1
282     Q.Att(:) = ChangeLeader(Q,loop,Quad_num,lg,Change_num);
283 end
284
285 %% リーダ属性のクワッドロータの機体番号を取得
286 Lnumber = find(Q.Att == 1);
287
288 %% 障害物センサの処理
289 %{
290     • 基本的にはリーダの障害物センサの値のみを利用する
291     • フォームーションが 1 でリーダが障害物を検知しなければ最後尾
292         の
293         クワッドロータの障害物センサの値を利用する
294         • センサの最大値は 4 であり, 4 のデータは障害物が存在しない
295         • フォロワと目標点も障害物センサに反応するため除外する
296 %}
297 %% 通常の障害物検知
298
299 %% リーダ機の障害物センサの値を取得する
300 [errorCode, tmp_ranges1] = sim.simxGetStringSignal(Lnumber,
301 ['scan ranges' num2str(Lnumber) '1'], sim.
302 simx_opmode_buffer);
303 [errorCode, tmp_ranges2] = sim.simxGetStringSignal(Lnumber,
304 ['scan ranges' num2str(Lnumber) '2'], sim.
305 simx_opmode_buffer);
306 % 数値データに変換
307 ranges1 = sim.simxUnpackFloats(tmp_ranges1);
308 ranges2 = sim.simxUnpackFloats(tmp_ranges2);
309 % リーダ機の姿勢の取得
310 [returnCode, QuadAng] = sim.simxGetObjectOrientation(find(Q.
311     Att == 1), Quad(Q.Att==1), -1, sim.simx_opmode_buffer);
312 pause(0.05);
313 % 障害物があるかの判定
314 obstacle_flag = obs_idf(ranges1,ranges2,QuadAng,stepnum,Q,loop
315 ,Lnumber,Quad_num,obstacle_flag,lg(1:2,Change_num));
316
317 %% チョークポイントを抜けたかの判定

```

```

311
312     if obstacle_flag == 0 && current_formation == 1
313         %最後尾のクワッドロータの障害物センサの値を取得
314         [errorCode, tmp_ranges1] = sim.simxGetStringSignal(find(Q.
315             Att == 2), ['scan ranges' num2str(find(Q.Att == 2))
316             '1'], sim.simx_opmode_buffer);
317         [errorCode, tmp_ranges2] = sim.simxGetStringSignal(find(Q.
318             Att == 2), ['scan ranges' num2str(find(Q.Att == 2))
319             '2'], sim.simx_opmode_buffer);
320         %数値データに変換
321         ranges1 = sim.simxUnpackFloats(tmp_ranges1);
322         ranges2 = sim.simxUnpackFloats(tmp_ranges2);
323         %最後尾のクワッドロータの姿勢を取得
324         [returnCode, QuadAng] = sim.simxGetObjectOrientation(find(
325             Q.Att == 2), Quad(Q.Att==2), -1, sim.
326             simx_opmode_buffer);
327         pause(0.05);
328         %障害物があるかの判定
329         obstacle_flag = obs_idf(ranges1,ranges2,QuadAng,stepnum,Q,
330             loop,find(Q.Att==2),Quad_num,obstacle_flag,lg(1:2,
331             Change_num));
332     end
333
334     %障害物の有無によりフォーメーションを指定
335
336     if obstacle_flag == 1
337         %フォーメーションを指定 (一直線の形)
338         current_formation = 1;
339     else
340         current_formation = 2;
341     end
342
343     %% リーダの処理
344     %
345     %リーダ機の速さ及び単位ベクトルの格納
346     %リーダ機の次ステップにおける座標の計算を行う
347     %
348     %リーダから見た目標地点のベクトル
349     G_1=lg(:,Change_num)-Q.Coord(:,loop,Lnumber);
350     %リーダの速度を算出,計算の仕方は単位ベクトル×スカラーの速さ
351     Q.speed(:,loop,Lnumber)=G_1/norm(G_1)*L_speed;

```

```

345 Q.speed_dir(:,loop,Lnumber) = Q.speed(:,loop,Lnumber) / norm(
346   Q.speed(:,loop,Lnumber));
347
348 %リーダの次ステップでの座標を算出
349 Q.Coord(:,loop+1,Lnumber)=Q.Coord(:,loop,Lnumber)+dt*Q.speed
350   (:,loop,Lnumber);
351
352 %ゼロ除算回避で0を直接代入している
353 Q.l_distance(:,loop,Lnumber) = 0;
354 Q.unit_l_distance(1:2,loop,Lnumber)= 0;
355
356 %% フォロワの処理
357
358 %通常動作の処理
359 if norm(G_1) > 30
360   %
361   % フォロワから見たリーダのベクトル及び単位ベクトルを算出
362   % 属性番号がリーダである機体はゼロ除算を回避するために分岐させる
363   %
364   for i = 1:Quad_num-1
365     Q.l_distance(:,loop,find(Q.Att==i+1))=Q.Coord(:,loop,
366       Lnumber)-Q.Coord(:,loop,find(Q.Att==i+1));
367     Q.unit_l_distance(1:2,loop,find(Q.Att==i+1)) = Q.
368       l_distance(1:2,loop,find(Q.Att==i+1)) / norm(Q.
369       l_distance(1:2,loop,find(Q.Att==i+1)));
370   end
371
372   %
373   % フォロワ間のベクトルを算出(接近禁止領域の規定)
374   % 属性番号に紐づける
375   %
376   for i = 1:Quad_num-1
377     for j = i+1:Quad_num-1
378       VBF(:,loop,j,i) = Q.Coord(:,loop,find(Q.Att==i
379         +1))-Q.Coord(:,loop,find(Q.Att==j+1));
380       VBF_dir(1:2,loop,j,i) = VBF(1:2,loop,j,i)/norm(
381         VBF(1:2,loop,j,i));
382     end
383   end
384
385 %% フォロワ番号により条件確認の範囲が異なる処理

```

```

380    %{
381        フォロワ番号でループを回す
382        フォロワが条件を満たすかの判定はflagで行う
383    %}
384    for i = 1:Quad_num-1
385        flag = 1;
386        %フォロワ単体に関する処理は以下の
387        %for ループと同じインデントで行う
388
389        %他のフォロワとの距離が適切かの判定
390        for j = i+1:Quad_num-1
391            if norm(VBF(1:2,loop,j,i)) >= thresho
392                flag = flag & 1;
393            else
394                flag = flag & 0;
395                break
396            end
397
398        %リーダとの距離が適切かの判定
399        if norm(Q.l_distance(1:2,loop,find(Q.Att == i+1))) >=
399            thresho
400            flag = flag & 1;
401        else
402            flag = flag & 0;
403        end
404
405        new_aixs = axis_transform(Q.speed_dir(:,loop,Lnumber
406                                ));
407        %%記録用
408        D_1 = d(i,current_formation)*rot(new_aixs(1,1),
409            new_aixs(2,1),new_aixs(3,1),AOR(i,2,
410            current_formation))*rot(new_aixs(1,3),new_aixs
411            (2,3),new_aixs(3,3),AOR(i,1,current_formation))*Q.
412            speed_dir(:,loop,Lnumber);
413        %フォロワからみた目標地点へのベクトル
414        l_Error(:,loop,i) = Q.Coord(:,loop,Lnumber)-Q.Coord
415            (:,loop,find(Q.Att==i+1))+D_1;
416
417        if flag == 1
418            %回転ベクトル
419            D = d(i,current_formation)*rot(new_aixs(1,1),
420                new_aixs(2,1),new_aixs(3,1),AOR(i,2,

```

```

        current_formation))*rot(new_aixs(1,3),new_aixs
        (2,3),new_aixs(3,3),AOR(i,1,current_formation
        ))*Q.speed_dir(:,loop,Lnumber);
414    % フォロワからみた目標地点へのベクトル
415    lt = Q.Coord(:,loop,Lnumber)-Q.Coord(:,loop,find(Q
        .Att==i+1))+D;
416
417    % 式 (3.13)
418    h = k01(i,1)+k01(i,2) / (1+norm(lt));
419    % 式 (3.11)
420    Q.Cin_dir(:,loop,find(Q.Att==i+1)) = (lt+h*Q.
        speed_dir(:,loop,Lnumber))/norm(lt+h*Q.
        speed_dir(:,loop,Lnumber));
421    % 式 (3.14)
422    Q.Cin(loop,find(Q.Att==i+1)) = norm(Q.speed(:,_
        loop,Lnumber))*(1+(2/pi)*kps(i,1)*atan(abs(dot
        (lt,Q.speed_dir(:,loop,Lnumber))/kps(i,2))));
423    % 最終的な速度の計算
424    Q.speed(:,loop,find(Q.Att==i+1)) = Q.Cin(loop,
        find(Q.Att==i+1))*Q.Cin_dir(:,loop,find(Q.Att
        ==i+1));
425
426
427    % リーダとフォロワの回避関係
428    Q.unit_l_distance(1:2,loop,find(Q.Att==i+1)) = Q.
        l_distance(1:2,loop,find(Q.Att==i+1)) / norm(Q.
        l_distance(1:2,loop,find(Q.Att==i+1)));
429
430    % リーダに近づきすぎた時の処理
431    if norm(Q.l_distance(1:2,loop,find(Q.Att==i+1))) <
        thresho
432        % norm の前の -1 は
        distance をどちらを起点にするかによって必要か変わる
433        Q.speed(1:2,loop,find(Q.Att==i+1))=-1*norm(Q.
            speed(1:2,loop,Lnumber))*Q.unit_l_distance
            (1:2,loop,find(Q.Att==i+1));% norm の前の -1 は
            distance をどっちを起点にするかによって必要か変わる
434        Q.speed(3,loop,find(Q.Att==i+1)) = 0;
435
436
437    %% フォロワ同士の回避アルゴリズム (若い番号が回避するとす
        る)

```

```

439 %他のフォロワとの距離が適正であるかの判別, フォロワ番号でル
440   for j = 1:Quad_num-1
441     for k= j+1:Quad_num-1
442       if norm(VBF(1:2,loop,k,j)) < thresho
443         Q.speed(1:2,loop,find(Q.Att==j+1))=norm(Q.
444           speed(1:2,loop,find(Q.Att==k+1)))*VBF_dir
445             (1:2,loop,k,j);
446             Q.speed(3,loop,find(Q.Att==j+1)) = 0;
447           end
448         end
449       end
450 %% 各機体の次の座標の計算と各機体が一直線に並んでいる状態
451   %リーダ時の処理 (折り返しのアルゴリズム)
452   %フォロワ番号でループ
453   for k=1:Quad_num-1
454     %全フォロワの次の座標を算出
455     Q.Coord(:,loop+1,find(Q.Att==k+1))=Q.Coord(:,loop,
456       find(Q.Att==k+1))+dt*Q.speed(:,loop,find(Q.Att==k
457         +1));
458     end
459   %目標地点に到達した時の処理
460   elseif norm(G_1) <= 30
461
462   %目標地点到達した時間を記録 (目標地点を表示する際に使用)
463   arrive_time(Change_num,1) = loop;
464   %次の目標地点移る
465   Change_num = Change_num + 1;
466   %フォーメーションの指定
467   currentFormation = 2;
468
469   %{
470     分岐 1: 全目標地点に到達
471       最初に全目標地点に到達した時の座標を保存する (停
472         留のため)
473       分岐 2: 全目標地点に到達していない
474         属性を再定義し, 1ステップ停留させる
475         次の目標地点に向かう
476   %}
477   if Change_num >= LG_num+1
478     %停留座標の取得
479     if Comp_flag == 0

```

```

475         for i = 1:Quad_num
476             Comp_Coord(:,i) = Q.Coord(:,loop,i);
477         end
478     end
479     Comp_flag = 1;
480     Change_num = Change_num-1;
481 else
482     Q.Att(:) = ChangeLeader(Q,loop,Quad_num,lg,Change_num
483                         );
484     for i = 1:Quad_num
485         Comp_Coord(:,i) = Q.Coord(:,loop,i);
486     end
487 end
488 %次のステップでの座標
489 for i = 1:Quad_num
490     Q.Coord(:,loop+1,i) = Comp_Coord(:,i);
491 end
492 %例外が発生した場合には全クワッドロータの動きを止める
493 else
494     %最初に例外になった時の座標を保存する(停留のため)
495     if exception_flag == 0
496         for k = 1:Quad_num
497             exception_Coord(:,k) = Q.Coord(:,k);
498         end
499         exception_flag = 1;
500     end
501
502     %全クワッドローダーを停留させる
503     for k = 1:Quad_num
504         Q.speed(:,loop:loop_num,k) = zeros(3,loop_num-loop
505                         +1);
506         Q.Coord(:,loop+1,k) = exception_Coord(:,k);
507     end
508 end
509 %CoppeliaSimへの反映
510 sim.simxSetObjectPosition(clientID1, target(1), -1, [Q.Coord
511 (1,loop+1,1)/100, Q.Coord(2,loop+1,1)/100,Q.Coord(3,loop
512 +1,1)/100], sim.simx_opmode_oneshot);
513 sim.simxSetObjectPosition(clientID2, target(2), -1, [Q.Coord
514 (1,loop+1,2)/100, Q.Coord(2,loop+1,2)/100,Q.Coord(3,loop
515 +1,2)/100], sim.simx_opmode_oneshot);

```

```

512     sim.simxSetObjectPosition(clientID3, target(3), -1, [Q.Coord
513         (1,loop+1,3)/100, Q.Coord(2,loop+1,3)/100,Q.Coord(3,loop
514         +1,3)/100], sim.simx_opmode_oneshot);
515     sim.simxSetObjectPosition(clientID4, target(4), -1, [Q.Coord
516         (1,loop+1,4)/100, Q.Coord(2,loop+1,4)/100,Q.Coord(3,loop
517         +1,4)/100], sim.simx_opmode_oneshot);
518     sim.simxSetObjectPosition(clientID5, target(5), -1, [Q.Coord
519         (1,loop+1,5)/100, Q.Coord(2,loop+1,5)/100,Q.Coord(3,loop
520         +1,5)/100], sim.simx_opmode_oneshot);
521     sim.simxSetObjectPosition(clientID6, Cylinder, -1, [lg(1,
522         Change_num)/100, lg(2,Change_num)/100,lg(3,Change_num
523         )/100], sim.simx_opmode_oneshot);
524     sim.simxSynchronousTrigger(clientID);
525     pause(0.05);
526 end
527 else
528 disp('Failed connecting to remote API server');
529 end

```

ソースコード A.2: 属性変更に関する関数

```

1  %{
2      リーダーの変更を行う関数
3      gd:1列目は目標点との距離を格納
4          :2列目は属性番号を格納
5  %}
6 function R=ChangeLeader(Q,loop,Quad_num,lg,Change_num)
7     gd = zeros(Quad_num,2);
8
9     %クワッドローターと目標地点の距離と属性番号格納
10    for t = 1:Quad_num
11        gd(t,1) = norm(lg(:,Change_num)-Q.Coord(:,loop,t));
12        gd(t,2) = Q.Att(t);
13    end
14
15    %目標点との距離が大きい順にソート
16    for t = 1:Quad_num-1
17        for s = 1:Quad_num-1
18            if gd(s,1) < gd(s+1,1)
19                tmp1 = gd(s,1);
20                tmp2 = gd(s,2);
21                gd(s,:) = gd(s+1,:);
22                gd(s+1,1) = tmp1;
23                gd(s+1,2) = tmp2;

```

```

24         end
25     end
26 end
27
28 R = zeros(Quad_num,1);
29
30 %属性番号を再決定する
31 for s = 1:Quad_num-1
32     R(Q.Att == gd(s,2)) = s+1;
33 end
34 R(Q.Att==gd(Quad_num,2)) = 1;
35
36 end

```

ソースコード A.3: 他の機体を障害物と識別しないための関数

```

1 %{
2     障害物センサで取得した値が他のクワッドロータであるかを識別する関数
3 %}
4
5 function obstacle_flag = obs_idf(ranges1,ranges2,QuadAng,stepnum,Q,
6     loop,Center_num,Quad_num,obstacle_flag,lg)
7
8     %空行列であれば0を代入している
9     if isempty(ranges1)==1)
10        ranges1 = zeros(1,stepnum);
11    end
12    if isempty(ranges2)==1)
13        ranges2 = zeros(1,stepnum);
14    end
15
16    %センサデータの値が4より小さい要素のインデックスを取得
17    index1 = find(ranges1 < 4);
18    index2 = find(ranges2 < 4);
19
20    %センサデータの値が4であれば障害物は存在しないため除外
21    ranges1 = ranges1(ranges1 < 4);
22    ranges2 = ranges2(ranges2 < 4);
23
24    %検知された障害物がフォロワーかの判定に使用
25    Quadindex = 1:5;
26    Quadindex = Quadindex(Quadindex ~= Center_num);
27
28    %センサ1で障害物を検知した場合

```

```

28 if ~isempty(ranges1)
29 %抽出したデータに対してループを回す
30 for j = 1: length(ranges1)
31     theta1 = 4/3*pi/stepnum * (index1(j)-1) - pi*1/6 -pi/2 +
32         QuadAng(3);
33 %ステップの角度にセンサの距離をかけて座標を求める
34     x = ranges1(j)*cos(theta1) + Q.Coord(1,loop,Center_num
35         )/100 + 0.1;
36     y = ranges1(j)*sin(theta1) + Q.Coord(2,loop,Center_num
37         )/100;
38     obs_Coord = [x*100,y*100];
39     obs_Coord = obs_Coord.';
40     obsflag = 1;
41     for k = 1:Quad_num-1
42         %フォロワーとの実座標の距離で分歧
43         if norm(obs_Coord-Q.Coord(1:2,loop,Quadindex(k))) >
44             100
45             obsflag = obsflag & 1;
46         else
47             obsflag = obsflag & 0;
48         end
49     end
50     if norm(obs_Coord-lg(1:2)) > 100
51         obsflag = obsflag & 1;
52     else
53         obsflag = obsflag & 0;
54     end
55     ranges1(j) = obsflag;
56 end
57
58 if sum(ranges1) > 0
59     obstacle_flag = 1;
60 else
61     obstacle_flag = 0;
62 end
63 %センサ 2で障害物を検知した場合
64 if ~isempty(ranges2)
65     for j = 1:length(ranges2)
66         theta2 = 4/3*pi/stepnum * (index2(j)-1) - pi*1/6 + pi/2
67         + QuadAng(3);

```

```

66         x = ranges2(j)*cos(theta2) + Q.Coord(1,loop,Center_num
67             )/100 -0.1;
68         y = ranges2(j)*sin(theta2) + Q.Coord(2,loop,Center_num
69             )/100;
70         obs_Coord = [x*100,y*100];
71         obs_Coord = obs_Coord.';
72         obsflag = 1;
73         for k = 1:Quad_num-1
74             if norm(obs_Coord-Q.Coord(1:2,loop,Quadindex(k))) >
75                 100
76                 obsflag = obsflag & 1;
77             else
78                 obsflag = obsflag & 0;
79             end
80         end
81         %目標地点の識別
82         if norm(obs_Coord-lg(1:2)) > 100
83             obsflag = obsflag & 1;
84         else
85             obsflag = obsflag & 0;
86         end
87         ranges2(j) = obsflag;
88     end
89
90     if sum(ranges2) > 0
91         obstacle_flag = obstacle_flag || 1;
92     else
93         obstacle_flag = obstacle_flag || 0;
94     end
95 end

```

ソースコード A.4: 回転行列に関する関数

```

1  %{
2      回転行列を返す関数
3      ロドリゲスの回転公式を使用
4  %}
5
6  function r=rot(x,y,z,th)
7  I = eye(3); % 3x3 の単位行列
8  % 直積行列

```

```

9 kron=[x^2,x*y,x*z;x*y,y^2,y*z;x*z,y*z,z^2];
10 % クロス積行列
11 Crossprd=[0, -z, y;z,0, -x;-y,x,0];
12 r=cos(th)*I+sin(th)*Crossprd+(1-cos(th))*kron;
13 end

```

ソースコード A.5: 座標軸回転の関数

```

1 %リーダ・フォロワ制御の座標軸変換に関する関数
2 %
3     ワールド座標系でみたローカル座標軸を取得する関数
4     引数:正規化されたリーダの速度
5     戻り値:ワールド座標系でみたローカル座標軸
6 %
7
8 function new_axis = axis_transform(speed_dir)
9
10 new_axis = zeros(3,3);
11
12 %元のプログラムに合わせるために変数名をvにして転置
13 v = speed_dir(:, .');
14
15 % 元のxyz 軸
16 x_axis = [1, 0, 0];
17 y_axis = [0, 1, 0];
18 z_axis = [0, 0, 1];
19
20 % 回転軸(元のy 軸と新しいy'軸の外積)
21 % 外積の結果として得られるベクトルは元の二つのベクトルに直交するの
22 % で回転軸となる
23 rotation_axis = cross(y_axis, v);
24
25 % 回転角を計算(元のy 軸と新しいy'軸の間の角度)
26 cos_theta = dot(y_axis, v);
27 sin_theta = norm(rotation_axis);
28 theta = atan2(sin_theta, cos_theta);
29
30 % 回転軸の正規化
31 rotation_axis = rotation_axis / norm(rotation_axis);
32
33 % Rodrigues の回転公式を使用して回転行列を計算
34 K = [0, -rotation_axis(3), rotation_axis(2);
35         rotation_axis(3), 0, -rotation_axis(1);
36         -rotation_axis(2), rotation_axis(1), 0];

```

```
36     R = eye(3) + sin(theta) * K + (1 - cos(theta)) * (K ^ 2);
37
38     new_axis(:,1) = R*x_axis';
39     new_axis(:,2) = R*y_axis';
40     new_axis(:,3) = R*z_axis';
41 end
```

付 錄B 協調制御に基づくチョークポイント通過のプログラミスト

ソースコード B.1: 協調制御に基づくチョークポイント通過のメインプログラム

```
1 %%式番号は元論文基準
2 %% メモリのクリア
3 clear;
4 close all
5 clc
6
7 %% coppeliasimとの連携
8 sim=remApi('remoteApi'); % using the prototype file (remoteApiProto.m
    )
9 sim.simxFinish(-1); % just in case, close all opened connections
10 clientID=sim.simxStart('127.0.0.1',19997,true,true,5000,5);
11 sim.simxStartSimulation(clientID,sim.simx_opmode_blocking);
12 pause(0.1)
13
14 %{
15     クワッドロータを増やす際にはコピー&ペースト
16     19999の場所に当たる数字のみcoppeliasimと一致させる
17 %}
18 Quad_clientID(1) = sim.simxStart('127.0.0.1',19999,true,true
    ,5000,5);
19 Quad_clientID(2) = sim.simxStart('127.0.0.1',19995,true,true
    ,5000,5);
20 Quad_clientID(5) = sim.simxStart('127.0.0.1',19993,true,true
    ,5000,5);
21 Quad_clientID(6) = sim.simxStart('127.0.0.1',19991,true,true
    ,5000,5);
22 Quad_clientID(7) = sim.simxStart('127.0.0.1',19989,true,true
    ,5000,5);
```

```
23 Quad_clientID(8) = sim.simxStart('127.0.0.1',19987,true,true
24 ,5000,5);
25 Quad_clientID(3) = sim.simxStart('127.0.0.1',19983,true,true
26 ,5000,5);
27 Quad_clientID(4) = sim.simxStart('127.0.0.1',19981,true,true
28 ,5000,5);
29 %目標地点(円筒形)を表示する
30 CylinderID = sim.simxStart('127.0.0.1',19985,true,true,5000,5);
31 %問題設定に関する定数
32 Quad_num = 8;
33
34 %フォロワの機体数
35 Follower_num = 4;
36
37 %リーダの機体数
38 Leader_num = 4;
39
40 %微小時間(制御入力と乗算して次ステップの座標を計算)
41 Delta_t = 0.2;
42
43 %% coppeliasimから情報を取得
44 if (clientID>-1)
45     disp('Connected to remote API server');
46
47 %handle setting('Quadcopter'等はcoppeliasimから取得)
48 [r, Quad(1)] = sim.simxGetObjectHandle(Quad_clientID(1), '
49     Quadcopter', sim.simx_opmode_blocking);
50 [r, target(1)] = sim.simxGetObjectHandle(Quad_clientID(1), '
51     Quadcopter_target', sim.simx_opmode_blocking);
52
53 [r, Quad(2)] = sim.simxGetObjectHandle(Quad_clientID(2), '
54     Quadcopter#0', sim.simx_opmode_blocking);
55 [r, target(2)] = sim.simxGetObjectHandle(Quad_clientID(2), '
56     Quadcopter_target#0', sim.simx_opmode_blocking);
57
58 [r, Quad(5)] = sim.simxGetObjectHandle(Quad_clientID(4), '
59     Quadcopter#1', sim.simx_opmode_blocking);
60 [r, target(5)] = sim.simxGetObjectHandle(Quad_clientID(4), '
61     Quadcopter_target#1', sim.simx_opmode_blocking);
```

```

57 [r, Quad(6)] = sim.simxGetObjectHandle(Quad_clientID(5), '
58     Quadcopter#2', sim.simx_opmode_blocking);
59 [r, target(6)] = sim.simxGetObjectHandle(Quad_clientID(5), '
60     Quadcopter_target#2', sim.simx_opmode_blocking);
61 [r, Quad(7)] = sim.simxGetObjectHandle(Quad_clientID(6), '
62     Quadcopter#3', sim.simx_opmode_blocking);
63 [r, target(7)] = sim.simxGetObjectHandle(Quad_clientID(6), '
64     Quadcopter_target#3', sim.simx_opmode_blocking);
65 [r, Quad(8)] = sim.simxGetObjectHandle(Quad_clientID(7), '
66     Quadcopter#5', sim.simx_opmode_blocking);
67 [r, target(8)] = sim.simxGetObjectHandle(Quad_clientID(7), '
68     Quadcopter_target#5', sim.simx_opmode_blocking);
69 [r, Quad(3)] = sim.simxGetObjectHandle(Quad_clientID(3), '
70     Quadcopter#4', sim.simx_opmode_blocking);
71 [r, target(3)] = sim.simxGetObjectHandle(Quad_clientID(3), '
72     Quadcopter_target#4', sim.simx_opmode_blocking);
73 [r, Quad(4)] = sim.simxGetObjectHandle(Quad_clientID(3), '
74     Quadcopter#6', sim.simx_opmode_blocking);
75 [r, target(4)] = sim.simxGetObjectHandle(Quad_clientID(3), '
76     Quadcopter_target#6', sim.simx_opmode_blocking);
77 % Get position and orientation data Setting
78 for i = 1:Quad_num
79     [returnCode, QuadPos(i,:)] = sim.simxGetObjectPosition(
80         Quad_clientID(i), Quad(i), -1, sim.simx_opmode_streaming);
81     [returnCode, QuadVel(i,:), dist] = sim.simxGetObjectVelocity(
82         Quad_clientID(i), Quad(i), sim.simx_opmode_streaming);
83     [returnCode, QuadAng(i,:)] = sim.simxGetObjectOrientation(
84         Quad_clientID(i), Quad(i), -1, sim.simx_opmode_streaming);
85     pause(0.1);
86
87     %リーダのみLiDAR と通信
88     if i >= Follower_num+1
89         [errorCode, dist] = sim.simxGetStringSignal(Quad_clientID(
90             i), ['scan ranges' num2str(i) '1'], sim.
91             simx_opmode_streaming);

```

```
84 [errorCode, dist] = sim.simxGetStringSignal(Quad_clientID(
85     i, ['scan ranges' num2str(i) '2'], sim.
86     simx_opmode_streaming);
87 pause(0.1);
88 end
89 end
90
91 %% 変数の規定 (シミュレーション状況により変更)
92
93 %ステップ数 (シミュレーション時間)
94 loop_num = 2500;
95
96 %式 (16),式 (9)を計算する際に使用する制御ゲイン
97 kp = 0.5;
98
99 %リーダー機の機体番号を格納
100 Lnumber = zeros(1,Leader_num);
101
102 %目標地点数
103 LG_num = 2;
104
105 %リーダーの速さ設定
106 L_speed = 0.5;
107
108 %リーダーの目標到達地点
109 lg = zeros(3,LG_num);
110 lg(:,1) = [9.5,0,2.5];
111 lg(:,2) = [8,7.224,2.5];
112
113 %目標点に到達した時間を格納し,目標点の格納に使用する
114 arrive_time = zeros(LG_num,1);
115
116 %目標地点の変更回数
117 Change_num = 1;
118
119 %全目標地点に到達した時点で停滯させるための座標
120 Comp_Coord = zeros(3,Quad_num);
121
122 %上記座標を取得するためのフラグ
123 Comp_flag = 0;
124
125 %例外に到達したことを示すフラグ
126 exception_flag = 0;
```

```

125
126 %例外時に停滞させるための座標
127 exception_Coord = zeros(3,Quad_num);
128
129 %障害物センサ配列の要素数
130 stepnum = 684;
131
132 %障害物があるかの判定
133 obstacle_flag = 0;
134
135 %群れが目標地点に到達したかの判定に用いる閾値
136 threshold1 = 0.2;
137
138 %式 (16)を計算する際のスケーリングベクトル
139 Scaling_vector = [0;0;0];
140
141 %相互作用トポジーの行列
142 %インデックスは属性番号
143 Topology = zeros(Quad_num+1,Quad_num+1);
144
145 %制御入力の計算時に用いる変位,  $\delta_{ij}$ 
146 Delta_i_j = zeros(3,Quad_num-Follower_num+1,Quad_num-Follower_num
+1);
147
148 %障害物座標を格納するセル (要素数は仮想リーダに隣接するリーダの数)
149 obs_Coord = cell(2,1);
150
151 %% クワッドローターの構造体の定義
152
153 %{
154     この構造体では機体番号 (属性番号で無い)に紐づく情報を保持してい
155         る
156     機体番号自体はインデックスが対応する
157     各クワッドローターの情報を格納する構造体の定義
158     Q:構造体名
159     Att:属性番号 (機体番号に対応する属性番号を格納)
160         1~4(1~Follower_num): フォロワ
161         5~8(Follower_num+1~Quad_num): リーダ
162         9(Quad_num+1): 仮想リーダ
163     Coord:座標
164         第1インデックス:x,y,z
165         第2インデックス:ステップ
166         第3インデックス:機体番号

```

```

166      speed : 速さ , インデックスは座標と同じ
167      Cin: 制御入力の大きさ
168          第1インデックス:x,y,z
169          第2インデックス:ステップ
170          第3インデックス:機体番号
171      %}
172      Q.Att = zeros(Quad_num+1,1);
173      Q.Coord = zeros(3,loop_num+1,Quad_num+1);
174      Q.speed = zeros(3,loop_num+1,Quad_num+1);
175      Q.Cin = zeros(3,loop_num+1,Quad_num+1);
176
177      %% 初期設定
178
179      %属性の初期設定
180      Q.Att = 1:Quad_num+1;
181
182      %{
183          相互作用トポロジの設定
184          属性番号がインデックス
185      %}
186      Topology(1,2) = 0.1; Topology(1,3) = 0.1; Topology(1,4) = 0.1;
187      Topology(2,1) = 0.1; Topology(2,3) = 0.1; Topology(2,4) = 0.1;
188      Topology(3,1) = 0.1; Topology(3,4) = 0.1;
189      Topology(4,2) = 0.1; Topology(4,3) = 0.1;
190      Topology(5,1) = 0.3; Topology(5,3) = 0.1; Topology(5,6) = 0.1;
191          Topology(5,7) = 0.2;
192      Topology(6,2) = 0.3; Topology(6,4) = 0.1; Topology(6,5) = 0.1;
193          Topology(6,8) = 0.1;
194      Topology(7,1) = 0.1; Topology(7,3) = 0.2; Topology(7,5) = 0.2;
195          Topology(7,8) = 0.1;
196      Topology(8,2) = 0.1; Topology(8,4) = 0.2; Topology(8,6) = 0.2;
197          Topology(8,7) = 0.1;
198      Topology(9,5) = 0.2; Topology(9,6) = 0.2;
199
200      %フォロワ機の初期座標設定
201      Q.Coord(1,1,1)=-5.65; Q.Coord(2,1,1)=-0.4; Q.Coord(3,1,1)=2.25;
202      Q.Coord(1,1,2)=-6; Q.Coord(2,1,2)=2; Q.Coord(3,1,2)=2.25;
203      Q.Coord(1,1,3)=-6.5; Q.Coord(2,1,3)=-0.8; Q.Coord(3,1,3)=2.25;
204      Q.Coord(1,1,4)=-6.8; Q.Coord(2,1,4)=2.2; Q.Coord(3,1,4)=2.25;
205
206      %リーダ機の初期座標設定
207      Q.Coord(1,1,5)=-5.1; Q.Coord(2,1,5)=0.375; Q.Coord(3,1,5)=2.5;
208      Q.Coord(1,1,6)=-5.5; Q.Coord(2,1,6)=1.45; Q.Coord(3,1,6)=2.5;

```

```

205     Q.Coord(1,1,7)=-7.4; Q.Coord(2,1,7)=0.525; Q.Coord(3,1,7)=2.5;
206     Q.Coord(1,1,8)=-7.5; Q.Coord(2,1,8)=1.7; Q.Coord(3,1,8)=2.5;
207
208 %仮想リーダの初期座標
209     Q.Coord(1,1,Quad_num+1)=-4.1; Q.Coord(2,1,Quad_num+1)=0; Q.Coord
210         (3,1,Quad_num+1)=2.5;
211
212 %変位制約 (デフォルト値)
213     x1 = 2; y1 = 3;
214     x2 = 0.8; y2 = 1.5;
215
216 %変位制約と変更する場合に使用
217     evacuation(1) = x1; evacuation(2) = y1;
218     evacuation(3) = x2; evacuation(4) = y2;
219
220 %{
221     第3引数からみた第2引数への変位制約
222     インデックスは属性番号に対応しているがフォロワ数だけマイナス
223 %}
224     Delta_i_j = displacement_change(Delta_i_j,x1,y1,x2,y2,Q,lg,
225         Quad_num,1,Change_num);
226
227 %縮小する際の減量
228     weight_loss = 0.05;
229
230 %進行方向の障害物の有無を示す
231     direction_obsflag = 1;
232
233 %クワッドロータの初期座標をコペラシムに反映している
234     for i= 1:Quad_num
235         sim.simxSetObjectPosition(Quad_clientID(i),Quad(i),-1,[Q.
236             Coord(1,1,i),Q.Coord(2,1,i),Q.Coord(3,1,i)],sim.
237             simx_opmode_oneshot);
238     end
239     sim.simxSetObjectPosition(CylinderID,Cylinder,-1,[lg(1,1),lg(2,1),
240             lg(3,1)],sim.simx_opmode_oneshot);
241     sim.simxSynchronousTrigger(clientID);
242
243 %% 制御処理
244 %ステップ分ループを回す
245     for loop=1:loop_num
246
247         % 実座標と速度の取得

```

```

243     for i = 1:Quad_num
244         [returnCode,QuadPos] = sim.simxGetObjectPosition(
245             Quad_clientID(i),Quad(i),-1,sim.simx_opmode_buffer);
246         Q.Coord(:,loop,i) = QuadPos(:);
247         pause(0.05);
248         [returnCode, QuadVel, dist] = sim.simxGetObjectVelocity(
249             Quad_clientID(i), Quad(i), sim.simx_opmode_streaming);
250         Q.speed(:,loop,i) = QuadVel(:);
251     end
252
253     %{
254         リーダー属性のクワッドロータの機体番号を取得
255         第1引数の条件を満たす数値を第2引数で指定する数だけ取得す
256         る
257     %}
258     Lnumber = find(Q.Att >= Follower_num+1 & Q.Att <= Quad_num,
259                     Leader_num);
260
261     %仮想リーダと隣接しているリーダ属性番号を取得 (Neighbours of
262     %Virtual Leader number)
263     NofVLnumber = find(Topology(Quad_num+1,Follower_num+1:Quad_num
264                         ) > 0) + Follower_num;
265
266     %属性番号を機体番号に変換
267     for i = 1:length(NofVLnumber)
268         NofVLnumber(i) = find(Q.Att == NofVLnumber(i));
269     end
270
271     %仮想リーダと隣接していないリーダ機の属性番号を取得
272     InNofVLnumber = find(Topology(Quad_num+1,Follower_num+1:
273                                 Quad_num) == 0)+Follower_num;
274
275     %属性番号を機体番号に変換
276     for i = 1:length(InNofVLnumber)
277         InNofVLnumber(i) = find(Q.Att == InNofVLnumber(i));
278     end
279
280     %仮想リーダから見た目標地点への距離 (到達判定に使用)
281     G_distance = norm(lg(:,Change_num)-Q.Coord(:,loop,Quad_num
282                         +1));
283
284     %% 条件分岐及び制御入力の算出
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
241
```

```

278     %群れが目標地点に到達したかの判定
279     if G_distance >= threshold1
280
281         %% 障害物判定及び座標の取得
282         %
283             障害物センサの値を取得し,それを数値に変換し格納
284             (2×リーダの機体数):行数
285             (障害物センサの分解能):列数
286         %
287         ranges = zeros(2*Leader_num,stepnum);
288
289         %進行方向の障害物の有無を示すフラグ (1:無, 0:有)
290         direction_obsflag = 1;
291
292         %仮想リーダに隣接するリーダによるチョークポイントの検知
293         for i = 1:length(NofVLnumber)
294             [errorCode, tmp_ranges1] = sim.simxGetStringSignal(
295                 Quad_clientID(NofVLnumber(i)), ['scan ranges'
296                 num2str(NofVLnumber(i)) '1'], sim.
297                 simx_opmode_buffer);
298             [errorCode, tmp_ranges2] = sim.simxGetStringSignal(
299                 Quad_clientID(NofVLnumber(i)), ['scan ranges'
300                 num2str(NofVLnumber(i)) '2'], sim.
301                 simx_opmode_buffer);
302             %
303             %数値データに変換
304             ranges(2*i-1,:) = sim.simxUnpackFloats(tmp_ranges1);
305             ranges(2*i,:)= sim.simxUnpackFloats(tmp_ranges2);
306
307             [returnCode, QuadAng] = sim.simxGetObjectOrientation(
308                 Quad_clientID(NofVLnumber(i)), Quad(NofVLnumber(i)),
309                 -1, sim.simx_opmode_buffer);
310             pause(0.05);
311             %
312             %障害物があるかの判定
313             [obstacle_flag,obs_Coord{i}] = obs_idf(ranges(2*i
314                 -1,:),ranges(2*i,:),QuadAng,stepnum,Q,loop,
315                 NofVLnumber(i),Quad_num,obstacle_flag,lg(1:2,
316                 Change_num));
317
318             %
319             %障害物を検知した際には進行方向成分に障害物が存在する
320             %かの判定も行う
321             if obstacle_flag == 1
322                 direction_obsflag = direction_obs_detction(
323                     obs_Coord{i},NofVLnumber,Q,loop,Change_num,

```

```

        Quad_num,lg,i,direction_obsflag);
308    end
309  end
310
311 %仮想リーダに隣接しないリーダによるチョークポイントの検出
312 if obstacle_flag == 0
313   for i = 1:length(InNofVLnumber)
314     [errorCode, tmp_ranges1] = sim.simxGetStringSignal
315       (Quad_clientID(InNofVLnumber(i)), ['scan ranges
316         ' num2str(InNofVLnumber(i)) '1'], sim.
317           simx_opmode_buffer);
318     [errorCode, tmp_ranges2] = sim.simxGetStringSignal
319       (Quad_clientID(InNofVLnumber(i)), ['scan ranges
320         ' num2str(InNofVLnumber(i)) '2'], sim.
321           simx_opmode_buffer);
322     %数値データに変換
323     ranges(2*i+2*length(NofVLnumber)-1,:) = sim.
324       simxUnpackFloats(tmp_ranges1);
325     ranges(2*i+2*length(NofVLnumber),:) = sim.
326       simxUnpackFloats(tmp_ranges2);
327
328     [returnCode, QuadAng] = sim.
329       simxGetObjectOrientation(Quad_clientID(
330         InNofVLnumber(i)),Quad(InNofVLnumber(i)), -1,
331         sim.simx_opmode_buffer);
332     pause(0.05);
333     %障害物があるかの判定
334     obstacle_flag = obs_idf(ranges(2*i+2*length(
335       NofVLnumber)-1,:),ranges(2*i+2*length(
336       NofVLnumber),:),QuadAng,stepnum,Q,loop,
337       InNofVLnumber(i),Quad_num,obstacle_flag,lg(1:2,
338       Change_num));
339
340   end
341 end
342
343 %% 制御入力の計算
344
345 %フォーメーションが完成判定のフラグ
346 perfect_flag = 1;
347
348 %フォーメーション完成の判定を行う関数
349 perfect_flag = formation_judge(perfect_flag,Follower_num,
350   Quad_num,Q,Delta_i_j,loop);

```

```

334
335 %フォーメーションの形成が出来ている場合
336 if perfect_flag == 1
337
338 %進行方向に障害物が存在しない場合は障害物の有無に関わ
339 %らず進行
340 if direction_obsflag == 1
341 %仮想リーダーから見た目標地点のベクトル
342 Goal_distance=lg(:,Change_num)-Q.Coord(:,loop,
343 Quad_num+1);
344 %仮想リーダーの速度を算出,計算の仕方は単位ベクトル
345 %スカラーの速さ
346 Q.Cin(:,loop,find(Q.Att==Quad_num+1))=Goal_distance
347 /norm(Goal_distance)*L_speed;
348 Q.speed(:,loop,find(Q.Att==Quad_num+1)) = Q.Cin(:,loop,
349 find(Q.Att==Quad_num+1));
350 else
351 %仮想リーダの速さを0に設定する
352 Q.Cin(:,loop,find(Q.Att == Quad_num+1)) = 0;
353 Q.speed(:,loop,find(Q.Att == Quad_num+1)) = 0;
354
355 %進行方向に対して垂直成分のみ減量
356
357 evacuation(2) = evacuation(2) - weight_loss;
358 evacuation(4) = evacuation(4) - weight_loss/2;
359 Delta_i_j = displacement_change(Delta_i_j,
360 evacuation(1),evacuation(2),evacuation(3),
361 evacuation(4),Q,lg,Quad_num,loop,Change_num);
362 end
363
364 %障害物が存在しなければ,元の広がった形に戻す
365 if obstacle_flag == 0
366 Delta_i_j = displacement_change(Delta_i_j,x1,y1,x2
367 ,y2,Q,lg,Quad_num,loop,Change_num);
368 end
369
370 %フォーメーションが完成していない場合
371 else
372 Q.Cin(:,loop,find(Q.Att==Quad_num+1)) = 0;
373 Q.speed(:,loop,find(Q.Att==Quad_num+1)) = 0;
374 end
375
376 %% 全リーダ機への制御入力の計算
377 %}

```

```

369 式(16)に基づいてリーダ機の制御入力を計算
370 制御入力を求めたいリーダについて以下の計算を行う
371 •注目するリーダ機から見て隣接しているクワッドローダ
372     ヘのエッジの重みの和,式(16)の $\gamma$ 
373 •隣接しているクワッドロータから見た注目するリーダ機への
374     ベクトルを算出,式(3)の $\delta$ 
375      $\delta$ のsの積を求める,式(14)の $\delta^{\sim}$ 
376      $\delta^{\sim}$ を用いて, $\xi^{\sim}$ を算出,式(15)
377 •取得した速度を使用して式(16)を算出
378 %}
379
380 %スケーリングベクトルの調整
381 %実際の値-理論値
382 preDistance1 = abs(Q.Coord(:,loop,find(Q.Att==Follower_num
383     +2))-Q.Coord(:,loop,find(Q.Att==Follower_num+1)))-abs(
384     Delta_i_j(:,2,1));
385 preDistance2 = abs(Q.Coord(:,loop,find(Q.Att==Follower_num
386     +1))-Q.Coord(:,loop,find(Q.Att==Follower_num+3)))-abs(
387     Delta_i_j(:,3,1));
388
389 %スケーリングベクトルの調整
390 Scaling_vector(1) = Scaling_adjustment(preDistance2(1));
391 Scaling_vector(2) = Scaling_adjustment(preDistance1(2));
392 Scaling_vector(3) = 1;
393
394 %注目する機体に対するループ
395 for i = Follower_num+1:Quad_num
396
397     %{
398         式(16)で使用する $\gamma_i$ 
399         隣接するノードからみた注目機への重みの和
400     %}
401     %%
402     Ganma_i = sum(Topology(Follower_num+1:Quad_num+1,i));
403
404     %式(16)で使用する $\xi_i$ ,式(15)
405     Guzai_Hat_i = 0;
406
407     %式(16)の $a_{i,j} \times p'_{-j}$ 
408     Weight_vector = 0;
409
410     for j = Follower_num+1:Quad_num+1
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1286
1287
1288
1288
1289
1290
1291
1292
1293
1294
1295
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1308
1309
1310
1310
1311
1312
1313
1314
1315
1316
1317
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1326
1327
1328
1328
1329
1330
1331
1332
1333
1334
1335
1336
1336
1337
1338
1338
1339
1340
1341
1342
1343
1344
1345
1346
1346
1347
1348
1348
1349
1350
1351
1352
1353
1354
1355
1356
1356
1357
1358
1358
1359
1360
1361
1362
1363
1364
1365
1366
1366
1367
1368
1368
1369
1370
1371
1372
1373
1374
1375
1375
1376
1377
1377
1378
1379
1380
1381
1382
1383
1384
1385
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1408
1409
1410
1410
1411
1412
1413
1414
1415
1416
1416
1417
1418
1418
1419
1420
1421
1422
1423
1424
1425
1425
1426
1427
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1435
1436
1437
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1455
1456
1457
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1465
1466
1467
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1475
1476
1477
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1485
1486
1487
1487
1488
1489
1489
1490
1491
1492
1493
1494
1494
1495
1496
1496
1497
1498
1498
1499
1500
1501
1502
1503
1504
1505
1505
1506
1507
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1515
1516
1517
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1525
1526
1527
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1535
1536
1537
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1545
1546
1547
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1555
1556
1557
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1565
1566
1567
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1575
1576
1577
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1585
1586
1587
1587
1588
1589
1589
1590
1591
1592
1593
1594
1594
1595
1596
1596
1597
1598
1598
1599
1600
1601
1602
1603
1604
1604
1605
1606
1606
1607
1608
1608
1609
1610
1611
1612
1613
1614
1614
1615
1616
1616
1617
1618
1618
1619
1620
1621
1622
1623
1624
1624
1625
1626
1626
1627
1628
1628
1629
1630
1631
1632
1633
1634
1634
1635
1636
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1644
1645
1646
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1654
1655
1656
1656
1657
1658
1658
1659
1660
1661
1662
1663
1664
1664
1665
1666
1666
1667
1668
1668
1669
1670
1671
1672
1673
1674
1674
1675
1676
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1684
1685
1686
1686
1687
1688
1688
1689
1690
1691
1692
1693
1693
1694
1695
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1704
1705
1706
1706
1707
1708
1708
1709
1710
1711
1712
1713
1714
1714
1715
1716
1716
1717
1718
1718
1719
1720
1721
1722
1723
1724
1724
1725
1726
1726
1727
1728
1728
1729
1730
1731
1732
1733
1734
1734
1735
1736
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1744
1745
1746
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1754
1755
1756
1756
1757
1758
1758
1759
1760
1761
1762
1763
1764
1764
1765
1766
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1774
1775
1776
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1784
1785
1786
1786
1787
1788
1788
1789
1790
1791
1792
1793
1794
1794
1795
1796
1796
1797
1798
1798
1799
1800
1801
1802
1803
1804
1804
1805
1806
1806
1807
1808
1808
1809
1810
1811
1812
1813
1814
1814
1815
1816
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1824
1825
1826
1826
1827
1828
1828
1829
1830
1831
1832
1833
1834
1834
1835
1836
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1844
1845
1846
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1854
1855
1856
1856
1857
1858
1858
1859
1860
1861
1862
1863
1864
1864
1865
1866
1866
1867
1868
1868
1869
1870
1871
1872
1873
1874
1874
1875
1876
1876
1877
1878
1878
1879
1880
1881
1882
1883
1884
1884
1885
1886
1886
1887
1888
1888
1889
1890
1891
1892
1893
1893
1894
1895
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1904
1905
1906
1906
1907
1908
1908
1909
1910
1911
1912
1913
1914
1914
1915
1916
1916
1917
1918
1918
1919
1920
1921
1922
1923
1924
1924
1925
1926
1926
1927
1928
1928
1929
1930
1931
1932
1933
1934
1934
1935
1936
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1944
1945
1946
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1954
1955
1956
1956
1957
1958
1958
1959
1960
1961
1962
1963
1964
1964
1965
1966
1966
1967
1968
1968
1969
1970
1971
1972
1973
1974
1974
1975
1976
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1984
1985
1986
1986
1987
1988
1988
1989
1990
1991
1992
1993
1993
1994
1995
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2004
2005
2006
2006
2007
2008
2008
2009
2010
2011
2012
2013
2014
2014
2015
2016
2016
2017
2018
2018
2019
2020
2021
2022
2023
2024
2024
2025
2026
2026
2027
2028
2028
2029
2030
2031
2032
2033
2034
2034
2035
2036
2036
2037
2038
2038
2039
2040
2041
2042
2043
2044
2044
2045
2046
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2054
2055
2056
2056
2057
2058
2058
2059
2060
2061
2062
2063
2064
2064
2065
2066
2066
2067
2068
2068
2069
2070
2071
2072
2073
2074
2074
2075
2076
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2084
2085
2086
2086
2087
2088
2088
2089
2090
2091
2092
2093
2093
2094
2095
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2104
2105
2106
2106
2107
2108
2108
2109
2110
2111
2112
2113
2114
2114
2115
2116
2116
2117
2118
2118
2119
2120
2121
2122
2123
2124
2124
2125
2126
2126
2127
2128
2128
2129
2130
2131
2132
2133
2134
2134
2135
2136
2136
2137
2138
2138
2139
2140
2141
2142
2143
2144
2144
2145
2146
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2154
2155
2156
2156
2157
2158
2158
2159
2160
2161
2162
2163
2164
2164
2165
2166
2166
2167
2168
2168
2169
2170
2171
2172
2173
2174
2174
2175
2176
2176
2177
2178
2178
2179
2180
2181
2182
2183
2184
2184
2185
2186
2186
2187
2188
2188
2189
2190
2191
2192
2193
2193
2194
2195
2195
2196
2197
2197
2198
2199
2199
2200
2201
2202
2203
2204
2204
2205
2206
2206
2207
2208
2208
2209
2210
2211
2212
2213
2214
2214
2215
2216
2216
2217
2218
2218
2219
2220
2221
2222
2223
2224
2224
2225
2226
2226
2227
2228
2228
2229
2230
2231
2232
2233
2234
2234
2235
2236
2236
2237
2238
2238
2239
2240
2241
2242
2243
2244
2244
2245
2246
2246
2247
2248
2248
2249
2250
2251
2252
2253
2254
2254
2255
2256
2256
2257
2258
2258
2259
2260
2261
2262
2263
2264
2264
2265
2266
2266
2267
2268
2268
2269
2270
2271
2272
2273
2274
2274
2275
2276
2276
2277
2278
2278
2279
2280
2281
2282
2283
2284
2284
2285
2286
2286
2287
2288
2288
2289
2290
2291
2292
2293
2293
2294
2295
2295
2296
2297
2297
2298
2299
2299
2300
2301
2302
2303
2304
2304
2305
2306
2306
2307
2308
2308
2309
2310
2311
2312
2313
2314
2314
2315
2316
2316
2317
2318
2318
2319
2320
2321
2322
2323
2324
2324
2325
2326
2326
2327
2328
2328
2329
2330
2331
2332
2333
2334
2334
2335
2336
2336
2337
2338
2338
2339
2340
2341
2342
2343
2344
2344
2345
2346
2346
2347
2348
2348
2349
2350
2351
2352
2353
2354
2354
2355
2356
2356
2357
2358
2358
2359
2360
2361
2362
2363
2364
2364
2365
2366
2366
2367
2368
2368
2369
2370
2371
2372
2373
2374
2374
2375
2376
2376
2377
2378
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
247
```

```

408      %式 (3)
409      %隣接から見たリーダ機へのベクトル
410      Distance = Q.Coord(:,loop,find(Q.Att==i))-Q.Coord
411          (:,loop,find(Q.Att==j));
412
413      %式 (15)
414      %隣接機から見たリーダへの変位制約を使用
415      Guzai_Hat_i = Guzai_Hat_i + Topology(j,i)*(
416          Distance-Scaling_vector.*Delta_i_j(:,i-
417          Follower_num,j-Follower_num));
418
419      if perfect_flag == 1
420          Guzai_Hat_i = Guzai_Hat_i + Topology(j,i)*(
421              Distance-Delta_i_j(:,i-Follower_num,j-
422              Follower_num));
423
424      end
425
426      %式 (16)の $a_{i-j} \times p'_{-j}$ 
427      Weight_vector = Weight_vector + Topology(j,i) * Q.
428          speed(:,loop,find(Q.Att==j));
429
430      %式 (16)を計算
431      Q.Cin(:,loop,find(Q.Att==i)) = -1/Ganma_i*(kp*
432          Guzai_Hat_i-Weight_vector);
433
434      %% 全フォロワーの制御入力の計算
435
436      %リーダ機の制御入力の計算とほぼ一致
437      %注目するフォロワ機に対するループ
438      for i = 1:Follower_num
439
440          %式 (9)で使用する $\gamma_k$ 
441          Ganma_k = sum(Topology(1:Quad_num,i));
442
443          %式 (9)で使用する $\xi_k$ 
444          Guzai_k = 0;
445
446          %式 (9)の $a_{k-j} \times p'_{-j}$ 
447          Weight_vector = 0;
448
449          for j = 1:Quad_num

```

```

444
445      %隣接機から見たフォロワへのベクトル
446      %式(3)
447      Delta_k_j = Q.Coord(:,loop,find(Q.Att==i))-Q.Coord
448          (:,loop,find(Q.Att==j));
449
450      %式(15)
451      Guzai_k = Guzai_k + Topology(j,i)*Delta_k_j;
452
453      %式(16)のa_i_j × p'_j
454      Weight_vector = Weight_vector + Topology(j,i)*Q.
455          speed(:,loop,find(Q.Att==j));
456      end
457
458      %式(16)を計算
459      Q.Cin(:,loop,find(Q.Att==i)) = -1/Gamma_k*(kp*Guzai_k
460          -Weight_vector);
461      end
462
463      %全クワッドロータの次の座標を算出
464      for i = 1:Quad_num
465          Q.Coord(:,loop+1,find(Q.Att==i))=Q.Coord(:,loop,find(Q.
466              Att==i))+Delta_t.*Q.Cin(:,loop,find(Q.Att==i));
467      end
468      Q.Coord(:,loop+1,find(Q.Att==Quad_num+1))=Q.Coord(:,loop,find
469          (Q.Att==Quad_num+1))+Delta_t.*Q.Cin(:,loop,find(Q.Att==
470          Quad_num+1));
471
472      %目標地点に到達した時の処理
473      elseif G_distance < threshold1
474
475          %目標地点到達した時間を記録(目標地点を表示する際に使用)
476          arrive_time(Change_num,1) = loop;
477          %次の目標地点移る
478          Change_num = Change_num + 1;
479
480          %{
481              分岐1:全目標地点に到達
482                  最初に全目標地点に到達した時の座標を保存する(停
483                      留のため)
484              分岐2:全目標地点に到達していない
485                  属性を再定義し,1ステップ停留させる
486                  次の目標地点に向かう

```

```

480    %}
481    if Change_num >= LG_num+1
482        %停留座標の取得
483        if Comp_flag == 0
484            for i = 1:Quad_num+1
485                Comp_Coord(:,i) = Q.Coord(:,loop,i);
486            end
487        end
488        Comp_flag = 1;
489        Change_num = Change_num-1;
490        for i = 1:Quad_num+1
491            Q.Coord(:,loop+1,i) = Comp_Coord(:,i);
492        end
493    else
494
495        %属性変更
496        Q.Att(:) = ChangeLeader(Q,loop,Quad_num,lg,Change_num,
497                                  Follower_num);
498
499        for i = 1:Quad_num
500            Comp_Coord(:,i) = Q.Coord(:,loop,i);
501            Q.Coord(:,loop+1,i) = Comp_Coord(:,i);
502        end
503
504        %{
505            次ステップでの仮想リーダの座標を規定
506            次ステップの座標を用いて変位制約を規定
507        %}
508        Q.Coord(:,loop+1,find(Q.Att==Quad_num+1)) =
509            VL_CoordChange(Q,loop,x2,y2);
510        Delta_i_j = displacement_change(Delta_i_j,x1,y1,x2,y2,
511                                         Q,lg,Quad_num,loop+1,Change_num);
512    end
513
514    %例外が発生した場合には全クワッドロータの動きを止める
515    else
516        %最初に例外になった時の座標を保存する(停留のため)
517        if exception_flag == 0
518            for k = 1:Quad_num
519                exception_Coord(:,k) = Q.Coord(:,k);
520            end
521            exception_flag = 1;
522        end

```

```

520
521      %全クワッドローダーを停留させる
522      for k = 1:Quad_num
523          Q.speed(:,loop:loop_num,k) = zeros(3,loop_num-loop
524                                      +1);
525          Q.Coord(:,loop+1,k) = exception_Coord(:,k);
526      end
527
528      %CoppeliaSimへの反映
529      for i = 1:Quad_num
530          sim.simxSetObjectPosition(Quad_clientID(i), target(i), -1,
531                                      [Q.Coord(1,loop+1,i), Q.Coord(2,loop+1,i),Q.Coord(3,
532                                      loop+1,i)], sim.simx_opmode_oneshot);
533      end
532      sim.simxSetObjectPosition(CylinderID, Cylinder, -1, [lg(1,
533          Change_num), lg(2,Change_num),lg(3,Change_num)], sim.
534          simx_opmode_oneshot);
533      sim.simxSynchronousTrigger(clientID);
534      pause(0.05);
535  end
536 else
537     disp('Failed connecting to remote API server');
538 end

```

ソースコード B.2: 他の機体を障害物と識別しないための関数

```

1  %{
2      障害物センサで取得した値がフォロワーであるかを識別する関数
3      注目機体のインデックスがCenter_num
4      障害物の有無を示すフラグと検知した障害物の座標を返す
5  %}
6
7 function [obstacle_flag,data] = obs_idf(ranges1,ranges2,QuadAng,
8                                         stepnum,Q,loop,Center_num,Quad_num,obstacle_flag,lg)
9
10    %空行列であれば0を代入している
11    if isempty(ranges1)==1)
12        ranges1 = zeros(1,stepnum);
13    end
14    if isempty(ranges2)==1)
15        ranges2 = zeros(1,stepnum);
16    end

```

```

17 %センサデータの値が4より小さい要素のインデックスを取得
18 index1 = find(ranges1 < 4);
19 index2 = find(ranges2 < 4);
20
21 %センサデータの値が4であれば障害物は存在しないため除外
22 ranges1 = ranges1(ranges1 < 4);
23 ranges2 = ranges2(ranges2 < 4);
24
25 %検知された障害物が他のクワッドロータかの判定に使用
26 Quadindex = 1:Quad_num;
27 Quadindex = Quadindex(Quadindex ~= Center_num);
28
29 %障害物の可能性のある座標の長さ分の配列
30 data = zeros(2,length(ranges1)+length(ranges2));
31
32 %センサ1で障害物を検知した場合
33 if ~isempty(ranges1)
34     %抽出したデータに対してループを回す
35     for j = 1: length(ranges1)
36         theta1 = 4/3*pi/stepnum * (index1(j)-1) - pi*1/6 -pi/2 +
37             QuadAng(3);
38         %ステップの角度にセンサの距離をかけて座標を求める
39         x = ranges1(j)*cos(theta1) + Q.Coord(1,loop,Center_num) +
40             0.1;
41         y = ranges1(j)*sin(theta1) + Q.Coord(2,loop,Center_num);
42         obs_Coord = [x,y].';
43
44         obsflag = 1;
45         for k = 1:Quad_num-1
46             %フォロワーとの実座標の距離で分岐
47             if norm(obs_Coord-Q.Coord(1:2,loop,Quadindex(k))) >
48                 0.3
49                 obsflag = obsflag & 1;
50             else
51                 obsflag = obsflag & 0;
52             end
53         end
54         if norm(obs_Coord-lg(1:2)) > 0.2
55             obsflag = obsflag & 1;
56         else
57             obsflag = obsflag & 0;
58         end
59     end
60 
```

```

57      %外部にデータを出力するため
58      if obsflag == 1
59          data(:,j) = obs_Coord;
60      end
61
62      ranges1(j) = obsflag;
63  end
64
65      if sum(ranges1) > 0
66          obstacle_flag = 1;
67      else
68          obstacle_flag = 0;
69      end
70  end
71  %センサ2で障害物を検知した場合
72  if ~isempty(ranges2)
73      for j = 1:length(ranges2)
74          theta2 = 4/3*pi/stepnum * (index2(j)-1) - pi*1/6 + pi/2
75              + QuadAng(3);
76          x = ranges2(j)*cos(theta2) + Q.Coord(1,loop,Center_num
77              )-0.1;
78          y = ranges2(j)*sin(theta2) + Q.Coord(2,loop,Center_num);
79          obs_Coord = [x,y].';
80
81          obsflag = 1;
82          for k = 1:Quad_num-1
83              if norm(obs_Coord-Q.Coord(1:2,loop,Quadindex(k))) >
84                  0.3
85                  obsflag = obsflag & 1;
86              else
87                  obsflag = obsflag & 0;
88              end
89          end
90
91          %目標地点の識別
92          if norm(obs_Coord-lg(1:2)) > 0.2
93              obsflag = obsflag & 1;
94          else
95              obsflag = obsflag & 0;
96          end
97
98          %外部にデータを出力するため
99          if obsflag == 1
100             data(:,j) = obs_Coord;
101         end
102     end
103 
```

```

97         end
98
99         ranges2(j) = obsflag;
100    end
101
102    if sum(ranges2) > 0
103        obstacle_flag = obstacle_flag || 1;
104    else
105        obstacle_flag = obstacle_flag || 0;
106    end
107 end
108
109 idx = data(1,:) == 0 & data(2,:) == 0;
110 data = data(:,~idx);
111
112 end

```

ソースコード B.3: フォーメーションの完成判定の関数

```

1 %{
2     フォーメーションが完成しているかの判定を行う関数
3 %}
4
5 function perfect_flag = formation_judge(perfect_flag,Follower_num,
6                                         Quad_num,Q,Delta_i_j,loop)
7
8     %リーダ機による凸包が形成出来ているかの判定
9     for i = Follower_num+1:Quad_num+1
10
11         for j = Follower_num+1:Quad_num+1
12
13             %隣接から見たリーダ機へのベクトル
14             Distance = Q.Coord(:,loop,find(Q.Att==j))-Q.Coord(:,loop,
15                               find(Q.Att==i));
16
17             if nnz(Delta_i_j(:,i-Follower_num,j-Follower_num)) ~= 0
18
19                 Variation_differences = Distance(1:2)-Delta_i_j(1:2,j
20                                         -Follower_num,i-Follower_num);
21                 Variation_differences = Variation_differences(
22                     Variation_differences > 0.2);
23
24             if sum(Variation_differences) == 0
25                 perfect_flag = perfect_flag & 1;

```

```

22         else
23             perfect_flag = perfect_flag & 0;
24         end
25     end
26 end
27
28
29
30 %フォロワーがリーダ機が形成する凸包内にフォロワが収束しているかの判
31 %定
32 %以下が追加機能
33 x = zeros(Quad_num-Follower_num,1);
34 y = zeros(Quad_num-Follower_num,1);
35 for i = Follower_num+1:Quad_num
36     x(i-Follower_num,1) = Q.Coord(1,loop,find(Q.Att==i));
37     y(i-Follower_num,1) = Q.Coord(2,loop,find(Q.Att==i));
38 end
39
40 shp = alphaShape(x,y);
41
42 for i = 1:Follower_num
43
44     tf = inShape(shp,Q.Coord(1,loop,find(Q.Att==i)),Q.Coord(2,
45         loop,find(Q.Att==i)));
46
47     if tf == 1
48         perfect_flag = perfect_flag & 1;
49     else
50         perfect_flag = perfect_flag & 0;
51     end
52 end
53 end

```

ソースコード B.4: 変位制約を決定する関数

```

1 %
2 変位制約を変更する関数
3 回転の無い状態で変位ベクトルを算出し,それに回転関数を掛け合わせ斜
4 め方向の変位を算出する
5 回転関数については2022年度(稻田)のプログラムより引用
6 変位は隣接しているクワッドロータ間でのみ定義
    仮想リーダの進行方向を回転

```

```

7      x1,y1 でリーダ同士の変位を規定
8      x2,y2 で仮想リーダと隣接のリーダの変位を規定
9  %}
10
11 function Delta_i_j = displacement_change(Delta_i_j,x1,y1,x2,y2,Q,lg,
12     Quad_num,loop,Change_num)
13
14 %% 仮想リーダの進行方向を取得する
15 VL_dir = lg(:,Change_num) - Q.Coord(:,loop,Quad_num+1);
16 VL_dir = VL_dir / norm(VL_dir);
17
18 %% 変位の規定
19 %% リーダ 1から見た隣接への変位
20 Delta_i_j(:,2,1) = y1*rot(0,0,1,deg2rad(90))*VL_dir;
21 Delta_i_j(:,3,1) = x1*rot(0,0,1,deg2rad(180))*VL_dir;
22 Delta_i_j(:,5,1) = sqrt(x2^2+y2^2)*rot(0,0,1,acos(x2/sqrt(x2^2+
23     y2^2)))*VL_dir;
24
25 %% リーダ 2から見た隣接への変位
26 Delta_i_j(:,1,2) = y1*rot(0,0,1,deg2rad(-90))*VL_dir;
27 Delta_i_j(:,4,2) = x1*rot(0,0,1,deg2rad(180))*VL_dir;
28 Delta_i_j(:,5,2) = sqrt(x2^2+y2^2)*rot(0,0,1,-acos(x2/sqrt(x2^2+
29     y2^2)))*VL_dir;
30
31 %% リーダ 3から見た隣接への変位
32 Delta_i_j(:,1,3) = x1*rot(0,0,1,deg2rad(0))*VL_dir;
33 Delta_i_j(:,4,3) = y1*rot(0,0,1,deg2rad(90))*VL_dir;
34
35 %% リーダ 4から見た隣接への変位
36 Delta_i_j(:,2,4) = x1*rot(0,0,1,deg2rad(0))*VL_dir;
37 Delta_i_j(:,3,4) = y1*rot(0,0,1,deg2rad(-90))*VL_dir;
38
39 %% 仮想リーダから見た隣接への変位
40 Delta_i_j(:,1,5) = sqrt(x2^2+y2^2)*rot(0,0,1,-pi+acos(x2/sqrt(x2
41     ^2+y2^2)))*VL_dir;
42 Delta_i_j(:,2,5) = sqrt(x2^2+y2^2)*rot(0,0,1,pi-acos(x2/sqrt(x2
     ^2+y2^2)))*VL_dir;
43
44 end

```

ソースコード B.5: チョークポイントを通過できるかを判定する関数

1 %{

```

2 進行方向に障害物が存在するかを判定する関数
3 仮想リーダに隣接するクワッドロータのみ
4 仮想リーダから見た目標地点へのベクトルを傾き,各クワッドロータの
5 座標を通る一点とし,進行方向を直線的に考える
6 その直線と点の距離を求め,その大きさが一定以下である場合は障害物が
7 あると判定する
8 ライダーのデータが2次元であるため1次関数を用いて計算している
9 仮想リーダに隣接する全機体の進行方向に障害物が無い場合のみフラグ
10 が1となる
11 仮想リーダからみた目標地点への傾きで通る一点をリーダにする
12 %}
13
14 function direction_obsflag = direction_obs_detction(obs_Coord,
15 NofVLnumber,Q,loop,Change_num,Quad_num,lg,i,direction_obsflag)
16
17 %閾値
18 threshold = 0.2;
19
20 %}
21 lg(x1,y1):目標地点の座標
22 Q.Coord(:,loop,Quad_num+1)=(x2,y2):仮想リーダの座標
23 Q.Coord(:,loop,NofVLnumber(i))=(x3,y3):仮想リーダに隣接するリ
24 ーダの座標
25 a : x の係数, b : y の係数,c : 定数
26 a=(y2-y1)
27 b=(x1-x2)
28 c=-x3(y2-y1)+y3(x2-x1)
29 %}
30 a = Q.Coord(2,loop,Quad_num+1)-lg(2,Change_num);
31 b = lg(1,Change_num)-Q.Coord(1,loop,Quad_num+1);
32 c = -Q.Coord(1,loop,NofVLnumber(i))*(Q.Coord(2,loop,Quad_num+1)-
33 lg(2,Change_num))+Q.Coord(2,loop,NofVLnumber(i))*(Q.Coord(1,
34 loop,Quad_num+1)-lg(1,Change_num));
35
36 %仮想リーダに隣接している機体のみ
37 for i = 1:length(NofVLnumber)
38
39 %障害物の各座標と直線の距離を計算
40 for j = 1:length(obs_Coord(1,:))

```

```

39         point_distance = abs(a*obs_Coord(1,j)+b*obs_Coord(2,j)+c)
40             / sqrt(a^2+b^2);
41
42         if point_distance > threshold
43             direction_obsflag = direction_obsflag & 1;
44         else
45             direction_obsflag = direction_obsflag & 0;
46         end
47     end
48 end

```

ソースコード B.6: 属性変更時に仮想リーダの位置を変更する関数

```

1 %{
2   属性変更アルゴリズム適用後の仮想リーダの初期座標を規定する
3   仮想リーダに隣接するリーダ間の変位を回転させて規定する
4   フォーメーションによプログラムを変更する必要がある
5 %}
6
7 function Coord = VL_CoordChange(Q,loop,x2,y2)
8   %リーダ1(属性番号5)から見たりーダ2(属性番号6)への単位ベクトル
9   VL_dir = (Q.Coord(:,loop,find(Q.Att == 6))-Q.Coord(:,loop,find(Q
10    .Att==5))) / norm((Q.Coord(:,loop,find(Q.Att==6))-Q.Coord(:,loop,find(Q.Att==5))));
11   %上記ベクトルを回転
12   Delta_i_j = sqrt(x2^2+y2^2)*rot(0,0,1,-(pi/2-acos(x2/sqrt(x2^2+
13    y2^2)))*VL_dir;
14   %絶対座標へ変換
15   Coord = Q.Coord(:,loop,find(Q.Att == 5))+Delta_i_j;
16 end

```

ソースコード B.7: スケーリングベクトルを調整する関数

```

1 %{
2   スケーリングベクトルを調整する関数
3   単純に範囲で場合分けをしている
4 %}
5 function Scaling_vector = Scaling_adjustment(preDistance)
6
7   %preDistance が負であれば拡大, 正であれば縮小
8
9   if preDistance <= -1.8
10      Scaling_vector = 2;

```

```

11    elseif preDistance < -1 && preDistance > -1.8
12        Scaling_vector = 1.8;
13    elseif preDistance >= -1 && preDistance < -0.2
14        Scaling_vector = 1.6;
15    elseif preDistance >= -0.2 && preDistance < 0.2
16        Scaling_vector = 1.0;
17    elseif preDistance >= 0.2 && preDistance < 1
18        Scaling_vector = 0.7;
19    elseif preDistance >= 1 && preDistance < 1.8
20        Scaling_vector = 0.5;
21    elseif preDistance >= 1.8
22        Scaling_vector = 0.3;
23    end
24 end

```

ソースコード B.8: 属性変更に関する関数

```

1  %{
2      リーダーの変更を行う関数
3      各リーダと目標地点の距離を計算してソートする
4      ソートした結果、距離が近い2個のリーダを仮想リーダに隣接させる
5  %}
6
7 function R = ChangeLeader(Q,loop,Quad_num,lg,Change_num,Follower_num)
8
9     distance = zeros(Quad_num-Follower_num);
10    Aircraft_number = zeros(Quad_num-Follower_num);
11
12    %各リーダと目標地点の距離と機体番号を格納
13    for t = 1:Quad_num-Follower_num
14        distance(t) = norm(lg(1:2,Change_num)-Q.Coord(1:2,loop,t+Follower_num));
15        Aircraft_number(t) = t+Follower_num;
16    end
17
18    %各リーダと目標地点の距離を昇順にソート
19    [distance,I] = sort(distance);
20
21    %目標地点との距離が小さい機体2機の機体番号を取得する
22    nearby_aircraft = [Aircraft_number(I(1)),Aircraft_number(I(2))];
23
24    %% 属性番号の決定
25
26    %進行方向がxの正方向

```

```
27 if sum(ismember(nearby_aircraft,[5,6])) == 2
28     R = [1,2,3,4,5,6,7,8,9];
29 %進行方向がy 正方向
30 elseif sum(ismember(nearby_aircraft,[6,8])) == 2
31     R = [3,1,4,2,7,5,8,6,9];
32 %進行方向がx 負方向
33 elseif sum(ismember(nearby_aircraft,[7,8])) == 2
34     R= [4,3,2,1,7,8,6,5,9];
35 %進行方向がy 負方向
36 elseif sum(ismember(nearby_aircraft,[5,7])) == 2
37     R= [2,4,1,3,6,8,5,7,9];
38 end
39 end
```
