석사학위논문

움직임 제약조건과 측정치가 부족한 상황에서의 경사하강법 기반 비순환 최소 퍼시스턴스 그래프를 가진 편대 제어

류인찬

기계공학부

광주과학기술원

2024

Gradient-based Acyclic Minimally Persistent Formation Control with Measurement Deficiency and Motion Constraints

Advisor: Professor Hyo-Sung Ahn

by

In-Chan Ryu

School of Mechanical Engineering

Gwangju Institute of Science and Technology

A dissertation submitted to the faculty of the Gwangju Institute of Science and Technology in partial fulfillment of the requirements for the degree of Master of Science in the School of Mechanical Engineering

Gwangju, Republic of Korea

2024 12, 27

Approved by

Professor Hyo-Sung Ahn

Committee Chair

움직임 제약조건과 측정치가 부족한 상황에서의 경사하강법 기반 비순환 최소 퍼시스턴스 그래프를 가진 편대 제어

류 인 찬

위 논문을 광주과학기술원 석사학위논문으로 인준함

2024년 12월 27일

심사위원장 안효성 (인)

심 사 위 원 최경환 (인)

심 사 위 원 김표진 (인)

To my family.

MS/ME 20221066 In-Chan Ryu. Gradient-based Acyclic Minimally Persistent Formation Control with Measurement Deficiency and Motion Constraints. School of Mechanical Engineering. 2024. 67p. Advisor: Prof. Hyo-Sung Ahn.

Abstract

The study focuses on distributed formation control, a field actively explored in multi-object cooperative control. This control method establishes formations in a global coordinate system based solely on information from each agent's local coordinate system. Particularly in formations where the sensing and actuation topology follow a directed graph, the acyclic persistent graph is widely employed, ensuring convergence through various formation control theories such as gradient descent techniques.

Nevertheless, challenges arise when forming a formation with a ground-dependent wheel mobile robot and a drone with an additional degree of freedom perpendicular to the ground. The gradient descent technique exhibits limitations, generating control inputs impractical for the wheel mobile robot. Furthermore, existing formation controllers pose difficulties when applied to agents with non-holonomic constraints, assuming a single integrator model for ease of deployment. Geometric constraints may also emerge based on the attached sensor type, making convergence challenging. Additionally, if complete sensing information required for the gradient descent controller is unavailable, an estimation algorithm must be devised.

This paper addresses gradient descent-based triangle formation control in scenarios involving heterogeneous robots with motion and visibility constraints and insufficient sensing information. The absence of sensing information regarding the locations of other agents is addressed through a distance-based gradient descent adaptation law. Even in cases where a basic gradient descent formation controller fails to achieve the control goal without violating constraints, assurance of convergence is provided by introducing a mode algorithm. This algorithm effectively partitions and applies inputs generated from information obtained from each agent so as not to violate constraints and achieve control goals. The proposed algorithm's efficacy was validated through simulations and experiments.

MS/ME 20221066
류 인 찬. 움직임 제약조건과 측정치가 부족한 상황에서의 경사하강법 기반 비순환 최소 퍼시스턴스 그래프를 가진 편대 제어. 기계공학부. 2024. 67p. 지도교수: 안 효 성.

국 문 요 약

에이전트의 지역 좌표계만 정보만으로 전역 좌표계에서 편대를 이루는 분산적 편대 제어는 다개체 협력 제어에서 활발하게 연구되고 있는 분야이다. 센싱 및 구동 토폴로지가 방향 그래프를 갖는 편대 중 비순환 퍼시스턴트 그래프는 경사하강 기법등 다양한 편대 제어 이론으로 그 수렴성이 보장되어 있다.

하지만 지면에 종속된 휠 모바일 로봇과 지면에 수직인 축의 자유도를 추가로 갖는 드론이 편대를 이룰 경우 경사하강 기법은 휠 모바일 로봇이 수행할 수 없는 제어 입력을 산출하는 한계가 존재한다. 또한 많은 편대 제어기는 전개의 편의상 에이전트의 모델을 단일 적분기로 가정하기 때문에 논홀로노믹 제약조건을 갖는 에이전트에 가용하기 어렵다는 단점이 있다. 많은 경우 에이전트에 부착되는 센서의 종류에 따라 추가적인 기하 제약조건이 발생하는데 이런 경우에 수렴성을 보장하기 어렵다. 추가적으로 경사하강 제어기에 필요한 센싱 정보를 모두 획득하지 못하는 경우에는 이를 위한 추정 알고리듬이 고안되어야 한다.

본 논문에서는 움직임 제약조건과 가시성 제약조건가지는 이종 로봇이 센싱 정보가 부족한 상황에서 경사 하강 기반의 삼각 편대 제어에 대해서 다룬다. 부족한 센싱 정보인 타 에이전트의 위치를 거리기반 경사하강 적응 법칙을 이용하여 추정한다. 또한 모든 제약조건을 위배하지 않으면서 제어 목표를 만족하지 못하는 상황에서 에이전트 별 입력으로 나누어 인가하는 방식의 모드 알고리듬을 제안하여 수렴성을 보장한다. 물리 엔진 기반 시뮬레이션과 실제 하드웨어를 통한 실험을 통해 알고리듬이 검증되었다.

목 차

표 목 차

그림목차

제 1장

서문

편대제어는 다개체 협력제어 중 활발하게 연구되고 있는 분야 중 하나이다. 에이전트의 지역 좌표계에서 획득되는 정보만을 가지고 전역 좌표계에서 일정한 편대를 이루는 분산적 편대제어는 실제 어플리케이션에서 다양하게 응용될 수 있는 분야이다. 실제 상황에서는 GPS와 나침판 센서(Compass)을 사용하여 전역 좌표계를 공유하지 못하는 상황이 빈번하고 이러한 고정밀 항법 센서는 매우 고갸이기 때문이다. 분산적 편대제어에서 무향 그래프를 가지며 양방향 통신을 하게되면 단방향 통신보다 통신이나 시스템 복잡도가 2배가 되며, 더욱이 편대를 이룰 시 센서 편향(bias)에 따라 추가적인 문제가 발생할 수 있다. 따라서 본 논문에서는 방향 그래프를 가지는 편대 중 비순환 최소 퍼시스턴스 그래프에 대해서 다룬다.

비순환 퍼시스턴스 그래프는 편대 제어에서 자주 언급되는 토폴로지 형태로서 제어가 용이하고 그 분석이 간단하다. 하지만 편대 제어가 쉬운 그래프에서도 실제 발생할 수 있는 에이전트의 운동학적 제약조건이나 센서의 기하적 제약조건이 포함되게 되면 그 수렴성을 보장하기 어렵다. 특히나 실제 상황에서는 편대제어를 위한 모든 센싱 변수들이 관측하기 어렵다. 그렇기 위해서는 다수의 센서가 부착되거나 센서 후 처리 과정에서 많은 연산이 소요되기 때문이다.

에이전트의 운동학적 제약조건을 해결하기 위한 시도는 다양하게 존재해왔다. 이는 단순히 되먹임 선형화를 통해 에이전트를 단일 적분기 모델로 만드는 시도부터 [1], 제어

입력의 적절한 변환을 통해 논홀로노믹 에이전트의 수렴성을 보장하기도 하였다 [2]. 부착되는 센서의 종류에 대한 제약조건들을 반영한 사례도 존재하나 대부분 리더가 일정한 속도로 이동하면서 일자 혹은 V자로 트레킹하는 것일 뿐 특정한 모양의 편대를 이루는 것이 아닌 경우가 많다 [3] [4]. 이러한 다양한 제약조건들을 하나의 알고리듬을 구현하고 검증한 사례는 흔치 않다.

따라서 본 논문에서는 대표적인 운동학적 제약조건인 논홀로노믹 에이전트와 포화조건을 고려한다. 센서의 제약조건으로는 카메라와 같은 센서처럼 관측을 위한 기하학적 조건을 만족해야하는 가시적 제약조건을 고려한다. 이에 더해 하나의 센싱 변수가 부족한 상황에서 거리 정보만을 가지고 거리와 각도 정보를 적응 법칙을 통해 추정하는 기법을 소개한다. 이러한 추정 알고리듬과 편대제어 알고리듬을 연결하여 결과적으로 원하는 편대를 이룰 수 있음을 보장한다.

제 2장

배경

2.1 그래프 이론

다개체 시스템에서 그래프 이론은 에이전트간의 관계를 묘사하기 위해 사용된다. 에이전트 사이의 센싱, 통신, 구동은 내재적으로 그래프 이론의 성격을 띄고 있기 때문에 그래프 이론으로 간편하게 모델링 될 수 있다.

그래프는 꼭지점(에이전트)과 간선으로 구성되어 있고 이는 다개체 시스템에서 각각 에이전트와 에이전트 사이의 관계를 표현한다. 간선 집합 $\mathcal{E} = \{\dots, (i,j), \dots\} \in \mathcal{V} \times \mathcal{V}$와 꼭지점 집합 $\mathcal{V} = \{1, \dots, N\}$을 포함하는 그래프 $\mathcal{G}$를 $\mathcal{G}(\mathcal{E}, \mathcal{V})$로 표현한다.

그래프는 방향의 특성에 따라 방향이 없는 무향 그래프(undirected)와 방향이 있는 방향 그래프(directed)로 분류된다. 편대제어 관점에서 유향 그래프는 무향 그래프에 비해 몇가지 장점을 가진다. 유향 그래프는 에이전트간 주고 받는 정보가 무향 그래프에 비해 절반이다. 무향 그래프는 에이전트가 서로 정보를 주고 받아야 하지만 유향 그래프는 한 에이전트만 정보를 획득하면 되기 때문이다. 같은 맥락으로 통신 복잡도와 전체적인 제어 복잡도가 무향 그래프의 절반이다. 또한 무향 그래프에서는 상대거리를 제어하는 역할을 두 에이전트가 담당하기 때문에 발생하는 문제들이 있다 [5]. 반면 유향 그래프는 이와같은 문제에서 자유롭다.

2.1.1 실현과 프레임워크

그래프는 본디 꼭지점과 꼭지점간의 추상적인 관계를 표현한다. 그래프가 편대제어에 사용될 경우 꼭지점과 간선은 물리적 의미를 부여받게 된다. 각 꼭지점은 유클리디언 공간에서 각 에이전트의 위치를 나타내게 되며, 따라서 n차원 공간에서 에이전트의 위치는 $\mathbf{p}_i \in \mathbb{R}^n$로 표현할 수 있다. 이러한 위치 벡터를 쌓아 만든 벡터의 좌표 $\mathbf{p} = [\mathbf{p}_1^T \dots \mathbf{p}_n^T]^T \in \mathbb{R}^{n \times N}$를 그래프의 실현(Realization)이라고 하고 그래프와 실현의 조합인 $(\mathcal{G}, \mathbf{p})$를 프레임워크(framework)라고 한다. 따라서 공간상에서 표현되는 편대는 프레임워크로 표현될 수 있다.

주어진 두 프레임워크 $(\mathcal{G}, \mathbf{p})$와 $(\mathcal{G}, \mathbf{q})$는 $||\mathbf{p}_i - \mathbf{p}_j|| = ||\mathbf{q}_i - \mathbf{q}_j||, \quad \forall (i,j) \in \mathcal{E}$이면 서로 동등하다(equivalent)고 하며 두 실현 $\mathbf{p}$와 $\mathbf{q}$가 $||\mathbf{p}_i - \mathbf{p}_j|| = ||\mathbf{q}_i - \mathbf{q}_j|| \quad \forall (i,j) \in \mathcal{V}$을 만족한다면 두 실현은 합동(congruent)라고 한다. 즉, 프레임 워크간 간선에 대한 길이가 동일하다면 두 프레임워크는 동등하고 꼭지점간의 모든 거리가 같다면 두 프레임워크는 합동이다.

2.1.2 모호성 문제와 무향 그래프의 단단함 성질

원하는 편대의 프레임워크가 주어졌다면, 이러한 프레임워크의 실현이 유일한지 확인할 필요가 있다. 만약 유일하지 않다면 주어진 프레임워크를 만족시키는 제어기를 설계했다 하더라도 기대했던 것과 다른 편대를 이룰 수 있기 때문이다.

전역 좌표계에서의 정보를 획득할 수 있는 상황이라면 주어진 프레임워크와 동일한 실현을 얻을 수 있다. 전역 좌표계상에서 표현된 위치에 해당 에이전트가 위치하면 되기 때문이다.

하지만 전역좌표계에 대한 정보가 가용하지 않고, 각 에이전트의 로컬 좌표계에서 얻

은 정보만을 가지고 편대를 이루어야 한다면 주어진 프레임워크의 유일성을 확인하는 것은 필수적이다.

유일하지 않은 프레임워크의 예로는 변의 길이가 고정되었음에도 연속적으로 특정 노드가 움직일 수 있는 유연한 프레임워크 (flexible framework)와 플립(flip)과 같이 불연속적으로 다른 실현을 갖는 프레임워크가 있다.

무향 그래프를 갖는 프레임워크가 유일한 실현을 갖는 성질을 단단함(rigid)이라고한다. 이러한 성질을 가지는 프레임워크에는 연속적으로 변형되지 않는 단단한 프레임워크(rigid framework)와 연속적 변형 뿐만 아니라 불연속적인 다른 실현을 갖지 않는, 즉 유일한 모양을 갖는 프레임워크인 전역적 단단한 (globally rigid) 프레임워크가 있다.

만약 이미 편대를 이룬상태에서 이를 유지하기만 하면 된다면 단단한 프레임워크만으로 충분하다. 하지만 임의의 위치에서 특정 편대를 이루고자 한다면 전역적 단단한 프레임워크가 필수적이다. 단단한 프레임워크와 전역적 단단한 프레임워크의 수학적 엄밀한 정의는 다음에 기술되어 있다 [6, 7].

원하는 프레임워크가 단단함 성질을 가지는지 분석하는 도구에는 단단함 행렬(rigidity matrix) 기법, Laman's Theorem을 기반으로하는 기법, 그리고 Henneberg construction을 이용하는 방법이 있다 [8]. 세 방법 모두 2차원에서는 그래프의 단단함 성질을 가지는지 쉽게 파악하지만 3차원 그래프에 대해서는 필요 조건과 충분조건만을 가지거나 추측만 가질 뿐 2차원에서 적용되던 방법이 손쉽게 확장되지 못한다. 이는 전역적 단단함을 분석할때도 동일하게 적용된다.

2.1.3 방향 그래프의 퍼시스턴스 성질

방향 그래프를 갖는 프레임워크가 유일한 실현을 갖기 위한 성질을 퍼시스턴스라고 한다 [9] [10]. 무향 그래프에서는 에이전트간 주어진 길이를 만들기 위한 책임이 양측 에이전트 모두에게 있다. 반면 유향그래프에서는 한 에이전트만이 그 변의 길이를 만들 책임을 가진다. 방향 그래프에서 한 에이전트는 다른 에이전트들이 가지는 거리 조건을 고려하지 않아 무향 그래프의 에이전트에 비해 추가적인 자유도가 생긴다. 이 때문에 특정 에이전트는 주어진 거리조건을 만족하는 것이 불가능한 상황이 발생할 수 있다. 따라서 방향 그래프에서는 무향 그래프의 단단함 조건에서 일관성 제약(constraint consistence)라는 조건이 추가된 퍼시스턴스라는 개념이 도입된다. 모든 에이전트가 에이전트간 주어진 거리 조건을 유지하는게 가능하다면 이를 일관성 제약을 만족했다고 한다. 즉 일관성 제약이 만족된 그래프에서는 불가능한 상대거리 조건을 가진 에이전트가 존재하지 않는다. 퍼시스턴스에 대한 수학적 엄밀한 정의는 [9]에 기술되어 있다.

반례가 존재하긴 하지만 일관성 제약조건을 검사하는 실용적안 방법은 2차원 무향 그래프의 모든 꼭지점에서 2개 이상의 진출간선(outgoing-edge)이 존재하는지 여부를 확인하는 것이다. 그러한 간선이 존재하지 않는다면 일관성 제약조건을 만족하는 그래프이다. 주어진 그래프가 퍼시스턴스 그래프인지 판별하는 퍼시스턴스 검사(persistent testing) 방법은 2차원의 경우 [9] 에 3차원의 경우 [10]에 기술되어 있다.

2.1.4 비순환 최소 퍼시스턴스 편대

본 논문의 그래프가 속하는 범주인 비순환 최소 퍼시스턴스 그래프(acyclic minimally persistent graph)에 대해서 알아보도록 하자. 한 꼭지점에서 진출 간선을 따라 다시 동일한 꼭지점으로 돌아오는 길(path)이 존재하지 않는 그래프를 비 순환(acyclic) 그래프라고

한다. 만약 주어진 그래프가 다수의 순환을 가진다면 특정 차원공간에서 편대가 어떻게 실현될지 분명하지 않다 [11]. 그렇기 때문에 방향 그래프를 가지는 편대 제어 문제의 경우 유일한 실현을 기대할 수 있고 직관적으로 제어문제를 이해가 쉬운 비순환 편대를 고려하는게 일반적이다. 즉, 비순환 퍼시스턴스 그래프가 유일한 실현을 기대하는 편대제어에서 자주 사용되는 그래프이다.

최소 퍼시스턴스(minimally persistent) 그래프란 퍼시스턴스 성질을 잃지 않으면서 제거할 수 있는 간선이 존재하지 않는 그래프를 뜻한다. 즉, 퍼시스턴스를 유지하기 위한 최소의 간선들로 구성된 그래프이다. 2차원에서 하나 이상의 꼭지점을 갖는 비순환 최소 퍼시스턴스 그래프는 다음과 같이 표현할 수 있다 [9].

- 진출차수(out-degree)가 0인 꼭지점이 하나 존재하며 이를 리더라고 부른다.
- 진출차수가 1이고 이 간선은 리더를 향해 하나 존재하며 이러한 에이전트를 첫번째 팔로워(first follower)라고 한다.
- 나머지 모든 꼭지점은 진출차수 2를 가진다.

제 3장

제안된 알고리듬

3.1 문제 정의

본 논문에서 이루고자 하는 편대는 그림 3.1과 같다. 드론(drone)를 리더로 하고 이를 2휠 구동 모바일 로봇(2 Wheel Mobile Robot) 2대가 팔로워가 되어 삼각편대를 이룬다. 이때 모바일 로봇1이 팔로워1, 모바일 로봇2가 팔로워2의 역할을 수행한다. 모바일 로봇은 카메라와 같은 피동센서를 통해 리더를 관측하고 자신과의 상대위치를 측정한다. 팔로워2는 추가적인 센서(UWB: Ultra Wide Band)가 부착되어 팔로워1과 자신과의 상대거리를 측정한다.

그림 3.1의 센싱 및 구동 토폴로지(sensing and actuation topology) 그래프는 그림3.2와 같다. 따라서 3.2는 $\mathcal{V} = \{1, 2, 3\}$,이고 $\mathcal{E} = \{(1, 2), (1, 3), (2, 3)\}$인 그래프 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$로 표현된다. 여기서 꼭지점 1은 리더, 꼭지점 2는 팔로워1, 그리고 꼭지점 3은 팔로워2와 짝을 이룬다. 그림 3.2 형태의 그래프는 2.1.4에서 소개한 2차원 공간에서의 비순환 최소 퍼시스턴스 그래프로 분류된다.

드론이 모바일 로봇에 비해 지면에 수직한 방향의 자유도를 추가로 갖기 때문에 주어진 편대는 3차원 공간에 속한다. 3차원에서 방향 그래프가 유일한 실현을 갖기 위해서는 구조적 퍼시스턴스(structural persistence)를 만족해야한다 [10]. 이를 위해서는 최소 4개의 에이전트가 필요하다. 주어진 문제에서 에이전트 수는 3대 뿐이기 때문에 3차원 편대는 모호성이 존재한다. 하지만, 팔로워가 지역 좌표계에서 리더의 상대위치를 측정할 수 있

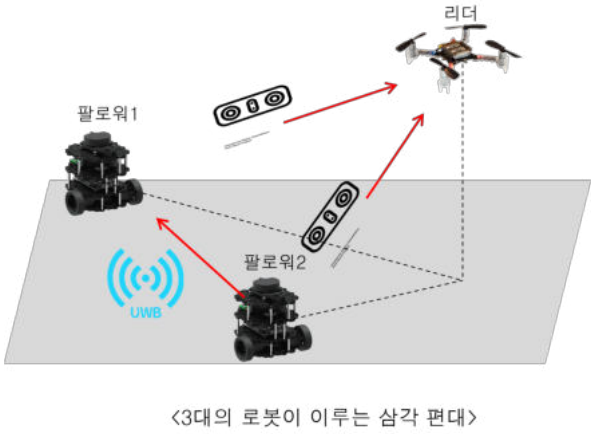


그림 3.1. 드론과 모바일 로봇2대의 삼각 편대

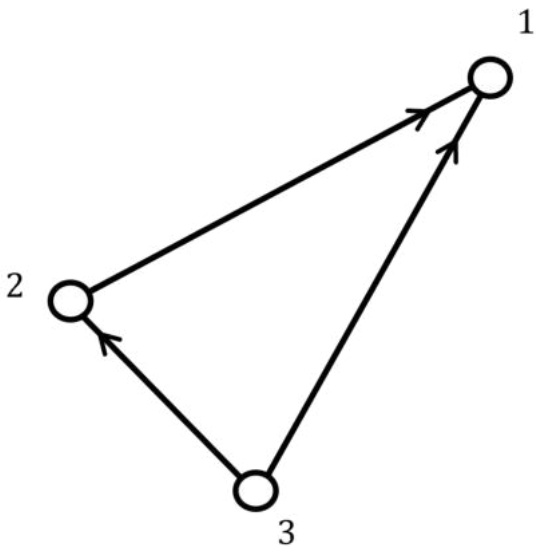


그림 3.2. 삼각 편대의 비순환 최소 퍼시스턴스 그래프

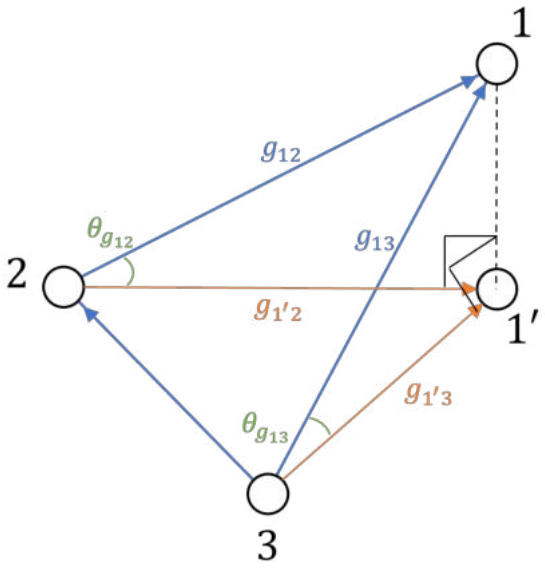


그림 3.3. 리더를 지면에 정사영 시켜 얻은 2차원 그래프

다면, 그림 3.3와 같이 리더(꼭지점 1)를 지면으로 정사영시켜 꼭지점 $1'$ 을 얻고 2차원 삼각편대 $\triangle 1'23$ 를 얻을 수 있다. 그림 3.3에서 g_{ij}는 꼭지점 i와 꼭지점 j사이의 거리를 의미하고 $\theta_{g_{ij}}$는 g_{ij}와 지면이 이루는 각도를 의미한다.

2차원으로 축소된 편대는 3차원에서의 편대와 다음의 직각 삼각형의 기하학적 변환관계가 성립한다.

$$g_{1'2} = g_{12} \cos\theta_{g_{12}} \tag{3.1}$$

여기서 $0 \leq \theta_{g_{12}} < \pi/2$ 이고 팔로워 i가 리더의 상대위치를 측정할 수 있다면 $g_{1i}, \cos\theta_{g_{1i}}$ 또한 계산할 수 있다. 식 3.1을 통해 변환된 2차원 편대를 이루면 해당하는 3차원 편대를 이룰 수 있다. 이 방법을 통해 본 논문에서는 3대의 에이전트의 3차원 편대의 모호성 문제를 2차원으로 축소시켜 회피한다. 2휠 구동 모바일 로봇은 3차원에서 바닥평면에 고정된 홀로노믹 제약조건(holonomic constraint)을 가진다. 경사하강 기반의 편대제어기를 사용할 경우 홀로노믹 제약조건에 위배되는 제어 입력을 산출하게 된다. 하지만 2차원으로 축소시키면 이러한 문제를 회피할 수 있다.

3.1.1 에이전트의 운동학적 제약조건

2.1.1에서 실현과 프레임워크에 대해서 다루면서 노드가 데카르트 좌표계 위의 위치 벡터로 표현될 수 있다고 언급한바 있다. 3차원 문제를 2차원 문제로 변환하였으니, i번째 에이전트의 위치는 $\mathbf{p}_i \in \mathbb{R}^2$로 나타낼 수 있다. 주어진 편대 그래프에서 $(i,j) \in \mathcal{E}$는 에이전트 j가 에이전트 i와의 주어진 상대거리를 만족시킬 의무(responsibility)가 있다는 것을 의미한다.

편대제어 문제에서 에이전트의 움직임은 해석의 용의성을 위해 흔히 단일 적분기로 묘사된다 . i번째 에이전트의 운동은

$$\dot{\mathbf{p}}_i = \mathbf{u}_i, \quad \forall i \in \mathcal{V} \tag{3.2}$$

여기서 $\mathbf{u}_i \in \mathbb{R}^2$는 i번째 에이전트에 인가되는 입력이고 $\mathbf{p} = [x_i, y_i]^T$이다.. 위의 가정은 리더인 드론에 대해서는 만족하지만 팔로워인 2휠 구동 모바일 로봇에게는 유효하지 않다. 2휠 구동 모바일 로봇은 논홀로노믹 제약조건(nonholonomic constraint)을 가진 대표적인 에이전트이기 때문이다. 따라서 에이전트 운동 모델은 단순 단일 적분기가 아니라 논홀로노믹 모델이여야 한다.

논홀로노믹 에이전트의 대표적인 예로는 외륜 모델(unicycle model)이나 자동차와 같은 차륜 모델(car-like vehicle model)등 이 있다. 이러한 논홀로노믹 제약조건을 가지는 에이전트들은 특정 상태에 도달하는데 가질 수 있는 상태가 경로에 따라 달라지기 때문에 운동학적 제어가 까다롭다는 특징이 있다. 본 논문의 2휠 구동 모바일 로봇은 그림 3.4와 같은 외륜모델로 분류될 수 있다. 그림 3.4의 ${}^{g}\sum$은 전역 좌표계를 의미하고 그 축인 X_g, Y_g은 데카르트 좌표계로 표현된다. 이때 각 에이전트는 자신의 지역 좌표계 ${}^{i}\sum$ 을 가지며 자신의 기하 중심(x_i, y_i)을 원점으로 하고 자신의 헤딩방향을 X_i축, 이에 반시계방

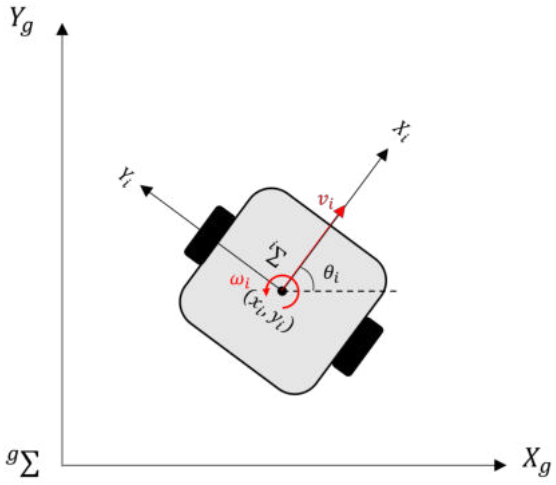


그림 3.4. 외륜 모델의 기하

향으로 수직한 축을 Y_i축으로 가진다. 이때 $^g\sum$과 $^i\sum$은 2차원 평면에 수직한 방향으로 θ_i 만큼의 회전변환 관계를 가진다. 외륜모델의 입력은 헤딩방향 선속도 v_i와 $^i\sum$의 중심을 지나고 바닥평면에 수직한 축으로 회전하는 각속도 ω_i로 이루어져 있다.

대표적인 외륜 모델의 운동 방정식을 $^g\sum$에서 표현하면

$$\dot{\mathbf{x}}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} cos(\theta_i) & 0 \\ sin(\theta_i) & 0 \\ 0 & 1 \end{bmatrix} \eta_i \tag{3.3}$$

여기서 $\eta_i = [v_i \ \omega_i]^T$. 논홀로노믹 에이전트는 단일 적분기 모델에서 자신의 헤딩각도에 대한 다이나믹스가 추가되었다. 단일 적분기의 입력은 위치의 미분에 바로 인가되는 반면 논홀로노믹 에이전트의 입력은 속도와 각속도가 비선형 변환되어 인가된다. 논홀로노믹 에이전트는 속도와 각속도 명령이 인가되면 부착된 두개의 모터가 차동 구동(differential drive)되어 각 바퀴가 회전하게 된다. 일반적으로 에이전트는 최대 속도와 각속도가 존재하는 포화(saturation)조건을 포함한다. 물리적인 한계 때문에 속도나 각속도가 무한정 증가할 수 없기 때문이다. 따라서 논홀로노믹 에이전트의 입력에는 아래와 같은 범위를

가진다.

$$\begin{aligned} -V_{SAT} \leq & v_i \leq V_{SAT} \\ -W_{SAT} \leq & \omega_i \leq W_{SAT} \end{aligned} \tag{3.4}$$

여기서 V_{SAT}, W_{SAT}은 에이전트의 최대출력인 최대 속도와 최대 각속도이다.

3.1.2 피동센서의 관측조건

팔로워가 리더를 관측하기 위해서는 피동 센서의 계측 범위안에 리더가 위치해야 함을 의미한다. 그렇지 않으면 팔로워는 리더의 상대 위치 정보를 얻을 수 없다. 따라서 팔로워와 리더의 상대 각도가 일정범위 내에 위치해야한다는 제약조건이 추가되며 이는 부등호 형태의 제약조건으로 나타나는데 이를 가시성 부등호 제약조건(visibility inequality constraint)이라 한다. 그림 3.5은 리더 i를 관측하는 팔로워 j를 도시한 것이다. 팔로워는 피동센서의 화각(field of view)을 의미하는 옅은 파란색 범위 내에서만 리더를 관측 할 수 있다. 리더와 팔로워 간의 시선각(line-of-sight)과 팔로워의 헤딩각도 θ_j의 차이의 절댓값이 화각 범위 내에 존재해야 한다. 이는 ${}^{i}\sum$의 X_i와 ${}^{i}\sum$에서 관측된 리더와의 상대각도 $\Delta\theta_{ij}$에 해당한다.

$$\left\| \arctan(\frac{y_i - y_j}{x_i - x_j}) - \theta_j \right\| = \Delta\theta_i j \leq \theta_{FOV} \tag{3.5}$$

3.1.3 삼각편대의 기하적 조건

그림 3.2에 도시된 삼각 편대를 형성하기 위해서는 원하는 편대가 삼각형의 기하학적인 조건을 만족해야 한다. 따라서 삼각편대의 원하는 실현(desired realization)은 그 어떤

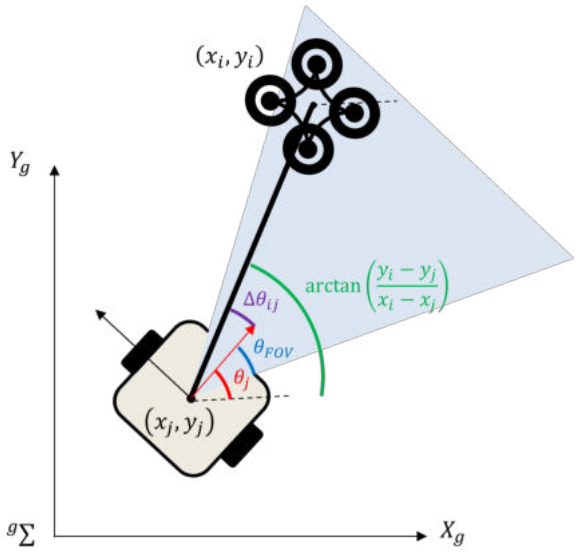


그림 3.5. 리더 관측을 위한 가시성 부등호 제약조건

변의 길이도 0이 아니며 아래의 삼각부등식을 만족해야한다.

$$
\begin{aligned}
||\mathbf{p}_1^* - \mathbf{p}_2^*|| &< ||\mathbf{p}_1^* - \mathbf{p}_3^*|| + ||\mathbf{p}_2^* - \mathbf{p}_3^*|| \\
||\mathbf{p}_1^* - \mathbf{p}_3^*|| &< ||\mathbf{p}_1^* - \mathbf{p}_2^*|| + ||\mathbf{p}_2^* - \mathbf{p}_3^*|| \\
||\mathbf{p}_2^* - \mathbf{p}_3^*|| &< ||\mathbf{p}_1^* - \mathbf{p}_2^*|| + ||\mathbf{p}_1^* - \mathbf{p}_3^*||
\end{aligned}
$$

3.1.4 편대 제어 문제 정의

편의를 위해 상대 거리 벡터를 다음과 같이 재정의 해보자. 상대거리는 각 에이전트의 차벡터의 크기에 해당한다.

$$\mathbf{z}_{ij} = \mathbf{p}_i(t) - \mathbf{p}_j(t), \quad \forall (i,j) \in \mathcal{V} \tag{3.6}$$

위의 조건들을 고려한 편대제어 문제는 다음과 같이 기술된다.

Problem 3.1. *그림 3.2과 같이 주어진 편대 그래프와 식 (3.3)의 운동도 식 3.5의 센싱 제약조건을 가지는 에이전트가 존재할때, 삼각편대를 이루는 원하는 실현 $\mathbf{p}^* \in \mathcal{R}^{2|\mathcal{V}|}$에 대하여 에이전트의 지역좌표계에서 획득된 상대거리 벡터 정보 (3.6)만을 가지고 $\mathbf{p}(t)$가 $\mathbf{p}^*$에 합동인 점으로 수렴하게 하는 분산제어기를 설계하라.*

위의 제어 목적을 수식으로 표현하면 아래와 같다.

$$\lim_{t\to\infty} ||\mathbf{p}_i(t) - \mathbf{p}_j(t)|| = ||\mathbf{p}_i^* - \mathbf{p}_j^*||, \quad \forall i,j \in \mathcal{V}.$$

원하는 편대는 원하는 실현 $\mathbf{p}^*$과 합동인 모든 실현들의 집합으로 표현할 수 있다.

$$\mathcal{D}_{\mathbf{p}^*} = \{\mathbf{p} \in \mathcal{R}^{2|\mathcal{V}|} : ||\mathbf{p}_i - \mathbf{p}_j|| = ||\mathbf{p}_i^* - \mathbf{p}_j^*||, \quad \forall i,j \in \mathcal{E}\}$$

따라서 설계한 분산제어기의 목적은 초기 실현 $\mathbf{p}_0$이 $\mathcal{D}_{\mathbf{p}^*}$에 속한 실현으로 수렴하게 하는 제어입력을 산출하는 것이다. 합동은 2.1.1에서 다뤘듯 모든 꼭지점에 대해서 조사해야하지만 삼각편대의 기하적 특성상 모든 변에 대해서 조사하는 것으로 합동의 정의를 만족할 수 있다.

3.2 모드 알고리듬

본 부분에서는 기존의 경사하강 기반 제어기의 아이디어를 채용해 논홀로노믹 에이전트와 포화조건, 그리고 가시성 제약조건이 존재하는 상황에서 수렴하는 제어 알고리듬과 모드 알고리듬을 제시한다. 부족한 측정치에 대한 추정문제는 다음 부분에서 소개한다. 본 부분에서는 측정치 추정이 완료되었다고 가정하고 전개한다.

3.2.1 경사하강 기반 제어기의 아이디어

기존의 비순환 최소 퍼시스턴스 그래프는 경사하강 제어기의 문제로 다루어졌다. N개의 에이전트의 비순환 최소 퍼시스턴스 편대에 대해서는 [12, 13]에서 소개되었다. [14]에서 3대의 에이전트로 이루어진 비순환 편대의 초기위치가 일적선(colinear)이 아니라면 지수적으로 수렴한다고 증명하였다. [15]에는 일적선 초기조건에서의 수렴성을 보장하기 위해 비최급경사 방법(non-steepest descent method)을 제시하였다. 일반적으로 사용되는 최급경사 방향이 아닌 방향으로 입력을 인가하여 일직선 초기상태에서의 수렴성을 보장한다. 하지만 이 방법은 일적선 초기조건에 존재하는 원치않는 불변 부분집합(undesired

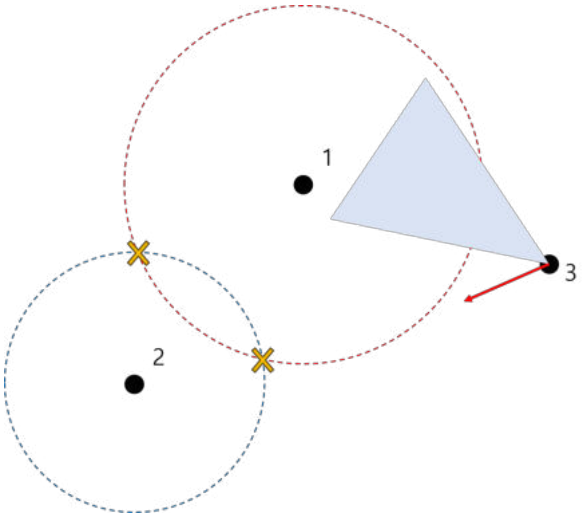


그림 3.6. 가시성 제약조건을 만족하지 못하는 최급경사하강 입력의 예

invariant subset)을 변환한 것일 뿐 그것은 여전히 존재한다. 이러한 불변 부분집합은 시스템의 특성이기 때문에 제거하기 어렵다.

기존의 기법들은 에이전트를 단일 적분기로 가정하고 수렴성을 증명하였다. 이러한 단점을 보완하기 위해 논홀로노믹 에이전트의 경사하강 기법에 대한 변형된 알고리듬이 제시되었다 [2]. 하지만 이 기법은 무향 그래프의 편대제어에 대해서 제시되었으며 에이전트가 경사하강 입력의 방향을 맞추기 위해 에이전트의 헤딩방향을 회전시켜야 한다는 운동학적 특징을 제거하지는 못하며 그 때문에 가시정 제약조건을 만족하지 못할 가능성이 크다. 그렇기 때문에 본 논문의 문제정의에 적용하였을때 에이전트가 수렴하지 못하는 것을 직관적으로 알 수 있다. 가시성 조건을 만족하기 위해 리더가 탐지되지 않았을 때 리더는 찾는 입력을 추가하여 리더를 찾았다고 해보자. 그림 3.6은 기존 기법으로 수렴불가능한 상황 중 하나이다.

그림 3.6에서 1은 리더, 2는 팔로워 1, 3은 팔로워 2를 의미한다. 이때 리더는 움직이지 않고 팔로워1는 이미 수렴구간에 도달했다고 해보자. 이때 빨간 점선은 리더와 팔로워2의 원하는 상대거리 $||\mathbf{z}_{13}^*||$ 집합을 파란색 점선은 팔로워1과 팔로워2간의 원하는 상대거리

$||\mathbf{z}_{23}^*||$ 집합을 의미한다. 이때 팔로워2의 최종 평형 집합은 두 원이 서로 만나는 두 점이 된다. 이때 팔로워2는 리더와의 거리와 팔로워1과의 거리를 모두 만족해야 하며 빨간 화살표의 방향과 크기는 평형집합에 도달하기 위한 경사하강 제어입력이다. [2]에서 제시한 변형된 알고리듬은 이러한 경사하강의 방향에 논홀로노믹 에이전트의 헤딩방향을 일치시키는데 이 방향이 가시성 제약조건을 만족시키는 방향이 아니기 때문이다.

따라서 본 논문에서는 기존의 경사하강 기법의 아이이더와 그 수렴 특징에 기반하여 새로운 알고리듬적 수렴성을 제시한다.

3.2.2 경사하강 기법

삼각편대 문제 3.1를 제곱의 거리 오차의 제곱으로 이루어진 목적함수를 최소화하는 형태의 제어입력을 생각해볼 수 있다. 시간에 따른 각 변에 대한 제곱의 거리 오차는

$$e_{ij}(t) = ||\mathbf{p}_i(t) - \mathbf{p}_j(t)||^2 - ||\mathbf{p}_i^* - \mathbf{p}_j^*||^2, \quad \forall i,j \in \mathcal{E}$$

팔로워1은 리더와의 변에 대한 책임만 있으므로 목적함수는

$$\phi_2(\mathbf{p}(t)) = k_{\phi_2} e_{12}^2 \tag{3.7}$$

이때 k_{ϕ_2}는 0보다큰 상수이다.

반면 팔로워2는 리더와의 변과 팔로워1과의 변에 책임이 있으므로 목적함수는 각 변에 대한 제곱의 거리오차의 제곱의 합으로 정의된다.

$$\phi_3(\mathbf{p}(t)) = k_{\phi_3} e_{13}^2 + k_{\phi_3} e_{23}^2 \tag{3.8}$$

이때 k_{ϕ_3}는 0보다큰 상수이다.

목적함수의 미분시 얻어지는 계수들을 고려하여 목적함수에 포함되는 상수 k_{ϕ_2}, k_{ϕ_3}은 $1/4$로 설정하는 경우가 많다. 미분시에 상수 부분을 깔끔하게 1로 만들기 위함이다. 이는 단순 거리 오차의 제곱으로 목적함수를 구성하지 않고 제곱의 거리 오차의 제곱으로

표현한 이유도 목적함수의 미분에 대한 값을 단순화 하기 위함이다. 목적함수 미분 항이 단순해야 시스템을 해석할때 매우 유용하다.

$\phi_i(\mathbf{p}(t)) \rightarrow 0$ 를 만족하는 분산 제어기를 설계한다면 원하는 변의 길이를 만족하는 것이고 이는 원하는 편대와 합동이다.

비순환 퍼시스턴스 그래프를 가지는 편대를 제어하기 위한 경사하강 제어입력은

$$\mathbf{u}_i = -\left[\frac{\partial \phi_i}{\partial \mathbf{p}_i}\right]^T, \quad \forall i \in \mathcal{V}, \tag{3.9}$$

먼저 단일 적분기로 묘사된 에이전트가 입력받는 최급경사 제어입력을 살펴보자. 먼저 리더는 그 어떤 간선에 대한 책임도 없기 때문에 감소해야할 목적함수가 존재하지 않는다. 따라서 편대를 이루기 위한 경사하강 제어 입력은 0이다.

팔로워 1은 리더와 자신과의 간선에 대한 책임이 있다. 목적함수 3.7를 제어 법칙인 3.9에 대입하면

$$\begin{aligned}
\phi_2(p(t)) &= \frac{1}{4}(e_{12})^2 \\
&= \frac{1}{4}(||\mathbf{p}_1(t) - \mathbf{p}_2(t)||^2 - ||\mathbf{p}_1^* - \mathbf{p}_2^*||^2)^2 \\
&= \frac{1}{4}(||\mathbf{z}_{12}||^2 - ||\mathbf{z}_{12}^*||^2)^2 \\
\frac{\partial \phi_2}{\partial p_2} &= -e_{12}\mathbf{z}_{12}
\end{aligned}$$

여기서, $||\mathbf{z}_{12}^*|| = ||\mathbf{p}_1^* - \mathbf{p}_2^*||$.

팔로워 2는 리더와 자신 그리고 팔로워1과 자신과의 간선에 대한 책임이 있다. 목적함수 3.8를 제어 법칙인 3.9에 대입하면

$$\begin{aligned}
\phi_3(p(t)) &= \frac{1}{4}(e_{13} + e_{23})^2 \\
&= \frac{1}{4}(||\mathbf{p}_1(t) - \mathbf{p}_3(t)||^2 - ||\mathbf{p}_1^* - \mathbf{p}_3^*||^2 + ||\mathbf{p}_2(t) - \mathbf{p}_3(t)||^2 - ||\mathbf{p}_2^* - \mathbf{p}_3^*||^2)^2 \\
&= \frac{1}{4}(||\mathbf{z}_{13}||^2 - ||\mathbf{z}_{13}^*||^2 + ||\mathbf{z}_{23}||^2 - ||\mathbf{z}_{23}^*||^2)^2 \\
\frac{\partial \phi_3}{\partial p_3} &= -(e_{13}\mathbf{z}_{13} + e_{23}\mathbf{z}_{23})
\end{aligned}$$

위 결과를 각 에이전트에 대한 입력으로 표현하면 아래와 같다.

$$\mathbf{u}_1 = 0, \quad \mathbf{u}_2 = e_{12}\mathbf{z}_{12}, \quad \mathbf{u}_3 = e_{13}\mathbf{z}_{13} + e_{23}\mathbf{z}_{23}$$

위 결과를 살펴보면 경사하강 기반의 제어입력은 상대 거리 벡터에 제곱의 거리 오차의 제곱항이 곱해진 형태라는 것을 알 수 있다. 따라서 제어입력은 에이전트를 에이전트간 상대거리 벡터의 방향으로 주어진 제어 이득만큼 움직이게 된다.

여기서 주의할 점은, 상대거리 정보 $||\mathbf{z}_{ij}||$만을 가지고 제어하는 것이 아니라 상대위치 벡터인 $\mathbf{z}_{ij}$을 통해 제어한다는 점이다. 즉, 센싱 변수는 거리와 각도 정보를 포함한 벡터성분이다. 본 편대제어 문제가 거리기반 편대제어로 분류되는 이유는 제어 변수가 에이전트간의 거리이기 때문이다. 에이전트는 주어진 거리를 제어할 뿐 각도는 제어하지 않는다. 현재 초광대역 센서를 통해 팔로워1과 팔로워2 사이의 상대거리만을 계측하고 있기 때문에 경사하강 기반의 방법을 사용하기 위해서는 추가적인 변수 추정이 필요하다. 이에 대한 내용은 다음 부분에서 다루고 본 부분에서는 이미 추정되었다 가정하고 식을 전개해 나간다.

이때 팔로워2의 제어입력은 종국에는 두 벡터의 합으로 정의된다. 이러한 서로 다른 벡터의 합은 가시성 제약조건을 만족시키지 못하게 하는 원인이 된다. 이러한 벡터의 합으로 구성된다는 것은 각 변의 제곱의 오차의 제곱에 대한 항을 한번에 만족한다는 것을 의미한다. 따라서 각 변의 길이를 나누어서 수렴시켜도 동일한 평형집합에 속함을 알 수 있다.

3.2.3 에이전트 별 모드 알고리듬

일차적으로 가시성 제약조건을 만족하기 위한 $SEARCH\ LEADER$ 모드를 생각해 보자. 피동센서에 리더가 탐지되지 않는다면 일정 값의 각속도(ω_{search})로 헤딩방향을 틀며

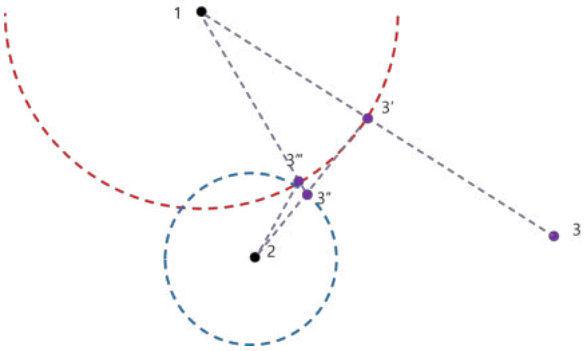


그림 3.7. 모드 알고리듬의 진행방법

리더를 찾는다. 이때 리더가 팔로워의 머리위등 관측할 수 없는 곳에 있는 상황은 배재한다. 리더를 찾은 후 에이전트간의 상대거리를 만족시키는 경사하강 알고리듬을 수행한다.

팔로워가 리더를 관측하면 주어진 경사하강 방향과 자신의 헤딩방향을 일치시키는 과정을 거친다. 단 두대의 에이전트간의 상대거리기 때문에 리더와 팔로워의 위치를 지나는 직선형태로 움직이게 된다. 이렇게 될 경우 리더를 놓치지 않을 수 있다. 이때 헤딩 방향 일치에 약간의 오차가 있어 리더를 놓치는 경우에는 $SEARCH\ LEADER$ 모드로 다시 리더를 관측하여 진행할 수 있다. 이러한 경우가 그림 3.7의 에이전트3이 3'으로 오는 과정이고 이때의 모드를 $FOLLOW\ LEADER$라고 한다. 이 과정에서 팔로워1의 경우 리더에 대한 경사하강의 제어입력의 크기 $||\eta_1||$가 주어진 문턱값(threshold) 이하가 되면 정지하게 된다. 팔로워1의 모드 알고리듬 진행은 그림 3.8에 도시되었다. 팔로워2의 경우 팔로워1과 같은 방법으로 제어입력의 크기가 문턱값 아래로 내려가면 팔로워1과의 경사하강 제어입력을 인가받는 $FOLLOW\ FOLLOWER1$ 모드를 수행한다. 팔로워1에 대한 경사하강 입력의 각속도만 인가하다가 경사하강 방향과 자신의 헤딩 방향이 일치하게 되면 나머지 경사하강 입력을 인가받는다. 팔로워1에 대한 경사하강의 제어입력의 크기 $||\eta_2||$가 주어진 문턱값 이하가 다시 $FOLLOW\ LEADER$ 모드로 전환하여 이를 반복하다두 모드 전부가 문턱값을 만족하게 되면 정지하게 된다. 이러한 과정을 반복하게 되면 그림 3.7의

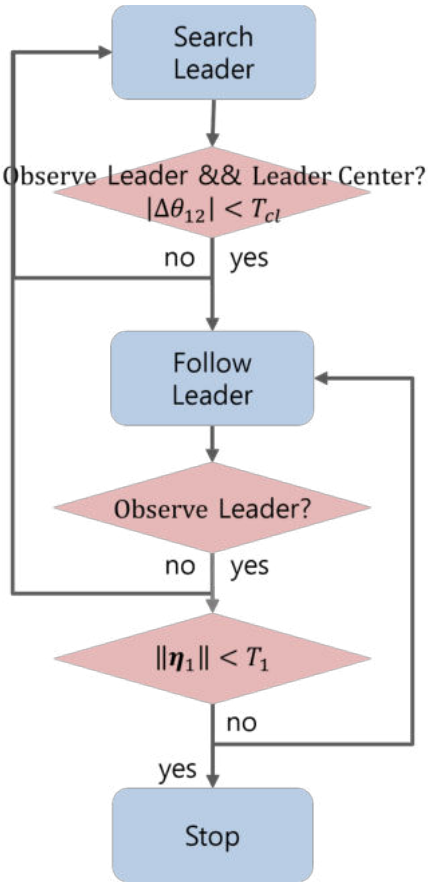


그림 3.8. 팔로워1의 모드 알고리듬 순서도

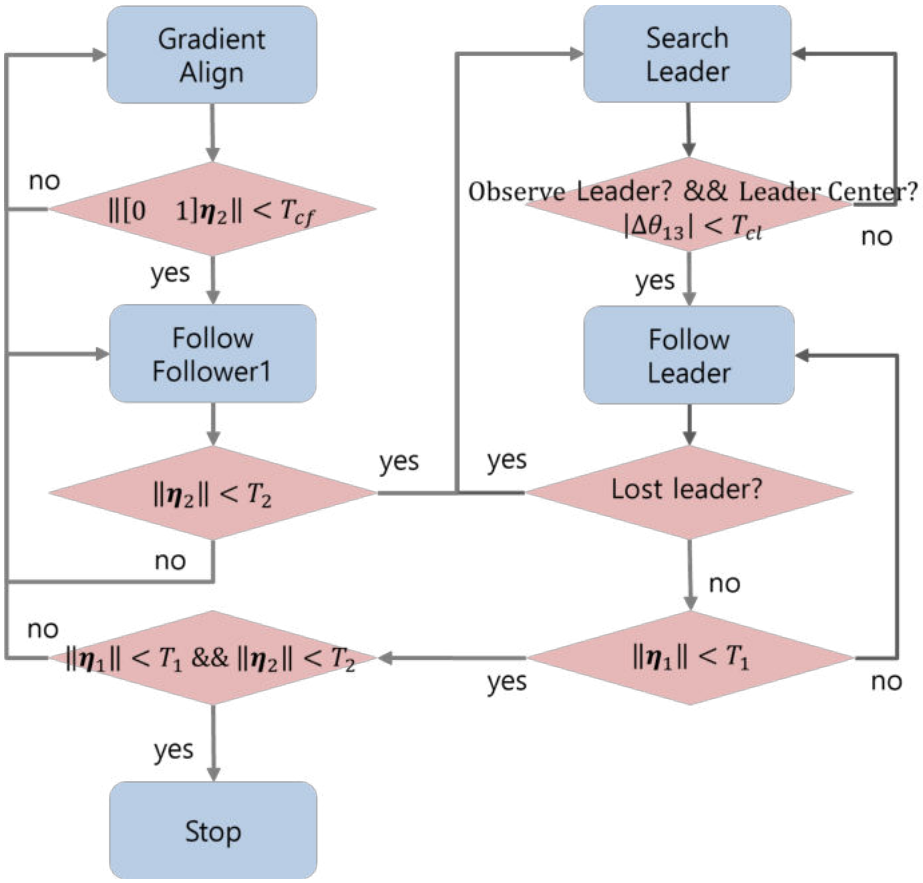


그림 3.9. 팔로워2의 모드 알고리듬 순서도

에이전트 $3 \rightarrow 3' \rightarrow 3'' \rightarrow 3''' \ldots$와 같이 경로가 이동되게 되고 빨간 점선과 파란 점선의 초기위치에서 가까운 교점으로 수렴하게 된다. 팔로워2의 모드 알고리듬 순서도는 그림 3.9에 도시되었다. 만약 직선으로 유지하지 않는다면 각 평형 집합 원위의 아무 점이나 이동하며 수렴을 보장하지 못한다. 또한 기존의 경사하강의 단점인 초기위치가 일직선일 때에도 수렴을 모장하지 못하며 이러한 경우는 배재하였다.

3.2.4 논홀로노믹 에이전트를 위한 경사하강 방법의 변형된 알고리듬

이제 경사하강 제어입력을 논홀로노믹 에이전트에 적용하기 위한 변형된 알고리듬을 살펴보자. 3.3에서는 에이전트의 위치 $\mathbf{p}_i$에에 헤딩각도의 다아니믹스가 추가된다. 따라서

논홀로노믹 조건을 갖는 외륜모델은 기존 모델 3.2 에서 헤딩각도의 미분까지 포함되어야 한다. $\mathbf{h}_i(t) \in \mathbb{R}^2$를 i번째 에이전트의 단위 헤딩 벡터라 하고 2차원 평면에 수직한 축을 z축이라 하자. 여기서 헤딩각도는 z축에 대한 회전이다. 따라서 헤딩각도의 미분 $\dot{\theta}_i$는 $\mathbf{h}_i$와 z축 성분을 각속도로 가진 벡터와의 외적의 형태로 기술할 수 있다. 즉 $\dot{\mathbf{h}}_i = \omega_i \times \mathbf{h}_i$로 정의할 수 있다. 이때 식 3.3의 입력인 v_i와 ω_i를 각각 $\mathbf{h}_i^T\mathbf{u}_i$와 $\mathbf{h}_i \times \mathbf{u}_i$ 로 설정하면 식 3.3은 아래와 같이 다시 기술될 수 있다.

$$\begin{aligned} \dot{\mathbf{p}}_i &= \mathbf{h}_i\mathbf{h}_i^T\mathbf{u}_i \\ \dot{\mathbf{h}}_i &= (\mathbf{I} - \mathbf{h}_i\mathbf{h}_i^T)\mathbf{u}_i, \quad i \in \{2,3\} \end{aligned} \tag{3.10}$$

여기서, $I \in \mathbb{R}^{2\times 2}$인 단위행렬이다. $\mathbf{h}_i\mathbf{h}_i^T$은 단위 헤딩벡터 $\mathbf{h}_i$에 대한 정사영 행렬이다. 그림 3.10에서 확인할 수 있듯이. $\mathbf{h}_i\mathbf{h}_i^T$와 $(I - \mathbf{h}_i\mathbf{h}_j^T)$은 서로 수직하다. 즉, 경사하강 입력을 자신의 헤딩벡터로 정사영 시킨 성분을 속도, 헤딩벡터 방향에 수직한 방향에 대한 정사영 성분을 각속도 산출한다. 외륜모델에 적용하여 산출한 속도 및 각속도는

$$\begin{aligned} v_i &= [\cos\theta_i,\ \sin\theta_i]\mathbf{u}_i \\ \omega_i &= [-\sin\theta_i,\ \cos\theta_i]\mathbf{u}_i, \end{aligned}$$

로 다시 쓸 수 있다.

외륜 모델의 속도가 헤딩벡터 방향과 일치한다는 점을 고려하였을때, 경사하강 제어입력의 방향과 헤딩방향을 맞춰 각속도 입력을 산출하는 것은 타당해 보인다. 특히, 경사하강 제어입력의 방향과 헤딩방향의 차이가 클수록 더 많은 헤딩방향이 제어입력을 방향과 일치시켜주기 위해 각속도 입력이 인가된어야 하는데 이러한 점을 정사영 연산이 잘 내포하고 있다.

안타깝게도 [2]에서는 가정한 그래프는 완전 무향 그래프(connected and undirected graph)이다. 방향 그래프의 경우, 제시한 제어기법이 동작할 수도 있지만 경사하강 기반

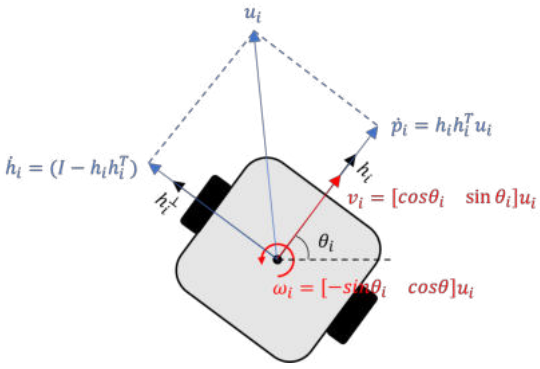


그림 3.10. 변형된 경사하강 알고리듬의 기하학적 해석

제어 법칙이 아닐수 있다고 언급하고 있다. 본 논문에서 제시한 모드 알고리듬은 한번에 한 에이전트와이 거리만을 수렴시키면 되기때문에 두대의 에이전트간의 상대거리르 유지하는 것만 증명하면 된다.

변형된 알고리듬의 제어입력을 인가받은 논홀로노믹 에이전트 j와 정지해있는 에이전트 i의 상대거리 벡터의 페루프 다이나믹스는 다음과 같다.

$$\dot{\mathbf{z}}_{ij} = -\mathbf{h}_j \mathbf{h}_j^T e_{ij} \mathbf{z}_{ij}$$

다음과 같은 리아푸노프 함수를 정의해보자.

$$V = \frac{1}{4} e_{ij}^2 \tag{3.11}$$

위 리아푸노프 함수의 미분이 항상 0보다 작음을 증명한다면 제시한 제어입력이 수렴함을 보일 수 있다.

$$\begin{aligned} \dot{V} &= \frac{1}{2} e_{ij} \dot{e}_{ij} \\ &= e_{ij} \mathbf{z}_{ij} \dot{\mathbf{p}}_i \\ &= -||e_{ij}||^2 (\mathbf{z}_{ij} \mathbf{h}_j \mathbf{h}_j^T \mathbf{z}_{ij}) \end{aligned}$$

이때, 정사영 행렬 $\mathbf{h}_j \mathbf{h}_j^T$이 정부호 행렬(positive semi-definite matrix)이기 때문에 위 함수도 정부호 이다. 위 함수가 0이 되는 경우를 조사해보자. $\mathbf{h}_j^T \mathbf{z}_{ij} = 0$이거나 $||e_{ij}||^2 = 0$인

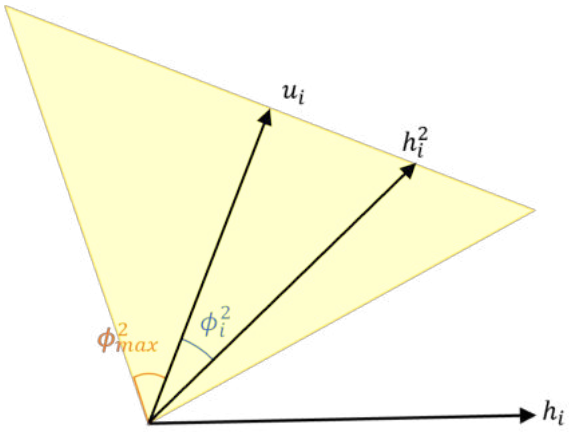


그림 3.11. 포화조건을 위한 유연한 알고리듬

경우이다. 첫번째 경우는 논 홀로노믹 에이전트의 헤딩벡터와 경사하강 기반 제어입력의 각도가 수직을 이루었을 때이다. 이 경우에는 경사하강의 변형된 알고리듬을 통해 각속도가 회전하므로 헤딩벡터 행렬이 변하게된다. 따라서 이러한 경우는 존재하지 않는다. 마지막 경우는 원하는 상대거리를 만족한 경우이다. 따라서 리아푸노프 함수는 제어목적을 만족했을때에만 0이 되므로 변형된 경사하강 알고리듬은 제시한 알고리듬을 수렴하게 함을 알 수 있다.

3.2.5 포화 조건을 반영한 유연한 알고리듬

논홀로노믹 제약조건에 더해 일반적인 제약조건은 포화조건이 있다. 실제 에이전트에 인가되는 입력에 대한 출력은 물리적 한계로 인해 제한되기 때문이다. 앞서 언급하였던 식 3.4인 상황에서도 주어진 제어기가 수렴하는지 조사할 필요가 있다. 식 3.10에 포화조건을 추가한 형태로 일반화하여 유연한 알고리듬을 유도해보자.

$$\begin{aligned} \dot{\mathbf{p}}_i &= \kappa_i \mathbf{h}_i \mathbf{h}_i^T \mathbf{u}_i \\ \dot{\mathbf{h}}_i &= (\mathbf{I} - \mathbf{h}_i \mathbf{h}_i^T)\mathbf{h}_i^2, \quad i \in \{2,3\} \end{aligned} \tag{3.12}$$

κ_i은 양의 상수, $\mathbf{h}_i^2 \in \mathbb{R}^2$ 이고 두 변수는 시간에 따라 변한다. 그림 3.11은 위 식을 도식화한 것이다. κ_i는 속도의 포화조건을 만족하기 위해 속도의 크기를 조절하기 위한 변수로 사용된다. $\mathbf{h}_i^2$은 바람직한 헤딩 벡터(desired heading vector)로 각속도의 포화 조건을 만족하기 위해 각속도를 조절하는데 사용된다. $(I - \mathbf{h}_i\mathbf{h}_i^T)$에 $\mathbf{u}_i$를 정사영시킨 값이 각속도 포화 조건을 벗어나게 하지 않기위해 제어입력 벡터보다 헤딩각도와 이루는 각도가 작은 벡터인 $\mathbf{h}_i^2$에 정사영 시킴으로서 포화조건을 만족하는 입력을 산출한다. 이때, $\kappa_i(t)$와 $h_i^2(t)$가 다음의 3가지 조건을 만족한다고 하자.

1. $\kappa_i(t)$가 시간 t에 대해 균등 연속이고 모든 i와 t에 대해 $0 < \kappa_{min} \leq \kappa_i(t) \leq \kappa_{max}$의 범위를 가진다.

2. $\phi_i^2(t)$는 모든 i와 t에 대해 $0 \leq \phi_i^2(t) \leq \phi_{max}^2 < \pi/2$의 범위를 가진다. $\phi_i^2(t)$는 $\mathbf{h}_i^2(t)$와 $\mathbf{u}_i$사이의 각도이다.

3. $||\mathbf{h}_i^2(t)||$는 모든 i와 t에 대해서 $0 \leq \mathbf{h}_i^2(t) \leq \mu_{max}^2$의 범위를 가지고 오직 $||\mathbf{u}_i|| = 0$일 때만 $||\mathbf{h}_i^2(t)|| = 0$이다.

3.12의 우변에는 시변 변수인 κ_i와 $\mathbf{h}_i^2$가 존재하므로 비자율 시스템(nonautonomous system)이다. 따라서 Barbalat's Lemma [16, Lemma 8.2] 를 이용하여 수렴성을 유도해야 한다.

Barbarlat's Lemma는 어떤 비자율 시스템 $f(t)$가 존재할때, $f(t)$가 어떤 상수 C에 수렴하고 $f'(t)$가 균등 연속이면 $f'(t)$는 0으로 수렴함을 말한다. 리아푸노프 함수 3.11는 비증가하고 아래로 경계진 함수이기 때문에 시간이 지남에 따라 수렴한다. 이제 $\dot{V}$가 균등 역속임을 보이자. 3.11의 미분은

$$\dot{V} = -\kappa(\mathbf{u}_j\mathbf{h}_i\mathbf{h}_i^T\mathbf{u}_j)$$

이므로 그 구성성분인 $\mathbf{h}_j, \mathbf{u}_j$가 시간에 대해 균등 연속임을 보임으로 $\dot{V}$가 시간에 대해 균등 연속임을 보인다. $\mathbf{h}_i$는 미분가능하고 그 미분이 $||\dot{\mathbf{h}}_i|| = ||(\mathbf{I} - \mathbf{h}_i\mathbf{h}_i^T)\mathbf{h}_i^2|| \leq \mathbf{h}_i^2 \leq \mu_{max}^d$ 인 유계를 갖기 때문에 시간에 대해 균등 연속이다. $\kappa_i < \kappa_{max}$이므로 $||\dot{\mathbf{p}}_i = \kappa_i\mathbf{h}_i\mathbf{h}_i^T\mathbf{u}_i||$가 유계를 갖는다. 따라서 $\mathbf{p}$는 미분가능하고 그 미분이 유계를 갖기 때문에 균등 연속이다. $\dot{\mathbf{p}}_i$이 유계를 가지므로 $\dot{e}_{ij} = (\partial e_{ij}/\partial \mathbf{p}_i)\dot{\mathbf{p}}_i$ 도 유계를 갖는다. 따라서 e_{ij}도 균등 연속이다. $\mathbf{u}_j(e, \mathbf{p})$는 e와 $\mathbf{p}$에 대해서 균등 연속이며 e와 $\mathbf{p}$가 시간에 대해서 균등 연속이기 때문에 $\mathbf{u}_j$는 시간 t에 대해 균등 연속이다. κ_j는 가정에 의해 균등 연속이므로 모든 균등 연속으로 이루어진 함수 $\dot{V} = -\kappa(\mathbf{u}_j\mathbf{h}_i\mathbf{h}_i^T\mathbf{u}_j)$은 균등 연속이고 Barbarlat's Lemma에 의해 $\dot{V}$는 시간이 지남에 따라 0으로 수렴한다. 시변 변수 $\kappa_i(t)$와 $\mathbf{h}_i^2(t)$에 대한 가정은 실제 상황에서 쉽게 적용할 수 있어 많은 유연성을 제공한다. κ_{min}은 임의로 작게, κ_{max}는 임의로 크게 설정할 수 있으며, ϕ_{max}^2는 $\pi/2$에 임의로 가깝게 설정할 수 있다. 포화조건 3.4을 적용한 논홀로노믹 에이전트의 운동 방정식을 아래와 같이 표현해보자.

$$\begin{aligned} \dot{\mathbf{p}}_i &= \mathbf{h}_i sat(\mathbf{h}_i^T\mathbf{u}_i) &&= \mathbf{h}_i v_i \\ \dot{\mathbf{h}}_i &= \mathbf{h}_i^{\perp} sat((\mathbf{h}_i^{\perp})^T\mathbf{u}_i) &&= \mathbf{h}_i \omega_i \end{aligned}$$

여기서

$$sat(x) = \begin{cases} -x_{SAT} & x \in (\infty, -x_{SAT}) \\ x & x \in (x_{SAT}, x_{SAT}) \\ x_{SAT} & x \in (x_{SAT}, \infty) \end{cases}$$

$sat(\mathbf{h}_i^T\mathbf{u}_i)$을 $\kappa_i\mathbf{h}_i^T\mathbf{u}_i$로 표현하면,

$$\kappa_i = \begin{cases} \frac{V_{SAT}}{-\mathbf{h}_i^T\mathbf{u}_i}, & \mathbf{h}_i^T\mathbf{u}_i \in (-\infty, -V_{SAT}), \\ 1 & \mathbf{h}_i^T\mathbf{u}_i \in (-V_{SAT}, V_{SAT}), \\ \frac{V_{SAT}}{\mathbf{h}_i^T\mathbf{u}_i}, & \mathbf{h}_i^T\mathbf{u}_i \in (V_{SAT}, +\infty), \end{cases}$$

같은 방법으로 $sat((\mathbf{h}_i^{\perp})^T\mathbf{u}_i)$을 $\rho_i(\mathbf{h}_i^{\perp})^T\mathbf{u}_i$로 나타내면

$$\rho_i = \begin{cases} \frac{W_{SAT}}{-\mathbf{h}_i^T\mathbf{u}_i}, & (\mathbf{h}_i^{\perp})^T\mathbf{u}_i \in (-\infty, -W_{SAT}), \\ 1 & (\mathbf{h}_i^{\perp})^T\mathbf{u}_i \in (-W_{SAT}, W_{SAT}), \\ \frac{W_{SAT}}{\mathbf{h}_i^T\mathbf{u}_i}, & (\mathbf{h}_i^{\perp})^T\mathbf{u}_i \in (W_{SAT}, +\infty), \end{cases}$$

3.3 적응형 매개변수 식별

본 논문의 문제정의에서 팔로워2는 팔로워1과의 상대거리 정보를 UWB와 같은 센서를 통해서 획득한다고 설명한 바 있다. 하지만 앞서 살펴보았던 경사하강 기반의 제어기는 팔로워1의 상대위치 정보를 모두 사용하여 제어입력을 산출한다. 그렇기 때문에 거리정보 뿐만 아니라 각도 정보 또한 획득해야 한다. 하지만 UWB와 같은 센서들은 이러한 정보가 가용하지 않기 때문에 팔로워2의 좌표계에서 팔로워1의 위치를 추정하는 위치추정(localization)문제가 도입된다.

거리 측정치 만을 가지고 타겟의 위치를 추정하는 문제는 매개변수 식별자 기반 적응형 제어기 프레임워크(parameter-identifier-based adaptive control framework)를 이용할 수 있다. 적응 제어란 파라미터를 실시간으로 추정하는 파라미터 추정기와 알지 못하는 시스템 파라미터나 시간에 따라 예측할 수 없는 형태로 변할 수 있는 파라미터를 포함한 시스템을 제어하기 위한 제어기의 조합이다 [17]. 적응 제어의 한 부분인 파라미터 식별 기법을 활용하여 팔로워2가 팔로워1의 위치를 추정하는 문제를 다음과 같이 정의해보자.

3.3.1 거리기반 위치측위 문제

Problem 3.2. *시간에 따른 에이전트 A의 위치를* $\mathbf{p}_A(t) \in \mathbb{R}^2$*라 하자. 그리고 에이전트 A가 위치를 알아내고 싶은 타겟 아에전트 T의 위치를* $\mathbf{p}_T(t) \in \mathbb{R}^2$*라 하자. 측정치 거리정보*

$$D(t) = ||\mathbf{p}_A(t) - \mathbf{p}_T(t)|| \tag{3.13}$$

만을 가지고 추정한 T의 추정위치 $\hat{\mathbf{p}}_T(t)$*가 있을 때, 다음 조건을 만족하며 T의 위치를 추정하는 적응 법칙을 설계하라.*

$$\lim_{t\to\infty} ||\hat{\mathbf{p}}_T(t) - \mathbf{p}_T|| = 0$$

[18]에서는 3.2를 경사하강 기반 위치측위 알고리듬(gradient-type localization algorithm)을 설계하여 해결하였다 . 매개변수 추정 프레임워크(parameter estimation framework)에서는 파라미터 모델을 선정하여 추정식을 유도하게 된다 [17, 19]. [18]에서는 A의 경로가 두번 미분 가능함을 전재로 하여 파라미터 모델을 얻기위해 거리의 제곱을 미분하였다.

$$\begin{aligned}\frac{d}{dt}D^2(t) &= 2\dot{\mathbf{p}}_A^T(t)(\mathbf{p}_A - \mathbf{p}_T) \\ &= \frac{d}{dt}||\mathbf{p}_A(t)||^2 - 2\dot{\mathbf{p}}_A^T(t)\mathbf{p}_T\end{aligned}$$

위의 형태는 다음과 같은 매개변수 추정 모델의 형태로 변환될 수 있다.

$$\bar{z} = x^T\bar{\phi}(t) \tag{3.14}$$

이때,

$$\bar{z} = \frac{1}{2}\frac{d}{dt}(||\mathbf{p}_A(t)||^2 - D^2), \quad \bar{\phi} = \dot{\mathbf{p}}_A$$

[18]에서는 위치를 알지못하는 에이전트 T 가 정지해 있다고 가정한다. 본 논문의 문제정의에서도 이 가정이 합당한 이유는 에이전트 A가 에이전트 T를 추정하는 시간보다 에이전트 T가 수렴하는 시간이 훨씬 빠르기 때문이다. 이러한 움직임을 표류(drift)처럼 해석하여 파라미터 추정기가 수렴함을 보일 수 있다. 파라미터 모델 3.14을 활용하여 $\mathbf{p}_T$의 $\hat{\mathbf{p}}_T$를 추정하기 위한 경사하강 알고리듬은 아래와 같이 기술된다.

$$\begin{aligned}\dot{\hat{\mathbf{p}}}_T(t) &= \gamma(\bar{z}(t) - \hat{z}(t))\bar{\phi}(t), \\ \hat{\bar{z}}(t) &= \hat{\mathbf{p}}_T^T(t)\bar{\phi}(t)\end{aligned} \tag{3.15}$$

이때, γ는 학습률(learning rate)로 측정값 $\bar{z}$과 추정값 $\hat{z}$간의 오차를 위치 추정치의 미분 $\dot{\hat{\mathbf{p}}}_T$에 업데이트 하는 비율을 결정하는 상수이다. 일반적으로 γ가 크면 수렴속도가 빨라지지만 불안정해지며 γ가 작다면 수렴속도는 느려지지만 안정성이 증가하는 경향을 보인다. 식 3.15에서 $\bar{\phi} = \dot{\mathbf{p}}_A$이 지속적인 여자(persistently exciting, 이하 $p.e.$)라면 $\hat{\mathbf{p}}_T$가 $\mathbf{p}_T$에 지수적으로 수렴함이 잘알려져 있다 [17, 19, 20].

3.3.2 지속적인 여자 조건

Definition 3.1. *[17, 19, 20] 임의의 양의 정수 n과 신호 $\mathbf{r} : \mathbb{R} \to \mathbb{R}^n$이 있다고 하자. 모든 시간 t에 대해 다음을 만족하는 양수인 α와 T_1이 존재한다면 $\mathbf{r}$은 $p.e.$이다.*

$$\alpha_1 I \leq \int_t^{t+T_1} \mathbf{r}(\tau)\mathbf{r}(\tau)^T d\tau \leq \alpha_2 I$$

[20]에서는 적응형 식별자의의 $p.e.$ 조건을 시변 선형 미분 방정식의 균등 완전 가관측성(uniformly completely observability) 조건과 동일한 조건임을 설명한다. $p.e.$의 물리적 의미를 살펴보자. 신호 $\mathbf{r}$이 $p.e.$이기 위해서는 특정 주기 시간동안의 제곱의 적분값이 특정 값 범위 내에 존재해야한다. 특히나 $\mathbf{r} \in \mathbb{R}^n$인 신호는 $\alpha > 0$에 $I \in \mathbb{R}^{n\times n}$이 곱해진 것으로 보아 n차원의 방향성분을 모두 가져야 한다는 것을 알 수 있다. 따라서 2차원이라면 직선운동 3차원이라면 평면운동을 하면 신호 $\mathbf{r}$은 $p.e.$ 신호의 정의를 만족하지 못한다. 3.2에 적용하여 살펴보면, 에이전트 T를 위치추정하기 위한 에이전트 A의 움직임은 직선이 아닌 운동을 해야한다. 이러한 조건을 만족하기 위한 입력신호의 일반적인 절차는 정현파(sinusoidal wave)나 주기 신호의 유한 합으로 구성하는 것이다. 이렇게 되면 입력신호는 주기 또는 거의 주기(almost periodic) 신호이며 만약 입력 신호에 충분히 다른 주파수가 존재한다면 지속적 여자 조건을 만족하게 된다.

3.3.3 상태변수 필터링 기법

3.15 구현 시에 측정치 $D(t)$값을 직접 미분해서 얻는 항이 존재하여 신호의 잡음에 취약하다. 측정신호에 대한 직접적인 미분은 노이즈를 증폭할 수 있는 위험이 있기 때문이다. 그 때문에 상태 변수 필터링 기법(state variable filtering method)이 적응 시스템에서 널리 사용되고 있다 [21]. 상태 $\bar{z}, \bar{\phi}$의 필터된 값을 아래와 같이 표현해보자.

$$z(t) = \dot{\zeta}_1(t) = -\alpha\zeta_1(t) + \frac{1}{2}(\mathbf{p}_A^T(t)\mathbf{p}_A(t) - D^2(t)),$$

$$\phi(t) = \dot{\zeta}_2(t) = -\alpha\zeta_2(t) + \mathbf{p}_A(t),$$

이때 $\alpha > 0, \zeta_1(0)$은 임의의 스칼라이고 $\zeta_2(0) \in \mathbb{R}^2$은 임의의 벡터이다. 필터된 값을 사용하여 수정된 적응형 파라미터 추정 모델과 위치측위 모델은 각각

$$z(\cdot) \equiv \mathbf{p}_T^T \phi(\cdot)$$

와

$$\begin{aligned} \dot{\hat{\mathbf{p}}}_T(t) &= \gamma(z(t) - \hat{z}(t))\phi(t), \\ \hat{z}(t) &= \hat{\mathbf{p}}_T(t)^T \phi(t) \end{aligned} \tag{3.16}$$

이다. [22]에서 측정 모델 3.13과 상태변수 필터링 기법을 사용한 적응형 매개변수 식별자 3.16가 수렴함을 보이고 있다. 추가적으로, 3.16은 느리고 유계를 가지지만 잠재적으로 지속적인 표류하는 잡음이 있는 신호를 추정 할 수 있음이 증명되었다 [18].

3.3.4 지속적인 여자 신호 선정

$p.e.$신호는 주어진 차원에서 충분한 정보를 얻을 수 있음을 보장해 주는 조건이며 주기함수로 구성되는 경우가 많다. $p.e.$입력 $pe(t) = [pe_x, pe_y]^T \in \mathbb{R}^2$ 은

$$pe(t) = [A\cos(\omega_{pe}t), A\sin(2\omega_{pe}t)]^T \tag{3.17}$$

여기서 A는 주기함수의 증폭, 그리고 ω_{pe}는 주파수를 의미한다. x방향 입력과 y방향 입력은 2배 차이나게 설정하였다. 위 변수값을 선정시에 모바일 로봇의 선속도 및 각속도 포화조건을 고려해야 한다.

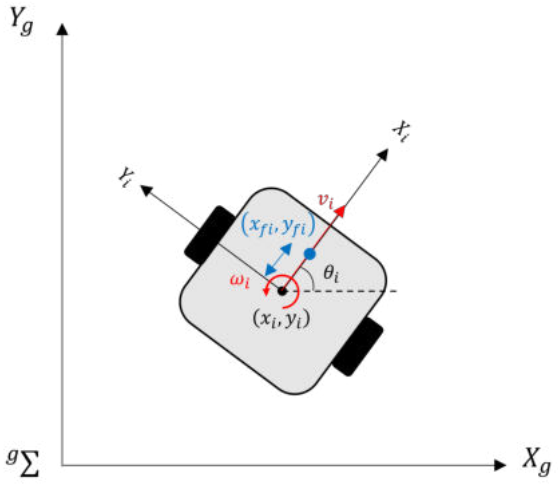


그림 3.12. 논홀로노믹 에이전트를 위한 되먹임 선형화

3.3.5 p.e.입력 수행을 위한 논홀로노믹 에이전트의 되먹임 선형화

3.17은 단일 적분기 모델에게는 바로 적용이 가능하지만 논홀로노믹 에이전트는 헤딩각도에 대한 다이나믹스 까지 존재하기 때문에 이러한 입력을 인가할 수 없다. [1]에서는 단일 적분기 모델을 대상으로 제안된 제어기를 외륜 모델에 적용하기위해 되먹임 선형화(feedback linearization)을 적용하였다. 그림 3.12은 로봇의 중앙점 (x_i, y_i)에서 d 만큼 떨어진 점인 (x_{fi}, y_{fi})을 도시하고 있다. 되먹임 선형화는 해당 점의 미분이 단일 적분기 모델을 만족하는 제어입력을 설계한다. $x_{fi} = x_i + d\cos\theta,\ y_{fi} = d\sin\theta$ 이고 $\dot{x}_i = v\cos\theta,\ \dot{y}_i = v\sin\theta$이므로

$$\begin{aligned}\dot{x}_{fi} &= v\cos\theta - d\sin\theta\,\omega \\ \dot{y}_{fi} &= v\sin\theta + d\cos\theta\,\omega\end{aligned}$$

이때 제어입력 v, ω와 $p.e.$입력 pe_x, pe_y과의 관계를 다음과 같이 설정해보자.

$$\begin{bmatrix} v_i \\ \omega_i \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\frac{1}{d}\sin\theta & \frac{1}{d}\cos\theta \end{bmatrix} \begin{bmatrix} pe_x \\ pe_y \end{bmatrix} \tag{3.18}$$

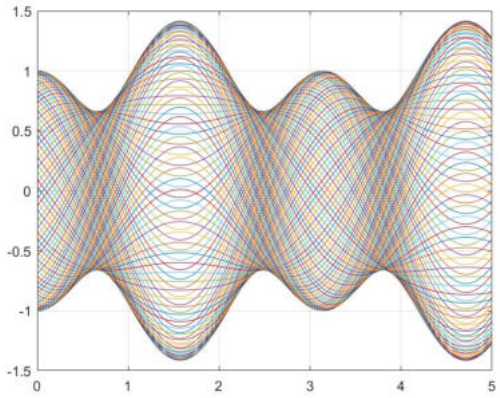


그림 3.13. $p.e.$ 신호 최대 크기

이러한 변환을 통해 pe입력을 받은 외륜모델은 점 (x_{fi}, y_{fi})에서 아래와 같은 단일 적분기 모델을 가지게 된다.

$$\begin{bmatrix} \dot{x}_{fi} \\ \dot{y}_{fi} \end{bmatrix} = \begin{bmatrix} pe_x \\ pe_y \end{bmatrix}$$

식 3.18 변환관계를 이용하여 모바일 로봇의 입력인 v_i, ω_i가 포화조건을 만족하도록 $pe(t)$의 변수들을 설정해보자. 식 3.18에 식 3.17를 대입하여 전개하면

$$|A\cos\theta\cos\omega_{pe}t + A\sin\theta\sin 2\omega_{pe}t| \le V_{SAT}$$
$$-\frac{1}{d}|A\sin\theta\cos\omega_{pe}t - A\cos\theta\sin 2\omega_{pe}t| \le W_{SAT}$$

위 식의 조건을 만족하는 θ, ω_{pe}를 찾기 위해서는 다변수 함수의 극점을 조사함으로 알 수 있지만, 수치계산을 통해 위 식을 각 변수별로 도시하여 쉽게 얻을 수 있다. A를 1로 놓고 $\theta, \omega_{pe} = 0 : 0.1 : 2\pi$ 로 시뮬레이션 한 결과를 3.13에 도시하였다. A값이 1일때 v의 최대 값은 약 $\sqrt{2}$, ω의 최대값은 약 8.33이며 모바일 로봇의 포화조건을 만족하지 않는다. $A = V_{SAT}/\sqrt{2}$로 놓으면 ω의 최대 인가 범위도 1.5 이기 때문에 두 포화조건을 모두 만족하는 v, ω값을 얻을 수 있다.

3.3.6 추정완료 판별조건

기존의 거리기반 위치 추정이 어느정도 기준을 넘어섰다면 그 다음 편대제어입력에 따라 에이전트는 이동해야한다. 위치 추정의 미분 값이 크다면 위치 추정이 많이 되고 작다면 위치 추정의 업데이터가 조금만 변한다. 따라서 추정하는 값의 미분값의 크기가 특정 크기 이하로 작아지게 되면 충분한 추정이 완료되었다는 의미로 볼 수 있다. 하지만 측정 센서의 노이즈에 따라 문턱값이 달라질 수 있고 추정치의 변화량 또한 주기함수처럼 줄어들었다 늘었다하기는 반복하기 때문에 그 값을 설정하기 까다롭다. 따라서 본 논문에서는 충분한 시간동안 위치 측위를 시행하고 그 후에 편대 입력을 인가받는 것으로 진행한다.

3.4 전체 알고리듬과 모의실험

앞선 알고리듬을 통합하여 모의실험을 진행한다. 모의실험은 메트랩 스크립트상에서 진행되었다.

3.4.1 에이전트의 지역좌표계 상에서 입력

팔로워1은 피동센서에 리더가 관측되지 않으면 $SEARCH\ LEADER$ 모드로 진입해 지정된 각속도(ω_{search})로 회전하며 리더를 찾는다. 리더가 관측되면 $FOLLOW\ LEADRE$ 모드로 진입하고 리더와의 상대거리 벡터와 자신의 헤딩 좌표계가 일치할때까지 ω_{search}로 회전한다. 일치되면 리더를 상대로 하는 경사하강 제어입력을 인가받는다. $FOLLOW\ LEADER$ 모드에서 인가되는 모바일 로봇의 입력은

$$\eta_1 = \begin{bmatrix} v_2 \\ \omega_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} e_{12}\mathbf{z}_{12}$$

이다. 입력의 크기 $||\eta_1||$가 지정한 문턱값 T_1 이하가 되면 원하는 상대거리와 가까워 졌다는 의미이므로 정지한다. 팔로워2는 편대를 이루기 전에 먼저 팔로워1의 위치를 측위한

다. 적응 제어기법 3.16가 사용되며 이때 팔로워2에 인가되는 $p.e.$입력은 3.18이다. 주어진 추정시간 T_l에 도달하면 위치 측위를 중단하고 리더를 탐색한다. 리더가 관측범위 내에 들어오면 설정한 초기값에 따라 $FOLLOW\ LEADER$ 나 $FOLLOW\ FOLLOWER1$ 모드를 시작한다. 팔로워2의 $FOLLOW\ LEADER$에 사용되는 경사하강 제어입력은

$$\eta_2 = \begin{bmatrix} v_3 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} e_{13}\mathbf{z}_{13}$$

$FOLLOW\ FOLLOWER1$ 에 사용되는 경사하강 제어입력은

$$\eta_3 = \begin{bmatrix} v_3 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} e_{23}\mathbf{z}_{23}$$

이 사용된다. 두 모드의 입력의 크기 $||\eta_2||, ||\eta_3||$가 특정 문턱값 T_1, T_2 이하가 되면 정지한다.

전체 알고리듬이 수치 시뮬레이션에서 잘 동작하는지 확인해보도록 하자. 시뮬레이션 환경은 메트랩 스트립트로 작성되었다. 시뮬레이션에서 사용된 변수들은 표 3.1에 기재되었다. $\mathbf{x}_i$의 단위는 $[m, m, rad]$이다. 추가적으로 위치 추정에 사용된 상태 변수 필터링의 초기값은 모두 0으로 설정되었다.

3.4.2 모의 실험

모의실험은 본 논문이 제시한 통합 알고리듬과 단순히 리더를 찾고 정사영 기법을 구현한것을 비교하며 이루어진다. 이것의 차이는 그림 3.9에서 표현된 경사하강 입력의 방향과 헤딩 방향을 일치시키는 것을 배재한 움직임이다. 이러한 움직임은 수렴할 수는 있으나 때에 따라서 논홀로노믹의 특성으로 인해 극소점에 갖힐 수 있다. 그림 3.21은 이러한 모습을 잘 모여준다. 경사하강 방향과 헤딩방향을 일치시켜 이동하는 것은 리더를 놓치지 않게 해줄 뿐만 아니라 그로인해 상대적으로 빠른 수렴성을 보장해준다.

변수	값	변수	값	변수	값
$t_0 : t_s : t_f$	$0 : 0.01 : 500$	$\mathbf{x}_1$	$[-1.17, 0.61, 1.80]^T$	d	$0.15[m]$
FOV	$\pm 40[deg]$	$\mathbf{x}_2$	$[0.15, -1.82, -2.22]^T$	α	1
V_SAT	$0.22[m/s]$	$\mathbf{x}_3$	$[0.07, -1.58, -0.04]^T$	γ	1
W_SAT	$2.84[rad/s]$	$\|\|\mathbf{z}^*_{12}\|\|$	$2.12[m]$	A	$0.22/\sqrt{2}$
ω_{search}	$1[rad/s]$	$\|\|\mathbf{z}^*_{13}\|\|$	$1.52[m]$	ω_{pe}	0.2
T_1, T_2, T_l	$2e^{-2},\ 2e^{-2},\ 180[sec]$	$\|\|\mathbf{z}^*_{23}\|\|$	$0.83[m]$	T_{cl}, T_{cf}	$1[deg], 0.05$

표 3.1. 전체 알고리듬 시뮬레이션 변수 값

그림 3.14에는 전체 알고리듬을 수행 결과이다. 각 에이전트는 그 번호가 매겨져있다. 1은 리더, 2는 팔로워1, 3은 팔로워2 이다. 초기 위치는 빨간 점으로 표현되었고 자세는 헤딩방향이 긴 화살표, 오른쪽으로 수직인 축은 짧은 화살표로 표시되었다. 마지막 위치는 파란 점이며 자세는 초기 자세와 동일한 방법으로 표시되었다. 각 에이전트가 움직인 궤적은 파란색 점선이다. 에이전트 간의 원하는 상대거리 $||\mathbf{z}^s_{ij}||$의 집합은 점선으로 표시하였으며 서로 다른 색으로 구분된다. $||\mathbf{z}^s_{13}||$에 해당하는 집합은 초록색, $||\mathbf{z}^s_{12}||$에 해당하는 집합은 빨간색, $||\mathbf{z}^s_{23}||$에 해당하는 집합은 짙은 노란색을 표시되었다. 이때 마젠타 색의 점과 마젠타 색의 점선으로 표시된 원은 팔로워2가 팔로워 1의 위치를 추정한 위치와 이에 따른 상대거리 집합을 나타낸다.

그림 3.15은 팔로워2가 팔로워1을 거리기반으로 위치를 측위한 결과를 나타낸다. 측위 시간을 180초인데 주어진 시간안에 오차를 빠르게 감소시키고 위치를 추정하였다. 측위 시간이 길어질 수록 측위 오차는 줄어들지만 많은 시간이 소요되기 때문에 어느정도 오차를 허용하여야 한다.

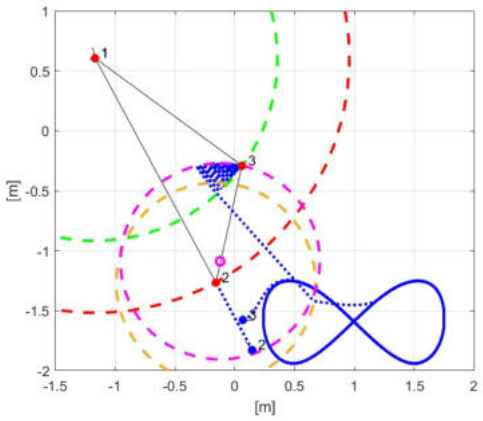


그림 3.14. 각 에이전트의 초기 위치 및 자세, 수렴 위치 및 자세, 궤적, 평형집합

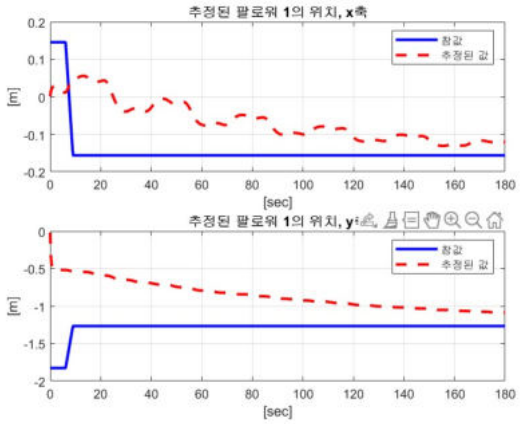


그림 3.15. 위치 측위의 추정값과 참값

그림 3.16은 각 원하는 상대거리와 실제 에이전트간의 상대거리의 차를 보여준다. 팔로워1은 리더와의 거리오차를 빠르게 만족시키는 것을 볼 수 있다. 팔로워2가 위치측위가 끝나고 리더와의 거리와 팔로워1과의 거리를 번가라 가면 만족시키며 팔로워1의 위치 측위 오차로인해 발생하는 상대거리 오차가 존재한다. 본 논문의 모드 분할 알고리듬의 영향은 그림 3.17에서 잘 표현되었다.

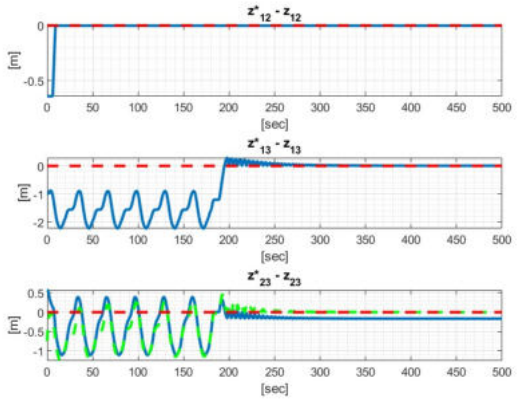


그림 3.16. 상대거리 오차

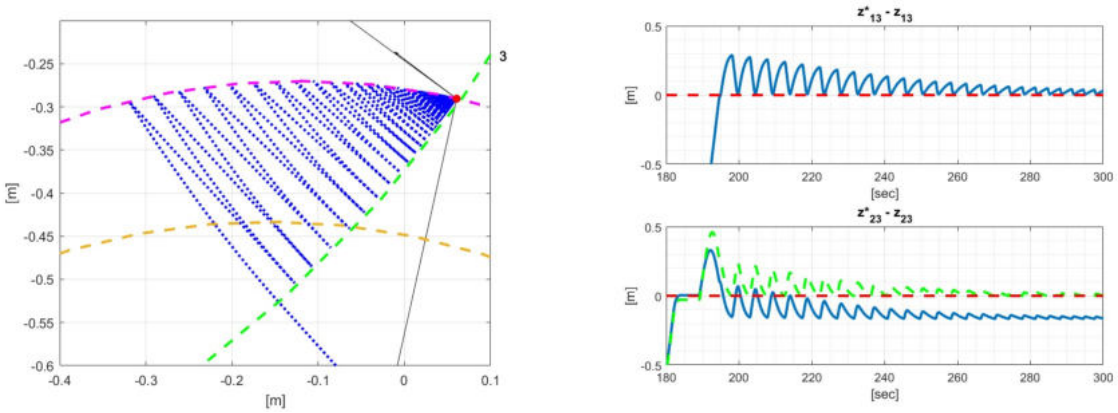


그림 3.17. 모드 알고리듬으로 인해 발생되는 진동현상

3.4.3 헤딩방향 불일치 알고리듬과 비교

제시한 알고리듬의 비교군으로 헤딩방향을 경사하강의 방향과 일치시키지 않고 리더를 찾으면 바로 편대제어 모드로 들어가는 알고리듬을 생각해볼 수 있다. 이러한 헤딩방향 불일치 알고리듬을 동일한 초기값에서 실행하면 그림 3.18처럼 비스듬하게 수렴하는 것을 확인할 수 있다. 하지만 이러한 방식은 다양한 초기값에서 수렴성을 보장하지 못할 수도 있다. 하나의 예로 에이전트의 위치와 헤딩방향의 초기값 $\mathbf{x}_1^T = [-1.65, -1.03, -0.36]$, $\mathbf{x}_2^T = [0.12, -0.72, 0.02]$, $\mathbf{x}_3^T = [-2.27, 2.41, -1.11]$와 원하는 상대거리$||\mathbf{z}_{12}^*|| = 0.82$,

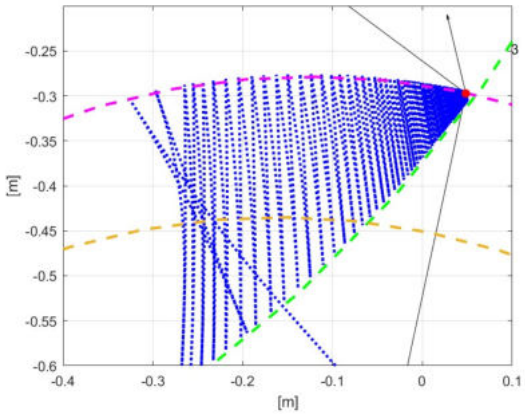


그림 3.18. 헤딩방향 불일치 알고리듬의 수렴 특성

$||\mathbf{z}_{13}^*|| = 5.56$, $||\mathbf{z}_{23}^*|| = 5.86$로 주어졌을 때, 두 평형 집합의 교집합에 수렴하지 못하고 극소점에 갖히는 것을 그림 3.19을 통해 확인할 수 있다. 헤딩방향 불일치 알고리듬의 국소점을 $x-y$평면에서 바라본 모습은 그림 3.21와 같다. 이는 헤딩방향을 일치시키지 않아서 삼각호를 그리며 이동하고 이 때문에 더이상 평형점을 향해 나아가지 못한다. 반면에 제안한 알고리듬은 문제없이 수렴하는 것을 그림 3.20통해 확인할 있다.

3.4.4 확장가능성과 유용성

본 논문이 가정한 최소 퍼시스턴트 그래프는 그 자체로 확장 가능성의 이점을 가지고 있다. 3대의 에이전트일때 가정되는 2번째 팔로워는 N대의 에이전트에 대해서 $N-1$번째 팔로워로 표현될 수 있으며, 이 에이전트의 조건은 두개의 간선을 가지는 것이다. 따라서 2개의 간선을 포함하는 에이전트가 추가되었을 때, 최소 퍼시스턴트의 토폴로지를 유지할 수 있다. 제시한 기법은 확장된 최소 퍼시스턴트 그래프에 바로 적용가능하다. 다만 추가된 에이전트가 얻는 정보가 2번째 팔로워와 같이 한 에이전트에 대한 상대거리 벡터와 다른 에이전트에 대한 상대거리 정보만을 획득할 수 있다면 부족한 측정치를 추정할 필요가 있다. 본 논문에서 제시한 거리기반 위치측위 기법이 그대로 사용될 수 있으나 $p.e.$

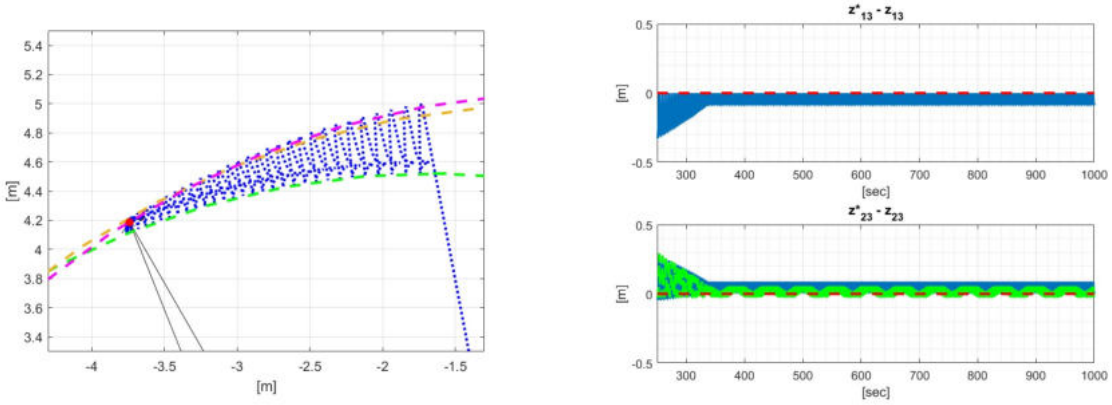


그림 3.19. 헤딩방향 불일치 알고리듬의 극소점

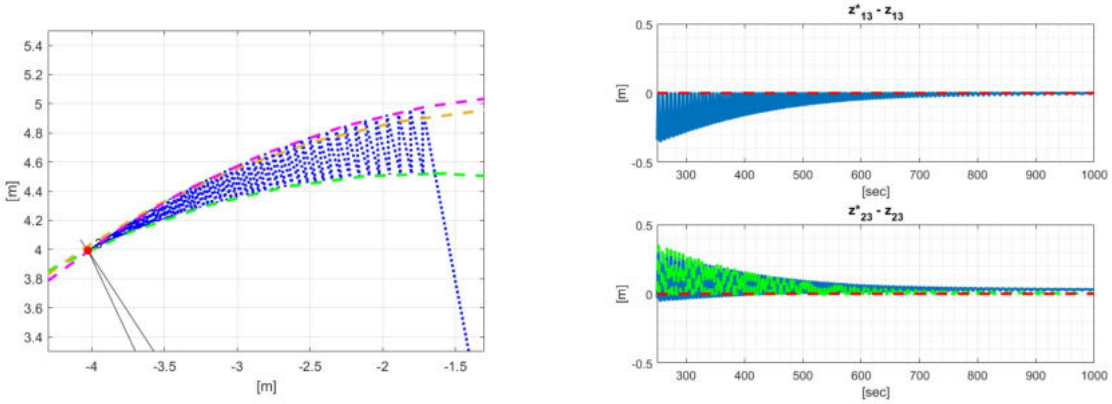


그림 3.20. 제안한 알고리듬의 수렴

입력을 수행하기 위한 공간이 확보되어야 한다. 만약 측정치가 부족한 에이전트의 수가 많아진다면 이를 추정하기 위한 움직임의 공간이 충분히 확보되어져야 할 것이다. 이를 위해 장애물 회피 기법이 추가로 고려되어야 한다. 최소 퍼시스턴스 그래프는 통신을 통해 에이전트간 어떠한 정보도 교환되지 않는다. 센싱 그래프 또한 단방향을 갖기 때문에 UWB와 같은 센서를 제외하면 피동센서로 리더를 관측한다. 따라서 리더를 탐지할 수만 있다면 다른 리더로 교체하는 것도 가능하다.

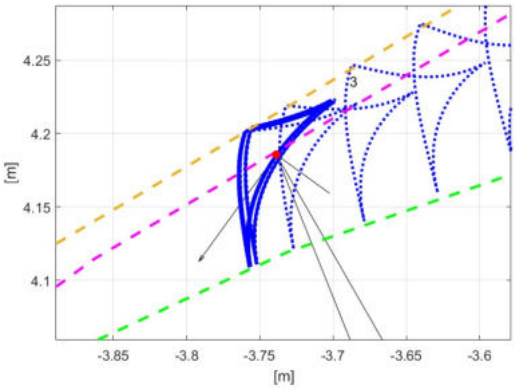


그림 3.21. 헤딩방향 불일치 알고리듬의 극소점

제 4장

실험

4.1 소프트웨어 플렛폼과 물리엔진 시뮬레이터

본 장에서는 실험에 사용된 소프트웨어 플렛폼과 물리엔진 기반 시뮬레이션을 소개한다. 각 에이전트의 다이나믹스를 고려한 시뮬레이션으로 다른 설계변수들을 정밀 조정하고 다양한 초기 위치 자세에서의 반응을 빠르게 파악할 수 있다.

ROS2: Robot Operating System 2

로봇 개발은 로봇에 해당하는 하드웨어를 선정하고 소프트웨어를 작성하는 전과정을 아우른다. 많은 경우, 특정 로봇에 특화된 기구들이 설계되고 맞춤 제작된다. 그렇기 때문에 로봇산업은 로봇의 종류나 제작회사에 따라 독자적인 개발방식을 가진다. 다양한 로봇을 사용하기 위해서는 그에 맞는 하드웨어나 소프트웨어적인 지식-특히나, 해당 로봇사의 API(Application Programming Interface)-들을 습득하는 것이 필수적이지만 이러한 사전지식을 습득하는데 많은 시간을 소요되고 로봇 개발을 지연시킨다. 이러한 문제를 해결하기 위해서 로봇 프로그래밍 개발에서 하나의 표준화된 프레임워크를 제시하는 오픈소스 플렛폼이 등장하기 시작하였고, 그 중 대표적이고 널리 사용되는 것이 ROS(Robot Operating System)이다.

ROS2는 ROS1의 장점을 유지하고 단점을 보완하여 ROS1이 탄생한 2007년의 10년 뒤에 출시되었다. 그림 4.1a는 ROS2의 로고이다. 로봇 어플리케이션을 개발하기 위해 사

용되는 소프트웨어 라이브러리와 툴을 모아놓은 것이다. ROS 상에서 하드웨어가 추상화되고 하위 디바이스 제어를 위한 프로그램들이 제공되기 때문에 개발자들-특히나 상위 제어기를 개발하는 개발자들-은 하드웨어에 대한 제어나 지식을 공부하지 않고 개발하고자 하는 분야에 몰두할 수 있게 되었다.

ROS에서는 노드(Node)라는 가장 작은 단위의 실행 프로그램을 연결하여 작업하기 때문에 협업에 유리하다. 노드들이 각자의 정보를 담은 메시지들을 주고 받으며 큰 단위의 프로그램을 구성하기 때문에 개발하고자 하는 노드만 개발자의 노드를 삽입하여 검증하기 매우 쉽다. 또한 이러한 노드는 재사용성과 확장성이 매우 뛰어나다. 특히나 노드들은 서로 정해진 규약을 따르는 메시지만 주고받으면 되므로 노드가 작성되는 프로그래밍 언어의 제약도 해소되었다.

ROS2에서는 데이터 분산화 서비스(DDS, Data Distribution Service)라는 데이터 통신 방식을 채택하여 그 범용성과 편리성을 극대화 시켰다. 데이터 분산화 서비스란 OMG(Object Management Group)라는 비영리단체에서 정의한 데이터 통신의 미들웨어이다. 로봇 개발에 있어서 수많은 데이터를 주고 받는 문제는 간단한 일이 아니다. ROS2에서는 데이터 분산화 서비스라는 미들웨어를 통해 일정한 성능을 보장하는 통신 미들웨어를 제공하여 로봇 개발자에게 통신에 대한 부담을 해소해주었다. [23]에서 ROS2에서 DDS의 성능에 대한 분석이 이루어졌다. ROS2의 자세한 구조나 상용에 대해서는 [24]에서 제공하고 있다. 이러한 장점들로 인해 거대한 오랜시간 커뮤니티가 유지되어져 왔고 다양한 오픈소스 패키지나 개발 도구들을 손쉽게 이용할 수 있다.

(a) ROS1과 ROS2 로고

(b) 위봇 로고

그림 4.1. ROS2와 위봇의 로고

Webots: 물리 엔진 기반 시뮬레이터

위봇(Webots)은 물리 엔진 기반 시뮬레이터 중 하나로, 뛰어난 그래픽과 직관적인 UX/UI를 가지고 있으며 다개체 시스템에 적용하기 편한 것이 특징이다. ODE(Open Dynamics Engine) 기반의 물리 엔진이 구축되어 있다. 그림 4.1b은 위봇의 로고가 도시되어 있다.

위봇은 연구개발에 널리 사용되는 구동기와 로봇들을 다수 제공하여 추가적인 모델링 없이 간편하게 사용할 수 있다는 장점이 있다. 본 논문에서 사용될 크레이지 플라이와 터틀봇 또한 모델을 제공한다. 위봇을 ROS2에서 사용하기 위해서는 위봇 데이터를 ROS2 데이터로 출판(publish) 해주어야하고 ROS2에서 출판된 메시지들을 위봇 시뮬레이터에 전송해주는 작업이 필요하다. 이러한 작업을 위해 Webot-ROS2 드라이버 패키지가 제공된다.

실제 하드웨어 실험에 앞서 위봇과 같은 물리엔진 기반 시뮬레이션을 수행하는 것은 매우 중요하다. 소프트웨어가 잘 동작하는지에 대한 검증을 시뮬레이션 상에서 미리 하여 시간을 절약하고 예상치 못한 동작에 미리 대비가 가능하기 때문이다. 또한 메트랩 시뮬레이션에서 로봇의 동역학을 배재한 상황에서 설정한 설계 파라미터들이 유효한지에 대한 검증도 가능하다.

그림 4.2. 위봇 시뮬레이션 월드와 에이전트

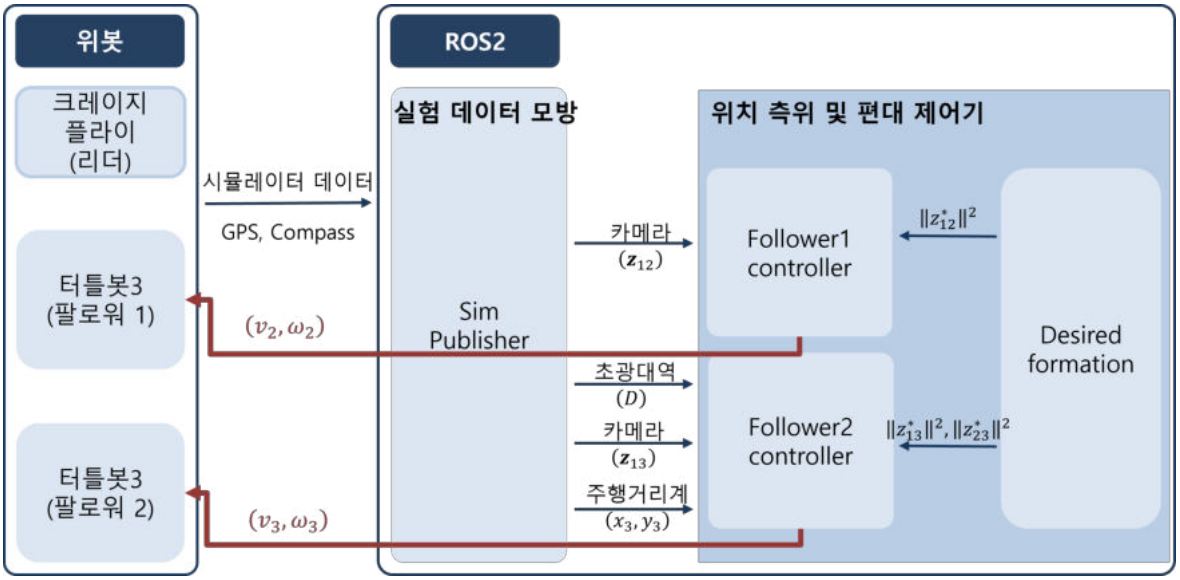


그림 4.3. 위봇과 ROS2 노드의 데이터 교환 흐름

위봇의 월드파일(world file)은 시뮬레이션의 배경이나 구조물 그리고 로봇들을 배치하는 정보를 포함한 파일이다. 그림 4.2은 원하는 시뮬레이션 환경과 로봇들을 배치하여 구축한 모습이다. 바닥의 격자 무늬는 1미터 간격으로 설정하여 로봇의 위치를 파악하기 쉽게 하였고, 본 논문에서는 장애물 회피에 대한 부분에 대해서는 다루지 않기 때문에 불필요한 구조물을 배치하지 않았다.

4.1.1 시뮬레이터 기반 소프트웨어 검증 및 설계변수 미세조정

그림 4.3은 위봇과 ROS2를 활용한 시뮬레이션의 데이터 교환을 도식화한 모습이다. 위봇을 그림 4.2의 월드파일과 함께 실행하게 되면 월드파일에 설정된 로봇의 초기 위치 및 자세로 생성(spawn)된다. 위봇에서의 로봇과 환경의 정보들은 위봇-ROS2 드라이버를 통해 ROS2의 메시지 형태로 출판된다. 위봇 월드에서의 참값을 받기 위해서는 감독권한(Supervisor)을 활성화 시켜 얻어낼 수 있지만 측정 잡음이 포함된 센서 데이터를 획득하여 좀더 신뢰성 있는 설계변수조정 과정을 거치도록 한다. 시뮬레이션 상에서 로봇에 부착된 GPS(Global Positioning System)과 나침반(Compass)센서는 위봇의 전역 좌표계 상의 정보를 제공한다.

Sim Publisher 실제 실험에서 획득될 데이터를 시뮬레이션 값으로 모방하는 역할을 한다. 위봇 시뮬레이터의 센서 정보를 모아 편대 제어기에 사용될 값으로 계산하여 출판한다. 실제로 물리엔진 기반 시뮬레이터에서 스테레오 카메라 모델을 사용하여 이를 통해 얻어진 카메라 프라엠의 이미지로 객체 탐지 딥러닝 모델을 학습시켜 구현하는 경우도 있지만 이는 실제 환경에서 다시 학습시켜야하는 번거로움이 있다. Sim Publisher에서 출판되는 메시지는 스테레오 카메라를 통해 획득되는 i번째 팔로워와 리더간의 상대 거리 벡터 정보($\mathbf{z}_{1i}$), UWB에서 획득되는 팔로워2와 팔로워1간의 상대 거리 정보(D), 그리고 팔로워1의 위치 측위시에 사용되는 팔로워 2의 주행거리계(x_3, y_3) 정보이다.

위치 측위 및 제어파트에는 3개의 노드로 구성되어 있다. 첫번째로 Desired formation 노드는 원하는 위치($\mathbf{p}^*$)의 값을 인가 받으면 $||\mathbf{z}_{ij}^*||$ 값을 계산하여 출판한다. Follower1 controller에서는 $||\mathbf{z}_{12}^*||$ 값과 리더와의 상대거리 및 상대 각도를 가지고 받아서 그림 3.8의 알고리듬을 수행한다. 이때 산출되는 제어입력 v_2, ω_2이 위봇의 팔로워1에 해당하는 터틀봇에게 인가되어 해당 입력에 맞게 로봇이 동작하게 된다. Follower2 controller는 Desired

변수	값
$\|\|\mathbf{z}^*_{12}\|\|$	1.42
$\|\|\mathbf{z}^*_{13}\|\|$	1.11
$\|\|\mathbf{z}^s_{23}\|\|$	1.47
$\mathbf{x}_1$	$[-0.85, -0.89, 0.43]^T [m, m, rad]$
$\mathbf{x}_2$	$[0.06, 0.56, -0.19]^T [m, m, rad]$
$\mathbf{x}_3$	$[1.57, -0.54, -3.07]^T [m, m, rad]$
T_1, T_2, T_l	$0.001, 0.05, 180$
T_{cl}, T_{cf}	$0.1, 0.1$

표 4.1. 위봇 시뮬레이터에서 사용된 초기 위치 및 자세와 설계변수

formation 노드에서 $||\mathbf{z}^*_{13}||, ||\mathbf{z}^*_{23}||$을 입력받는다. 또한 Sim Publisher에서 센서값들을 인가받아 그림 3.9의 알고리듬을 수행하여 산출되는 v_3, ω_3이 시뮬레이터의 팔로워2에 인간되어 동작하게 된다.

시뮬레이션에서 활용된 파라미터들은 표 4.1에 기재되었다. 이외의 변수들은 이전 메트랩 시뮬레이션에서 사용한 변수 표 3.1과 동일한 값이 사용되었다.

그림 4.4은 물리엔진 시뮬레이션에서 에이전트가 움직인 궤적을 도시한다. 팔로워2가 위치 측위를 하기 위해 움직인 궤적의 차이가 모바일 로봇의 다이나믹스가 반영되어 커진 것을 볼 수 있다. 메트랩 시뮬레이션 상에서는 그 크기가 매우 작았다. 상대거리 값을 도시한 그림 4.5을 보면 각 상대거리를 만족하여 정지한 것을 확인할 수 있다. 위봇 시뮬레이션의 궤적 영상은 https://youtu.be/_r9HwDL_zFQ에 데이터 영상은 https://youtu.be/wXaGDfnMvO0에 업로드 되었다.

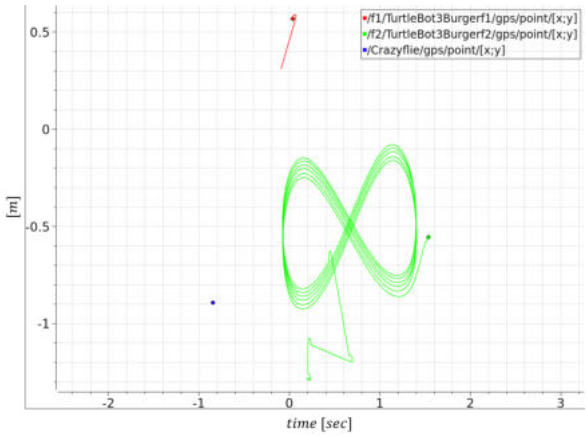
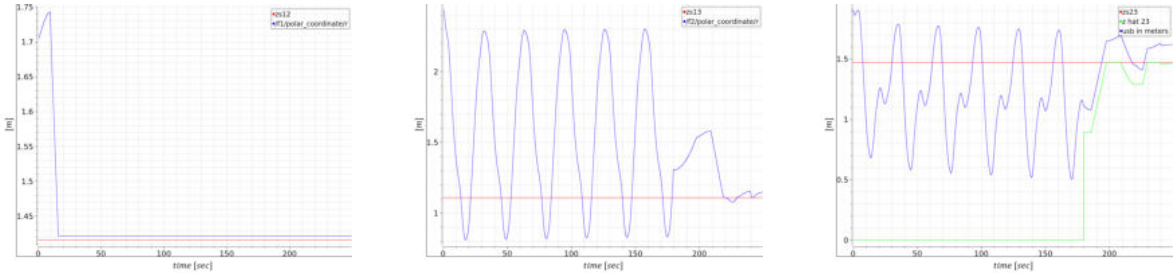


그림 4.4. 위봇 시뮬레이션 환경에서의 에이전트 궤적



(a) 위봇에서 리더와 팔로워1 간의 상대 거리

(b) 위봇에서 리더와 팔로워2 간의 상대 거리

(c) 위봇에서 팔로워1과 팔로워2간의 추정 상대거리와 UWB값

그림 4.5. 위봇 시뮬레이션 환경에서의 에이전트간 상대거리

4.2 실험을 위한 하드웨어 구축

그림 3.1의 편대를 이루기 위한 에이전트와 에이전트에 부착된 센서들을 소개한다.

4.2.1 터틀봇3

팔로워1과 팔로워2는 로보티즈사의 터틀봇3을 이용한다. 터틀봇3은 ROS2 터틀봇3 패키지를 제공하여 ROS2 환경에서 손쉽게 구동과 개발이 가능하다. 로보티즈사의 많은 제품들이 ROS철학에 기반하여 오픈소스 하드웨어 및 소프트웨어를 제공한다. 터틀봇3에 부착되는 제어기 보드와 터틀봇3의 외관등 모두 오픈소스로 무료배포되고 있다.

특징	값
최대 병진 속도	0.22 m/s
최대 회전 속도	2.84 rad/s (162.72 deg/s)
최대 유효 탑재량	15kg
크기 (가로 x 세로 x 높이)	138mm x 178mm x 192mm

표 4.2. 터틀봇3 크기와 최대 성능

터틀봇3의 외관적 특징과 성능은 표4.2에 요약되어 있다. 또한 실험 전략에 맞춰 새롭게 구성한 터틀봇3 내부 요소들은 표 4.3에 기재되었고 그림 4.6에 도시되었다.

단일 보드 컴퓨터 - Jetson Nano

기본적인 터틀봇3의 단일 보드 컴퓨터로는 라즈베리 파이(Raspberry PI)가 사용된다. 본 논문에서는 카메라의 효율적이고 빠른 처리를 위해 그래픽 카드가 내장된 젯슨 나노(Jetson Nano) 보드를 사용한다. 젯슨 나노 보드의 GPU(128-core Maxwell)에 하드웨어 비디오 인코더 디코더가 내장되어 있어서 CPU의 부담을 덜어주고 비디오 인코딩 및 디코딩시 빠른 성능을 보인다. 나노의 비디오 인코딩 디코딩의 해상도 별 수행시간은 표 4.4에 정리되었다.

카메라는 젯슨 나노와 USB로 통신한다. 카메라 프레임 정보는 와이파이 스트리밍 방식으로 지상 관제 컴퓨터로 전송된다. 표 4.4에 따라 4x 1080p 해상도에 30Hz로 전달된다.

젯슨 나노 보드는 AI를 개발하기 위해 주로 사용된다. 본 논문에서 Nano 보드위에서 객체 탐지 딥러닝과 ROS2 미들웨어 등을 모두 탑재하려 시도하였으나 CPU 및 GPU의 사양 부족으로 인해 시스템 지연이 크게 발생하였다. 젯슨 개열의 컴퓨팅 보드는 그 성능이

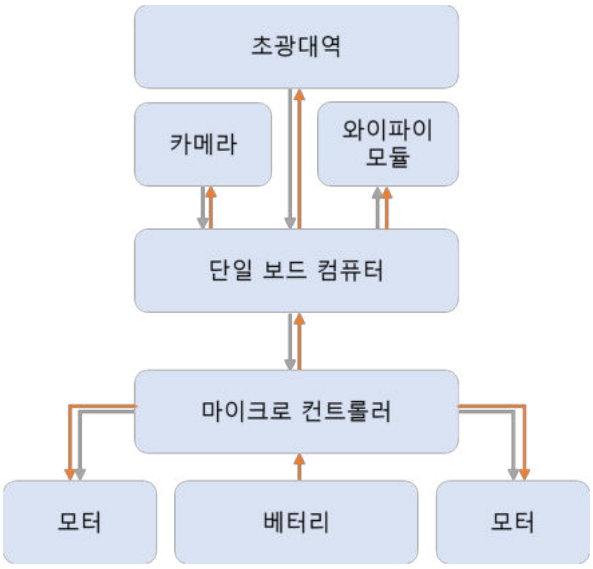


그림 4.6. 터틀봇3의 부품 개략도

올라가면서 가격이 기하급수적으로 증가하기 때문에 무리하게 성능을 높혀 알고리듬을 검증하는 것은 적절하지 못하다고 판단하였다. 따라서 지상 관제 컴퓨터가 에이전트의 데이터를 넘겨받아 메인 알고리듬을 수행하고 에이전트의 단일 보드 컴퓨터는 지상 관제 컴퓨터와의 통신과 편대 제어기에서 산출된 제어 입력을 수행한다..

마이크로 컨트롤러와 관성 측정 장치 - OpenCR 1.0

마이크로 컨트롤러인 OpenCR 1.0은 CPU로 ARM Cortex-M7이 사용되었고 관성 측정 창치인 3축 각속도센서와 3축 가속도센서인 ICM-20648가 내장되어 있다. 로보티즈사의 구동모터인 다이나믹셀(DYNAMIXEL)을 구동시키는 역할을 한다. 마이크로 컨트롤러의 내부에서는 다이나믹셀 SDK(software Development Kit)와 아두이노 라이브러리를 기반으로 한 제어기 코드가 동작한다. 단일 보드 컴퓨터는 지상 관제 컴퓨터에서 최종 제어 명령을 전달받아 마이크로 컨트롤러로 전달한다. 마이크로 컨트롤러는 요청받은 명령을 모터에 인가하여 모터가 구동된다.

종류	제품명
단일 보드 컴퓨터 (Single Board Computer)	Jetson Nano
마이크로 컨트롤러 (Micro Controller Unit)	OpenCR 1.0
스테레오 카메라	ZED
관성 측정 장치 (Inertial Measurment Unit)	ICM-20648
초광대역 센서 (Ultra Wide Band)	pozyx Tag or Anchor
와이파이 모듈	iptime A3000UA
모터	DYNAMIXEL(XL430-W250)
베터리	Lithium polymer 11.1V 1800mAh

표 4.3. 터틀봇3의 구성요소

카메라 - ZED

스테레오랩스(Sterolabs)사의 ZED 카메라는 스테레오 비전을 활용한 뎁스 카메라로 상당히 정밀한 포인트 클라우드를 제공해 준다. 스테레오랩스에서 제공하는 딥러닝 객체 탐지 모델이 존재하나 본 논문에서는 추가 학습과 관련하여 연구실에 자료가 있는 Yolov8을 사용하였다. 카메라 센서는 객체 탐지 딥러닝 모델에 카메라 프레임을 제공한다. 객체 탐지 모델이 카메라 프레임 내에 객체를 탐지하여 바운딩박스(bounding box)의 픽셀 정보를 넘겨주면 ZED는 그 영역의 깊이정보 및 포인트 클라우드 정보를 추출한다. 이 정보는 편대제어에서 리더와 팔로워 간의 상대거리 및 상대 각도 정보로 사용된다.

종류	해상도 @ FPS(Frame Per Seconds) (압축)
비디오 인코드	4K @ 30 — 4x 1080p @ 30 — 9x 720p @ 30 (H.264/H.265)
비디오 디코드	4K @ 60 — 2x 4K @ 30 — 8x 1080p @ 30 — 18x 720p @ 30 (H.264/H.265)

표 4.4. 젯슨 나노 보드의 하드웨어 인코더, 디코더의 해상도별 FPS 성능

초광대역센서 - pozyx

포직스(pozyx)사의 초광대역 센서는 앵커(Anchor, transmitter)와 태그(Tag, receiver)로 구성되어져 있다. 앵커가 태그의 정보를 받아 두 장치 간의 거리를 산출한다. UWB는 아두이노 우노(Arduino Uno)와 동일한 핀맵을 공유하고 아두이노 SDK를 제공해주어 사용자가 손쉽게 프로그래밍 하여 사용할 수 있다.

팔로워 1에는 초광대역 앵커가 팔로워 2에는 초광대역 태그가 부착된다. 그림 4.8a와 그림 4.8c와 같이 태그와 수신기는 로봇의 위치나 자세에 영향을 받지 않기위해 가장 위에 배치되었다. 초광대역 센서는 팔로워간 상대거리를 측정하는데 사용된다.

와이파이 모듈 - iptime AC3000UA

iptime AC3000UA는 5GHz와 2.4GHz 듀얼벤드를 사용한다. 실험에서는 빠른 데이터 송수신을 위해 5GHz 대역의 밴드가 사용되었고 이 밴드에서는 무선랜 구축 기술의 규격인 802.11ac가 사용되어 866Mbps의 통신속도를 가진다. USB3.0을 통해 단일 보드 컴퓨터와 통신한다. 터틀봇3에서의 모든 데이터가 와이파이 통신을 통해 지상 관제 컴퓨터로 전송된다.

모터 - DYNAMIXEL, XL430-W250

로보티즈사의 다이나믹셀 XL430-W250 모터는 마이크로 컨트롤러(ARM CORTEX-M3 (72 [MHz], 32Bit))와 자사의 제어기가 탑재되어 있다. 다른 로보티즈 제품과 다르게 오픈 소스로 공개되어 있지 않다. 비접촉식 절대 엔코더(contactless absolute encoder)가 피드백 정보를 측정한다. 사용자의 설정에 따라서 속도 제어 모드, 위치 제어모드$(0\ 360\ \mathrm{deg})$,PWM 제어 모드(전압 제어 모드)등 을 제공한다. 편대 제어기에서 산출되는 터틀봇3의 선속도 및 회전 속도 값은 터틀봇3의 하드웨어 크기와 차동 구동 기구학(differential drive kinematics)를 고려하여 각 모터의 회전 속도가 결정된다. 각 바퀴에 알맞은 속도 명령을 마이크로 컨트롤러인 OpenCR이 받아 모터에 명령을 인간한다. 모터의 내부 제어기가 속도 명령에 알맞은 속도를 내기위해 PID제어기가 작동하고 터틀봇3의 선속도와 회전속도가 제어된다.

베터리

베터리는 터틀봇3 패키지에서 제공하는 리튬 폴리머 11.1V 1800mAh를 사용하며 마이크로 컨트롤러인 Open CR 1.0에 연결된다. OpenCR은 베터리로 부터 전력을 인가받아 단일 보드 컴퓨터와 모터에 전력을 분배하는 역할을 수행한다. 최대 구동시간은 약 2시간 30분이며 예상 충전시간은 이와 동일하다.

4.2.2 크레이지 플라이

크레이지 플라이(Crazyflie)는 손바닥만한 크기에 27g의 작은 무게인 다용도 오픈소스 비행 개발 플렛폼이다. 다양한 확장덱(extension deck)을 장착하여 추가적인 센서(초광대역, 옵티컬 플로우, AI 덱을 활용한 카메라 등)나 기능(LED, 높은 추력의 모터 등)들을 추가

(a) 크레이지 플라이 정면 사진

(b) 크레이지 플라이 평면 사진

그림 4.7. 리더, 옵티컬 플로우 덱을 장착한 붉은 색으로 도색된 크레이지 플라이의 정면 및 평면 사진

할 수 있다. 크레이지 플라이 또한 ROS2 환경에서 손쉽게 구성할 수 있도록 CrazySwarm2라는 패키지가 존재한다. 이는 다수의 크레이지 플라이fmf 동작하기에 적합하게 설계되어 있다.

본 논문에서는 크레이지 플라이가 리더의 역할을 수행하게 된다. 리더는 어떠한 변의 길이를 유지할 책임이 없기 때문에 자유롭게 움직이는 것이 가능하지만, 본 논문에서는 호버링(hovering)이외의 동작은 수행하지 않는다. 다른 측위 장치 없이 옵티컬 플로우 덱을 활용하여 스스로 자신의 로컬 좌표계 기준으로 움직일 수 있다.

실험실 환경과 크레이지 플라이의 비슷한 색을 가져 객체 탐지 시에 불리하게 작용한다. 그림 4.7과 같이 겉 표면에 붉은 색 락카를 도색하여 객체 탐지에 유리하게 하도록 하였다.

앞서 소개한 팔로워1, 팔로워2의 역할을 하는 터틀봇3과 리더 역할을 하는 크레이지 플라이의 실제 사진을 그림 4.8에 도시하였다.

종류	[회사] 제품명
CPU	[AMD] 라이젠 9 버미어 5900X
메인보드	[MSI] MAG B500M 박격포
메모리	[삼성] DDR4 8G PC4-25600 [2EA]
GPU	[MSI] GeForce RTX 3080Ti
파워	[FSP] Hydro G Pro 850W 80 plus
와이파이	iptime A2004MU

표 4.5. 지상 관제 컴퓨터 주요 부품 및 와이파이 모듈

4.2.3 지상 관제 컴퓨터

젯슨 나노의 사양 부족으로 인해 본 논문에서 제시한 주요 알고리듬은 지상 관제 컴퓨터에서 계산된다. 딥러닝 기법과 카메라 데이터와 같이 고성능 연산장치를 요구하는 작업을 처리하는게 주요 역할이다. 지상 관제 컴퓨터에는 고성능의 CPU와 GPU가 탑제되어 있다. 지상 관제 컴퓨터는 와이파이 통신을 통해 각 에이전트와 통신하며 데이터를 주고 받기 때문에 와이파이 모듈인 iptime A2004MU가 사용된다. 팔로워 에이전트가 사용하는 iptime A3000UA와 동일한 스펙을 가지고 있으며 복수의 기기가 동시에 무선랜 접근시 속도저하를 감소하는 기술(MU-MIMO)이 탑재되어 다수의 에이전트와 통신할때 좋은 성능을 발휘한다. 지상 관제 컴퓨터의 주요 부품과 와이파이 모듈은 표 4.5에 기제되었다.

4.2.4 시스템의 전체 통신

실험의 전체적인 흐름은 그림 4.9에 도시되었다. 팔로워1과 팔로워2는 ROS2의 터틀봇3 패키지를 통해서 자신의 정보들을 DDS 공간 상에 메시지 형태로 전송한다. 같은 와이파이에 연결되어 있는 지상 관제 컴퓨터는 팔로워들과 도메인 아이디를 공유하여 해당 메시지들에 접근할 수 있다. 팔로워1과 팔로워2는 자신의 카메라 정보를 와이파이 스트리밍을 통해서 지상 관제 컴퓨터에 전송한다. 즉, 지상 관제 컴퓨터는 팔로워들의 카메라 정보와 센서 정보를 와이파이 통신을 통해 모두 접근할 수 있다. 받은 정보를 기반으로 객체 탐지와 데이터 후처리를 수행한다. 본 논문에서 제시한 위치 측위 및 편대 알고리듬을 계산하여 산출된 각 팔로워의 제어입력은 DDS에 메시지 형태로 보내진다. 각 팔로워는 자신에 해당하는 메시지를 받아 명령을 수행하게 된다.

리더는 호버링이외에 다른 동작을 수행하지 않으므로 굳이 프로그래밍으로 구현하지 않고 수동으로 명령창을 이용하여 이착륙 한다. 리더는 cfclient라는 지상 관제 프로그램을 통해 버튼으로 간단하게 이착륙 및 이동할 수 있다.

4.3 ROS2 런치 파일 및 실험 진행 방법

그림 4.9을 ROS2로 구현하기 위한 런치파일(launch file)과 그 내용을 소개한다. 그리고 전체 실험 진행 순서를 소개한다.

4.3.1 ROS2 런치 파일 구성

실험의 진행 과정에 대해 알아보자. ROS2에서는 런치파일이라는 묶음 실행 파일을 제공한다. 이 런치파일에는 다수의 노드나 다른 런치 파일을 실행할 수 있고 배쉬파일(bash shell script file)도 실행할 수 있다.

팔로워1과 팔로워2의 컴퓨터 보드에는 하나의 런치파일과 하나의 배쉬파일로 구성된 런치파일이 존재한다. 런치파일은 기존의 ROS2에서 제공하는 $turtlebot3\_bringup$ 패키지의 $robot.launch$에 추가적인 노드와 배쉬파일을 추가한 것이다. 이 런치파일은 크게 3가지 역할을 한다. 하나는 마이크로 컨트롤러의 포트를 인식하고 통신하여 상태 값을 읽고 모터에 제어명령을 쓰는 역할이다. 다른 하나는 터틀봇3의 모델과 관성측정장치의 값을 기반으로 주행거리계를 계산한다. 마지막으로 rviz와 같은 ROS2 시각화 툴을 위한 로봇 기술서(robot description)와 상태값을 출판한다.

배쉬파일은 ROS2 노드가 아닌 파이썬(python) 파일을 실행시키는 역할을 한다. 이 파이썬 파일은 ZED SDK로 와이파이를 통해 카메라 프레임을 스트리밍하는 코드이다. 입력변수로 지상관제 컴퓨터의 ip주소와 원하는 해상도와 FPS가 존재하고, 배쉬파일에는 각 변수에 맞게 파이썬 파일을 실행하는 명령어가 기술되어 있다.

팔로워2의 경우 초광대역 센서의 포트를 인식하고 이와 연결하여 ROS2 데이터로 출판하는 노드가 런치파일에 추가적으로 포함되게 된다.

런치파일에서 출판되는 모든 메시지에는 에이전트에 해당하는 네임스페이가 추가되어 구분할 수 있게 한다. 팔로워1은 /f1, 팔로워2는 /f2의 네임스페이스를 가진다,

지상관제 컴퓨터에서는 두 개의 런치 파일이 존재한다. 하나는 $formation\_experiment$ 패키지의 $formation_experiment.launch$파일이다. 이는 팔로워별 센서 데이터 취득 후 후처리하는 노드를 묶은 런치 파일이다. 각 노드는 자신에 해당하는 네임스페이스의 메시지를 취득해 다음을 수행한다. ZED 카메라 스트리밍을 받고 이를 객체 탐지 딥러닝에 입력한다. 그 결과로 객체 탐지가 되면 바운딩 박스를 획득하고 아니면 바운딩 박스는 빈(empty)값이 된다. 획득한 바운딩 박스의 중앙 픽셀 값의 깊이 정보와 포인트 클라우드 정보를 ZED SDK를 이용하여 추출한다. 편대 제어기에 필요한 변수들을 출판한다. 이는

그림4.9의 카메라 스트리밍 수신과 객체 탐지 딥러닝 및 센서 후처리에 해당한다.

다른 런치파일은 그림 4.9에서 위치 측위 및 편대제어 알고리듬에 해당하는 역할을 수행하며 그림 4.3에서 사용된 런치파일이 동일하게 사용된다. $formation\_controller$ 패키지의 $formation_controller.launch$파일이다.

실험 진행 순서

팔로워1과 팔로워2는 지상 관제 컴퓨터의 와이파이 라우터의 무선랜에 자동으로 접속하며 고정된 아이피를 할당받도록 설정되어 있다.

지상 관제 컴퓨터에서 ssh(secure shell, 리눅스 원격접속 프로그램)을 통해서 할당된 아이피와 비밀번호를 통해 팔로워1과 팔로워2의 단일 컴퓨터 보드에 원격접속할 수 있다.

접속 후 $robot.launch$파일을 실행시키면 ZED 카메라가 지상 관제 컴퓨터에 스트리밍을 시작하고 ROS2의 DDS에 메시지들을 출판한다.

지상 관제 컴퓨터에서 ZED 스트리밍과 팔로워1과 팔로워2의 터틀봇3 메시지들을 받아 편대 제어기에 필요한 데이터를 출판하는 $formation_experiment.launch$를 실행한다. 모든 통신이 정상 작동하는지 확인 후 위치 측위 및 편대 제어 알고리듬을 수행하는 $formation_controller.launch$을 실행한다.

4.4 실험 결과 및 분석

그림 4.10은 실험 장소와 에이전트들의 초기위치를 보여준다. 각 초기위치는 위봇 시뮬레이터에서 진행한 초기 위치와 자세는 유사하게 배치되었다. 실험 영상은 https://youtu.be/VK1IEnrBT2A에 데이터 영상은 https://youtu.be/Xc73hQO0ur0에 도시되었다. 실험환경에서는 글로벌 좌표계에서의 위치를 측정할 수 있는 장비의 부

재로인해 위봇과 같은 위치의 궤적을 획득하기 어렵다. 하지만 분산적 삼각 편대제어의 특징으로 각 변의 길이가 원하는 길이로 수렴하는 것만 관찰하면 된다. 그림 4.11는 실험에서 획득한 각 에이전트 별 상대거리를 도시하고 있다. 기준값을 추종하여 수렴하는 것을 확인할 수 있다. 그림 4.11a와 그림 4.11b의 상대거리는 제드 카메라의 깊이 정보를 통해 얻어낸 생대거리 정보이다. 그림 4.11c에서는 초광대역 센서를 통해 관측한 팔로워1과 팔로워2 간의 상대거리와 거리기반 적응형 위치 측위 기법을 통해 얻은 추정된 상대거리를 도시하고 있다. 일정 시간동안의 추정 오차가 발생하였지만 근사한 값을 가지는 것을 확인할 수 있다.

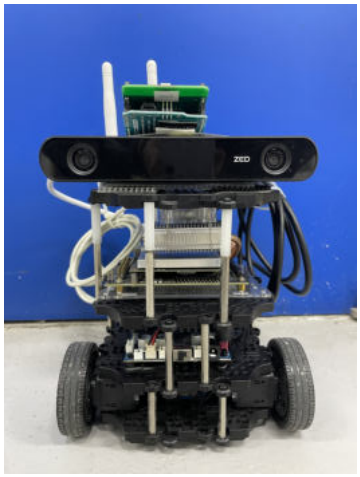
4.5 한계점 및 향후 계획

다양한 제약조건을 가지는 편대제어를 단순한 형태로 변환하여 문제를 해결하고자 하였기 때문에 그 한계점 또한 명확하다. 먼저 부족한 측정치를 추정하는 과정에서 상당한 시간이 소요된다 단점이 있다. 이는 설계변수인 추정 시간 의해 조절될 수 있지만 위치 측위에 충분한 시간이 소요되지 않는다면 충분히 정확한 추정치를 기대할 수 없다. 추정 시간을 줄이기 위해 학습률 매개변수가 조정될 수 있지만, 이는 무한정 키울 수 없고 센서의 노이즈가 존재하는 상황에서 적응형 상수를 과감하게 키우는 방법은 신중한 고려가 필요하다. 두번째는 간선의 길이를 조절할 책임이 있는 팔로워 에이전트는 많은 모드를 가지며 이를 판단하기 위해 많은 문덕값들이 존재한다는 것이다. 이러한 문턱값은 실제 노이즈와 바이어스 값에 의해 신중하게 조정되어져야 하는데 이러한 매개변수들이 많은 것은 다수의 매개변수 조정 과정을 거치게 한다. 마지막으로 초기값의 위치에 따라 두번째 에이전트의 수렴속도가 느리게 작용할 수 있다. 이는 리더와 팔로워2 그리고 팔로워1의 추정된 위치와 팔로워2의 평형 집합의 모양에 따라 결정된다. 이러한 원의 반지름이

서로 비슷할 수 록 수렴속도는 느려진다. 경사하강의 방향이 헤딩방향에 수직방향으로 일치되기 때문에 그 각도가 평형 집합의 교집합으로 수렴하는 일종의 학습률이된다. 향후에 연구로 세가지를 생각해 볼 수 있다. 첫번째로, 위치측위의 경우 팔로워1이 정지해 있지 않고 움직이는 상황일때도 팔로워1의 위치를 추정할 수 있다면 편대제어가 빠르게 수렴 가능하다. 두번째로, 편대제어 자체에 대해서는 경사하강 방향과 헤딩방향을 일치시키는 것은 느린 수렴성을 초래하기 때문에 빠른 수렴성을 유도할 수 있는 일정한 각도를 찾는 것이다. 마지막으로, 정지해 있는 리더가 아닌 경우에 통신을 통해 리더의의 정보를 팔로워가 전달받아 제시한 논홀로노믹 제어기를 변형할 수 있다. 이렇게 되면 통신 복잡도에 따른 문제점을 발생할 수 있지만 리더가 움직이는 상황에서도 팔로워가 일정한 대형을 이룸을 보일 수 있다.

(a) 팔로워 1, 초광대역 앵커를 부착한 터틀봇3의 정면 사진

(b) 팔로워 1, 초광대역 앵커를 부착한 터틀봇3의 평면 사진



(c) 팔로워 2, 초광대역 태그를 부착한 터틀봇3의 정면 사진

(d) 팔로워 2, 초광대역 태그를 부착한 터틀봇3의 평면 사진

그림 4.8. 터틀봇3의 정면 및 평면 사진

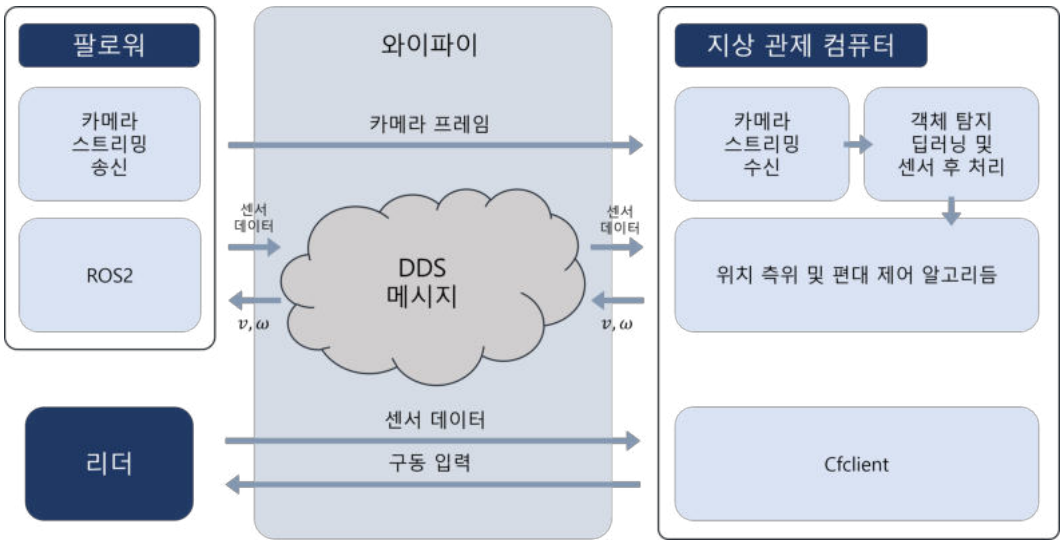


그림 4.9. 전체 시스템의 통신 흐름도

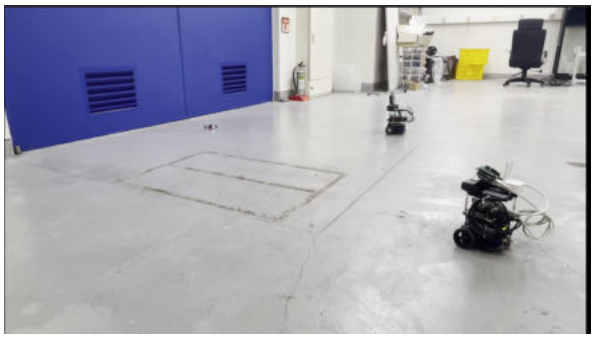
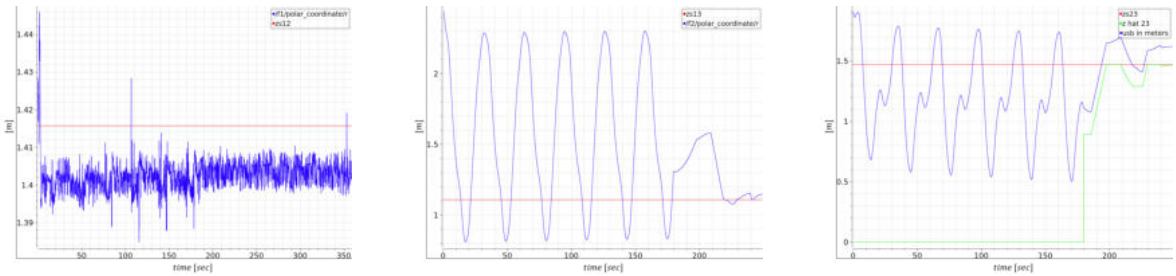


그림 4.10. 실험 장소 및 에이전트의 초기 위치 및 자세



(a) 실험에서 리더와 팔로워1 간의 상대 거리

(b) 실험에서 리더와 팔로워2 간의 상대 거리

(c) 실험에서 팔로워1과 팔로워2간의 추정 상대거리와 UWB값

그림 4.11. 실험 환경에서의 에이전트간 상대거리

제 5장

결론

본 논문에서는 비순환 최소 퍼시스턴스 그래프의 경사하강기반 편대제어에 대해서 다루었다. 에이전트가 운동학적 제약조건을 가지고 센서에 가시성 제약조건이 추가될 경우, 그리고 두번째 팔로워가 팔로워1의 거리정보만을 관측할 수 있을 때, 편대제어의 수렴성을 보장하기 어렵게 한다.

본 논문에서는 3대의 에이전트의 비순환 최소 퍼시스턴스 그래프의 정사영 기법을 통해 운동학적 제약조건을 해결한다. 또한 두번째 팔로워의 경사하강 기법이 리더와 첫번째 팔로워 정보의 합이라는 사실에 기반하여 각 에이전트를 통해 산출된 개별 입력을 모드로 나누어 인가하여 가시성 제약조건을 만족하면서 원하는 평형점에 도달할 수 있음을 보였다. 마지막으로 거리기반 경사하강 적응 법칙을 통해 위치 측위 알고리듬을 사용하여 센싱 변수가 부족한 상황에서도 경사하강 기법을 그대로 사용하는 방법을 제시하였다.

제시한 알고리듬은 위치측위에 가용되는 시간 소요가 크다는 점과 모드를 변환하기 위한 판단에 다양한 문턱값이 존재하여 이러한 문턱값을 미세조정하는 과정이 요구된다는 것이다. 하지만 실제 편대제어에서 가질 수 있는 다양한 제약조건들을 반영하여여 전체 알고리듬으로 반영하였고 이는 시뮬레이션과 실험을 통해 성능이 검증되었다.

제시한 알고리듬의 대표적인 한계인 느린 수렴속도는 향후 연구를 통해 개선될 수 있다.

참 고 문 헌

1. W. Ren and N. Sorensen, “Distributed coordination architecture for multi-robot formation control,” *Robotics and Autonomous Systems*, vol. 56, no. 4, pp. 324–333, 2008.

2. S. Zhao, D. V. Dimarogonas, Z. Sun, and D. Bauso, “A general approach to coordination control of mobile agents with motion constraints,” *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1509–1516, 2017.

3. D. Panagou and V. Kumar, “Cooperative visibility maintenance for leader–follower formations in obstacle environments,” *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 831–844, 2014.

4. X. Liu, S. S. Ge, and C.-H. Goh, “Vision-based leader–follower formation control of multiagents with visibility constraints,” *IEEE Transactions on Control Systems Technology*, vol. 27, no. 3, pp. 1326–1333, 2018.

5. J. Baillieul and A. Suri, “Information patterns and hedging brockett’s theorem in controlling vehicle formations,” in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, vol. 1, pp. 556–563, IEEE, 2003.

6. L. Asimow and B. Roth, “The rigidity of graphs,” *Transactions of the American Mathematical Society*, vol. 245, pp. 279–289, 1978.

7. L. Asimow and B. Roth, “The rigidity of graphs, ii,” *Journal of Mathematical Analysis and Applications*, vol. 68, no. 1, pp. 171–190, 1979.

8. T.-S. Tay and W. Whiteley, "Generating isostatic frameworks," *Structural Topology 1985 Núm 11*, 1985.

9. J. M. Hendrickx, B. D. Anderson, J.-C. Delvenne, and V. D. Blondel, "Directed graphs for the analysis of rigidity and persistence in autonomous agent systems," *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, vol. 17, no. 10-11, pp. 960–981, 2007.

10. C. Yu, J. M. Hendrickx, B. Fidan, B. D. Anderson, and V. D. Blondel, "Three and higher dimensional autonomous formations: Rigidity, persistence and structural persistence," *Automatica*, vol. 43, no. 3, pp. 387–402, 2007.

11. M. A. Belabbas, "On global stability of planar formations," *IEEE Transactions on Automatic Control*, vol. 58, no. 8, pp. 2148–2153, 2013.

12. M. Cao, A. S. Morse, C. Yu, B. D. Anderson, and S. Dasguvta, "Controlling a triangular formation of mobile autonomous agents," in *2007 46th ieee conference on decision and control*, pp. 3603–3608, IEEE, 2007.

13. M. Cao, B. D. Anderson, A. S. Morse, and C. Yu, "Control of acyclic formations of mobile autonomous agents," in *2008 47th IEEE Conference on Decision and Control*, pp. 1187–1192, IEEE, 2008.

14. L. Krick, M. E. Broucke, and B. A. Francis, "Stabilisation of infinitesimally rigid formations of multi-robot networks," *International Journal of control*, vol. 82, no. 3, pp. 423–439, 2009.

15. M.-C. Park and H.-S. Ahn, "Distance-based acyclic minimally persistent formations with non-steepest descent control," *International Journal of Control, Automation and Systems*, vol. 14, pp. 163–173, 2016.

16. M. Vidyasagar, *Nonlinear systems analysis*. SIAM, 2002.

17. P. Ioannou and B. Fidan, *Adaptive control tutorial*. SIAM, 2006.

18. S. H. Dandach, B. Fidan, S. Dasgupta, and B. D. Anderson, "A continuous time linear adaptive source localization algorithm, robust to persistent drift," *Systems & Control Letters*, vol. 58, no. 1, pp. 7–16, 2009.

19. P. A. Ioannou and J. Sun, *Robust adaptive control*, vol. 1. PTR Prentice-Hall Upper Saddle River, NJ, 1996.

20. B. Anderson, "Exponential stability of linear equations arising in adaptive identification," *IEEE Transactions on Automatic Control*, vol. 22, no. 1, pp. 83–88, 1977.

21. P. M. Lion, "Rapid identification of linear and nonlinear systems.," *AIAA Journal*, vol. 5, no. 10, pp. 1835–1842, 1967.

22. B. Fidan, S. Dasgupta, and B. D. Anderson, "Adaptive range-measurement-based target pursuit," *International Journal of Adaptive Control and Signal Processing*, vol. 27, no. 1-2, pp. 66–81, 2013.

23. Y. Maruyama, S. Kato, and T. Azumi, "Exploring the performance of ros2," in *Proceedings of the 13th International Conference on Embedded Software*, pp. 1–10, 2016.

24. S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, “Robot operating system 2: Design, architecture, and uses in the wild,” *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022.

감 사 의 말

공학을 전공한 것은 단순한 호기심 때문이였습니다. 그렇게 시작한 전공이 어느덧 석사 학위까지 이어지게 된것이 놀라울 따름입니다. 저는 수많은 공학의 산물들을 동경해 왔습니다. 이제는 그 놀라운 결과물들이 천재적인 아이디어 보다는 엄밀한 수학과 역학으로부터 탄생하였다는 것을 깨닫습니다. 창의적인 아이디어는 견고한 기초학문 위에서 사고하고 이를 수식으로 정리하고 검증하는 능력이 없다면 실현될 수 없기 때문입니다.

저는 평범한 사람이지만 그 누구보다 정직한 엔지니어가 되고 싶었습니다. 그렇지 않았다면 이토록 평범한 학위 논문을 감히 올릴 수 없었을 것입니다. 정직하기 위해서는 부족한 부분을 인정하는 것도 중요하지만 무엇보다 뛰어난 실력이 뒷받침 되어야 합니다. 정직하지만 실력이 없는 엔지니어는 무책임한 엔지니어에 불과하기 때문입니다. 앞으로 엔지니어라는 업을 삼은 기간까지 저를 믿어주고 응원해준 사람들에게 부끄럽지 않은 엔지니어가 되고자 노력할 것입니다.

문제를 놓고 고군분투하는 시간들을 기다려주신 교수님께 감사의 말을 올립니다. 쉽게 정답을 알고자 했던 게으른 습관에서 벗어나 문제 해결을 위해 고민하는 시간을 가지지 않았다면 타의적인 사고방식에서 벗어나지 못했을 것입니다. 저를 응원해주고 함께 고민해준 연구실 사람들에게 감사의 말을 전합니다. 고뇌의 시간동안 혼자였다면 절대 헤쳐나가지 못했을 것입니다. 마지막으로 결과와 상관없이 저를 사랑해주고 응원해주는 가족들과 지인들에게 진심으로 감사를 표합니다.