다중 센서 융합을 사용한 자동차형 로봇의 효율적인 실외 지역 위치 추정 방법

An Efficient Outdoor Localization Method Using Multi-Sensor Fusion for Car-Like Robots

배 상 훈*, 김 병 국
(Sang-Hoon Bae¹ and Byung-Kook Kim¹)

¹Korea Advanced Institute of Science and Technology (KAIST)

Abstract: An efficient outdoor local localization method is suggested using multi-sensor fusion with MU-EKF (Multi-Update Extended Kalman Filter) for car-like mobile robots. In outdoor environments, where mobile robots are used for explorations or military services, accurate localization with multiple sensors is indispensable. In this paper, multi-sensor fusion outdoor local localization algorithm is proposed, which fuses sensor data from LRF (Laser Range Finder), Encoder, and GPS. First, encoder data is used for the prediction stage of MU-EKF. Then the LRF data obtained by scanning the environment is used to extract objects, and estimates the robot position and orientation by mapping with map objects, as the first update stage of MU-EKF. This estimation is finally fused with GPS as the second update stage of MU-EKF. This MU-EKF algorithm can also fuse more than three sensor data efficiently even with different sensor data sampling periods, and ensures high accuracy in localization. The validity of the proposed algorithm is revealed via experiments.

Keywords: outdoor local localization, multiple sensor fusion, mobile robot, LRF, GPS

1. 서론

로봇의 실외 위치 추정은 자동화된 로봇의 동작 및 제어에 있어 가장 우선시 되어야 할 선결 과제라 할 수 있다. 여러 센서들을 통해 얻어진 주변 환경의 정보를 종합하여 자신의 위치를 알아내는 위치추정 과정이 명확하고 정확하게 확립 되어야 주어진 임무의 수행이 가능하다. 실외 환경에서의 로봇은 현재 주로 군사 목적, 탐사목적으로서 이용되는 경우가 많고, 그 임무의 성격 상, 보다 정밀하고 정확한 동작을 요구 받는 경우가 많다. 하지만 실외 환경은 실내 환경에 비해 그 규모가 훨씬 방대할 뿐만 아니라, 주변 환경의 정보가 확실치 않으며, 여러 가지 불확정 요소가 존재 할 수 있기 때문에 정확한 정보의 획득 및 처리가 어려워진다. 따라서 실외 환경에서 측정된 주변 정보를 융합한 정확한 위치 추정은 오랜 기간 동안 연구되어 온 분야이며, 정확화하면서도, 센서의 오동작에 강인한 실외 위치 추정을 위한 다양한 방법들이 제시되어 왔다[1,3,4,6].

실외 환경의 주변 정보를 얻기 위해 사용 되는 센서는 엔코더, GPS, 레이저 거리 측정기 등이 주로 사용된다. 각각의 센서는 동작 특성 및 장, 단점이 다르기 때문에, 다수의 센서 정보를 융합하여 보다 효과적인 위치추정을 수행하는 연구가 진행되어왔다. 송의규는 자동차형 로봇을 기반으로 DGPS와 로봇 엔코더 정보를 EKF로 융합한 실외 위치 추정을 수

행하였고, 가속도 센서를 이용해 노면 상태에 적응력 있는 위치 추정 알고리즘을 제안 하였다[1]. 손희진은 LRF를 이용한 vector matching 기법을 기반으로 한 실내 위치 추정 알고리즘을 제안하였다[2]. 얻어진 스캔 데이터를 sequential least square 알고리즘을 통해 vector를 추출 한 후, 이를 주어진 지도와 비교, 대응하여 현재의 위치를 추정하였다. Takahashi는 2개의 LRF를 수평, 수직으로 배치하여 3차원 map으로부터 실외 위치 추정을 수행하였고[3], Guivant는 나무, 가로등과 같은 landmark와 반사성이 높은 물체로 만들어진 beacon을 이용하여 LRF를 통한 실외 위치 추정을 구현하였다[4]. Wang은 GPS, INS, 비전 센서를 이용한 2개의 EKF 시스템으로부터 각각의 EKF 결과를 상대적으로 비교한 최종 항법 결과를 얻는 방법을 이용하였고[5], Agrawal은 비전 센서, GPS를 EKF를 이용해 위치 추정 결과를 얻었고, 엔코더, IMU등의 내부 센서는 비전 센서가 실패하였을 경우에 대체품으로서 이용하였다[6]. 기존의 연구들에서 LRF를 이용한 위치 추정 알고리즘은 주변 환경을 선분으로서 인식하거나, 이미 알려진 landmark를 통해서 위치 추정을 수행하였다. 하지만 실외 환경은 매우 광대하므로 이러한 한정된 특성으로는 적합한 결과를 얻는데 한계가 있다. 다수의 센서를 이용한 연구들은 추가적인 센서들을 주 센서의 대체품으로서의 이용, 혹은 추정 과정을 보조하는 방식으로 사용되었다.

다중 센서 융합 역시 다양하게 연구가 진행되었다. Wilner는 다중 센서 융합을 위한 Kalman filter의 사용에 관한 기본적인 구조를 제시하였다[7]. Parallel filter, sequential filter, data compression filter의 3가지 종류를 제시하였고, 이후에 진행된 다중 센서 융합은 이를 기반으로 발전시킨 연구가 많이 발견되었다. Rao는 이 중 parallel filter를 발전시킨 완전히 분산된

* 책임저자(Corresponding Author)

논문접수: 2011. 2. 10., 수정: 2011. 3. 16., 채택확정: 2011. 8. 10.

배상훈, 김병국: 한국과학기술원 전기 및 전자 공학과

(shbae@rtcl.kaist.ac.kr/bkkim@ee.kaist.ac.kr)

※ 본 연구는 (고려대학교 로봇자율주행기술 전문인력양성센터를 통한) 지식경제부/한국산업기술진흥원 융복합형로봇전문인력양성사업의 지원으로 수행되었음.

알고리즘을 제안 하였고[8], Feng은 sequential filter에서 센서간의 correlation을 고려하도록 발전시킨 알고리즘을 제안하였다[9]. Junjun은 sequential filter에서 사용되는 센서들이 asynchronous한 경우에 적용될 수 있는 필터 알고리즘을 제안하였다[10]. A Smyth 는 서로 다른 주기의 센서 정보 융합을 위한 Multi-rate Kalman filter를 설계하였다. 느린 주기로 동작하는 센서의 동작을 예측하여 높은 주기의 센서 정보와 융합하는 방법을 사용 하였지만, on-line 위치 추정에는 부적합한 알고리즘이다 [11]. S Niwa는 서로 다른 주기로 동작하는 센서의 정보를 처리하기 위해서 Time-varying Kalman gain을 설정, 각 단계별로 사용 가능한 센서에 따른 Kalman gain을 계산하여 정보를 융합하였다. 하지만 다수의 센서 정보를 처리하기에는 부적합한 방법이다[12].

본 연구에서는 서로 다른 입력 주기를 가지는 다수의 센서 정보를 융합하여, 보다 정확한 위치 추정 결과를 얻기 위한 MU-EKF (Multi-Update Extended Kalman Filter) 알고리즘을 제안한다. MU-EKF의 특정한 예로써 GPS, LRF, 모터 엔코더 3개의 센서 각각을 이용한 위치 추정을 수행하며, 각각의 위치 추정 결과를 효율적으로 융합한다. LRF는 위치 추정 정확도가 가장 뛰어나지만, 범위를 SLAM으로 확장 할 경우, 초기 위치 오차에 굉장히 취약해진다. 이에 비해 GPS는 상대적으로 오차가 가장 크지만, 초기 위치 설정이 매우 효율적이며, 갑자기 옮겨졌을 경우에 대해서도 처리가 가능해진다. 엔코더는 다른 두 센서에 비해서 가장 빠른 주기로 위치가 업데이트 된다. 이렇게 다양한 센서의 정보를 조합 함으로서, 보다 정확하면서도 불확정요소에 강인한 위치 추정 시스템을 구축할 수 있다.

우선, 각 센서마다 독립적으로 위치 추정 결과를 얻는다. 엔코더는 추측항법을 통해 현재 위치를 계산하고, GPS는 제공되는 절대 좌표로부터 현재 위치를 계산 할 수 있다. LRF에 의한 위치 추정은, 우선 주변 환경을 2차원의 선/원 개체로서 표현한 뒤, 지도와의 대응을 통해 이루어진다. 기존 연구 [2]에서는 선분과 선분간의 대응관계를 통해 위치 관계를 설정하였으나, 실외 환경은 선분으로 표현 되기 어려운 가로수, 가로등과 같은 원형의 개체들이 많으므로, 본 연구에서는 원형 개체와의 대응관계를 추가하여 기존의 알고리즘을 확장하였다.

다음으로 각 센서의 위치 추정 결과를 융합한다. MU-EKF 알고리즘은 sequential filter [7,10]에서 제시한 복수의 중첩적인 update구조를 응용한 알고리즘이다. 각각의 센서 주기마다 추가적인 센서 입력이 들어올 경우 추가적인 update를 수행함으로써, 센서마다 다른 횟수의 update step을 갖는 가변적인 횟수의 multi-update구조를 가지며, 이전 단계의 추정 결과를 보정하여 최종적인 위치 추정 결과를 얻는다. 기존의 다중 센서 융합 칼만 필터알고리즘은 사용하는 센서가 같은 주기로 동작되는 경우만을 고려하거나[7], 서로 시간 주기가 다른 센서 정보 융합을 위해 센서 데이터를 예측하는 방법[11], 단계별로 사용 가능한 센서에 따라 Kalman gain을 새로 설정하는 방법을 고안하였다[12]. 하지만 이러한 방법은 다수의 센서 사용과 실시간 위치 추정에는 부적합하다. 또한 추가적인 센서를 사용할 경우의 적용도 어렵다. MU-EKF 알고리즘은

각 단계마다 사용 가능한 센서에 따라 update 횟수를 다르게 설정 할 수 있으므로, 일부 센서의 오작동 시에도 안정적인 위치 추정 결과를 얻을 수 있으며, 센서의 추가 역시 용이하므로, 확장성이 좋다.

알고리즘의 검증을 위하여 상용 전동 스쿠터를 개조하여 주행, 조향 엔코더를 장착하고, GPS와 LRF센서를 장착하고, 센서 정보의 처리 및 모터 컨트롤을 위한 임베디드 보드를 장착한 자동차형 로봇 시스템을 사용하였다[1,13].

본 논문의 구성은 다음과 같다. 제 II 장에서는 실외 위치 추정 시스템을 포함한 전동 스쿠터 시스템의 구조에 대해 소개한다. 제 III 장에서는 LRF를 기반으로 한 실외 위치 추정 알고리즘을 제안한다. 제 IV 장에서는 위의 결과와 GPS, 엔코더 정보를 기반으로 한 각각의 위치 추정 결과를 융합하기 위한 MU-EKF 알고리즘을 제안한다. 제 V 장에서는 시뮬레이션과 실험을 통해 제안한 알고리즘을 검증할 것이며, 제 VI 장에서는 연구에 대한 결론 및 추후 과제에 대해 기술한다.

II. 설계 및 구현

이 장에서는 사용된 전동 스쿠터 로봇의 구조와 위치 추정 알고리즘 구현을 위한 소프트웨어 구조를 설명 한다.

1. 기구부

1.1 자동차형 로봇의 구조

본 연구에서 사용하는 자동차형 로봇은 상용 전동 스쿠터를 개조하여 구성된 시스템[13]을 사용하였다. 자동 주행을 위한 조향 제어를 위해 조향 핸들과 바퀴 축을 분리 후, 조향 모터 엔코더를 포함한 조향 모터로 바퀴 축을 제어하고, 조향 의도 파악을 위해 조향 핸들 엔코더가 별도로 장착되었고, 조향 핸들 엔코더로부터 측정된 회전각은 제어 보드에서 PID 각도 제어를 이용하여 조향모터가 제어되는 시스템이다.

1.2 자동차형 로봇의 기구학

자동차형 로봇의 기구학 모델은 다음과 같다[1].

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v_s \cos \psi_s \cos \theta \\ v_s \cos \psi_s \sin \theta \\ \frac{v_s \sin \psi_s}{l} \end{bmatrix} \quad (1)$$

여기서 x, y 는 전동 스쿠터 뒷바퀴 축의 주행평면상의 중심 위치, θ 는 전동 스쿠터의 방향, v_s 는 선속도, ψ_s 는 스쿠터의 조향각을 의미하며, l 은 앞바퀴와 뒷바퀴 사이의 거리 (wheel base)를 의미한다. 샘플링 주기 Δt 로 이산시간으로 나타내면 다음과 같다.

그림 1. 자동차형 로봇의 기구학 모델.

Fig. 1. Kinematics model of car-like robot.

그림 2. 하드웨어 블록 다이어그램.
Fig. 2. Hardware block diagram.

$$\begin{aligned} x(k+1) &= x(k) + v(k) \cos \psi_s(k) \cos \theta(k) \cdot \Delta t \\ y(k+1) &= y(k) + v(k) \cos \psi_s(k) \sin \theta(k) \cdot \Delta t \\ \theta(k+1) &= \theta(k) + \frac{v(k) \sin \psi_s(k)}{l} \cdot \Delta t \end{aligned} \quad (2)$$

이 시스템의 상태 벡터 $X(k)$ 는

$$X(k) = \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} \quad (3)$$

이고, 제어 입력 벡터 $u(k)$ 는

$$u(k) = \begin{bmatrix} v_s(k) \\ \psi_s(k) \end{bmatrix} \quad (4)$$

이다.

2. 임베디드 하드웨어

본 연구를 위해 다음과 같이 임베디드(embedded) 하드웨어가 구축되어있다[1,13]. 그림 2와 같이 전동 스쿠터의 제어 및 센서 정보의 취득 및 융합을 위한 임베디드 프로세서 보드와, DGPS 모듈과 LRF 모듈, 각 센서의 데이터 처리 및 제어를 위한 추가 인터페이스 보드로 구성되어있다.

DGPS는 위성 신호를 잘 받을 수 있도록 의자 등받이 상단에 높게 설치하였다. MTK chipset을 탑재한 FGPMOPA6 모듈을 사용하였다. 출력은 1Hz (1000ms)마다 전송된다[1].

LRF는 전방부를 스캔할 수 있도록 스쿠터의 핸들 부에 추가로 장착하였다. Hokuyo사의 UTM-30LX를 이용하였다. 최대 스캔 범위는 30 m이며 스캔 범위는 전방 270도이다. USB를 통해 PC로 연결되어 지면과 평행한 면으로 스캔 되도록 설정해 주었다. 데이터는 최대 40Hz (25ms) 주기로 전송 가능하다.

3. 소프트웨어 구조

임베디드 리눅스에 RTAI를 패치하여 사용한다. 어플리케이션과 모듈 프로그램 간의 지연 없는 통신을 위하여 RT-FIFO를 사용하며, 시리얼 통신을 위한 SPDRV를 사용한다. USB를 사용하는 LRF 데이터는 RTAI에서 USB드라이버를

그림 3. 소프트웨어 구조.
Fig. 3. Software structure.

제공하지 않으므로, PC의 USB 포트를 이용해 어플리케이션 영역에서 처리를 해준다. 센서 스펙 상의 출력주기를 그대로 쓰는 것이 힘들어 지므로, PC에서 위치 추정 알고리즘을 통해 얻어진 위치 추정 결과를 0.1초 마다 시리얼 포트를 통해 임베디드 보드로 전송하여 데이터 융합을 진행한다.

최종적인 위치 추정 수행 과정은 그림 3과 같은 단계로 진행된다. 어플리케이션 영역 프로그램은 POSIX thread를 이용한 multi thread 구조로 구성한다. 각각의 thread는 RT-FIFO와 PC에서의 입력을 pending하였다가 처리함으로써, 각각의 센서 입력에 동기화 되고, 최종적인 위치 추정 결과는 PC를 통해 사용자에게 결과를 보인다.

III. 레이저 거리 측정기 기반의 실외 위치 추정 알고리즘

MU-EKF를 설명하기에 앞서, 레이저 거리 측정기 기반의 실외 위치 추정에 대하여 설명한다. LRF 센서에서는 현재 위치로부터 전방 270도 스캔 범위 내의 거리 데이터들로 얻어진다. 이 거리들에 해당하는 점 데이터들을 실외 지도를 구성하는 개체들과 같은 형태로 추출되어야 미리 작성된 실외 지도와 스캔 데이터의 비교를 통한 대응관계로부터 위치 추정을 수행할 수 있게 된다[2].

본 장에서는 실외 지도 작성과 아울러, 다음과 같은 과정들로 구성된 스캔 데이터 처리 및 대응 과 그로부터 얻어지는 위치 추정 과정에 대해 자세히 설명한다.

- 단계 1. 유효 데이터 분류 및 분할
- 단계 2. 분할 데이터로부터 선분/원 개체 구분 및 추출
- 단계 3. 추출된 개체와 지도 개체와의 대응
- 단계 4. 대응된 개체정보를 이용하여 현재 로봇의 위치 추정

1. 실외 지도 작성

위치추정에서 사용하는 실외 지도는 실외 환경을 2차원으로 단순화 시켜, 선분과 원 두 종류의 개체로 구분하여 미리 작성한다. 건물의 외벽과 같은 개체는 선분으로 표시되며, 나무, 가로등, 기둥과 같은 개체는 원으로서 표시된다.

선분 개체 1은 다음과 같이 표현된다(그림 4(a)).

$$\begin{aligned} x \cos \theta_l + y \sin \theta_l &= \rho_l \\ (x_{l1} \leq x \leq x_{l2}, y_{l1} \leq y \leq y_{l2}) \end{aligned} \quad (5)$$

여기서 θ_i 는 원점에서 선분을 포함하는 직선을 향한 수선이 이루는 각, ρ_i 은 수선의 길이, 그리고 $(x_{i1}, y_{i1}), (x_{i2}, y_{i2})$ 는

그림 4. 선분과 원 개체의 표현.

Fig. 4. Expression of line and circle objects.

선분을 구성하는 양 끝점을 나타낸다. 이에 따라 선분 개체는 다음과 같이 표시한다.

$$M_{line,l} = \{\theta_l, \rho_l, (x_{l1}, y_{l1}), (x_{l2}, y_{l2})\} \quad (6)$$

원 개체 c는 다음과 같이 표현된다(그림 4(b)).

$$(x - x_c)^2 + (y - y_c)^2 = R_c^2 \quad (7)$$

즉, 원의 중심점 (x_c, y_c) 와 원의 반지름(R_c)로서 구성된다. 따라서 다음과 같이 표시할 수 있다.

$$M_{circle,c} = \{R_c, (x_c, y_c)\} \quad (8)$$

2. 단계 1. 유효 데이터 분류 및 분할

단계 1에서는 측정된 거리 데이터들 중, 센서의 유효 측정 범위 외의 데이터를 제거한 후, 불연속점을 기준으로 분할한다. 점 데이터간의 거리가 threshold

(th_{grp})보다 크다면, 서로 다른 그룹으로 분할한다. 이렇게 분할된 데이터들 각각의 집합은 원 개체 1개, 혹은 다수의 선분 개체들의 집합으로서 구분 되어야 한다. 불연속점을 인식하기 위한 th_{grp} 는 스캔된 거리 및 각도에 따라 가변적으로 계산 되도록 한다. 다음과 같이 계산된다[15].

$$th_{grp} = \|p_n^h - p_{n-1}\| + 3\sigma \quad (9)$$

$$\|p_n^h - p_{n-1}\| = r_{n-1} \cdot \frac{\sin(\Delta\phi)}{\sin(\lambda - \Delta\phi)}$$

p_{n-1} : 점 데이터, p_n^h : 예측되는 다음 점 데이터,

r_n : 측정 거리, $\Delta\phi$: 두 측정거리간의 각도,

λ : 임의의 각, σ : LRF std. deviation

3. 단계 2. 분할 데이터로부터 선분/원 개체 구분 및 추출

분할된 각각의 데이터 집합을 원과 직선으로 근사한 후, 어떤 근사가 더 적합한지를 판단함으로써, 해당 집합이 원 개체인지 선분 개체인지를 판별한다. 선분으로의 근사는 sequential least square fit [2]을 이용하고, 원으로의 근사는 least square fit을 이용한다. Sequential least square fit은 일반적으로 사용되는 line fitting 방법인 split-merge method보다 더 빠른 성능을 나타낸다. 분할된 각각의 데이터 집합을 선분/원으로 근사 후, 측정 데이터와 근사된 개체의 오차를 비교하여 오차가 적은 근사형태를 취함으로써 스캔된 데이터가 원형 개체인지 선분 개체인지를 판별, 개체 추출을 완료한다.

그림 5. 개체 구분/추출 과정.

Fig. 5. Object distinguishing/extracting.

4. 단계 3. 추출된 개체와 지도 개체와의 대응

스캔 데이터에서 추출된 개체 정보와 지도를 구성하는 개체 정보를 비교하여, 각각의 추출된 개체가 지도상의 어떤 개체에 대응되는지를 확인한다. 이렇게 대응된 관계로부터 위치 추정을 수행한다. 대응을 확인하기 전에, 센서 데이터는 local 좌표계 상에 존재하므로, 이를 이전 단계의 로봇 위치와 현재 핸들의 각도를 이용하여 지도 기준의 global 좌표계로 변환해 주어야 한다. 좌표계 변환이 완료된 각 개체들은 다음 과정을 통해 대응을 확인한다. 본 연구에서는 각 선분, 원 개체에 대하여 확률을 사용하지 않고 deterministic 한 개체로 사용함으로써, 계산의 간단화, 실시간화를 도모하였다.

5. 단계 4. 대응된 개체정보를 이용하여 현재 로봇의 위치 추정

추출된 개체와 지도 개체와의 대응 관계를 바탕으로 현재 로봇의 위치를 추정한다. 대응 관계에 있는 개체들간의 위치 관계로부터 비용함수를 설정한다. 이 비용함수는 병진 벡터 T와 회전 각도 ϕ 의 함수로 나타낼 수 있고, 지역 위치 추정 문제이므로 이 비용함수를 최소화 하는 병진 벡터와 회전 벡터로부터 로봇의 현재 위치를 추정하는 것이 가능하다. 비용함수는 대응된 개체의 형태(선분/원)에 따라 다르게 정의된다.

5.1 선분 비용 함수

다음과 같이 비용 함수를 정의 한다. 선분 개체(s_j)를 회전, 병진 이동한 선분($\bar{s}_j$)과 이루는 양 끝 점으로부터 지도 선분을 이루는 직선과의 수직 거리를 $v_j(j=1,2)$, 개체 선분의 길이를 l_j , 선분을 이루는 데이터의 수를 N_j , 선분을 이루는 데이터 들이 갖는 분산을 σ_j^2 라고 할 때, 비용함수는 다음과 같이 정의된다[2].

$$E_{line}(\phi, T) = \eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} \sum_{j=1}^2 v_{ij}^2 \quad (10)$$

여기서 σ_l , l_l , N_l 와 같은 값들은 가중치로서 적용된다. 데이터로부터 얻어진 선분이 길고, 많은 데이터를 포함하고 있으며, 분산이 적을수록 더 높은 가중치를 갖도록 설계된 것이다. η 은 normalizing factor이다.

v_{ij} 는 추출된 선분 개체를 회전/병진 이동시킨 선분과 대응되는 지도 개체간의 거리를 의미하며 다음과 같이 표현된다.

$$v_{ij} = [\cos \theta_l \quad \sin \theta_l] (R_\phi \cdot r_{ij} + T) - \rho_l \quad (11)$$

여기서 $R_\phi = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$, $r_{ij} = \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix}$ 이다.

벡터 표현은 다음과 같다.

$$v_{ij} = \Theta_l (\Delta_{ij} \Phi + T) - \rho_l \quad (12)$$

여기서 $\Theta_l = \begin{bmatrix} \cos \theta_l \\ \sin \theta_l \end{bmatrix}$, $\Delta_{ij} = \begin{bmatrix} x_{ij} & -y_{ij} \\ y_{ij} & x_{ij} \end{bmatrix}$, $\Phi = \begin{bmatrix} \cos \phi \\ \sin \phi \end{bmatrix}$.

ρ_l 와 θ_l 는 해당 되는 지도 선분 m_l 의 정보이다. 이로부터 M_l 개의 대응된 선분에서의 비용함수를 다음과 같이 전개할 수 있다[2].

$$\begin{aligned} E_{line}(\Phi, T) &= \eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} \sum_{j=1}^2 (\Theta_l^T (\Delta_{ij} \Phi + T) - \rho_l)^2 \\ &= \Phi^T A_{line} \Phi + 2T^T B_{line} \Phi + T^T C_{line} T \\ &\quad - 2U_{line}^T \Phi - 2V_{line}^T T + \tau_{line} \end{aligned} \quad (13)$$

여기서

$$\begin{aligned} A_{line} &= \eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} \sum_{j=1}^2 \Delta_{ij}^T \Theta_l \Theta_l^T \Delta_{ij} & B_{line} &= \eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} \sum_{j=1}^2 \Theta_l \Theta_l^T \Delta_{ij} \\ C_{line} &= 2\eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} \Theta_l \Theta_l^T & U_{line} &= \eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} \sum_{j=1}^2 \rho_l \Delta_{ij}^T \Theta_l \\ V_{line} &= 2\eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} \rho_l \Theta_l & \tau_{line} &= 2\eta \cdot \sum_{l=1}^{M_l} \frac{N_l l_l}{\sigma_l} (\rho_l)^2 \end{aligned}$$

5.2 원 비용 함수

원 개체와의 대응 관계를 위한 비용 함수를 새롭게 제안한다. 직선에서의 비용 함수 전개 결과와 같이 Quadratic 한 형태가 되도록 설정하였다. 원 개체(s_c)를 회전, 병진 이동한 원($\bar{s}_c$)을 구성하는 모든 점에서 대응되는 지도의 원 개체(m_c)에 대해 다음과 같은 거리 관계를 가지게 된다(그림 6참조).

$$\begin{aligned} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} x_c \\ y_c \end{bmatrix} + R_c \begin{bmatrix} \cos \theta_c \\ \sin \theta_c \end{bmatrix} \\ (\Delta_{cm} \Phi + T) &= r_{c,c} + R_c \Theta_{circle,c} \\ r_{cm} &= ((\Delta_{cm} \Phi + T) - r_{c,c} - R_c \Theta_{circle,c}) \end{aligned} \quad (14)$$

여기서 $\Theta_{circle,c} = \begin{bmatrix} \cos \theta_c \\ \sin \theta_c \end{bmatrix}$, $\Delta_{cm} = \begin{bmatrix} x_{cm} & -y_{cm} \\ y_{cm} & x_{cm} \end{bmatrix}$, $\Phi = \begin{bmatrix} \cos \phi \\ \sin \phi \end{bmatrix}$.

이로부터 정의되는 M_c 개의 대응된 원에서의 비용 함수는 다음과 같다. 직선의 경우와 같이 원 개체를 구성하는 데이

그림 6. 원 비용함수 정의.

Fig. 6. Definition of circle cost function.

터의 수(N_c), 원호의 길이(l_c), 분산(σ_c)값을 가중치로 두었다.

$$\begin{aligned} E_{circle}(\Phi, T) &= \eta \cdot \sum_{c=1}^{M_c} \frac{N_c l_c}{\sigma_c} \sum_{m=1}^{N_c} \|r_{cm}\|_2^2 \\ &= \eta \cdot \sum_{c=1}^{M_c} \frac{N_c l_c}{\sigma_c} \sum_{m=1}^{N_c} \{((\Delta_{cm} \Phi + T) - r_{c,c} - R_c \Theta_{circle,c})^T \\ &\quad \cdot ((\Delta_{cm} \Phi + T) - r_{c,c} - R_c \Theta_{circle,c})\} \end{aligned} \quad (15)$$

여기서 m 은 개체 집합 내의 데이터 index를 의미한다.

이 식 역시 직선에서의 비용 함수와 같이 전개할 수 있다.

$$\begin{aligned} E_{circle}(\Phi, T) &= \Phi^T A_{circle} \Phi + 2T^T B_{circle} \Phi + T^T C_{circle} T \\ &\quad - 2U_{circle}^T \Phi - 2V_{circle}^T T + \tau_{circle} \end{aligned} \quad (16)$$

여기서

$$\begin{aligned} A_{circle} &= \eta \cdot \sum_{c=1}^{M_c} \frac{N_c l_c}{\sigma_c} \sum_{m=1}^{N_c} \Delta_{cm}^T \Delta_{cm} & B_{circle} &= \eta \cdot \sum_{c=1}^{M_c} \frac{N_c l_c}{\sigma_c} \sum_{m=1}^{N_c} \Delta_{cm} \\ C_{circle} &= \eta \cdot \sum_{c=1}^{M_c} N_c \frac{N_c l_c}{\sigma_c} I \\ U_{circle} &= \eta \cdot \sum_{c=1}^{M_c} \frac{N_c l_c}{\sigma_c} \sum_{m=1}^{N_c} (\Delta_{cm}^T r_{c,c} + R_c \Delta_{cm}^T \Theta_{circle,c}) \\ V_{circle} &= \eta \cdot \sum_{c=1}^{M_c} N_c \frac{N_c l_c}{\sigma_c} (r_{c,c} + R_c \Theta_{circle,c}) \\ \tau_{circle} &= \eta \cdot \sum_{c=1}^{M_c} N_c \frac{N_c l_c}{\sigma_c} (r_{c,c}^T r_{c,c} + 2R_c \Theta_{circle,c}^T r_{c,c} + R_c^2 \Theta_{circle,c}^T \Theta_{circle,c}) \end{aligned}$$

5.3 총 비용 함수 및 위치 추정

원과 선분에서의 비용함수가 모두 비슷한 형태의 quadratic equation으로 나타나므로, (16)과 (19)로부터 다음과 같이 총비용함수를 설정하여, 이를 최소로 하는 회전, 병진 벡터를 구할 수 있다.

$$\begin{aligned} E_{total}(\Phi, T) &= E_{line}(\Phi, T) + E_{circle}(\Phi, T) \\ &= \Phi^T A_{tot} \Phi + 2T^T B_{tot} \Phi + T^T C_{tot} T \\ &\quad - 2U_{tot}^T \Phi - 2V_{tot}^T T + \tau_{tot} \end{aligned} \quad (17)$$

여기서 singular case가 발생 할 수 있다. 선분 대응을 통한 위치 추정에서, 대응되는 모든 선분이 평행하여

단일 해를 구할 수 없는 경우에 나타난다. 하지만 본 연구에서는 원과의 대응 관계 역시 포함되기 때문에, 모든 선분이 평행하더라도 원 개체의 대응관계가 있다면 단일 해를 구할 수 있다. 센서의 범위가 30미터로 충분히 넓을 뿐 아니라,

실험 환경에는 나무와 가로등과 같은 원 개체가 매우 많으므로, singular case는 본 연구에서 고려하지 않는다.

총 비용 함수를 최소로 하는 유일해로서 병진 벡터와 회전 각도를 구할 수 있으며, 이것은 곧 이전 단계의 위치 추정 결과에서의 상대적인 이동 벡터를 의미한다. 따라서 이로부터 현재 로봇의 위치 추정 할 수 있다. 로봇의 현재 위치를 X_k 라 할 때 LRF에 의한 현재 위치 추정 결과 $X_{lrf,k}$ 는 다음과 같이 나타난다.

$$X_{lrf,k} = R_{\theta} X_k + T \quad (18)$$

IV. 다중 센서 융합 위치 추정 알고리즘

1. 개요

본 연구가 지향하는 목표는 각각의 센서 정보를 효과적으로 융합함으로써, 각 센서들이 가지는 오차와 불안정성을 다른 센서의 데이터를 이용해 보정하는 것이다. 이렇게 보정된 위치 추정 결과는 새로운 센서의 입력이 들어올 때마다 보정을 중첩하여, 최적의 위치 추정 결과를 얻을 수 있다. 따라서 구현 되어야 할 다중 센서 융합 위치 추정 알고리즘은 각각의 센서에서 발생하는 오차를 다른 센서들의 측정 결과를 통해 보정할 수 있어야 한다. 또한 각각의 센서들은 모두 다른 주기로 작동하므로, 동작 주기의 불일치를 고려한 융합 알고리즘을 세워야 한다.

센서 동작시간이 일치하지 않는 환경에서 여러 센서의 측정 결과를 융합하는 방법으로는, sequential Kalman filter [7]에서 사용되는 복수의 update step을 통한 중첩보정 구조가 적합하다.

MU-EKF 알고리즘은 한 번의 prediction step에서 얻어진 state 예측 결과를 추가 prediction 없이 복수의 measurement update를 통해 state estimation을 보정하는 sequential Kalman filter [7,10]를 응용한 알고리즘으로, 다른 시간 주기로 출력되는 복수의 센서 측정값을 이용해 이전 단계의 추정 결과를 반복적으로 보정함으로써 보다 정확하고 신뢰성 있는 위치 추정 결과를 얻을 수 있다.

융합에 사용되는 센서들의 주기에 맞추어, 센서 정보들이 순차적으로 update 될 수 있도록 알고리즘 동작의 시간 단위를 다음과 같이 설정해준다.

T_e : 엔코더 출력 주기로 10ms 단위로 나타난다.

$$mT_e (m=0, 1, 2, \dots)$$

T_l : LRF 출력 주기로 100ms 단위로 나타난다.

$$lT_l (l=0, 1, 2, \dots)$$

T_g : GPS 출력 주기로 1000ms 단위로 나타난다.

$$kT_g (k=0, 1, 2, \dots)$$

각 시간 단위 사이에는 다음과 같은 관계가 있다.

$$100T_e = 10T_l = T_g$$

$$lT_l = \left\lfloor \frac{m}{10} \right\rfloor T_l = \left\lfloor \frac{m}{10} \right\rfloor 10T_e$$

$$kT_g = \left\lfloor \frac{m}{100} \right\rfloor T_g = \left\lfloor \frac{m}{100} \right\rfloor 100T_e$$

1. T_e 주기마다 입력되는 엔코더 오도메트리 정보를 이용해 차량의 기구학 모델 (2)로부터 로봇의 현재위치 및 방향을 예측한다.

2. LRF 데이터가 입력되는 T_l 주기마다 그 동안 예측한 위치 및 방향 정보를 업데이트 함으로써 이전의 위치 추정 결과를 보정한다.

3. GPS 데이터가 입력되는 T_g 주기마다 2번째 업데이트를 통해 1번째 업데이트에서 보정한 위치 추정 결과를 한번 더 보정한다. 이렇게 최종적인 위치 추정 결과를 얻게 된다.

즉, 각각의 time step마다 사용 가능한 sensor measurement 만큼 update step을 중첩함으로써, time step에 따른 가변적인 update step을 갖는다. 각 filtering step에서의 최종 위치 추정 결과는 가장 느린 GPS의 입력 결과를 이용한 2번째 update step이며, 이는 T_e , T_g 의 최소 공배수 step 마다 나타나게 된다. 전체적인 알고리즘의 구조 및 데이터 흐름은 그림 7과 같다.

복수의 measurement가 존재할 경우 순차적인 update 구조를 통해 반복적으로 보정함으로써, 1개의 measurement만을 이용한 추정 결과에 비해 보다 향상된 정확도를 갖는다. prediction, 1st update step의 추정결과는 다음 step의 pre-state로서 활용함으로써, 최종 time step까지 그 추정 결과를 누적하며, 이렇게 누적된 결과는 최종 위치 추정 step에서 사용함으로써, 가장 느린 입력의 센서까지의 정보를 모두 효과적으로 이용한 위치 추정 결과를 얻을 수 있다.

이러한 융합 구조는 일부 센서가 오작동 하거나, 위치 정보를 얻지 못하였을 경우에도, 안정적인 동작이 가능하다. 갑자기 로봇이 옮겨질 경우(kidnapping), 엔코더와 LRF는 순간적인 오차가 존재하게 된다. 하지만 LRF는 지도와 주변환경의 수 회의 matching을 통해 극복할 수 있으며, GPS는 옮겨진 사실을 바로 알 수 있다. 이렇게 두 추가적인 센서 정보의 융합을 통해 kidnapping으로 인한 오차를 최소화 할 수 있다.

센서 정보가 유실되거나 오작동으로 인해 이전 값과 큰 변화가 있을 경우, 해당 센서의 update 스텝을 무시하고, 최종적인 update 혹은 prediction 결과를 현재 위치 추정 값으로서 이용할 수 있다. 따라서 일부 센서의 오작동에 대해 보다 강한 위치 추정이 가능하다.

그림 7. MU-EKF 구조.

Fig. 7. MU-EKF structure.

2. MU-EKF 설계

2.1 Vector의 정의

2.1.1 State vector의 정의

전동 스쿠터의 현재 위치 및 방향 X 를 EKF의 state vector로 정의 한다.

2.1.2 Process model의 정의

시스템 입력은 엔코더로부터 입력되는 선속도(v_s)와 조향각(ψ_s)을 사용한다. 엔코더 입력과 전동스쿠터 기구학 모델을 process 모델로서 사용한다.

2.1.3 Measurement 1 $Y_i(IT_i)$ 의 정의

$$Y_i(IT_i) = h_i(X(IT_i), IT_i) + w_i(IT_i) \quad (19)$$

$$\text{여기서 } h_i(X(IT_i), IT_i) = \begin{bmatrix} x_{lrf}(IT_i) \\ y_{lrf}(IT_i) \\ \theta_{lrf}(IT_i) \end{bmatrix}$$

III 장의 알고리즘을 통해 얻어진 LRF를 이용한 위치 추정 결과 X_{lrf, IT_i} 를 첫 번째 measurement vector로서 정의한다.

$w_i(IT_i)$ 은 measurement 1 noise vector로, zero-mean이고, covariance matrix가 $W_i(IT_i)$ 인 white Gaussian noise를 가정한다.

2.1.4 Measurement 2 $Y_g(kT_g)$ 의 정의

$$Y_g(kT_g) = h_g(X(kT_g), kT_g) + w_g(kT_g) \quad (20)$$

$$\text{여기서 } h_g(X(kT_g), kT_g) = \begin{bmatrix} x_{gps}(kT_g) \\ y_{gps}(kT_g) \\ \theta_{gps}(kT_g) \end{bmatrix}$$

GPS로부터 얻어진 위도 경도 데이터를 UTM으로 변환 후, 전역 지도의 원점의 UTM 좌표와의 상대적인 거리로부터 얻어진 GPS 실외 위치 추정 값 (x_{gps} , y_{gps} , θ_{gps})을 2번째 measurement vector로서 정의 한다. $w_g(kT_g)$ 는 measurement 2 noise vector로, zero-mean이고, covariance matrix가 $w_g(kT_g)$ 인 white Gaussian noise를 가정한다

2.2 Prediction step

현재 주어진 state estimate $X((m-1)T_e)$ 로부터 엔코더 입력 값과 전동 스쿠터 기구학 모델을 이용한 dead reckoning을 통해 T_e 마다 state 예측을 수행하고, predicted covariance matrix를 계산한다.

예측된 state의 covariance matrix 계산시, 엔코더의 입력 오차 성분인 process noise vector $v(mT_e)$ 가 사용된다. Process noise vector 역시 zero-mean이고, covariance matrix가 $V(mT_e)$ 인 white Gaussian noise를 가정한다.

2.2.1 Predicted state $\bar{X}$:

$$\bar{X}(mT_e) = f(\bar{X}((m-1)T_e), u((m-1)T_e), (m-1)T_e)E \quad (21)$$

여기서

$$f(X(mT_e), u(mT_e), mT_e) = \begin{bmatrix} mT_e \cdot v_s(mT_e) \cos \psi_s(mT_e) \cos \theta(mT_e) + x(mT_e) \\ mT_e \cdot v_s(mT_e) \cos \psi_s(mT_e) \sin \theta(mT_e) + y(mT_e) \\ \frac{mT_e \cdot v_s(mT_e) \sin \psi_s(mT_e)}{l} + \theta(mT_e) \end{bmatrix}$$

2.2.2 Predicted Covariance matrix $\bar{P}$:

$$\bar{P}(mT_e) = F_x((m-1)T_e)\bar{P}((m-1)T_e)F_x^T((m-1)T_e) + F_u((m-1)T_e)V((m-1)T_e)F_u^T((m-1)T_e) \quad (22)$$

여기서

$$F_x(mT_e) = \begin{bmatrix} 1 & 0 & -mT_e \cdot v_s(mT_e) \cos \psi_s(mT_e) \sin \theta_k(mT_e) \\ 0 & 1 & mT_e \cdot v_s(mT_e) \cos \psi_s(mT_e) \cos \theta_k(mT_e) \\ 0 & 0 & 1 \end{bmatrix}$$

$$F_u(mT_e) = \begin{bmatrix} mT_e \cdot \cos \psi_s(mT_e) \cos \theta(mT_e) \\ mT_e \cdot \cos \psi_s(mT_e) \sin \theta(mT_e) \\ \frac{mT_e \cdot \sin \psi_s(mT_e)}{l} \\ -mT_e \cdot v_s(mT_e) \sin \psi_s(mT_e) \cos \theta(mT_e) \\ -mT_e \cdot v_s(mT_e) \sin \psi_s(mT_e) \sin \theta(mT_e) \\ \frac{mT_e \cdot v_s(mT_e) \cos \psi_s(mT_e)}{l} \end{bmatrix}$$

2.2.3 Process noise covariance matrix

엔코더의 오차 정보를 noise vector로서 사용한다. σ_{v_s} 는 주행 속도 엔코더의 standard deviation, σ_{ψ_s} 는 조향 모터 엔코더의 standard deviation을 사용한다.

$$V(mT_e) = \begin{bmatrix} \sigma_{v_s}^2 & 0 \\ 0 & \sigma_{\psi_s}^2 \end{bmatrix} \quad (23)$$

LRF 입력이 들어오지 않을 경우, 이번 step의 추정 결과 $\bar{X}(mT_e)$, $\bar{P}(mT_e)$ 를 다음 prediction step에서 사용하는 pre-state로서 사용 한다.

2.3 1st update step

첫 번째 업데이트에는 LRF로부터 얻어진 위치 추정 결과를 measurement vector로서 사용하여, prediction step에서 얻어진 state 예측 결과를 보정한다. T_i 시간 주기마다 수행된다.

2.3.1 Kalman gain 계산

$$R_i(IT_i) = \bar{P}(mT_e)H_i(IT_i)^T S_i^{-1} \quad (24)$$

여기서 $S_i = H_i(IT_i)\bar{P}(mT_e)H_i(IT_i)^T + W_i(IT_i)$

2.3.2 Measurement 1 noise covariance matrix

LRF 위치 추정의 오차 정보를 noise vector로서 사용한다. 각각 LRF 위치 추정 결과의 위치($\sigma_{x_{lrf}}$, $\sigma_{y_{lrf}}$) 방향($\sigma_{\theta_{lrf}}$)에 서의 오차 정보로 다음과 같이 정의 된다.

$$W_i(IT_i) = \begin{bmatrix} \sigma_{x_{lrf}}^2 & 0 & 0 \\ 0 & \sigma_{y_{lrf}}^2 & 0 \\ 0 & 0 & \sigma_{\theta_{lrf}}^2 \end{bmatrix} \quad (25)$$

2.3.3 State estimate

Innovation error:

$$v_i = Y_i(IT_i) - H_i(IT_i)\bar{X}(mT_e) \quad (26)$$

State estimation $\hat{X}$:

$$\hat{X}(IT_i) = \bar{X}(mT_e) + R_i(IT_i)v_i(IT_i) \quad (27)$$

2.3.4 Covariance matrix estimation $\hat{P}$:

$$\hat{P}(IT_i) = \bar{P}(mT_e) - R_i(IT_i)H_i(IT_i)\bar{P}(mT_e) \quad (28)$$

$$\text{여기서 } H_i(IT_i) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

역시 GPS 신호 주기가 아닐 경우, 이번 step까지의 추정 결과 $\hat{X}(IT_i)$, $\hat{P}(IT_i)$ 를 다음 prediction step 에서의 pre-state로서 사용한다.

2.4 2nd update step

두 번째 업데이트에는 GPS로부터 얻어진 위치 추정 결과를 measurement vector 로서 사용하여, 1st update에서 추정된 state와 covariance matrix를 한번 더 보정 해 주면서, 최종적인 state와 covariance matrix를 추정하게 된다. 2번째 update step을 통한 최종 추정 결과 $\tilde{X}$, $\tilde{P}$ 가 3개의 센서 정보를 융합함으로써 얻어지는 위치 추정 결과이다.

2.4.1 Kalman gain 계산

$$R_g(kT_g) = \hat{P}(IT_i)H_g(kT_g)^T S_g^{-1} \quad (29)$$

$$\text{여기서 } S_g = H_g(kT_g)\hat{P}(IT_i)H_g(kT_g)^T + W_g(kT_g)$$

2.4.2 Measurement 2 noise covariance matrix

GPS의 위치 오차 정보를 noise vector로서 사용한다. 각각 GPS 위치 추정 결과의 위치 ($\sigma_{x_{gps}}$, $\sigma_{y_{gps}}$) 방향($\sigma_{\theta_{gps}}$)에서의 오차 정보로 다음과 같이 정의 된다.

$$W_g(kT_g) = \begin{bmatrix} \sigma_{x_{gps}}^2 & 0 & 0 \\ 0 & \sigma_{y_{gps}}^2 & 0 \\ 0 & 0 & \sigma_{\theta_{gps}}^2 \end{bmatrix} \quad (30)$$

2.4.3 State estimate

Innovation error:

$$v_g = Y_g(kT_g) - H_g(kT_g)\hat{X}(IT_i) \quad (31)$$

State estimation $\tilde{X}$:

$$\tilde{X}(kT_g) = \hat{X}(IT_i) + R_g(kT_g)v_g(kT_g) \quad (32)$$

2.4.4 Covariance matrix estimation $\tilde{P}$:

$$\tilde{P}(kT_g) = \hat{P}(IT_i) - R_g(kT_g)H_g(kT_g)\hat{P}(IT_i) \quad (33)$$

$$\text{여기서 } H_g(kT_g) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

이렇게 얻어진 최종 추정 결과 $\tilde{X}(IT_i)$, $\tilde{P}(IT_i)$ 는 다음 prediction step에서의 pre-state로서 사용한다.

2.5 노이즈 covariance 값 설정

설계한 EKF가 정확한 추정결과를 얻기 위해서는 process

noise model과 measurement noise model을 조절 해줄 필요가 있다. 사용되는 센서는 모두 그 성질이 다르고, 독립적으로 작동하기 때문에, 서로 independent하고, uncorrelated 하다는 가정하에 noise model을 설정해 주었다.

각 센서의 datasheet의 error performance와 기존의 연구를 참조해서 예측되는 노이즈 covariance 값을 먼저 잡아주고, 실험과 시뮬레이션을 통해서 보다 정확한 결과를 얻을 수 있도록 보정하면서 최종적인 노이즈 covariance 값을 설정할 것이다.

V. 시뮬레이션 및 실험

본 장에서는 가상의 센서 데이터를 통한 시뮬레이션 및 실외 주행을 통한 센서 데이터를 이용한 시뮬레이션

선으로 각각의 위치 추정 결과를 융합한 실외 다중 센서 융합 위치 추정 알고리즘을 검증한다.

1. 시뮬레이션

1.1 위치 추정

LRF를 이용한 실외 위치 추정을 위하여 벽과 나무로 이루어진 선분과 원으로 구성된 가상의 야외지도와 경로(그림 8)를 작성하였다. 이를 바탕으로 LRF 위치 추정 시뮬레이션을 수행한 뒤, 엔코더, GPS의 위치 추정 결과를 추가하고, 최종적으로 3 센서의 위치 추정 결과를 MU-EKF로 융합하여, 최종 위치 추정 결과를 얻었다.

LRF의 노이즈는 $\sigma = 1.5\text{cm}$ 로 설정하였고, 그 외 나머지 센서의 노이즈 covariance 값은 다음과 같이 실험적으로 얻었다.

Process noise: $\sigma_v = 0.5\text{m/s}$ $\sigma_\psi = 5\text{deg}$

Measurement 1 noise: $\sigma_{lrf_x} = 0.1\text{cm}$ $\sigma_{lrf_y} = 0.1\text{cm}$

$\sigma_{lrf_t} = 0.01\text{rad}$

Measurement 2 noise: $\sigma_{gps_x} = 7.5\text{m}$ $\sigma_{gps_y} = 7.5\text{m}$

$\sigma_{gps_t} = 5\text{deg}$

결과는 그림 9와 표 1과 같다. 위치 추정 단계가 진행 될 수록, 정확도가 향상되는 것을 확인 할 수 있다. 같은 전동 스쿠터 시스템에서, 엔코더와 GPS 2개의 센서를 융합하는 기존 알고리즘[1]의 결과(그림 9(f), 표 1(b))와 비교해 볼 때, 제안된 알고리즘에서 더 정확한 위치 추정 결과가 나타나는 것을 확인 할 수 있다.

그림 8. 시뮬레이션 환경.

Fig. 8. Simulation environments.

그림 9. 다중 센서 융합 위치 추정 시뮬레이션 결과.

Fig. 9. Multi-sensor fusion localization simulation result.

표 1. 다중 센서 융합 위치 추정 시뮬레이션 결과.

Table 1. Multi-sensor fusion localization simulation result.

(a) MU-EKF 시뮬레이션 결과				
Error	거리 오차 (m)		방향 오차 (rad)	
	평균	표준편차	평균	표준편차
Dead Reckoning	25.09	9.369	0.1735	0.0699
LRF Pose	0.009959	0.006087	0.0006	0.0003
GPS Pose	4.076	1.291	0.0531	0.0347
1st EKF Pose (LRF)	0.009806	0.006212	0.0124	0.0094
2nd EKF Pose (GPS)	0.007044	0.005725	0.0106	0.0068
(b) 기존 알고리즘 시뮬레이션 결과				
Error	거리 오차 (m)		방향 오차 (rad)	
	평균	표준편차	평균	표준편차
Dead Reckoning	25.09	9.369	0.1735	0.0699
GPS Pose	3.667	1.512	0.0731	0.0464
EKF Pose	0.5432	0.4179	0.0560	0.0415

1.2 Kidnapping

로봇의 위치가 갑자기 옮겨진 경우(kidnapping)에도, 제안된 알고리즘은 대처가 가능하다. 엔코더에 의한 Dead reckoning은 일시적으로 오차가 나타나지만, 주변 환경과의 matching을 통한 LRF의 정보와, 절대 좌표를 제공하는 GPS의 정보를 융합함으로써, 실제 로봇의 pose를 빠르게 추정할 수 있다. Kidnapping이 일어날 경우를 가정한 시뮬레이션 결과, Dead

그림 10. Kidnapping 시뮬레이션 결과.

Fig. 10. Kidnapping simulation result.

그림 11. 실외 환경 및 지도.

Fig. 11. Outdoor environment and map.

reckoning에 의한 단독 위치 추정은 위치가 크게 벗어나지만, MU-EKF에 의한 위치 추정 결과는 일시적인 오차를 빠르게 회복함을 확인할 수 있다.

2. 실외 측정데이터를 이용한 다중 센서 융합 위치 추정

2.1 야외 지도 작성

학교 주변의 한 지역을 선정하여, 다음과 같이 야외 지도를 작성하였다(그림 11).

2.2 직선 경로 실외 위치 추정 결과

위와 같은 환경에서, 직선 경로를 따라 주행하면서 얻은 측정 데이터를 이용해 제안된 알고리즘으로부터 실외 위치 추정을 수행하였다. 이 때의 노이즈 covariance 값은 시뮬레이션에서 얻은 값을 토대로, 실측 데이터를 분석하여 적합한 값을 찾아 주었다. 설정된 노이즈 covariance는 다음과 같다.

Process noise: $\sigma_v = 0.1\text{m/s}$ $\sigma_\psi = 15\text{deg}$

Measurement 1 noise: $\sigma_{lrf_x} = 1\text{cm}$ $\sigma_{lrf_y} = 1\text{cm}$

$\sigma_{lrf_t} = 0.5\text{rad}$

Measurement 2 noise: $\sigma_{gps_x} = 2.5\text{m}$ $\sigma_{gps_y} = 2.5\text{m}$

$\sigma_{gps_t} = 5\text{deg}$

결과는 다음과 같다. 실제 실외 환경에서도 EKF step이 진행될수록 보다 정확한 위치 추정 결과로 수렴함을 확인할 수 있다.

그림 12. 직선 경로 MU-EKF 위치 추정 실험 결과.

Fig. 12. Pose estimation using MU-EKF for line trajectory.

표 2. 다중 센서 융합 위치 추정 결과.

Table 2. Multi-sensor fusion localization result.

Error	거리 오차 (m)	
	평균	표준편차
Dead Reckoning	0.3774	0.3550
LRF Pose	0.2316	0.1298
GPS Pose	1.0925	0.5765
1st EKF Pose (LRF)	0.2299	0.1246
2nd EKF Pose (GPS)	0.2106	0.1221

그림 13. 직선-곡선 경로 MU-EKF 위치 추정 실험 결과.

Fig. 13. Pose estimation using MU-EKF for line-arc trajectory.

2.3 복잡한 경로에서의 실험

직선 경로 보다 좀 더 복잡하고 긴 구간을 주행하면 서 위치 추정을 수행하였다. 이 경우, 정확한 위치추정 오차를 계산하기에는 어려우므로, 스쿠터의 중간지점 및 최종 지점을 마크하여, 위치 추정 결과가 이 지점으로 효과적으로 수렴하는지의 여부를 살펴보겠다. 직진-우회전-좌회전-직진의 경로를 그리며 주행하였다.

직선에서의 실험에 비해, 엔코더의 위치 추정 결과오차가 훨씬 크게 누적 되고 있으며, 실제 주행지점과 크게 벗어나 볼 수 있다. 하지만 1, 2번째 update 된 위치 추정 결과는 기존의 주행 방향을 따라 위치 추정을 수행 하는 것을 확인할 수 있다.

VI. 결론 및 추후 과제

본 연구에서는 보다 정확하고 신뢰성 있는 위치 추정을 위한 다수의 센서 정보를 융합한 MU-EKF 알고리즘을 제안하

였고, 자동차형 로봇인 전동 스쿠터를 기반으로 실외 위치 추정 시스템을 구현하여 LRF, GPS, 엔코더 3개의 센서를 융합하여 그 성능을 검증하였다. 그 결과 보다 안정적이고 신뢰성 있는 위치 추정 결과를 얻기 위한 다중 센서 융합 실외 위치 추정 알고리즘을 제안 및 입증함으로써, 실외 환경에서도 보다 정확한 자동차형 로봇의 동작을 기대 할 수 있게 되었다.

추후의 과제로는, 우선 LRF에 의한 실외 위치 추정 알고리즘은, 원형 개체를 감지함으로써 무한해가 발생할 가능성을 줄였지만, 항상 결과를 보장하지는 못한다. 이러한 singular case에 대한 고려가 더 필요할 것이다. 또한 실외 환경은 2차원 환경으로 간략화 하기에는 너무 많은 정보를 가지고 있으므로, 센서의 tilting 등을 통해 둔턱, 계단 과 같은 개체 또한 감지 할 수 있도록 발전시킬 수 있을 것이다. 야외 실험 역시 보다 넓고 긴 경로에 대한 실험이 이루어짐과 동시에, 정량적인 위치 추정 정확도 비교를 위한 기준을 확립함으로써, 보다 완성도 있는 연구로 발전 시킬 수 있을 것이다.

또한 칼만 필터의 성능 향상을 위해, 보다 적합한 노이즈 코베리언스 값을 찾고, 센서간의 correlation을 고려하도록 설계함으로써 연구를 발전시킬 수 있을 것이다.

참고문헌

- [1] U. K. Song, "Development of a DGPS-based localization and semi-autonomous path following system for electric scooters," *M.S. Thesis*, Korea Advanced Institute of Science and Technology, p. 49, 2010.
- [2] H. J. Sohn, "Vector-based localization and map Building for mobile robots using laser range finders in an indoor environment," *Ph.D. Thesis*, Korea Advanced Institute of Science and Technology, p. 169, 2008.
- [3] T. Takahashi, "2D localization of outdoor mobile robots using 3D laser range data," *M.S. Thesis*, Carnegie Mellon University, p. 39, 2007.
- [4] J. Guivant, "High accuracy navigation using laser range sensors in outdoor applications," *International Conference on Robotics & Automation 2000*, vol. 4, pp. 3817-3822, 2000.
- [5] J. Wang, "Integration of gps/ins/vision sensors to navigate unmanned aerial vehicles," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XXXVII, Part B1, pp. 963-970, 2008.
- [6] M. Agrawal, "Real-time localization in outdoor environments using stereo vision and inexpensive," *The 18th International Conference on Pattern Recognition*, pp. 1063-1068, 2006.
- [7] D. Wilner, "Kalman filter algorithms for a multi-sensor system," *1976 IEEE Conference on Decision and Control including the 15th Symposium on Adaptive Processes*, pp. 570-574, 1976.
- [8] B. S. Rao, "Fully decentralized algorithm for multi-sensory Kalman filtering," *IEE PROCEEDINGS-D. Control Theory and Applications*, vol. 138, no. 5, pp. 413-420, 1991.
- [9] X. Feng, "An optimal sequential filter for the linear system with correlated noises," *CCDC '09. Control and Decision Conference*, pp. 5073-5078, 2009.
- [10] Y. Junjun, "Sequential fusion for asynchronous multi-sensor system based on kalman filter," *CCDC '09. Control and*

Decision Conference, pp. 4677-4681, 2009.

- [11] A. Smyth, "Multi-rate Kalman filtering for the data fusion of displacement and acceleration response measurements in dynamic system monitoring," *Mechanical Systems and Signal Processing*, vol. 21, no. 2, pp. 706-723, 2007.
- [12] S. Niwa, "Kalman filter with time-variable gain for a multisensor fusion system," *Proc. of the 1999 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 56-61, 1999.
- [13] W. C. Lim, "Development of a cost-effective safety system with curb detection and avoidance for electric scooters," *M.S. Thesis*, Korea Advanced Institute of Science and Technology, p. 72, 2010.
- [14] Intel PXA255 Processor Developer Manual, Intel (<http://www.intel.com>), 2003.
- [15] G. A. Borges, "Line extraction in 2D range images for mobile robotics," *Journal of Intelligent & Robotic Systems*, vol. 40, no. 3, pp. 267-297, 2004.

배 상 훈

2009년 한국과학기술원 전자전산학부 전기 및 전자공학 전공. 2011년 동 대학원 전기 및 전자공학과 석사. 2011년~현재 씨앤에스 테크놀로지 재직중. 관심분야는 모바일 로봇, 센서 융합, 위치 추정.

김 병 국

1952년 10월 5일생. 1975년 서울대학교 전자공학과 졸업. 1975년 한국과학기술원 전기 및 전자 공학과 석사. 1981년 동 대학원 박사. 1981년~1986년 우진계기(주) 연구실장. 1982년~1984년 University of Michigan 방문연구. 1986년~현재 한국과학기술원 전기 및 전자공학과 교수. 관심분야는 실시간 시스템, 로봇틱스, 임베디드 제어.