FastSLAM에서 파티클의 밀도 정보를 사용하는 향상된 Resampling 기법

An Improved Resampling Technique using Particle Density Information in FastSLAM

우 종 석*, 최 명 환, 이 범 희
(Jong Suk Woo, Myoung Hwan Choi, and Beom Hee Lee)

Abstract: FastSLAM which uses the Rao-Blackwellized particle filter is one of the famous solutions to SLAM (Simultaneous Localization and Mapping) problem that estimates concurrently a robot's pose and surrounding environment. However, the particle depletion problem arises from the loss of the particle diversity in the resampling process of FastSLAM. Then, the performance of FastSLAM degenerates over the time. In this work, DIR (Density Information-based Resampling) technique is proposed to solve the particle depletion problem. First, the cluster is constructed based on the density of each particle, and the density of each cluster is computed. After that, the number of particles to be reserved in each cluster is determined using a linear method based on the distance between the highest density cluster and each cluster. Finally, the resampling process is performed by rejecting the particles which are not selected to be reserved in each cluster. The performance of the DIR proposed to solve the particle depletion problem in FastSLAM was verified in computer simulations, which significantly reduced both the RMS position error and the feature error.

Keywords: FastSLAM, particle filter, particle resampling, particle density, clustering

I. 서론

SLAM은 알려지지 않은 환경에 대한 탐사나 항법 등의 영역에서 로봇이 자율적으로 임무를 수행하기 위한 방법으로 주행로봇(mobile robot) 분야에서 필수적인 기술이다. SLAM의 문제는 로봇이 환경에 대한 지도뿐만 아니라, 환경에서 자신의 위치까지 알지 못할 때 발생한다. 이러한 SLAM 문제 해결을 위해 두 가지의 중요한 방법이 있다. EKF-SLAM (Extended Kalman Filter SLAM)과 FastSLAM (Factored Solution to the Simultaneous Localization and Mapping)이 그것이다. EKF-SLAM은 지난 수년간 SLAM 문제에 대한 주요한 접근 방법으로 적용되어 왔다. 그러나 제곱승의 계산량의 증가(quadratic computational complexity)와 하나의 가정을 가지는 데이터 조합(single-hypothesis data association)이라는 두 가지 단점을 지니고 있다. 이러한 EKF-SLAM 문제의 대안적인 접근 방법으로 최근에 FastSLAM 알고리즘이 제안되었다[1]. 파티클 필터에 의해 로봇의 위치를 추정하고, 칼만 필터(Kalman filter)를 사용하여 지도를 작성하는 FastSLAM은 EKF-SLAM에 비해서 두 가지의 중요한 장점을 가진다. 첫째, SLAM의 경험적 확률(posterior)을 로봇의 위치 추정에 대한 경험적 확률과 지도 작성에 대한 경험적 확률로 인수분해(factorization)함으로써 선형적인 계산량(linear computational complexity)을 갖게 된다. 둘째, 각 파티클이 주변 환경 지도에 대한 데이터 조합(data association)을 수행함으로써 다수의 가정을 가진 데이터 조합(multi-hypothesis data association)이 가능하게 된다[2].

그러나 FastSLAM은 시간이 지남에 따라 그 성능이 퇴보한다는 단점을 가진다[1,3]. 이는 로봇의 위치를 추정하는 파티클들이 다양성(diversity)을 상실하기 때문이다. 파티클이 다양성을 상실하는 주요한 이유는 FastSLAM의 재추출 과정(resampling process) 때문이다. 확률적으로 낮은 중요성을 가지는 파티클이 제거됨에 따라 해당 파티클이 가지고 있는 로봇의 경로 정보가 버려지게 되고, 결국 로봇의 경로에 대한 공통된 history를 가지는 파티클들이 많아짐에 따라 파티클의 다양성을 상실하는 결과를 초래한다[4].

FastSLAM의 재추출 과정을 개선시키기 위한 많은 연구들이 이루어져 왔다. Bailey는 정확하게 불확실성(uncertainty)를 추정하는 필터의 능력인 FastSLAM의 일관성(consistency)에 대해 연구했다[3]. 이 연구는 FastSLAM이 짧은 시간 동안에는 일관성 있는 불확실성 추정값(consistency uncertainty estimate)을 생산하였으나, 파티클의 수와 특징점의 밀도가 유지됨에도 불구하고 시간이 흐름에 따라 성능이 저하됨을 보였다. 또한, 파티클이 다양성(diversity)를 잃어버리는 속도와 어떠한 매개변수(parameter)들이 추정오차(estimation error)에 영향을 끼치는지 등에 대해 실험을 통해 증명하고 있다. Gordon은 파티클의 수를 증폭시킴으로써 파티클 고갈 문제를 개선시키기 위한 연구를 진행했다[5]. 여기에서는 요구된 PDF (Probability Density Function)를 무작위 샘플(random sample)의 집합으로써 나타내고자 했다. 샘플의 수가 늘어남에 따라, 정확하고 샘플 수에 상응하는 PDF의 표현(representation)을 얻을 수 있다는 것이 이 연구의 주요 내용이다. 그러나 파티클의 수를 증폭시키는 특별한 안을 내놓지 못하였고, 샘플의 수가 늘어날수록 계산량이 증가하는 단점을 보였다. 이를 위해, Kwak은 neural network training 방법을 사용하여 파티클의 수를 제어하는 adaptive prior boosting 기법을 제안하였다[6]. 이 방법을 사용함으로써 파티클의 수를 증폭시킴에 따라 성능의 향상을 보였으나, 여전히 계산량 증가의 문제는 남아있었다. 또 다른 방법으로, Kim에 의해 제안된 GRR (Geometric

* 책임저자(Corresponding Author)
논문접수: 2008. 12. 24., 채택확정: 2009. 3. 4.
우종석, 이범희: 서울대학교 전기컴퓨터공학부
(woojs@snu.ac.kr/bhlee@asri.snu.ac.kr)
최명환: 강원대학교 전기전자공학부(mhchoi@kangwon.ac.kr)

Relation Resampling) 기법이 있다[7]. 이는 파티클 사이의 기하학적인 관계를 이용하여 FastSLAM의 재추출 과정에 적용한 방법이라 할 수 있다. 우선, KD-tree를 사용하여 파티클의 분포 정보를 알아낸 후, 파티클의 분포가 가장 높은 곳을 로봇의 현재 pose로 추정하여 재추출 과정을 실행하였다. 그러나 파티클이 많이 분포되지 않은 지역에 실제 로봇이 위치하는 상황이 생기면 파티클 고갈 문제가 발생하게 된다. 이러한 상황이 연속적으로 생기게 될 때, FastSLAM의 전체적인 성능이 퇴보하게 된다.

본 연구에서는 파티클 고갈 문제를 개선시키기 위해 파티클의 밀도정보를 이용한 새로운 재추출 기법을 제안한다. 이 기법은 파티클의 분포를 사용하여 클러스터링을 수행하고, 각 클러스터의 파티클 밀도에 따라 재추출 과정을 위한 가중치를 계산한다. 따라서 파티클이 많이 분포된 지역에 있는 파티클 뿐만 아니라 파티클이 많이 분포되지 않은 지역에 있는 파티클까지 고려함으로써 파티클 고갈 문제가 해결되었다.

본 논문의 구성은 다음과 같다. II 장에서는 FastSLAM 알고리즘의 전반적인 내용에 대해서 기술한다. III 장에서는 파티클의 밀도정보를 사용한 향상된 재추출 기법(DIR)에 대해 제안한다. IV 장에서는 컴퓨터 시뮬레이션을 통해, 기존 재추출 기법들과의 성능 비교를 통해 DIR의 향상된 성능을 검증하고, V 장에서 본 연구에 대한 결론을 내린다.

II. Factored Solution to the Simultaneous Localization and Mapping (FastSLAM)

FastSLAM은 SLAM의 경험적 확률(posterior)을 다음과 같이 인수분해 함으로써 로봇의 위치와 지도 작성을 독립적으로 수행하는 알고리즘이다[2].

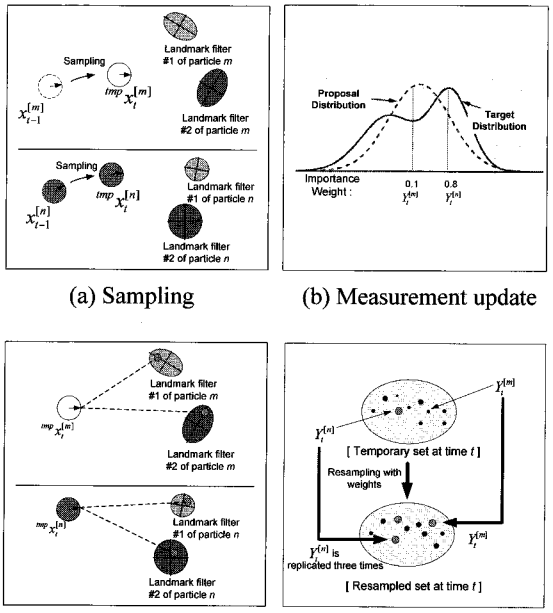
$$\begin{aligned} & p(x_{1:t}, M \mid z_{1:t}, u_{1:t}, c_{1:t}) \\ &= p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t}) p(M \mid x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) \\ &= p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t}) \prod_{n=1}^{N_f} p(m_n \mid x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) \end{aligned} \quad (1)$$

여기에서 $x_{1:t}$는 시간 t까지의 로봇의 위치(robot pose)를, M은 지도(map)를, $z_{1:t}$, $u_{1:t}$, $c_{1:t}$는 각각 시간 t에서의 센서정보(measurements), 제어정보(controls), 일치정보(correspondences)를 나타낸다. 여기에서 m_n은 지도 M에서의 n번째 특징점이고, N_f는 특징점의 개수다. FastSLAM은 $p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t})$에 의해 나타내어진 바와 같이, 로봇 경로에 대한 확률(posterior)을 계산하기 위해 파티클 필터를 사용하였다. 지도에서 각 특징점에 대해, FastSLAM은 그것의 위치 $p(m_n \mid x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t})$에 대해 분리된 추정함수(estimator)를 사용하였고, 여기에서 n은 1부터 N_f까지의 수이다. 특징점 추정함수는 로봇 경로에 의해 영향을 받는데, 이는 각 특징점 추정함수의 분리된 복사본이 있다는 것을 의미한다. 인수분해(factorization)로 인하여, FastSLAM은 각 특징점마다 분리된 EKF를 유지할 수 있고, 따라서 EKF-SLAM보다 더 효율적으로 업데이트된 추정값을 얻을 수 있다. FastSLAM에서는 각 특징점 추정값을 독립적으로 유지시킴에 따라, 지도를 합칠 때 공분산(covariance)이 제곱승으로 늘어나는 계산량을 회피할 수 있다. 시간 t에서의 파티클 $Y_t^{[k]}$는 다음과 같이 나타낼 수 있다.

$$Y_t^{[k]} = \left\langle x_t^{[k]}, \mu_{1,t}^{[k]}, \Sigma_{1,t}^{[k]}, \ldots, \mu_{N_f,t}^{[k]}, \Sigma_{N_f,t}^{[k]} \right\rangle \quad (2)$$

여기에서 $[k]$는 파티클의 index를 나타내고, $x_t^{[k]}$는 시간 t에서의 로봇의 위치 추정값이다. $\mu_{n,t}^{[k]}$와 $\Sigma_{n,t}^{[k]}$는 k번째 파티클과 관련된 n번째 특징점 위치를 나타내는 Gaussian의 평균값(mean)과 공분산(covariance)이다. 이 요소들은 k번째 파티클 $Y_t^{[k]}$를 형성하고, 파티클 집합에는 N_p개의 파티클과 N_f개의 특징점 추정값이 있다.

FastSLAM 2.0 알고리즘 수행 과정은 그림 1을 통해 간단하게 살펴볼 수 있다. 그림 1(a)에서 보는 바와 같이, 각 파티클은 센서정보(measurement) z_t를 고려한 제안분포(proposal distribution)를 사용하여 로봇의 위치정보(pose)를 추출하였다. 추출된 모든 로봇 위치정보들은 임시적인 파티클 집합을 구성한다. 그 후, 그림 1(b)에 나타난 바와 같이, 각 파티클은 센서정보(measurement) z_t와 추출된 pose ${}^{tmp}x_t^{[k]}$를 기반으로 특징점에 대한 경험적 확률(posterior)을 업데이트한다. 다음 단계는 다음과 같이 k번째 파티클의 가중치를 계산하는 것



(a) Sampling (b) Measurement update

(c) Importance weight (d) Resampling

그림 1. FastSLAM 2.0 알고리즘. (a) 추출 과정 후의 두 개의 파티클의 위치정보(pose): 제어정보(control) 입력과 현재의 센서정보(measurement)를 사용한 ${}^{tmp}x_t^{[m]}$와 ${}^{tmp}x_t^{[n]}$. tmp는 파티클이 임시적인 파티클 집합에 포함되어 있음을 의미. (b) 센서정보(measurement)의 업데이트. (c) 가중치 부여. (d) 재추출 과정. 여기에서 파티클은 집합 안에서 복제되거나 제거.

Fig. 1. The FastSLAM 2.0 algorithm. (a) Pose sampling of two particles: ${}^{tmp}x_t^{[m]}$ and ${}^{tmp}x_t^{[n]}$ using control input and current measurement where the superscript tmp means a particle is included in a temporary particle set. (b) Measurement update. (c) Importance weight. (d) Resampling. Particles are replicated or rejected.

이다.

$$w_t^{[k]} = \frac{\text{target distribution}}{\text{proposal distribution}}$$

그림 1(c)를 보면, 목표분포(target distribution)가 제안분포(proposal distribution)보다 큰 지역은 파티클이 높은 가중치를, 목표분포가 제안분포보다 작은 지역은 파티클이 낮은 가중치를 갖게 된다. 그림 1(d)에서 보는 바와 같이, 높은 가중치를 가진 파티클 $Y_t^{[n]}$ 은 낮은 가중치를 가진 파티클 $Y_t^{[m]}$ 이 재추출 과정에서 제거되는 반면에 세 배가 복제됨을 확인할 수 있다. 이것은 제거된 파티클들의 로봇 경로와 특징점 추정값이 상실됨을 의미하고, 파티클 고갈(particle depletion) 문제로 연결된다[3]. 따라서 재추출 과정은 파티클의 다양성(particle diversity)에 영향을 끼치는 중요한 과정이다.

FastSLAM의 성능을 저하시키는 두 가지 주요한 원인이 있다. 첫 번째 원인은 추출 과정에서의 샘플 빈곤(sample impoverishment) 문제이다. 이것은 목표분포(target distribution)와 제안분포(proposal distribution)의 불일치로 인해 발생한다. 결과적으로, 재추출 과정에서는 추출 과정에서 발생한 확률적으로 낮은 파티클들은 제거되고, 높은 가중치를 부여 받은 파티클들이 선택됨에 따라 파티클 고갈 문제를 야기하게 된다. 샘플 빈곤 문제의 또 다른 원인으로는 파티클 수의 부족을 들 수 있겠다. 파티클이 많으면 많을수록 샘플 빈곤 문제는 약해지게 되나, 이는 계산량의 증가 문제를 가져오게 된다. 이 문제는 adaptive prior boosting 기법에 의해 성능이 개선되었다[6]. FastSLAM의 성능을 저하시키는 두번째 원인은 파티클 고갈(particle depletion) 문제이다. 이는 재추출 과정이 반복됨에 따라 높은 가중치를 가진 파티클만이 복제되면서 독특한 특성을 가진 파티클의 수가 점점 줄어들게 되는 것이다. 즉, 파티클이 다양성(particle diversity)을 상실하게 된다. 일반적으로, 파티클의 다양성은 추정값의 정확성을 더 향상시킨다. 따라서, 파티클의 다양성을 최대화시킬 필요가 있다.

III. Density Information-based Resampling (DIR)

본 연구에서는 FastSLAM의 문제점을 해결하기 위해, 파티클의 밀도정보를 이용하는 재추출 기법인 DIR을 제안하였다. 파티클 필터를 사용하는 FastSLAM에서, 파티클들이 많이 모여 있는 지역에 실제 로봇이 위치할 확률이 높다. GRR은 이 지역에 분포한 파티클들에 높은 가중치를 부여함으로써 FastSLAM의 성능을 향상시켰다[7]. 하지만, 다른 지역에 있는 파티클들은 상대적으로 지나치게 낮은 가중치를 부여 받게 되기 때문에 과도하게 제거될 수 있고, 이는 파티클 고갈 문제를 심화시킬 수 있다. 따라서, 파티클들이 많이 분포한 지역에 있는 파티클 뿐만 아니라 다른 지역에 있는 파티클의 가중치를 적절하게 유지함으로써 파티클 다양성을 유지할 필요가 있다. DIR은 파티클 분포를 이용하여 유사성이 높은 파티클들을 클러스터링하고 각 클러스터의 밀도정보를 사용하여 파티클 다양성을 유지시킨다. 많은 클러스터링 방법 중에서 DIR에서 사용하는 클러스터링 방법은 밀도 기반 클러스터링 방법이다[8-11] (그림 2). 이 방법은 특정한 점을 중심으로 기준점 자신을 포함하여 Eps(반지름) 내의 점들의 개수를 세는 것이다. 여기에서는 핵심점(core point) 근처에 위치하

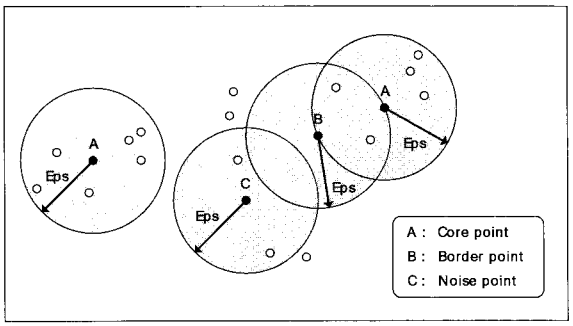


그림 2. 밀도기반(density-based) 클러스터링 방법.
Fig. 2. Density-based clustering method.

는 점을 경계점(border point)으로 나타내고, 객체의 경계점이 적어도 객체의 최소한의 MinPts(cluster 안 점들의 최소 개수)를 가지고 있는 경우는 핵심점으로 표현하며, 핵심점도 경계점도 아닌 점들은 잡음점(noise point)으로 나타낸다. 이 방법은 점의 밀도들이 정해진 반지름에 의존적이라는 단점이 있으나, 밀도기반 정의를 이용하므로 잡음에 강하고, 불규칙한 모양과 크기의 클러스터를 다룰 수 있다는 장점이 있다.

알고리즘 수행을 위해 우선 파티클의 밀도가 무엇인지에 대해 정의할 필요가 있다. 여기에서는 밀도를 임의의 파티클의 인접 지역 안에 있는 파티클의 개수로 정의하였다. 다음 식이 파티클의 밀도를 계산하기 위해 사용되었다.

$$Density(\alpha) = size(\{\beta \mid Dist(\beta,\alpha) \le T\}) \tag{3}$$

α와 β 두 개의 파티클이 있다고 하자. 위 식에서 보는 바와 같이, $Density(\alpha)$ 는 파티클 α의 밀도를, $size(X)$ 는 파티클 집합 X의 크기를 나타낸다. $Dist(\beta,\alpha)$ 는 파티클 α와 β 사이의 유사성을 거리를 토대로 나타낸 함수이다. T는 주어진 문턱값 (threshold)이다. 거리함수(distance function)에는 많은 종류가 있는데, 아래 식과 같은 Minkowsky distance가 일반적으로 사용된다.

$$Dist_\lambda(\alpha,\beta) = \left[\sum_{i=1}^{d} |\alpha_i - \beta_i|^\lambda\right]^{1/\lambda} \tag{4}$$

여기에서, λ 는 양의 정수이고, λ 가 2일 때의 Euclidean

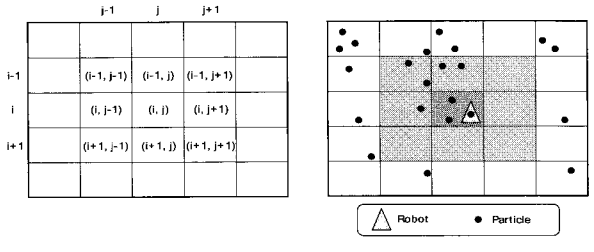


그림 3. (a) 파티클 공간의 격자화 (b) 인접 셀의 정의. 로봇의 이동에 따라 분포된 파티클의 공간을 2차원의 직사각형 셀로 분할.
Fig. 3. (a) The grid partition of a particle space (b) The definition of neighborhood cells. Particle space distributed as robot moves is divided into the 2-dimensional rectangular cells.

distance를 사용한다.

DIR 알고리즘의 구체적인 수행 과정은 다음과 같다. 우선, 파티클의 공간을 격자화한 후 임의의 셀을 중심으로 인접 셀을 정의한다. 그림 3(a)에서 보는 바와 같이, 로봇의 이동에 따라 파티클의 공간을 격자화 한다. 파티클의 공간을 m_i의 인터벌로 나눈다 (i = 1,2,…,d). 그러면, 전체 파티클의 공간은 M-직사각형 안에 작은 직사각형 셀들로 나눠지게 된다 ($M = \prod_{i=1}^{d} m_i$). 각 차원(dimension)에 대한 인터벌은 0부터 m-1까지 번호로 구별된다. 각 셀은 배열된 수를 가지고 분류된다. 예를 들어 2차원 공간의 경우, 셀 C_{35}는 차원 1의 인터벌 3과 차원 2의 인터벌 5 안의 셀이 된다.

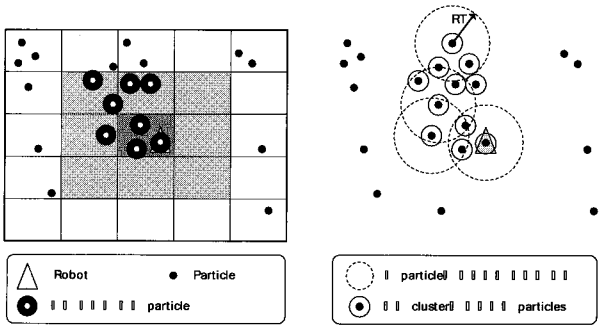
다음은 그림 3(b)와 같이 인접 셀을 정의하는 단계이다. 셀 $C_{i_1 i_2 \cdots i_d}$ 와 $C_{j_1 j_2 \cdots j_d}$ 는 아래 식을 만족할 때, 인접 셀이 성립이 된다.

$$|i_p - j_p| \le 1, \quad (1 \le p \le d) \tag{5}$$

여기에서, $i_1 i_2 \cdots i_d$ 는 셀 $C_{i_1 i_2 \cdots i_d}$ 의 인터벌의 배열된 수이고, $j_1 j_2 \cdots j_d$ 는 셀 $C_{j_1 j_2 \cdots j_d}$ 의 인터벌의 배열된 수이다. 모든 셀은 정의에 의해 인접 셀을 가지게 되고, 2차원의 경우에 각 셀은 자신을 포함하여 최대 9개, 최소 4개의 인접 셀을 갖게 된다. 따라서, 파티클 α 의 밀도를 계산할 때, 셀 C_α 와 그것의 인접 셀의 파티클 사이의 거리만을 계산하면 된다. 그러므로, 파티클 α 에 대하여 인접 셀의 범위를 넘어서는 파티클과의 거리는 고려하지 않아도 된다. 파티클의 공간을 격자화하고, 인접 셀의 정의가 완료되면, RT (Radius Threshold)를 계산해야 한다. 파티클의 밀도를 정의하는 데 있어서, RT의 범위를 결정하는 일은 매우 중요하다. 만약 RT가 너무 작게 설정된다면, 각 파티클의 밀도는 매우 작게 될 것이다. 그런 경우, 파티클의 밀도는 매우 작은 범위 안에 놓이게 되고 파티클의 분포를 묘사하기가 어렵게 된다. 만약 RT가 너무 크다면, 파티클의 밀도는 매우 커지게 되고 밀도값이 다양성을 잃게 된다. 따라서, RT값은 모든 파티클 사이의 거리의 최소값보다는 커야 하고, 최대값보다는 작아야 한다. 다음 식을 통하여 RT를 계산되었다.

$$RT = \frac{mean(Dist)}{d \times coefRT} \tag{6}$$

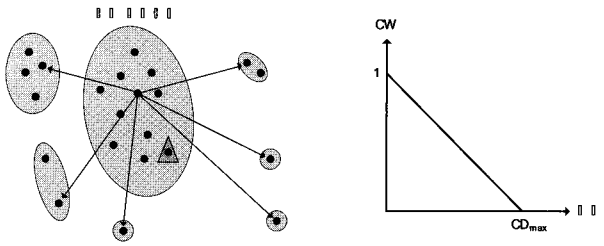
$mean(Dist)$ 는 파티클 사이의 평균거리를 나타내고, d는 차원(본 논문에서는 2차원 적용), $coefRT$ 는 좋은 결과를 산출하기 위한 조절 가능한 RT의 계수로 실험적으로 $coefRT$ 가 50일 때 좋은 결과를 얻을 수 있었다. RT의 계산이 끝난 뒤에는 DT (Distance Threshold)를 계산하였다. 클러스터링의 결과는 DT 값에 의해 큰 영향을 받는다. 만약 DT값이 너무 작다면, 인접한 클러스터들이 하나로 통합되었다. 반대로 DT 값이 너무 크다면 하나의 클러스터로 이루어져야 될 파티클이 여러 개의 클러스터로 나눠지고, 많은 파티클이 이상치(outlier)로써 잘못 판단될 수 있다. 다음 식을 통하여 DT를 계산할 수 있다.



(a) (b)

그림 4. (a) 거리판별을 위한 파티클의 선택 (b) 밀도를 기준으로 한 클러스터링.

Fig. 4. (a) The selection of particles for the distance discrimination (b) The density-based clustering.



(a) (b)

그림 5. 재추출 과정. (a) 파티클의 수가 가장 많은 클러스터가 핵심 클러스터. 각 클러스터에서 대표 파티클을 선정. 핵심 클러스터의 대표 파티클과의 거리를 계산. (b) 각 클러스터에서 추출할 파티클의 수 결정을 위해, (a)에서 계산된 거리에 따라 CW(Cluster Weight)를 계산.

Fig. 5. Resampling. (a) The core cluster is a cluster which includes the largest number of particles. A representative particle in each cluster is selected, and the distances between the representative particle in the core cluster and the representative particles in other particles are computed. (b) CW(Cluster Weight) is computed based on the distance computed in (a) to decide the number of particles to be sampled in each cluster.

$$DT = \frac{mean(Density)}{\log_{10}(n)} \times coefDT \tag{7}$$

$coefRT$ 는 조절 가능한 DT 계수로써 0.7과 1사이의 범위를 가지며, 실험적으로 0.95일 때 좋은 결과를 얻을 수 있었다. 주어진 파티클의 집합에 대하여, RT와 DT 둘 다 식 (6)과 (7)을 사용하여 기계적으로 계산하였다. RT와 DT 값이 설정된 후에는 이 값들에 따라 파티클을 클러스터링하게 된다. 모든 파티클에 대해서 각각의 파티클을 중심으로 RT 이내의 밀도가 DT 이상인 파티클을 확인하여 클러스터링한다(그림 4).

만약 두 파티클 사이의 거리가 RT를 벗어나더라도, 같은 클러스터에 속하는 공통된 파티클을 소유하고 있다면, 함께 클러스터링한다. 이러한 조건을 만족시키지 못하는 파티클은

그 자신이 하나의 클러스터를 형성한다. 파티클의 클러스터링이 수행된 후 그림 5와 같이 재추출 과정을 수행한다. 먼저, 클러스터 중에서 가장 밀도가 높은 클러스터를 핵심 클러스터로 설정하고, 핵심 클러스터를 중심으로 각 클러스터까지의 거리를 구한다. 이 거리를 기준으로 각 클러스터의 가중치가 다음과 같이 계산된다.

$$CW = -\frac{1}{CD_{\max}} \times CD + 1 \tag{8}$$

CW(Cluster Weight)은 각 클러스터가 가지는 가중치이며, CD(Cluster Distance)는 핵심 클러스터를 중심으로 각 클러스터까지의 거리이다. CD_{max}는 가장 멀리 떨어져 있는 클러스터까지의 거리값이 된다. 이렇게 부여된 가중치에 따라 각 클러스터에서 재추출될 파티클의 개수가 결정된다.

IV. 시뮬레이션 결과

그림 6은 50개의 특징점을 가진 지도의 시뮬레이션 환경을 나타내고 있다. 지도의 크기는 729m^2(24m × 33m)이다. 특징점은 작은 별 모양으로 나타냈고, 로봇에 의해 추정된 특징점의 위치는 작은 검정색 원으로 표시했다. 로봇은 큰 원으로 나타내었으며 파티클은 로봇 안이나 주위의 작은 점들로 나타냈다. 로봇의 주어진 경로는 실선으로, 파티클에 의해 추정된 최적의 경로는 점선으로 표시하였다. 루프폐쇄(loop-closure) 지점은 A, B, C 지점에서 발생하였다. 루프폐쇄는 로봇이 루프에서 특징점을 관측하고 다시 그 특징점을 보았을 때 나타나는 루프의 불일치성의 정도에 대한 문제다. 로봇은 큰 별모양의 출발지점을 시작으로 A, B, C의 순서대로 점선으로 표시된 주어진 경로를 따라 움직인다. 시뮬레이션에서 모든 파티클의 가중치는 재추출 과정 후에 0.1로 초기화되고, 파티클은 총 50개가 사용되었다. 실제와 같은 환경 구성을 위해 로봇의 동적 모델(motion model)과 센서정보 모델(measurement model)에서의 오차에 대해 고려하였다. 동적모델은 다음 식과 같이 유한의 분산을 가지는 zero-centered random variable에 의해 모델링할 수 있다.

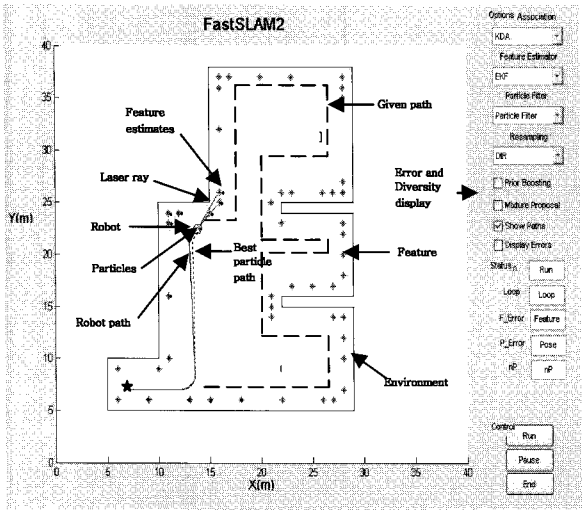


그림 6. 시뮬레이션 환경: 50개의 특징점을 가진 지도.
Fig. 6. Simulation environment: sparse map with 50 features.

$$\begin{pmatrix}\hat{v}\\ \hat{w}\end{pmatrix} = \begin{pmatrix}v\\ w\end{pmatrix} + \begin{pmatrix}\varepsilon_{\alpha_1|v|+\alpha_2|w|}\\ \varepsilon_{\alpha_3|v|+\alpha_4|w|}\end{pmatrix} \tag{9}$$

여기에서 v와 w는 로봇의 병진속도와 회전속도이며, $\hat{v}$와 $\hat{w}$는 표준편차 b를 가진 zero-mean error variable ε_b를 고려한 추정치이다. α_1에서 α_4는 로봇의 odometry 센서의 정확성을 나타내는 매개변수 값이다. Odometry 센서가 부정확할수록 매개변수의 값은 더 커진다. 여기서는 병진속도와 회전속도에서의 정규분포(normal distribution)의 분산을 매개변수 값을 조절하여 (0.3m^2/s^2, 3°/s^2)로 설정하였다.

센서정보 역시 로봇의 속도에 영향을 받는데 다음과 같이 모델링할 수 있다.

$$\begin{pmatrix}\hat{r}\\ \hat{\theta}\end{pmatrix} = \begin{pmatrix}r\\ \theta\end{pmatrix} + \begin{pmatrix}\varepsilon_{\alpha_5|v|+\alpha_6|w|}\\ \varepsilon_{\alpha_7|v|+\alpha_8|w|}\end{pmatrix} \tag{10}$$

여기에서 r와 θ는 특징점에 대한 거리와 방향이며, $\hat{r}$와 $\hat{\theta}$는 표준편차 b를 가진 zero-mean error variable ε_b를 고려한 추정치이다. α_5에서 α_8는 로봇 센서의 정확성을 나타내는 매개변수 값이다. 본 연구에서는 센서정보에서의 거리와 방향에 대한 정규분포(normal distribution)의 분산을 매개변수 값을 조절하여 (0.1m^2, 1°)로 설정하였다. 제어시간(control time)과 관측시간(observation time)은 25ms와 200ms이다. DIR과 기존 재추출 기법의 특징오차(RMS feature error)와 위치오차(position error)에 대한 결과는 1GB RAM을 가진 3.0GHz PC를 사용하여 얻어진 10개의 시뮬레이션 결과에 대한 평균값으로 나타내었다.

DIR 알고리즘은 FastSLAM의 기존 재추출 기법이었던 PSR (Partial Stratified Resampling), 파티클의 기하학적 관계를 이용한 재추출 기법인 GRR (Geometric Relation Resampling)과 위치오차(RMS position error)와 특징오차(feature error)를 비교함으로써 성능의 향상을 증명하였다. SLAM에서의 객체(objective)는 환경에 대한 지도를 올바르게 구조화하고, 이 지도를 기반으로 로봇은 정확한 localization을 실행한다. 그러므로, 위치오차(RMS position error)와 특징오차(feature error)는 SLAM의 성능 평가에 중요한 요소라 할 수 있다[12].

$$f_{rmse} = \sqrt{\frac{1}{M \cdot N}\sum_{k=1}^{M}\sum_{i=1}^{N}(\theta_i - \mu_{i,t}^{[k]})^2} \tag{11}$$

$$p_{rmse} = \sqrt{\frac{1}{M}\sum_{k=1}^{M}(x_t^{true} - x_t^{[k]})^2} \tag{12}$$

여기에서, f_{rmse}와 p_{rmse}는 각각 특징오차(RMS feature)와 위치오차(position error)를 나타낸다. θ_i와 x_t^{true}는 특징점 i와 로봇의 실제 위치이고, M은 파티클의 수를, N은 파티클이 가지는 특징점의 개수를 나타낸다.

그림 7은 PSR, GRR, DIR 재추출 알고리즘의 특징오차(RMS feature error)를 보이고 있다. PSR은 작은 점선, GRR은 긴 점선, DIR은 실선으로 표시하였다. 그래프에서 보는 바와 같이 A 지점에서 모든 알고리즘의 특징오차(RMS feature error)가 급격하게 증가하는 것을 확인할 수 있다. 이는 많은

특징점이 관찰되고, 로봇이 선회하기 때문에 생기는 현상이다. 로봇의 회전운동(rotation)에서 직선운동(translation)보다 odometry error가 더 커진다. A 지점에서 B 지점까지 PSR은 0.35m 이상, GRR은 0.15m 이상 error가 증가하는 반면, DIR은 A 지점에서 0.05m error가 증가한 후, B 지점까지 error가 0.02m 안팎으로 변화가 거의 없음을 알 수 있다. B 지점에서 C 지점까지 PSR과 DIR의 특징오차(feature error)는 약간씩 상승하긴 했으나 거의 변화가 없었다. 그러나, GRR은 0.1m 이상 error가 급격히 감소했다가, C 지점까지 다시 0.1m 이상 오차가 증가하였다. C 지점에서는 PSR과 DIR이 다소 느리긴 하였지만, 모든 알고리즘이 특징오차(RMS feature error)를 감소시켰음을 알 수 있다. 그래프에서 나타난 바와 같이, DIR 알고리즘은 다른 알고리즘에 비해 적은 특징오차(feature error) 폭을 유지하면서 안정적으로 특징점을 감지했음을 알 수 있다.

그림 8은 각 재추출 알고리즘의 위치오차(RMS position error)를 보이고 있다. 그래프에서 보는 바와 같이, 특징오차(feature error)와는 다르게, 각 지점마다 위치오차(position error)

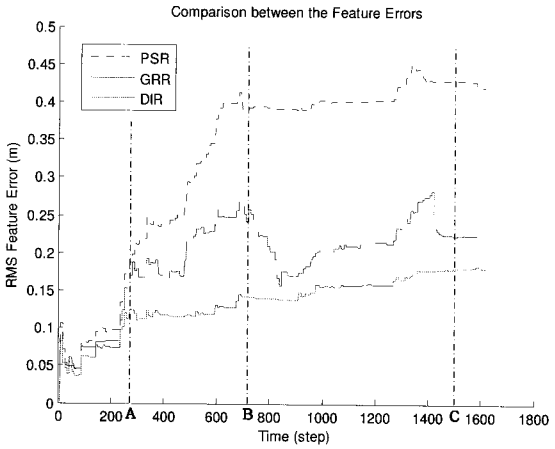


그림 7. 재추출 알고리즘에 대한 RMS feature error의 비교.

Fig. 7. Comparison between the RMS feature error of resampling algorithms.

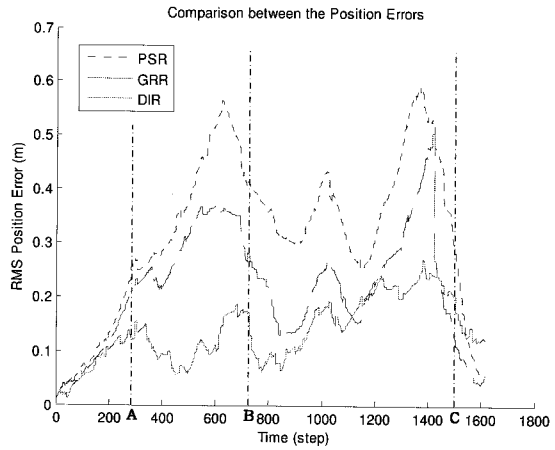


그림 8. 재추출 알고리즘에 대한 RMS position error의 비교.

Fig. 8. Comparison between the RMS position error of resampling algorithms.

가 줄어들고 있다. 이는 파티클이 과거보다 더 정확한 초기 특징점의 추정치를 고려하여 추출되기 때문이다. 다시 말하자면, 특징오차(feature error)는 새로운 관측정보(observation)가 과거의 특징점 추정치에 영향을 줄 수 없기 때문에 감소되지 않는다. 결과적으로, 모든 재추출 알고리즘의 위치오차(position error)가 작은 값으로 수렴됨을 알 수 있다. 제안한 DIR 알고리즘이 다른 알고리즘에 비하여 C 지점 부근에서 더디게 작은 오차 값으로 수렴하고 있으나, 전체적으로 작은 오차 폭을 유지하고 있음을 알 수 있다. A 지점과 C 지점 사이에서, PSR은 최대 0.6m, 최소 0.25m의 오차 값을 갖는다. GRR은 최대 0.5m, 최소 0.13m의 오차 값을 갖는다. 반면에 DIR은 최대 0.25m, 최소 0.05m의 오차 값을 갖는다. 따라서, 다른 두 재추출 알고리즘에 비하여 위치오차(RMS position error)를 줄이면서 안정적으로 자신의 위치를 계산하고 있음을 알 수 있다.

V. 결론

본 연구에서는, FastSLAM에서의 파티클 고갈 문제를 개선시키기 위하여 파티클의 밀도정보를 이용한 재추출 기법인 DIR 알고리즘을 제안하였다. 이 알고리즘은 서로 인접한 파티클들에 대해 여러 개의 클러스터를 구성한 후, 각 클러스터 사이의 거리를 기반으로 클러스터들의 가중치를 계산한다. 그리고 가중치에 따라 각 클러스터에서 재추출할 파티클의 개수를 결정함으로써, 파티클이 많이 분포된 지역의 파티클 뿐만 아니라 다른 지역에 분포한 파티클까지 재추출하는 기법이다. 따라서 DIR을 사용하는 FastSLAM은 공간 내에서 밀도에 따라 나누어진 파티클을 전체적으로 이용함으로써, 파티클의 다양성을 유지하면서 시간의 흐름에 따라 나타나는 파티클 고갈 현상을 최소화하게 된다. 컴퓨터 시뮬레이션을 통하여 기존 재추출 기법이었던 PSR, GRR에 의한 FastSLAM의 위치오차(RMS position error)와 특징오차(feature error)를 비교함으로써 DIR의 향상된 성능을 확인할 수 있었다.

참고문헌

[1] M. Montemerlo, "FastSLAM: A factored solution to the simultaneous localization and mapping problem with unknown data association," Ph.D. thesis, Carnegie Mellon University, 2003.

[2] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," *IEEE International Conference on Robotics and Automation*, pp. 1985-1991, 2003.

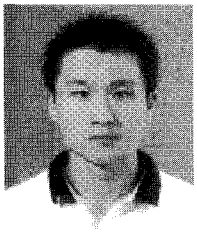
[3] T. Bailey, J. Nieto, and E. Nebot, "Consistency of the FastSLAM algorithm," *IEEE International Conference on Robotics and Automation*, pp. 424-429, 2006.

[4] S. Thrun, W. Burgard, and D. Fox, "Probabilistic robotics," MIT Press, Cambridge, 2005.

[5] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-gaussian Bayesian state estimation," *IEE-Proceedings- F*. vol. 140, no. 2. pp. 107-113.

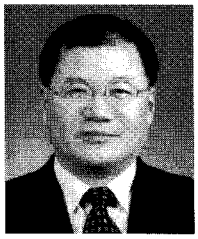
[6] N. Kwak, I. K. Kim, and H. C. Lee, *et al.*, "Adaptive prior boosting technique for the efficient sample size in FastSLAM," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[7] I. K. Kim, N. Kwak, H. C. Lee, and B. H. Lee, "Improved particle filter using geometric relation between particles in FastSLAM," *Robotica*, Oct. 2008. (accepted).
[8] Y. Zhao and J. Song, "GDILC:A grid-based density-isoline clustering algorithm," *Proceedings of International Conference on Info-tech and Info-net*, 2001.
[9] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *in Proc. KDD*, pp. 226-231, 1996.
[10] E. Schikuta, "Grid clustering: An efficient hierarchical clustering method for very large data sets," *Proc. 13th Int. Conf. on Pattern Recognition*, vol. 2, pp. 101-105, 1996.
[11] K. Alsabti, S. Ranka, and V. Singh, "An efficient K-means clustering algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, issue. 7, pp. 881-892, 2002.
[12] N. Kwak, G. W. Kim, and B. H. Lee, "A new compensation technique based on analysis of resampling process in FastSLAM," *Robotica*, vol. 26, no. 2, pp. 205-217, Mar. 2008.
[13] S. Lee and S. Lee, "Recursive particle filter with geometry constraints for SLAM," *IEEE. Int. Conf. Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, pp. 395-401, 2006.
[14] S. Thrun, D. Fox, and W. Burgard, "Monte carlo localization with mixture proposal distribution," *American Association for Artificial Intelligence*, pp. 859-865, 2000.
[15] G. Grisetti, G. D. Tipaldi, and C. Stachniss, *et al.*, "Fast and accurate SLAM with rao-blackwellized particle filters," *Robotics and Autonomous Systems*, vol. 55, pp. 30-38, Jan 2007.
[16] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using fastslam," *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, pp. 185-1991.
[17] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," *Proceedings of IEEE International 1995 Conference on Neural Networks*, vol. 4, pp. 1942-1948, 1995.



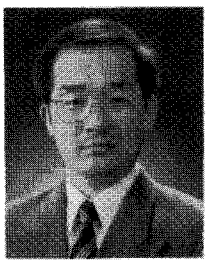
우 종 석

2002년 육군사관학교 전자공학과 졸업. 2009년 서울대학교 대학원 전기컴퓨터공학부 석사졸업. 관심분야는 FastSLAM.



이 범 희

1978년 서울대학교 공과대학 전자공학과 졸업. 1980년 서울대학원 전자공학과 석사졸업. 1985년 Univ. of Michigan. Computer, Information & Control Eng 박사 졸업. 1985년~1987년 미국 퍼듀대학교 전기공학과 조교수. 1987년~현재서울대학교 전기컴퓨터공학부 교수. 관심분야는 멀티에이전트 시스템 Coordination, Control, and Application.



최 명 환

1986년 서울대학교 공과대학 제어계측공학과 졸업. 1988년 서울대학원 제어계측공학과 석사졸업. 1992년 서울대학원 제어계측공학과 박사졸업. 1992년~현재 강원대학교 전기전자공학부 교수. 관심분야는 의료영상, 로봇공학.