Journal of the Korea Academia-Industrial cooperation Society
Vol. 12, No. 6 pp. 2760-2769, 2011

DOI : 10.5762/KAIS.2011.12.6.2760

자율 로봇의 오류 보정을 위한 이정표 상태 생성 방법

한현구[1*]
[1]한국외국어대학교 컴퓨터공학과

Milestone State Generation Methods for Failure Handling of Autonomous Robots

Hyungoo Han[1*]
[1]**Department of Computer Science and Engineering, Hankuk University of Foreign Studies**

요 약 지능형 자율 로봇은 주어진 목표를 달성하기 위하여 계획을 수립한다. 계획은 로봇이 목표를 달성하기 위한 행위들의 나열이며 모든 행위들이 순차적으로 그리고 성공적으로 실행되었을 때 목표를 달성하게 된다. 그러나 복잡하고 역동적인 현실 세계에서는 여러 가지 요인에 의하여 발생할 수 있는 예상하지 못한 상황으로 인하여 로봇이 계획한 행위들을 더 이상 실행하지 못하는 경우가 발생할 수 있다. 그러므로 지능형 자율 로봇은 이러한 돌발 상황을 적절히 대처하여 주어진 임무를 성공적으로 완수할 수 있는 효과적인 처리 방법을 가지고 있어야 한다. 이정표 상태를 이용한 계획 보정 방법은 이러한 상황을 효과적으로 처리할 수 있는 방법으로서 다른 계획 보정 방법들의 장점을 가지고 있다. 본 논문은 이정표 상태를 생성하는 방법으로 후진 방법과 이정표 상태들을 구성하는 조건들에 가중치를 부여하는 방법을 제안한다. 오류 보정 방법은 부여된 가중치를 보정해야 할 조건들의 우선순위로 사용할 수 있다. 후진 방법에 의하여 생성된 이정표 상태들은 복잡정도가 낮고 또한 적당한 값의 가중치를 가지게 되므로 이정표를 이용한 효율적인 계획 오류 보정 방법을 보장하게 된다.

Abstract An intelligent autonomous robot generates a plan to achieve a goal. A plan is a sequence of robot actions that accomplish a given mission by being successfully executed. However, in the complex and dynamic real world, a robot may encounter unexpected situations and may not execute its planned actions any more. Therefore, an intelligent autonomous robot must prepare an efficient handling process to cope with these situations to successfully complete a given mission. Plan repair with milestone states is an efficient method to cope with the situation. It retains the advantages of other plan repair procedures. This paper proposes a regressive method of formulating milestone states and a method of assigning weighting values on conditions that compose a milestone state. The task to repair a plan may employ the weighting values as its job priority. The regressive method formulates less complex milestone states and leads to the conditions of a milestone state to take pertinent weighting values for an efficient handling procedure to repair a plan with milestone states.

Key Words : Intelligent Autonomous robots, Planning, Actions, Intelligent robots, Plan repair, Regressive methods, Weighting values, Artificial intelligence

1. 서 론

지능형 자율 로봇에 대한 연구는 로봇을 단순 노동 대체 수단에서 벗어나 IT와의 융합으로 로봇과 인간과의 공존에 의한 삶의 질 향상의 수단을 목표로 한다. 2009년부터 2012년 동안 서비스 로봇 시장은 세계적으로 약 98.7억 달러, 개인 서비스 로봇 시장은 약 31억 달러의 시장을 형성할 것으로 전망된다고 한다. 따라서 한국은 높은 성장 잠재력과 세계시장 선점의 효과를 얻기 위하여 로봇 산업을 10대 신 성장 동력 산업으로 선정하였다[1]. 인간들은 양치질하는 것과 같은 일상적이고 매우 간단한 일이라도 자신도 모르게 계획을 수립하게 된다[2]. 본 논

이 연구는 2011학년도 한국외국어대학교 교내학술연구비의 지원에 의하여 이루어진 것임.
*교신저자 : 한현구(hghan@hufs.ac.kr)
접수일 11년 03월 02일 수정일 (1차 11년 04월 05일, 2차 11년 05월 02일, 3차 11년 05월 25일) 게재확정일 11년 06월 09일

문에서는지능형 자율 로봇은 인간과 유사하게 주어진 문제에 대하여 계획을 수립하고 그 계획을 실행 할 수 있으며 주위 환경을 인식할 수 있는 고성능 센스들을 장착하여 이동 및 물체 인식과 운반이 자율적으로 가능한 로봇을 말한다.

지능형 자율 로봇들도 문제 도메인에서 주어진 임무를 달성하기 위하여 사전에 계획을 수립한다. 계획을 수립하는 과정은 로봇이 주어진 문제 도메인에서 목표 달성을 위하여 자신이 실행해야 할 행위들을 순서 있게 나열하는 것이다[3]. 즉 주어진 문제 도메인의 현재 상태인 초기 상태에서 목표 상태로 상태 전이를 일으키는 행위들을 순서적으로 나열하는 것이다. 계획 수립 과정은 로봇이 행위를 실제로 실행하기 전에 일어나며 계획의 성공은 계획의 행위들이 순서적으로 실행되어 목표를 달성하는 것이다[4]. 본 논문에서는 로봇의 행위를 정의하기 위하여 Stanford Research Institute Problem Solver(STRIPS)[5]의 행위 정의 방법을 수정하여 사용하였다.

계획의 모든 행위는 그 행위의 선행조건들이 만족될 때 로봇이 실행한다. 그러나 현실 세계에는 계획이 수립될 때 예상하지 못했던 많은 돌발 상황들이 발생할 수 있고 이로 인하여 로봇이 계획을 더 이상 실행할 수 없는 경우가 발생할 수 있다. 이러한 상태를 오류 상태라고 본 논문에서는 부른다. 오류 상태의 발생 요인은, 로봇의 이동 모터 고장 등 행위 주체의 고장, 로봇 행위의 부 정확성, 일관 되지 못한 외부 입력자료 등 다양하며 불확실하고 가변적인 실세계에서는 언제라도 오류 상태가 발생할 수 있다. 이러한 오류 상태의 유발 요인들을 모두 예상하여 계획을 수립하는 것은 거의 불가능 하므로 로봇은 성공적인 계획 실행을 위하여 예상하지 못한 상황에 대처할 수 있어야 한다[4,6].

오류 상태가 발생하더라도 목표 달성이 가능하도록 계획을 수정하거나 새로운 계획을 수립하는 작업을 본 논문에서는 계획 보정이라 한다. 계획을 보정하는 방법으로 크게 2가지를 들 수 있다[7]. 첫 번째 방법은 계획 재수립 방법으로서 오류 상태가 발생하면 기존의 계획을 모두 버리고 계획을 다시 수립하는 방법이다[4]. 이 방법은 계획 보정 작업은 간단하나 기존에 수립한 계획을 모두 폐기하고 다시 계획을 수립해야 하는 시간적 손실을 입을 수 있다.

두 번째 방법은 국부적 계획 보정으로서 오류 상태를 오류가 발생하지 않은 예상된 정상적인 상태로 복원시키기 위한 부분 계획을 수립하는 작업이다[4]. 이 방법은 기존의 계획을 버리지 않고 다시 사용하는 방법으로 계획을 재사용한다는 장점이 있다. 그러나 예상되는 모든 정상적인 상태를 로봇이 항상 저장하고 있어야 한다.

이정표 상태를 이용한 계획 보정은 위의 두 가지 방법의 장점을 모아서 제안된 방법이다[7]. 국부적 계획 보정 방법에서와 같이 이 방법은 예상된 상태를 모두 저장하는 것이 아니고 예상된 상태들 중에서 적당한 수의 상태를 선택하여 이정표 상태로 저장하는 방법이다. 계획 보정을 위하여 오류 상태로부터 가까운 이정표 상태로의 부분계획을 수립하게 된다. 그러므로 이 방법은 오류 상태를 처리하기 위하여 기존 계획의 일부분만 재수립하면 되고 또한 전체 예상된 정상적인 상태의 저장할 필요가 없다. 본 논문에서는 이정표 상태를 역방향으로 생성하는 방법인 후진 상태 생성방법을 제안하고 기존의 순방향 생성 방법과 그 효율성을 비교 분석한다. 또한 효과적인 오류 보정을 위하여 이정표 상태를 구성하는 조건들에 가중치를 부여하는 방법을 제안한다.

2. 관련 연구

국부적 계획 보정 방법은 이론적으로 계획 재수립 방법보다 많이 효과적인 것은 아니라고 알려져 있다[9]. 그러나 많은 저자들은 실제적인 면에서 계획의 대부분이 유효한 경우에는 국부적 계획 보정 방법이 효과적일 수 있다고 한다[10]. 관련된 연구로서 전진 이정표 상태 생성 방법은 3장에서 후진 생성 방법과 같이 소개하기로 한다.

2.1 계획 재수립

계획 재수립 방법은 오류 상태를 정상 상태로 되돌리는 것이 아니고 기존의 계획을 모두 버리고 오류 상태를 새로운 초기 상태로 하여 기존의 목표 상태로의 계획을 다시 수립하는 것이다. 따라서 오류 상태를 보정하기 위한 다른 기능이 필요하지 않으며 시스템이 비교적 간단하다[7]. 그러나 대체로 계획을 수립하는 과정자체가 시간과 저장 공간을 많이 필요로 하는 작업이므로 효과적인 계획 보정 방법이 아니다[7]. 또한 계획 실행 초기에 오류 상태가 발생하면 기존 계획의 대부분을 버리고 주어진 목표를 달성하기 위한 매우 긴 계획을 다시 수립해야 하므로 막대한 시간적, 공간적인 비용을 지불해야 한다.

2.2 국부적 계획 보정

일반적으로 국부적 계획 보정 방법은 오류 상태를 오류 상태가 발생하지 않았을 때 생성될 예상된 정상 상태로 복원하는 부분계획을 수립하는 것이다. 대부분의 국부적 계획 보정 방법은 오류상태 발생으로 인하여 현재 실행 불가능한 행위의 선행조건들 만을 위한 국부적인 부

분 계획을 수립한다. 그러나 이렇게 할 경우 선행조건들만을 만족시키기 위하여 새로이 수립된 부분 계획이 현재 행위 이후의 행위들의 실행에 필요한 조건들, 목표 달성을 위하여 꼭 필요한 조건들 등 중요한 조건들을 부정할 수 있다. 그러므로 오류 상태 보정을 위하여 수립된 부분 계획이 이러한 조건들을 부정하게 될 것인지를 검사해야 한다. 만약 부정되어지는 조건들이 있다면 이들을 위한 부분 계획도 수립하여야 하므로 예상된 모든 상태들을 저장하고 있어야 한다.

예상되는 모든 정상 상태를 저장하지 않고 로봇은 최초 행위부터 모든 행위를 실행할 때 마다 마지막 실행된 행위에 의하여 예상되는 상태만을 모의적으로 명확하게 생성하고 저장할 수 있다. 이럴 경우, 오류 상태가 발생하면 마지막으로 저장된 상태로 되돌아가는 부분 계획을 수립하면 된다. 그러나 모든 행위에 대한 예상된 상태를 실시간으로 명확하게 예측하여 생성하는 것은 시간적 제약을 받을 수 있다.

2.3 이정표 상태 계획 보정

수립된 계획을 모의적으로 실행시켜 예상된 정상 상태들을 생성하고 그 중에서 적당한 수의 상태들을 선택하여 이정표 상태로 정한다. 이정표 상태 계획 보정 방법은 오류 상태가 발생하면 먼저 목표 상태 방향으로 오류 상태와 가장 가까운 이정표 상태를 선택한다. 그리고 현재 실행 불가능한 행위부터 선택한 이정표 상태를 생성하는 행위까지를 기존의 계획에서 제거한다. 마지막으로 오류 상태를 초기 상태로 하고 선택한 이정표 상태를 목표 상태로 하는 부분 계획을 수립하고 제거한 부분에 삽입한다[7]. 이 방법은 계획 재수립 방법과 같이 모든 계획을 버리지 않으며, 또한 국부적 계획 보정 방법과 같이 예상한 정상 상태들을 저장하지 않기 때문에 효과적인 방법이다. 그러나 이정표 상태를 생성하는 방법에 따라서 이 방법도 효과적이지 못할 경우가 있으며 이는 3.1절에서 설명한다.

또한 본 방법은 경로 탐색 문제 도메인에서는 효과적인 계획 수정을 보장할 수 없는 경우가 있을 수 있다. 예를 들어 예기치 못한 사정으로 목적 경로에서 벗어나 다른 곳에 잠깐 들러야 할 경우 그 곳에서 바로 다음 이정표 상태까지 의 거리가 예기치 못한 사정을 인지한 곳에서 동일한 이정표 상태까지의 거리보다 짧을 수 있다. 이러한 상황을 serendipity 상황이라고 하며 경로 탐색 문제가 아닌 도메인에서도 발생할 수 있으나 발생 여부를 파악하기 위하여 목표 상태에서부터 현재 상태까지 모두 비교해야 한다.

3. 이정표 상태 생성 방법

본 논문에서는 이정표 상태 생성 방법을 설명하기 위하여 로봇의 행위 실행을 동적 환경에서 개념적으로 모델링하였다. 문제 도메인에서 행위 실행에 의한 상태전이 T는 계획의 4개 요소(S, A, C, τ)로 정의한다[11]. 즉

T = (S, A, C, τ) 로서,

- S = $\{s_1, s_2, \ldots, s_n\}$,
- A = $\{a_1, a_2, \ldots, a_n\}$,
- C = $\{c_1, c_2, \ldots, c_n\}$,
- n: 임의의 정수, s: 상태, a: 행위, c: 조건,
- $\tau : S \times A \times C \rightarrow S$,
  $\tau(s_i, a_i, \text{precond}(a_i)) \rightarrow s_{i+1}$.

S는 문제 도메인의 상태들의 집합이고 초기 상태와 목표 상태를 포함한다. τ는 상태 전이함수로서 $\tau(s_i, a_i, \text{precond}(a_i))$로 표현한다. $\text{precond}(a_i)$가 s_i에서 만족될 때 a_i를 s_i에서 실행하여 $\text{postcond}(a_i)$와 $\text{postcond}^-(a_i)$를 생성하고 상태전이를 일으켜 s_i의 다음 상태 s_{i+1}을 생성한다. A는 로봇이 상태공간에서 취할 수 있는 모든 행위들의 집합이다. C는 상태공간에서 상태 들을 구성하는 조건들의 집합이다. 즉 상태 s_i는 임의의 조건 c_i들의 결합으로 표현된다. C는 행위 a의 선행조건들의 집합인 precond(a), 긍정적 후행조건들의 집합인 postcond(a), 부정적 후행조건인 $\text{postcond}^-(a)$, 그리고 a의 영향을 받지 않는 조건들의 집합인 stillcond(a)로 구성된다. precond(a)는 행위 a가 실행되기 위하여 필수적인 조건들이며 postcond(a)는 행위 a에 의하여 새로이 생성되거나 만족되어지는 조건들을 의미하고 $\text{postcond}^-(a)$는 행위 a에 의하여 삭제되거나 부정되는 조건들을 의미한다. 그리고 stillcond(a)는 행위 a가 실행되어지는 현 상태에 있는 조건들로서 행위 a의 실행과는 무관한 조건들이다. 따라서 어떤 행위 a의 precond, postcond, 그리고 postcond^-는 문제 도메인에서 정의되지만 stillcond는 행위 a가 실행되는 현재 상태에 따라 다르므로 정의되지 않는다. 이정표 상태의 전진 방법과 후진 방법으로 생성되는 상태들의 수는 같으며 생성된 모든 상태에서 적당한 수의 상태를 선택하여 이정표 상태로 정한다.

3.1 이정표 상태 전진 생성 방법

이정표 상태를 순방향으로 생성하기 위한 전진 함수는 상태 전이함수 τ를 $\tau^f : S \times A \times C \rightarrow F$로 나타내고 τ^f함수에 의하여 생성되는 상태들의 집합을 F라고 할 때 F는 S의 부분 집합이다. $\tau^f(f_i, a_i, \text{precond}(a_i)) \rightarrow f_{i+1}$에서 f_i와

f_{i+1}는 F의 원소이며 f_1은 초기 상태이고 계획의 길이(계획을 이루는 행위들의 개수)가 n이면 f_{n+1}은 τ^f에 의하여 생성된 목표 상태이다. 여기서 f_1은 초기 상태 s_1과 동일하나 f_{n+1}은 목표상태 s_{n+1}과 동일하지 않으며 그 이유는 바로 다음에서 설명된다. 상태 f_i는 행위 a_i의 실행 결과인 postcond(a_i)와 postcond$^-$(a_i), 그리고 stillcond(a_i)로 형성되며 형성된 상태에서 서로 상충되는 조건들은 배제한다.

어떤 행위가 실행되어질 때 그 행위의 stillcond에는 나중 행위들의 선행조건인 것과 그 행위들과는 아무런 상관이 없는 조건들도 있을 수 있다. 또한 stillcond에는 현재 상태의 다음에 있는 이정표 상태에 이르기 까지 변하지 않는 조건들 즉 필요 없는 조건들도 있을 수 있으며 목표 상태에 이를 때까지도 이러한 현상이 지속 될 수도 있다. 전진 방법은 순방향으로 상태들을 생성하므로 이러한 조건들이 자연히 존재하게 된다. 따라서 일반적으로 f_{n+1}과 목표 상태 s_{n+1}이 동일하지 않다. 각 행위들에 의하여 생성되는 조건들도 이러한 조건들이 될 수 있으므로 계획 실행 후반부로 갈수록 이러한 조건들이 많이 누적될 것이다. 따라서 오류 상태 보정을 위하여 선택된 이정표 상태에도 필요하지 않은 조건들이 있을 수 있고 선택된 이정표 상태로의 부분계획은 이러한 불필요한 조건들도 만족 시켜야 하므로 효과적인 오류 상태 보정을 기대할 수 없을 것이다. 기존의 이정표를 이용한 계획 수정에서는 이정표를 전진방법으로 생성한다[7]. 그러므로 이정표를 이용하기 때문에 계획재수립 방법과 국부적 계획수정 방법보다는 2장에서 언급한 장점들이 있으나 생성한 이정표에 불필요한 조건들의 존재로 효과적인 계획수정이 이루어지지 않는다. 이와 관련된 간단한 예는 3.4절에서 볼 수 있다.

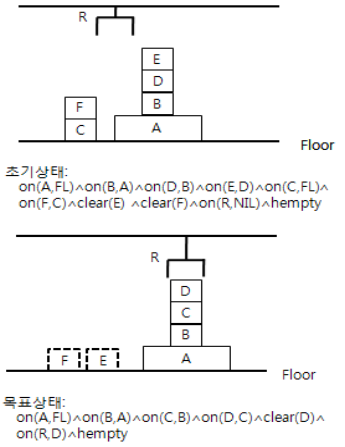
3.2 이정표 상태 후진 생성 방법

앞 절에서 설명한 전진 방법에 의하여 생성된 이정표 상태에는 불필요한 조건들을 포함하게 되어 오류 상태 보정이 효과적으로 일어 날 수 없는 단점이 있었다. 이러한 단점을 보완하기 위하여 본 절에서는 이정표 상태 후진 생성 방법을 제안한다. 이정표 상태를 후진으로 생성하기 위한 후진 함수는 상태 전이함수 τ를 $\tau^b : S \times A \times C \rightarrow B$로 나타낸다. 후진 함수 τ^b에 의하여 생성되는 상태들의 집합을 B라고 할 때 B는 S의 부분 집합이다. $\tau^b(b_i, a_{i-1}, \text{precond}(a_{i-1})) \rightarrow b_{i-1}$에서 상태 b_i와 b_{i-1}는 B의 원소이다. 이때 3.1절에서와 같이 계획의 길이를 n으로 할 경우, b_1은 τ^b에 의하여 생성된 초기 상태이고 b_{n+1}은 문제 도메인에서 제시된 목표 상태이다. 여기서 b_{n+1}은 목표 상태 s_{n+1}과 동일하나 b_1은 초기 상태 s_1과 동일하지 않을 수 있으며 그 이유는 바로 다음에서 설명된다. $\tau^b(b_i, a_{i-1}, \text{precond}(a_{i-1})) \rightarrow b_{i-1}$는 다음과 같이 정의 한다.

$$\tau^b(b_i, a_{i-1}, \text{precond}(a_{i-1})) \rightarrow b_{i-1},$$
$$b_{i-1}=\text{precond}(a_{i-1}) \wedge$$
$$(\forall c_i \in b_i, c_i \notin (\text{postcond}(a_{i-1}) \vee \text{postcond}^-(a_{i-1}))),$$
$$b_{i-1}=b_{i-1} - (\forall c_i \in b_{i-1}, \ni((c_i \in b_{i-1}) \wedge (\neg c_i \in b_{i-1}))).$$

상태 b_{i-1}에서 a_{i-1}이 실행될 것이므로 상태 b_{i-1}에는 a_{i-1}의 선행 조건인 precond(a_{i-1})가 필수적으로 존재해야 한다. 그리고 b_i에 있는 a_{i-1}의 후행 조건들은 b_{i-1}에서 제외하며 나머지 b_i의 모든 조건들을 포함한다. 후행 조건들이 제거되는 이유는 b_{i-1}에서 a_{i-1}의 실행으로 후행 조건들이 생성되어 b_i에 포함되기 때문이다. 따라서 앞에서 언급한 대로 b_1과 s_1이 동일하지 않는 경우가 발생할 수 있다. 또한 전진 방법에서와 마찬가지로 상태 b_i에서 서로 상충되는 조건들은 배제한다.

3.3 Blocks world 문제 도메인



[그림 1] 문제 도메인
[Fig. 1] Problem Domain

본 논문에서는 이정표 상태 생성 방법을 설명하기 위하여 그림 1의 blocks world를 사용하였다. 그림 1과 같은 도메인은 부두에서 크레인을 이용한 컨테이너 하역 작업, 창고 등에서의 상품 적재 작업, 위험한 곳에서의 특정 물체 탈취 작업 등 많은 분야에 적용될 수 있다. 표 1에는 로봇 행위의 정의와 그림 1의 초기 상태와 목표 상태로부터 수립된 계획이 있다. 그림 1의 문제 도메인에는 로봇은 항상 1개 이며 몇 가지 제약 조건을 가지고 있다. 예를 들어 상자 A 외 다른 상자들 위에는 1개 이상의 상자를 둘 수 있는 공간이 없고 Floor 에는 항상 충분한 공간이 있으며 R이 어떤 상자를 집거나 다른 상자 위에 쌓으

려면 대상 상자로 이동 하여야 하는 등이다. 또한 그림 1의 목표 상태에서 상자 E와 F가 점선으로 표시된 것은 상자 E와 F는 목표 상태와 무관함을 나타낸다. 즉 문제 도메인의 목표는 상자 A위에 상자 B, C, D를 차례로 쌓는 것이다.

[표 1] 행위 정의와 수립된 계획
[Table 1] Action definitions and a plan

행위 정의
Pick(X,Y): Pick X from Y -precond: on(X,Y)∧clear(X)∧on(R,X)∧hempty -postcond: hold(X)∧clear(Y) -postcond⁻: ¬on(X,Y)∧¬clear(X)∧¬on(R,X)∧¬hempty
Put(X,Y): Put X on Y -precond: clear(Y)∧on(R,Y)∧hold(X) -postcond: on(X,Y)∧clear(X)∧on(R,X)∧hempty -postcond⁻: ¬hold(X)∧¬clear(Y)∧¬on(R,Y)
Goto(X): Go to X -precond: clear(X)∧hempty -postcond: on(R,X) -postcond⁻: NIL
Moveto(X, Y): Moves X to Y -precond: clear(Y)∧hold(X) -postcond: on(R,Y) -postcond⁻: NIL
수립된 계획
1.Goto(E), 2.Pick(E,D) 3.Moveto(E,FL), 4.Put(E,FL), 5.Goto(D), 6.Pick(D,B), 7.Moveto(D,FL), 8.Put(D,FL), 9.Goto(F), 10.Pick(F,C), 11.Moveto(F,FL), 12.Put(F,FL), 13.Goto(C), 14.Pick(C,FL) 15.Moveto(C,B), 16.Put(C,B), 17.Goto(D), 18.Pick(D,FL), 19.Moveto(D,C), 20.Put(D,C)

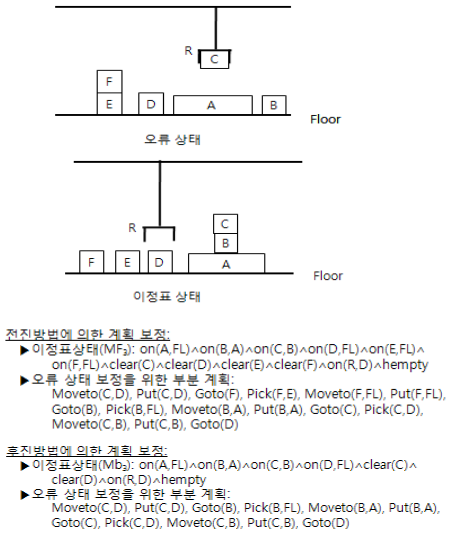
본 논문에서는 초기 상태부터 시작하여 매 6번째의 상태를 이정표 상태로 정하였다. 목표 상태는 항상 최종 이정표 상태이므로 총 4개의 이정표 상태가 선택 되었다. 표 2에는 표 3-1과 표 3-2의 전진 방법과 후진 방법을 사용하여 생성된 예상되는 모든 정상 상태들에서 선택된 4개의 이정표 상태들이 표현되어 있다. 이정표 상태들의 이름을 전진 방법에 의한 것은 Mf로 그리고 후진 방법에 의한 것은 Mb로 각 각 정하였다.

[표 2] 이정표 상태
[Table 2] Milestone States

상태 번호	전진방법에 의한 이정표 상태
f_6 (Mf_1)	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(E,FL)∧on(F,C)∧clear(D)∧clear(E)∧clear(F)∧on(R,D)∧hempty
f_{12} (Mf_2)	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(E,FL)∧clear(B)∧clear(C)∧clear(D)∧clear(E)∧on(R,FL)∧hold(F)
f_{18}	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,FL)∧on(E,FL)∧on(F,FL)∧clear(C)∧clear(D)∧clear(E)∧clear(F)∧on(R,D)∧hempty
f_{21} (Mf_4)	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,C)∧on(E,FL)∧on(F,FL)∧clear(D)∧clear(E)∧clear(F)∧on(R,D)∧hempty
상태 번호	후진 방법에 의한 이정표 상태
b_6 (Mb_1)	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(F,C)∧clear(D)∧clear(F)∧on(R,D)∧hempty
b_{12} (Mb_2)	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧clear(B)∧clear(C)∧clear(D)∧on(R,FL)∧hold(F)
b_{18} (Mb_3)	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,FL)∧clear(C)∧clear(D)∧on(R,D)∧hempty
b_{21} (Mb_4)	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,C)∧clear(D)∧on(R,D)∧hempty

3.4 오류 상태 보정의 예

그림 2의 오류 상태는 로봇이 행위 16번을 실행하려고 할 때 발생한 상태이다. R은 16번 행위를 실행할 때 필요한 선행조건 중 하나인 on(R,B)가 만족 되어져 있지 않으므로 더 이상 계획을 실행할 수 없다. 이 경우 오류 보정을 위하여 현재 상태에서 가장 가까운 이정표 상태인 Mf_3 혹은 Mb_3을 선택하고 오류 상태에서 선택한 이정표 상태로의 부분계획을 수립할 것이다. 여기서 유의해야할 것은 $\tau^f(f_i, a_i, \text{precond}(a_i)) \rightarrow f_{i+1}$의 a_i와 $\tau^b(b_{i+1}, a_i, \text{precond}(a_i)) \rightarrow b_i$에서의 a_i는 같은 행위이다. 그림 2의 오류 보정을 위하여 수립된 각 각의 부분 계획을 보면 후진 방법으로 생성된 이정표 상태에는 필요 없는 상태를 포함하고 있지 않음으로 상대적으로 간단하다. 일반적으로 후진 함수에 의하여 생성된 이정표 상태에는 전진 함수에 의하여 생성된 이정표 상태보다 작은 수의 조건들을 가지게 되며 따라서 오류 보정도 간단해진다.



[그림 2] 오류 상태 보정의 예
[Fig. 2] An example of error recovery

3.5 이정표 상태의 조건들에 대한 가중치

이정표 상태들을 구성하는 각 조건들에 대한 가중치를 부여하는 이유는 오류 상태를 보정할 때 보정해야 할 조건들의 우선순위를 정하기 위함이다. 이와 같이 조건들의 우선순위가 정해지면 오류 상태를 보정하기 위하여 만족시켜야 할 조건들의 순위가 결정되므로 효과적으로 오류 상태를 보정할 수 있다.

가중치를 부여하는 방법의 원칙은 조건들이 이정표 상태에 연속해서 나타나는 정도에 따라 가중치를 높여주는 것이다. 연속적으로 나타나는 조건들은 행위들의 실행에 직접적으로 필요한 선행조건은 아닐지라도 그 행위들이 실행될 때 만족되어져 있어야 하는 조건들이 대부분이다. 즉 현재 이정표 상태를 구성하는 어떤 조건이 바로 전 이정표 상태에 있으면 그 조건의 가중치에 2를 증가 시키며 이 조건이 로봇에 관련된 조건이면 1을 증가 시킨다. 그 외의 새로 나타나는 조건들은 가중치를 0으로 한다. 이때 로봇에 관련된 조건에 가중치 1을 부여하는 이유는 오류 상태 보정을 위하여 로봇이 조건들을 만족시켜야 하는 주체이므로 다른 조건을 먼저 만족시키고 자신과 관련된 조건은 그 이후에 만족시키도록 하기 위함이다. 본 논문에서는 전진 방법과 후진 방법으로 생성된 이정표 상태에 같은 방법으로 가중치를 부여 하였다. 다음 장의 표 4에서 가중치가 부여된 상태를 볼 수 있다.

4. 이정표 상태 생성 방법의 비교

4.1 이정표 상태의 복잡 정도 비교

3장 표 1의 20개의 행위들에 의하여 생성되는 전체 상태는 표 3-1과 표 3-2와 같이 모두 21개이다. 그림 3은 이러한 21개 상태들을 표현 하는데 필요한 조건들의 수를 바탕으로 그린 그래프이다. 가로축은 21개 상태들의 일련번호를 그리고 세로축은 각 상태를 구성하는 조건들의 수를 나타낸다. 그림 3에서도 알 수 있듯이 전진 방법으로 생성된 상태들은 후진 방법으로 생성된 상태들 보다 항상 많은 조건으로 구성되어 있다. 상태 21개 각 각을 구성하는 모든 조건들 수의 합은 전진 방법이 225개, 후진 방법이 167개 이고 평균적으로 3개의 차이를 보인다. 또한 그림 3에서 목표 상태로 갈수록 상태를 구성하는 조건들 수의 차이가 커짐을 알 수 있다. 이는 로봇이 목표 상태에 가까이 갈수록 오류 상태를 점 점 더 빨리 보정할 수 있게 됨을 의미 하므로 계획의 길이가 긴 복잡한 문제일수록 후진 방법이 효과적임을 알 수 있다. 이와 같이 blocks world와 같은 간단한 문제에서도 후진 방법에 의하여 생성된 상태들이 전진 방법으로 생성된 상태들 보다 복잡 정도가 많이 낮음을 알 수 있다.

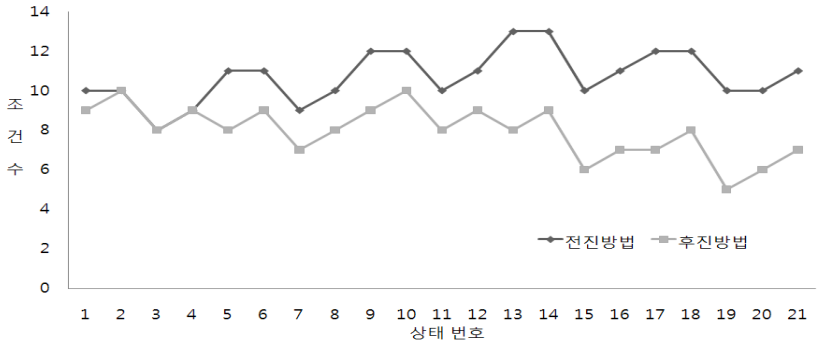
3.4에서와 같이 이정표 상태의 수는 목표 상태를 포함하여 총 4개이다. 계획의 길이가 길면 그에 따라 이정표

[표 3-1] 전진 방법에 의하여 생성된 상태들
[Table 3-1] States generated by the progression

상태 번호	조건 개수	전진 방법에 의하여 생성된 상태
f_1	10	on(A,FL)∧on(B,A)∧on(D,B)∧on(E,D)∧on(C,FL)∧on(F,C)∧clear(E)∧clear(F)∧on(R,NIL)∧hempty
f_2	10	on(A,FL)∧on(B,A)∧on(D,B)∧on(E,D)∧on(C,FL)∧on(F,C)∧clear(E)∧clear(F)∧on(R,E)∧hempty
f_3	8	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(F,C)∧clear(D) ∧clear(F)∧hold(E)
f_4	9	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(F,C)∧clear(D)∧clear(F)∧on(R,FL)∧hold(E)
f_5	11	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(E,FL)∧on(F,C)∧clear(D)∧clear(E)∧clear(F)∧on(R,E)∧hempty
f_6	11	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(E,FL)∧on(F,C)∧clear(D)∧clear(E)∧clear(F)∧on(R,D)∧hempty
f_7	9	on(A,FL)∧on(B,A)∧on(C,FL)∧on(E,FL)∧on(F,C)∧clear(B)∧clear(E)∧clear(F)∧hold(D)
f_8	10	on(A,FL)∧on(B,A)∧on(C,FL)∧on(E,FL)∧on(F,C)∧clear(B)∧clear(E)∧clear(F)∧on(R,FL)∧hold(D)
f_9	12	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(E,FL)∧on(F,C)∧clear(B)∧clear(D)∧clear(E)∧clear(F)∧on(R,D)∧hempty
f_{10}	12	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(E,FL)∧on(F,C)∧clear(B)∧clear(D)∧clear(E)∧clear(F)∧on(R,F)∧hempty
f_{11}	10	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(E,FL)∧clear(B)∧clear(C)∧clear(D)∧clear(E)∧hold(F)
f_{12}	11	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(E,FL)∧clear(B)∧clear(C)∧clear(D)∧clear(E)∧on(R,FL)∧hold(F)
f_{13}	13	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(E,FL)∧on(F,FL)∧clear(B)∧clear(C)∧clear(D)∧clear(E)∧clear(F)∧on(R,F)∧hempty
f_{14}	13	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(E,FL)∧on(F,FL)∧clear(B)∧clear(C)∧clear(D)∧clear(E)∧clear(F)∧on(R,C)∧hempty
f_{15}	10	on(A,FL)∧on(B,A)∧on(D,FL)∧on(E,FL)∧on(F,FL)∧clear(B)∧clear(D)∧clear(E)∧clear(F)∧hold(C)
f_{16}	11	on(A,FL)∧on(B,A)∧on(D,FL)∧on(E,FL)∧on(F,FL)∧clear(B)∧clear(D)∧clear(E)∧clear(F)∧on(R,B)∧hold(C)
f_{17}	12	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,FL)∧on(E,FL)∧on(F,FL)∧clear(C)∧clear(D)∧clear(E)∧clear(F)∧on(R,C)∧hempty
f_{18}	12	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,FL)∧on(E,FL)∧on(F,FL)∧clear(C)∧clear(D)∧clear(E)∧clear(F)∧on(R,D)∧hempty
f_{19}	10	on(A,FL)∧on(B,A)∧on(C,B) ∧on(E,FL)∧on(F,FL) ∧clear(C)∧clear(E)∧clear(F)∧on(R,FL)∧hold(D)
f_{20}	10	on(A,FL)∧on(B,A)∧on(C,B) ∧on(E,FL)∧on(F,FL) ∧clear(C)∧clear(E)∧clear(F)∧on(R,C)∧hold(D)
f_{21}	11	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,C)∧on(E,FL)∧on(F,FL)∧clear(D)∧clear(E)∧clear(F)∧on(R,D)∧hempty

[표 3-2] 후진 방법에 의하여 생성된 상태들
[Table 3-2] States generated by the regression

상태 번호	조건 개수	후진 방법에 의하여 생성된 상태
b_1	9	on(A,FL)∧on(B,A)∧on(D,B)∧on(E,D)∧on(C,FL)∧on(F,C)∧clear(E)∧clear(F)∧hempty
b_2	10	on(A,FL)∧on(B,A)∧on(D,B)∧on(E,D)∧on(C,FL)∧on(F,C)∧clear(E)∧clear(F)∧on(R,E)∧hempty
b_3	8	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(F,C)∧clear(D)∧clear(F)∧hold(E)
b_4	9	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(F,C)∧clear(D)∧clear(F)∧on(R,FL)∧hold(E)
b_5	8	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(F,C)∧clear(D)∧clear(F)∧hempty
b_6	9	on(A,FL)∧on(B,A)∧on(D,B)∧on(C,FL)∧on(F,C)∧clear(D)∧clear(F)∧on(R,D)∧hempty
b_7	7	on(A,FL)∧on(B,A)∧on(C,FL)∧on(F,C)∧clear(B)∧clear(F)∧hold(D)
b_8	8	on(A,FL)∧on(B,A)∧on(C,FL)∧on(F,C)∧clear(B)∧clear(F)∧on(R,FL)∧hold(D)
b_9	9	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(F,C)∧clear(B)∧clear(D)∧clear(F)∧hempty
b_{10}	10	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧on(F,C)∧clear(B)∧clear(D)∧clear(F)∧on(R,F)∧hempty
b_{11}	8	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧clear(B)∧clear(C)∧clear(D)∧hold(F)
b_{12}	9	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧clear(B)∧clear(C)∧clear(D)∧on(R,FL)∧hold(F)
b_{13}	8	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧clear(B)∧clear(C)∧clear(D)∧hempty
b_{14}	9	on(A,FL)∧on(B,A)∧on(C,FL)∧on(D,FL)∧clear(B)∧clear(C)∧clear(D)∧on(R,C)∧hempty
b_{15}	6	on(A,FL)∧on(B,A)∧on(D,FL)∧clear(B)∧clear(D)∧hold(C)
b_{16}	7	on(A,FL)∧on(B,A)∧on(D,FL)∧clear(B)∧clear(D)∧on(R,B)∧hold(C)
b_{17}	7	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,FL)∧clear(C)∧clear(D)∧hempty
b_{18}	8	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,FL)∧clear(C)∧clear(D)∧on(R,D)∧hempty
b_{19}	5	on(A,FL)∧on(B,A)∧on(C,B) ∧clear(C)∧hold(D)
b_{20}	6	on(A,FL)∧on(B,A)∧on(C,B)∧clear(C)∧on(R,C)∧hold(D)
b_{21}	7	on(A,FL)∧on(B,A)∧on(C,B)∧on(D,C)∧clear(D)∧on(R,D)∧hempty



[그림 3] 상태 복잡도 비교 그래프
[Fig. 3] Comparison graph of state complexity

상태의 수도 많아질 것이며 오류가 발생했을 때 오류 보정을 위하여 선택해야 할 이정표 상태를 쉽게 찾아 낼 수 있어야 한다. 찾고자 하는 이정표의 인덱스를 i라 하고 오류 상태를 직면한 행위의 번호는 k, p를 이정표 상태를 선택할 때 임의로 정한 상수라고 하면 i는 min(k < i × p + 1)에 의하여 구할 수 있다. 예를 들면 3.3절에서 매 6번째 상태를 이정표 상태로 정하였으므로 p는 6이고 현재 오류 상태로 인하여 실행이 불가능한 행위가 a_{16}이므로 찾아야 할 이정표 상태의 인덱스는 16 < i × 6 + 1 을 만족하는 i 중 가장 작은 정수인 3이다.

4.2 오류 상태 보정의 효율성 비교

본 논문에서는 인간과 비슷한 지능형 자율 로봇을 연구 대상으로 하고 있으나 그러한 로봇을 접하기가 어려워 현대중공업의 HA010L 기반중량 10Kgf 산업용 로봇의 카탈로그와 판매 업체 직원의 도움으로 오류 상태 보정의 효율성을 비교 하였다[12]. 로봇 행동의 실행 속도는 작업 환경 즉 물체의 크기, 무게, 로봇 각 관절의 회전각도, 로봇 암의 이동 거리 등에 따라 상이할 수 있으므로 정확하게 계산하기는 어렵다. 상자 A를 제외한 물체의 무게는 모두 5Kg, 모양은 1모서리가 10cm인 정육면

[표 4] 오류 보정을 위한 부분 계획의 효율성 비교
[Table 4] Efficiency comparison of partial plans for error recovery

행위	오류 precond.	오류 내용	부분계획				
			전진 방법			후진 방법	
			부분계획	생성시간(sec)	실행시간(sec)	부분계획	생성시간(sec)	실행시간(sec)
4.Put(E,FL)	on(R,FL)	on(R,A)	Moveto(E,FL),Put(E,FL),Goto(D)	0.0499	6.0	Put(E,A),Goto(D)	0.0333	4.0
8.Put(D,FL)	hold(D)	on(D,E)	Goto(D),Pick(D,E),Moveto(D,FL),Put(D,FL),Goto(F),Pick(F,C),Moveto(F,FL)	0.1164	13.5	Goto(F),Pick(F,C),Moveto(F,FL)	0.0501	5.5
16.Put(C,B)	on(B,A)	on(B,FL),on(F,E)	Moveto(C,D),Put(C,D),Goto(F),Pick(F,E),Moveto(F,FL),Put(F,FL),Goto(B),Pick(B,FL),Moveto(B,A),Put(B,A),Goto(C),Pick(C,D),Moveto(C,B),Put(C,B),Goto(D)	0.2494	30.0	Moveto(C,D),Put(C,D),Goto(B),Pick(B,FL),Moveto(B,A),Put(B,A),Goto(C),Pick(C,D),Moveto(C,B),Put(C,B),Goto(D)	0.1829	22.0
19.Moveto(D,C)	hold(D)	on(D,FL),on(E,F)	Goto(E),Pick(E,F),Moveto(E,FL),Put(E,FL),Goto(D),Pick(D,FL),Moveto(D,C),Put(D,C)	0.1335	16.0	Goto(D),Pick(D,FL),Moveto(D,C),Put(D,C)	0.0664	8.0

체, 그리고 이동 높이와 이동 거리는 모두 최대 50cm로 하여 각 로봇 행위의 평균속도를 얻었다. Pick(X,Y)은 2초, Put(X,Y)는 2.5초, Goto(X)는 1.5초, 그리고 Moveto(X,Y)는 2초가 소용된다. 부분 계획은 2.8GH dual core Intel I7 860 PC에서 C++로 작성된 planner로 생성하고 생성 시간을 측정하였다. 표 4는 3장 표 1의 수립된 계획의 20개 행위 중 각 이정표 상태들 사이에서 1개씩 모두 4개의 행위를 선택하고 각 행위의 precond. 중 임의로 하나를 만족시키지 못하는 오류 상태를 만들어 이를 보정하기 위하여 생성된 부분 계획과 생성 시간 그리고 부분 계획의 길이를 나타내고 있다. 표 4에서 보듯이 부분 계획 수립 시간은 로봇 행위 실행 시간에 비하여 상대적으로 매우 짧다. 따라서 전진 방법과 후진 방법의 효율성은 생성된 부분 계획의 길이에 따라 많은 차이가 난다. 즉 생성된 부분 계획을 로봇이 실행해야 오류 보정이 이루어지므로 부분 계획을 실행 하는데 소용되는 시간으로 효율성을 비교 하여야 할 것이다. 표 4의 실행시간을 보면 최대 8초의 차이를 보이고 있다. Blocks world와 같은 간단한 도메인이 아닌 복잡한 도메인에서는 그 차이가 더욱 클 것이다. 표 3-1과 표 3-2에서 알 수 있듯이 전진 함수가 생성한 상태들에는 상태를 구성하는 조건들의 수가 후진 함수로 생성된 상태들 보다 항상 많거나 같다. 이정표 상태는 이러한 상태들 중에서 선택되므로 전진 함수에 의하여 선택된 이정표 상태에는 오류 보정을 위하여 만족 시켜야 할 조건들이 많고 따라서 부분 계획의 길이가 길어진다.

4.3 이정표 상태의 가중치 비교

표 5는 전진과 후진 방법에 의하여 생성된 이정표 상태들의 가중치를 보이고 있다. 표의 상태 조건은 4개의 이정표 상태를 구성하는 조건들이다. 각 이정표 상태를 구성하는 조건들은 가중치 값으로 0, 1, 혹은 2를 가지는 조건들이다. 전진 방법에 의하여 생성된 이정표 상태에는 중요하지 않은 조건들이 높은 가중치를 가질 수 있다. 표 5의 강조된 부분에서 보는 바와 같이 전진 방법으로 생성된 이정표에는 상자 E와 F에 관련된 조건들에 높은 가중치가 부여되었다. 반면에 후진 방법으로 생성된 이정표에는 상자 E와 F에 관련된 조건에 낮은 가중치가 부여되거나 아예 부여되지 않았다.

[표 5] 가중치 테이블
[Table 5] Weighting value table

상태 조건	전진 방법				후진 방법			
	Mf_1	Mf_2	Mf_3	Mf_4	Mb_1	Mb_2	Mb_3	Mb_4
on(R,FL)		0				0		
on(R,C)				0				0
on(R,D)	0		0		0		0	
hempty	1		0	1	1		0	1
hold(F)		0				0		
on(A,FL)	2	4	6	8	2	4	6	8
on(B,A)	2	4	6	8	2	4	6	8
clear(B)		0				0		
on(C,FL)	2	4			2	4		
on(C,B)			0	2			0	2
clear(C)		0	2			0	2	
on(D,FL)		0	2			0	2	
on(D,B)	2				2			
on(D,C)				0				0
clear(D)	0	2	4	6	0	2	4	6
on(E,FL)	0	2	4	6				
clear(E)	2	4	6	8				
on(F,FL)			0	2				
on(F,C)	2	4			2			
clear(F)	2		0	2	2			

예를 들어 조건 clear(E)는 4번 행위 Put(E,FL)에 의하여 f_5에 생성되어 마지막 목표 상태 f_{21}까지 계속해서 나타나므로 f_5이후의 모든 이정표 상태에서 높은 가중치를 가지게 될 것이다. 그러나 후진 방법은 τ^b를 목표 상태 b_{21}부터 역으로 적용하여 상태들을 생성하며 clear(E)는 목표 상태에 없고 2번 행위 Pick(E)의 선행조건으로서 b_2에 처음으로 나타난다. 그러므로 b_3부터 b_{21}까지에는 나타나지 않는 조건이므로 가중치가 부여되지 않는다. 이는 3.1절에서 설명한 대로 전진 함수를 적용하여 생성된 이정표 상태에는 계획실행과 무관한 조건들을 포함하게 되고 이들 조건들이 지속적으로 이정표 상태에 나타나기 때문이다. 이럴 경우 가중치가 높은 조건을 우선적으로 만족시키려고 할 것이므로 효과적으로 오류 상태를 보정할 수 없게 된다.

5. 결론

지능형 자율 로봇이 예상하지 못한 상황에 처하여 계획 실행이 불가능할 경우를 해결하는 한 방법인 이정표 상태 계획 보정 방법에서 후진 이정표 상태 생성 방법을 제안하였다. 전진 방법으로 생성된 이정표 상태에는 필요 없는 상태 조건들을 포함할 수 있으나 후진 방법으로 생성된 이정표 상태에는 이러한 조건들이 배제되었다. 따라서 후진 방법은 전진 방법보다 보정해야 할 조건의 수가 적게 되므로 오류 상태 보정을 위한 부분 계획의 길이가 짧게 된다. 긴 부분 계획은, 수립하는 데도 시간을 많이 소요해야 하고 또한 수립한 긴 부분 계획을 실행하는데도 더 많은 시간이 필요하게 된다.

또한 본 논문에서는 각 이정표 상태를 구성하고 있는 조건들에 가중치를 부여하는 방법도 제안 하였다. 오류 상태의 보정 순서를 조건들의 가중치에 의하여 결정할 수 있으므로 효과적으로 오류 상태를 보정할 수 있을 것이다. 전진 방법으로 생성된 이정표 상태들에는 필요 없는 조건들이 높은 가중치를 갖게 될 수 있다. 따라서 효과적인 이정표 상태 계획 보정 방법을 위하여 이정표 상태 후진 생성 방법이 필수적임을 알 수 있다.

경로 탐색을 목적으로 하는 문제 도메인에서는 이정표 상태 계획 보정 방법의 적용은 가능하지만 효율적인 오류 상태 보정을 보장할 수 없다. 이정표 상태 계획 보정 방법을 여러 분야에 적용하기 위해서는 이정표 상태의 적정 개수 결정, 로봇의 수, 이정표 상태들의 유사성 추출, 이정표 상태들 간의 지리적 거리 계산, 오류 상태 보정의 경험적 지식 활용 등, 여러 방면에 대한 연구가 필요할 것이다.

References

[1] Joo, Eunchul, 'Industrial Trends & Development Strategies of IT Fusion - Robots', Business Information Research', http://www.birbook.com

[2] Do, Yongtae et al., 'Artificial Intelligence Conceps and Applications', 3rd Edition, SciTech, Seoul Korea, 2009

[3] Malik Ghallab, Dana Nau, Paolo Traverso. 'Automated Planning Theory and Practice', Morgan Kaufmann Publishers, New York, 2004

[4] Hyungoo Han, Kai Chang, William Day, 'A Comparison of Failure Handling Approaches for Planning Systems - Replanning vs. Recovery', Journal of Applied Intelligence, Vol. 3, Kluwer academic publishers, pp. 275-300, 1993

[5] Richard E. Fikes, Nils J. Nilsson, 'STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving', Artificial Intelligence (2), pp. 189-208, 1971

[6] Jianming Guo, Liang Liu, 'A Study of Improvement of D* Algorithms for Mobile Robot Path Planning in Partial Unknown Environments', Kybernetes Vol. 39, Emerald Group Publishing Limited, Issue 6, pp. 935-945, 2010

[7] Han, Hyungoo, 'Plan Repair with Milestone States', Journal of Information Industrial Engineering, Institute of Information Industrial Engineering Hankuk Univ. of Foreign Studies'. Vol. 13, pp. 205-216, 2009.

[8] Bernhard Nebel, Jana Koehler, 'Plan Reuse versus Plan Generation: A Theoretical and Empirical Analysis,' Artificial Intelligence, 76, pp. 427-454, 1995

[9] Roman van der Krogt, Mathijs de Weerdt, 'Plan Repair as an Extension of Planning', ICAPS, pp. 161-170, 2005

[10] C. A. Broverman and W. B. Croft, 'Reasoning about Exceptions During Plan Execution Monitoring', Proc. Intl. Conf. Artificial Intelligence, Seattle, WA, pp .190-195, 1987

[11] T. Dean and M. Wellman, 'Planning and Control', Morgan Kaufmann, 1991

[12] Hyundai Heavy Industry, T/G Part of the Engine Machinery Dept. http://www.hhi.co.kr/korea/EngineMachinery

한 현 구(Hyungoo Han) **[정회원]**

- 1980년 2월 : 서강대학교 수학과 (이학사)
- 1986년 5월 : Univ. of South Florida 전산학과 (공학석사)
- 1990년 12월 : Auburn University 전산학과 (공학박사)
- 1992년 3월 ~ 현재 : 한국외국어대학교 컴퓨터공학과 교수

<관심분야>
Robot planning, Wireless Networking