트리구조 기반의 개선된 GP 연산자를 이용한 4족 보행로봇의 걸음새 자동생성

방철혁, 현수환, 서기성
서경대학교 전자공학과

Automatic Gait Generation of Quadruped Robot using Improved Genetic Programming Operators based on Tree Structure

Cheulhyuk Pang, Soohwan Hyun, Kisung Seo
Dept. of Electronic Engineering, Seokyeong University

Abstract - 본 논문은 GP(Genetic Programming)을 이용한 4족 보행 로봇의 새로운 걸음새 생성 방식에 대해 소개한다. 4족 보행로봇의 걸음새 생성 문제는 다양한 파라미터를 동시에 최적화해야 하는 매우 어려운 문제이다. 본 논문에서는 GP를 기반으로 관절좌표계에서 로봇의 관절 궤적을 직접 제어하는 방식을 사용한다. 이는 기존의 특정한 형태의 발끝의 자취를 사용하는 방법들에 비해 효율적이며 구조적으로 제한되지 않는 특징을 가진다. 또한, 새로운 트리구조기반의 GP 연산자의 적용을 통해 더 좋은 결과를 얻을수 있었다.

1. 서 론

보행로봇은 바퀴로봇보다 장애지형에 있어 높은 이동성을 가지는 장점을 가지고 있다. 4족 보행로봇의 걸음새 생성 문제는 로봇의 보행 계획을 결정하는 핵심적인 부분으로, 발의 궤적, 스텝 수, 초기자세 등 다수의 파라미터를 동시에 설계해야 하는 복잡한 문제이다[1,2]

걸음새 생성에 분석적인 방법도[3] 시도되고 있으나, 성능 면에서 우수한 결과를 보여주고 있는 진화적인 기법을 사용한 방법이 주로 이루어지고 있다[2-4]. 기존의 진화적 접근법들은 로봇을 위한 파라미터를 제한적으로 고정시키고, 그 파라미터에 대한 최적해를 찾는 방법으로 수행되어지고 있다. 문제점으로는 걸음새에 관련된 수많은 파라미터가 존재하여 이를 모두 파악하기가 어렵고, 일부 주요 파라미터만을 고정시켜 탐색할 경우, 최적의 걸음새 생성이 가능한지 판단하기 힘들다. 일부 파라미터만을 선별하여 사용하는 기존의 접근법의 경우에서도, 최소 10개 이상의 파라미터를 사용하기 때문에 이들의 최적 값을 구하는 것만도 매우 어려운 문제이다.

본 논문에서는 이러한 GA 접근법의 문제점을 극복하기 위해서, GP[5,6]를 이용한 관절 궤적 중심의 걸음새 생성 방법을 제안한다. GP는 해의 탐색공간이 구조적으로 제한되지 않는(open-ended) 설계 특성을 만족시킬 수 있다. 그러므로 GP의 이런 특성을 바탕으로 일부 파라미터에 의존적이지 않는 걸음새의 구현이 가능하다. 기존에 쓰이던 파라미터들 대신에 관절좌표계 상의 관절의 궤적을 구하는 것으로 걸음새를 생성할 수 있는 방법이다.

부가적으로 기존 GP의 탐색 성능을 개선시킬 수 있는 트리구조기반의 진화 연산자가 제안되었다. 구조기반의 진화 연산자는 트리의 깊이와 노드수 정보를 이용해 트리구조 공간상에서 해의 분포를 조절해주는 연산자이다. 4족 보행로봇에 대해서, 제안된 GP기반의 걸음새 궤적 자동생성법을 실험하였으며, 표준 진화연산자와 새로운 구조기반의 진화연산자를 단일 군집과 다중 군집을 조합한 방법에 대한 생성결과를 비교하였다.

Sony 사의 Aibo ERS-7[7]모델에 대해 ODE 기반의 물리적 특성을 포함한 시뮬레이터인 Webots[8]을 이용하여 실험을 수행하였다.

2. 로봇 걸음새의 진화적 생성

2.1 로봇의 걸음새

4족 보행 로봇의 걸음새 생성 문제는 로봇 보행 계획을 결정하는 핵심적인 부분으로서, 발의 궤적, 스텝 수, 초기 자세 등의 수많은 파라미터를 동시에 설계해야 하는 복잡한 문제이다.

기존에 사용된 대표적인 걸음새 생성방법은 다음과 같다. Hornby[2] 는 4족 보행 로봇의 모델로 ERS-110을 대상으로 20 여개의 걸음새 파라미터를 정의하여 직교 좌표 공간에서 스윙과 게인 방식을 통해 주기적인 걸음새를 생성하였다. UNSW 팀은 걸음새 생성을 위해 Powell's Minimization 과 같은 기법을 사용하였다[3]. 17-18 개의 파라미터를 미리 결정된 사각형의 자취에 대해 위의 분석적 기법을 통해 구해내는 방법이다. 또 다른 접근 방법으로 Mercli[4] 의 타원형의 자취를 선택한 방법이 있는데, 이는 10여개의 파라미터를 GA 를 통해 최적화를 시도하였다.

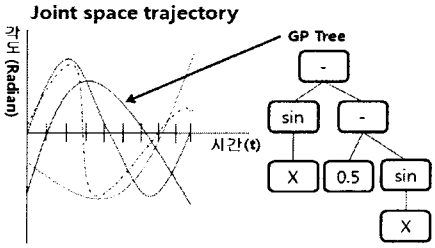
2.2 유전 프로그래밍(GP)

유전 프로그래밍(GP)[5,6] 은 큰 부류에서 GA와 함께 진화 연산(Evolutionary Computation)에 속하지만, GA와는 몇 가지 다른 특징이 있다. 가장 큰 차이점으로는 GA는 일반적으로 비트스트링 혹은 실수형태로 개체를 표현하지만, GP는 트리로 개체를 표현하는 것이 있다. 이때 트리의 노드는 하나의 함수를 나타내며 각 개체는 이러한 함수들의 집합으로 표현된다. 이 함수의 집합은 하나의 컴퓨터 프로그램으로서 나타나게 된다. 그러므로 GA는 개체가 하나의 해의 표현의 구성요소를 표현하고, GP는 해 자체를 생성하는 하나의 프로그램을 표현하게 된다. 이러한 차이 외에도 GA 가 대부분 고정 크기의 염색체를 이용하는데 반해, GP는 가변크기의 염색체를 사용하는 등의 차이가 있다.

2.3 GP 기반의 걸음새 생성 기법

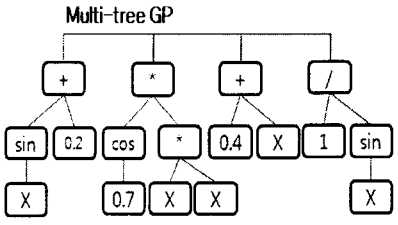
기존의 방법들은 대부분 로봇의 다리에 대한 역기구학을 해석하고, 직교좌표계 상의 발끝의 자취를 제어하는 접근법을 사용하고 있다[3,4]. 이러한 기법들은 먼저 어떤 모양의 발끝 자취를 선택해야 하는지에 대한 어려움이 있고, 자취가 선택되면 해당 자취에 대한 최적화만 이루어지므로 전체적인 최적화와는 거리가 있다.

이러한 기존 방법들의 문제점을 해결하기 위해 본 논문에서는 GP를 이용해 로봇이 가지는 관절을 직접 제어하는 방식을 사용한다. 그림 1과 같이 GP의 트리구조를 이용하여 로봇의 관절 궤적을 형성하고, 그 관절궤적을 따라 로봇을 움직임으로써 걸음새를 생성하게 된다.



<그림 1> 관절의 궤적 그래프 및 GP 트리 표현

걸음새 생성을 위한 대상 모델은 Sony의 Aibo ERS-7[7]이다. 로봇의 각 다리는 3개의 관절로 구성되어있지만 직선 보행을 위해 어깨와 무릎, 두 개의 관절의 움직임만을 생성하였다. 그리고 양 측면의 다리는 동일한 움직임을 시간차를 두고 이용하며 되므로 앞, 뒷면의 다리에 대한 움직임만을 생성하였다. 각 관절의 독립적인 제어를 위해 다중트리 방식의 구성을 이용하여 진화연산을 수행하였다(그림 2).



<그림 2> Multi-tree GP

3. 구조기반의 개선된 GP 연산자

3.1 GP의 구조적 어려움

GP는 트리구조를 이용하여 개체를 표현하며, 구조의 가변성을 이용하여 복잡도가 높은 실용적 문제와 최적화 문제에 많은 응용이 이루어지고 있다[5].

GP 진화연산이 가지는 탐색 공간은 트리의 형태, 즉, 깊이와 노드수 두가지 정보를 이용하여 표현될 수 있다. 트리의 구조적인 문제에 대한 Daida의 연구[9]에서 발생 가능한 트리 구조의 범위와 실제적으로 해로써 작용하는 트리 구조의 분포 범위는 차이가 있으며, 전체적인 범위 내에서 균등하게 나타나지 않음이 발견된 바 있다.

트리 구조로 표현가능한 영역에 대하여 연산자에 의해 조합되는 구조의 분포형태가 일정 지역에 치중되어 나타나기 때문에 알고리즘의 탐색능력을 저하시키는 요인이 된다.

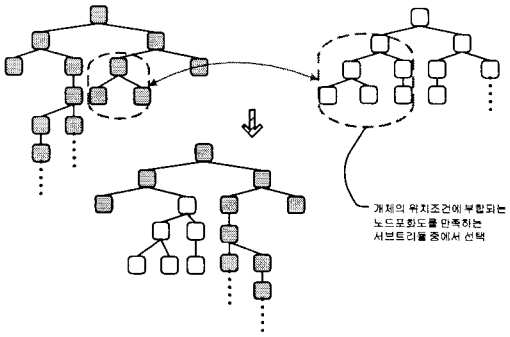
3.2 트리 구조 기반의 진화 연산자

트리 구조 기반의 진화연산자의 주 목적은 교배와 돌연변이 연산을 통해 재조합되는 개체들을 해로써의 작용 확률이 높은 특정영역에 분포되

는 확률을 높일 수 있게 하기 위함이다.

트리가 가지는 깊이와 노드수 정보를 이용하여 트리의 노드포화도를 계산하고, 이 정보를 바탕으로 진화연산을 수행한다. 일정 깊이에 대하여 완전(full)트리의 노드포화도를 1로 하여 노드가 채워진 비율을 노드포화도로 나타내게 된다.

제안된 진화연산자는 구조 공간 내에서 부모개체의 위치를 파악하고, 연산후의 결과에 대해 노드포화도를 제어함으로써 트리의 위치를 원하는 영역에 가깝게 자손개체를 생성하게 된다.



<그림 3> 구조기반 진화연산자

4. 실험 및 결과

4.1 시뮬레이션 환경

시뮬레이션 환경은 Cyberbotics 사의 Webots[8] 을 사용하였다. Webots 은 모바일 로봇에 대한 모델링, 프로그래밍, 그리고 시뮬레이션 기능을 제공하는 모바일 로봇 시뮬레이션 s/w 이다. 주요 기능으로 각종 센서와 액튜에이터에 대한 라이브러리 제공하며, 정확한 physics 시뮬레이션을 위한 ODE(Open Dynamics Engine) 라이브러리 제공, 그리고 실제 모바일 로봇에 컨트롤러를 트랜스퍼 할 수 있는 기능을 가지고 있다.

4.2 적합도 함수 및 GP 파라미터

적합도 판정을 위하여 식(1)을 사용하였다. 단위시간동안 이동한 직선거리 x와 좌/우로 이동한 거리 z를 지표로 사용하였다. 단위시간동안 좌/우로 벗어나지 않고, 직선으로 이동한 거리에 대하여 속도로 환산하고 적합도 지표로 사용하였다.

$$fitness = (0.8 \times x) - (0.4 \times |z|) \quad (1)$$

걸음걸이 생성을 위해 lil-GP[10]를 이용하였다. 기존의 진화연산자와 제안한 진화연산자를 단일군집(single-pop)과 다중군집(multi-pop)으로 수행하였고, 로봇의 이동속도를 기준으로 걸음새를 비교하였다.

진화연산에 사용한 GP의 파라미터 설정은 표 1의 조건으로 수행하였으며, 연산에 사용할 개체수를 일치시키기 위해 다중군집조건에서는 하나의 군집을 20개의 개체로 구성하고 5개의 군집을 형성하였다.

<표 1> GP 파라미터

Number of Generations	100
Population sizes	100 (single-pop) 20 × 5 (multi-pop)
Initialization method	half_and_half
Initialization depths	1-3
Maximum depth	12
Selection method	tournament (size=7)
Corssover rate	0.8
Mutation rate	0.1
Reproduction rate	0.1
Migration method	ring-migration (only multi-pop case)

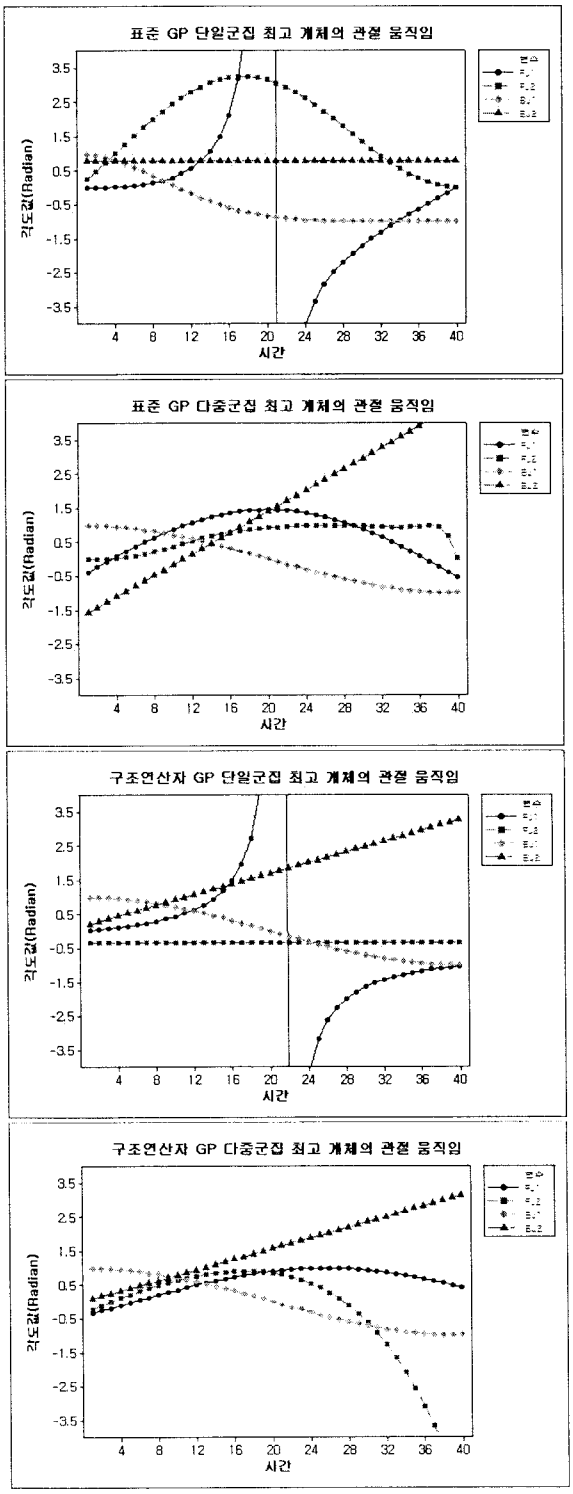
4.3 실험 결과 및 분석

표 2는 실험 결과를 보여주고 있다. 각 조건마다 10회의 실험을 실시하였고, 평균값을 비교하였다. 제안한 연산자를 사용하여 다중군집을 구성하였을 때, 가장 우수한 결과를 가져왔으나 단일군집에서는 오히려 일반연산자가 더 높은 평균 이동 속도를 보였다.

그림 4는 각 조건에서 생성된 관절궤적을 보여주고 있다. 시간은 로봇에 사용된 step을 나타내며, 한 step은 8ms이다. FJ1은 앞다리의 어깨관절, FJ2는 앞다리의 무릎관절, BJ1은 뒷다리의 어깨관절, BJ2는 뒷다리의 무릎관절을 나타낸다.

<표 2> GP 연산 방식에 따른 성능

진화연산조건	적합도	이동속도 (cm/s)
표준연산자 & 단일군집	1.20447	38.54
표준연산자 & 다중군집	1.17977	40.51
구조기반연산자 & 단일군집	0.7621	33.05
구조기반연산자 & 다중군집	1.4829	48.78



<그림 4> GP에 의해 생성된 관절궤적

<그림 5> 표준연산자 단일군집의 걸음새

<그림 6> 표준연산자 다중군집의 걸음새

<그림 7> 구조기반연산자 단일군집의 걸음새

<그림 8> 구조기반연산자 다중군집의 걸음새

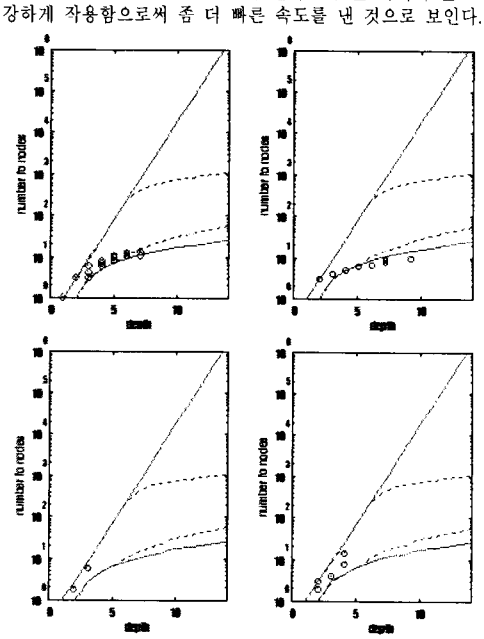
표준 GP의 단일군집에서 생성된 개체의 특징은 관절궤적 그래프에서 앞다리 관절의 움직임이 매우 크게 나타나며(그림 4의 첫 번째), 이러한 큰 움직임 때문에 로봇의 균형이 앞으로 쏠리는 현상을 그림 5의 걸음새 동작에서, 확인할 수 있다.

그리고 비슷한 형태의 그래프를 그리는 구조연산자 GP의 단일군집의 결과는(그림 4의 세 번째) 이와는 조금 다른 양상으로 나타난다. 다른 관절들의 움직임이 매우 적게 때문에 로봇의 균형자체는 그림 7과 같이 매우 안정됨을 보이나, 속도 면에서는 매우 저조하다.

속도가 빠르게 나타난 다중군집의 두 개체는(그림 4에서 두 번째와 네 번째) 상대적으로 속도가 느린 단일군집의 결과들에 비해, 비교적 부드러운 관절 궤적의 움직임을 보여주는 것을 확인할 수 있으며, 이는 그림 6과 그림 8의 걸음새 동작에서도 어느 정도 나타난다.

표준 GP의 다중군집의 결과는 그 변화의 정도가 그래프의 한쪽 방향으로 치우쳐 나타난 것을 확인할 수 있다. 또한, 모든 관절의 움직임이 다양하게 나타나고 있다. 그림 6의 움직임을 볼 때, 이러한 특성에 의해 로봇 안쪽으로의 움직임은 매우 적게 나타나며, 전체적으로 모든 관절을 부드럽게 사용하며 걸어가는 것을 확인할 수가 있다.

구조연산자 GP의 다중군집의 결과는 표준 GP의 다중군집의 결과와 비슷하게 모든 관절을 부드럽게 움직이지만, 앞다리 무릎관절의 움직임이 음수 방향으로 매우 크게 나타나는 것 확인할 수가 있다. 이러한 현상에 의해 그림 8의 7번째 움직임을 보면 다리의 무릎관절이 안쪽으로 많이 꺾여있는 것을 확인할 수 있다. 앞의 표준 GP의 결과보다 앞 다리의 앞으로 미는 힘을 더 강하게 작용함으로써 좀 더 빠른 속도를 낸 것으로 보인다.



<그림 10> 각 조건별 개체 분포도 - 표준연산자 & 단일군집(좌상), 표준연산자 & 다중군집(우상), 구조기반연산자 & 단일군집(좌하), 구조기반연산자 & 다중군집(우하)

부가적으로, 실험한 4 가지 조합에 대한 GP 수행시 트리 개체 해의 깊이와 노드수의 관계 분포를 조사하였다. 해가 수렴된 마지막 세대에서 개체 100개에 대한 분포를 작은 원으로 표시하였다. 참고로, 점선으로 표시된 가운데 부분이 트리의 형태상 깊이와 너비가 균형을 이루며, Daida[9]의 연구에서 알려진 것처럼 해의 발견율이 가장 높은 영역이다.

그림 9의 상단 좌우는 표준 GP 연산자에 의한 결과로서 상당수의 트리 개체가 가운데 영역을 벗어난 것을 알 수 있고, 개체들이 평균적으로 깊이가 긴 트리를 구성하고 있음을 알 수 있다. 이에 비해서, 그림 하단 좌우에 있는 트리구조기반 GP 연산자에 의한 결과는 개체들이 모두 가운데 영역에 존재하며, 트리의 깊이도 짧아서 bloat 없이 효율적인 해를 구성함을 확인할 수 있다.

5. 결 론

로봇산업이 발전함에 따라 로봇의 형태가 다양해지고, 동작의 복잡도가 증가하고 있다. 이에 따라, 로봇의 걸음새 생성 문제도 보다 자동적이고 지능적인 방법이 요구되고 있다. 기존의 시도되고 있는 연구들에서 진화연산을 이용한 방법들이 있으나, 선행적으로 로봇에 필요한 파라미터들을 결정해야 하며, 수많은 파라미터들을 최적화해야 하는 어려움이 존재한다.

본 논문에서는 GP를 기반으로 기존의 로봇 파라미터에 독립적으로 관절 좌표계에서 관절 궤적을 직접 제어하는 방식을 사용하였다. 이는 기존의 특정한 형태의 발끝의 자취를 사용하는 방법들에 비해 효율적이며 구조적으로 제한되지 않는 특징을 가진다. 또한, 새로운 트리구조기반의 GP 연산자의 구현을 통해서 표준 진화연산자에 비해서 개선된 결과를 얻을수 있었다. 결과적으로 평균 48.78cm/s의 이동속도를 가지는 4족 보행 걸음새를 생성할 수 있었다.

GP를 이용한 관절궤적의 생성으로서 보행로봇의 걸음새를 제어한 방식은 능동적인 동작 생성이 가능하며, 구조가 다른 로봇에 대해서도 적용이 가능하다. 향후 4족 로봇 뿐 아니라 다양한 형태의 로봇에 응용이 필요하며, 실제 로봇에 대한 실험도 병행되어야 할 것으로 생각된다.

[참 고 문 헌]

[1] M. Wahde, Evolutionary Robotics - The Use of Artificial Evolution in Robotics, A tutorial presented at IROS2004, TR-BBR-2004-001, Tetsuo, KOTOKU, September 28-29, 2004

[2] G. S. Hornby, S. Takamura, T. Yamamoto, M. Fujita, "Autonomous Evolution of Dynamic Gaits with Two Quadruped Robots," IEEE Trans. Robotics, Vol. 21, No. 3, pp.402-410, 2005

[3] Z. D. Wang, J. Wong, T. Tam, B. Leung, M. S. Kim, J. Brooks, A. Chang, N. V. Huben, *The 2002 rUNSWift Team Report*, 2002

[4] T. Mericli, H. L. Akın, C. Mericli, K. Kaplan, B. Celik, *The Cerbus'05 Team Report*

[5] J. R. Koza, Genetic Programming: On the Programming of Computers by means of Natural Selection, MIT Press, Cambridge, MA, USA, 1992

[6] J. R. Koza, F. H. Bennett, D. Andre, M. A. Keane, *Genetic Programming III, Darwinian Invention and Problem Solving*, Morgan Kaufmann Publishers, USA, 1999.

[7] Sony Corporation, *Model information for ERS-7*, OPEN-R SDK, 2004.

[8] L. Hohl, R. Tellez, O. Michel, A. J. Ijspeert, "Aibo and Webots: Simulation, wireless remote control and controller transfer," Robotics and Autonomous Systems, 54(2006), pp. 472-485, 2006.

[9] J. M. Daida, A. M. Hilss, "What Makes a Problem GP Hard? Validating a Hypothesis of Structural Causes," in Proceeding of the Genetic and Evolutionary Computation Conference (GECCO2003), LNCS 2724, pp.1665-1677, Chicago, IL, USA, July 12-16, 2003

[10] D. Zongker B. Punch, *Lil-GP User's Manual*. Michigan State University, July 1995.