최적화 기법인 mDEAS의 개발 및 휴머노이드 이족보행 시 최적 관절궤적 생성에의 적용

論 文
58-2-26

Development of Modular DEAS (mDEAS) and its Application to Optimal Trajectory Generation of Biped Walking

金 銀 宿* · 金 祚 煥* · 金 鍾 旭†
(Eunsu Kim · Johwan Kim · Jong-Wook Kim)

Abstract - This paper newly proposes a modular type dynamic encoding algorithm for searches (DEAS) which partitions the whole parameters into several modules and carries out exhaustive DEAS for each module. uDEAS is used to measure parameter sensitivities to the cost function, and the variables whose sensitivities are similar are grouped to make a module. The proposed optimization method is applied to optimal trajectory generation for biped walking of a humanoid, and the optimization result is compared with those of the former versions of DEAS.

Key Words : Numerical optimization, DEAS, Heuristics, Humanoid, Biped walking

1. 서 론

DEAS(dynamic encoding algorithm for searches)는 현재 탐색점에서 비용함수의 미분 값을 계산할 필요 없이, 규칙적으로 생성되는 이웃 점의 비용함수 값들을 순차적으로 비교함으로써 결과적으로 전역해를 찾을 수 있는 메타휴리스틱(metaheuristic) 최적화 알고리듬이다[1]-[6]. DEAS는 유전알고리듬(genetic algorithm, GA)[7]처럼 이진수로 실수 해를 표현하지만 다변수 문제의 경우 GA와는 달리 이진 행렬로써 실수 벡터를 표현한다. 특히 탐색 정밀도를 높이기 위해 이진 행렬의 각 행 최하위 비트(least significant bit, LSB) 오른쪽에 하나의 이진 비트를 삽입한다.

DEAS의 최초 형태는 모든 행의 LSB에 동시에 0 또는 1을 추가함으로써 모든 가능한 행렬의 조합을 만들었으므로, 파라미터가 n개인 경우 총 2^n개의 이웃점이 생성되었다. 이렇게 조밀하게 지역탐색(local search)을 수행하도록 고안된 탐색법을 exhaustive DEAS(eDEAS)라고 명명했고, 탐색 파라미터의 수가 비교적 작은 변압기 코아 최적설계[3]와 PID 제어기 설계[5]에 대해 우수한 최적화 성능을 보였다. 그러나 탐색 파라미터의 수가 증가할 수록, 계산량이 기하급수적으로 증가하는 문제가 발생하므로 이를 위한 해결책으로 univariate DEAS(uDEAS)가 개발되었다[4],[6].

uDEAS는 탐색점의 표현형인 이진 행렬에서 한 번에 한 행씩 각 LSB에 0 또는 1을 추가하고 연이어 증가/감소 연산을 수행함으로써, 한 번의 지역해 천이(transition)를 위해 각 파라미터 당 최대 2번씩 비용함수를 계산한다. 그러므로 n개의 파라미터에 대해 순차적으로 탐색을 마치면 약 $2n$번의 비용함수 계산 횟수가 소요된다. 이러한 지역탐색법으로 인해 uDEAS는 탐색 파라미터의 수가 최대 30인 테스트 함수 최적화[4]와 13개의 유도전동기 파라미터 추정[6]에서 탁월한 탐색 속도와 정확도를 보였다. 그러나, uDEAS는 변수별로 단방향적인 탐색을 수행함으로써 변수의 탐색 순서에 영향을 받는 단점이 있다.

본 논문에서는 상기한 eDEAS의 높은 탐색 성능과 uDEAS의 빠른 탐색 속도를 보완한 modular DEAS (mDEAS)를 새롭게 제안하고, 이를 휴머노이드 로봇의 이족보행을 위한 최적 관절 궤적 생성에 적용하여 성능을 비교분석한다. 휴머노이드 로봇의 이족보행은 지능형 로봇의 핵심 기술 중 하나이지만, 영모먼트점(zero moment point, ZMP) 안정도를 확보하면서 모터의 구동 에너지를 최소화하는 관절 궤적에 대한 연구는 초기 단계에 있다고 할 수 있다[8]. 기존의 방법은 집단 기반 탐색법인 GA를 사용하므로 탐색 시간을 줄이는데 한계가 있으며, 이는 실시간 제어에 적용하기에 어려운 문제점이 있다.

본 논문에서는 제안한 mDEAS의 성능을 비교하기 위해 eDEAS와 uDEAS, 그리고 mDEAS를 이용해서 휴머노이드 하지 관절들의 최적 보행 궤적을 생성한 후 각 방법에서 소요된 시간과 구해진 해의 정확도를 비교 분석함으로써 제안된 mDEAS의 성능을 검증한다.

2. mDEAS

eDEAS, uDEAS, 그리고 제안된 mDEAS는 크게 지역최적화(local search)와 전역최적화(global search) 루틴으로 나눌 수 있으며, 지역최적화는 탐색점 주변을 양분해서 좀 더 면밀히 탐색하는 bisectional search(BSS)와 BSS에서 얻은 정보를 기반으로 개선된 해가 존재할 가능성이 높은 영역을

† 교신저자, 準會員 : 東亞大學 電子工學科 專任講師 · 工博
E-mail : kjwook@dau.ac.kr
* 準 會 員 : 東亞大學 電子工學科 碩師課程
接受日字 : 2008年 12月 2日
最終完了 : 2009年 1月 14日

확장 탐색하는 unidirectional search(UDS)로 구성된다. DEAS에서는 한 번의 BSS와 연이은 다수의 UDS 탐색조합을 session으로 정의하고, session이 진행됨에 따라 이진 행렬의 열의 길이가 1씩 증가한다. 그러므로 변수가 n개인 최적화 문제에 대해 탐색 시작점의 표현형이 $n\times m$ 행렬이고, k번의 session이 DEAS에 의해 수행되었다면 결과적으로 $n\times(m+k)$ 이진행렬이 현재 지역해의 표현형으로 얻어진다.

일반적인 공학적 최적화 문제에는 여러 개의 지역해가 존재하므로, 전역최적화에 대한 고려가 반드시 수반되어야 한다. DEAS는 전역최적화 기법 중 가장 용이하면서도 간단하게 구현할 수 있는 multistart 기법을 채용했다. multistart 기법은 탐색 시작점을 무작위로 정하고 그 시작점으로부터 지역최적화 기법을 수행해서 지역해를 발견한 후, 이 작업을 다시 임의의 시작점으로부터 주어진 횟수만큼 재시작(restart) 함으로써 최종적으로 얻어진 지역해 중 가장 좋은 지역해를 전역해로 간주하는 전역최적화법이다.

mDEAS는 eDEAS, uDEAS와는 동일한 전역최적화 기법을 사용하므로 본 논문에서는 지역최적화 기법에 대해 주로 설명한다. 제안된 mDEAS는 각 모듈을 uDEAS가 계산한 비용함수 민감도 값들을 이용해서 분할하고, 탐색 정밀도가 높은 eDEAS를 각 변수 모듈에 적용하므로 본 절에서는 eDEAS와 uDEAS의 지역최적화 원리를 간략히 설명한다.

2.1 eDEAS

2.1.1 Bisectional Search(BSS)

BSS는 이진 행렬로 표현되는 현재의 지역해로부터 이웃점을 생성하는 과정과, 이렇게 생성된 이진 행렬들을 복호화(decoding)하고 비용함수를 계산하는 과정, 그리고 그 결과로서 최소의 비용함수를 갖는 행렬을 선택하고 최적 탐색방향을 도출하는 과정으로 구성된다. 여기에서 도출된 최적 탐색 방향은 곧이어 진행되는 UDS의 기본 탐색방향이 된다.

2개의 변수 x_1, x_2를 최적화하는 문제를 예로 들어 eDEAS에 대한 BSS의 각 단계를 설명하도록 한다. 예를 들어 직전 session의 결과로서 구해진 최적해의 표현형 행렬이 다음과 같다고 하자.

$$M=\begin{bmatrix}1&0&1\\0&1&1\end{bmatrix}$$

BSS는 현 탐색점의 이웃점을 다음과 같이 현재의 최적행렬 M의 최하위열에 0과 1의 조합으로 이루어진 열을 삽입함으로써 총 2^2개의 이웃행렬을 생성시킨다.

$$M^{(1)}=\begin{bmatrix}M&0\end{bmatrix}=\begin{bmatrix}1&0&1&0\\0&1&1&0\end{bmatrix},\quad M^{(2)}=\begin{bmatrix}M&0\end{bmatrix}=\begin{bmatrix}1&0&1&0\\0&1&1&1\end{bmatrix},$$

$$M^{(3)}=\begin{bmatrix}M&1\end{bmatrix}=\begin{bmatrix}1&0&1&1\\0&1&1&0\end{bmatrix},\quad M^{(4)}=\begin{bmatrix}M&1\end{bmatrix}=\begin{bmatrix}1&0&1&1\\0&1&1&1\end{bmatrix}.$$

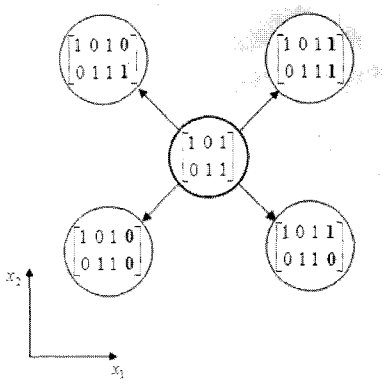

그림 1 BSS에서 탐색 시작점과 이웃점 간의 이진행렬 관계도
Fig. 1 Relational diagram of neighboring binary matrices in BSS

그림 1은 M과 4개의 이웃행렬 $M^{(i)}$, $i=1,\cdots,4$와의 관계를 도식화한 것이다. 어떤 이진 스트링의 최하위 비트에 0을 붙이고 복호화하면 최초 이진 스트링으로부터 복호화한 실수값보다 감소하고, 1을 붙이고 복호화하면 이전의 실수값보다 증가하는 특성이 있으므로[1], 그림에서 보듯이 M의 각 행에 0이 붙은 경우는 해당 변수에 대해 감소하는 방향, 1이 붙은 경우는 증가하는 방향에 이웃점이 존재함을 알 수 있다.

그 다음 단계로 각 이웃행렬을 행별 복호화하고, 그 결과 얻어진 실수 벡터를 이용해서 비용함수를 계산하는 과정이 수행된다. 복호화를 위해서는 길이가 m인 이진 스트링 $[b_m b_{m-1}\cdots b_1]$을 $[r_l, r_u]$ 범위에서 실수로 변환시키는 아래 복호화 함수를 사용한다.

$$f_d([b_m b_{m-1}\cdots b_1],r_u,r_l)=r_l+\frac{r_u-r_l}{2^{m+1}}\left(\sum_{j=1}^{m}b_j 2^j+1\right),b_i\in 0,1 \tag{1}$$

여기에서 r_l과 r_u는 복호화하는 실수값의 최소값과 최대값을 의미한다. 식 (1)은 한 스트링에 대해 복호화를 행한 것을 나타내며 위의 이웃점 행렬들을 복호화 하면 다음과 같이 이웃점 실수 벡터를 얻을 수 있다.

$$X^{(1)}=\begin{bmatrix}f_d([1\,0\,1\,0],x_1^u,x_1^l)\\f_d([0\,1\,1\,0],x_2^u,x_2^l)\end{bmatrix},\quad X^{(2)}=\begin{bmatrix}f_d([1\,0\,1\,0],x_1^u,x_1^l)\\f_d([0\,1\,1\,1],x_2^u,x_2^l)\end{bmatrix},$$

$$X^{(3)}=\begin{bmatrix}f_d([1\,0\,1\,1],x_1^u,x_1^l)\\f_d([0\,1\,1\,0],x_2^u,x_2^l)\end{bmatrix},\quad X^{(4)}=\begin{bmatrix}f_d([1\,0\,1\,1],x_1^u,x_1^l)\\f_d([0\,1\,1\,1],x_2^u,x_2^l)\end{bmatrix}$$

여기에서 x_1^u, x_1^l과 x_2^u, x_2^l은 x_1과 x_2 변수 탐색영역의 최대값과 최소값을 각각 의미한다.

BSS의 마지막 단계는 상기 과정으로 얻어진 이웃점들에 대해 비용함수를 구한 후 비용함수 값이 최소인 이웃점 X^*를 갱신된 지역해로 결정하는 것이다. 비용함수를 J라고 하면 X^*는 다음과 같이 표현된다.

$$X^*=\arg\min\{J(X^{(1)}),J(X^{(2)}),J(X^{(3)}),J(X^{(4)})\} \tag{2}$$

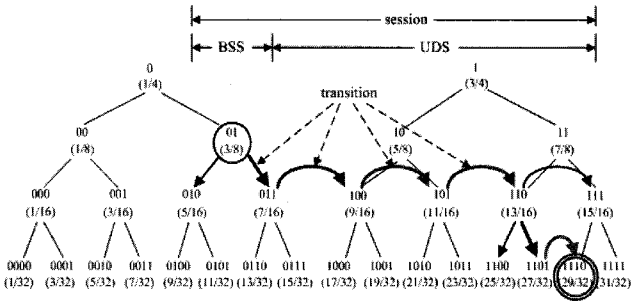

그림 2 1차원 탐색 영역에서 BSS와 UDS 실행 예(괄호 안은 0과 1 사이에서 복호화한 실수값)

Fig. 2 Example of BSS and UDS in one-dimensional search space

식 (2)에서 최소 비용함수 값을 갖는 파라미터 벡터 X^*의 최하위 열을 조사하면 BSS의 결과로 얻어진 최적방향 벡터 d_{opt}를 구할 수 있다. 상기 예에서 만약 $X^{(3)}$가 X^*로 판정되었다면 d_{opt}는 해당 행렬인 $M^{(3)}$의 최하위 열인 $[1\ \ 0]^T$이 된다. 이는 현 탐색점 M에서의 비용함수 강하(descent) 방향이 x_1 방향으로 양의 방향(1), x_2 방향으로 음의 방향(0)임을 의미한다. 그리고 d_{opt}는 후속적인 UDS를 지도하는 역할을 한다.

2.1.2 Unidirectional Search(UDS)

BSS가 탐색 폭(step length)을 반감시키면서 심화 탐색(exploitation)을 수행하는 것에 비해, UDS는 탐색 간격을 유지하면서 광역 탐색(exploration)을 수행하는 특성을 갖는다. 만약 지역해가 탐색점 주변에 존재할 경우 BSS만으로도 조기에 수렴할 수 있지만, 그렇지 않은 경우 d_{opt} 방향으로 여러 번의 둥거리 탐색을 수행해야 한다. 그림 2는 1차원 탐색에서의 BSS와 UDS를 예시한 것으로, multistart 기법에서 사용되는 초기 이진 스트링 길이를 2로 설정했다고 가정한다. 이때 무작위로 생성되는 초기 이진 스트링은 변수당 총 $4(=2^2)$개가 되며 각 초기 행렬이 BSS만으로 도달할 수 있는 영역은 전체 구간의 1/4이 된다. 그림에서처럼 지역해가 [1110]인 상황에서 [01]이 초기 이진 스트링이라면 한번의 BSS 후에 4번의 연속적인 UDS로써 신속히 인근 이진 스트링([110])로 접근할 수 있다. 이를 위해 UDS는 스트링 길이를 변화시키지 않고 직전의 BSS에서 계산된 d_{opt}를 기반으로 이진 가산(increment addition)이나 이진 감산(decrement subtraction)을 수행한다. 즉, 그림 2에서 첫 번째 BSS결과 오른쪽 방향(양의 방향)이 d_{opt}로 판정되었으므로, UDS는 비용함수 값이 감소하는 한 011→100→101→… 의 순서로 이진 가산을 수행한다.

앞 절의 2차원 예제의 경우처럼 BSS의 결과 $[1\ 0]^T$가 d_{opt}인 경우, UDS는 x_1의 스트링에 대해서는 이진 가산을, x_2의 스트링에 대해서는 이진 감산을 수행한다. UDS는 효과적인 탐색방향 생성을 위해 확장 벡터(extension vector)를 이용하는데 BSS 방향으로의 확장탐색을 수행하면 1, 그렇지 않으면 0으로 표현한다. 확장 벡터는 BSS와 유사한 원리에 의해 n차원의 문제에 대해 최대 2^n-1개가 생성되는

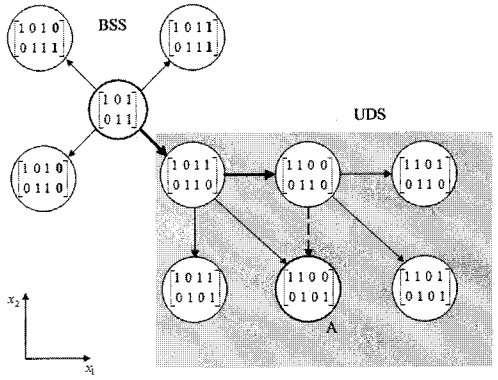

그림 3 2차원에서의 eDEAS session 예시

Fig. 3 Example of eDEAS's one session in two-dimensional search space

데 그 이유는 $[0\ 0]^T$는 아무런 확장 탐색이 일어나지 않은 것이므로 제외되기 때문이다.

그림 3은 상기 2차원 문제에 대해 한 번의 session이 실행되는 예를 도시한 것인데, 굵은 선으로 표시된 것은 지역해의 천이 방향을 나타낸다. 그런데, 두 번째 UDS에서 생성된 이웃 행렬 중 A 행렬이 직전 UDS에서 이미 탐색되었음을 알 수 있다. 이는 반복적으로 수행되는 UDS의 특성에 의해 발생되는 재탐색(revisit) 문제에 의해 발생되며, eDEAS에서는 이진 스트링의 특성을 활용한 masking 기법을 사용함으로써 이러한 재탐색 문제를 미리 방지할 수 있다[1]. 이러한 session은 각 행의 길이가 주어진 최대값(*maxRowLen*)이 될 때까지 반복되며, 이 경우 multistart 기법에 의해 초기행렬 길이(*initRowLen*)의 이진 행렬이 무작위로 생성되어 이로부터 session이 계속 수행된다.

2.2 uDEAS

그림 3에서 알 수 있듯이 eDEAS는 n차원의 최적화 문제에 대해 한 번의 session 동안 BSS는 2^n번, UDS는 최대 2^n-1번의 비용함수 계산을 해야 한다. 이러한 계산량은 n이 증가할 수록 큰 문제가 되므로 10차원 이상의 문제에는 적용하기가 부적합하다. 이에 대한 해결책으로 저자는 uDEAS를 개발했다[4],[6].

uDEAS는 모든 변수에게 동시에 변화를 주며 이웃행렬을 생성하는 방법 대신, 나머지 변수는 고정한 채 한 변수에 대해서만 BSS와 UDS를 수행한 후 그 다음 변수로 진행시키는 방식으로 개선된 DEAS이다. x_1부터 x_n까지 생성된 이웃행렬의 총 수는 파라미터의 약 2배수에 불과하지만, 30차원의 고차 함수는 물론 저차의 테스트 함수에 대해서도 eDEAS와 거의 동등한 탐색 성능을 보였다[4].

앞 절의 2개의 변수 x_1, x_2를 최적화하는 문제를 예로 들어 uDEAS의 BSS와 UDS를 설명하도록 한다. uDEAS는 행렬 M의 x_1에 대해 먼저 BSS와 UDS를 한다. 즉, 아래와 같이 2개의 이웃행렬을 생성한 후,

$$M_B^{(1)}=\left[M\ \ 0\right]=\begin{bmatrix}1&0&1&0\\0&1&1&\end{bmatrix},\quad M_B^{(2)}=\left[M\ \ 1\right]=\begin{bmatrix}1&0&1&1\\0&1&1&\end{bmatrix}$$

아래와 같이 각 행별로 복호화하여 실수 벡터를 구한다.

$$X^{(1)}=\begin{bmatrix} f_d([1\,0\,1\,0],x_1^u,x_1^l) \\ f_d([0\,1\,1\ ],x_2^u,x_2^l)\end{bmatrix},\quad X^{(2)}=\begin{bmatrix} f_d([1\,0\,1\,1],x_1^u,x_1^l) \\ f_d([0\,1\,1\ ],x_2^u,x_2^l)\end{bmatrix}.$$

그리고 계산된 비용함수 $J(X_1)$과 $J(X_2)$를 계산하여 비교한다. eDEAS의 경우와 동일한 문제라면 BSS의 결과 $J(X_2)<J(X_1)$이 되어 X_2의 x_1행에 대해 양의 방향으로 UDS가 수행된다. 즉, 아래와 같이 이웃점을 생성 후,

$$X_U^{(1)}=\begin{bmatrix} f_d([1\,1\,0\,0],x_1^u,x_1^l) \\ f_d([0\,1\,1\ ],x_2^u,x_2^l)\end{bmatrix},\quad X_U^{(2)}=\begin{bmatrix} f_d([1\,1\,0\,1],x_1^u,x_1^l) \\ f_d([0\,1\,1\ ],x_2^u,x_2^l)\end{bmatrix},$$
$$X_U^{(3)}=\begin{bmatrix} f_d([1\,1\,1\,0],x_1^u,x_1^l) \\ f_d([0\,1\,1\ ],x_2^u,x_2^l)\end{bmatrix},\cdots$$

만약 $X_U^{(2)}=\arg\min\left\{J(X_U^{(1)}),J(X_U^{(2)}),J(X_U^{(3)}),\cdots\right\}$임이 판명되었다면, x_1에 대해서 BSS와 UDS가 완료되었다. 이렇게 구해진 $X_U^{(2)}$로부터 x_2에 대해 BSS를 행하면 다음과 같이 두 개의 이웃행렬이 생성된다.

$$M_B^{(1)}=\begin{bmatrix}1\,1\,0\,1\\0\,1\,1\,0\end{bmatrix},\ M_B^{(2)}=\begin{bmatrix}1\,1\,0\,1\\0\,1\,1\,1\end{bmatrix}.$$

동일한 방법으로 x_2에 대해 UDS를 수행하면 단봉함수(unimodal function)의 경우 eDEAS와 동일한 최적 행렬을 얻을 수 있다.

uDEAS는 eDEAS와 달리 기본 session이 변수별로 이루어지므로, 변수 탐색 순서가 탐색성능에 영향을 줄 가능성이 있다. 그러므로 주어진 비용함수에 적합한 탐색 순서를 찾아 낼 필요가 있는데, 각 변수가 비용함수에 미치는 민감도(sensitivity) 정보를 이용하는 것이 바람직하다고 할 수 있다. 즉, 평균 민감도가 큰 변수의 순서로 BSS와 UDS를 행하면 효율적이고 강건한 탐색 성능을 가질 것으로 예상된다. 어떤 변수 $x_i \in X$가 비용함수 $J(X)$에 대해 탐색 중 계산되는 민감도는 다음과 같이 정의할 수 있다.

$$S(x_i^{(k)})=\left|\frac{J([x_1,\cdots,x_i^{(k)},\cdots,x_n])-J([x_1,\cdots,x_i^{(k-1)},\cdots,x_n])}{x_i^{(k)}-x_i^{(k-1)}}\right| \tag{3}$$

식 (3)에서 $x_i^{(k-1)}$와 $x_i^{(k)}$는 $k-1$번째와 k번째 지역해의 x_i 값을 의미한다. 본 논문에서는 mDEAS에서 전체 변수를 모듈로 나눌 때 상기 민감도 정보를 활용한다.

2.3 mDEAS

상기한 eDEAS와 uDEAS의 각 문제점에 대한 해결책으로 본 논문에서는 전체 파라미터를 몇 개의 그룹으로 나누고, 각 그룹에 대해 eDEAS를 수행하는 mDEAS를 개발했다. 그림 4는 변수가 10개인 일반적인 문제에 대해 mDEAS의 한 session을 수행한 예를 도시한다.

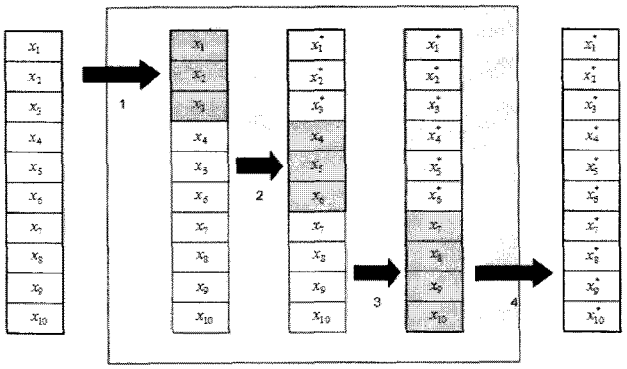

그림 4 3개의 모듈에 대한 mDEAS session별 탐색 개념도
Fig. 4 Search aspect of mDEAS sessions for the problem that has three modules

그림에서 총 10개의 변수가 3개의 모듈로 나누어졌고, 첫 번째 모듈에서는 $x_1 \sim x_3$의 변수를 eDEAS로 최적화하며, 두 번째 모듈에서는 $x_4 \sim x_6$의 변수를, 세 번째 모듈에서는 나머지 $x_7 \sim x_{10}$의 변수를 eDEAS로 최적화한다. 그림 4에서 주의할 점은 각 모듈 별 최적화를 수행할 때 비용함수를 계산하는 단계에서는 전체 파라미터가 모두 필요하다는 사실이다. 그러므로 두 번째 모듈에서 최적화 시 첫 번째 모듈에 해당되는 변수는 기 최적화된 $x_1^* \sim x_3^*$을 사용하고, 세 번째 모듈에 해당되는 변수는 초기 변수 $x_7 \sim x_{10}$을 사용한다. 마찬가지로 세 번째 모듈에 대해 최적화를 수행할 때는 앞에서 최적화된 $x_1^* \sim x_3^*$와 $x_4^* \sim x_6^*$을 사용한다. 이는 직전에 구해진 최적 값들이 지역해에 보다 근사하다고 가정하고 나머지 변수들을 최적화하는 것이 일반적으로 탐색 속도가 빠르기 때문이다.

그림 4에서 예시한 3개의 모듈에 대해 eDEAS를 적용할 경우, x_1에서 x_{10}까지 전체 변수에 대해 한 번의 지역해 천이를 위해서는 첫 번째 모듈에 대해 2^3번, 두 번째 모듈에 대해 2^3번, 세 번째 모듈에 대해 2^4번이 소요되어 총 32번의 비용함수 계산이 소요된다. 이는 $1024(=2^{10})$번의 계산이 필요한 eDEAS의 약 3%, $20(=2\times10)$번의 계산이 필요한 uDEAS의 약 1.6배에 해당되는 작은 계산량이다. 일반적으로 n개의 변수를 m개의 모듈로 분해하고, 각 모듈의 크기(변수의 개수)를 c_i라고 하면 transition 당 비용함수 계산량은 다음 식으로 얻을 수 있다.

$$C=\sum_{i=1}^{m}2^{c_i},\qquad c_1+c_2+\cdots+c_m=n \tag{4}$$

mDEAS 적용시 계산량을 감소하기 위해 모듈을 많이 나눌수록 uDEAS의 성능에 가까워지므로 문제의 특성에 맞도록 적절하게 모듈을 분할하는 것이 중요하다. mDEAS에서 전체 변수를 모듈화할 때 종합적으로 고려해야 할 사항은 다음과 같다.

- 모듈의 개수
- 모듈 구분 기준
- 모듈의 탐색 순서

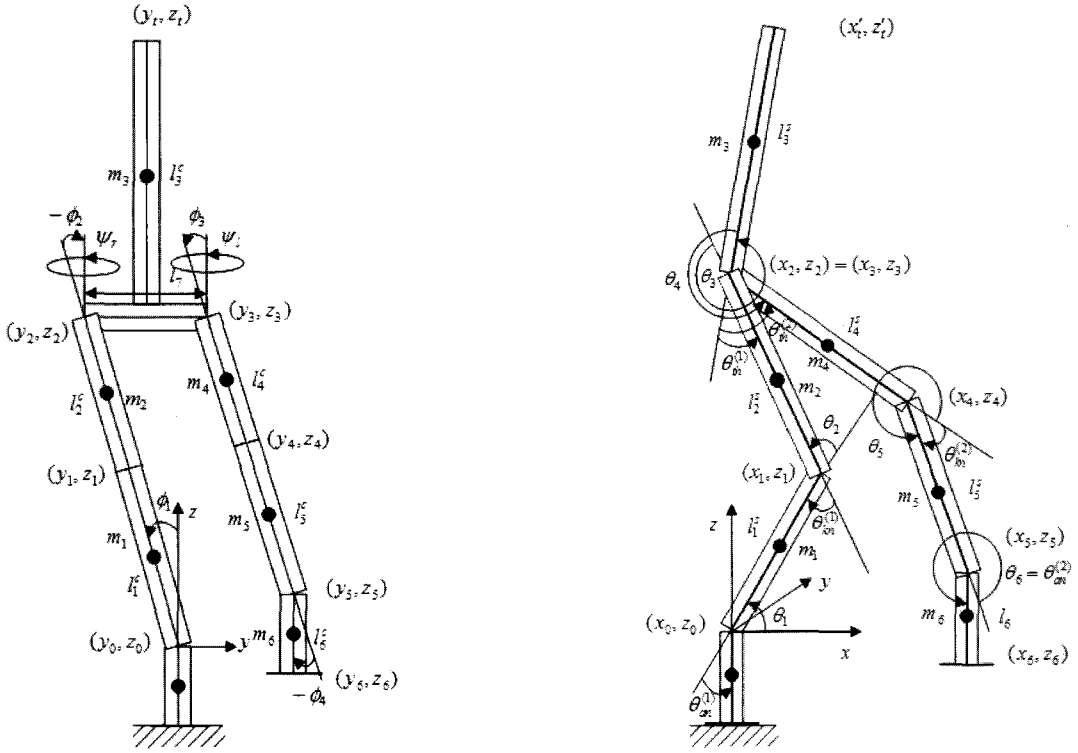

그림 5 오른발지지 시 이족 로봇의 평면별 모델

Fig. 5 Models of a biped robot in supporting with the right leg viewed in separate planes

본 논문에서는 휴머노이드의 이족 보행 문제에 특화된 mDEAS 개발을 위해 다음과 같은 방법으로 모듈을 분할하고 모듈별 탐색 순서를 정하였다.

1) uDEAS를 이용해서 변수별 비용함수 민감도를 계산한다.
2) 변수별 민감도 순서로 정렬 후 등분에 가깝게 clustering 한다.
3) 그 결과로 얻어진 각 module에 대해 eDEAS를 수행한다.
4) multistart 기법을 이용하여 전역 최적화를 수행한다.

3. 휴머노이드의 이족보행

3.1 휴머노이드의 기구학과 동역학

휴머노이드가 넘어지지 않으면서 목표 위치에 정확히 발을 내려놓기 위해서는 정밀한 기구학(kinematis)과 정확한 동역학(dynamics)이 요구된다. 기구학은 일반적으로 사용되는 Denavit-Hartenberg(DH) 방법을 이용해서 휴머노이드의 모든 관절에 대해 비교적 체계적으로 계산할 수 있지만, DH법을 적용하기 위해서는 SCARA 로봇처럼 기저에서 말단장치까지 모든 링크가 연속적으로 연결되어 있어야 하며, 상대 좌표계에서 y축 방향으로의 움직임이 없어야 한다[9]. 그러나 인간 신체의 관절-링크 구조는 상기한 두 가정을 위배하므로, 보다 적합하고 효율적인 kinematics 체계를 구축해야 한다. 아울러 관절의 수만큼 DH 변환행렬을 곱해야 하므로 관절 수가 총 20~30개에 이르는 휴머노이드 로봇에 대해서는 계산 시간이 긴 문제점이 발생한다.

저자는 [10]에서 휴머노이드의 이족 보행시 측면에서 본 평면(sagittal plane, 시상면)과 정면에서 본 평면(coronal plane, 관상면), 그리고 머리 위에서 본 평면(transverse plane, 횡평면) 내에서의 움직임을 각각 따로 모델링하고, 이를 동시에 결합하는 방법으로 3차원상에서 모델링한 기법을 제안했다. 그림 5는 이해를 돕기 위해 휴머노이드가 오른발로 지지하고 왼 발을 들었을 때 관상면과 시상면에서 본 모습을 링크-관절 모델로 표현한 것이다.

휴머노이드 모델에서 하체는 각 다리를 3개의 링크가 구성하며, 대퇴부를 l_7 링크가 지지하므로 총 7개의 링크로 모델링되고, 상체는 머리와 팔, 몸통을 대표하는 하나의 링크 l_3로 모델링되었다. 또한 모델에서 독립적으로 회전할 수 있는 관절은 시상면의 6개 관절 $\theta_i, i=1,\cdots,6$, 관상면의 4개 관절 $\phi_i, i=1,\cdots,4$, 횡평면의 2개 관절 $\psi_i, i=l,r$이 되어 총 12개 관절이 하체에 존재한다. 결국 휴머노이드의 하체만을 모델링하는 데만 해도 최소 7개의 링크와 12개의 관절이 필요함을 알 수 있다. 저자의 선행 연구 [10]에서는 투영(projection)기법을 이용하여 3차원 공간 상에서 움직이는 링크를 상기한 3개의 평면에서 2차원으로 표현하는 새로운 기구학을 제안했다. 즉, 오른 발로 지지한 상태에서 시상면의 관절과 관상면의 관절을 모두 움직이면서 오른쪽 다리의 대퇴부 관절 ψ_r을 회전시키면 다음과 같은 기구학을 얻을 수 있다.

$$
\begin{aligned}
&x_1 = x_0 + l_1^s C_1, \quad y_1 = y_0 - l_1^c S_{\phi}, \quad z_1 = z_0 + l_1^s S_1,\\
&x_2 = x_1 + l_1^s C_{12}, \quad y_2 = y_1 - l_2^c S_{\phi}, \quad z_2 = z_1 + l_1^s S_{12},\\
&x_3 = x_2 - l_7 S_{\psi_r}, \quad y_3 = y_2 + l_7 C_{\psi_r}, \quad z_3 = z_2,\\
&x_4 = x_3 + l_4^s C_{\psi_r} C_{1234} + l_4^c S_{\psi_r} S_{\phi},\\
&y_4 = y_3 + l_4^s S_{\psi_r} C_{1234} - l_4^c C_{\psi_r} S_{\phi}, \quad z_4 = z_3 + l_4^s S_{1234},\\
&x_5 = x_4 + l_5^s C_{\psi_r} C_{12345} + l_4^c S_{\psi_r} S_{\phi},\\
&y_5 = y_4 + l_5^s S_{\psi_r} C_{12345} - l_5^c C_{\psi_r} S_{\phi}, \quad z_5 = z_4 + l_5^s S_{12345},\\
&x_6 = x_5 + l_6^s C_{\psi_r} C_{123456},\\
&y_6 = y_5 + l_6^s S_{\psi_r} C_{123456}, \; z_6 = z_5 + l_6^s S_{123456}
\end{aligned}
\tag{5}
$$

식에서 대문자 C와 S는 각각 cos 함수와 sin 함수를 나타내며, 아래첨자 $1\cdots n$은 $\theta_1+\theta_2+\cdots+\theta_n$을 의미한다. 즉, $C_{1234}=\cos(\theta_1+\cdots+\theta_4)$이며 $S_{\psi_r}=\sin\psi_r$을 나타낸다. 또한, 링크 l_i^s와 l_i^c은 i번째 링크를 시상면과 관상면으로 투영한 링크 길이를 의미하며 원래의 i번째 링크 길이 l_i와는 다음과 같은 관계가 있다.

$$
\begin{aligned}
&l_1^s = l_1 C_{\phi}, \quad l_2^s = l_2 C_{\phi}, \quad l_3^s = l_3,\\
&l_4^s = l_4 C_{\phi}, \quad l_5^s = l_5 C_{\phi}, \quad l_6^s = l_6
\end{aligned}
\tag{6}
$$

$$
\begin{aligned}
&l_1^c = l_1 S_1, \quad l_2^c = l_2 S_{12}, \quad l_3^c = l_3 S_{123},\\
&l_4^c = l_4 S_{1234}, \quad l_5^c = l_5 S_{12345}, \quad l_6^c = l_6 S_{123456}.
\end{aligned}
\tag{7}
$$

식 (5)와 (6)은 로봇이 보행시 상체를 관상면과 시상면에서 모두 직각으로 유지한다는 조건에서 유도되었다. 이를 위해 그림 5(a)의 4개의 관상면 관절에 대해 $\phi_1=-\phi_2=\phi_3=-\phi_4=\phi$의 조건을 부여했다. 이렇게 상체를 직각으로 유지하는 것은 모델링의 편리성도 있지만, 로봇 머리 부분에 카메라 센서를 부착할 경우 전방 영상의 흔들림을 최소화하기 위한 목적도 있다. 본 논문에서는 mDEAS의 성능 검증이 목표이므로, 보행시 회전은 하지 않도록 했으며 이 경우 기구학은 식 (5)에서 ψ_r 혹은 ψ_l에 0을 대입하면 얻을 수 있다. 왼발로 지지하는 경우는 (5)에서 ψ_r을 ψ_l로 바꾸고, $y_3=y_2+l_7C_{\psi_r}$을 $y_3=y_2-l_7C_{\psi_l}$로 바꾸면 된다.

이족보행 시 로봇의 동역학을 계산하기 위해 각 관절 모터에서 발생시키는 토크는 다음과 같이 Euler-Lagrange 수식을 사용했다.

$$
\sum_{j=1}^{6} d_{kj}(\theta)\ddot{\theta}_j + \sum_{i=1}^{6}\sum_{j=1}^{6} c_{ijk}\dot{\theta}_i\dot{\theta}_j + g_k(\theta) = \tau_k, \quad k=1,\cdots,6
\tag{8}
$$

식에서 d_{kj}는 관성행렬 $D(q)$의 원소, g_k는 중력 벡터, τ_k는 k번째 관절의 모터가 발생시키는 토크, 그리고 c_{ijk}는 Christoffel 기호로서 다음과 같이 정의된다.

$$
c_{ijk} = \frac{1}{2}\left\{\frac{\partial d_{kj}}{\partial\theta_i} + \frac{\partial d_{ki}}{\partial\theta_j} - \frac{\partial d_{ij}}{\partial\theta_k}\right\}.
\tag{9}
$$

본 논문에서는 휴머노이드 로봇의 이동성을 극대화하기 위해 시상면 관절 모터에서 발생시키는 총 토크(모터에서 소비하는 전기 에너지)를 최소화하는 것을 목적으로 하므로, 식 (8)에서 볼 수 있듯이 시상면 관절에 대한 총 토크만을 계산했다.

3.2 최적 관절궤적 생성

본 논문에서는 각 관절모터의 각도 궤적을 혼합 다항식(blending polynomial)으로써 근사화했다. 혼합 다항식은 전체 궤적을 부분 궤적(segment)으로 분할하고 각 부분 궤적 경계점에서의 함수 값과 기울기 값을 부여해주면, 연속조건을 이용해서 전체 궤적을 수식으로 표현하는 방법이다[9]. 혼합 다항식은 저차의 다항식을 이어 붙인 것이기 때문에, 고차의 다항식이 필요한 부드러운 곡선을 보다 효율적으로 표현할 수 있다.

본 논문에서는 혼합 다항식의 부분 궤적을 다음과 같이 3차 다항식으로 표현했다.

$$
\begin{aligned}
&q(t) = a_0 + a_1(t-t_0) + a_2(t-t_0)^2 + a_3(t-t_0)^3,\\
&a_0 = q_0, \; a_1 = v_0,\\
&a_2 = \frac{3(q_1-q_0) - (2v_0+v_1)(t_f-t_0)}{(t_f-t_0)^2},\\
&a_3 = \frac{2(q_0-q_1) + (v_0+v_1)(t_f-t_0)}{(t_f-t_0)^3}.
\end{aligned}
\tag{10}
$$

여기에서 t_0와 t_f는 초기 시간과 최종 시간, q_0와 q_f는 초기 각도와 최종 각도, 그리고 v_0와 v_f는 초기 각속도와 최종 각속도를 각각 의미하며 다음과 같은 관계가 있다.

$$
q(t_0)=q_0,\; q(t_f)=q_f,\; \left.\frac{dq(t)}{dt}\right|_{t=t_0} = v_0,\; \left.\frac{dq(t)}{dt}\right|_{t=t_f} = v_f.
\tag{11}
$$

식 (10)에서 알 수 있듯이 하나의 부분 궤적을 3차의 다항식으로 표현하기 위해서는 총 6개의 계수(t_0, t_f, q_0, q_f, v_0, v_f)가 필요하다. 만약 두 개의 부분 궤적을 $t_0\sim t_m$구간과 $t_m\sim t_f$ 구간에서 만든 후, 이어 붙여서 하나의 궤적을 만드는 경우 결정해야 할 계수는 다음과 같이 총 9개가 된다.

$$
\boldsymbol{p_2} = [t_0\; t_m\; t_f\; q_0\; q_m\; q_f\; v_0\; v_m\; v_f]
\tag{12}
$$

식 (12)에서 중복탐색을 피하기 위해 시간 계수는 $t_m=\frac{t_0+t_f}{2}$로 설정하고, 각 관절의 초기 속도와 최종 속도는 보행의 시작과 끝을 부드럽게 하기 위해 $v_0=v_f=0$으로 설정했다. 또한 휴머노이드는 발을 바꾸면서 주기적으로 보행하고 시작과 끝부분에서 ZMP 불안정성을 보이므로, 보행 시작과 종료시의 관절 각도 q_0와 q_f는 적절한 각도값을 할당해 주었다. 그러므로 식 (12)에서 최종 미지수로 남는 값

은 q_m과 v_m이 되며 이를 mDEAS로 구한다. 여기에서 주의할 점은 시상면 관절각 θ_i, $i=1,\cdots,6$는 기하하적인 각도이므로 관절 모터의 실제적인 궤적과는 다르며, 혼합 다항식은 실질적인 관절 모터의 궤적을 표현한다는 사실이다. 그러므로 mDEAS가 관절 모터의 궤적과 관련한 이웃점을 탐색하면 그로부터 다음 관계를 이용해서 시상면 관절각을 구하고, 이로부터 기구학과 동역학을 계산하면 된다.

$$\theta_1=\frac{\pi}{2}-\theta_{an}^{(1)},\ \theta_2=\theta_{kn}^{(1)},\ \theta_3=-\theta_{th}^{(1)},$$
$$\theta_4=\pi+\theta_{th}^{(2)},\ \theta_5=-\theta_{kn}^{(2)},\ \theta_6=\theta_{an}^{(2)}. \tag{13}$$

여기에서 $\theta_i^{(j)}$, $i=an,kn,th$, $j=1,2$는 관절 모터 각도를 나타내며, an과 kn, th는 발목, 무릎, 대퇴부를 의미하고, 윗첨자 1과 2는 지지하는 다리와 들어올린 다리를 의미한다. 이러한 시상면 모터 각도는 요구되는 특정 패턴이 없으므로 2궤적 다항식으로 표현할 수 있다.

관상면 관절각 ϕ는 시상면 관절과는 달리 보행 안정성을 위해 0°에서 시작하여(직립상태, $q_0=0$), ZMP 안정도를 만족시키는 일정한 각도(ϕ_{ZMP})를 장시간 유지한 후, 다시 0°로 복귀해야 한다($q_f=0$). 이 경우 최소 세 개의 부분 궤적이 필요하며, 전체 궤적을 구성하는 계수는 다음과 같이 총 12개가 된다.

$$\boldsymbol{p_3}=[t_0\ t_{m1}^{\phi}\ t_{m2}^{\phi}\ t_f\ q_0\ q_{m1}\ q_{m2}\ q_f\ v_0 v_{m1} v_{m2}\ v_f] \tag{14}$$

식 (14)에서 ZMP 안정도에 중요한 역할을 하는 것은 두 중간 시간 t_{m1}^{ϕ}, t_{m2}^{ϕ}와 중간 각도 $q_{m1}=q_{m2}=\phi_{ZMP}$이며, 각 경계점에서 부드러운 각도 변화를 위해 $v_0=v_{m1}=v_{m2}=v_f=0$이라는 조건을 주면 결국 3개의 미지수 $t_{m1}^{\phi}, t_{m2}^{\phi}, \phi_{ZMP}$가 남고 이를 mDEAS로 최적화한다.

결과적으로 안정되면서도 최소한의 토크로 휴머노이드가 이족보행을 하기 위해 mDEAS로 최적화해야 할 전체 파라미터를 정리하면 다음과 같다.

$$X=[\theta_{an}^{(1)}(t_m),\dot{\theta}_{an}^{(1)}(t_m),\theta_{kn}^{(1)}(t_m),\dot{\theta}_{kn}^{(1)}(t_m),$$
$$\theta_{th}^{(2)}(t_m),\dot{\theta}_{th}^{(2)}(t_m),\theta_{kn}^{(2)}(t_m),\dot{\theta}_{kn}^{(2)}(t_m),$$
$$\theta_{an}^{(2)}(t_m),\dot{\theta}_{an}^{(2)}(t_m),t_{m1}^{\phi},t_{m2}^{\phi},\phi_{ZMP}]. \tag{15}$$

상기 파라미터 중 $\theta_{th}^{(1)}$과 $\dot{\theta}_{th}^{(1)}$이 없는 이유는, 상체를 직각으로 유지하는 조건 $\theta_1+\theta_2+\theta_3=\frac{\pi}{2}$에 식 (13)의 관계를 대입하면 $\theta_{th}^{(1)}$이 자동적으로 계산되기 때문이다. 기존의 연구[8]에서는 GA를 이용해서 θ_i의 최적 궤적을 직접적으로 구하려 했지만 본 연구에서는 최적화와 제어의 편의성을 위해 모터 궤적을 mDEAS로 최적화하는 점이 다르다.

마지막으로 mDEAS가 최소화해야 할 비용함수는 다음과 같이 표현된다.

$$J(X)=\gamma_c T_s\sum_{m=1}^{N}\tau(m)^T\tau(m)+P(X),$$
$$\tau(m)=[\tau_1(m)\,\tau_2(m)\cdots\tau_6(m)]^T,$$
$$P(X)=\gamma_1\left|\frac{x_6(T)-(x_0+S)}{S}\right|+\gamma_2\left|\frac{z_6(T)}{h_f}\right|+$$
$$\gamma_3(violation\ of\ swaying\ leg's\ minimum\ height)+$$
$$\gamma_4(violation\ of\ joint\ limit\ angles)+$$
$$\gamma_5(violation\ of\ ZMP\ stability). \tag{16}$$

식에서 T와 N은 시뮬레이션 시 보행 주기와 샘플 데이터 개수를 의미하며(샘플링 타임 $T_s=T/N$), S와 h_f는 보폭과 원하는 발 높이를 의미한다. 그리고 γ_c와 γ_i, $i=1,\cdots,5$는 최소화해야 할 토크의 가중치와 벌칙함수 $P(X)$의 각 항에 대한 가중치를 각각 의미한다. 벌칙함수의 각 항은 반드시 만족시켜야 하는 조건들이므로 γ_i, $i=1,\cdots,5$는 γ_c에 비해 상대적으로 수십, 수백배 큰 값을 설정해야 한다.

4. 실험 결과

본 논문에서는 표 1에서 주어진 사양의 소형 휴머노이드 모델에 대해 이족 보행을 simulation했다. 그리고 보행 조건은 다음과 같이 정했다.

- $S=0.1m$, $T=4\text{sec}$, $T_s=0.1\text{sec}$, $h_f=0.02m$

표 1 휴머노이드 모델의 링크 명세값
Table 1 Specification of links comprising the humanoid model

	l_1, l_5	l_2, l_4	l_3	l_6
길이(cm)	7.65	7.65	20.6	2.95
무게(g)	79.1	29.3	1246.4	153.8

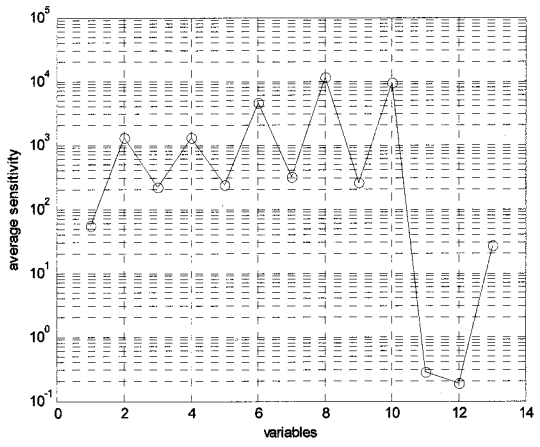

그림 6 변수별 평균 민감도
Fig. 6 Mean sensitivity per variable

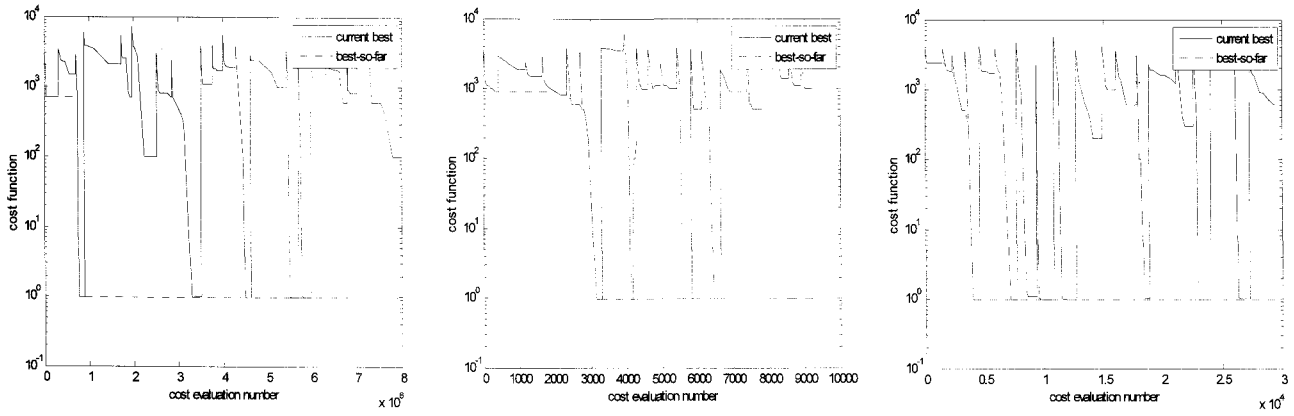

그림 7 변수별 평균 민감도

Fig. 7 Mean sensitivity per variable

본 논문에서는 mDEAS의 모듈을 만들기 위해 uDEAS를 통해 변수별 민감도를 계산했다. 이진 행렬의 행길이를 3(*initRowLen*=3)에서 10(*maxRowLen*=10)까지 증가시키며 20번의 multistart를 수행 후, 계산된 변수별 민감도의 평균값을 그림 6에 나타냈다. 이 민감도 값들을 이용해서 높은 순서대로 정렬 후 등분에 가깝게 분할하면 다음과 같은 eDEAS 모듈을 얻을 수 있다.

- module 1: $[x_6, x_8, x_{10}] = \left[\theta_{th}^{\dot{(2)}}(t_m), \theta_{kn}^{\dot{(2)}}(t_m), \theta_{an}^{\dot{(2)}}(t_m)\right]$
- module 2: $[x_2, x_4, x_7] = \left[\theta_{an}^{\dot{(1)}}(t_m), \theta_{kn}^{\dot{(1)}}(t_m), \theta_{kn}^{(2)}(t_m)\right]$
- module 3: $[x_1, x_3, x_5, x_9] = \left[\theta_{an}^{(1)}(t_m), \theta_{kn}^{(1)}(t_m), \theta_{th}^{(2)}(t_m), \theta_{an}^{(2)}(t_m)\right]$
- module 4: $[x_{11}, x_{12}, x_{13}] = \left[t_{m1}^{\phi}, t_{m2}^{\phi}, \phi_{ZMP}\right]$

위의 모듈은 1차적으로 가장 민감도가 높은 파라미터 순으로 거의 같은 크기로 등분한 것이다. 그러므로 가장 상위 모듈의 파라미터($\theta_{th}^{\dot{(2)}}(t_m), \theta_{kn}^{\dot{(2)}}(t_m), \theta_{an}^{\dot{(2)}}(t_m)$)들이 보행 시 총 토크에 가장 큰 영향을 미침을 알 수 있으며 이는 직관적인 견해와 일치한다. 그러나 이족보행에 있어서 우선적으로 고려되어야 할 것은 보행 안정도이다. 13개의 탐색 파라미터 중 마지막 세 파라미터 $t_{m1}^{\phi}, t_{m2}^{\phi}, \phi_{ZMP}$는 ZMP 안정도를 만족시키는 한, 비용함수 (16)에 거의 영향을 미치지 않는다. 그러므로, 비록 민감도는 가장 낮지만, mDEAS의 매 session에서 네 번째 모듈에 대해 가장 먼저 탐색하고, 민감도가 높은 순서로 다음과 같이 모듈별 탐색 순서를 정했다.

- module 별 탐색 순서: module 4 → module 1 → module 2 → module 3

본 논문에서는 mDEAS의 성능을 검증하기 위해 상기한 이족보행과 탐색 조건에 대해 eDEAS와 uDEAS를 이용하여 전역 최적해를 구하고 컴퓨터 연산 시간을 측정했다. 이때 탐색 조건은 mDEAS와 동일하다.

표 2 mDEAS의 탐색성능 비교

Table 2 Performance Comparison of mDEAS

	eDEAS	uDEAS	mDEAS
운용시간 (sec)	16687	34.2	**60.1**
전역최소값	0.9658	0.9675	**0.9650**

표 2는 세 가지의 DEAS 알고리듬에 대해 비교한 것으로 mDEAS가 uDEAS에 비해 2배의 시간 밖에 소요되지 않았지만 eDEAS의 전역최적해 보다 약간 더 좋은 해를 찾았음을 알 수 있다. 이론적으로는 eDEAS의 결과가 mDEAS보다 좋아야 하지만, 무작위로 선택된 시작점으로부터 제한된 횟수만큼 재시작을 행하기 때문에 다소 차이가 있을 수 있다.

그림 7은 세 가지 형태의 DEAS에 대해 20번의 multistart를 수행하는 동안 비용함수가 감소되는 양상을 보이고 있다. 그림에서 보이는 뾰족한 첨두는 재시작을 수행할 때 발생한다. 그림 7에서 알 수 있듯이 eDEAS의 경우도 초기 값에 따라 좋은 지역해를 찾지 못하는 경우가 많으며, uDEAS의 경우 계산량이 작다고 해서 eDEAS에 비해 탐색 성능이 크게 나쁜 것은 아님을 알 수 있다. mDEAS는 그림 7(c)에서 보듯이 계산량은 uDEAS에 가까우면서도 전역해는 eDEAS에 가깝거나 더 나은 탐색 성능을 보인다. 그림 8은 mDEAS로부터 얻은 전역 최적해 X^*를 이용해서 이족보행 simuation을 수행한 것을 보인다. 보행 분석에서 전체 관절이 부드럽게 회전하며, ZMP 안정도 조건과 다른 조건들을 모두 만족시킴을 확인할 수 있었다.

5. 결 론

본 논문에서는 eDEAS와 uDEAS의 장점을 융합한 mDEAS를 새롭게 개발하고, 이를 13개의 탐색 파라미터를 갖는 휴머노이드 이족보행 최적 관절 궤적 생성에 적용한 결과를 보였다. 개발된 mDEAS는 전체 파라미터를 적절히

모듈화하여 eDEAS를 수행함으로써, uDEAS에 가까운 탐색 속도를 가지면서도 eDEAS와 거의 같은 고정밀도 전역해를 찾아낼 수 있었다. 그러나 mDEAS를 수행하는 데 있어서 모듈을 적절히 분할하는 것이 중요하며, 본 논문에서는 보행 안정도에 큰 영향을 미치는 변수나 비용함수에 큰 영향을 미치는 변수를 모듈화해서 우선적으로 탐색하는 방법을 제안했다. 제안된 mDEAS는 SoC(system-on-a-chip)로 구현해서 향후 실시간 로봇 제어와 실시간 로봇 영상인식 등에 적용할 계획이다.

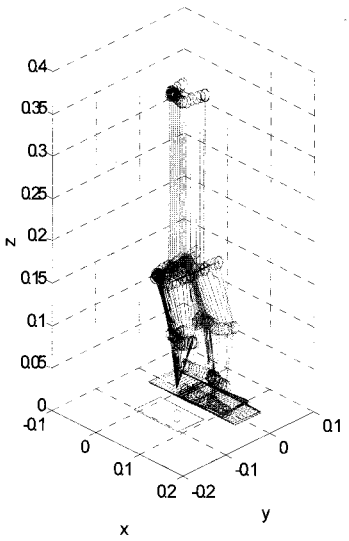

그림 8 최적 파라미터를 이용한 이족보행 simulation

Fig. 8 Biped walking simulation using the optimized trajectory parameters.

감사의 글

이 논문은 2006년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구로서 관련자에게 감사 드립니다(KRF-2006-331-D00666).

참 고 문 헌

[1] J. -W. Kim and S. W. Kim, "Numerical method for global optimization: dynamic encoding algorithm for searches (DEAS)," *IEE Proc.-Control Theory and Appl.*, vol. 151, no. 5, pp. 661-668. Sept. 2004.

[2] J. -W. Kim and S. W. Kim, "Parameter identification of induction motors using dynamic encoding algorithm for searches (DEAS)," *IEEE Trans. Energy Conversion*, vol. 20, no. 1, pp. 16-24, March 2005.

[3] 김태규, 김종욱, "DEAS를 이용한 변압기 코아의 최적설계," 대한전기학회 논문지, 56권, 6호, pp. 1055-1063, June 2007.

[4] J. -W. Kim and S. W. Kim, "A fast computational optimization method: univariate dynamic encoding algorithm for searches (uDEAS)," *IEICE Trans. Fundamentals*, vol. E90-A, no. 8, pp. 1679-1689, Aug. 2007.

[5] J. -W. Kim and S. W. Kim, "PID control design with exhaustive dynamic encoding algorithm for searches (eDEAS)," *International Journal of Control, Automation, and Systems*, vol. 5, no. 6, pp. 691-700, Dec. 2007.

[6] J. -W. Kim, T. Kim, Y. Park, and S. W. Kim, "On-load motor parameter identification using univariate dynamic encoding algorithm for searches," *IEEE Trans. on Energy Conversion*, vol. 23, no. 3, pp. 804-813, 2008.

[7] D. E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison Wesley, 1989.

[8] 최무성, 권오홍, 강민성, 박종현, "유전자 알고리즘을 이용한 이족 보행 로봇의 최적 설계 및 최적 보행 궤적 생성", 제어 자동화·시스템공학 논문지 제10권 제9호, 2004.

[9] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, John Wiley & Sons, Inc. 2006.

[10] T. Kim, E. Kim, and J. -W. Kim, "Development of a humanoid walking command system using a wireless haptic controller", *International Conference on Control, Automation and System*, pp.1178-1183, Seoul, Korea, Oct. 2008.

저 자 소 개

김 은 수 (金 銀 宿)

1983년 1월 26일생. 2008년 동아대 전자공학과 졸업. 2008년~현재 동 대학원 전자공학과 석사과정

Tel : 051-200-5579

E-mail : tacctics@hotmail.com

김 조 환 (金 祚 煥)

1983년 10월 29일생. 2002년~현재 동아대 전자공학과 학사과정 재학중.

Tel : 051-200-5579

E-mail : kimhades@hotmail.com

김 종 욱 (金 鍾 旭)

1970년 10월 24일생. 1998년 포항공대 전자전기공학과 졸업. 2000년 동 대학원 전자전기공학과 졸업(석사). 2004년 동 대학원 전자전기공학과 졸업(박사). 2004년~2006년 포스코 기술연구소 전기강판연구그룹 연구원, 2006년~현재 동아대학교 전자공학과 조교수

Tel : 051-200-7714

Fax : 051-200-7712

E-mail : kjwook@dau.ac.kr