

유전자 알고리즘을 이용한 Fuzzy Data Fitting

김성용^o 한준희

포항공과대학교 전자계산학과 컴퓨터비전 연구실

Fuzzy Data Fitting with Genetic Algorithm

S.Y. Kim Joon H. Han

Computer Vision Lab, Dept. of Comp. Sci. & Eng., POSTECH

요 약

Noise가 있는 data에서 shape나 parameter를 찾을때 일반적으로 Hough transform이나 regression을 적용한다. Hough transform은 parameter space의 차수가 커지면 memory 문제가 존재하며, regression 모델은 한 개의 변수를 다른 변수의 함수로 가정하여 error를 최소화하여 데이터중 1 set의 parameter만 존재한다는 가정을 하여야 하는 문제점이 있다. 본 논문에서는 이러한 두 방법의 단점들을 보완하며, Fuzzy개념을 도입한 data fitting 방법을 제안하였다. 이 문제는 genetic algorithm을 도입하여 data를 Fuzzy membership을 갖는 것으로 가정한 최적화 문제로 해결하였다. 직선과 평면에 대한 실험 결과를 보인다.

1. 서론

주어진 data set에 대하여 이를 잘 나타낼 수 있는 도형의 식을 찾아야 할 경우가 많다. 컴퓨터 비전에서 일반적으로 영상에서 직선이나 곡선을 찾기 위한 방법으로 여러 가지가 있으며 가장 대표적인 것으로 Hough transform이 있다

본 논문에서는 이를 유전자 알고리즘을 이용하여 접근해보고자 한다[1, 2]. 앞으로 기술할 내용은 다음과 같다. 먼저 유전자 알고리즘의 기본적인 개념과 이를 어떻게 적용할 것인가에 대하여 설명하고 기존의 방법과 다른 점을 언급한 다음 실제로 직선과 평면에 대하여 실험을 해보도록 한다.

2. 유전자 알고리즘의 적용

유전자 알고리즘은 형질의 유전과 적자생존등 유전자와 관련된 자연 현상을 기초로 한 확률론적 알고리즘이다 주로 optimization에 이용되며 좋은 결과를 내는 잠재적 해는 재생산되고 나쁜 결과를 내는 잠재적 해는 도태시키는 것이 유전자 알고리즘의 핵심적인 개념이다.

2.1 유전자 시스템의 구조

특정 문제를 해결하기 위한 유전자 시스템은 여러 가지 구성요소를 가지게 된다. 이를 정리하면 다음과 같다.

이름	내용
유전자(chromosome) 초기화 루틴	해의 유전자적 표현(binary string) 해의 초기치 생성 루틴
evaluation function	결과를 계산하는 함수
genetic operator	유전자 재구성 루틴(교차, 돌연변이)
various parameter	시스템이 사용하는 파라미터들

2.2 유전자 알고리즘의 수행 과정

먼저 특정 문제에 대한 해를 유전자의 형태로 변환한 후 초기화 과정을 수행함으로써 잠재적 해의 population $P(t) = \{x_1^t, \dots, x_n^t\}$ 가 생성된다. 그런 다음 각각의 해 x_i^t 를 evaluation function으로 넘겨서 계산하여 각 해에 대한 결과를 산출한다.

이 값들을 기반으로 하여 좋은 결과를 내는 해들을 선택하고 상대적으로 나쁜 결과를 내는 해는 제거하여 새로운 population을 생성한 다음, 교차와 돌연변이등의 genetic operator를 적용, 재구성을 함으로써 이전의 population과는 다른 새로운 population $P(t+1)$ 을 생성하게 된다. 앞에서 설명한 과정을 원하는 결과가 나올때까지 반복함으로써 global optimam을 구한다.

2.3 유전자의 선택과 재구성

선택과정을 수행하기 위해서 spinning roulette wheel 방식을 사용한다. 먼저 각 염색체에 대하여 함수값을 계산한 후 이를 전부 합한 값 F 에 대하여 각 염색체가 가지는 누적확률을 계산한다. 수식으로 나타내면 다음과 같다.

$$p_i = eval(v_i)/F$$

$$q_i = \sum_{j=1}^i p_j$$

그런 다음 각 염색체에 대하여 0부터 1사이의 난수를 발생시킨 후 그보다 큰 누적확률을 가지는 첫번째 염색체를 선택한다. 이렇게 하여 생성된 새로운 population에 대하여 다시 재구성 과정을 수행한다. 먼저 교차 과정에서는 각 염색체에 대하여 0부터 1사이의 난수를 발생시킨 후 그 값이 교차 확률 p_c 보다 작을 경우 교차를 수행할 염색체로써 선택한다. 그 다음 두 개씩 짝을 지어서 교차를 시킨다. 돌연변이도 같은 방법이지만, 각 염색체의 각 원소에 대하여 돌연변이 확률 p_m 을 기초로 수행한다.

3. Hough transform과의 비교

Hough transform은 영상에서 직선이나 원등의 도형을 찾고자 할 때 많이 이용된다[3]. 직선의 경우 먼저 fitting하려고 하는 점(그림1(a))들에 대하여 적용하면 그림1.(b)와 같은 결과를 얻게 된다.

하지만 그림1.(b)와 같이 가장 많은 계수의 점들을 지나는 여러 개의 직선보다는 주어진 점들을 가장 가깝게 지나는 하나의 직선을 찾아야 할 때가 있다. 이를 위하여 Fuzzy개념을 도입한

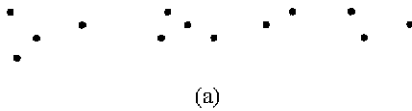
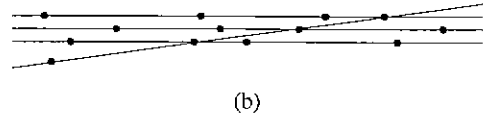


그림 1: (a) 직선을 찾기 위한 점들.



(b) Hough transform을 이용하면 4개의 직선이 발견된다.

것이 Fuzzy Hough transform이다[4, 5]. 이는 각 점을 중심으로 하여 일정한 크기의 반지름을 가지는 원을 가정한다. 직선이 원의 범위 내에 들어올 경우 점과 직선사이의 거리를 멤버십 함수로 넘겨 계산하는 과정을 모든 점에 대하여 실행한 후 그 값이 가장 큰 직선을 선택함으로써 그림2와 같이 원하는 직선을 찾게 된다. 하지만, Hough transform과 Fuzzy Hough transform은 공통적으로, 다차원의 parameter space의 경우 memory 문제와 더불어 계산량이 아주 늘어나므로 적용이 그다지 쉽지 않다. 이런 문제점을 개선하기 위하여 유전자 알고리즘을 이용한다.

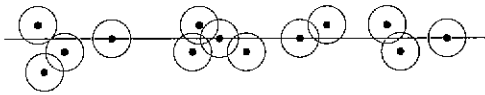


그림 2. Fuzzy Hough transform을 이용하면 1개의 직선을 찾는다

4. Data Fitting

이번 장에서는 data fitting에 유전자 알고리즘이 어떻게 이용되는지에 대해 설명하도록 한다. 먼저 직선의 경우 수직선에 가까운 직선을 표현하기 위해서 일반식 $ax + by = c$ 를 다음과 같이 변형한다

$$\frac{a}{\sqrt{a^2 + b^2}} + \frac{b}{\sqrt{a^2 + b^2}} = \frac{c}{\sqrt{a^2 + b^2}}$$

이를 치환하여 표현하면 $g_x x + g_y y = d$, 즉 $g \cdot x = d$ 과 같이 벡터의 내적으로 표현된다. 즉 단위벡터 $g(g_x, g_y)$ 와 d 로 2차원 상에서 하나의 직선을 표시하며 또한 하나의 유전자 단위로 취급한다

가장 먼저 해야할 일은 초기화 과정이다. 이 때 (g_x, g_y) 가 단위벡터이므로, 0~360사이의 값을 난수 발생시켜 각각 \cos 와 \sin 을 취함으로써 얻는다 d 는 일정한 범위내의 값을 난수 발생시켜 얻는다. 이와 같이 초기 population을 생성하고 나면, 결과를 계산하기 위하여 먼저 점과 직선 사이의 수직거리 r_i 를 계산한다.

$$r_i = \begin{pmatrix} g_x \\ g_y \end{pmatrix} \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix} - d \quad (1)$$

그리고 위의 식에서 얻어진 r_i 를 evaluation하는 식을 다음과 같이 정의한다

$$v(r_i) = \begin{cases} 0 & r > R, \\ 1 - (\frac{r}{R})^2 & r \leq R. \end{cases} \quad (2)$$

위의 식에서 알 수 있듯이 한 점의 범위내로 직선이 지나갈 경우 계산하고, 그렇지 않을 경우 계산을 하지 않는다. 또한 점이 직선에 가까울수록 더 큰 값을 가지게 된다. 이를 모든 점에 대해서 수행하면 결국 다음의 식을 최대로 하는 (g_x, g_y) 와 d 를 찾게 된다.

$$E = \sum_{i=0}^n v(r_i) \quad (3)$$

이와 같은 계산을 바탕으로 하여 새로운 population을 생성하기 위하여 유전자의 선택과 재구성 과정이 필요하다. 이는 앞서 설명한 바와 같다

이거시 한가지 깊고 넓어갈 점은 바로 Least Squared Method와의 차이점이다. Least Squared Method에선 점과 직선사이의 거리를 단순히 Y축과 평행한 방향, 즉 Y값의 차이에 의하여 계산하게 되므로, 직선의 기울기가 클 경우 부정확한 결과를 낼 수 있다[3]. 반면에 유전자 알고리즘을 이용한 방법에서는 식(1)에서와 같이 점과 직선사이의 수직거리를 이용하므로 보다 정확한 결과를 도출하게 된다. 또한 여러 개의 직선을 찾고자 할 때에 유전자 알고리즘을 이용할 경우에는 식(2)에서와 같이 일정 범위내의 점만을 fitting하고 난 다음 그 점들을 모두 제거하고 나머지 점들을 이용하여 다시 알고리즘을 적용하면 되므로, 좋은 결과를 내게 된다.

평면의 경우에도 마찬가지이다. 직선과의 차이는 단지 유전자의 구성요소인데, 방향벡터가 3차원 벡터가 되므로 그림3과 같은 원리에 의하여 다음과 같은 식들이 성립된다.

$$\begin{aligned} g_x &= \cos\alpha \cdot \cos\beta \\ g_y &= \sin\alpha \cdot \cos\beta \\ g_z &= \sin\beta \end{aligned}$$

즉, $g(g_x, g_y, g_z)$ 와 d 로 3차원 상의 평면을 표시하며 이를 하나의 유전자 단위로 취급한다. 나머지는 직선과 동일하다

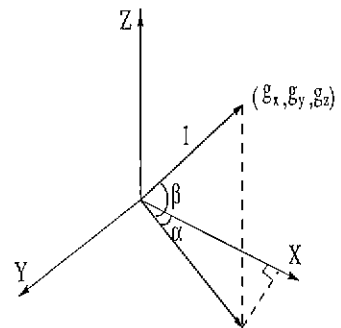


그림 3 3차원 상에서의 벡터.

5. 실험 및 결과

실험은 앞 장에서 설명한 직선과 평면에 대하여 같은 방법으로 행하였다. 실험에 사용된 파라미터는 다음과 같다.

pop_size	p_c	p_m	radius
20	0.25	0.02	2

pop_size란 population의 크기, 다시 말해서 잠재적 해의 개수를 나타낸다. p_c 는 교차 확률, p_m 은 돌연변이 확률을 나타낸다. 그리고 radius는 fitting시 계산이 적용되는 유효범위를 나타낸다. 실험에 사용된 data는 직선식을 입력받은 다음 임의로 생성한 직선위의 점들에 gaussian noise를 더하여 만들었으며, 그림4에 나와 있다. 평면에 대해서도 같은 방법으로 실험을 하였으며, 그림5에 그 결과가 나와 있다.

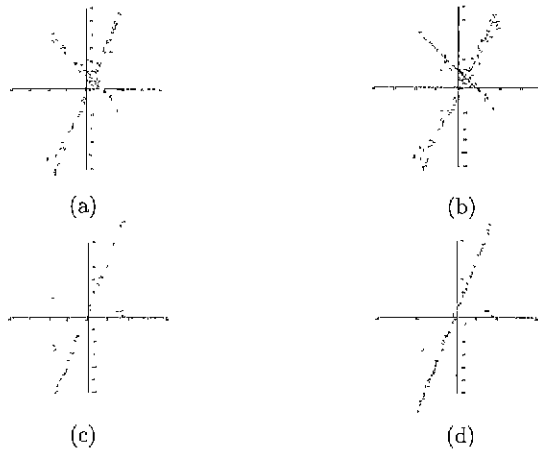


그림 4: 유전자 알고리즘을 line fitting에 적용한 실험 결과. (a),(c)는 fitting하고자 하는 점들이며, (b),(d)는 결과이다.

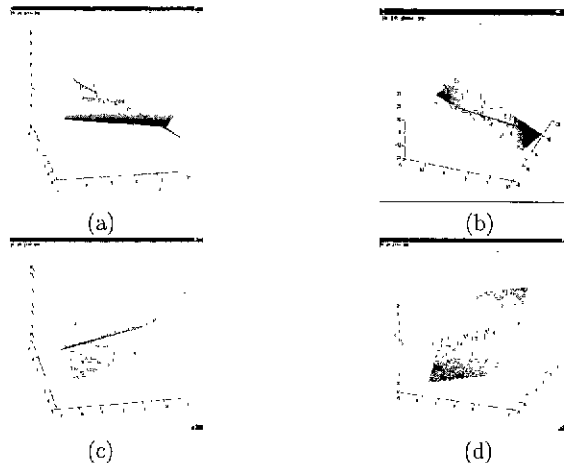


그림 5: 유전자 알고리즘을 surface fitting에 적용한 실험 결과. (a),(b)와 (c),(d)는 각각 같은 그림을 여러 각도에서 잡은 것이다.

6. 결론 및 토의

Fuzzy개념을 도입한 유전자 알고리즘을 적용할 경우 Hough transform이나 Least Squared Method와 비교하여 볼때 비슷하거나 더 좋은 결과를 낸다고 여겨지며, Hough transform이나 regression모델의 단점을 보완하여 여러 개의 식을 찾거나 다차원으로의 확장이 용이하다는 장점을 가지고 있다. 실험과정에서 나타나는 문제점은 다음과 같다. 먼저 유전자 알고리즘에서 난수 발생이 많이 사용된다는 점때문에, 결과적으로 같은 입력에 대해서도 다른 결과를 내는 경우가 있다. 이 외에도 global optimam에 대한 수렴조건을 명확히 제시하지 못했으며, 어느 정도의 결과가 나왔을 경우 일정한 횟수를 반복 수행시켜 결과를 도출한 관계로, 정확한 값을 내지 못하는 경우도 간혹 있다. 이는 반드시 개선되어야 할 점이다.

Acknowledgement: 본 연구는 공정산업의 지능자동화 연구센터(ERC)와 한국과학재단으로부터 (부분적) 지원을 받았음.

참고문헌

- [1] Z. Michalewicz, *Genetic Algorithms+Data Structures=Evolution Programs*. Springer-Verlag, 2nd, extended ed.
- [2] P. J. Angeline, K. E. Kinnear, and Jr., *Advances in Genetic Programming*. The MIT Press, 2nd ed.
- [3] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision Volume I*. Addison-Wesley, 1st ed.
- [4] J. H. Han, L. T. Koczy, and T. Poston, "Fuzzy hough transform," *Pattern Recognition Letters*, vol. 15, pp. 649-658, 1994.
- [5] J. M. Keller, J. G. M. R. Gray, and Jr., "A fuzzy k-nearest neighbor algorithms," *IEEE Trans Syst, Man Cybern.*, vol. SMC-15, pp 580-585, 1985.