거리정보를 이용해 위치를 추적하는 알고리즘과 해당 알고리즘의 최적화 방향성

Myung Woo Song, erados@korea.ac.kr
Dept. of Electrical Engineering, Korea University, Seoul, South Korea 02841

소스 코드 및 시뮬레이션

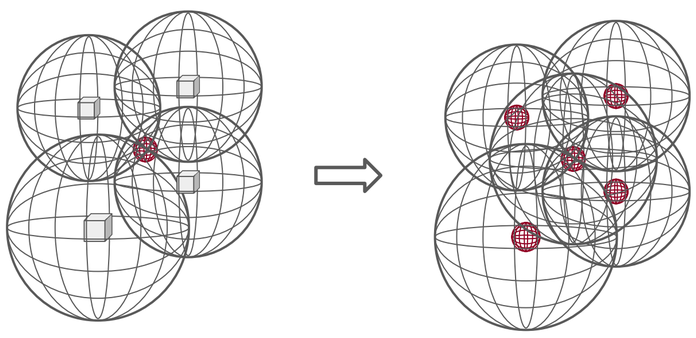
요약

본 연구에서는 GPS 의 거리정보 기반의 위치추적 시스템을 응용, 간소화하여 사물의 변형이나 움직임을 거리 정보를 통해 추적하는 알고리즘을 제안하고 알고리즘의 성능에 영향을 미치는 변수를 조정해가며 해당 알고리즘의 최적화 방법에 대한 방향성을 실험적으로 확인했다.

서론

GPS 는 4개 이상의 위성으로부터 시간 정보가 담긴 신호를 받아 거리 정보를 역산하여 기준국과의 변위를 측정하고 자신의 위치를 알 수 있는 위성항법 시스템이다. 이를 확장하여 생각해보면 서로의 거리정보를 알 수 있는 장치가 다수 있다고 가정했을때, 위성항법 시스템을 응용하여 서로간의 상대적인 위치를 정확히 추측해낼 수 있다는 것을 알 수 있다.

본 연구에서는 모든 장치가 시간 정보의 송신자이면서 수신자인 경우를 생각하여 각 장치가 서로간의 실제 거리를 알게 되었을때 장치의 위치를 추정하는 알고리즘을 구현하였다. 그리고 특정 개수의 장치가 존재할 때, 각 장치가 몇 개의 장치와 정보를 교환하는 것이 가장 효율적인지를 실험적으로 확인했다.



알고리즘

거리정보를 가지고 위치를 알아내는 알고리즘은 크게 두 가지로 구분할 수 있다. 먼저, 3차원에서 어떤 점에대해 거리가 일정한 점을 모두 모으면 구가 되므로 n-1 개의 장치에 대해 (n-1)! 개의 3차 연립 방정식을 푸는 방법이다. 이 때, 3 * n 개의 변수가 주어지므로 n >= 4 일 때 해를 구할 수 있음을 알 수 있다. 그러나 이 방법은 지나치게 어려우므로 보다 쉬운 방법인 경사하강법을 이용하여 알고리즘을 만들었다.

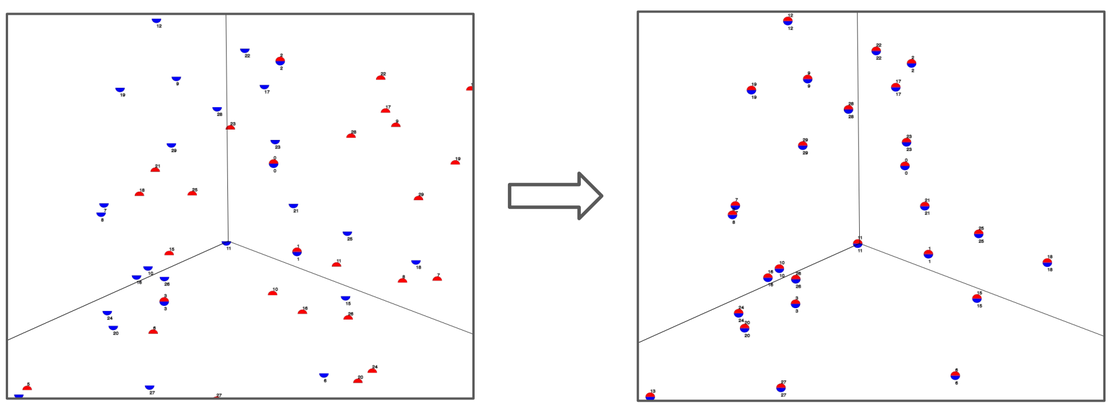
경사하강법을 위해 모든 지점들 간의 실제 거리와 추정 거리의 차의 제곱을 오차함수로 정하였다. 컴퓨터의 연산력에 영향을 미치는 제곱근을 수식에서 제외하기위해 거리의 제곱을 사용하였으며 수식으로 나타내면 다음과 같다.

$$\mathbf{x}_{n+1,g} = \mathbf{x}_{n,g} - \gamma_n \nabla F(\mathbf{x}_{n,g}), n > 0$$

$$F(\mathbf{x}_{n,g}) = \sum_{i \neq n} \left[\left\{ (x_{n,g} - x_{i,g})^2 + (y_{n,g} - y_{i,g})^2 + (z_{n,g} - z_{i,g})^2 \right\} - \left\{ (x_{n,a} - x_{i,a})^2 + (y_{n,a} - y_{i,a})^2 + (z_{n,a} - z_{i,a})^2 \right\} \right]^2$$

$$\nabla F(\mathbf{x}_{n,g}) = 4 * \sum_{i \neq n} \left[(\mathbf{x}_{n,g} - \mathbf{x}_{i,g}) \left(\left\{ (x_{n,g} - x_{i,g})^2 + (y_{n,g} - y_{i,g})^2 + (z_{n,g} - z_{i,g})^2 \right\} - \left\{ (x_{n,a} - x_{i,a})^2 + (y_{n,a} - y_{i,a})^2 + (z_{n,a} - z_{i,a})^2 \right\} \right) \right]$$

해당 알고리즘을 HTML5 의 Canvas 태그와 Javascript 를 사용하여 시뮬레이션하였으며 그 결과 아래 그림과 같이 각 장치가 자신의 실제 위치를 추정하는 것을 시각적으로 확인할 수 있었다. 다만 상대적인 좌표가 아닌 실제 좌표를 추정하기 위해 기준점이 되는 4 개의 장치는 자신의 실제 위치를 알고 있다고 가정하였다.



* 파란색 반원 : 실제 위치, 빨간색 반원 : 추정한 위치

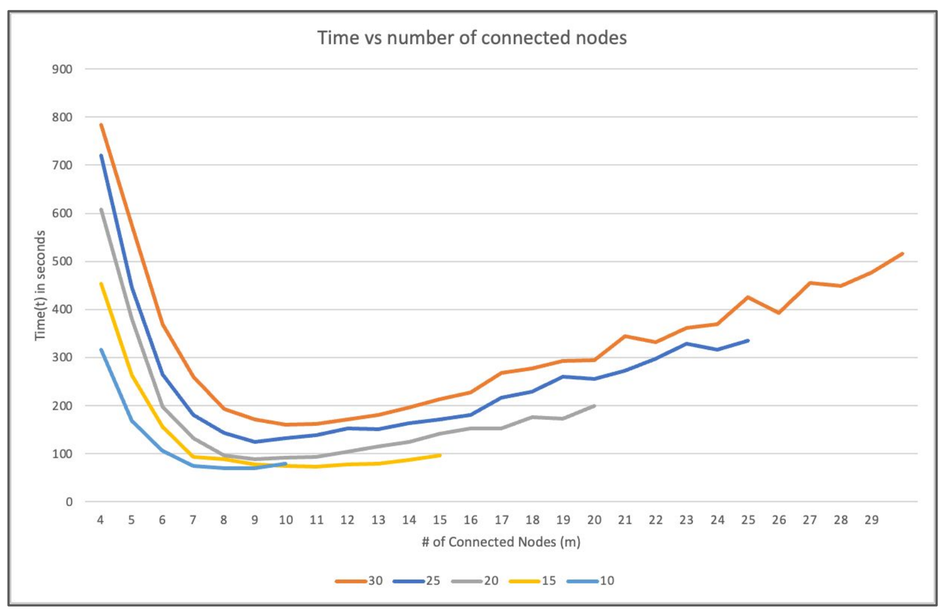
실험방법

각 점은 4개의 점과의 거리를 알 때에 상대적인 위치 정보를 정확히 알 수 있으므로 최소 4개의 점과의 거리를 알아야 한다는 사실은 명확하다. 그리고 거리 정보가 하나 추가될 때마다 연산은 점의 총 갯수만큼 증가한다. 그러나 연산 자원이 한정되어 있는 만큼 연산 자원이 모자라는 순간 거리정보가 추가될 수록 알고리즘이 느려질 것이다. 따라서 컴퓨팅 자원이 많을 수록 거리 정보는 많으면 좋을 것이다. 그러나 그 추세가 수렴할 것으로 예상되고 모든 점을 연결 시키기 전 모든 점을 연결 시킨 것과 유사한 결과를 내는 지점이 있을 것이며 이 지점이 연산을 최적화할 수 있는 지점이다.

알고리즘 최적화를 실험적으로 수행하기 위하여 총 장치의 수 n 을 10, 15, 20, 25, 30 으로 변화해가며 각 장치가 몇 개(m)의 다른 장치와의 거리 정보를 가지고 있는지를 변인으로 삼아 1000 개의 랜덤한 배치에 대해 실제 위치를 추측해내는 시간을 성능 분석의 기준치로 삼아 그래프를 그렸다. 각 장치가 연결되는 장치는 랜덤으로 설정하였다. 또한, 알고리즘을 특정 횟수 반복할 때까지 오차값이 특정 범위 안에 들지 않으면 추정 위치를 재배치하여 다시 알고리즘을 수행하였고 Local Minima 가 많은 배치에서 시간이 지나치게 많이 걸리는 문제를 해결하기 위해 5번의 실패가 있는 경우 장치의 실제 위치를 랜덤하게 재배치했다. 실험 장비는 M1 프로세서를 탑재한 애플 맥북 프로를 사용하였다.

결과

실험 결과 (n, m) 이 (10,7), (15,10), (20, 8), (25, 8), (30, 9) 일 때 가장 성능이 좋았다.



결론

먼저 기대와 달리 프로세서의 연산적이 떨어져 시간이 수렴하는 모습을 확인하기 어려웠다. 다만 n = 10 과 n = 15 의 경우 약 t = 90 에 알고리즘이 수렴하는 것을 확인할 수 있으며 이를 통해 실제로 알고리즘을 특정 효율(t/n)으로 최적화할 수 있는 지점이 m < n 에 존재한다는 것을 알 수 있다.

실험 결과에서 확인할 수 있듯이 정보를 공유하는 장치의 수가 많아질 때 일정 수준까지는 급격하게 속도가 빨라지다가 이후 다시 속도가 느려진다. m 이 7 이하일 때의 그래프를 보았을 때 해당 구간에서의 접선의 x 절편이 n 값보다 작다는 점에서 실험환경과 관계없이 연결된 장치의 수가 많아질 수록 그래프의 기울기가 완만해진다는 것을 알 수 있다. 즉 실험환경에 상관 없이 m 이 늘어날수록 효율성이 떨어진다. 따라서 해당 알고리즘을 효율적으로 최적화하기 위해서는 m=4 부터 시작하여 m 을 늘려가면서 계산하는 것이 좋으며 연산을 병렬화 할 수 있으므로 각 장치에서 자신의 위치를 추정하게 한다면 속도가 비약적으로 증가할 것이다.

실험적으로 거리 정보를 통해 위치를 추적할 수 있다는 것을 확인했지만, 소형 물체의 움직임을 추적하려면 장치간의 거리가 너무 가까워 시간 정확도가 상당히 커야한다는 한계가 있다. 이 때문에 거리를 구하는 경우 비용적인 측면에서 속도가 빠른 전자기파보다는 초음파를 사용하는 경우가 많으나 초음파는 지향성이 있어 해당 알고리즘에 적용하기 어렵다. 따라서 해당 알고리즘을 현실적으로 활용하는데에는 현재로서 무리가 있다.

향후 이러한 한계점을 극복할 수 있다면 실내측위, 모션 캡쳐, IoT, 자율주행 등 많은 분야에 활용될 수 있을 것이다.