Dilated Convolution:

基于进化算法的脉动阵列快速硬核布局方法

张年崧 16308149

2020 年 5 月 10 日

School of Electronics and Information Technology, Sun Yat-sen University

Table of Contents

Introduction

Dilated Convolution

Assuming a source is emitting signal in all directions and an array of sensors are picking up the signal, it will arrive at different sensors at different time.

TDOA : *Time Difference Of Arrival*

Then the location of the source can be estimated by measuring the delay of signal arriving at different sensor.

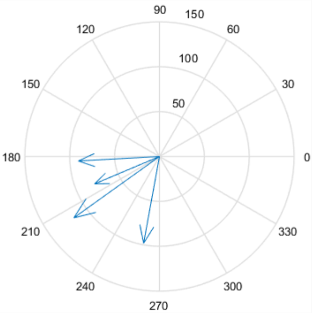


Figure 1: Sensors and source.

Position Fix by TDOA: Noise

Vector of TDOA:

$$\mathbf{d} = \begin{bmatrix} d_2 - d_1 & d_3 - d_1 & \cdots \end{bmatrix}^T$$

However, 100% precise measurement of TDOA is not possible. Noise is always present.

Naturally, the noise in TDOA is a *multivariate Gaussian distribution* centered at true value, with covariance given by matrix:

$$\mathbf{Q} = \sigma^2 \begin{bmatrix} 1 & 0.5 & \cdots & 0.5 \\ 0.5 & 1 & \cdots & 0.5 \\ \vdots & \vdots & \ddots & \vdots \\ 0.5 & 0.5 & \cdots & 1 \end{bmatrix}$$

where σ^2 is the noise power.

3-Dimensional Proposed Method

3D Proposed Method

- By assuming x, y, z and r_1 are independent, the non-linear equations can be reduced into linear ones.

3D Proposed Method

- By assuming x, y, z and r_1 are independent, the non-linear equations can be reduced into linear ones.
- Given different situations, use *Maximum Likelihood* (ML) estimator to solve the linear equations.

3D Proposed Method

- By assuming x, y, z and r_1 are independent, the non-linear equations can be reduced into linear ones.
- Given different situations, use *Maximum Likelihood* (ML) estimator to solve the linear equations.
- Incorporate the dependent relationship back into the solution (if necessary) to get more accurate solution.

3D Proposed Method: Work-flow

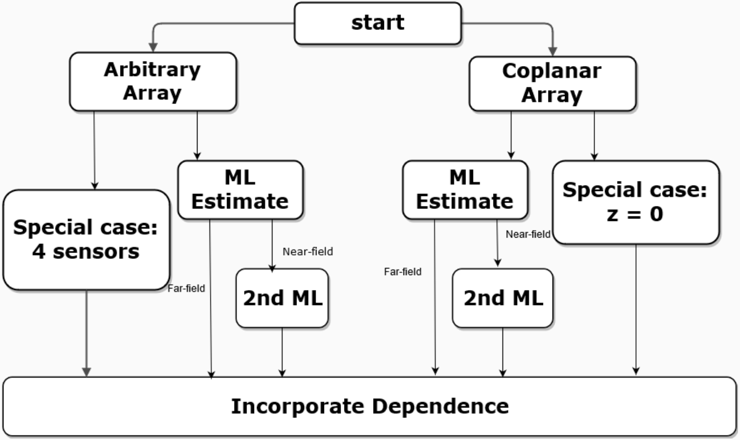


Figure 2: Flowchart of the proposed method in 3D.

3D Proposed Method: Arbitrary Array

"Arbitrary" refers to that sensors are arranged in a non-linear manner, thus avoiding singularity during computation.

- With 4 sensors ($M = 4$), system is not overdetermined.

$$\mathbf{z}_p = -\begin{bmatrix} x_{2,1} & y_{2,1} & z_{2,1} \\ x_{3,1} & y_{3,1} & z_{3,1} \\ x_{4,1} & y_{4,1} & z_{4,1} \end{bmatrix}^{-1} \times \left\{ \begin{bmatrix} r_{2,1} \\ r_{3,1} \\ r_{4,1} \end{bmatrix} r_1 + \frac{1}{2} \begin{bmatrix} r_{2,1}^2 - K_2 + K_1 \\ r_{3,1}^2 - K_3 + K_1 \\ r_{4,1}^2 - K_4 + K_1 \end{bmatrix} \right\}$$

$$r_1^2 = K_1 - 2 \begin{bmatrix} x_1 & y_1 & z_1 \end{bmatrix} \mathbf{z}_p + \mathbf{z}_p^T \mathbf{z}_p$$

  where $K_i = x_i^2 + y_i^2 + z_i^2, \mathbf{z}_p = \begin{bmatrix} x & y & z \end{bmatrix}^T$.

- Therefore, the solution can be found simply by
  1. solving a quadratic equation of r_1;
  2. insert r_1 back and solve the linear equation groups of x and y.

Proposed Method: Arbitrary Array (cont'd)

- With more than 4 sensors, the system is overdetermined.
- Here we denote:

$$\mathsf{h} = \frac{1}{2}\begin{bmatrix} r_{2,1}^2 - K_2 + K_1 \\ r_{3,1}^2 - K_3 + K_1 \\ \vdots \\ r_{M,1}^2 - K_M + K_1 \end{bmatrix}$$

$$\mathsf{G}_a = -\begin{bmatrix} x_{2,1} & y_{2,1} & z_{2,1} & r_{2,1} \\ x_{3,1} & y_{3,1} & z_{3,1} & r_{3,1} \\ \vdots & \vdots & \vdots & \\ x_{M,1} & y_{M,1} & z_{M,1} & r_{M,1} \end{bmatrix}$$

$$\mathsf{B} = \mathsf{diag}\{r_2^0, r_3^0, \cdots, r_M^0\}$$

- Additional dimension reflects on the change in G_a.

Proposed Method: Dependency of Arbitrary Array

- The next step is to consider the dependency between x, y and r_1.
- Again using ML, we have

$$\mathbf{z}_a^{'} = (\mathbf{G}_a^{'T}\Psi^{'-1}\mathbf{G}_a^{'})^{-1}\mathbf{G}_a^{'T}\Psi^{'-1}\mathbf{h}^{'}$$

where

$$\mathbf{h}^{'} = \begin{bmatrix} (x-x_1)^2 \\ (y-y_1)^2 \\ (z-z_1)^2 \\ r_1^2 \end{bmatrix}, \quad \mathbf{G}_a^{'} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

and

$$\Psi^{'} = 4\mathbf{B}^{'}(\mathbf{G}_a^{0T}\Psi^{-1}\mathbf{G}_a^{0})^{-1}\mathbf{B}^{'}, \quad \mathbf{B}^{'} = \mathbf{diag}\{x-x_1, y-y_1, z-z_1, r_1^0\}$$

Proposed Method: Solution of Arbitrary Array

- Similar simplification can be made when the source is far away, by substituting Ψ with $\mathbf{Q}$.
- After acquiring $\mathbf{z}_a'$, we have

$$\mathbf{z}_a' = \begin{bmatrix} (x^0 - x_1)^2 \\ (y^0 - y_1)^2 \\ (z^0 - z_1)^2 \end{bmatrix}$$

- Then x^0, y^0, z^0 can be calculated by taking square roots of the positive values.
- Interested solution will be chosen among different sign combinations yield by square root.

Proposed Method: Coplanar Array

If the sensors are coplanar, i.e. sit on the same plane, then the positions can be described by

$$z = ax + by + c$$

Here the procedure used for *Arbitrary Array* would fail due to matrix singularity.

However, a similar solution can be developed with slight modifications.

Coplanar array corresponds to linear array in 2D scenario.

Proposed Method: Coplanar Array (cont'd)

- Taking the relationship $z = ax + by + c$, rewrite $\mathbf{z}_a$ and $\mathbf{G}_a$ as

$$\mathbf{z}_l = \begin{bmatrix} x + az \\ y + bz \\ r_1 \end{bmatrix}, \quad \mathbf{G}_l = -\begin{bmatrix} x_{2,1} & y_{2,1} & r_{2,1} \\ x_{3,1} & y_{3,1} & r_{3,1} \\ \vdots & \vdots & \\ x_{M,1} & y_{M,1} & r_{M,1} \end{bmatrix}$$

- We can have something very similar to that in arbitrary array

$$\mathbf{z}_l = (\mathbf{G}_l^T \Psi^{-1} \mathbf{G}_l)^{-1} \mathbf{G}_l^T \Psi^{-1} \mathbf{h}$$

  where Ψ and $\mathbf{h}$ are the same as in arbitrary Array.
- Similar approach of taking $\mathbf{Q}$ for Ψ and use of iteration can be applied here.

Proposed Method: Solution of Coplanar Array

We now have $\mathbf{z}_l = \begin{bmatrix} x + az \\ y + bz \\ r_1 \end{bmatrix}$, denoted as $\mathbf{z}_l = \begin{bmatrix} w \\ v \\ r_1 \end{bmatrix}$.

- x, y, z can be solved through quadratic equation

$$z = \frac{-E \pm \sqrt{E^2 - 4AC}}{2A}, \quad x = w - az, \quad y = v - bz$$

  where

$$A = 1 + a^2 + b^2$$

$$E = -2aw - 2bv - 2z_1 + 2x_1 + 2by_1$$

$$C = w^2 + v^2 - 2x_1 w - 2y_1 v + K_1 - r_1^2$$

- When $a = 0$ and $b = 0$, the solution becomes

$$z = \pm\sqrt{r_1^2 - (w - x_1)^2 - (v - y_1)^2} + z_1, \quad x = w, \quad y = v$$

Taylor-series Method

Taylor-Series Method

We are given:

$$r_{i,1} = cd_{i,1} = r_i - r_1$$

Linearize above equation by Taylor-series expansion and then solve iteratively:

- Compute position deviation
- Add position deviation to initial guess
- Solve again until deviation is considerably small

Convergence is not guaranteed

Taylor-series Method

The position deviation is computed by:

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = (\mathbf{G}_t^T \mathbf{Q}^{-1} \mathbf{G}_t)^{-1} \mathbf{G}_t^T \mathbf{Q}^{-1} \mathbf{h}_t$$

where h_t and G_t are given as follows

$$\mathbf{h}_t = \begin{bmatrix} r_{2,1} - (r_2 - r_1) \\ r_{3,1} - (r_3 - r_1) \\ \\ r_{M,1} - (r_M - r_1) \end{bmatrix} \tag{1}$$

$$\mathbf{G}_t = \begin{bmatrix} (x_1 - x)/r_1 - (x_2 - x)/r_2 & (y_1 - y)/r_1 - (y_2 - y)/r_2 \\ (x_1 - x)/r_1 - (x_2 - x)/r_3 & (y_1 - y)/r_1 - (y_3 - y)/r_3 \\ \\ (x_1 - x)/r_1 - (x_M - x)/r_M & (y_1 - y)/r_1 - (y_M - y)/r_M \end{bmatrix} \tag{2}$$

Spherical-Interpolation Method

The Equation-Error Formulation

We first map the spatial origin to an arbitrary sensor j, this gives:

$$\underline{\mathrm{x}}_{\mathrm{j}} \triangleq \underline{0} \Longrightarrow \begin{cases} R_j & = & 0 \\ D_j & = & R_s \end{cases}$$

From the Pythagorean theorem, we have:

$$(R_s + d_{ij})^2 = R_i^2 - 2\underline{\mathrm{x}}_{\mathrm{i}}^{\mathrm{T}}\underline{\mathrm{x}}_{\mathrm{s}} + R_s^2$$

which is also:

$$0 = R_i^2 - d_{ij}^2 - 2R_s d_{ij} - 2\underline{\mathrm{x}}_{\mathrm{i}}^{\mathrm{T}}\underline{\mathrm{x}}_{\mathrm{s}}$$

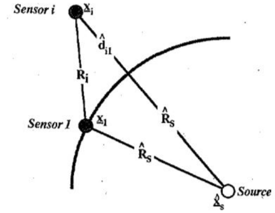


Fig. 4. Geometric representation of the relationship given in (5).

The Equation-Error Fomulation

If we take the first sensor as origin, i.e. $j = 1$

As the delays are typically not measured precisely, we introduce ”equation error”

$$\epsilon_i = R_i^2 - d_{ij}^2 - 2R_s d_{ij} - 2\underline{\mathsf{x}}_\mathsf{i}^\mathsf{T}\underline{\mathsf{x}}_\mathsf{s} \quad (i = 2, 3, \ldots, N)$$

where ϵ_i is to be minimized. With N-1 measurements, this equation can be written in matrix notaion:

$$\underline{\boldsymbol{\epsilon}} = \underline{\boldsymbol{\sigma}} - 2R_s\underline{\mathsf{d}} - 2\mathsf{S} \cdot \underline{\mathsf{x}}_\mathsf{s}$$

where

$$\underline{\boldsymbol{\sigma}} \triangleq \begin{bmatrix} R_2^2 - d_{21}^2 \\ R_3^2 - d_{31}^2 \\ \vdots \\ R_N^2 - d_{N1}^2 \end{bmatrix} \qquad \underline{\mathsf{d}} \triangleq \begin{bmatrix} d_{21} \\ d_{31} \\ \vdots \\ d_{N1} \end{bmatrix} \qquad \mathsf{S} \triangleq \begin{bmatrix} x_2 & y_2 \\ x_3 & y_3 \\ \vdots & \vdots \\ x_N & y_N \end{bmatrix}$$

The Spherical-Interpolation Method

The formal least-squares solution for $\underline{\mathbf{x}}_\mathbf{s}$ given R_s is

$$\underline{\mathbf{x}}_\mathbf{s} = \frac{1}{2}\mathbf{S}_\mathbf{W}^*(\underline{\boldsymbol{\sigma}} - 2R_s\underline{\mathbf{d}})$$

where

$$\mathbf{S}_\mathbf{W}^* \triangleq (\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T$$

The SI method is to minimize the equation error agian with respect to R_s. i.e. rewriting the equation error to eliminate $\underline{x}_s$ by substituting it with R_s, yielding a new equation error $\underline{\epsilon}'$ which is linear in R_s:

$$\begin{aligned} \underline{\boldsymbol{\epsilon}}' &= \underline{\boldsymbol{\sigma}} - 2R_s\underline{\mathbf{d}} - \mathbf{S}\mathbf{S}_\mathbf{W}^*(\underline{\boldsymbol{\epsilon}} - 2R_s\underline{\mathbf{d}}) \\ &= (\mathbf{I} - \mathbf{S}\mathbf{S}_\mathbf{W}^*)(\underline{\boldsymbol{\epsilon}} - 2R_s\underline{\mathbf{d}}) \end{aligned}$$

We notice that the formal least-squares estimate of $\underline{\mathbf{x}}_\mathbf{s}$ given R_s is itself LINEAR in R_s. When the minimizing R_s value is found in this new equation, the corresponding value of $\underline{\mathbf{x}}_\mathbf{s}$ is automatically a minimizer of the squared equation-error norm.

The Spherical-Interpolation Method

The solution is given by

$$\hat{R}_s = \frac{\underline{d}^T P_s^\perp V P_s^\perp \underline{\boldsymbol{\sigma}}}{2\underline{d}^T P_s^\perp V P_s^\perp \underline{d}}$$

where $P_s^\perp$ is defined as

$$P_s^\perp \triangleq I - SS_W^*$$

Substituting this solution into

$$\underline{x}_s = \frac{1}{2} S_W^* (\underline{\boldsymbol{\sigma}} - 2R_s \underline{d})$$

yields the source location estimate

$$\hat{\underline{x}}_s = \frac{1}{2}(S^T W S)^{-1} S^T W \left(I - \frac{\underline{d}\,\underline{d}^T P_s^\perp V P_s^\perp}{\underline{d}^T P_s^\perp V P_s^\perp \underline{d}}\right)\underline{\boldsymbol{\sigma}}$$

The Spherical-Interpolation Method

When $\mathsf{W} = \mathsf{V}$, this estimator is the minimizer of the weighted norm of the projected equation error

$$Z_{\underline{x}_s} = \underline{\boldsymbol{\epsilon}}^T P_{\underline{d}}^{\perp} W P_{\underline{d}}^{\perp} \underline{\boldsymbol{\epsilon}}$$

Minimizing $Z_{\underline{x}_s}$, one gets a simplified expression of the estimator

$$\hat{\underline{x}}_s = \frac{1}{2}(S^T P_{\underline{d}}^{\perp} W P_{\underline{d}}^{\perp} S)^{-1} S^T P_{\underline{d}}^{\perp} \underline{\boldsymbol{\sigma}}$$

By now, the Maximum Likelihood Estimation of the $\underline{x}_s$ is obtained.

IN SIMULATION:

The weighting matrices W and V are both set to Q^{-1}

Cramér-Rao Lower Bound

Cramér-Rao Lower Bound

In the simplest form, the CRLB states that the variance of any unbiased estimator is no smaller than the inverse of the Fisher information matrix.

$$\mathrm{var}(\hat{\theta}) \geqslant \mathrm{J}(\theta)^{-1}$$

It is derived from the APPENDIX that the CRLB of the localization problem is given by

$$\mathbf{\Phi^0} = c^2(\mathbf{G}_t^{0T}\mathbf{Q}^{-1}\mathbf{G}_l^0)^{-1}$$

Also, it is proved that the proposed method with arbitrary sensor array can achieve CRLB, therefore, one can use the following method to compute the lower bound of covariance matrix

$$\begin{aligned}\mathbf{\Phi^0} &= \mathrm{cov}(\mathbf{z}_p) = \frac{1}{4}\mathbf{B}''^{-1}\mathrm{cov}(\mathbf{z}_a')\mathbf{B}''^{-1} \\ &= c^2(\mathbf{B}''\mathbf{G}_a'^T\mathbf{B}'^{-1}\mathbf{G}_a^{0T}\mathbf{B}^{-1}\mathbf{Q}^{-1}\mathbf{B}^{-1}\mathbf{G}_a^0\mathbf{B}'^{-1}\mathbf{G}_a'\mathbf{B}'')^{-1}\end{aligned}$$

which is the covariance matrix of $\mathbf{z}_p$ when errors are ignored.

PREDEFINED MATRICES ARE GIVEN IN THE NEXT SLIDE

Cramér-Rao Lower Bound

$$\boldsymbol{\Phi}^{\mathbf{0}} = c^2(\mathbf{B}^{''}\mathbf{G}_a^{'T}\mathbf{B}^{'-1}\mathbf{G}_a^{0T}\mathbf{B}^{-1}\mathbf{Q}^{-1}\mathbf{B}^{-1}\mathbf{G}_a^{0}\mathbf{B}^{'-1}\mathbf{G}_a^{'}\mathbf{B}^{''})^{-1}$$

where

$$\mathbf{G}_a^0 = -\begin{bmatrix} x_2 & y_2 & r_2 - r_1 \\ x_3 & y_3 & r_3 - r_1 \\ \vdots & \vdots & \vdots \\ x_M & y_M & r_M - r_1 \end{bmatrix} \qquad \mathbf{B}^{''} = \begin{bmatrix} (x^0 - x_1) & 0 \\ 0 & (y^0 - y_1) \end{bmatrix}$$

$$\mathbf{B}^{'} = \mathrm{diag}\{(x^0 - x_1), (y^0 - y_1), r_1^0\} \qquad \mathbf{G}_a^{'} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}^{'}$$

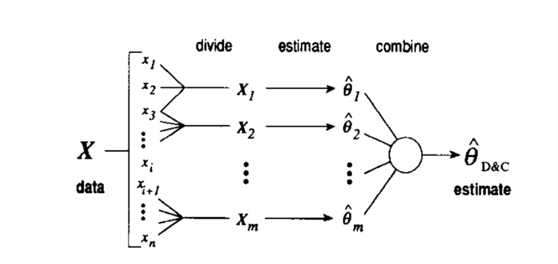
$$\mathbf{B} = \mathrm{diag}\{r_2^0, r_3^0, \cdots r_M^0\}$$

The covariance matrix of position estimate contains the uncertainty information in localization. In particular, the position mean-square error (MSE) is equal to the trace of $\boldsymbol{\Phi}$.

Divide and Conquer

Divide and Conquer

- The divide and conquer estimate is formed by combining maximum likelihood parameter estimates based on subsections of the data vector.



- The observation X is partitioned into m possibly overlapping subvectors X_i.
- Each subvector is used to estimate, via maximum likelihood, parameters $\boldsymbol{\theta}_i$.

Divide and Conquer (cont'd)

- The parameter $\boldsymbol{\theta}_{\mathsf{i}}$ is given by:

$$\boldsymbol{\theta}_{\mathsf{i}} = \mathsf{S}_i\boldsymbol{\theta}$$

  where S_i is a selection matrix (the identity matrix with appropriate rows removed).
- The estimates, denoted by $\hat{\boldsymbol{\theta}}_{DAC}$ is given by:

$$\hat{\boldsymbol{\theta}}_{DAC} = (\mathsf{S}^{\mathsf{T}}\mathsf{W}\mathsf{S})^{-1}\mathsf{S}^{\mathsf{T}}\mathsf{W}\hat{\boldsymbol{\upsilon}}$$

  where $\hat{\boldsymbol{\upsilon}}$ and S are concatenations of the $\hat{\boldsymbol{\theta}}_i$ and the S_i,

$$\hat{\boldsymbol{\upsilon}}_i = \begin{bmatrix} \hat{\boldsymbol{\theta}}_1 \\ \hat{\boldsymbol{\theta}}_2 \\ \vdots \\ \hat{\boldsymbol{\theta}}_m \end{bmatrix}, \quad \mathsf{s} = \begin{bmatrix} \mathsf{S}_1 \\ \mathsf{S}_2 \\ \vdots \\ \mathsf{S}_m \end{bmatrix}$$

  and W is a weighting matrix chosen to minimize MSE.

Divide and Conquer (cont'd)

- In our estimation, the maximum likelihood estimator of z_a is given by

$$\mathsf{z}_a = arg[min\{(\mathsf{h} - \mathsf{G_a z_a})^{\mathsf{T}} \boldsymbol{\Psi}^{-1} (\mathsf{h} - \mathsf{G_a z_a})\}]$$

- Therefore, X corresponds to h and $\hat{\boldsymbol{\theta}}$ corresponds to $\mathsf{z_a}$.

- Dividing h into subsections and performing ML estimation each yields $\hat{\boldsymbol{\theta}}_i$

- Combine $\hat{\boldsymbol{\theta}}_i$ to get $\hat{\boldsymbol{\theta}}_{DAC}$

Simulation Results

Simulation Results

TABLE I

Comparison of MSE for the SI, Taylor-Series and Proposed Methods; Arbitrary Array and Near Source

MSE	M=3	M=4	M=5	M=6	M=7	M=8	M=9	M=10
A	no.sol.	1.5741	0.1585	0.1487	0.1241	0.1165	0.1138	0.1106
B	2.1646	0.7095	0.1463	0.1338	0.1153	0.1059	0.1031	0.0947
C	2.1701	0.7003	0.1451	0.1411	0.1155	0.1105	0.1050	0.0981
D	2.1546	0.6854	0.1492	0.1361	0.1150	0.1077	0.1037	0.0961
E	1.9766	0.6873	0.1448	0.1332	0.1141	0.1052	0.1030	0.0941

A: SI method. B: Taylor series method. C: proposed method, { (14b), (22b), (24)}. D: proposed method, {(14b),(14a), (22a), (24)}.

E: theoretical MSE of the new method = CRLB.

- SI performs worst and our solution method gives a slightly smaller MSE than the Taylor-series method.
- The proposed method with the simplified formulae still performs better than the SI method.

Simulation Results

TABLE II

COMPARISON OF MSE FOR THE PROPOSED AND TAYLOR-SERIES METHODS; LINEAR ARRAY AND NEAR SOURCE

MSE	M=3	M=4	M=5	M=6	M=7	M=8	M=9	M=10
B	no.sol.	1.3342	0.3794	0.1247	0.0618	0.0283	0.0175	0.0096
C	8.2421	1.1107	0.3563	0.1222	0.0619	0.0286	0.0178	0.0098
D	8.2289	1.1020	0.3566	0.1219	0.0613	0.0286	0.0174	0.0096
E	7.2718	1.0984	0.3543	0.1217	0.0611	0.0283	0.0174	0.0095

B: Taylor series method. C: proposed method, { (28) with $\Psi = Q$, (29)}. D: proposed method, {(28) with $\Psi = Q$, (28),(29)}.

E: theoretical MSE of the new method = CRLB.

CONCLUSIONS

- The localization MSE decreases as the number of sensor increases.
- The Taylor-series method gives almost identical results as the new method.

Simulation Results

TABLE III

Comparison of MSE for the SI, Taylor-Series and Proposed Methods;

Arbitrary Array and Distant Source

MSE	M=4	M=5	M=6	M=7	M=8	M=9	M=10
A	14796	212.10	48.16	40.70	42.02	39.99	37.57
B	346.52	147.50	44.98	38.77	39.40	37.56	33.39
C	450.65	143.92	44.74	38.69	38.53	36.85	34.14
E	328.36	143.73	44.00	38.48	38.47	36.41	33.68

A: SI method. B: Taylor series method. C: proposed method, { (14b), (22b), (24)}

E: theoretical MSE of the new method = CRLB.

Conclusions

- The proposed method performs much better than SI and slightly better than Taylor-series method.
- The proposed method performs significantly better when M is small.

Simulation Results

TABLE IV

Comparison of MSE for the Proposed and Taylor-Series Methods;

Linear Array and Distant Source

MSE	M=4	M=5	M=6	M=7	M=8	M=9	M=10
B	1802.42	435.16	155.40	68.87	34.52	18.77	10.83
C	1583.04	406.42	153.59	68.45	34.30	18.81	10.87
E	1435.26	407.60	153.83	67.96	34.20	18.54	10.87

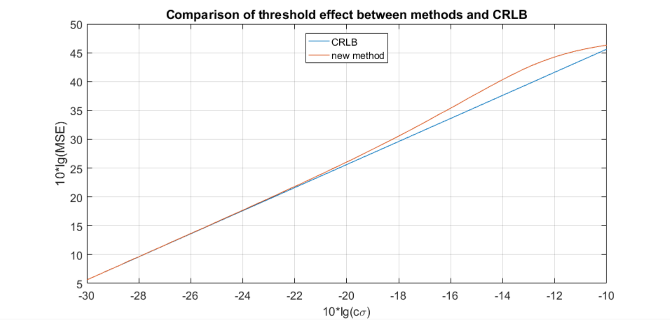
B: Taylor series method. C: proposed method, { (14b), (22b), (24)}.

E: theoretical MSE of the new method = CRLB.

Conclusions

- The proposed method performs better than Taylor-series method especially when M is small.

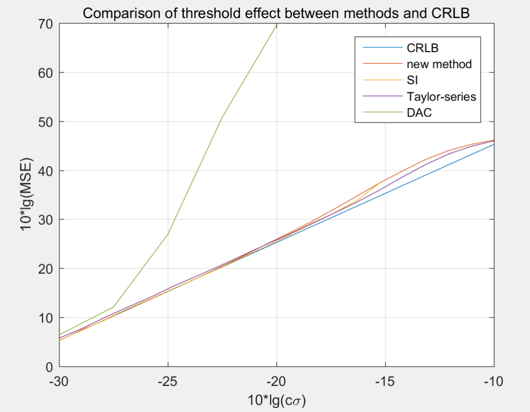
Simulation Results



Conclusions

- The proposed method performs only a little worse than the CRLB.
- The error thresholding effect doesn't occur until $\sigma_d^2 = 0.0001$

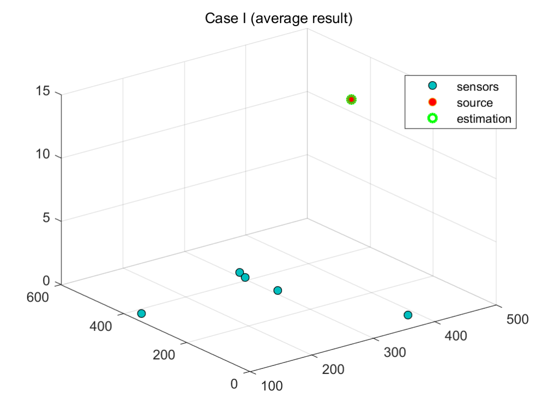
Comparison between methods



Conclusions

- DAC has the worst performance.
- Taylor-series, SI and proposed method has similar performance.

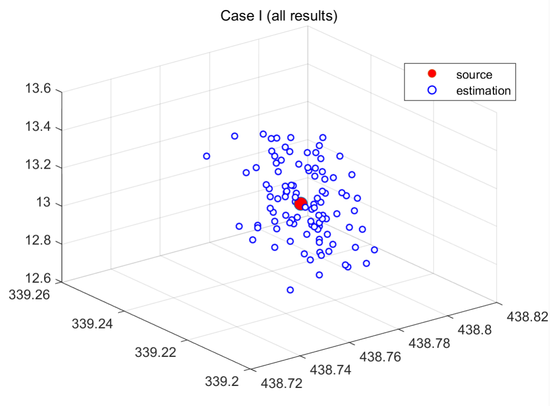
3D Simulation: Case I



CASE I: 3D MAP

- Test Space: $500m \times 500m \times 120m$
- All sensors are randomly placed on ground.
- 10000 independent tests were run, result averaged.

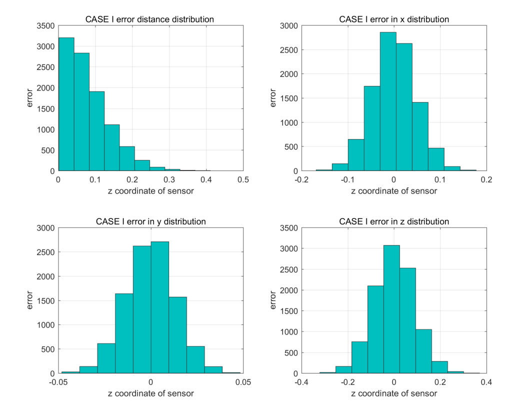
3D Simulation: Case I



CASE I: INDEPENDENT RESULTS

- Each result of independent tests is displayed (simplified for visual convenience).
- The error in every direction is no larger than 0.5.

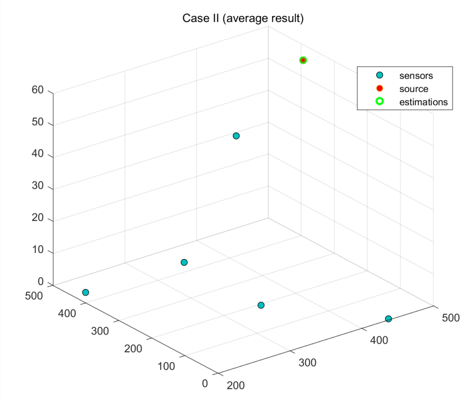
3D Simulation: Case I



CASE I: ERROR DISTANCE, ERROR IN X, Y, Z DIRECTION

- The distance between estimated position and real position is no larger than 0.4m, and mostly concentrated between [0, 0.1].
- The error in x, y, z direction is roughly normally distributed.
- The error in Case I concentrates in a very small region, thus verifies the

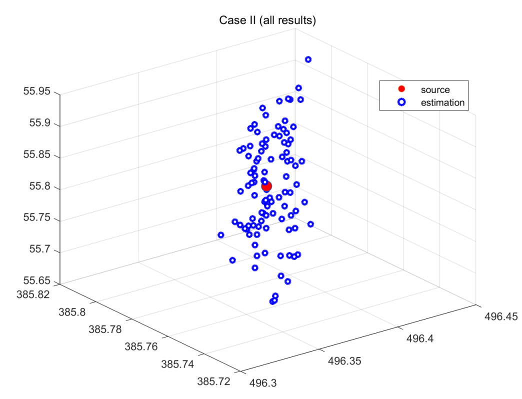
3D Simulation: Case II



CASE II: 3D MAP

- Test Space: $500m \times 500m \times 120m$
- N-1 sensors are randomly placed on ground, 1 sensor at (0,0,40).

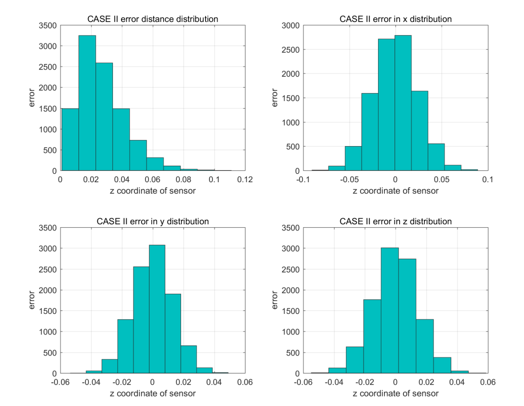
3D Simulation: Case II



CASE II: INDEPENDENT RESULTS

- Each result of independent tests is displayed (simplified for visual convenience).
- The error in every direction is no larger than 0.5m.

3D Simulation: Case II



CASE II: ERROR DISTANCE, ERROR IN X, Y, Z DIRECTIONS

- The distance is no larger than 0.1m, performance is even better than coplanar case.
- Small error region verifies the algorithm in arbitrary case.

Discussion: Distribution of Sensors

Question: How does the distribution of sensors affect the results?

AVAILABLE CHOICES:

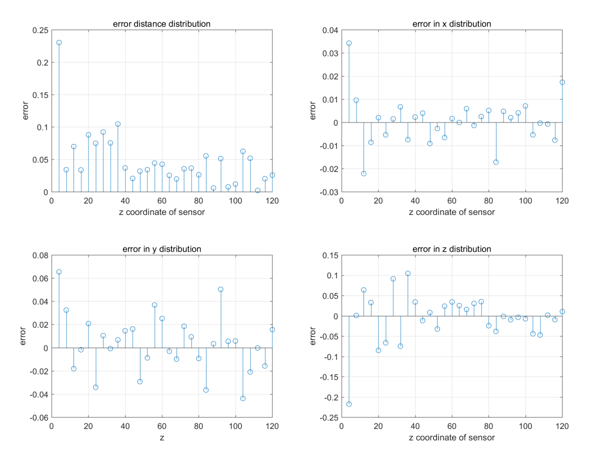
- sensors wrap up the source
- sensors distributes on one side of the source

EXPERIMENT CONDITIONS:

- 1st sensor placed at: $(0, 0, z), z \in \{4, 8, 12, \ldots, 120\}$.
- Other sensors stay on the ground.
- Source is placed at $(x, y, 60)$.
- Record the error distance, and error in x, y, z directions of each z.

When $z < 60$, sensors distributes on one side of source;
When $z > 60$, sensors wrap up the source.

Discussion: Distribution of Sensors



Simulation Result of different sensor distribution

Discussion: Distribution of Sensors

Conclusion

- When $z < 60$, the error distance is larger than case of $z > 60$
- The difference in y directions is not obvious
- When $z > 60$, the error in x, y direction gets smaller as z grows.
- Error in ”wrapping up” case is smaller than ”one side” case.